UNIVERSIDAD DE GRANADA

Escuela Técnica Superior de Ingenierías
Informática y de Telecomunicación

Departamento de Ciencias de la Computación
Inteligencia Artificial

Modelos de Programación Genética Gramatical para Aprendizaje con Múltiples Instancias

Memoria de Tesis presentada por

Amelia Zafra Gómez

como requisito para optar al grado

de Doctor en Informática

Granada — Julio de 2009

UNIVERSIDAD DE GRANADA

Departamento de Ciencias de la Computación

Inteligencia Artificial

Modelos de Programación Genética Gramatical para Aprendizaje con Múltiples Instancias

Memoria de Tesis presentada por

Amelia Zafra Gómez

como requisito para optar al grado

de Doctor en Informática

Director

Dr. Sebastián Ventura Soto

Granada Julio de 2009

La memoria titulada "*Modelos de Programación Genética Gramatical para Aprendizaje con Múltiples Instancias*", que presenta Amelia Zafra Gómez para optar al grado de doctor, ha sido realizada dentro del programa de doctorado "*Diseño, Análisis y Aplicaciones de Sistemas Inteligentes*" del Departamento de Ciencias de la Computación e Inteligencia Artificial de la Universidad de Granada, bajo la dirección del doctor Sebastián Ventura Soto.

Granada, Julio de 2009

El Doctorando

Fdo. Amelia Zafra Gómez

El Director

Fdo. Sebastián Ventura Soto

Tesis Doctoral parcialmente subvencionada por la Comisión Interministerial de Ciencia y Tecnología (CICYT) con los proyectos **TIN2008-06681-C05-02** y **TIN2008-06681-C06-03** y por la Junta de Andalucía con fondos FEDER con el proyecto de excelencia **P08-TIC-3720**.

También ha sido subvencionada por el programa predoctoral de Formación de Personal Universitario (FPU, referencia de beca **AP2005-01746**) del Ministerio de Educación.

Aprender es como remar contra corriente:
en cuanto se deja, se retrocede.

Edward Benjamin Britten

Agradecimientos

La presente memoria es el resultado de un gran esfuerzo, dedicación y sobretodo, aprendizaje. Por lo que resulta complicado resumir en unas líneas el agradecimiento que debo a tanta gente que me ha acompañado en este recorrido.

En primer lugar, quisiera expresar mi agradecimiento a mi director, Sebastián Ventura, por el tiempo que ha dedicado a culminar este proyecto. Su capacidad de trabajo, su apoyo y su constante dedicación han supuesto una motivación para introducirme en este mundo de la investigación y un ejemplo en todo momento, como investigador y como persona.

También quisiera agradecer el apoyo de mis compañeros Eva, María y Soto, gracias por el cariño y amistad que me habéis brindado desde que os conozco. A mis compañeros del departamento con los que he compartido despacho, Pedro, Luisma, José Raúl, Carlos, Enrique y Rafael por el buen ambiente que se respiraba en el trabajo. Así, como a los profesores que hoy son compañeros y de los que guardo muy buen recuerdo, gracias por enseñarme más que números y letras.

Debo acordarme también de la inestimable ayuda de Enrique Herrera para la realización de esta Tesis, y en general, de todos los miembros del grupo de investigación de AYRNA por su colaboración y entrega, debiendo destacar especialmente la labor del director del mismo, César Hervás, por la confianza que deposita en cada uno de nosotros y que con su entusiasmo y entrega es un gran ejemplo para todos los que tenemos la suerte de trabajar con él.

No me puedo olvidar de mis padres, Julián y Mari, a los que tengo tanto que agradecer que no habría palabras suficientes para expresarlo. Gracias por vuestro amor, cariño, comprensión y apoyo desinteresado. Vuestra determinación, trabajo, entrega y humildad me han enseñado mucho. A mi hermano, Julián y mi cuñada Esther, porque siempre puedo contar con vosotros, gracias por apoyarme y estar siempre disponibles cuando se necesita cualquier cosa. A Luis, porque a tu lado todo es más sencillo, gracias por ser como eres y estar siempre a mi lado.

Y por último, quisiera agradecer a todos mis amigos su amistad porque gracias a ellos sé lo que significa esta valiosa palabra, gracias por aconsejarme, ayudarme, compartir risas y llantos en todo este tiempo.

Sinceramente gracias.

Indice de Contenidos

Indice de Figuras

Indice de Tablas

Indice de Acrónimos

Acrónimo	Descripción
APR	Rectángulo de Ejes Paralelos (*Axis Parallel Rectangle*)
BNF	Forma Normal de Backus (*Backus-Naur Form*)
DM	Minería de Datos (*Data Mining*)
EA	Algoritmo Evolutivo (*Evolutionary Algorithm*)
EC	Computación Evolutiva (*Evolutionary Computation*)
EP	Programación Evolutiva (*Evolutionary Programming*)
ES	Estrategia Evolutiva (*Evolutionary Strategy*)
GA	Algoritmo Genético (*Genetic Algorithm*)
GLS	Búsqueda Local Genética (*Genetic Local Search*)
GP	Programación Genética (*Genetic Programming*)
G3P	Programación Genética Guiada por Gramática (*Grammar Guided Genetic Programming*)
ILP	Programación Lógica Inductiva (*Inductive Logic Programming*)
KNN	Vecino más cercano (*K Nearest Neightbour*)
MI	Múltiples Instancias, Multi-Instancia (*Multiple Instance, Multi-instance*)
MIL	Aprendizaje con Múltiples Instancias (*Multiple Instance Learning*)
ML	Aprendizaje Automático (*Machine Learning*)
MOEAs	Algoritmos Evolutivos Multi-objetivo (*Multi-objective Evolutinary Algorithms*)
MOGLS	Búsqueda Local Genética Multi-objetivo (*Multi-objective genetic local search*)
MOP	Problemas de Optimización Multi-objetivo (*Multi-objective Optimization Problems*)
NFL	Teorema de *No free lunch*

Acrónimo	Descripción
NSGA2	Algoritmo Genético de Ordenación de Frentes No Dominados (*Non-dominated Sorting Genetic Algorithm*)
POF	Frente de Pareto Óptimo (*Pareto Optimal Front*)
RBF	Función de Base Radial (*Radial Basis Function*)
RBS	Sistema Basado en Reglas (*Rule-Based Systems*)
SPEA2	Algoritmo Evolutivo de Fortaleza de Pareto (*Strength Pareto Evolutionary Algorithm*)
SVM	Máquinas de Soporte Vectorial (*Support Vector Machines*)
UM	Modelo de Usuario (*User Model*)

1 Introducción

1.1. Planteamiento

El aumento del volumen y la variedad de la información que se encuentra almacenada en bases de datos digitales y otras fuentes ha crecido espectacularmente en las últimas décadas. Estamos rodeados por datos, ya sean numéricos, categóricos o de cualquier otra índole, que para ser de utilidad deben ser analizados y procesados para poder extraer conocimiento que nos informe, instruya, responda y ayude en la toma de decisiones. No hay nada de nuevo en este procedimiento. Desde sus orígenes, el ser humano ha buscado la interpretación de la información con la que contaba y se ha valido de técnicas manuales o automáticas para lograr tal fin. Extraer información y conocimiento de los datos es, por tanto, un concepto bien establecido en estudios científicos y médicos. Lo realmente novedoso es el contexto en el que nos encontramos, caracterizado por tres situaciones principalmente. La primera, un crecimiento constante de información que parece no llegar a tener fin, gran cantidad de información se recoge y acumula diariamente encontrándonos con bases de datos que cuentan con cientos de millones de documentos; en segundo lugar, un avance de la tecnología sin precedentes, se dispone de ordenadores y sensores

de bajo coste que proporcionan un número de recursos ilimitado a unos precios razonables, lo que provoca que sea una práctica muy común almacenar gran cantidad de información sin pensar en eliminar otra previamente almacenada; y por último, un aumento de las oportunidades para encontrar patrones en los datos, prácticamente cualquier área, industrial, comercial, científica e incluso personal, cuenta con gran información almacenada que necesita algún tipo de procesamiento.

En este contexto, la disciplina de minería de datos, un área que surgió para facilitar la extracción de conocimiento útil y comprensible previamente desconocido, desde grandes cantidades de datos almacenados en distintos formatos, ha encontrado una ocasión única y no es de extrañar el extraordinario progreso que se está produciendo en la actualidad en el estudio de nuevos métodos más eficientes, eficaces y comprensibles para descubrir nuevo conocimiento que se adapte a las nuevas necesidades que van surgiendo. A medida que el mundo crece en complejidad, y se posee más información, la minería de datos se convierte en una técnica esencial para el esclarecimiento de los patrones que subyacen en ella, siendo impracticable un enfoque manual al tratarse de métodos lentos, costosos y altamente subjetivos. El objetivo de la minería de datos se reduce a convertir datos en conocimiento. Este objetivo no es sólo ambicioso, sino muy amplio, por ello, son muchas las tareas que se pueden distinguir dentro de este proceso, como la clasificación, la regresión, el análisis de series temporales, el agrupamiento, la sumarización y la asociación, entre otras. Cada una de estas tareas pueden considerarse como un tipo de problema diferente a ser resuelto por un algoritmo de minería de datos. Esto significa que cada tarea tiene sus propios requisitos, y que el tipo de información obtenida con una tarea puede diferir mucho de la obtenida con otra.

Si bien la minería de datos es un área muy extensa en la que se incluyen una gran cantidad de tareas, la mayoría de ellas tienen como base la realización de un aprendizaje automático [211]. El aprendizaje automático proporciona las bases técnicas de la minería de datos y se emplea para extraer información de los datos, información que es expresada de forma comprensible y puede usarse para una variedad de procesos. Concretamente, el aprendizaje inductivo es en el que está basado la mayoría de las tareas de la minería de datos [93], tratándose a su vez del área de aprendizaje automático más extensamente estudiada. Se basa en descubrir descripciones generales que permitan capturar las características comunes de un grupo de

ejemplos, para lo cual se fundamenta en observar la similitud entre los ejemplos y la suposición de que se puede generalizar a partir de un número limitado de observaciones. El razonamiento inductivo permite obtener conclusiones generales a partir de información específica. Estas conclusiones evidentemente son conocimiento nuevo, ya que no estaba presente en los datos iniciales.

Las distintas formas de realizar razonamiento inductivo dentro de la disciplina de aprendizaje automático permiten diferenciar tres paradigmas clásicos: aprendizaje supervisado, no supervisado y con refuerzo [145].

- El aprendizaje supervisado tiene lugar cuando el conocimiento del que se parte contiene la salida esperada. De este modo, se posee información sobre cómo se está realizando el aprendizaje.
- El aprendizaje no supervisado es un paradigma de aprendizaje más complejo, que tiene lugar cuando el conocimiento del que se parte no contiene la salida esperada. En este caso, el objetivo consiste en realizar una estructuración del conocimiento lo más acertada posible.
- El aprendizaje con refuerzo es un paradigma en el que inicialmente no se dispone de la información de salida, pero conforme se van obteniendo resultados el sistema va realizando una realimentación del mundo exterior en respuesta a las acciones que se realizan.

En el mundo en el que nos encontramos, donde la sobrecarga de datos se ha convertido en una realidad en nuestras vidas y cada vez nos encontramos con información más compleja y abundante, la posibilidad de extraer información comprensible de los datos más eficientemente y eficazmente sin perder precisión cobra mayor importancia. Está demostrado que el funcionamiento de los modelos depende de la información con la que trabajen y de la representación de dicha información. En este trabajo nos centraremos en un paradigma de aprendizaje que ha aparecido recientemente conocido como aprendizaje con multi-instancias o múltiples instancias (*Multiple Instance Learning*, MIL), que ha adquirido cierta importancia en la comunidad de aprendizaje automático en los últimos años y que introduce una nueva forma de representar la información basada, como su nombre indica, en conceptos representados por múltiples instancias. MIL introducido por Dietterich. et al. [57],

es un tipo especial de aprendizaje inductivo, y como tal, su finalidad consiste en aprender un concepto a partir de una serie de ejemplos de los que se dispone. La diferencia con el aprendizaje tradicional radica en que en el conjunto de entrenamiento cada patrón está compuesto por bolsas, cada una de las cuales contiene un conjunto de instancias y en el que se mantiene información relativa a las bolsas, no conociéndose la clase a la que pertenece cada instancia individual.

El estudio de este paradigma ha generado controversia con respecto a su consideración. Muchos consideran MIL una generalización del aprendizaje supervisado, mientras que otros lo consideran un nuevo paradigma de aprendizaje diferenciado de las técnicas clásicas, al presentar diferencias significativas con ellas. Ambas teorías cuentan con argumentos interesantes de sus detractores y defensores, si bien en los últimos estudios se está notando un mayor acercamiento hacia la primera opción. En esta tesis no pretendemos entrar en polémica para situarnos en un lado o en el otro, lo que sí nos interesa es determinar las principales razones que han provocado que MIL se haya convertido actualmente en un área con una gran actividad investigadora que, independientemente de donde se clasifique, ha encontrado su propia evolución paralelamente a los paradigmas de aprendizaje existentes. Entre estas razones, podemos mencionar:

- Mantiene diferencias de representación con los paradigmas de aprendizaje clásicos: aprendizaje supervisado, aprendizaje no supervisado y aprendizaje por refuerzo. En contraste con el aprendizaje supervisado, donde todas las instancias contienen etiquetas desconocidas, en MIL las etiquetas de las instancias de entrenamiento son desconocidas. En contraste con el aprendizaje no supervisado, donde todas las instancias de entrenamiento no tienen etiquetas conocidas, en MIL las etiquetas de las bolsas de entrenamiento son conocidas. Finalmente, en contraste con el aprendizaje por refuerzo, donde las etiquetas de las instancias de entrenamiento son aplazadas a los resultados obtenidos, en MIL no existe información aplazada.

- Las técnicas clásicas existentes que ignoran las características de los problemas MIL no funcionan correctamente cuando trabajan con datos multi-instancia [57]; debido a las diferencias que mantiene con el resto de aprendizajes, es necesario una adaptación de los métodos o diseño de nuevos métodos que trabajen en un escenario de múltiples instancias. Actualmente la mayoría de los paradigmas

procedentes del aprendizaje supervisado tradicional han sido extendidos a este aprendizaje como se verá en la revisión realizada en la sección 2.2.4.

- Existen numerosos problemas a los que se les puede aplicar una representación del conocimiento en forma de múltiples instancias. Este paradigma proporciona una mayor flexibilidad en la representación, permitiendo que cada ejemplo pueda estar formado por un número variable de instancias y permitir diferentes interacciones entre dichas instancias. Aunque inicialmente fue abordado para la predicción de una determinada actividad en los fármacos [57; 126; 242], actualmente una gran variedad de problemas de diferente índole, como son la categorización de textos [4], la recuperación de imágenes basadas en contenido [148], [92], [35], la recomendación de páginas web [236; 224], la recuperación semántica en vídeo [32] y la detección de conceptos en vídeo [88], han empleado este paradigma de aprendizaje para su representación, demostrando en todas ellas, mayor precisión en la resolución y mayor comprensión en la representación. De este modo, hoy en día siguen planteándose nuevos problemas que retan a este paradigma de aprendizaje.

Estas características han originado que durante los últimos años, se hayan realizado avances muy significativos en la investigación del aprendizaje con múltiples instancias, tanto desde el diseño de aplicaciones que emplean esta representación como desde el desarrollo de nuevos métodos para resolverlas. Aún así, todavía queda trabajo pendiente en esta área: no existe una revisión general de los métodos existentes, no todos los paradigmas han sido adaptados a este aprendizaje y no existen verdaderas comparaciones entre las diferentes propuestas. Por todo ello, entre las metas más destacadas de nuestro trabajo se encuentra realizar una revisión de las aplicaciones y métodos desarrollados hasta la fecha. También nos planteamos como objetivo principal introducir modelos del Aprendizaje Evolutivo en el escenario MIL, de estos métodos todavía no existe ninguna propuesta desarrollada para llevar a cabo un aprendizaje que considere la resolución de problemas con múltiples instancias. Concretamente, se llevará a cabo el diseño de modelos basados en el paradigma de Programación Genética. También se pretende definir nuevas aplicaciones que aprovechen la flexibilidad en la representación que permite este aprendizaje. Por último, se pretende elaborar un estudio comparativo entre las diferentes técnicas más significativas de este paradigma. Hasta la fecha solamente

se han comprobado algunos métodos con algunos conjuntos de datos, pero no existe un verdadero análisis que considere las propuestas más significativas con los mismo conjuntos de datos y particiones, para que la comparación sea lo más equitativa posible.

1.2. Objetivos

En este trabajo se pretende avanzar en la investigación en los métodos del paradigma de aprendizaje basado en múltiples instancias. Con este fin, realizaremos una revisión de los métodos propuestos para MIL que nos determine el estado actual en el que se encuentra el aprendizaje con múltiple instancias y a partir de esta revisión, plantearemos nuevas propuesta de MIL.

Concretamente, este trabajo presenta la incorporación de la Programación Genética a este marco de aprendizaje. En general, los Algoritmos Evolutivos constituyen una buena alternativa en los diferentes paradigmas de aprendizaje donde han sido aplicados, hecho que queda demostrado con el gran número de publicaciones aparecidas desde sus orígenes y que continúan apareciendo en la literatura. Resulta extraño que no hayan sido extendidos a este nuevo paradigma de aprendizaje, siendo métodos flexibles y eficaces que permiten generar conocimiento comprensible a través de la información de partida. Otro propósito que se desea lograr es realizar una comparativa y recapitulación de los conjuntos de datos más utilizados y ponerlos a disposición de la comunidad científica, a la vez que se realiza una unificación de las comparaciones realizadas. Existen estudios comparativos pero consideran únicamente algunos métodos o algunos conjuntos de datos. Un estudio general sería necesario para poder avanzar en esta disciplina. Finalmente, también se pretende aplicar los modelos desarrollados a dos aplicaciones concretas: la recomendación de páginas web índice y la predicción del rendimiento académico de un estudiante a partir del trabajo desarrollado en la plataforma educativa; este último problema ha sido abordado siempre desde una perspectiva de aprendizaje supervisado tradicional y como veremos, debido a las características de la representación clásica, se generan muchos valores perdidos, lo que dificulta la correcta clasificación. Se pretende buscar una representación más flexible con múltiples instancias que solucione los problemas de la representación tradicional.

Más concretamente, los objetivos científicos que se persiguen son los siguientes:

1. Revisión de los modelos propuestos para MIL. Nos proponemos llevar a cabo un repaso general a todos los modelos y paradigmas que se han propuesto para solucionar problemas basados en el aprendizaje multi-instancia, así como las aplicaciones que han utilizado esta representación.

2. Proponer un modelo evolutivo basado en programación genética guiada por gramática para el aprendizaje basado en múltiples instancias. La programación genética guiada por gramática ha demostrado su eficiencia y ventajas en la solución de problemas de aprendizaje en general y clasificación en particular, se pretende proponer un modelo basado en este paradigma en un entorno con múltiples instancias.

3. Proponer un modelo evolutivo multi-objetivo basado en programación genética guiada por gramática para el aprendizaje con múltiples instancias. Los algoritmos evolutivos multi-objetivo han demostrado su utilidad para optimizar los diferentes objetivos contradictorios que nos encontramos en la resolución de problemas de aprendizaje. Se pretende realizar un estudio de los principales enfoques multi-objetivo evolutivos más ampliamente utilizados y extenderlos a un modelo basado en programación genética guiada por gramática y que funcione en un escenario multi-instancia.

4. Analizar experimentalmente el funcionamiento de los modelos desarrollados. Se pretende realizar una comparativa de nuestras propuestas con los modelos más relevantes del resto de paradigmas que han sido desarrollados para solucionar los problemas MIL hasta la fecha. Hasta el momento las referencias comparten algunos algoritmos con algunos conjuntos de datos, pero no hay una verdadera comparativa de los principales modelos con las mismas aplicaciones. Este hecho dificulta las comparaciones en este área debido a que es difícil encontrar los datos con los que se ha experimentado y más difícil aún, por no decir imposible, encontrar las particiones concretas con las que se ha trabajado, así como las técnicas propuestas. En este trabajo se pretende hacer una revisión de las aplicaciones más utilizadas en la literatura

que estén disponibles, preparar los conjuntos de datos y sus particiones y dejarlos disponibles para que todos los modelos utilicen la misma información de partida y la comparaciones sean lo más equitativas posible.

5. Aplicación de los modelos de programación genética gramatical diseñados a un problema de recomendación de páginas web índice. Este problema, clasificado dentro de los problemas de recomendación, consiste en indicar al usuario solamente aquellas páginas que puedan incluir algún tema de su interés. Para llevar a cabo esta finalidad se requiere reconocer los gustos o preferencias de los usuarios para poder identificarlas en las páginas que se le mostrarán. Existen distintas técnicas que solucionan este problema desde la perspectiva de múltiples instancias, el problema principal es que se trata de técnicas que crecen linealmente en el tiempo de cómputo con el número de datos de los que se dispone y actúan como cajas negras que no generan conocimiento acerca de las preferencias de los usuarios. Nuestros modelos se presentan como técnicas que nos permiten generar perfiles de usuarios a través de los que podemos obtener información de éstos y que, una vez generados, permiten indicarle al usuario nuevas páginas de su interés de forma eficaz.

6. Diseño de una representación con múltiples instancias para el problema de predicción del rendimiento académico de los estudiantes. Este problema consiste en la predicción de la calificación final de los alumnos basándonos en el trabajo que han llevado a cabo en una plataforma de aprendizaje virtual. Este problema, abordado desde una perspectiva de simples instancias, contaba con una gran cantidad de ejemplos que contenían muchos valores perdidos, debido a que por un lado, cada curso tiene diferentes tipos de actividades y por otro lado dentro de un mismo curso, cada alumno puede realizar las actividades que considere más adecuadas. La representación multi-instancia nos permite una representación más flexible que representa únicamente la información disponible en cada patrón, eliminando el problema de los valores perdidos que se encuentran en el paradigma tradicional. Se compararán ambas representaciones con los modelos más representativos de cada paradigma para ver la representación que resulta más adecuada.

1.3. Estructura

Esta memoria está compuesta por seis capítulos en los que se desarrolla la investigación realizada y una sección de comentarios finales en la que se incluyen las conclusiones generales de la misma y los trabajos futuros. Los apéndices se utilizarán para ampliar información más detallada de determinadas temáticas.

En el segundo capítulo describiremos el aprendizaje automático junto con las principales clasificaciones que existen del mismo. A continuación nos centraremos en el aprendizaje con múltiples instancias donde repasaremos los distintos modelos y aplicaciones que se han propuesto en la literatura. Daremos una definición de este aprendizaje considerando tanto la definición inicial como todas las extensiones que ha ido experimentado. A continuación, estudiaremos los distintos modelos que se han propuesto en la literatura para abordar los problemas de MIL.

En el tercer capítulo revisamos algunos conceptos básicos que se emplearán a lo largo de la tesis. Se comentarán las características fundamentales de la Programación Genética y de la Programación Genética Guiada por Gramáticas, que será en la que se base nuestra propuesta. También se describirán los sistemas de aprendizaje basados en reglas que se han utilizado para resolver problemas de aprendizaje automático, repasando la estructura general de estos sistemas, las diferentes representaciones de las reglas y su proceso de evaluación considerando algunas de las métricas más relevantes. Centrándonos en la Programación Genética, se comentarán los principales progresos realizados en los últimos años. Por último, repasaremos los algoritmos evolutivos multi-objetivo, dando los principales conceptos y definiciones, las diferentes familias de este tipo de algoritmos, así como las métricas utilizadas para la comparación entre estas técnicas.

El cuarto capítulo de esta memoria estará dedicado al estudio de nuestra primera propuesta para la resolución de problemas de aprendizaje multi-instancia, un algoritmo de programación genética guiada por gramática denominado G3P-MI. Primero, se dará una introducción para especificar la motivación que nos ha llevado a desarrollar este modelo. A continuación, se describirá el modelo que se ha desarrollado comentando su estructura y funcionamiento. Para evaluar su rendimiento se comparará con las técnicas más relevantes de los diferentes paradigmas que se han desarrollado hasta la fecha, empleando tres aplicaciones ampliamente utilizadas en

la comparación de los algoritmos de MIL, que son la predicción de fármacos, la categorización de imágenes y el problema del reto de Este-Oeste, que supone un total de diez conjuntos de datos diferentes. Todos los métodos son comparados con las mismas particiones y los mismos datos para garantizar una comparación lo más equitativa posible entre los métodos.

En el quinto capítulo especificaremos los modelos multi-objetivo que se han desarrollado para solucionar los problemas con múltiples instancias. Para ello se han diseñado tres de los modelos evolutivos multi-objetivo más ampliamente utilizados, NSGA2 [51], SPEA2 [245] y MOGLS [104]; para cada uno de ellos se detallará su funcionamiento y estructura considerando su adaptación al paradigma de programación genética gramática y a un escenario MIL. Para comprobar el desempeño de las diferentes propuestas multi-objetivo se realizará un estudio que considerará las diferentes métricas para evaluar los frentes de Pareto que obtienen como resultado, así como una comparativa con el modelo anterior basado en una optimización mono-objetivo. Además, se llevará a cabo un estudio comparativo general con el resto de técnicas MIL y conjunto de datos para comprobar el rendimiento global del sistema. Finalmente, de todos los estudios obtendremos, MOG3P-MI, que será nuestra propuesta multi-objetivo para este aprendizaje.

En el capítulo sexto se aborda un problema encuadrado dentro de los problemas de personalización web, que consiste en la recomendación de páginas web índice. En este apartado se comparan nuestros modelos con las propuestas previas que se han aplicado a este problema.

En el capítulo séptimo propondremos la representación con múltiples instancias del problema de predecir el rendimiento académico de los estudiantes. Para ello, se describe una representación del problema utilizando tanto una representación basada en instancias simples como una representación basada en múltiples instancias. Ambas representaciones serán evaluadas utilizando las diferentes técnicas multi-instancia y sus contrapartidas del aprendizaje supervisado tradicional, posteriormente se llevará a cabo un estudio comparativo entre los resultados de ambas propuestas para determinar la representación más adecuada. Finalmente, se realizará una comparación entre las técnicas multi-instancia con nuestros modelos para ver su comportamiento con esta nueva aplicación.

El último capítulo, está dedicado a conclusiones y trabajo futuro. En este capítulo se resumirá los resultados obtenidos en esta memoria, se presentarán algunos comentarios sobre los mismos y se plantearán algunos de los trabajos futuros que se pueden abordar en el área.

2

Aprendizaje con Múltiples Instancias

2.1. Introducción al aprendizaje automático

El Aprendizaje Automático (*Machine Learning*, ML), comprende diferentes mecanismos, reglas, enfoques y tecnologías mediante los cuales se dota a un computador con la cualidad humana del aprendizaje. Pero ¿cómo definimos el aprendizaje?, dar una definición universalmente aceptada del concepto de aprendizaje, como le ocurre a la mayoría de los conceptos psicológicos, es una tarea compleja. Las principales definiciones del ML que nos encontramos en la literatura son:

1. Cualquier cambio en un sistema que le permite desempeñarse mejor la próxima vez, sobre la misma tarea u otra tomada de la misma población [178].

2. Un programa de ordenador *aprende* a partir de una experiencia E a realizar una tarea T (de acuerdo con una medida de rendimiento P), si su rendimiento al realizar T, medido con P, mejora gracias a la experiencia E [136].

3. Cualquier sistema que se considere *inteligente* debería poseer la habilidad de aprender, es decir mejorar automáticamente con la experiencia [164].

Tras estas definiciones, podemos considerar como característica esencial en el proceso de aprendizaje la mejora del comportamiento a partir de la experiencia. De este modo, el aprendizaje debe permitirnos incrementar o adquirir conocimiento, que nos lleve a realizar las mismas tareas con mayor eficiencia o exactitud o llevar a cabo nuevas tareas. Muchas actividades pueden ser consideradas como distintas formas de aprendizaje, desde actividades que implican procesos elementales como la memorización directa de una experiencia y el procedimiento de observar y explorar hasta actividades que requieren procesos más complejos como la generalización a partir de ejemplos e incluso el descubrimiento de nuevos conceptos. El procedimiento general del aprendizaje automático puede verse en la figura 2.1.

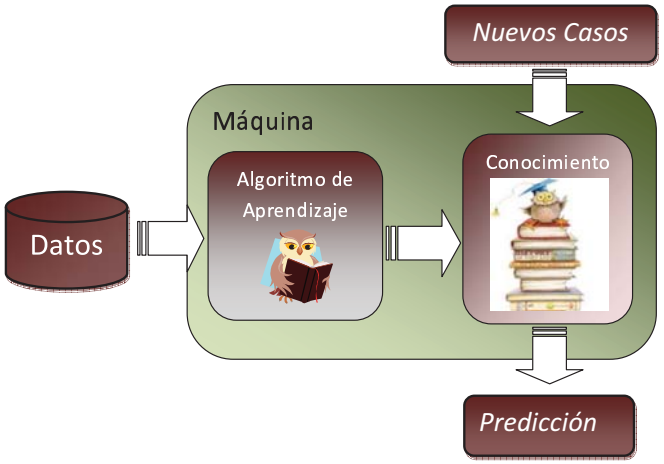


Figura 2.1: Procedimiento del aprendizaje automático

La importancia del ML y la popularidad con la que ha sido acogido se debe principalmente a los siguientes motivos:

1. La dificultad de los expertos para describir problemas complejos y diseñar manualmente un algoritmo para resolverlo. La aplicación de ML se convierte en esencial cuando el problema es demasiado complejo para diseñar y codificar manualmente un algoritmo para solucionarlo apropiadamente. Estos tipos de problemas complejos abundan en ingeniería, se trata de tareas que los seres humanos hacemos de forma natural y rápida pero no podemos describir cómo

las hacemos, como por ejemplo: reconocer imágenes, entender el lenguaje natural y escrito o tomar decisiones. Todas estas son tareas complicadas a la hora de programarse ya que requieren de inteligencia para poder ser realizadas por una máquina.

2. La necesidad de programas que continuamente se adapten a un entorno cambiante. Otra razón que hace el uso de ML necesario es la existencia de entornos dinámicos que están en continuo cambio. Independientemente de si nosotros podemos proporcionar una solución inicial del problema, el sistema puede necesitar adaptarse a situaciones cambiantes. Por ejemplo, en los sistemas de reconocimiento del habla, el programa podría adaptarse a otros lenguajes o simplemente a otros acentos diferentes del mismo lenguaje.

3. La necesidad de procesar un enorme volumen de datos. La última razón que puede llevarnos a la aplicación de ML es cuando nos enfrentamos a problemas donde el volumen de datos que necesita ser procesado para extraer conocimiento novedoso, interesante y útil hace imposible cualquier análisis manual.

Estas dos últimas características han hecho que el ML adquiera una gran importancia en la actualidad y nos lleva a la naturaleza empírica del aprendizaje. Debido a que los dominios reales suelen ser muy grandes y cambiantes, el sistema que aprende deberá, a partir de una experiencia limitada, adquirir el conocimiento que le permita generalizar correctamente cuando le sean presentados nuevas instancias del dominio. Estamos, por tanto, hablando del aprendizaje inductivo. De acuerdo al razonamiento empleado por este aprendizaje, se establece que si una hipótesis aproxima bien la función objetivo sobre un conjunto de entrenamiento suficientemente grande, también aproximará bien la función objetivo sobre los ejemplos no observados.

La última característica es la que relaciona esta disciplina con la Minería de Datos (*Data Mining*, DM), entendida esta última como el proceso de extraer conocimiento útil y comprensible, previamente desconocido, desde grandes cantidades de datos almacenados en distintos formatos [211]. Tendríamos así que tanto la minería de datos como el aprendizaje automático son técnicas relacionadas con el tratamiento de grandes cantidades de datos. De forma más concreta, se trata de crear programas

capaces de generalizar comportamientos a partir de una información no estructurada suministrada en forma de ejemplos. Los principios seguidos en el aprendizaje automático y en la minería de datos son así los mismos: la máquina aprende un modelo a partir de ejemplos y lo usa para resolver el problema. Podemos ver que ambas están unidas y comparten un nicho importante del trabajo que desarrollan. De este modo se puede determinar que las técnicas de ML y DM son un modo de hacer tareas similares [211]. No obstante, si decimos que son dos modos de hacer la misma cosa, no es del todo cierto, es verdad que engloban conceptos similares, pero mantienen algunas diferencias aceptadas por ambas disciplinas:

- En DM no sólo es importante el rendimiento obtenido sino que se requiere una representación explícita del conocimiento adquirido de manera que las decisiones puedan ser explicadas. Cada vez más, esta particularidad tiende a ser una generalización para todas las técnicas debido a que hoy en día la comprensibilidad e interpretabilidad de los modelos es tan importante como la precisión que alcancen.
- En DM la elaboración de la entrada del proceso y el análisis de la salida suele requerir una participación humana considerable. En ML estas responsabilidades suelen ser asignadas a otras componentes del sistema.
- DM incluye técnicas originadas en la modelización estadística que no son propias del ML.

Aunque el ML tiene sus orígenes y principales aplicaciones en la Inteligencia Artificial, se trata de un campo inherentemente multidisciplinar que está estrechamente relacionado con otros campos como la probabilidad, la estadística, la teoría de la complejidad computacional, la teoría de control, la teoría de la información, la filosofía, la psicología y la neurobiología. De esta manera, el aprendizaje automático aborda las mismas preguntas de investigación en todas las áreas, pero considera diferentes aspectos del proceso de aprendizaje en cada una de ellas. La estadística se centra en la comprensión de los fenómenos que han generado los datos con el objetivo de probar diferentes hipótesis sobre esos fenómenos. Los estudios de psicología aspiran a comprender los mecanismos subyacentes en el comportamiento considerando los diferentes tipos de aprendizaje (concepto de aprendizaje, adquisición de destrezas, estrategia de cambio, etc.). La teoría de control se centra en los

procedimientos para el control de procesos. La teoría de la información utiliza las medidas de entropía, codificación de hipótesis, el principio de la longitud de descripción mínima (*Minimum Description Length*, MDL) para obtener conocimiento; la teoría computacional evalúa los límites teóricos de la complejidad de las tareas de aprendizaje y la neurología inspira las redes neuronales artificiales.

Para ilustrar la diferencia,consideremos el problema de reconocimiento del habla: el aprendizaje automático se centraría en la construcción de un sistema preciso y eficiente de reconocimiento de voz y un estadístico podría colaborar con un psicólogo para poner a prueba hipótesis sobre los mecanismos que subyacen en el reconocimiento de la voz. Finalmente, un enfoque de minería de datos podría buscar patrones de expresión de datos que podrían aplicarse a determinados oradores en función de la edad, sexo o nivel de la educación.

2.1.1. Etapas en el diseño de sistemas de aprendizaje automático

El desarrollo de un sistema de aprendizaje tiene que pasar por una serie de etapas que se deben especificar con cuidado si queremos que el sistema funcione de manera correcta. Recordemos que la clave de los sistemas de aprendizaje es, que a partir de una experiencia limitada, el sistema sea capaz de aprender de ella y realizar la misma tarea de forma más eficiente o ser capaz de realizar nuevas tareas. De acuerdo a esta definición, todo sistema deberá basarse en los siguientes cinco pasos:

1. *Elegir la experiencia que usaremos para entrenar el sistema de aprendizaje.* Se trata de un factor clave en el proceso de aprendizaje inductivo, ya que a partir de dicha información se va a realizar el proceso de generalización. Si la información de partida no es consistente, está incompleta o contiene ruido, será muy normal que no obtengamos unos resultados aceptables.

   De este modo, habrá que garantizar que la información sea representativa del problema, siendo conscientes de que normalmente los datos que consideramos en el entrenamiento pueden tener un número excesivo de atributos (alta dimensionalidad), ruido en las clases de los ejemplos (ejemplos mal etiquetados), o en los atributos (atributos con ruido en sus valores), atributos irrelevantes, redundantes o con valores perdidos para determinados ejemplos.

2. *Definir el tipo de función objetivo que pretendemos aprender.* Se debe dejar bien definido qué queremos aprender exactamente, debemos aprender solamente lo necesario, cuanto más pretencioso y ambiguo sea el objetivo, más complicado será el proceso.

   La función objetivo que queremos aprender se diseña para alcanzar la finalidad especificada y el proceso de aprendizaje aproximará este objetivo mediante una función del mismo tipo que denominaremos hipótesis. Existen diferentes tipos de funciones dependiendo de la clase de problema que estemos tratando: clasificación, regresión, agrupación o cualquier otro problema.

3. *Determinar la representación que utilizaremos.* Al fijar la representación, estamos definiendo un espacio de búsqueda para la hipótesis considerada con la que pretendemos aproximar la función objetivo. A este espacio se le denomina espacio de hipótesis. Es importante que el espacio de hipótesis contenga a la función objetivo. Si no es así, nuestra hipótesis está condenada a equivocarse en algunos ejemplos.

   Cuanto más complejo sea el espacio de hipótesis más difícil será aprender la hipótesis y más ejemplos de entrenamiento serán necesarios, pero más capacidad tendrá para aproximar la función objetivo. En muchas ocasiones, cuando no se tiene muy claro qué tipo de representación es más adecuada, se prueban varias y se evalúa la calidad de las soluciones alcanzadas.

4. *Seleccionar el algoritmo que se va a utilizar y aplicarlo para aprender la función objetivo.* Para cada tipo de función objetivo, espacio de hipótesis o tipo de conocimiento tendremos distintos algoritmos que pueden aplicarse. Es posible utilizar varios algoritmos distintos o diferentes implementaciones y comparar posteriormente su rendimiento.

   El funcionamiento normal suele seleccionar el algoritmo o la implementación en función de la plataforma o de otras herramientas con las que deba utilizarse. Es habitual ejecutar el algoritmo de ML con distintos parámetros para evaluar y comparar su rendimiento

5. *Evaluar el sistema.* Es necesario evaluar el sistema para determinar si funciona bien en la práctica y generaliza en situaciones no vistas durante el entrenamiento. Un sistema que solamente funcione con ejemplos utilizados como

experiencia en la fase de entrenamiento del proceso de aprendizaje y que no sea capaz de abordar nuevos ejemplos para enfrentarse a nuevas situaciones, no es un sistema válido. La evaluación se puede realizar:

- Durante el aprendizaje, para seleccionar una hipótesis adecuada.
- Al final del aprendizaje, para indicar el nivel de confianza de la hipótesis elegida.

2.1.2. Proceso de validación del aprendizaje automático

Los métodos de aprendizaje, como se ha estudiado, permiten construir modelos o hipótesis a partir de una experiencia previa o de una evidencia. De este modo, parten de una determinada información a partir de la que van a adquirir conocimiento para realizar nuevas tareas o realizar las mismas tareas pero de un modo más eficiente.

En la mayoría de los casos es necesario evaluar la calidad de las hipótesis obtenidas de la manera más exacta posible, siendo la etapa de evaluación de modelos crucial para la aplicación real de las técnicas de aprendizaje. El proceso de evaluación debe estimar la capacidad de un clasificador para predecir nuevas instancias que le lleguen en el futuro, entendiendo este proceso como *validación del modelo*.

Una primera aproximación para evaluar hipótesis consiste en utilizar la evidencia completa para el aprendizaje y entonces emplear la misma evidencia para calcular el error de muestra de las hipótesis. El problema fundamental en el que pueden caer estos modelos es que se ajusten demasiado a las instancias de entrenamiento, en estos casos se vuelven incapaces de generalizar, funcionando de manera errónea con nuevas instancias. Se produce lo que se conoce como un sobreajuste de los datos iniciales, *overfitting*. Dado un espacio de hipótesis H, una hipótesis $h \in H$ se dice que sobreajusta los datos de entrenamiento, si existe alguna hipótesis alternativa $h_0 \in H$, tal que h tiene un error más pequeño que h_0 sobre los ejemplos de entrenamiento, pero h_0 tiene un error más pequeño que h sobre la distribución completa de instancias. Por otra parte, si se intenta corregir sin ninguna referencia exterior (por ejemplo, podando demasiado un árbol de decisión) existe el problema de subajuste (*underfitting*), en este caso las hipótesis generalizan demasiado dejando gran cantidad de datos como excepciones. Dado un espacho de hipótesis

H, una hipótesis $h \in H$ se dice que subajusta los datos de entrenamiento, si existe alguna hipótesis alternativa $h_0 \in H$, tal que h tiene un error más pequeño que h_0 sobre los ejemplos de entrenamiento y h_0 tiene un error más pequeño que h sobre la distribución completa de instancias.

Un mecanismo más conveniente para realizar el proceso de validación que nos permita calcular el error de una muestra de una hipótesis con respecto a una evidencia, consiste en separar el conjunto de datos que forman la evidencia en dos subconjuntos disjuntos como se muestra en la figura 2.2. El primer subconjunto, denominado de entrenamiento (*training*), se utiliza para el aprendizaje de las hipótesis. El segundo subconjunto, denominado conjunto de prueba (*test*), sólo se emplea, en este caso, para calcular el error de muestra de las hipótesis construidas con los datos de entrenamiento.

El hecho de utilizar dos conjuntos de datos independientes, uno para aprender la hipótesis y el otro para evaluarla, permite resolver en cierta medida el problema del sobreajuste en la evaluación de hipótesis. Sin embargo, existen todavía algunos problemas, ya que el resultado es demasiado dependiente del modo en el que se realice la partición de estos dos subconjuntos. Dado que, normalmente, esta partición se realiza de manera aleatoria, puede ocurrir que dos experimentos realizados sobre la misma evidencia y con el mismo método de aprendizaje, obtengan resultados muy dispares. Otro problema es que muchas veces tenemos pocos datos y reservar parte de ellos para el test puede hacer que podamos utilizar muy pocos datos para el entrenamiento, obteniendo peores modelos. A continuación, mostraremos diferentes técnicas de evaluación para realizar esta partición de los datos de una evidencia en dos partes.

- El método H (*hold-out*) es el método más sencillo de validación. Particiona el conjunto de casos en dos grupos: uno de entrenamiento y otro de test. El grupo de entrenamiento es usado para inducir un modelo de clasificación y el de test para estimar la tasa de error verdadera. El grupo de entrenamiento suele ser, habitualmente, dos terceras partes del conjunto total de casos, utilizando el resto para el grupo de test.

- El método de remuestreo o *random subsampling* es una generalización del método H. Este método consiste en realizar el método H múltiples veces sobre

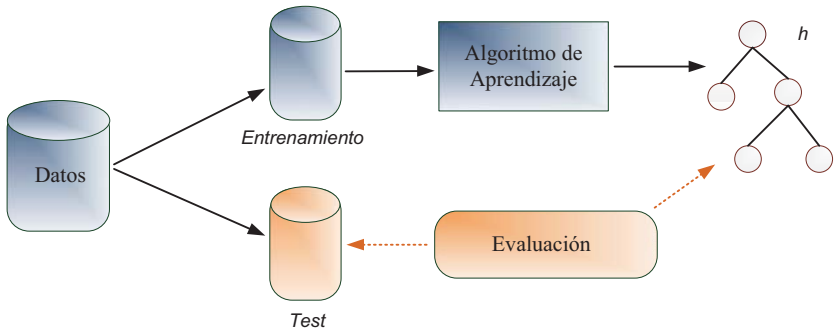


Figura 2.2: Procedimiento de los métodos de validación

diferentes particiones independientes del grupo de entrenamiento y grupo de test. Así, la estimación de la tasa de error se efectúa a partir de la media de las tasas de error obtenidas en los diferentes experimentos.

- El método de la validación cruzada o *cross-validation* [185] es también una generalización del método H, que permite reducir la dependencia del resultado. La validación cruzada consiste en dividir el conjunto de la evidencia en k subconjuntos disjuntos de tamaño similar. Entonces, se aprende una hipótesis utilizando el conjunto formado por la unión de *k-1* subconjuntos y el subconjunto restante se emplea para calcular un error de muestra parcial. Este procedimiento se repite k veces, utilizando siempre un subconjunto diferente para estimar el error de muestra parcial.

  El error de muestra final se calcula como la media aritmética de los k errores de muestra parciales. De esta manera, el resultado final recoge la media de los experimentos con k subconjuntos de prueba independientes. Una ventaja de la validación cruzada es que la varianza de los k errores de muestra parciales, permite estimar la variabilidad del método de aprendizaje con respecto a la evidencia. Considerando la importancia del parámetro k en la validación cruzada, a menudo se la suele denominar *validación cruzada de k hojas* o *k-fold cross validation*.

  Un caso particular de la validación cruzada es el método de validación de *dejar uno fuera* o *leave-one-out*, en el cual el parámetro k viene a ser igual al número de instancias que tenemos para inducir el modelo final. De esta

forma, el subconjunto de test está formado por una única instancia y el de entrenamiento por la cardinalidad del conjunto total menos esa única instancia.

- El método de *Bootstrapping* está especialmente indicado en casos en los que se tienen pocos ejemplos. La idea es, en cierto modo, similar a la validación cruzada, aunque la forma de proceder es diferente.

  Supongamos que tenemos un conjunto de ejemplos de cardinalidad N. A partir de este conjunto realizamos un muestreo aleatorio con reposición de N ejemplos. Esta muestra, que será el conjunto de entrenamiento, al ser con reemplazamiento, puede contener ejemplos repetidos. Lógicamente, esto significa que no contendrá algunos ejemplos del conjunto original. Precisamente, los ejemplos no elegidos por la muestra se reservan para el conjunto de test. Esto nos da un conjunto de entrenamiento de N ejemplos y un conjunto de test de aproximadamente $0{,}368 \times N$ ejemplos.

  El proceso anterior se repite un número prefijado k de veces y después se actúa como en el caso de la validación cruzada, promediando los errores/precisiones. Quizá lo interesante de este proceso es que las k repeticiones del proceso son independientes y esto es más robusto estadísticamente.

  Respecto a los tamaños, el valor de 0,368 se obtiene calculando la probabilidad de que un ejemplo no salga en una extracción (que es $1 - \frac{1}{n}$) y multiplicando este número por las veces que se realiza la extracción, es decir n. Más formalmente, tenemos:

$$\lim_{n\to\infty} \left(1 - \frac{1}{n}\right)^n = \frac{1}{e} \approx 0,368$$

  De hecho, esta cantidad se utiliza a veces para obtener una variante del error estimado, que se calcula de la siguiente manera:

$$Error_{estimado} = 0,368 \cdot Error_{entrenamiento} + 0,632 \cdot Error_{test}$$

2.1.3. Paradigmas de aprendizaje automático

El aprendizaje es el proceso mediante el cual un ente adquiere conocimiento. Este conocimiento puede ser suministrado por otro ente denominado profesor o puede adquirirse sin la ayuda del mismo. De esta definición encontramos una taxonomía clásica de los distintos paradigmas de ML en función del grado de realimentación, que diferencia entre aprendizaje supervisado, no supervisado y con refuerzo. Se puede realizar otra clasificación en función del razonamiento realizado encontrándonos con el aprendizaje inductivo y el deductivo. Existen muchas otros criterios para clasificar el aprendizaje como puede ser según lo que se aprende o según el problema que se resuelve, que clasifica el aprendizaje en aprendizaje basado en conceptos y basado en resolución de problemas. Nos centraremos en describir los paradigmas de aprendizaje de acuerdo a las dos primeras taxonomías del aprendizaje que hemos mencionado, que son las más ampliamente utilizadas.

2.1.3.1. Aprendizaje según el grado de realimentación

Según la información que se le proporciona al sistema en el proceso de aprendizaje, se habla de tres paradigmas de aprendizaje clásicos, el aprendizaje supervisado, no supervisado y con refuerzo. Más recientemente se está hablando también de un aprendizaje semi-supervisado que está alcanzando gran popularidad. Tenemos que tener presente que algunas taxonomías consideran el aprendizaje con refuerzo y semi-supervisado incluidos dentro del aprendizaje supervisado debido a que trabaja con algún tipo de realimentación, en contraposición al no supervisado que como veremos no utiliza ninguna información acerca de las salidas esperadas de cada ejemplo.

Esta taxonomía puede seguir incorporando nuevos grupos o subgrupos debido a nuevas tendencias en ML; precisamente, en esta tesis trataremos un nuevo subgrupo en función de la información de la que se dispone, que presenta diferencias con las variantes clásicas y que es conocido como aprendizaje con múltiples instancias y se tratará con profundidad en la sección 2.2.2.

Aprendizaje Supervisado

El aprendizaje supervisado consiste en aprender una función a partir de ejemplos etiquetados anteriormente, los cuales ayudan a establecer una correspondencia entre las entradas y las salidas deseadas del sistema. No siempre es posible hacer este tipo de entrenamiento ya que tenemos que disponer de la salida esperada. Es decir, de cada ejemplo debemos conocer el nombre del concepto al que pertenece, lo que denominaremos la salida o etiqueta de dicho ejemplo.

De manera más formal, se define como, dado un conjunto de datos $D = \{(x_1, y_1), (x_2, y_2), (x_m, y_m)\}$, donde $x_1, x_2, \ldots, x_m$ son las entradas y los $y_1, y_2, \ldots, y_m$ son las salidas, encontrar una hipótesis (función) que aproxime lo mejor posible el valor de y a partir de los valores de x.

El aprendizaje supervisado se emplea en tareas de predicción y se clasifica en función del número y del tipo de las etiquetas de los ejemplos, encontrándonos con:

- Clasificación: los ejemplos se presentan como un conjunto de pares de elementos $\delta = (x, y)$, donde $x \in X$ pertenece a las entradas e $y \in Y$ representa las salidas. El objetivo es aprender una función, denominada clasificador, que representa la correspondencia existente en los ejemplos, es decir para cada valor de entrada tenemos un único valor de salida. Además, la salida es nominal, es decir, puede tomar un conjunto de valores discretos denominados clases (cuando el número de clases es dos, tenemos lo que se denomina clasificación binaria). La función aprendida será capaz de determinar la clase para cada nuevo ejemplo, es decir dará un valor de salida para cada valor de entrada. Ejemplos de esta clasificación podrían ser clasificar un mensaje de correo electrónico como *spam* o clasificar entre varios medicamentos cuál es el mejor para una determinada patología. Existen aplicaciones donde los ejemplos pueden pertenecer a más de un clase a la misma vez, en este caso hablamos de clasificación multi-etiqueta [63; 231].

- Regresión: de nuevo los ejemplos se presentan como un conjunto de pares de elementos $\delta = (x, y)$, que representan las entradas y las salidas como se ha comentado en clasificación. El objetivo es aprender una función que represente la correspondencia existente en los ejemplos, es decir, para cada valor de entrada tenemos un único valor de salida. La diferencia respecto a la

clasificación es que la salida es un valor numérico. Ejemplos serían predecir el número de unidades defectuosas de una partida de productos o predecir la presión de una válvula a partir de las entradas.

Aprendizaje no supervisado

El aprendizaje no supervisado consiste en aprender a partir de patrones de entrada, para los cuales no se especifican los valores de sus salidas. En este caso el conjunto de observaciones no tiene clases asociadas.

Este aprendizaje se emplea en tareas de descripción y consiste en encontrar la mejor manera de estructurar los ejemplos. El proceso de aprendizaje se guía por la similitud/diferencia de los ejemplos, construyendo grupos en los que los ejemplos similares están juntos y separados de otros ejemplos menos similares.

Entre las diferentes tareas que se resuelven utilizando aprendizaje no supervisado nos encontramos con:

- Agrupamiento o *Clustering*: en este paradigma los ejemplos aparecen como conjuntos $\delta = (x)$, donde $x \in X$ pertenece a las entradas y no se posee información de la salida. El objetivo de esta tarea es obtener grupos o conjuntos entre los elementos de δ, de tal manera que los elementos asignados al mismo grupo sean *similares*. Lo importante del agrupamiento respecto a la clasificación es que son precisamente los grupos y la pertenencia a los grupos lo que se quiere determinar y, a priori, no se sabe ni cómo son los grupos ni cuántos hay. En algunos casos se puede proporcionar el número de grupos que se desea obtener. Otras veces, este número se determina por el algoritmo de agrupamiento, según las características de los datos. La función a obtener es idéntica a la de clasificación, con la diferencia de que los valores de salida y sus miembros se *crean* o *inventan* durante el proceso de aprendizaje.

- Correlaciones y factorizaciones: el objetivo consiste en ver la relevancia de los atributos, detectar atributos redundantes o dependencias entre los atributos y seleccionar un subconjunto de atributos. Los estudios correlacionales y factoriales se centran exclusivamente en los atributos numéricos (ya sean inicialmente numéricos o después de una numerización). El objetivo es ver, si

dados los ejemplos de entrada, dos o más atributos numéricos están correlacionados linealmente o de algún otro modo.

- Asociación: el objetivo, en cierto modo, es similar a la correlación, pero para los atributos nominales. La finalidad es buscar generalmente conjuntos de reglas de asociación que establezcan las relaciones entre los atributos.

- Dependencias funcionales: el descubrimiento de dependencias funcionales se puede incluir dentro de la variedad de tareas con el nombre de asociación, pero considerando todos los tipos de datos. De este modo, dado un conjunto de valores de entrada pretende determinar el valor de otra de las variables de entrada. Es decir, el valor de una entrada está en función de valores de ciertos atributos. Por ejemplo, una dependencia funcional podría ser dada la edad (discretizada en seis intervalos), el nivel de ingresos (discretizado en cinco intervalos), el código postal y si está casado o no determinar con bastante fiabilidad si el cliente tiene vehículo.

Aprendizaje semi-supervisado

El aprendizaje semi-supervisado o aprendizaje débilmente supervisado se encuadra entre el aprendizaje supervisado y el no supervisado, trabajando en situaciones donde se dispone de muy pocos ejemplos etiquetados. La idea que subyace detrás de los sistemas semi-supervisados consiste en utilizar tan poca información de entrenamiento como sea necesaria (evitando así tener que añadir demasiados ejemplos etiquetados cuando la obtención de éstos es muy costosa), pero que a la vez sea suficiente información como para mejorar los resultados de los algoritmos no supervisados que no cuentan con ninguna información.

El interés de este aprendizaje se ha incrementado en los últimos años, motivado por la existencia de una gran cantidad de aplicaciones donde resulta muy complicado asignar etiquetas a los datos de entrenamiento, por ejemplo clasificar texto, detectar fraudes o etiquetar imágenes. Podemos encontrar diferentes técnicas para su resolución como son los métodos de *co-training*, de *assemble* y *re-weighting* [29].

Aprendizaje por refuerzo

El aprendizaje por refuerzo dependiendo de la taxonomía lo podemos encontrar incluido como un tipo de aprendizaje supervisado por contener algún tipo de información en contraste con los no supervisados. La idea consiste en construir una función que tenga el comportamiento del sistema, para ello se observan los datos de entrada y se actúa en función de ellos. A continuación, un proceso de realimentación nos guía acerca de cómo se ha realizado el procedimiento. En este aprendizaje no se conoce la salida exacta para cada entrada, sólo se conoce como debería ser el comportamiento general ante diferentes entradas.

Se trata de un aprendizaje en línea que produce una señal de refuerzo indicando el éxito o fracaso de la salida del sistema. El sistema, tras mostrar la salida, obtiene una señal de refuerzo. Así, si una acción tomada por el sistema es seguida de un estado satisfactorio, dicha acción es reforzada y en caso contrario es disminuida. El aprendizaje por refuerzo es adecuado cuando no existe un conocimiento *a priori* del entorno o éste es demasiado complejo como para utilizar otros métodos.

2.1.3.2. Aprendizaje según el razonamiento

De acuerdo al razonamiento que se sigue para realizar el proceso de aprendizaje nos encontramos con el aprendizaje inductivo y el deductivo.

Aprendizaje inductivo

El objetivo consiste en construir la descripción de un concepto en la que encajen todos los ejemplos y ninguno de los contraejemplos (aprendizaje a partir de ejemplos). Las principales dificultades son que nunca sabemos si disponemos de un número suficiente de ejemplos para aprender el concepto correspondiente, y que los datos en los que tenemos que basar el aprendizaje pueden contener ruido considerando ejemplos mal clasificados, ejemplos sesgados o poco representativos e información irrelevante. Los sistemas de este tipo pueden trabajar recibiendo todos los ejemplos simultáneamente o incrementalmente. El aprendizaje inductivo no garantiza la corrección de lo aprendido, aún no habiendo problemas de ruido, generalizar a partir de ejemplos no es un método que garantice la corrección de las generalizaciones.

Así, nuevo conocimiento puede invalidar en cualquier momento lo que hemos dado por cierto mediante el proceso inductivo, debido a que no existe una teoría o procedimiento que lo fundamente de manera sólida.

Aprendizaje analítico

Este paradigma de aprendizaje parte de muy pocos ejemplos, normalmente uno, junto con una teoría del dominio. El ejemplo se utiliza para guiar las cadenas deductivas que deben seguirse para resolver nuevos problemas o para formular reglas de control de búsqueda que posibiliten una aplicación más eficiente del conocimiento del dominio. Estas técnicas no intentan ampliar lo que sabe hacer el sistema, sino mejorar su eficiencia. El ejemplo del que se parte suele ser una traza de la resolución del problema junto con anotaciones sobre la justificación de las decisiones adoptadas. Este razonamiento tiene una base teórica sólida soportada por la lógica matemática que sí preserva la verdad, en el sentido de que ningún nuevo ejemplo puede invalidar lo que deducimos.

2.2. Aprendizaje con múltiples instancias

El aprendizaje con múltiples instancias (*Multiple Instance Learning*, MIL) ha recibido una gran aceptación en la comunidad del aprendizaje automático, que ha participado de manera muy activa en sus avances científicos. Su origen se remonta al trabajo pionero realizado por Dietterich et al. [57] en 1997 quienes establecieron las bases de este aprendizaje, a la vez que propusieron los primeros modelos para abordarlo.

El gran auge que ha experimentado este nuevo paradigma de aprendizaje, que cuenta con una gran cantidad de contribuciones en poco más de un década, quizás se pueda encontrar en el tipo de aprendizaje que representa, el cual, si bien tiene su base en la misma idea en la que se basa cualquier paradigma de aprendizaje inductivo tradicional (*aprender conceptos a partir de la experiencia previa*), incorpora una mayor flexibilidad en la representación del conocimiento. De esta forma, permite que la representación se adapte a la información disponible en cada momento considerando que cada patrón de entrenamiento está formado por un número variable de instancias. Se eliminan así, las restricciones que supone la correspondencia

unívoca que existía entre patrón e instancia en el aprendizaje tradicional. Esta mayor flexibilidad se introduce a costa de una mayor ambigüedad en el conocimiento, debido a que la información de la que se dispone no es completa.

En esta sección se intentará aclarar el concepto de MIL aportando una descripción y mostrando la notación y terminología necesaria para su comprensión. Además describiremos las diferentes contribuciones que se han desarrollado hasta la fecha en este área de conocimiento.

2.2.1. Definición

Los problemas basados en múltiples instancias (*Multiple Intances*, MI) surgieron en el aprendizaje automático situados en un contexto entre el paradigma de aprendizaje supervisado y no supervisado, manteniendo diferencias con ambos. A diferencia del aprendizaje no supervisado, tenemos información sobre los patrones y a diferencia del supervisado, no se tiene información de las instancias individuales. Por tanto, la diferencia radica en que, por un lado contamos con información sobre la salida de los ejemplos, pero por otro lado esta información es incompleta, característica que permite una mayor flexibilidad en la representación.

Como paso previo a la descripción, se expondrá un problema clásico que muestra este tipo de aprendizaje. Supongamos que la cerradura de la puerta del almacén de una oficina se encuentra bloqueada. Cada empleado tiene un llavero con diferentes llaves que abren alguna puerta de dicha oficina (por ejemplo, la puerta principal, la puerta de su despacho o la puerta de la sala de conferencias). Algunos trabajadores tienen la llave que abre el almacén, mientras que otros no poseen dicha llave.

Para desbloquear la puerta del almacén, el cerrajero necesita conocer la llave de dicho cuarto. Lo que complica el trabajo del cerrajero es que los trabajadores no le facilitan la tarea. En lugar de mostrarle la llave que abre el almacén, solamente le dan el llavero completo indicándole si su llavero contiene la llave que abre la puerta de ese cuarto. Como el cerrajero no tiene acceso a la puerta del almacén no puede examinar cada una de las llaves para comprobar cuál es la que abre la puerta y no le queda más remedio que examinar la forma de todas las llaves del llavero y deducir la respuesta.

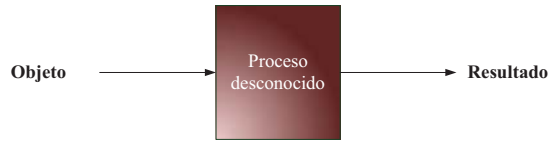
Una vez visto un ejemplo básico de un problema de aprendizaje con múltiples instancias donde se puede apreciar que se trabaja con un conocimiento parcial o incompleto de la información inicial, pasaremos a describir este aprendizaje mostrando sus diferencias más apreciables con respecto el aprendizaje supervisado tradicional.

En los problemas de aprendizaje supervisado, el sistema de aprendizaje trabaja con un conjunto de ejemplos clasificados a partir de los cuales se desea realizar un proceso de aprendizaje que permita clasificar nuevos ejemplos. El proceso de entrenamiento parte de algunos ejemplos descritos por un conjunto fijo de atributos, conocido como vector de características y una etiqueta de clase por cada ejemplo. La fase de entrenamiento consiste en inferir una relación, normalmente representada como una función, desde los atributos a las etiquetas de las clases.

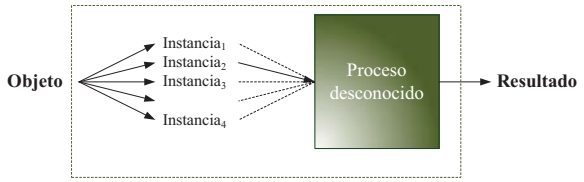
En el aprendizaje con múltiples instancias se adopta básicamente la misma configuración descrita anteriormente. Se parte de ejemplos con una serie de atributos que los caracterizan y una etiqueta de clase por cada ejemplo. La tarea sigue siendo la inferencia de la relación entre los atributos y las etiquetas de clase. La diferencia es que cada uno de los ejemplos está representado por más de un vector de características. Si consideramos un vector de características como una instancia de un ejemplo, el aprendizaje supervisado tradicional tiene una única instancia por ejemplo, mientras que el aprendizaje MI tendría varias instancias por ejemplo, de aquí el nombre de *aprendizaje con múltiples instancias*.

La diferencia puede ser representada gráficamente como se muestra en la Figura 2.3. En esta figura, el *objeto* es un ejemplo descrito por algunos atributos, el *resultado* es la etiqueta de la clase y el *proceso desconocido* es la relación que se pretende descubrir. La figura 2.3(a) representa el caso del aprendizaje supervisado tradicional donde un objeto es representado por un único vector de características, que es asociado con una etiqueta, *Resultado* y el *proceso desconocido* generan una función f aplicada sobre la instancia.

En MIL las aplicaciones de ML se vuelven más complejas, la situación se muestra en la figura 2.3(b) donde existen varias instancias en un ejemplo. De este modo el *objeto* puede estar formado por diferentes instancias, cada una de ellas representada por un vector de características distinto. Aquí, el sistema de aprendizaje en vez de conocer que cada ejemplo de entrenamiento puede ser representado como un vector de características, solamente sabe que cada ejemplo puede representarse por uno



(a) Aprendizaje con instancias simples



(b) Aprendizaje con múltiples instancias

Figura 2.3: Representación del aprendizaje con simples y múltiples instancias

de un conjunto de vectores de características. Cogiendo como símil el problema del cerrajero que se ha comentado, en lugar de conocer la llave (de cada llavero) que abre el almacén, el sistema solamente conoce si ese llavero contiene la llave para abrir la puerta, pero desconoce la llave concreta que abre dicha puerta.

Interpretamos que el *proceso desconocido* en la figura 2.3(b) es diferente del representado en la Figura 2.3(a) debido a que la entrada es diferente. En el primer caso la entrada es un vector de características y en el segundo de un conjunto de vectores de características. Debemos tener en cuenta que las flechas punteadas y sólidas que representan la entrada del proceso en 2.3(b) implican que sólo algunas de las instancias de entrada pueden representar el concepto que se quiere aprender. Por lo tanto, mientras que el *proceso desconocido* correspondiente al aprendizaje supervisado y representado en la figura 2.3(a) es simplemente un problema de clasificación, el *proceso desconocido* que se muestra en la figura 2.3(b) y que indica un aprendizaje con múltiples instancias, comúnmente es visto como un proceso de dos fases: un primer paso consiste en un problema de clasificación para cada instancia y una segunda etapa que consiste en un proceso de selección basado en el primer paso y alguna hipótesis que relacione la clasificación realizada de las instancias con la etiqueta que se asignará a la bolsa.

2.2.2. Notación. Hipótesis estándar

Una vez descrito el problema de MIL, vamos a dar una definición formal del mismo basándonos en la hipótesis estándar con la que se definió inicialmente este tipo de aprendizaje.

Definimos M como el conjunto de entrenamiento, que contiene un conjunto m_i de ejemplos. Cada ejemplo m_i tiene un resultado observado, $f(m_i)$. La función f representa el *proceso desconocido* que se ha comentado en la sección anterior. La meta de aprendizaje consiste en encontrar una buena aproximación de la función f analizando el conjunto de ejemplos de entrenamiento etiquetados del conjunto M.

En el aprendizaje supervisado tradicional, cada objeto m_i es representado por un vector de n características V_i y una etiqueta, *Resultado*, dada por la función f aplicada a dicho objeto $f(m_i)$. En este caso no hay distinción entre objetos y sus vectores de características debido a que existe una correspondencia unívoca, quedando cada ejemplo de entrenamiento etiquetado por $(V(m_i), f(m_i))$.

Por otro lado, en el aprendizaje con múltiples instancias, cada objeto m_i puede tener V_i instancias denotadas por $m_{i,1}, m_{i,2}, \ldots, m_{i,v_i}$. Cada una de estas variantes será representada por un vector de n características distinto $V(m_{i,j})$. Un ejemplo de entrenamiento es, por lo tanto, descrito como $(V(m_{i,1}), V(m_{i,2}), \ldots, V(m_{i,vi}),$ $f(m_i))$.

La meta del aprendizaje multi-instancia consiste en encontrar una buena aproximación de la función $f(m_i)$, $\overline{f}(m_i)$, analizando un conjunto de ejemplos de entrenamiento. Trabajando en un contexto de dos clases, con clases positivas y negativas, para obtener esta función Dietterich et al. definen una hipótesis que asume que si el resultado observado es positivo, al menos una de sus instancias produce un resultado positivo. Por el contrario, si el resultado observado es negativo, entonces ninguna de las instancias produce un resultado positivo. Esto puede ser modelado mediante la introducción de una segunda función $g(V(m_{i,j}))$ que tiene una única instancia y produce un resultado. El resultado observado externamente, $f(m_i)$, puede ser definido como sigue:

$$f(m_i) = \begin{cases} 1 & \text{Si } \exists\ j \mid g(V(m_{i,j})) = 1 \\ 0, & \text{de otro modo} \end{cases}$$

Esta hipótesis está basada en etiquetas de clase a nivel de instancias, por lo que el *proceso desconocido* especificado en la figura 2.3(b) constaría de dos pasos, el primer paso para proporcionar las etiquetas de cada clase a cada instancia y la hipótesis de MI sería aplicada en la segunda etapa.

Esta especificación es solamente una posibilidad para modelar problemas MI. En general, cuando se amplía el número de instancias de un ejemplo de uno a muchos, podemos encontrarnos con un gran número de posibilidades para modelar la relación entre el conjunto de instancias y su etiqueta de clase. Para las instancias dentro de un ejemplo, puede haber ambigüedad, redundancia, interacciones y muchas otras propiedades por explorar. El modelo de dos etapas y la hipótesis de MI comentadas pueden ser adecuado para explorar la ambigüedad [126], pero si estamos interesados en otras propiedades de los ejemplos, otros modelos e hipótesis pueden ser más convenientes y apropiados. En la siguiente sección trataremos con diferentes extensiones que se han elaborado durante la evolución que ha experimentado este paradigma de aprendizaje.

2.2.3. Otras hipótesis de trabajo

Si nos remontamos a los primeros años de investigación acerca del aprendizaje multi-instancia, todos los trabajos se basaban en la hipótesis estándar y el modelo en dos fases, es decir, las formulaciones MIL codificaban explícitamente o implícitamente la hipótesis de que una bolsa es positiva si y sólo si al menos una de sus instancias era positiva y negativa en caso contrario. Esta formulación es conocida en la comunidad científica como la hipótesis estándar o hipótesis de Dietterich, siendo a día de hoy la más ampliamente empleada en las diferentes aplicaciones.

Más recientemente, han aparecido otros modelos de aprendizaje multi-instancia generalizados que formalizan diferentes hipótesis para determinar la etiqueta de una bolsa a partir de las instancias que posee. Estas hipótesis requieren que se satisfagan restricciones más complejas que simplemente tener al menos una instancia positiva.

En este contexto, Weidmann et al. [205] indicaron diferentes hipótesis sobre cómo la clasificación de las instancias determina la etiqueta de la bolsa, las cuales dan lugar a diferentes tipos de problemas de predicción multi-instancia. Formalmente,

si llamamos Λ el espacio de instancias y $\Omega = \{0, 1\}$ al conjunto de etiquetas de clase. Un concepto de multi-instancia es una función $\nu_{MI} : 2^{\Lambda} \rightarrow \Omega$. En la hipótesis estándar esta función es definida como

$$\nu_{MI}(X) \Leftrightarrow \exists\ x \in X\ :\ c_I(x) \tag{2.1}$$

donde $c_I \in C$ es un concepto específico del espacio de conceptos C definido sobre Λ, y $X \subseteq \Lambda$ es un conjunto de instancias.

Basada en esta caracterización, Weidmann et al. [205] definieron tres tipos de problemas multi-instancia generalizados, cada uno de ellos emplea diferentes hipótesis de cómo la clasificación de las instancias determina la etiqueta de la bolsa. Las tres propuestas que definieron son: MI basado en presencia, MI basado en umbral, y MI basado en cuentas.

MI basado en presencia es definido en términos de la presencia de al menos una instancia de cada concepto en una bolsa (esta hipótesis es una caso especial de la hipótesis estándar, ambas coinciden si consideramos que cada bolsa considera un único concepto), su definición se puede ver en la ecuación 2.2.

$$\nu_{BP}(X) \Leftrightarrow \forall c_i \in C\ :\ \Delta(X, c_I) \geq 1 \tag{2.2}$$

MI basado en umbral requiere que un cierto número de instancias de cada concepto estén presentes en la bolsa, su definición se puede ver en la ecuación 2.3.

$$\nu_{BU}(X) \Leftrightarrow \forall c_i \in C\ :\ \Delta(X, c_I) \geq t_i \tag{2.3}$$

MI basado en cuentas requiere un número máximo, así como un número mínimo de instancias de un cierto tipo en la bolsa, su definición se muestra en la ecuación 2.4.

$$\nu_{BC}(X) \Leftrightarrow \forall c_i \in C\ :\ t_i \leq \Delta(X, c_I) \geq z_i \tag{2.4}$$

En las ecuaciones de 2.2 a 2.4, ν_{BP}, ν_{BU} y ν_{BC} son funciones definidas sobre $2^{\Lambda} \rightarrow \Omega$, $C \subset \zeta$ es un conjunto dado de conceptos, Δ es una función $\Delta : 2^{\Lambda} \times \zeta \rightarrow \mathbb{N}$ que

cuenta el número de instancias que representa un concepto determinado en una bolsa, $t_i \in \mathbb{N}$ y $z_i \in \mathbb{N}$ son respectivamente el umbral más alto y más bajo para el concepto c_i. Dos algoritmos han sido propuestos para trabajar con este modelo generalizado: TLC [205] y CCE [242], los cuales también trabajan bien para resolver problemas con la hipótesis estándar del aprendizaje con múltiples instancias.

De forma independiente, Scott et al. [169] definieron otro modelo de aprendizaje multi-instancia generalizado en el que una etiqueta de la bolsa no se basa en la proximidad de una única instancia a un único punto de destino. Específicamente, definen los conceptos por un conjunto de q puntos de atracción $C = \{c_1, \ldots, c_q\}$ y un conjunto de q' puntos de repulsión $\bar{C} = \{\bar{c_1}, \ldots, \bar{c_{q'}}\}$.

La etiqueta para un bolsa $X = \{x_1, \ldots, x_p\}$ es positiva si hay un subconjunto de r puntos $C' \subseteq C \cup \bar{C}$ tales que cada punto de atracción $c_i \in C'$ esté cerca de algún punto en X y cada punto de repulsión $\bar{c_j} \in C'$ no esté cerca de ningún punto en X.

Es decir, si definimos un atributo booleano, a_{c_i}, para cada punto de atracción $c_i \in C$ que toma un valor 1 si existe algún punto $x \in X$ cerca de él y definimos un atributo booleano, $\bar{a}_{\bar{c_i}}$, para cada punto de repulsión $\bar{c_j} \in \bar{C}$ que toma un valor 1 si no hay ningún punto X cerca de él, entonces la etiqueta de la bolsa es una función umbral, r que depende de los atributos $(q + q')$.

Diferentes algoritmos han sido especialmente diseñados para este tipo de hipótesis generalizada, tales como GMIL-1 [169], GMIL-3 [191], $k_\wedge$ [192] y k_{min} [193]. Al igual que TCL y CCE, estos algoritmos pueden también utilizarse para resolver el problema empleando la hipótesis estándar de este aprendizaje. Podemos encontrar problemas de versión generalizada en visión artificial, recuperación de imagen basada en contenido y análisis de secuencia biológica [169].

Los modelos generalizados del aprendizaje con múltiples instancias, en general resultan ser más expresivos que los que emplean el modelo estándar. Entre las dos propuestas que se han definido existen diferencias y semejanzas bajo ciertas circunstancias. De este modo, estableciendo $t_i \geq 2 (i \in \{1, \ldots, q\})$ para cada punto de atracción, entonces tanto los modelos *basados en umbral* como los *basados en cuentas* no son representaciones posibles en el modelo generalizado de Scott et al. [169]. Por otro lado, como indicaron Tao et al. [192], cuando $r < q + q_\prime$, la capacidad de

representación del modelo de Scott et al. irá más allá de la generalización cubierta por Weidmann et al. También existen coincidencias entre ambas propuestas, donde las distintas generalizaciones que se han definido se solapan. Por ejemplo, estableciendo $t_i = z_i = 1(i \in \{1, \ldots, q\})$ para cada punto de atracción y estableciendo $t_j = z_j = 0(j \in \{1, \ldots, q_r\})$ para cada punto de repulsión entonces el modelo de Weidmann et al. *basado en cuentas* es igual que el de Scott et al. cuando $r = q + q'$.

Otras extensiones que se han realizado de este aprendizaje con respecto a su definición original consisten en la variación de los valores de salida que puede tomar una bolsa. Existen trabajos que resuelven la tarea de regresión mediante este aprendizaje introduciendo que las bolsas tengan etiquetas de valores reales y denominando al problema, problema de regresión multi-instancia (MIR). Amar et al. [3] presentaron la primera propuesta para utilizar etiquetas de valores reales y diferentes estudios han continuado por esta línea, entre ellos podemos citar el trabajo de Ray y Page [158] que modifican la regresión para MIL asumiendo que si un algoritmo puede identificar la mejor instancia en cada bolsa, entonces la regresión podría ser utilizada para aprender un método de etiquetado para cada instancia y, a continuación, para cada bolsa. Ramón y Raedt [156] adaptaron el método de etiquetado utilizando redes neuronales para aprender una función que toma instancias de valores reales. Finalmente, Dooly et al. [58] presentan extensiones del algoritmo de los k-vecinos más cercanos (kNN), Citation-kNN, y el algoritmo de diversidad de la densidad para un entorno de valores reales y estudiaron su funcionamiento sobre datos booleanos y reales.

2.2.4. Trabajo previo

Desde la aparición de este paradigma de aprendizaje no han faltado la propuesta de métodos que intentan, desde diferentes enfoques, obtener una mayor eficacia y eficiencia para resolver las diferentes aplicaciones que han aparecido en esta área. En esta sección llevaremos a cabo un estudio de los distintos trabajos realizados hasta la fecha considerando tanto las técnicas desarrolladas como las aplicaciones que emplean en su representación la definición de un concepto basado en múltiples instancias.

2.2.4.1. Estudio de los métodos aplicados a MIL

En el estudio de los trabajos desarrollados, se pueden encontrar dos etapas bien diferenciadas. Inicialmente, se diseñaron algoritmos que resolvían los problemas del aprendizaje con múltiples instancias de forma explícita, es decir, se idearon para resolver específicamente estos problemas. Más recientemente, se están realizando trabajos que consideran adaptaciones de los paradigmas clásicos utilizados en el aprendizaje tradicional al escenario de MIL.

La primera solución al problema del aprendizaje con múltiples instancias fue propuesta por Dietterich et al. [57]. Los autores presentan el problema del aprendizaje multi-instancia, así como la primera aplicación que es formulada mediante este paradigma (la predicción de actividad en los fármacos). Para resolver este problema, proponen el aprendizaje mediante un rectángulo paralelo a los ejes (*Axis Parallel Rectangle*, APR), que representará el concepto aprendido, conteniendo al menos una instancia de cada bolsa positiva, y ninguna instancia de las bolsas negativas. De esta forma, una bolsa es clasificada como positiva si al menos una de sus instancias cae dentro del APR, de otro modo, la bolsa es clasificada como negativa. Un ejemplo del funcionamiento de esos algoritmos puede verse en la figura 2.4.

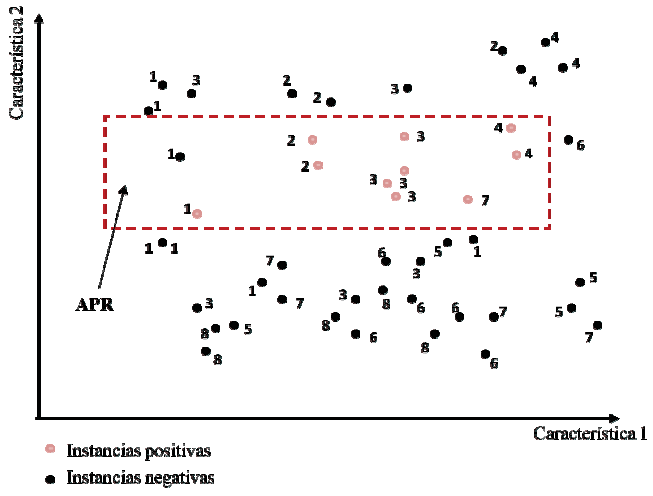


Figura 2.4: Funcionamiento de los algoritmos de aprendizaje basados en APR

Las tres versiones del algoritmo de aprendizaje de APRs propuestas son:

- Versión estándar: construye un rectángulo que contenga todas las instancias positivas de los objetos etiquetados como positivos.
- Versión de fuera hacia dentro (outside-in): parte del algoritmo estándar y, posteriormente, estrecha los límites del rectángulo para excluir a los falsos positivos.
- Versión de dentro hacia fuera (inside-out): este algoritmo parte de un punto y, posteriormente, forma el menor rectángulo que cubra al menos una instancia positiva de cada objeto positivo.

En general, los resultados muestran que los algoritmos de aprendizaje de APRs funcionan mejor que los tradicionales en este tipo de problemas.

A raíz de la idea base del trabajo de Dietterich et al. aparacieron un gran número de estudios teóricos que han contribuido en gran medida al crecimiento del aprendizaje multi-instancia. Long y Tan [123] iniciaron la investigación de la aprendibilidad-PAC de los rectángulos de ejes pararelos bajo el marco del aprendizaje multi-instancia. Llegaron a describir un algoritmo teórico de tiempo polinomial y demostraron, que si las instancias en las bolsas eran independientes, presentarían una distribución de producto, su trabajo concluyó que APR era aprendible-PAC. Auer et al. [6] demostraron que, si las instancias en las bolsas no son independientes entonces el aprendizaje APR bajo el marco de aprendizaje multi-instancia sería NP-duro. Además, también presentaron un algoritmo teórico que no requiere distribución de producto y que necesita menos tiempo y complejidad que el algoritmo de Tan y Long [123]. Más tarde, este algoritmo teórico se transformó en un algoritmo práctico llamado Multinst [5].

Además de los trabajos prácticos de Dietterich et al. [57] y Auer [5], podemos encontrar otros algoritmos planteados específicamente para MIL. Así, hay contribuciones como las de Maron y Lozano [126] que propusieron uno de los algoritmos más populares de MIL, Diverse Density (DD). El algoritmo se basa en encontrar un punto del espacio de características que esté suficientemente cercano al menos a una instancia de cada objeto positivo y significativamente lejos de todas las instancias de los objetos negativos. Para este fin, los autores definen la medida de la densidad de

la diversidad, una medida que nos determina la cercanía o lejanía de las instancias de los objetos positivos y negativos al punto estimado. La clave de este algoritmo reside en la elección del punto que maximice la densidad de la diversidad, que se obtiene adaptando un clasificador bayesiano estándar que considere bolsas con un conjunto de instancias en lugar de instancias individuales. En la figura 2.5 se encuentra la representación de un ejemplo utilizando este modelo.

DD ha sido extendido en varias ocasiones, Zhang y Goldman [233] lo combinaron con el algoritmo de Maximización de la Esperanza (*Expectation Maximization*, EM), desarrollando EM-DD, un algoritmo de propósito general cuya idea básica consiste en determinar las instancias que corresponden a la etiqueta de la bolsa como si se tratase de un atributo perdido, el cual puede ser estimado utilizando el enfoque de EM. Recientemente, otra extensión ha sido propuesta por Pao et al. [148], quienes propusieron un algoritmo de aprendizaje basado en el enfoque EM para proporcionar un procedimiento comprensible que maximizase la medida de DD sobre múltiples instancias dadas. En las propuestas que se detallan a continuación también veremos algunos algoritmos que incorporan los fundamentos de este algoritmo.

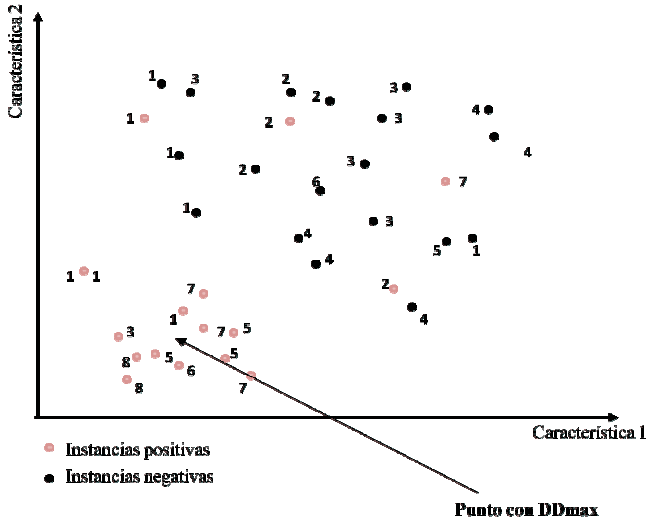


Figura 2.5: Funcionamiento de los algoritmos de aprendizaje basados en DD

Además de los trabajos relacionados con el desarrollo de algoritmos diseñados específicamente para MIL, desde su comienzo, la mayoría de la investigación se ha centrado en diseñar algoritmos de aprendizaje multi-instancia derivados de las técnicas desarrolladas en el aprendizaje supervisado tradicional. Actualmente, la mayoría de los algoritmos más populares de ML han sido considerados para resolver problemas de aprendizaje con múltiples instancias. Realizaremos un resumen de los diferentes paradigmas que han sido extendidos a MIL junto con las propuestas más relevantes que se han desarrollado.

Empezamos con los métodos de aprendizaje basados en distancias, caracterizados por no construir ningún modelo. Presentan como principal inconveniente el tiempo de respuesta, que empieza a degradarse cuando el número de ejemplos es muy grande (debido a que tiene que consultar todos los ejemplos para cada nueva instancia que se quiere clasificar). De este paradigma podemos encontrar los trabajos de Wang y Zucker [201] quienes propusieron dos alternativas, Citation-KNN y Bayesian-KNN, que extienden los algoritmos de los k vecinos más cercanos al contexto MIL. Estos algoritmos necesitan modificar la definición de la métrica de la distancia para su adaptación al aprendizaje multi-instancia. Wang y Zucker definen un cálculo de la distancia, denominada distancia de Hausdorff, para medir la proximidad entre las bolsas. De este modo, la distancia entre dos bolsas se define como la distancia más corta entre dos de sus instancias, en la figura 2.6, se representa gráficamente el cálculo de dicha distancia. Más tarde, Zhou et al. [237] presentaron el algoritmo Fretcit-KNN, que es una variante del Citation-KNN que obtiene mejores resultados. Este algoritmo emplea la distancia mínima de Hausdorff y utiliza tanto referencias como citas de las nuevas bolsas para determinar su etiqueta.

Las técnicas de aprendizaje basadas en árboles de decisión y en la inducción de reglas son uno de los paradigmas más ampliamente utilizados en la extracción de conocimiento debido a que son fáciles de usar, aportan conocimiento comprensible y suelen resultar métodos bastantes eficientes. Por ello, han sido también uno de los primeros paradigmas en introducirse a MIL. En este contexto, una de las principales aportaciones fue realizada por Chevaleyre y Zucker [34] que realizaron el primer estudio que demostró que la extensión de los algoritmos clásicos para resolver los problemas con múltiples instancias era una opción realmente interesante. Dichos

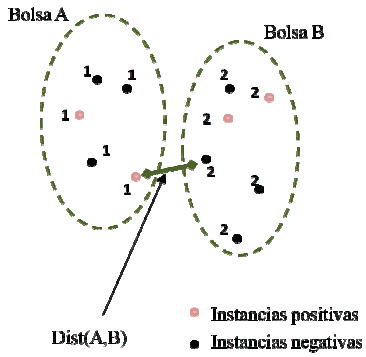


Figura 2.6: Funcionamiento de los algoritmos de aprendizaje basados en kNN

autores realizaron la adaptación de dos algoritmos muy populares en el aprendizaje tradicional, el algoritmo ID3 y RIPPER.

ID3 es un algoritmo clásico de árboles de decisión. Para llevar a cabo su adaptación es necesario adaptar los conceptos de entropía y ganancia de información al contexto del aprendizaje multi-instancia. En este algoritmo las instancias son divididas en cada nodo del árbol en lugar de las bolsas, lo que implica que las instancias de una bolsa pueden ir a diferentes ramas del árbol. Para clasificar una bolsa, todas sus instancias son pasadas a través del árbol multi-decisión que genera el clasificador, si una hoja positiva es alcanzada por una de las instancias, la bolsa es clasificada como positiva, de otro modo es etiquetada como negativa.

RIPPER es un algoritmo de inducción de reglas, en el que las reglas se van induciendo de una en una y todos los datos de entrenamiento cubiertos por dicha regla serán borrados después de que la regla se haya establecido. Para llevar a cabo la adaptación del algoritmo RIPPER, es necesario adaptar el concepto de cubrimiento de una regla al contexto de los objetos multi-instancia. Esta nueva definición se logra sustituyendo el enfoque de discriminación de instancias por el de discriminación de bolsas.

Al mismo tiempo, Ruffo [163] presentó una versión multi-instancia de algoritmo de árbol de decisión C4.5, que llamó RELIC, realizando también las adaptaciones necesarias con el concepto de ganancia de información y realizando en el algoritmo las tareas de pre-procesado y de post-procesado.

Las redes neuronales, paradigma bien conocido en el aprendizaje supervisado tradicional donde ha logrado altas precisiones siempre que se ha aplicado a problemas reales de complejidad considerable, también ha sido incorporado al aprendizaje con múltiples instancias. En la literatura podemos encontrar un número importante de contribuciones que extienden a las redes neuronales al contexto de MIL. Como trabajo pionero en esta área se encuentra el de Ramón y Raedt [156] quienes presentaron un marco de trabajo de las redes neuronales en un entorno de aprendizaje multi-instancia. A raíz de su trabajo han aparecido otras propuestas que mejoran o amplían en cierta medida a la original. Zhou y Zhang [240] propusieron BP-MI, una red neuronal basada en un sistema de aprendizaje multi-instancia derivada de la red neuronal tradicional de retropropagación. Este algoritmo fue mejorado por los mismos autores [229] incorporando dos métodos de selección de características (uno de ellos escalando características con el algoritmo DD y otro que reduce características mediante análisis de componentes principales). Más tarde, en [230] diseñaron un algoritmo de aprendizaje multi-instancia llamado RBF-MIP que deriva del método basado en una función con base radial (*Radial Basis Function*, RBF). La primera capa de esta red está compuesta de *clusters* de bolsas, donde la métrica de Hausdorff es utilizada para medir las distancias entre bolsas y entre *clusters*. Los pesos de la segunda capa de la red son optimizados minimizando una función de error de la suma al cuadrado y calculando los valores singulares a través de descomposición. Chai y Yang [28] proponen un algoritmo de redes neuronales para MIL basada en una red RBF normalizada, definiendo una nueva función de *kernel* para tratar con bolsas formadas por múltiples instancias.

Las Máquinas de Soporte Vectorial (*Support Vector Machines*, SVM), cuentan con un gran reconocimiento en el aprendizaje tradicional al tratarse de técnicas muy eficientes que permiten trabajar con datos con alta dimensionalidad, proporcionando modelos muy precisos. Este paradigma, como no podía ser de otra manera, también ha sido considerado en MIL. Existen numerosas propuestas de este enfoque, Gartner et al. [80] adaptaron la medida de distancia del *kernel* para trabajar con datos MIL. Los *kernel* con los que trabajan separan los conjuntos positivos y negativos en un entorno de múltiples instancias. Por otro lado, los trabajos de Chen y Wang [33] y Chen et al. [31] se centraron en adaptar las SVMs modificando la forma de los datos en lugar de modificar los algoritmos SVM. En estos trabajos formulan MIL como un problema cuyo margen de error está definido por la función

DD. El enfoque que proponen, llamado DD-SVM, se realiza en dos pasos. Primero, se determina mediante DD, el prototipo de las instancias, es decir las características del espacio de características de las instancias que se van a considerar y cada prototipo es elegido para ser un maximizador local de la función DD. Como DD mide la co-ocurrencia de instancias similares desde bolsas diferentes con la misma etiqueta, un prototipo representa la clase de instancias que es más probable que aparezca en unas bolsas con unas determinadas etiquetas que en otras. En el segundo paso, se define una correspondencia no lineal usando los prototipos creados, de esta forma se evalúa cada bolsa con respecto a un punto en un nuevo espacio de características, que es llamado espacio de características de las bolsas. En el espacio de características de la bolsa, el problema original de MIL se convierte en un problema corriente de aprendizaje supervisado y puede entonces entrenarse por SVMs estándar. Este método emplea una hipótesis generalizada de MIL, asumiendo que una bolsa positiva debe contener un número de instancias que satisfacen diferentes propiedades, para poder se considerada positiva.

Andrews et al. [4] consiguieron obtener uno de las mejores métodos de clasificación adaptando directamente las funciones *kernel* de SVM para MIL. Realizó dos propuestas. La primera de ellas, mi-SVM, para clasificación a nivel de instancias. Explícitamente trata las etiquetas de las instancias como variables ocultas no observables con restricciones definidas por sus etiquetas de la bolsa. La meta es maximizar conjuntamente el margen de error de las etiquetas de las instancias no conocidas y una función discriminante dada. La segunda, MI-SVM, para clasificación a nivel de bolsa. Intenta maximizar el margen de error de la bolsa, que es definido como el margen de error de la instancia más positiva en cada una de las bolsas positivas, o el margen de las instancias menos negativa en el caso de bolsas negativas.

Más recientemente, Qi and Han [153] han propuesto MIL-based SVMs. Este método hace uso de forma conjunta del algoritmo de DD y de un algoritmo de búsqueda optimizado que mejora la eficiencia y la precisión de la extracción de las características de la bolsa. Integra dos conjuntos de máquinas de soporte vectorial, uno basado en características globales y otro basado en características de las bolsas. Una vez que estos dos conjuntos complementarios de SVMs son configurados, sus valores de decisión son convertidos a valores de probabilidad que incorporan un

método de estimación de pesos automático para obtener los resultados finales de cada ejemplo de test no categorizado.

Otros trabajos que utilizan este enfoque son los de Mangasarian y Wild [125] y de Gu et al. [88]. Los primeros formularon una SVM con una función núcleo lineal y no lineal así como la minimización de una función lineal en un espacio real finito dimensional sujeto a restricciones lineales y bilineales. Los segundos proponen un algoritmo llamado MILC2 para modelar explícitamente relaciones multi-instancia a través de las capas y resolver así el problema de propagación ambigua. MILC2 reformula la función objetivo con una restricción de consistencia explícita entre las capas.

Por último, hay trabajos que muestran el uso de *ensembles* para mantener múltiples instancias. Como características más reseñables de estos modelos están que han demostrado ser normalmente más precisos que los sistemas simples y son, actualmente, una de las áreas más activas de investigación en el aprendizaje supervisado. Como se ha estudiado con los otros paradigmas, sus algoritmos también han sido tenidos en cuenta en MIL. A continuación mostraremos algunas adaptaciones propuestas. Zhou y Zhang [241] emplean un algoritmo relativamente simple, *Bagging*, para construir *ensembles* en el aprendizaje multi-instancia. El procedimiento que emplean consiste en generar diferentes conjuntos de entrenamiento desde el conjunto de entrenamiento original y entrenar componentes de aprendizaje desde cada conjunto de entrenamiento generado. Las predicciones de las componentes de aprendizaje son combinadas por medio del método de voting (por mayoría). Construyen *ensembles* multi-instancia para cuatro sistemas de aprendizaje base, que son, Iterated-discrim APR, Diverse Density, Citation-kNN, y EM-DD. Posteriormente, Xu y Frank [217] introducen los algoritmos MILogisticRegression y MIBoosting, que son versiones multi-instancia de los métodos de regresión logística lineal y regresión logística aditiva, respectivamente. Para realizar la adaptación, conectan las predicciones de las instancias con la estimación de la probabilidad a nivel de bolsa, asumiendo que todas las instancias contribuyen de igual manera y de forma independiente a las etiquetas de las bolsas. Más recientemente, Zhou y Zhang [242] proponen el método CCE. Este método emplea un proceso de *clustering* para adaptar la representación multi-instancia a las empleadas por los algoritmos que trabajan con una instancia. Además, CCE utiliza el poder del paradigma de

ensembles para lograr una mayor capacidad de generalización. Un resultado interesante, es que muestran que CCE puede aplicarse a problemas generalizados de multi-instancia sin ninguna modificación, lo que resulta difícil para la mayoría de los algoritmos actuales de aprendizaje multi-instancia. Pang et al. [147] proponen un método de regresión logística que hace uso de *boosting* para solucionar el problema del reconocimiento de objetos de imágenes, el método que proponen es llamado LMIB.

El gran número de adaptaciones del aprendizaje supervisado que se han realizado, ha llevado a numerosos estudios teóricos que analizan las relaciones entre el aprendizaje supervisado tradicional y el de múltiples instancias. Ray y Craven [158] estudiaron empíricamente la relación entre el aprendizaje supervisado y el aprendizaje de múltiples instancias comparando distintos métodos MIL con sus homólogos supervisados. Observaron que las técnicas de aprendizaje supervisado pueden mejorar a algunos enfoques de MIL, pero no llegaron a demostrar la existencia de un claro ganador. Zhou [235] mostró que los sistemas de aprendizaje de múltiples instancias pueden derivar de los supervisados cambiando su enfoque de discriminación de instancias a discriminación de bolsas. Recientemente, Zhou y Xu [238] han establecido un puente entre el aprendizaje con múltiples instancias y el aprendizaje semi-supervisado que demuestran que el aprendizaje multi-instancia puede verse como un caso especial de aprendizaje semi-supervisado.

MIL también ha atraído la atención de la comunidad de la Programación Lógica Inductiva (*Inductive Logic Programming*, ILP). De Raedt [49] mostró que los problemas multi-instancia pueden considerarse como un *bias* de la programación lógica inductiva. También sugirió que el paradigma multi-instancia podría ser la clave entre la representación proposicional y relacional, siendo más expresivo que el primero, y más fácil de aprender que el segundo. Zucker y Ganascia [248; 249], presentaron REPEAT, un sistema ILP basado en un ingenioso bias que primero reformula los ejemplos relacionales en una base de datos multi-instancia y después induce la hipótesis final con un sistema multi-instancia. Más tarde, Alfonso y Matwin [2] emplearon con éxito MIL para ayudar al aprendizaje relacional, donde el poder expresivo de la representación relacional y la facilidad de selección en una representación proposicional son combinados. Este trabajo confirma que MIL podría actuar como un puente entre el aprendizaje proposicional y relacional.

Por último comentar que, aunque todas las referencias que se han estudiado están basadas en un contexto de aprendizaje supervisado, al poseer información parcial de la salida de los ejemplos a la hora de llevar a cabo el proceso del aprendizaje. Están empezando a aparecer propuestas que introducen el aprendizaje no supervisado a MIL, donde las bolsas del conjunto de entrenamiento no poseen etiquetas. El trabajo de Zhang y Zhou [232] investigan el *clustering* en el aprendizaje con múltiples instancia con el método BAMIC (BAg-level Multi-Instance Clustering). Concretamente, BAMIC, intenta particionar el conjunto de bolsas no etiquetas en k grupos disjuntos de bolsas, donde se utilizan diferentes formas de la distancia de Hausdorff para medir las distancias entre las bolsas con las que el algoritmo k-MEDOIDS se adapta para cumplir con la tarea de *clustering*.

2.2.4.2. Estudio de las aplicaciones de MIL

La primera aplicación que se desarrolló basada en la representación con múltiples instancias, propuesta por Dietterich et al., fue la predicción de actividad en los fármacos [57], actividad que se ha convertido en un *benchmark* con el que cualquier método nuevo se debe enfrentar. El problema consiste en determinar si una molécula de un fármaco podrá llevar a cabo una unión con una proteína final. Las moléculas que permiten la unión son ejemplos positivos y las que no lo permiten serían los ejemplos negativos. Una molécula puede adoptar un amplio número de formas o conformaciones moviendo sus enlaces. Se define una molécula positiva si tiene al menos una configuración que permita crear la cavidad necesaria para que se pueda realizar la unión y producir el resultado esperado. Por el contrario, una molécula será negativa si ninguna de sus conformaciones se puede unir bien con el destino. El problema consiste en inferir los requerimientos necesarios que debe tener la molécula para producir una determinada actividad. Definido así este problema, su representación a MIL resulta inmediata: cada molécula se representa como una bolsa y las formas que puede adoptar serían las instancias de la bolsa. Las características de una instancia que se consideran son la distancia desde un origen a las diferentes posiciones que la superficie de la molécula puede realizar. Esta aplicación será tratada con mayor profundidad en el apéndice A, ya que será una de las aplicaciones con las que se realizarán experimentos con nuestro modelo.

Los primeros modelos que se diseñaron utilizan al menos dos conjuntos de datos que representan este problema (conocidos como Musk1 y Musk2). Así, que normalmente cualquier estudio de una nueva propuesta suele acudir a estos conjuntos de datos para comprobar su funcionamiento. Los trabajos de [4; 126; 201] son algunos que los han utilizado, aunque como se ha comentado prácticamente cualquier trabajo nuevo ha comparado con este conjunto de datos.

Tras esta propuesta, una gran cantidad de problemas del mundo real han sido adaptados a este paradigma demostrando un mejor funcionamiento. Algunas representaciones se realizan de forma muy directa, aunque no todas ellas tienen una adaptación tan inmediata a este aprendizaje, siendo necesario en algunos casos realizar algunas consideraciones para poder modelarlo y aprovechar la flexibilidad que nos proporciona este aprendizaje. A continuación detallaremos algunas de las más utilizadas:

Recuperación de imágenes basada en contenido y anotación de imágenes, el problema de clasificación y recuperación de imágenes consiste en identificar un determinado objeto en las imágenes. Este proceso es complicado porque las imágenes pueden contener múltiples y heterogéneos objetos. De este modo, identificar los objetos dentro de las imágenes que son relevantes continúa siendo hoy en día un problema muy complejo de resolver mediante métodos de aprendizaje supervisado. MIL permite una representación que se ajusta muy bien a la información que se dispone. Cada imagen puede ser tratada como una bolsa y los segmentos de dicha imagen son instancias de dicha bolsa. Estudiaremos esta aplicación más profundamente en el apéndice A, al ser una de las aplicaciones que se utilizan en la experimentación.

Existen numerosas referencias que han solucionado este problema, pero considerando diferentes imágenes a identificar. No obstante, la representación que emplean los diferentes sistemas, se basan en el mismo modelo. Ratan y Maron [157] particionaron las imágenes de escenas naturales pertenecientes a la galería de imágenes de Corel en sub-imágenes de tamaño fijado y aplicaron el algoritmo DD para clasificarlas en clases semánticas. Yang y Lozano [220] usaron un enfoque similar para recuperación de imágenes basada en contenido indicando que los métodos tienden a funcionar mejor cuando se consideran imágenes con objetos, en lugar de considerar escenarios naturales. Zhang et al. [234] compararon los algoritmos DD y EM-DD para recuperación de imágenes. En lugar de particionar las imágenes en

regiones de tamaño fijo, este trabajo usa algoritmos de segmentación de k-medias para generar regiones de imágenes más significativas. Más recientemente, Pang et al. [147] proponen resolver el problema del reconocimiento de personas, con la complejidad añadida de mantener las variaciones con respecto a las diferentes ropas que pueden llevar. Otras aplicaciones recientes que han solucionado este problema son [148; 92; 35].

Categorización de textos, un argumento similar al de las imágenes se puede dar en la categorización de texto. Un texto consiste en múltiples párrafos, cada uno de los cuales puede tratar temas diferentes. El problema consiste en aprender a clasificar documentos de acuerdo a su temática. La definición de los documentos tienen una estructura compleja, que lo convierte en una aplicación compleja para el aprendizaje tradicional. La representación con MIL, facilita la tarea de representación donde cada documento sería una bolsa y los párrafos en el documento serían instancias en la bolsa. Construir un modelo de categorización de textos a partir de las regiones relevantes parece ser una alternativa prometedora. Por ello MIL, parece ser una buena alternativa para esta tarea, ya que un documento es clasificado como relevante para una categoría si al menos una parte del documento trata de ese tema, no requiriendo que cada párrafo del documento sea precisamente del mismo tema.

Andrews et al. [4], realizan dos tareas de categorización aplicando un método basado en SVM. Los elementos con los que trabajan son 400 documentos con 200 documentos relevantes y 200 irrelevantes. Cada documento es divido en párrafos representados por 50 palabras, lo que resulta en 3000 instancias de cada conjunto de datos. El trabajo es evaluado sobre el conjunto de documentos TREC9, aunque por desgracia no hay comparaciones con métodos de categorización tradicionales ni con otras propuestas de MIL para poder obtener unas conclusiones sobre la representación.

Predicción de los valores de la bolsa de mercados, en este problema nos encontramos que cada mes hay valores que suben debido a razones con fundamentos y otros que no atienden a ningún fundamento; basándose en obtener información de los valores que suben de acuerdo a algún fundamento se realiza una representación MIL del problema, donde cada mes se recogen los 100 valores que más han subido en la bolsa y se ponen en bolsas positivas, esperando que al menos uno de ellos haya sido por razones fundamentadas. Las bolsas negativas son creadas con los 5

valores más bajos de cada mes. Cada instancia es representada por 17 características relacionadas con las medidas de los valores en la bolsa de mercados.

Esta aplicación ha sido resuelta por Maron y Lozano-Perez [126] utilizando datos de 600 valores principales desde 1978 y utilizando su algoritmo DD. Aunque resuelven el problema obteniendo buenos niveles de predicción, no tenemos resultados acerca de otras propuestas basadas en la representación tradicional ni con otros algoritmos de MIL.

Recomendación de las páginas web índice, este problema consiste en la recomendación de páginas web índice que puedan resultar interesantes para un usuario basándose en las página que el usuario ha consultado previamente y en las que mostró interés. Un usuario muestra interés en una página si ésta contiene algún tema que le interese y por el contrario no es interesante si no contiene ningún tema de interés. Para poder resolver el problema es necesario detectar los intereses de los usuarios, y en función de sus preferencias, realizar la recomendación. Esta es una de las aplicaciones seleccionadas para probar nuestro modelo y por tanto será detallada en el capítulo 6.

MIL también parece un campo prometedor en esta área. Los sistemas de recomendación se están convirtiendo, cada vez más, en herramientas esenciales debido a la gran cantidad de información que está disponible y en la necesidad de sistemas que nos orienten para encontrar lo que realmente nos puede interesar. Este problema cuenta con un problema adicional, que consiste en la necesidad de conocer los gustos de los usuarios para poder realizar recomendaciones de calidad y al principio se necesita que el usuario nos facilite algo de información. Esta etapa, en la que se requiere la colaboración del usuario, es la más difícil de abordar, por tanto es muy importante el diseño de sistemas que funcionen bien con la mínima información posible. MIL permite en este ejemplo que el usuario simplemente proporcione información sobre si una página le interesa o no, sin tener que detallar el o los enlaces concretos que son de su interés.

Zhou et al. [239] realizaron un primer estudio entre técnicas de aprendizaje supervisado tradicional y multi-instancia demostrando la mejoría de estos últimos y pusieron los datos disponibles. Otros trabajos relacionados son [218; 224].

Recuperación y detección de video, el video en la representación multi-capa es construido por medio de distinta información por capa, en este caso una toma está compuesta por la toma, el marco principal y la región. Por medio de multi-instancia cada ejemplo tiene correspondencia con múltiples instancias: por ejemplo, en la estructura de tres capas, un marco principal puede verse como una bolsa de regiones y también como una instancia de una toma. A partir de cada capa se generan las hiperbolsas. La hiperbolsa es positiva si y solo si una de sus instancias de la subcapa cae dentro del concepto, y es negativa si todas las instancias de las subcapas son negativas.

Este problema ha sido abordado muy recientemente en los trabajos [88; 32] que emplean un conjunto de datos llamado TRECVID (*Video Retrieval Evaluation*, TREC). Gu et al. [88] diseñan el modelo MILC2 para detectar el video sobre el corpus mencionado. Chen et al. [32] proponen un marco interactivo para recuperar los eventos semánticos del video de forma automático. Después del preprocesamiento, las trayectorias del objeto y los eventos del modelo son enviados al sistema de aprendizaje que analiza la serie de datos que le llega. Además, emplea realimentación humana en los resultados de recuperación mostrado para guiar las siguientes recuperaciones.

Detección asistida por ordenador (*Computer-Aided Detection*, CAD), las aplicaciones CAD consisten en detectar tumores malignos y lesiones partiendo de las imágenes médicas (escáner, radiografías, etc). En los algoritmos CAD este problema es direccionado en tres pasos, identificación de regiones con problemas (regiones no saludables), el cálculo de las características descriptivas para cada candidato, y etiquetar cada candidato como sano o enfermo por un clasificador.

El problema desde una perspectiva MIL es ligeramente diferente a las descripciones de las aplicaciones anteriores por dos motivos principalmente. Primero, en CAD no se tiene el concepto de bolsa negativa, es decir cada instancia negativa por si es una bolsa y segundo, no se tiene el concepto de objetivo único, es decir, la lesión puede aparecer en diferentes formas y características.

Las bolsas son representadas por medio de la idea de la envolvente convexa. En esta representación se fuerza a una minimización ya que cada instancia negativa es

considerada como un bolsa individual. El motivo de realizar esto es que la clasificación correcta de todos los candidatos no es tan importante como ser capaces de detectar los casos que no presentan ningún problema con gran seguridad.

El trabajo que introduce este problema es el de Murat el al. [59] y demuestran que esta representación permite detectar los casos mejor que los anteriores clasificadores que se habían utilizado.

3

Programación Genética y Aprendizaje Basado en Reglas

3.1. Introducción

La Computación Evolutiva (*Evolutionary Computation*, EC) comprende cuatro paradigmas fundamentales que son los Algoritmos Genéticos (*Genetic Algorithm*, GA), la Programación Genética (*Genetic Programming*, GP), las Estrategias Evolutivas (*Evolution Strategies*, ES) y la Programación Evolutiva (*Evolutionary Programming*, EP). Todas ellas tienen su base en el modelo de la evolución natural, por lo que comparten un conjunto de características comunes, que se pueden resumir en:

- Utilizan una estrategia de aprendizaje colaborativo a partir de un conjunto de individuos. Normalmente, cada individuo representa o codifica un punto del espacio de búsqueda de soluciones en un problema dado. Cada individuo incorpora información adicional que permite llegar a la solución final del problema.

- La descendencia, obtenida a partir de los individuos de la población, se genera de forma pseudo-aleatoria mediante procesos de cruce y recombinación donde se intercambia información entre dos o más individuos. La mutación es un proceso de auto-replicación errónea de los individuos que produce pequeños cambios en los mismos.
- Mediante la evaluación de los individuos, se consigue una medida de la adaptación de cada uno de ellos a su entorno. De acuerdo con esta medida de adaptación, el proceso de selección favorece más a los individuos mejor adaptados, que serán por tanto los que se reproduzcan más frecuentemente.

Son muchas las aplicaciones donde los algoritmos de EC han ofrecido soluciones eficientes alcanzando unas precisiones aceptables en diferentes problemas complejos. En un primer momento, los principales dominios donde se utilizaron estaban orientados hacia:

- Optimización: la evolución en sí es un proceso de optimización [129] en el que se busca adaptar, lo mejor posible, los individuos al entorno en el que habitan. Sin embargo, optimización no significa perfección. La evolución descubre soluciones funcionales altamente precisas para problemas concretos que se dan en el medio ambiente de un determinado individuo.
- Sistemas robustos: los problemas del mundo real casi nunca son estáticos y los problemas de optimización temporal son cada vez más comunes. Estas circunstancias requieren un cambio en la estrategia que se aplica para resolver el problema.
- Inteligencia artificial: como se puede comprobar, la evolución ha creado especies con una inteligencia cada vez más desarrollada. Por tanto, la forma de conseguir inteligencia artificial, además de las alternativas clásicas, tiene una nueva posibilidad: simular la evolución para construir algoritmos predictivos.
- Vida artificial: en el campo de la biología, más que utilizar la evolución como una herramienta, se intenta capturar la esencia de la propia evolución en una simulación por computador y usar entonces esta simulación para conseguir una nueva información sobre la física de los procesos de la evolución natural [159].

3.2. Programación genética

La Programación Genética propuesta por John R. Koza [117] surgió a comienzos de la década de los 90's, y se caracteriza porque los individuos codifican programas de longitud variables capaces de solucionar un problema determinado. La población de individuos es sometida, a los operadores de selección, cruce, mutación y reemplazo, debidamente adaptados para simular el funcionamiento de la evolución natural.

La GP mantiene un enfoque similar al resto de paradigmas con respecto al proceso evolutivo. La GP se caracteriza porque la representación de sus individuos no sigue en esquema de fijo sino que se escogen los elementos que los conforman, es decir, se generan a partir de las acciones de los programas (funciones) y los datos sobre los que actúan los programas (terminales), y se opera con ellos hasta encontrar una estructura que represente una solución óptima al problema. Se puede decir, que la GP busca construir programas de computación sin que ellos sean diseñados y programados expresamente. La creación de esos programas es completamente aleatoria, es decir, el computador no ha sido explícitamente programado para encontrar una solución predeterminada, sino que a esta solución se llega partiendo de un estado inicial dado, aplicando un proceso de evolución que selecciona las estructuras intermedias que se van encontrando [117].

El conjunto de las posibles estructuras lo integran todas las combinaciones de funciones que se pueden componer recursivamente a partir de un alfabeto, compuesto por un conjunto de N_{func} funciones ($F = \{f_1, f_2, \ldots, f_{N_{func}}\}$) y un conjunto formado por N_{term} terminales ($T = \{a_1, a_2, \ldots, a_{Nterm}\}$).

Cada función f_i del conjunto F, trabaja con un número específico de argumentos, ejemplos de funciones pueden ser:

- Operadores aritméticos (+, -, *, ...).
- Funciones matemáticas (sin, cos, exp, log, ...).
- Operadores booleanos (AND, OR, NOT, ...).
- Operadores condicionales(if-then-else).
- Funciones de iteración (Do-until).

- Funciones recursivas.
- Funciones específicas del dominio que deben ser definidas.

Los terminales pueden ser variables (pudiendo representar entradas, sensores, detectores o estados de un sistema) o constantes.

Como se ha comentado, el objetivo de la GP en sus orígenes estaba orientado a evolucionar programas de ordenador siguiendo una representación basada en árboles. No obstante, esta técnica se emplea hoy día para evolucionar otras abstracciones de conocimiento, como por ejemplo expresiones matemáticas o sistemas basados en reglas.

Existen una serie de factores y problemas propios de la GP que le permiten distinguirse claramente de los GAs, y en general de los otros tipos de EAs existentes. La principal diferencia radica en el tipo de representación usada, la GP hace uso de un lenguaje de representación complejo para codificar los individuos de la población. En GP los individuos consisten no sólo en estructuras de datos, sino también en operaciones (operadores y funciones). Así, un individuo representado como un árbol, las hojas corresponden con los símbolos terminales (variables y constantes), mientras que los nodos internos se corresponden con los no terminales (operadores y funciones). Esta estructura hace que sea necesario que el conjunto de no terminales cumpla dos propiedades que son características de la GP: la suficiencia y la clausura.

La suficiencia hace referencia al hecho de que el poder expresivo de los conjuntos de terminales y no terminales debe bastar para poder representar una solución para el problema en cuestión.

La clausura (*closure*) se refiere a que una función u operador debería ser capaz de aceptar como entrada cualquier salida producida por cualquier función u operador del conjunto de no terminales. En la práctica, no es raro encontrarse con situaciones en las que los programas a evolucionar tengan que manejar variables de distinto tipo, lo cual hace que sea difícil de satisfacer esta propiedad, en la siguiente sección trataremos diferentes variantes de la GP en función de cómo resuelven esta propiedad.

3.2.1. Variantes de la programación genética

Desde que Koza diseñase los algoritmos de GP, éstos han evolucionado hacia diferentes formas generando distintas variantes de la GP que emplean diferentes formalismos para resolver uno de los problemas que tiene la GP, que es la propiedad de cierre. Esta propiedad es una de las de las principales restricciones que debe satisfacer la GP cuando se realiza el proceso evolutivo, ya que su no consideración generaría individuos no válidos con la consiguiente pérdida de eficiencia que supone este hecho en el proceso. Es necesario, por tanto, hacer una consideración sobre cómo imponer restricciones a los individuos que evolucionan. La solución tradicional a este problema consistía en conseguir que todas las funciones tengan la propiedad de cierre [117], es decir, que todas las funciones pueden aceptar todo tipo de argumentos.

No obstante, en muchas ocasiones es difícil mantener esta restricción sintáctica si queremos usar funciones que emplean diferente tipo de argumentos. Por ejemplo, podemos querer limitar la función suma a argumentos que sean enteros, por lo que dicha función solo podrá tomar como argumentos o bien constantes enteras, o bien funciones que solo devuelvan enteros. En este caso, la solución anterior ya no sería válida y es necesario modificar tanto el proceso de generación de la población inicial como los operadores genéticos para que sólo generen individuos sintácticamente correctos.

Daremos una breve descripción de alguna de las variantes más importantes que se han desarrollado:

- GP fuertemente tipada, el propósito de esta variante es forzar la propiedad de cierre permitiendo el uso de variables de diferente tipo de datos. Para estos problemas, se define un conjunto de reglas sintácticas para cada nodo padre, especificando el tipo de nodos que se pueden usar como sus nodos hijos. Estas restricciones se imponen siempre que se crea un nuevo árbol mediante los operadores genéticos [138].

- GP basada en gramáticas, otra solución propuesta para resolver el problema del cierre se basa en la utilización de gramáticas que especifiquen un lenguaje que los individuos deben respetar [207; 214]. La programación genética guiada por gramática (*Grammar Guided Genetic Programming*, G3P) hace uso

de operadores genéticos que tienen en cuenta una gramática, de forma que se garantiza que cada individuo generado tanto los que forman la población inicial como los que se van generando a lo largo del proceso evolutivo, son legales con respecto a la gramática. Esta opción es la escogida para los modelos que se diseñan en esta tesis. Por lo que en la sección 3.2.2 se tratarán con mayor detalle.

- GP lineal, la principal característica de GP lineal [10] es que los individuos tiene una estructura lineal que representan una secuencia de instrucciones de un lenguaje de programación imperativo o lenguaje máquina, normalmente se emplea más la representación utilizando un lenguaje imperativo (como por ejemplo el lenguaje C).

3.2.2. Programación genética guiada por gramática

La G3P surge como una extensión a los sistemas tradicionales de GP [206] que utiliza gramáticas de contexto libre (*Context Free Grammar*, CFG) para establecer la definición formal de las restricciones sintácticas del problema. La G3P ha demostrado ser una aproximación con un gran rendimiento en problemas con dominios estructurados [206; 207] y es considerada como una de las áreas más prometedoras dentro de la investigación de la GP.

En esencia, la G3P utiliza los mismos componentes y operadores genéticos que la GP tradicional. Sin embargo existen diferencias significativas entre ambos sistemas:

- La primera diferencia fundamental aparece a la hora de generar aleatoriamente la población inicial: los individuos que conforman la población deben ser palabras aceptadas por el lenguaje definido por la gramática libre de contexto en cuestión. La utilización de los métodos tradicionales para inicializar la población en los sistemas de GP no son válidos para la G3P dado que los individuos se generan aleatoriamente a partir del conjunto de funciones y de terminales, esto conlleva la generación de individuos no válidos que impiden la convergencia hacia la solución.

- Otra diferencia fundamental aparece a la hora de diseñar los operadores genéticos; por un lado, el operador de cruce tradicional no es válido puesto

que los operadores tradicionales de cruce para GP no aseguran la generación de descendencia válida sintácticamente de acuerdo a la gramática. Por otra parte, el operador de mutación, una vez elegido el lugar de mutación, debe utilizar las reglas de producción de la gramática de forma similar a como ocurre en la generación de individuos para asegurar la mutación del individuo por otro igualmente válido.

3.2.3. Proceso evolutivo

La GP como los diferentes paradigmas de la EC se basa en los procesos de selección, reproducción y reemplazamiento, que son considerados fundamentales desde los orígenes de estos sistemas [97; 85]. Cada uno de ellos desarrolla una labor específica, bien diferenciada de las de los demás. La figura 3.1 muestra el funcionamiento genérico de la GP y en general de cualquier EA. A continuación se detallan los principales pasos que sigue el proceso evolutivo.

- Proceso de selección

  El operador de selección se encarga de seleccionar a los individuos destinados a tener descendencia agrupándolos en un conjunto que contienen a los padres que van a participar en el proceso de recombinación. Los individuos mejor adaptados de acuerdo a los principios propios de la naturaleza son aquellos que tienen mayor probabilidad de ser escogidos, por lo que la mayoría de los operadores de selección se basan en el valor proporcionado por la función objetivo para realizar la selección de los individuos. El método de selección más utilizado actualmente es el método del torneo [23]. Otros métodos de selección utilizados habitualmente son el método de la ruleta [9] y cuando se trabaja con poblaciones con un número elevado de individuos, el método de truncamiento [43].

- Operadores Genéticos

  Los operadores genéticos que se emplean para realizar la reproducción de los individuos más ampliamente utilizados son el cruce y la mutación.

  El operador de cruce se considera el más importante de los operadores genéticos. Su función consiste en cruzar cada pareja de padres que se encuentran en el

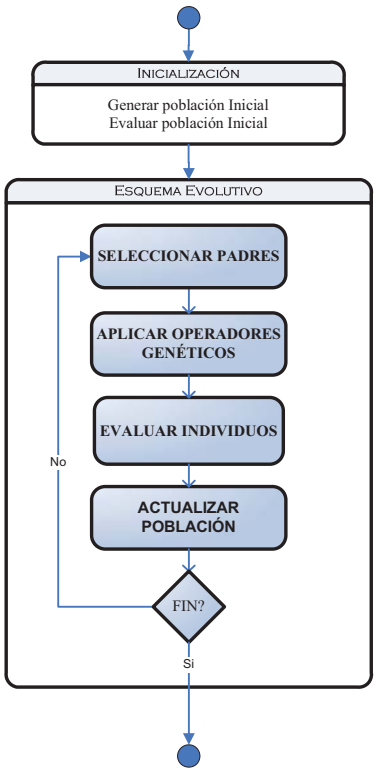


Figura 3.1: Proceso evolutivo de los algoritmos de programación genética

conjunto seleccionado en el proceso anterior para obtener su descendencia y así continuar con la evolución de la población. De este modo, cuanto mejor sea el operador de cruce elegido, más rápidamente converge la población hacia la solución del problema. Su funcionamiento es sencillo, se trata de elegir un nodo de cruce o lugar de cruce para cada uno de los dos padres y a continuación intercambiar los subárboles cuyas raíces son esos nodos de cruce. De esta forma, se obtienen dos nuevos individuos que, dependiendo del grado de adaptación alcanzado y de la política de reemplazo escogida, pueden pasar a formar parte (o no) de la nueva población.

El operador de mutación, actúa sobre un determinado individuo con una probabilidad p_m. Cuando un individuo resulta afectado por este operador, se selecciona aleatoriamente un lugar de mutación y el subárbol cuya raíz

es el nodo elegido para la mutación es reemplazado por un subárbol cuyo símbolo del nodo raíz coincide con el símbolo del nodo elegido para realizar la mutación. El operador de mutación encuentra su justificación como método para obtener nuevos puntos en los que recomenzar la búsqueda, evitando de esta forma la pérdida de diversidad genética en la población, lo cual tiene implicaciones directas en el proceso de convergencia del algoritmo. Tanto es así que, según ciertos estudios, el operador de mutación va ganando en importancia a medida que la población de individuos va convergiendo [45].

- Proceso de reemplazo de individuos

  Una vez obtenidos los individuos descendientes de una determinada población tras la aplicación de los diferentes operadores, es necesario formar una nueva población (la nueva generación) de λ individuos a partir de los λ individuos de la población de partida y los nuevos individuos obtenidos.

3.3. Sistemas evolutivos de aprendizaje basados en reglas

Aunque inicialmente los EAs se estudiaron para problemas de optimización, han demostrado un buen rendimiento en otras aplicaciones, entre ellas se encuentran los sistemas de aprendizaje. Desde que Holland [96] señalase su teoría de los sistemas adaptativos, los EAs han sido usado como los principales componentes en los sistemas de aprendizaje basados en reglas. Hasta el punto de que se están estableciendo como la principal heurística de descubrimiento en las distintas técnicas del aprendizaje automático. Entre las principales características que han promovido su utilización se encuentran: la búsqueda basada en poblaciones, la robustez que ofrecen, la flexibilidad en el representación del conocimiento y los grandes avances que se están produciendo con respecto a su eficiencia y competencias [86; 87; 151]. Esto ha resultado en la aparición de diferentes enfoques para usar EAs en el aprendizaje automático, que han mostrado ser realmente competitivos con respecto a los sistemas tradicionales no evolutivos.

En esta sección veremos en primer lugar una introducción a las sistemas basados en reglas. Posteriormente, se estudiarán los sistemas de representación utilizados

en el aprendizaje evolutivo y finalmente, centrándonos en la GP, realizaremos una revisión de los principales modelos de aprendizaje basado en reglas que se han propuesto

3.3.1. Introducción a los sistemas basados en reglas

Los sistemas basados en reglas (*Rule-Based Systems*, RBSs) se encuentran englobados dentro de los sistemas basados en conocimiento. La Ingeniería del Conocimiento es la disciplina de la Inteligencia Artificial encargada de hacer explícitos los conocimientos de un dominio en una base de conocimiento separada del resto del sistema en el que se integra.

Estos sistemas se han extendido mucho en el últimos años debido a que el uso de reglas es una de las formas más populares de representación del conocimiento debido, entre otras razones, a su sencillez, capacidad de expresión y escalabilidad. La estructura de estos sistema se muestra en la figura 3.2, siendo sus componentes principales una base de conocimiento y el motor de inferencia:

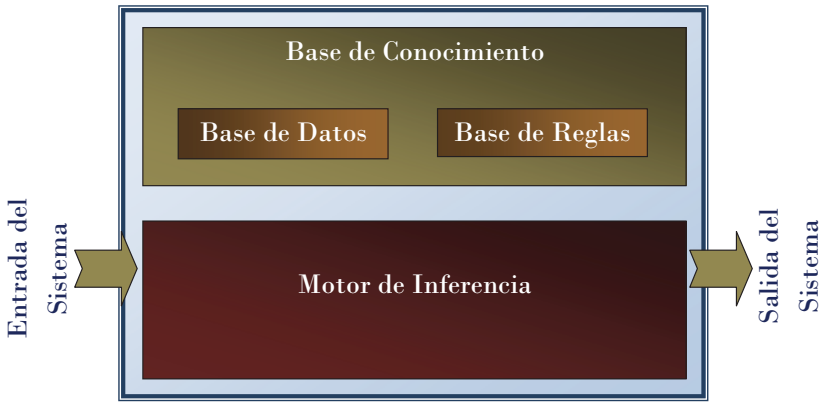


Figura 3.2: Estructura de los sistemas basados en reglas

- *Base de conocimiento*, almacena la representación del conocimiento sobre el dominio de aplicación del sistema. Se divide en la base de datos y en la base de reglas. En la base de datos se representa el conocimiento en las variables de entrada y salida del sistema, que forman parte de las reglas

semánticas almacenadas en la base de reglas. Una regla semántica es una representación del conocimiento en forma de sentencia condicional. En cada regla se combinan variables de la base de datos con las partículas condicionales SI y ENTONCES, así como con los operadores lógicos AND y OR. La parte de una regla situada entre SI y ENTONCES es el antecedente de la regla, siendo el resto el consecuente. En el antecedente se sitúan las premisas que deben cumplirse para que la regla sea aplicable. El consecuente es el conjunto de acciones derivadas de la aplicación de la regla.

Dependiendo de la naturaleza del conocimiento que almacenan, se ha establecido una tipología informal para este tipo de estructuras [72]. Así, se habla de reglas de clasificación, de decisión, de asociación, de predicción, de causalidad, de optimización etc. En el ámbito del aprendizaje automático, las más estudiadas han sido las reglas de asociación, de clasificación y de predicción.

Las reglas de clasificación tienen como objetivo almacenar conocimiento encaminado a la construcción de una clasificador preciso. En su antecedente contienen una serie de requisitos (en forma de condiciones) que debe cumplir un objeto determinado para que pueda considerarse que pertenece a la clase que identifica el consecuente de la regla.

Por otra parte, el propósito de las reglas de asociación es buscar relaciones entre unos atributos de la base de datos, que se colocan en el antecedente de la regla, y otros, que se colocan en el consecuente. Reglas de asociación típicas son las que informan sobre las preferencias de compra de un cliente de un establecimiento de comercio electrónico: SI (el usuario X adquiere el producto A) ENTONCES (adquiere el producto B).Estas reglas pueden presentar en su antecedente una o más condiciones donde se compara un atributo con un valor o dos atributos entre sí; el consecuente puede también presentar una o más condiciones. La principal diferencia que presentan con las anteriores es que las reglas de clasificación presentan una sola condición en el consecuente, que además es un identificador de clase.

Finalmente, las reglas de predicción comparten características sintácticas con las reglas de asociación y con las de clasificación. Al igual que sucede en el caso de las reglas de clasificación, el consecuente sólo presenta una condición. Sin embargo, el contenido de este consecuente no tiene que ser un identificador

de categoría o clase, sino que puede tratarse de una condición de cualquier tipo (que es la que se pretende predecir, de ahí el nombre de reglas de predicción). El significado semántico de una regla de predicción es que si todas las condiciones especificadas por el antecedente de la regla son satisfechas por los atributos predictores de un ejemplo, la regla predice que el atributo objetivo (el que aparece en el consecuente) de esa instancia tendrá el valor especificado en el consecuente de la regla.

- *Motor de Inferencia*, por cada entrada al sistema, el motor de inferencia utiliza el conocimiento almacenado en forma de reglas para generar una salida. La inferencia en un RBS se basa en técnicas de la lógica proposicional o booleana, donde cada variable (también llamada hecho) es representada mediante un símbolo (X) y únicamente puede tomar dos valores: presente (X) o ausente ($\neg$ X). La expresión más habitual de inferencia en lógica proposicional es el modus ponens, SI X ENTONCES Y, donde X e Y son variables booleanas, también llamadas bivalentes. En primer lugar, la regla proposicional *SI X ENTONCES Y* indica que, ante la presencia de X se deduce la presencia de Y. A raíz de la posterior observación de la presencia de X, se infiere la presencia de Y. La lógica proposicional permite la inclusión de los operadores lógicos de conjunción ($\wedge$) y disyunción ($\vee$) en los antecedentes de una regla de la forma, SI X $\wedge$ Y ENTONCES Z, siendo Z una variable booleana. Las variables de un RBS, en lugar de ser bivalentes, toman valores dentro de intervalos acotados definidos en la base de hechos y los operadores AND y OR hacen las veces de los operadores lógicos de conjunción ($\wedge$) y disyunción ($\vee$) respectivamente.

3.3.2. Sistemas de representación de reglas

En esta sección se pretende dar una introducción a la representación utilizada en estos sistemas, comenzaremos describiendo las dos representaciones más ampliamente utilizadas, y luego trataremos también algunos modelos híbridos que combinan ambas propuestas generando nuevos paradigmas.

Los dos principales enfoques para la codificación de los individuos cuando evolucionan clasificadores basados en reglas son,

- El enfoque *individuo = regla*, esta representación se caracteriza porque cada individuo codifica una regla que determina una salida concreta. Por lo tanto, la solución final de un determinado problema, el clasificador final, requiere de métodos que consideren un subconjunto de la población a partir de la cual se forme el clasificador completo. La evaluación global de estos sistemas requiere que todo el subconjunto de reglas esté creado, con lo que antes de la obtención del clasificador completo, los individuos son evaluados de acuerdo al funcionamiento que cada regla individual tiene sobre el conjunto de datos.
- El enfoque *individuo = conjunto de reglas*, cada individuo representa un clasificador completo, por tanto el clasificador final será el mejor individuo evolucionado. Puesto que cada individuo tiene un conjunto de reglas que representan un clasificador completo no hay necesidad de evaluar la calidad de cada uno de estos clasificadores de forma independiente. De ahí, que inmediatamente después de su creación, cada individuo es evaluado con una serie de ejemplos, que o bien han sido proporcionados al comienzo de la ejecución en la forma de datos estáticos o han sido recogidos durante el proceso de aprendizaje. Con este enfoque, no hay un control específico sobre la contribución de cada regla al desempeño del individuo completo.

Todas las implementaciones evolutivas siguen uno de los enfoques explicados anteriormente o un modelo híbrido que los combinen, por ejemplo

Iterative rule learning (IRL) [200], propone aprender iterativamente reglas que cubren un subconjunto de las instancias de entrada. Para ello realiza un procedimiento basado en dos pasos:

1. aprender una regla que cubra parte (o todos) los ejemplos de entrenamiento.
2. borrar los ejemplos cubiertos del conjunto de entrenamiento.

Este proceso es repetido hasta que no quede ningún ejemplo de entrenamiento. Al final del proceso, la solución es la concatenación de las reglas creadas en cada iteración. Este enfoque genera incrementalmente nuevas reglas y, a la misma vez, reduce el espacio de búsqueda ya que los ejemplos cubiertos son borrados del conjunto de datos de entrenamiento. Este método es también referencia como estrategia de cubrimiento [133].

Organizational Classifier System (OCS) [209], este enfoque también toma ideas de los dos enfoques básicos para estimar las dimensiones más apropiadas para las organizaciones, simulando la idea de transacciones de coste. OCS hereda las ideas principales de los sistemas de clasificación simple y se centran en intentar distinguir reglas que dirijan a decisiones óptimas a partir de las que dirigen a soluciones subóptimas para evolucionar conjuntos de reglas ideales. Para este propósito, el sistema distribuye los clasificadores de la población en diferentes organizaciones de tamaño variable. Estas organizaciones pueden interaccionar entre ellas. Para controlar el tamaño de las organizaciones, OCS incorpora ideas de la teoría de transacción de costes usando la reputación para el reclutamiento organizacional y pone atención al dimensionado de las organizaciones. Es decir, por un lado, OCS incluye un esquema de localización de créditos que usa reglas y reputación de organizaciones para determinar las interacciones entre los clasificadores y las organizaciones. Por otro lado, el sistema implementa una organización con un componente de crecimiento que controla el tamaño de las organizaciones aplicando diferentes operadores genéticos para ampliar o reducir organizaciones, que preserven la idea de que las organizaciones con mayor reputación puede ser mayor que las organizaciones con menor reputación.

3.3.3. Criterios de evaluación de las reglas

Para realizar la evaluación de la calidad de las reglas se puede distinguir entre medidas subjetivas de interés y medidas objetivas de calidad [176], ambas son necesarias para resolver de manera adecuada las tareas que se estén llevando a cabo. Los criterios más adecuadas para la evaluación dependen de la aplicación concreta. Detallaremos algunas de las medidas subjetivas y objetivas más utilizadas en los diferentes sistemas de aprendizaje.

Algunas de las medidas subjetivas más utilizadas son:

- Utilidad, mide el interés de una regla relacionándola con los objetivos del usuario [110].
- Accionabilidad, determina que una regla es interesante si aporta al usuario información que le permita llevar a cabo alguna acción que les suponga algún beneficio [82].

- Sorpresividad, una regla es interesante si es sorprendente para el usuario [82].
- Novedad, una regla es interesante si se aparta del conocimiento previo del usuario [110].
- Redundancia, tiene en cuenta la similitud de una regla con respecto a otras y mide hasta qué punto una regla se puede derivar de otra [110] o de un conjunto de reglas que cubren los mismo ejemplos.

Con respecto a las medidas de calidad objetivas, podemos distinguir entre medidas de calidad con mayor carácter predictivo y con mayor carácter descriptivo. En los enfoques de extracción de reglas de asociación, se suele utilizar medidas descriptivas para valorar la calidad de las reglas extraídas, mientras que en clasificación se utilizan medidas predictivas. A continuación se proponen distintas medidas predictivas objetivas utilizadas en clasificación para evaluar el interés de las reglas obtenidas.

- Exactitud es la proporción de casos clasificados correctamente considerando tanto los que cumplen una determinada propiedad como los que no.
- Sensibilidad es la proporción de casos clasificados correctamente considerando solamente los ejemplos que poseen una determinada propiedad.
- Especificidad es la proporción de casos clasificados correctamente considerando solamente los ejemplos que no poseen una determinada propiedad.

Estas medidas se basan en la matriz de confusión. Teniendo m clases, la matriz de confusión es una matriz de tamaño mxm. Una celda de la matriz, $c_{i,j}$, indica para la clase real del ejemplo, representada en la fila i, cual es la clase predicha,

Tabla 3.1: Matriz de confusión para dos clases

		Clase Predicha	
		$Clase_1$	$Clase_2$
Clase Actual	$Clase_1$	verdaderos positivos (t_p)	falsos negativos (f_n)
	$Clase_2$	falsos positivos (f_p)	verdaderos negativos (t_n)

representada en la columna j. Para tener una clasificación perfecta, solamente debería estar rellena los elementos de la diagonal de esta matriz.

Una matriz de confusión para dos clases se puede ver en la tabla 3.1. Cuando trabajamos con dos clases, el problema se puede abreviar y hablamos de ejemplos positivos y negativos. De esta manera de acuerdo a la notación utilizada los elementos verdaderos positivos (*true positives*) se refieren a los ejemplos positivos que son clasificados correctamente, los falsos positivos (*false negatives*) son los ejemplos positivos que se clasifican incorrectamente, los verdaderos negativos (*true negatives*) son los ejemplos negativos clasificados correctamente y finalmente, los falsos positivos (*false positive*) son los ejemplos negativos clasificados incorrectamente.

La exactitud es la mejor medida para evaluar el número de ejemplos que identifica de manera correcta nuestro clasificador, ya que considera la tasa de reconocimiento global del clasificador, que referencia como de bien el clasificador reconoce los ejemplos de las distintas clases. No obstante, esta medida tiene un inconveniente cuando las clases no están balanceadas. Supongamos que se ha generado un clasificador para clasificar ejemplos médicos como "cáncer" o "no cáncer". Un índice de exactitud de, por ejemplo, el 90 % puede hacer que el clasificador parezca bastante exacto, pero realmente el 3-4 % de los ejemplos pertenecen a la clase "cáncer". Evidentemente, un índice de exactitud del 90 % donde solamente se clasifiquen ejemplos que pertenecen a la clase "no cáncer", no sería un clasificador adecuado. Por ello, se hace necesario el tener información específica de cómo se clasificada cada una de las clases. Las medidas de sensibilidad y especificidad sirven para solucionar este propósito. La sensibilidad controlaría la proporción de ejemplos positivos que se clasifican correctamente, mientras que la especificidad hace referencia a la proporción de ejemplos negativos que se clasifiquen correctamente.

Estas medidas se definen como

$$sensibilidad = \frac{tp}{tp + fn} \tag{3.1}$$

$$especificidad = \frac{tn}{tn + fp} \tag{3.2}$$

Puede demostrarse que la exactitud final se puede definir en función de la sensibilidad y la especificidad:

$$exactitud = sensibilidad \cdot \frac{tp + fp}{tp + fn + tn + fp} + \tag{3.3}$$

$$+ especificidad \cdot \frac{tn + fp}{tp + fn + tn + fp}$$

Las medidas de la matriz de confusión son también útiles para determinar el coste y los beneficios asociados con un modelo de clasificación. Por ejemplo, el coste asociado con los f_n, es decir, clasificar incorrectamente una paciente que tiene cáncer es más perjudicial que los f_p que conllevaría ser más conservativo etiquetando a los pacientes que padecen cáncer cuando verdaderamente no padecen dicha enfermedad. En tales casos, se puede compensar un tipo de error sobre el otro asignando diferente coste a cada uno de ellos. De forma similar, los beneficios asociado con los t_p puede ser diferente que los de un t_n. Hasta ahora, para calcular la precisión del clasificador, hemos asumido los mismos costos, de este modo se divide la suma de t_p y t_n entre el número total de ejemplos. Alternativamente, podemos incorporar penalizaciones en cada una de las decisiones. Existen muchas otras aplicaciones donde existe una análisis de los problemas de equivocarse en la clasificación.

También existen otros problemas donde la exactitud por sí sola no es apropiada para la evaluación, por ejemplo cuando se asume que los casos que se pretende clasificar pueden pertenecer a más de una clase. En estos casos la medida de exactitud ya que no tiene en cuenta la posibilidad de que los ejemplos pertenezcan a más de una clase. En estos casos, es útil determinar una probabilidad de la distribución de las clases, de este modo la clasificación es correcta si se clasifica que pertenece a la primera o segunda clase más probable.

3.3.4. Modelos de programación genética basados en reglas

En esta sección nos centraremos en un estudio de los modelos de GP que emplean una representación basada en reglas aplicadas a tareas de clasificación, que será sobre lo que verse nuestro trabajo.

Los clasificadores basados en reglas, mantienen diferentes estructuras en función del número de clases con el que se trabaje, diferenciándose entre:

- Clasificación binaria, hay solamente dos clases para diferenciar.
- Clasificación multiclase, hay más de dos clases entre las que diferenciar. Estos algoritmos son más generales, a costa de introducir una mayor complejidad.

Los problemas binario y multiclase son algunas veces mantenidos de modo diferentes. Algunos métodos están restringidos para problemas binarios, mientras otros algoritmos pueden cubrir de un modo parecido cualquier número de clases. Cualquier problema de n clases puede ser reducido a $n - 1$ problemas de clasificación binaria, con lo que los métodos limitados a problemas binarios pueden ser aplicados a problemas multiclase. Sin embargo, los métodos capaces de mantener cualquier número de clases generalmente se adaptan mejor a la consideración de varias clases. En este estudio llevaremos a cabo una división entre los sistemas de clasificación binario y los que llevan a cabo la clasificación de varias clases [65].

Clasificación binaria

La clasificación binaria se lleva a cabo en las propuesta descritas en [61; 62], donde cada individuo de la población codifica una regla y el clasificador final del proceso evolutivo consiste en un individuo que representa el clasificador final, todos los ejemplos cubiertos por la regla que representa el individuo pertenecen a una clase y el resto de ejemplos los predice como pertenecientes a la otra clase. Las condiciones de las reglas simplemente consideran comparaciones entre atributos y valores, son por tanto, condiciones lineales univariantes. La evaluación mide la precisión de la clasificación, asignando diferentes pesos a cada instancia, lo que permite realizar una ponderación de la evaluación parcial de cada instancia. Los pesos también

evolucionan en el proceso evolutivo, de tal manera que los casos que son más difíciles de clasificar tienen un peso más alto.

En [202] los individuos son de longitud fija, con el fin de obtener clasificadores más sencillo y comprensibles. Cada individuo codifica una regla y la precisión es la función empleada para evaluar a los individuos. En [61; 62; 183], al igual que en los casos anteriores, cada individuo codifica una única regla para una de las clases, teniendo en cuenta que los ejemplos no cubiertos por dicha clase pertenecen a la otra. Estos sistemas permiten el uso de condiciones que comparan el valor de dos atributos (condiciones lineales multivariantes), además de las condiciones simples que comparan un atributo con un valor. La función de evaluación se basa en la exactitud de la clasificación. Un sistema similar se propone en [165], pero sólo se emplea clasificadores lineales univariantes. En [105], una única regla es desarrollada en cada ejecución, y el sistema se ejecuta dos veces, una para cada una de las clases. La función de evaluación se basa en la media cuadrática de error de predicción, incluyendo un factor de penalización para el tamaño. Operaciones aritméticas y comparaciones entre atributos pueden aparecer en las condiciones dando lugar a clasificadores no lineales.

En [222], también un individuo codifica una única regla, donde se admiten tanto condiciones lineales univariantes como multivariantes. Cuando termina la evolución, el clasificador final está formado por las n mejores reglas, donde n es un valor elegido por el usuario, de modo que se obtienen clasificadores con varias reglas. La función de evaluación se basa en las medidas de ROC/AUC.

Reglas difusas son evolucionadas en [11]. En este caso, un clasificador contiene varias reglas, pero todas las reglas predicen la misma clase, los ejemplos que no cubren dichas reglas pertenecen a la otra clase. Cada individuo codifica una regla, que emplea condiciones lineales univariantes. En el proceso evolutivo en cada ejecución se obtiene una regla y todas las instancias correctamente clasificadas por esta regla se eliminan para la siguiente iteración. Un enfoque multi-objetivo se utiliza con el fin de optimizar simultáneamente cuatro funciones de evaluación, tres de ellas relacionadas con la exactitud de la regla y la otra con el tamaño.

Un algoritmo de dos etapas se propone en [190]. En la primera etapa, un método híbrido de GP/GA se emplea para desarrollar un número inicial de reglas. Cada individuo representa una regla, para ser más exactos, una parte de una regla,

GP es empleado para evoucionar la parte del antecedente que involucra atributos categóricos y GA es usado para evolucionar las condiciones que conciernen a los atributos numéricos. Una técnica de nichos (*token competition*) se utiliza para obtener un cubrimiento adecuado de todas las instancias, y la medida de evaluación es la exactitud con el fin de obtener una adecuada cobertura de todos los datos de los ejemplos. La segunda fase del sistema usa un conjunto de reglas producidas en la primera fase. Cada individuo, en este modelo, representa un conjunto de reglas, y se forman distintas subpoblaciones. Si el número de reglas en el conjunto candidato es n, entonces habrá n sub-poblaciones y la *i-ésima* subpoblación será evolucionada para optimizar el conjunto de reglas que contiene i reglas. Al final de la evolución, la salida de cada sub-población que representa el mejor conjunto de reglas candidato, compiten (de acuerdo al valor de exactitud que logran en la clasificación) con los mejores conjuntos de reglas generadas por otros grupos de población para obtener el conjunto óptimo de reglas.

El sistema descrito en [79] tiene por objeto solucionar el problema del balanceo en los datos. Se dice que los datos no están balanceados cuando algunas clases tienen un mayor número de elementos que otros. El problema con los datos no balanceados surge porque los algoritmos de aprendizaje tienden a pasar por alto las clases menos frecuentes, prestando atención sólo a las más frecuentes. El planteamiento seguido en este trabajo consiste en la generación de reglas para la clase menos frecuente. Cada individuo en la población puede representar un conjunto de reglas unidas por conjunciones, cuando un operador lógico *OR* se encuentra en un árbol, se separa un antecedente de otro; de este modo, las reglas se extraen de todos los individuos en la población. Se mantiene un repositorio con el fin de construir el clasificador final, donde cada regla extraída se añade al repositorio si su exactitud es superior a un determinado umbral y si es diferente a las reglas que ya están en el repositorio. Las condiciones pueden comparar el valor de dos atributos, por lo que se trata de clasificadores lineales multivariantes.

Clasificación multiclase

Existen varias formas de llevar a cabo esta clasificación. En [18], se desarrolla una única regla en cada ejecución del sistema, ejecutándose el sistema n veces para un problema de clasificación con n clases. De esta manera, el clasificador final tiene

una única regla por clase. La función de evaluación combina tres términos, dos de ellos (especificidad y sensibilidad) son utilizados para medir la calidad de la regla, y el tercero mide su complejidad. Las condiciones de las reglas son lineales univariantes. Un enfoque muy similar se sigue en [47].

En el mismo sentido que las referencias anteriores, el sistema presentado en [189] aborda la clasificación multiclase realizando diferentes ejecuciones para cada una de las clases que hay que distinguir. Sin embargo, varias diferencias se pueden observar con respecto a los sistemas descritos anteriormente. Cada individuo representa una regla y se emplea un mecanismo de nichos (token competition). Un conjunto de reglas, todas prediciendo la misma clase se obtiene al final del proceso evolutivo, por lo que el clasificador final puede tener varias reglas para cada una de las clases. Condiciones simples univariantes se utilizan en este trabajo y la exactitud es utilizada como medida de evaluación.

En el sistema descrito en [154] cada individuo representa una única regla y el clasificador final estará formado por varias clases. Para la evaluación de los individuos se utilizan dos medidas, la sensibilidad y la especificidad. Las condiciones del antecedente de las reglas son representadas por comparaciones simples univariantes, permitiendo la generación de varias reglas por clase. Un sistema de características similares pero que evoluciona reglas difusas, es propuesto en [172].

En [26] el planteamiento es diferente, cada individuo codifica una regla para una clase, realizándose una ejecución independiente para cada clase que se quiere diferenciar. Cuando una ejecución finaliza, los individuos con el valor de aptitud más alto se almacenan. La función de evaluación es una combinación de su precisión y su soporte, sesgando la búsqueda hacia soluciones comprensibles por medio del proceso de selección que clasifican los individuos probabilísticamente de acuerdo con uno de los tres criterios posibles: el soporte, la comprensibilidad o la precisión. Cada criterio tiene una probabilidad asignada y los valores de dichas probabilidades son auto-adaptativos con el proceso evolutivo. En [27] emplean un sistema híbrido que combina el GA y la GP. Cada individuo representa una regla, y en una misma iteración se mantienen individuos que clasifican clases diferentes. La función de evaluación toma en cuenta tres condiciones: el soporte, la confianza y el tamaño de la regla. Este sistema permite el uso de las condiciones que comparan el

valor de dos atributos; de esta manera, clasificadores lineales multivariantes pueden obtenerse.

En [214] se generan clasificadores lineales multivariantes que contienen varias reglas por clase y evolucionan individuos que contienen una regla para cada clase, con lo que el clasificador final es representado por un único individuo. Utiliza un método de nichos (token competition) y la función de evaluación emplea los conceptos de soporte y confianza. En [144] se sigue un proceso de dos etapas que combina los paradigmas de EP y GP. En la primera fase, un método de EP se utiliza para desarrollar una red bayesiana, lo que representa una estructura general de las relaciones entre los atributos. Una red bayesiana proporciona los conocimientos en sí, y se utiliza como base para definir la gramática que se utilizará por el algoritmo de GP en la segunda etapa. Este sistema es extendido en [213], cuando un GA se añade al sistema de EP y GP. El GA es utilizado para discretizar los atributos continuos. Propuestas similares pueden encontrarse en [12] y [184]. El sistema descrito en [84] emplea una función de evaluación basada en la exactitud de la clasificación con una penalización para el tamaño. Cada individuo codifica un conjunto de reglas, donde se permiten operaciones aritméticas y la comparaciones entre atributos y combinaciones de los mismos, obteniéndose clasificadores lineales multivariantes.

El sistema híbrido GA-P [101] se aplica a la evolución de reglas difusas en [78]. Se emplea un enfoque co-evolutivo donde el algoritmo de GP evoluciona una población de conjuntos de reglas, mientras que GA evoluciona los parámetros que definen las funciones miembros de los atributos utilizados en el antecedente de las reglas. Cada individuo codifica un clasificador completo, en la que sólo se permiten las condiciones univariante, este clasificador puede contener varias reglas por clase.

El sistema propuesto en [132] evoluciona reglas de clasificación difusas para problemas multiclase. En este trabajo los atributos numéricos son fuzificados, y se sigue un enfoque coevolutivo, en el que un algoritmo de GP y una ES [15] se ejecutan simultáneamente de forma cooperativa. Cada individuo de GP codifica un conjunto de reglas y cada uno de los de la ES representa las funciones de pertenencia correspondientes los atributos continuos fuzificados. A continuación, nos centramos en el algoritmo de GP. Cada individuo en la población representa un conjunto de reglas, pero todas las reglas de cada individuo predicen la misma clase y, por tanto, el sistema debe ser ejecutado una vez por cada una de las clases que hay

que distinguir. Condiciones simples univariantes se usan en los antecedentes de las reglas. La función de evaluación mide la exactitud de acuerdo a la sensibilidad y especificidad. El tamaño no está incluido en la evaluación, pero se considera en la selección de las individuos. Así, cuando los individuos son seleccionados para el proceso de recombinación de la siguiente generación, los individuos con mayor aptitud son seleccionados, pero cuando varios individuos tienen igual aptitud, el más pequeño tiene mayor preferencia.

El sistema propuesto en [64] evoluciona individuos que codifican varias reglas que predicen la misma clase. El clasificador final, por tanto, estará formado por un conjunto de individuos. En el antecedente de las reglas se usa condiciones simples univariantes. La función de evaluación se basa en la exactitud. Un enfoque similar para la evolución de reglas difusas, se emplea en [194]. En [19], un individuo puede contener múltiples reglas de clasificación, sujetas a la restricción de que todas sus reglas tienen el mismo consecuente. La población estará compuesta de un conjunto de individuos de este tipo. Cuando termina el proceso evolutivo, el mejor individuo para cada clase se utiliza para formar el clasificador final, por lo tanto, pueden existir varias reglas para cada clase con condiciones simples univariantes. La función de evaluación combina tres métricas, la sensibilidad y la especificidad se utilizan para evaluar la exactitud de la regla, y una medida del tamaño de las reglas se emplea para evaluar la complejidad.

3.4. Algoritmos evolutivos multiobjetivo

3.4.1. Introducción

Los problemas reales usualmente requieren la búsqueda de soluciones que satisfagan de forma simultánea múltiples criterios de desempeño u objetivos los cuales pueden (y normalmente ocurre) ser contradictorios. Cuando la formulación matemática de un problema se plantea como la optimización de varios objetivos a la vez que son contradictorios, el problema se denomina optimización multiobjetivo. La resolución de este problema no es un único punto, como en el caso de la optimización global, sino un conjunto de puntos que todos ellos serán óptimos.

Desde que nos levantamos por la mañana, las personas estamos resolviendo problemas multiobjetivo, desde la proporción de azúcar, leche y café de nuestro desayuno para que nos aporte un buen comienzo del día, hasta la cantidad de dinero a invertir en fondos de inversión, la cantidad que se puede gastar sin que pasemos necesidades, el coche que nos compramos teniendo en cuenta relaciones de calidad, precio, consumo, comodidad, capacidad, etc. Resulta evidente que si todos nosotros diésemos la misma importancia a estos objetivos, todos tendríamos el mismo coche, la misma casa, comeríamos lo mismo, ahorraríamos el mismo dinero, tendríamos el mismo trabajo, etc. Sin embargo es obvio que, en la vida real, esto no sucede. Pero entonces cabe preguntarse, ¿es que mi coche es peor que el del vecino? o ¿es mi casa menos confortable?... Bueno, eso depende de lo que consideremos por *peor* o por *confortable*, es decir, depende de qué parámetro o parámetros consideremos más importantes.

En este sentido se puede decir que existe un conjunto de soluciones óptimas, todas y cada una de ellas igual de eficientes en el cómputo global, pero todas formadas por parámetros de distinto valor. De este modo podríamos escoger entre un coche que sea más barato aunque gaste más combustible y uno que cueste un poco más pero que el consumo de carburante sea menor. La decisión depende del uso que vayamos a darle a nuestro automóvil, y por tanto de la persona que lo va a usar. Esta es seguramente la razón de que no todos seamos iguales, con las mismas conductas, las mismas pautas ni los mismos bienes materiales. Todos hemos elegido de entre una serie de opciones y todos estamos convencidos de haber realizado la mejor elección según nuestras posibilidades y necesidades.

Por todo lo anterior, se puede decir que toda optimización verdadera debe ser multiobjetivo. Los problemas reales por lo general requieren que más de una función objetivo sea optimizada. Habitualmente estas funciones no pueden ser combinadas en una sola. Consideremos por ejemplo el diseño de un aparato electrónico. Uno podría querer maximizar el desempeño del aparato dadas ciertas características electrónicas, minimizar el tiempo promedio de fallos del aparato, y minimizar el costo de producir el aparato. Estos objetivos no pueden ser considerados en una sola función porque son inconmensurables, es decir, miden propiedades que no pueden ser relacionadas entre sí directamente. Un diseño particular podría tener muy alto desempeño pero ser de muy alto costo; otro diseño podría tener bajo costo pero

a cambio reducir el desempeño del aparato y reducir el tiempo promedio entre fallos. No es posible decir que un diseño es mejor que otro sin alguna información acerca de las preferencias de los usuarios típicos del aparato. Pero encontramos problemas reales en muchas otras áreas que necesitan considera diferentes objetivos contrapuestos a optimizar. Estos problemas, llamados problemas de optimización multi-objetivo (*Multiobjective Optimization Problems*, MOP) no los encontramos en campos que incluyen la logística [24], manejo del entorno [107], aéro dinámica [14], química [137], inteligencia artificial [55], energías renovables [106], medicina [53], control óptimo [134], ingeniería química [89] y muchas más. Cualquier problema que requiera considerar varios objetivos, en los que cuando uno de ellos se optimiza el otro se empeora serían de este tipo, maximizar beneficios y minimizar costes de un producto, maximizar el desarrollo y minimizar el consumo de gasolina de un vehículo, minimizar el peso y maximizar la fortaleza de un componente serían ejemplos de optimización multi-objetivo. Así pues, en todos estos problemas tomamos conciencia de que existe un conjunto de soluciones que son todas óptimas, en el sentido de inmejorables, pero que dan mayor o menor importancia a unos objetivos frente a los otros. Si fuésemos capaces de disponer de este conjunto de soluciones factibles y óptimas podríamos decidir sobre cuál nos conviene más estando seguros de que no existe una solución mejor de acuerdo a nuestros requerimientos concretos.

3.4.2. Conceptos básicos y terminología

Mientras que en optimización monobjetivo se busca un vector de decisión n-dimensional que optimice una función escalar, en optimización multiobjetivo se intenta encontrar un vector que optimice una función vectorial cuyos elementos representan las distintas funciones objetivo. A continuación definiremos que es un problema de optimización multiobjetivo (Multiobjective Optimization Problem, MOP), así como varios conceptos esenciales en la optimización multiobjetivo.

Definición1. Problema de optimización multiobjetivo: el problema de optimización multiobjetivo puede formularse de forma general como el problema de encontrar el vector $\vec{x^*} = [x_1^*, x_2^*, \cdots, x_n^*]^T$ que satisfaga las m restricciones de desigualdad:

$$g_i(\vec{x}) \geq 0 \quad i = 1, 2, \ldots, m$$

Las p restricciones de igualdad

$$h_i(\vec{x}) = 0 \quad i = 1, 2, \ldots, p$$

y que optimice

$$\vec{f_i}(\vec{x}) = [f_1(\vec{x}), f_2(\vec{x}), \ldots, f_k(\vec{x})]^T$$

La solución de un MOP minimiza o maximiza (según corresponda) los componentes del vector $\vec{f}(x)$. Por tanto, el problema tiene k objetivos y las funciones $f(\cdot) : \Omega \rightarrow A$ es la correspondencia entre el espacio de búsqueda Ω y el espacio de las funciones objetivo A. De la definición de MOP se infiere que la solución del problema no es única ya que los diversos objetivos están en conflicto de forma que la optimización de uno de ellos lleva a un decremento del valor del otro. De hecho, la optimización global de un problema multiobjetivo es un problema NP-completo.

Sea, por ejemplo, el problema de minimización de una función de dos dimensiones $f(x) \in A \subseteq R^k$. Los cuatros puntos A, B, C y D dibujados en la figura 3.4.2 son cuatro posibles soluciones del problema. A la hora de elegir cual de ellos es mejor sólo podemos afirmar que el punto D es peor que el B por ser mayor que él en todos los componentes, pero los puntos A, B y C no son comparables. Esta especial característica de la optimización multiobjetivo es lo que lleva al concepto de optimalidad de Pareto.

Definición2. Optimalidad de Pareto: decimos que un punto $\vec{x^*} \in \Omega$ es un óptimo de Pareto si para toda $\vec{x} \in \Omega$ e $I = \{1, 2, \cdots, k\}$ ya sea,

$$\forall_{i \in I}(f_i(\vec{x}) = f_i(\vec{x^*}))$$

o hay al menos una $i \in I$ tal que

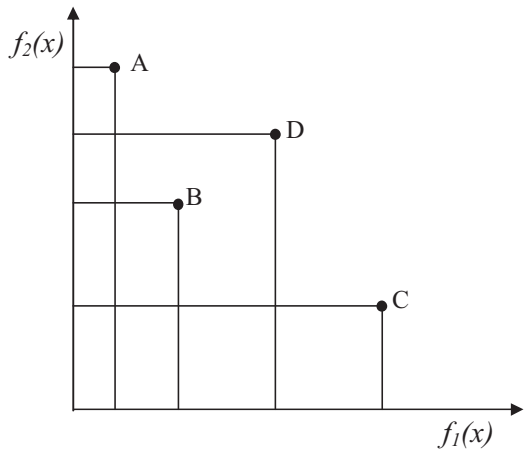


Figura 3.3: Problema de optimización de una función de dos dimensiones.

$$f_i(\vec{x}) > f_i(\vec{x}^*)$$

Definición3. Dominancia de Pareto: en un problema de minimización, un vector $\vec{u} = (u_1, u_2, \ldots, u_k) \in A \subseteq \Re^n$ domina a otro $\vec{v} = (v_1, v_2, \ldots, v_k) \in A \subseteq R^n$ (denotado meidante $\vec{u} \preceq \vec{v}$ si y sólo si u es parcialmente menor a v, es decir, $\forall i \in 1, \ldots, k,\ u_i \leq v_i \wedge \exists i \in 1, \ldots, k : u_i < v_i$.

Definición4. Conjunto de óptimos de Pareto: para un problema multiobjetivo dado $\vec{f}(x)$, el conjunto de óptimos de Pareto

$$(P^*) := x \in \Omega | \neg \exists x' \in \Omega\ \vec{f}(x') \preceq \vec{f}(x)$$

Definición5. Frente óptimo de Pareto: para un problema multiobjetivo dado $\vec{f}(x)$ y un conjunto de óptimos de Pareto P^*, el frente óptimo de Pareto $(ParetoOptimalFront, POF^*)$ se define como:

$$POF^* := \vec{u} = \vec{f} = (f_1(x), \ldots, f_k(x)) | x \in P^*$$

Definición6. Mínimo global de un MOP: dado un vector de funciones $f(\cdot) : \Omega \subseteq R^k \longrightarrow R^n,\ \Omega \neq 0,\ k \geq 2$ el conjunto $PF^* : f(x^*)$ es llamado mínimo global si y sólo si

$$\forall x \in \Omega : f(x^*) \prec f(x),$$

donde $x^* \in \Omega$ es llamado conjunto de soluciones de mínimo global, $f(\cdot)$ es el vector de funciones objetivos y Ω el espacio de búsqueda.

A raíz de estas definiciones se deduce que el mínimo global de un MOP es la frontera del Pareto de dicho problema y las soluciones de mínimo global forman el conjunto de óptimos de Pareto. Un problema de optimización de Pareto donde el espacio de búsqueda sea $\Re^k$ tiene, en general, infinitas soluciones. Se puede restringir el espacio de búsqueda a un número finito de elementos, por ejemplo, en el problema planteado en la figura 3.4.2, siendo el espacio de búsqueda $\Omega = \{x_A, x_B, x_C, x_D\}$ el conjunto de óptimos del Pareto es $P^* = x_A, x_B, x_C$, y el frente de Pareto $FP^* = \{f(xA), f(xB), f(xC)\}$. Esto se deduce debido a que $f(x_B)$ domina a $f(x_D)$ con lo que este elemento (x_D) no es un óptimo de Pareto.

Definición7. Dominancia débil y fuerte: en conjunto con el concepto de optimalidad de Pareto, existe un par de conceptos relacionados que han sido ampliamente difundidos. A dichos conceptos se les conoce como dominancia débil de Pareto y dominancia fuerte de Pareto. Un vector es un óptimo de Pareto débil si no existe otro vector para el cual todos sus componentes en el espacio de las funciones objetivo sean mejores. Formalmente hablando, lo podemos definir como sigue:

Definición7.1 No Dominancia débil: una solucion $\vec{x}^* \in \Omega$ es una solución débilmente no dominada si no existe otra solución $\vec{x} \in \Omega$ tal que $f_i(\vec{x}) < f_i(\vec{x}^*)$, para $i = 1, 2, \ldots, k$.

Definición7.2 No Dominancia fuerte: una solucion $\vec{x}^* \in \Omega$ es una solución fuertemente no dominada si no existe otra solución $\vec{x} \in \Omega$ tal que $f_i(\vec{x}) \leq f_i(\vec{x}^*)$, para $i = 1, 2, \ldots, k$ y existe al menos un valor j para el cual $f_j(\vec{x} < f_j(\vec{x}^*)$.

Si un vector es fuertemente no dominado, entonces también es débilmente no dominado, pero lo contrario no necesariamente es cierto en todas las ocasiones.

3.4.3. Clasificación de los EAs multiobjetivo

Los EAs son una de las técnicas más empleadas para la resolución de los MOP. Estos deben conseguir dirigir a una población inicial de individuos del espacio de búsqueda hacia el conjunto de óptimos del Pareto. En general, los EAs que emplean optimización multi-objetivo son conocidos como MOEAs (*Multi-Objective Evolutionary Algorithms*).

Encontramos distintas formas de realizar una taxonomía de los MOEAs. A continuación se detallan algunas de ellas:

- Según [70] pueden considerarse, en general, dos tipos principales de algoritmos evolutivos multiobjetivo:

  1. Los algoritmos que no incorporan el concepto de óptimo del Pareto en el mecanismo de selección del Algoritmo Evolutivo. Por ejemplo, los que usan funciones agregativas lineales.
  2. Los algoritmos que jerarquizan a la población de acuerdo a si un individuo es no dominado o no (usando el concepto de óptimo de Pareto). Ejemplos: MOGA, NSGA, NPGA, ...

- En [195] se dividen estos algoritmos en tres clases, los algoritmos a priori, progresivos y a posteriori.

  1. Se consideran algoritmos a priori a aquellos que suponen conocimiento sobre la forma de la solución del problema.
  2. Se consideran algoritmos a posteriori a los que tienen como objetivo la obtención del conjunto de óptimos del Pareto, sin presuponer nada, eligiendo luego el elemento que se desee.
  3. Los algoritmos progresivos cambian el objetivo de forma interactiva en el curso de las generaciones.

- Otra clasificación que se puede llevar a cabo es la que considera dos generaciones de Algoritmos Evolutivos Multiobjetivo:

  1. Primera Generación: caracterizada por el uso de jerarquización de Pareto y nichos. Algoritmos relativamente simples. También se produjeron enfoques más rudimentarios (p.ej., funciones agregativas lineales).

2. Segunda Generación: se introduce el concepto de elitismo en dos formas principales: usando selección $(\mu+\lambda)$ y usando una población secundaria.

Tras haber gozado de mucho éxito, los algoritmos de primera generación han comenzado a caer en desuso (NSGA, NPGA, MOGA y VEGA) quedando reducidos a emplearlos en algunos dominios (por ejemplo, la optimización combinatoria) con relativo éxito. Desde finales de 1990 los Algoritmos Evolutivos Multiobjetivo que usan elitismo reemplazaron a los anteriores debido a la inminente mejora que producían, (por ejemplo, SPEA, SPEA2, NSGA2, MOGLS, PESA, PESA2, ...).

De acuerdo a la clasificación dada por [70], vamos a abordar los distintos métodos que nos podemos encontrar dentro de EAs Multiobjetivo.

3.4.3.1. Métodos clásicos de resolución de problemas multiobjetivo (no crean Pareto)

Los métodos clásicos de resolución de un problema de optimización multiobjetivo (MOP) tienden a convertir dicho problema en otro de optimización global.

Cuando es factible combinar los objetivos de un problema de manera adecuada, es posible considerar un único objetivo a optimizar. En este caso, para obtener la solución del problema basta con encontrar el mínimo o el máximo de una única función que resume todos los objetivos que se desean optimizar. Sin embargo, lo usual es que no se conozca la manera óptima de combinar los diferentes objetivos o sea inadecuado, cuando no imposible hacerlo. En este caso, se dice que el problema es un Problema de Optimización Multiobjetivo.

Veremos algunos ejemplos de este tipo de métodos.

Método de Optimización de suma con pesos

En este método el MOP es convertido en un problema de minimización de una suma convexa de funciones objetivo.

$$min_x \ f_{sum} = \sum_{i=1}^{n} \omega_i \cdot f_i(x) \ x \in \Omega$$

donde $\omega_i \geq 0$, $\sum_{i=1}^{n} \omega_i = 1$. La solución de este problema es un punto del frente de Pareto, en función de los pesos, $\omega_i \ i = 1, ..., n$ elegidos. Este método sólo es útil cuando el frente de Pareto es convexo, siendo complicada la elección de pesos a priori.

Método de optimización de máximo con pesos

En este método se debe efectuar una optimización de $min - max$. Se define un vector formado por los términos del vector de funciones objetivo multiplicados por diferentes pesos. El MOP se convierte en la minimización con respecto al vector x del máximo valor del vector definido, para unos pesos dados a priori.

$$min - max_{x \in \Omega} \ f_1(x) \cdot \omega_1, ..., f_n(c) \cdot \omega_n.$$

Otra variación de este método se produce al fijar un vector constante que se resta al vector de funciones objetivos, de forma que la formulación del problema queda

$$min - max_{x \in \omega} (f_1(x) - \gamma_1) \cdot \omega_1, ..., (f_n(x) - \gamma_n) \cdot \omega_n.$$

Método de permutación $\in$

Este método reduce el MOP a un problema de optimización para un simple objetivo con restricciones

$$\begin{aligned} & min_x \ f_k(x) \\ & s.a. f_j(x) \leq e_j \ \forall j \neq k \\ & x \in \Omega \end{aligned}$$

Este método sirve para un frente de Pareto convexo o no convexo.

Aproximación a un punto utópico

En este método se considera un punto utópico (f^*) al que se desea llegar con la función objetivo. El problema se replantea como

$$min_{x\in\Omega} \ \|f^* - f(x)\|$$

donde se elige una métrica, la euclidea por ejemplo, como medida. Tiene el inconveniente de tener que definir a priori el punto utópico.

Soluciones óptimas lexicográficas

Este método convierte el problema de optimización de Pareto, donde todas las componentes del vector de funciones $f(\cdot)$ tienen la misma importancia, en un problema donde las componentes de dicho vector tienen una importancia decreciente. Así, el problema de optimización lexicográfica se formula como una secuencia de n minimizaciones P_i:

$$\begin{aligned}
&(P_1)\ min_{x\in\Omega}\ f_1(x) = \alpha_1 \\
&(P_2)\ min_{x\in\Omega}\ \{f_2(x)|\ f_1(x) \leq \alpha_1\} = \alpha_1 \\
&\qquad \vdots \\
&(P_n)\ min_{x\in\Omega}\ \{f_n(x)\ |\ f_1(x) \leq \alpha_1, ..., f_{n-1}(x) \leq \alpha_{n-1}\}
\end{aligned}$$

La formulación del problema (P_n) puede escribirse como:

$$\begin{aligned}
&min_{x\in\Omega}\ f_n(x) \\
&sujetoa f_k(x) \leq \alpha_k\ k = 1, \ldots, n-1.
\end{aligned}$$

La formulación de este problema rompe el espíritu de un verdadero MOP y sólo es aplicable a los casos donde el orden de prioridad de los objetivos esté claro.

3.4.3.2. Métodos que crean conjunto Pareto

En problemas de optimización multiobjetivo con objetivos contradictorios no siempre existe una única solución que pueda ser considerada como la mejor, sino un conjunto de soluciones que representan los mejores compromisos entre los distintos criterios, y sería conveniente obtener cada una de ellos (conjunto Pareto).

Varias son las causas que hacen que los métodos clásicos no sean eficaces para hallar el conjunto de óptimos de Pareto de un MOP.

- Estos algoritmos sólo dan un punto del frente de Pareto como solución, siendo necesaria la repetición del algoritmo un gran número de veces para obtener todas las soluciones.
- Muchos de estos algoritmos necesitan un conocimiento previo del problema para ser viables. Es el caso del valor de las funciones peso o del punto utópico.
- Alguno de estos algoritmos son muy sensibles a la forma del frente de Pareto.
- Estos algoritmos no son validos en problemas con incertidumbre o problemas estocásticos.
- Los algoritmos de optimización clásicos no son válidos para resolver problemas de un sólo objetivo con un espacio de búsqueda discreto, luego tampoco lo serán para resolver MOP con espacio de búsqueda discreto.

Estas razones son algunas de las causas que han potenciado el desarrollo de Algoritmos Evolutivos para el tratamiento de objetivos contrapuestos, proporcionando el conjunto Pareto.

Existe una gran cantidad de métodos elitista, debido a que sus resultados mejoran con creces a los métodos previos que no introducían el elitismo en su proceso evolutivo. Tres modelos ampliamente utilizados son NSGA2 [51], SPEA2 [245] y MOGLS [104], sus principales características serán detalladas en la sección 5 al tratarse de los modelos en los que están basados nuestras propuestas. Otros modelos incluidos en esta clasificación son PAES [112] que emplea una estrategia evolutiva como proceso evolutivo, PESA2 [41] se trata de un algoritmo evolutivo que introduce

un proceso de selección basada en regiones, manteniendo una población externa de soluciones no dominadas, $\in$-MOEA [52] se trata de un algoritmo estacionario que usa el concepto de ϵ-dominancia, se basa en co-evolucionar durante el proceso evolutivo tanto una población de algoritmos evolutivos como una población externa, los individuos se van seleccionando de la población y de la población externa para generar nuevas soluciones, parEGO [111] se trata de una adaptación del algoritmo de optimización mono-objetivo *Efficient Global Optimization* (EGO). Entre sus características más relevantes se encuentra que funciona con muy pocas evaluaciones, pero presenta la limitación de que sólo puede usarse en problemas de baja dimensionalidad. Estas propuestas aunque son más actuales que las seleccionadas, no gozan de las misma popularidad que las anteriores.

3.4.4. Métricas para la comparación de algoritmos multiobjetivo

En optimización multiobjetivo no existe un criterio único que permita establecer con facilidad si un conjunto de aproximación al frente óptimo del Pareto (*Pareto Optimal Front*, POF) es mejor que otro, debido a que se pueden considerar varias medidas de calidad. Un POF debería cumplir los siguientes requisitos:

1. Estar compuesto por un número alto de soluciones distintas.
2. Minimizar la distancia existente con respecto al frente del Pareto óptimo para el problema en cuestión (es decir, el conjunto real de soluciones de dicho problema, el cual es a veces desconocido).
3. Presentar una buena distribución de las soluciones que lo componen (en el mejor de los casos, una distribución uniforme). La evaluación de este criterio puede basarse en el uso de una métrica.
4. Maximizar la extensión del frente no dominado (Pareto) obtenido, es decir, para cada objetivo, las soluciones no dominadas deben cubrir la mayor amplitud de valores posible.

Para comparar los resultados obtenidos por distintos EAs se han desarrollado varias métricas de desempeño, las cuales buscan capturar las características que hacen a

una aproximación del conjunto Pareto-óptimo mejor que otra en algún criterio. Los tres valores fundamentales que miden las métricas en la actualidad son:

1. Minimizar la distancia del frente de Pareto producido por el algoritmo con respecto al frente verdadero (suponiendo que lo conocemos).
2. Maximizar la distribución de soluciones obtenidas, de manera que podamos tener una distribución de vectores tan uniforme como sea posible.
3. Maximizar la cantidad de elementos del conjunto de óptimos de Pareto generados.

El problema fundamental de las métricas es que los tres valores antes mencionados deben combinarse de diferente manera para poder establecer el desempeño de un algoritmo de manera cuantitativa. Sin embargo, dichas combinaciones no son más que combinaciones lineales de pesos. Irónicamente, el problema de las métricas resulta ser también de naturaleza multiobjetivo.

Numerosas métricas han sido empleadas en las diferentes investigaciones. Desarrolladores iniciales de benchmark para MOEA incluyen diferentes métricas [195; 50; 243]. Pronto, aparecieron el uso de métricas individuales que evalúan diferentes características de un POF [168; 181; 69; 44; 38]. Otros trabajos más recientes extendieron métricas y añadieron unas nuevas, con descripciones más formales [198; 113; 40; 114]. Resúmenes de métricas interesantes para utilizarse en la comparación estadísticas de los MOEAs las encontramos en [198; 40; 250; 114; 247]. A continuación, comentaremos algunas de las medidas de comparación más empleas caracterizadas como *métricas o indicadores de calidad*, existiendo en las referencias mostradas muchas otras posibilidades y pudiéndose seleccionar un conjunto de ellas para evaluar el desempeño de las técnicas multi-objetivo dependiendo del contexto en el que estamos trabajando.

Los indicadores que se van a mostrar en general reflejan un valor de cardinalidad relacionado con el número de puntos de soluciones, un valor real de proximidad para el frente de Pareto o otro conjunto de puntos, o valores reales para medir las características del POF (diversidad, uniformidad, ...).

1. Ratio de Error (*Error Ratio*, ER): esta métrica informa del número de vectores en el $POF_{conocido}$ que no son miembros del $POF_{verdadero}$ [195; 197]. Esta

métrica requiere conocer el $POF_{verdadero}$ y matemáticamente es representado por la ecuación:

$$ER \triangleq \frac{\sum_{i=1}^{|POF_{conocido}|} e_i}{|POF_{conocido}|} \tag{3.4}$$

donde $|POF_{conocido}|$ es el número de soluciones en el $POF_{conocido}$ y e_i es cero cuando el *iésimo* vector de $POF_{conocido}$ es un elemento del $POF_{verdadero}$ o e_i es uno *iésimo* vector de $POF_{conocido}$ no es un elemento del $POF_{verdadero}$ [40].

De este modo si $ER = 0$, significará que $POF_{conocido}$ es el mismo que el $POF_{verdadero}$. Por el contrario, si $ER = 1$, indicará que ninguno de los puntos en $POF_{conocido}$ es igual al del $POF_{verdadero}$.

2. Distancia Generacional (*Generational Distancia*, GD): esta métrica determina la distancia a la que se encuentra el $POF_{conocido}$ con respecto al $POF_{verdadero}$ [40; 196; 197]. Esta métrica también requiere conocer el $POF_{verdadero}$. Matemáticamente se define por:

$$GD \triangleq \frac{(\sum_{i=1}^{n} d_i^p)^{\frac{1}{p}}}{|PF_{conocido}|} \tag{3.5}$$

donde $|POF_{conocido}|$ es el número de soluciones en el $POF_{conocido}$, $p = 2$, y d_i es la distancia euclídea entre cada miembro de i, del $POF_{conocido}$ y el miembro más cercano en $POF_{verdadero}$ a dicho miembro. Cuando $GD = 0$ significa que $POF_{conocido}$ es el mismo que el $POF_{verdadero}$.

3. Hiper-área (*Hyperarea*, HA) y Radio del hiper-área (*Hyperarea Ratio*, HR) , el hiper-área mide el área de cubrimiento del $POF_{conocido}$ con respecto al espacio objetivo [40; 246] para un problema multi-objetivo con dos objetivos. Es igual a la suma de todas las áreas rectangulares limitadas por algún punto de referencia y las funciones objetivo $(f_1(x), f_2(x))$. Matemáticamente se expresa como:

$$HA \triangleq \left\{ \bigcup_i area_i | vec_i \in PF_{conocido} \right\} \tag{3.6}$$

donde vec_i es un vector no dominado del $POF_{conocido}$ y $area_i$ es el área entre el origen y el vector vec_i. Si el $POF_{conocido}$ no es convexo los resultados pueden no entenderse bien [196]. Se asume que el punto de referencia para el hiper-área es el valor mínimo de cada objetivo. El hiper-área y el hiper-volumen (HV) son similares, excepto que el hiper-volumen puede usarse con dos dimensiones.

También se comenta el radio del hipe-área que se define como.

$$HA \stackrel{\triangle}{=} \frac{H_1}{H_2} \tag{3.7}$$

donde HA_1 es el hiper-área del $POF_{conocido}$ y HA_2 es el hiper-área del $POF_{verdadero}$.

4. Espaciado (*Spacing*, S): esta métrica describe numéricamente la extensión del los vectores en el $POF_{conocido}$ [40; 168]. Esta métrica mide la varianza de la distancia de las soluciones vecinas en el $POF_{conocido}$. Matemáticamente esta métrica se puede definir de la siguiente manera:

$$HA \stackrel{\triangle}{=} \sqrt{\frac{1}{POF_{conocido} - 1} \cdot \sum_{i=1}^{|POF_{conocido}|} (\bar{d} - d_i)^2} \tag{3.8}$$

$$d_i = min_j(|f_1^i(x) - f_1^j(x)| + |f_2^i(x) - f_2^j(x)|) \tag{3.9}$$

donde $d_i = min_j(|f_1^i(x) - f_1^j(x)| + |f_2^i(x) - f_2^j(x)|)$, $i, j = 1, \ldots, n, \bar{d}$ es la media de todas los d_i, y n es el número de soluciones en el $POF_{conocido}$. Cuando $S = 0$, todas las soluciones están igualmente espaciadas.

5. ϵ-indicador: este indicador dada dos aproximaciones del conjunto, A y B, mide la cantidad más pequeña, ϵ, que debe usarse para que cada punto del conjunto aproximado de A sea cubierto por los de B. Formalmente esta medida se define como:

   Dado $A, B \subseteq X$. Entonces, el ϵ-indicador $I_\epsilon(A, B)$ se define como el mínimo $\epsilon \in \Re$ tal que cualquier solución $b \in B$ es ϵ-dominada por al menos una solución $a \in A$:

$$I_\epsilon(A,B) = min\{\epsilon \in \Re \mid \forall b \in B \exists a \in A : a \succ_\epsilon b\} \tag{3.10}$$

Cuando $I_\epsilon(A,B) < 1$, todas las soluciones en B son dominadas por una solución en A. Si $I_\epsilon(A,B) = 1$ y $I_\epsilon(B,A) = 1$ entonces A y B representan la misma aproximación del frente de pareto. Si $I_\epsilon(A,B) > 1$ y $I_\epsilon(B,A) > 1$m entonces A y B son incomparables (ambos contienen soluciones no dominadas por el otro conjunto).

6. Cubrimiento de dos conjuntos (*Two Set Coverage*, CS): Zitzler et al. [244] propone una métrica para comparar el cubrimiento relativo de dos conjuntos. Dados dos conjuntos de soluciones no dominadas (Paretos) $X^{'}$ y $X^{''}$. La medida CS se define como la correspondencia de pares de conjuntos $(X^{'}, X^{''})$ que nos determina el grado de dominancia de $X^{'}$ sobre $X^{''}$:

$$CS(X^{'}, X^{''}) \triangleq \frac{|a'' \in X''; \exists a' \in X' : a' \succ= a''|}{|X''|} \tag{3.11}$$

Si todo los puntos en $X^{'}$ dominan o son igual a los puntos en $X^{''}$, entonces por definición $CS = 1$. $CS = 0$ implica lo opuesto. En general, $C(X^{'}, X^{''})$ y $C(X^{''}, X^{'})$ consideran un conjunto de intersección que no es vacío. Esta métrica es fácil de calcular y proporciona una comparación basada en las dominancias que se producen entre las soluciones de dos conjuntos no dominados.

4

Modelo basado en Programación Genética Gramatical para Aprendizaje Multi-Instancia

4.1. Introducción

Después del estudio realizado sobre las propuestas desarrolladas para resolver problemas de aprendizaje con múltiples instancias, resulta, cuanto menos sorprendente, que no existan referencias sobre la introducción de los EAs a este paradigma. Los EAs han demostrado su buen funcionamiento en tareas de aprendizaje supervisado tradicional, con lo que pensar en su adaptación al aprendizaje basado en múltiples instancias, a priori parece una opción acertada.

En principio, de acuerdo al teorema de *no-free-lunch* (NFL) [212] que determina que no puede existir ningún algoritmo que resuelva todos los problemas, en general, de forma superior a cualquier otro, no tendría sentido plantearnos si los EAs son superiores o inferiores a cualquier otra aproximación. Sin embargo, si que podemos afirmar que se comportan mejor que otros métodos con respecto a problemas específicos y bajo determinadas circunstancias. Existen infinidad de estudios

científicos que nos permiten corroborar que los EAs son más eficientes resolviendo problemas discontinuos, no diferenciables, multimodales, con ruido y cualquier tipo de superficie de búsqueda no convencional que otros métodos clásicos de optimización. Por el contrario, también se ha demostrado que su efectividad disminuye al afrontar problemas más simples para los que se han desarrollado algoritmos específicos en su resolución. Debido a que los problemas a los que nos enfrentamos se caracterizan por ser complejos y por presentar información incompleta. pensar en su adaptación para el aprendizaje con múltiples instancias parece una buena idea.

Entre las implementaciones exitosas de la CE, GP mantiene una posición importante, debido a características tan valiosas como: la flexibilidad de la representación de sus soluciones, el hecho de que no es necesario un conocimiento previo acerca de la distribución estadística de los datos, los datos en su forma original se pueden usar para operar directamente sobre ellos, permite detectar relaciones desconocidas que existen entre los diferentes datos y permite especificar las relaciones entre los datos mediante una expresión matemática. Sin duda, la principal ventaja de esta técnica sobre otras técnicas de CE es que posibilita la creación de individuos de tamaño variable, ya que el proceso evolutivo y el hecho de que la base de conocimiento sea codificada como individuos de la población es similar al resto de paradigmas. Esta flexibilidad introducida en la representación puede resultar valiosa cuando es aplicada a problemas de clasificación. Al no existir limitación con respecto a la representación de los individuos, se pueden utilizar diferentes tipos de representaciones como por ejemplo, representaciones basadas en árboles, en reglas o en funciones discriminantes.

Estas características convierten estos algoritmos en un paradigma de creciente interés tanto para la obtención de reglas de clasificación [119; 226], como, en otras tareas relacionadas con la predicción, como la selección de características [46; 141] y la generación de funciones discriminantes [36; 109]. Existe una gran cantidad de propuestas que utilizan el paradigma GP para evolucionar conjuntos de reglas para diferentes problemas de clasificación, ya sean de dos clases [68; 179] o de múltiples clases, [142; 227], que demuestran que la GP es un campo avalado con una experiencia y que logra de manera eficiente promedios de errores de clasificación bajos en el aprendizaje supervisado.

Estos resultados y características nos han llevado al desarrollo de un modelo de GP para el aprendizaje con múltiples instancias para comprobar su rendimiento. Concretamente, en el que nos hemos basado es en una extensión de la GP clásica, G3P, que proporciona un modo más sistemático para solucionar el problema de cierre, evita el problema de la explosión de código y permite limitar el espacio de búsqueda para que sólo se generen individuos gramaticalmente correctos. Para alcanzar estos objetivos, emplea un gramática libre de contexto que establece una definición formal de las restricciones sintácticas del problema a resolver y sus posibles soluciones. En nuestro caso particular, la gramática permite especificar un lenguaje que genere las reglas de un clasificador. La implementación de las restricciones mediante una gramática es una forma muy habitual de expresar la sintaxis de las reglas de clasificación del tipo SI-ENTONCES que ofrecen una extensión natural de la representación del conocimiento. Obtener información del conocimiento extraído es un área de creciente interés en la actualidad, considerándose casi tan importante como la obtención de una alta precisión en la predicción. Es importante que el usuario pueda entender los resultados del sistema y combinarlos con sus propios conocimientos para finalmente tomar una decisión con fundamento, en lugar de confiar ciegamente en la salida incomprensible que proporcionan los sistemas de caja negra. En este sentido, nuestro sistema tiene la ventaja de añadir claridad y comprensibilidad al conocimiento que se descubre.

Como resultado de todo lo discutido anteriormente, G3P se alza como una de las técnicas evolutivas más adecuada para el desarrollo de un sistema de clasificación para MIL. Por tanto, en este trabajo, se tiene por objetivo diseñar un modelo de generación de reglas mediante G3P para resolver problemas con múltiples instancias, que añada claridad y comprensibilidad al conocimiento que se descubra. Se han tenido en cuenta los siguientes componentes para su desarrollo: un sistema de codificación de individuos apto para la generación de su base de reglas específica, una gramática libre de contexto que permite la generación de individuos sujetos a dicha codificación, un método de inicialización y reemplazo, los operadores genéticos de cruce y mutación, un método de evaluación de individuos especializado para el propósito del dominio de aplicación de sistema y los esquemas evolutivos de cada uno de ellos.

4.2. Algoritmo G3P-MI

En esta sección se especifica el desarrollo del algoritmo que se propone para solucionar problemas con múltiples instancias, G3P-MI (*Grammar Guided Genetic Programming for Multiple Instances*). Se trata de la primera extensión de un algoritmo basado en G3P para trabajar en un entorno de aprendizaje con múltiples instancias, siendo además la primera propuesta de EAs que existen. Especificaremos el sistema teniendo en cuenta aspectos tan importantes como, la representación de los individuos, los operadores genéticos, la función de evaluación y el proceso evolutivo que sigue.

4.2.1. Representación de las soluciones

En nuestro sistema, cada individuo codifica una regla que determina si una bolsa debería ser considerada positiva (es decir, si es una instancia del concepto que nosotros queremos representar) o negativo (si no lo es).

$$
\begin{aligned}
&\textbf{\textit{SI}}(cond_B(bolsa))\ \textbf{\textit{ENTONCES}}\\
&\quad \textit{bolsa es una instancia del concepto.}\\
&\textbf{\textit{SINO}}\\
&\quad \textit{bolsa no es una instancia del concepto.}\\
&\textbf{\textit{FIN-SI}}
\end{aligned}
\tag{4.1}
$$

donde $cond_B$ es una condición que se aplica a la bolsa. De acuerdo a la hipótesis de Dietterich et al., $cond_B$ se puede expresar como:

$$
cond_B(bolsa) = \bigvee_{\forall\ instancia\ \in\ bolsa} cond_I(instancia) \tag{4.2}
$$

donde $\vee$ es el operador de disyunción, y $cond_I$ es una condición que se aplica sobre cada una de las instancias pertenecientes a una bolsa dada.

Como puede comprobarse, la única parte de la expresión 4.1 susceptible de evolucionar es la definición de la expresión $cond_I$. Por esta razón, y de cara a simplificar el genotipo de los individuos y su manipulación, éste sólo contendrá la parte correspondiente a dicha expresión. Como veremos posteriormente, la evaluación de los individuos requerirá la obtención de la regla completa (fenotipo) reemplazando la expresión del genotipo del individuo en las expresiones 4.1 y 4.2.

4.2.1.1. Gramática empleada en la representación de los individuos

Como ya se ha comentado, el genotipo de los individuos que evolucionan en nuestro sistema coincide con las condiciones que se aplican sobre las instancias para decidir si las bolsas a la que pertenecen representan al concepto que deseamos aprender. Dichas expresiones viene descritas de acuerdo al lenguaje especificado por una gramática libre de contexto, G, definida mediante los siguientes cuatro elementos:

$$G = (\sum N, \sum T, S, P)$$

tal que $\sum N \cap \sum T = \oslash$, donde $\sum N$ es el alfabeto de símbolos no terminales, $\sum T$ es el alfabeto de símbolos terminales, S el axioma y P el conjunto de producciones escritas en forma Backus-Naur (Backus-Naur form, BNF) [115].

La Figura 4.1 muestra la gramática libre de contexto diseñada para generar los antecedentes de las reglas de nuestro sistema, que de acuerdo a la notación que se ha empleado corresponde con la especificación de $cond_I$. Según esta gramática, el antecedente de nuestras reglas estará formado por una combinación de comparaciones de atributo/valor unidas mediante los operadores lógicos de conjunción (AND) y de disyunción (OR) indistintamente. En esta gramática el número de atributos y los valores que admite cada uno de ellos se obtienen de forma automática a partir de la información proporcionada sobre el problema con el que se esté trabajando. Con esta información, se generan los símbolos terminales y no terminales asociados a cada atributo y a su rango de valores, que completan la gramática de forma apropiada para el problema concreto que se esté resolviendo.

$\sum N$ = $\langle cond_I \rangle$, $\langle$variable-cat$\rangle$, $\langle$variable-num$\rangle$, $\langle$valores-cat$\rangle$, $\langle$valores-num$\rangle$, $\langle$cmp$\rangle$, $\langle$cmp-num$\rangle$, $\langle$cmp-int$\rangle$, $\langle$cmp-cat$\rangle$, $\langle$op-cat$\rangle$, $\langle$op-num$\rangle$, $\langle$op-int$\rangle$
$\sum T$ = GE, LT, EQ, NOT-EQ, IN, OUT, OR, AND

$\langle cond_I \rangle$:= $\langle$cmp$\rangle$
| **OR** $\langle$cmp$\rangle$ $\langle cond_I \rangle$
| **AND** $\langle cond_I \rangle$
$\langle$cmp$\rangle$:= $\langle$op-num$\rangle$ $\langle$cmp-num$\rangle$
| $\langle$op-cat$\rangle$ $\langle$cmp-cat$\rangle$
| $\langle$op-int$\rangle$ $\langle$cmp-int$\rangle$
$\langle$op-cat$\rangle$:= **EQ**
| **NOT EQ**
$\langle$op-num$\rangle$:= **GE**
| **LT**
$\langle$op-int$\rangle$:= **IN**
| **OUT**
$\langle$cmp-cat$\rangle$:= $\langle$variable-cat$\rangle\langle$value-cat$\rangle$
$\langle$cmp-int$\rangle$:= $\langle$variable-num$\rangle\langle$value-int$\rangle\langle$value-int$\rangle$
$\langle$cmp-num$\rangle$:= $\langle$variable-num$\rangle\langle$value-num$\rangle$
$\langle$variable-cat$\rangle$:= *Any valid attribute in dataset*
$\langle$variable-num$\rangle$:= *Any valid attribute in dataset*
$\langle$valores-cat$\rangle$:= *Any valid value*
$\langle$valores-num$\rangle$:= *Any valid value*

Figura 4.1: Gramática para representar el genotipo de los individuos

```
//Nombre de la base de datos
@relation Base de Ejemplo

//Se especifican los atributos, detallando
//su nombre, tipo y rango de valores
@attribute bolsa integer [1,350]
@attribute atributo1 integer [1, 150]
@attribute atributo2 {etiqueta1, etiqueta2}
@attribute atributo3 real [-1, 1]
@attribute atributo4 real [0, 50]
@attribute clase {clase1, clase2 }

//Se especifican los atributos que son
//de entrada y salida
@inputs atributo1, atributo2, atributo3, atributo4
@outputs clase
```

Figura 4.2: Ejemplo de un fichero de entrada con formato KEEL

Para clarificar un poco la representación, a continuación veremos un ejemplo de gramática desarrollada para un problema concreto. La información de los problemas viene dada por un fichero de entrada que contiene especificado el número de atributos, sus valores y las clases con el que se está trabajando, se admiten dos formatos de trabajo, el formato KEEL [1] y el formato de WEKA [211], que han sido adaptados a MIL desde sus versiones convencionales. Una descripción detallada de los formatos puede verse en el apéndice B.

La figura 4.2 muestra un encabezado de un fichero en formato KEEL multi-instancia, donde se detalla la información del problema de ejemplo que se ha tomado. Podemos ver que este ejemplo contiene cuatro atributos de entrada llamados *atributo1*, *atributo2*, *atributo3* y *atributo4*, todas las variables de entrada toman valores dentro del conjunto de los números reales (se proporciona el rango de valores), excepto *atributo2*, que únicamente puede tomar los valores *etiqueta1* y *etiqueta2*. También contiene un atributo de salida *clase*. De acuerdo a la información mostrada, la variable de salida puede tomar los valores *clase 1* y *clase 2*.

La gramática concreta para este ejemplo, añadiría nuevos símbolos terminales y no terminales correspondientes a los atributos y sus valores definidos en el dominio del

$\sum N$ = ⟨$cond_I$⟩, ⟨cmp⟩, ⟨cmp-num⟩, ⟨cmp-int⟩, ⟨cmp-cat⟩, ⟨op-cat⟩, ⟨op-num⟩, ⟨op-int⟩, ⟨atributo1⟩, ⟨atributo2⟩, ⟨atributo3⟩, ⟨atributo4⟩, ⟨valores-atributo1⟩, ⟨valores-atributo2⟩, ⟨valores-atributo3⟩, ⟨valores-atributo4⟩
$\sum T$ = GE, LT, EQ, NOT-EQ, IN, OUT, OR, AND

⟨$cond_I$⟩ := ⟨cmp⟩
| **OR** ⟨cmp⟩ ⟨$cond_I$⟩
| **AND** ⟨$cond_I$⟩
⟨cmp⟩ := ⟨op-num⟩ ⟨cmp-num⟩
| ⟨op-cat⟩ ⟨cmp-cat⟩
| ⟨op-int⟩ ⟨cmp-int⟩
⟨op-cat⟩ := **EQ**
| **NOT EQ**
⟨op-num⟩ := **GE**
| **LT**
⟨op-int⟩ := **IN**
| **OUT**
⟨cmp-cat⟩ := ⟨atributo2⟩⟨valores-atributo2⟩
⟨cmp-num⟩ := ⟨atributo1⟩⟨valores-atributo1⟩
| ⟨atributo3⟩⟨valores-atributo3⟩
| ⟨atributo4⟩⟨valores-atributo4⟩
⟨atributo1⟩ := **atributo1**
⟨atributo2⟩ := **atributo2**
⟨atributo3⟩ := **atributo3**
⟨atributo4⟩ := **atributo4**
⟨valores-atributo1⟩ := **[1,150]**
⟨valores-atributo2⟩ := **etiqueta1**
| **etiqueta2**
⟨valores-atributo3⟩ := **[-1,1]**
⟨valores-atributo4⟩ := **[0,50]**

Figura 4.3: Gramática particular de un problema para representar el genotipo de los individuos

$cond_I$	*<atributo2 = 'etiqueta1' AND (atributo1 <25 OR atributo3 ≤ 0.5)*

Figura 4.4: Condición generada a partir de la gramática del ejemplo

problema con el que se está trabajando. Por ejemplo: la variable de entrada *atributo1* genera una nueva derivación en el símbolo no terminal *cmp-num*, aportando dos nuevos no terminales *atributo1* y *valores-atributo1* y dos símbolos terminales que son el *nombre de la variable* y *los valores que puede tomar*. La gramática resultante, llamada G1, se puede ver en la Figura 4.3.

Un ejemplo de condición aplicada a instancias, $cond_I$, generado a partir del lenguaje especificado en la gramática de la figura 4.3 podría tener la estructura que se muestra en la figura 4.4. El árbol de derivación asociado asociado a la $cond_I$, se muestra en la figura 4.5. Los nodos azul más oscuro son símbolos no terminales y los nodos azul más claro corresponden a símbolos terminales.

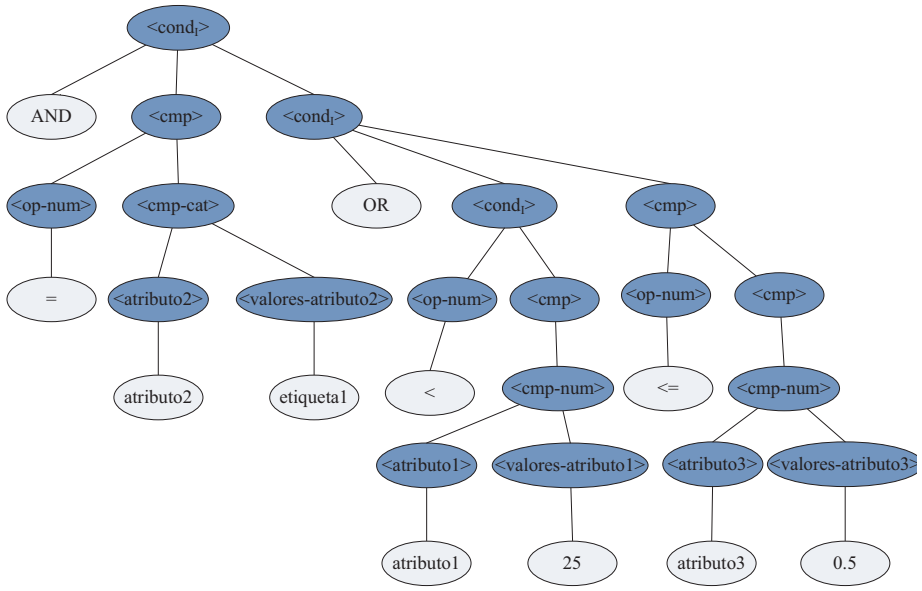


Figura 4.5: Árbol de derivación a partir del clasificador de ejemplo

4.2.2. Generación de la población inicial

El proceso de inicialización de una población consiste en generar un grupo de individuos o soluciones iniciales, que en nuestro caso será un conjunto de clasificadores formados por reglas de aprendizaje. Este tema ha recibido poca atención en la literatura de la GP [124]. Así, desde los algoritmos iniciales de Grow, Full and Ramped Half-and-half [117], y algunos métodos posteriores que tenían en cuenta un tamaño dado y garantizaba las generación de árboles con el mismo o menor tamaño [17; 30], muy pocos artículos han aparecido referenciando este tema [124].

El algoritmo de generación que utiliza nuestra propuesta usa un enfoque inspirado en el método de Geyer and Shultz [83]. Este método, como todos los métodos propuestos para establecer la población inicial de un sistema de G3P, consiste en, dada la gramática libre de contexto que representa el problema en cuestión, utilizar las producciones que conforman la gramática para ir formando cada uno de los individuos que componen la población. Los individuos generados así, son válidos sintácticamente, puesto que se generan siguiendo las producciones de la gramática. El algoritmo comienza escogiendo, de forma aleatoria, alguna producción de la gramática que parta del símbolo inicial de la gramática. Posteriormente, para cada símbolo no terminal que aparece en la producción seleccionada se escoge, también de forma aleatoria, alguna producción que contenga en la parte izquierda a dicho símbolo no terminal, y así recursivamente hasta que se alcancen únicamente símbolos terminales.

La característica fundamental que presenta nuestro método reside en que a la hora de generar los individuos, no se eligen las producciones que lo conforman de manera totalmente aleatoria, sino que se eligen producciones que aseguran en todo momento que el individuo representa el lenguaje generado por la gramática y que la profundidad de su árbol de derivación no va a superar una cota previamente establecida. Esto proporciona un ahorro computacional importante en términos de tiempo y memoria dado que ningún individuo es descartado a la hora de formar parte de la población inicial por ser un individuo no válido. Además, para fomentar la creación de individuos de diferentes tamaños en la población, cada individuo que se genera toma un tamaño diferente, decidido en el momento de su creación y comprendido entre el tamaño mínimo de derivación que permite la gramática y el valor máximo especificado por el usuario.

El método de inicialización recibe dos parámetros obligatorios y uno opcional. Como obligatorios tenemos el número de individuos (N) que componen la población y la profundidad máxima permitida (D) de los árboles de derivación. Como parámetro opcional se encuentra la profundidad mínima permitida (d), que en caso de no ser definida por el usuario, tomará el valor mínimo de las longitudes de todas aquellas producciones derivadas del símbolo inicial de la gramática.

El proceso de generación de la población consta de tres pasos:

1. Se determina la longitud para cada producción de la gramática. Para ello se tienen en cuenta las siguientes definiciones:

   - La longitud de un símbolo no terminal, A, es el mínimo de las longitudes de todas sus producciones, y se denota por $L(A)$.
   - La longitud de un símbolo terminal, a, es cero porque no deriva a nada, denotándose por $L(a) = 0$.
   - La longitud de una producción, $A \rightarrow \alpha$, es el resultado de sumar uno al máximo de las longitudes de los símbolos que componen la parte derecha, y se denota por $L(A \rightarrow \alpha)$.
   - Las producciones que generan sólo símbolos terminales tienen longitud uno, denotándose por $\mathrm{L(A}\rightarrow a) = 1, \forall\ \mathrm{A} \in \sum N$ y $\forall\ \mathrm{a} \in \sum T$.

2. Se calcula la longitud del símbolo inicial de la gramática, seleccionando el mínimo de las longitudes de todas aquellas producciones derivadas del símbolo inicial de la gramática. Esta longitud determina la profundidad mínima permitida (d) para un individuo válido de esa gramática que puede ser calculada de forma automática.

   - Si el usuario determinó un tamaño mínimo (d') el primer paso es comprobar si $d' \geq d$, en cuyo caso la profundidad mínima pasa a ser la establecida por el usuario: $d = d'$.
   - Si el usuario no determinó un tamaño mínimo o en caso de introducirlo, $d' < d$, la profundidad mínima es el valor calculado automáticamente d.

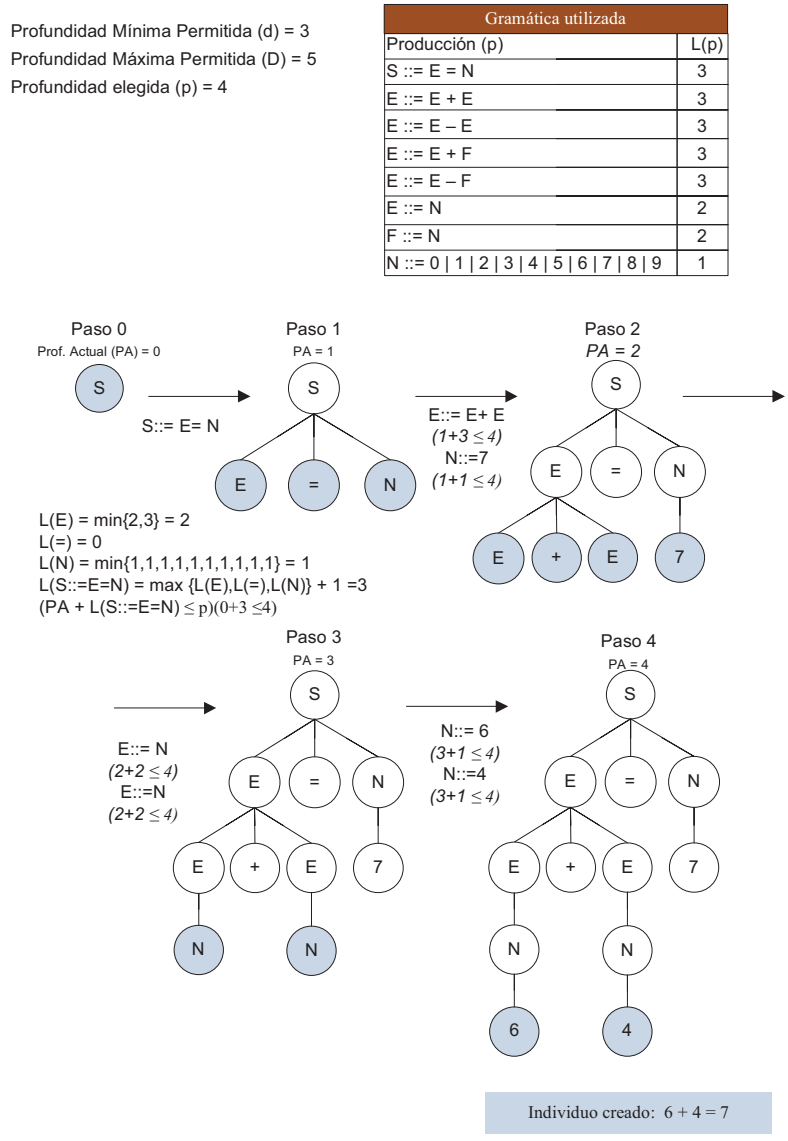


Figura 4.6: Proceso de generación de un individuo

3. Repetir N veces (siendo N el número de individuos de la población):

   a) Se escoge aleatoriamente un valor comprendido entre la profundidad mínima permitida (d) y la profundidad máxima permitida (D). Este valor corresponde a la profundidad máxima elegida (p) para el presente individuo. El algoritmo garantiza la generación de un individuo cuya profundidad está comprendida entre d y p.

   b) Se etiqueta el axioma como el actual no terminal A y se asigna el valor cero a la profundidad actual (PA).

   c) Se selecciona una producción de la forma $A \rightarrow \alpha$, ($\alpha \ \in \{\sum N \cup \sum T\}$) que cumpla: $PA + L(A \rightarrow \alpha) \leq p$. Esta selección no es una selección totalmente aleatoria, sino que para garantizar el tamaño determinado de un individuo, el sistema calcula las probabilidad de selección para cada símbolo en la gramática de acuerdo al número especificado de derivaciones para ese símbolo y el tamaño determinado para dicho individuo. Esta tabla de probabilidades, aunque implica un coste computacional, solamente se calcula una vez y es almacenada con el resto de la información estructural sobre los individuos.

   d) Por cada no terminal $B \in \alpha$, se señala B como el símbolo actual no terminal A y se repiten los pasos 3 y 4, incrementando en una unidad el valor de PA. La figura 4.6 muestra el proceso de generación, mediante el algoritmo propuesto, de un individuo perteneciente a una gramática de igualdades aritméticas con sumas y restas.

4.2.3. Operadores genéticos

G3P-MI usa dos operadores genéticos para generar nuevos individuos en cada generación del algoritmo evolutivo. En esta sección describimos sus principios básicos y su funcionamiento.

Operador de cruce

Este operador está basado en la propuesta de Whigham [206]. Se trata de un operador de propósito general para la resolución de problemas con el paradigma de la G3P y presenta tres características reseñables:

1. Evita el crecimiento desmesurado del tamaño de los árboles de derivación que representan a los individuos [42], fenómeno conocido como explosión de código (*code bloat* en inglés) [146].

2. Proporciona un equilibrio adecuado entre la capacidad de exploración del espacio de búsqueda y la capacidad de explotación, preservando el contexto en el cual los subárboles aparecen en los árboles padre.

3. Permite configurar una lista de símbolos seleccionables que pueden ser utilizados en el cruce, a la vez que se puede incrementar la probabilidad del cruce para ciertos símbolos y decrementar la probabilidad para otros.

La unión de todas estas características proporciona un operador de cruce eficiente que logra una elevada velocidad de convergencia. El operador de cruce gramatical dado dos padres $padre_1$ y $padre_2$ consta de los siguientes pasos:

1. Se crea un conjunto NT (símbolos no terminales) compuesto por los símbolos no terminales del $padre_1$ (salvo la raíz) y que se encuentren dentro de los símbolos posibles que se han especificado para poder ser utilizados en la operación de cruce.

2. Se evalúa el conjunto de símbolos NT.

   a) Si $NT \neq \oslash$, entonces se elige un elemento de dicho conjunto de acuerdo a las probabilidades asignadas a cada símbolo. El símbolo seleccionado lo denominamos símbolo de cruce (SE_1, símbolo elegido).

   b) Si $NT = \oslash$, entonces el cruce entre los individuos no es posible y se elige la raíz como nodo de cruce para ambos árboles, dando lugar a descendientes idénticos a los padres y se termina.

3. Se busca un nodo en el $padre_1$ que contenga el símbolo SE_1, (LN_1, localización del nodo en el árbol del $padre_1$). Este nodo puede verse como la raíz del subárbol que se desea intercambiar.

4. Se calcula la longitud de la derivación del subárbol establecido a partir del símbolo seleccionado en el $padre_1$ (LD_1, longitud de derivación del subárbol del $padre_1$). La longitud de derivación se calcula como el número de símbolos terminales y no terminales incluidos en el subárbol.

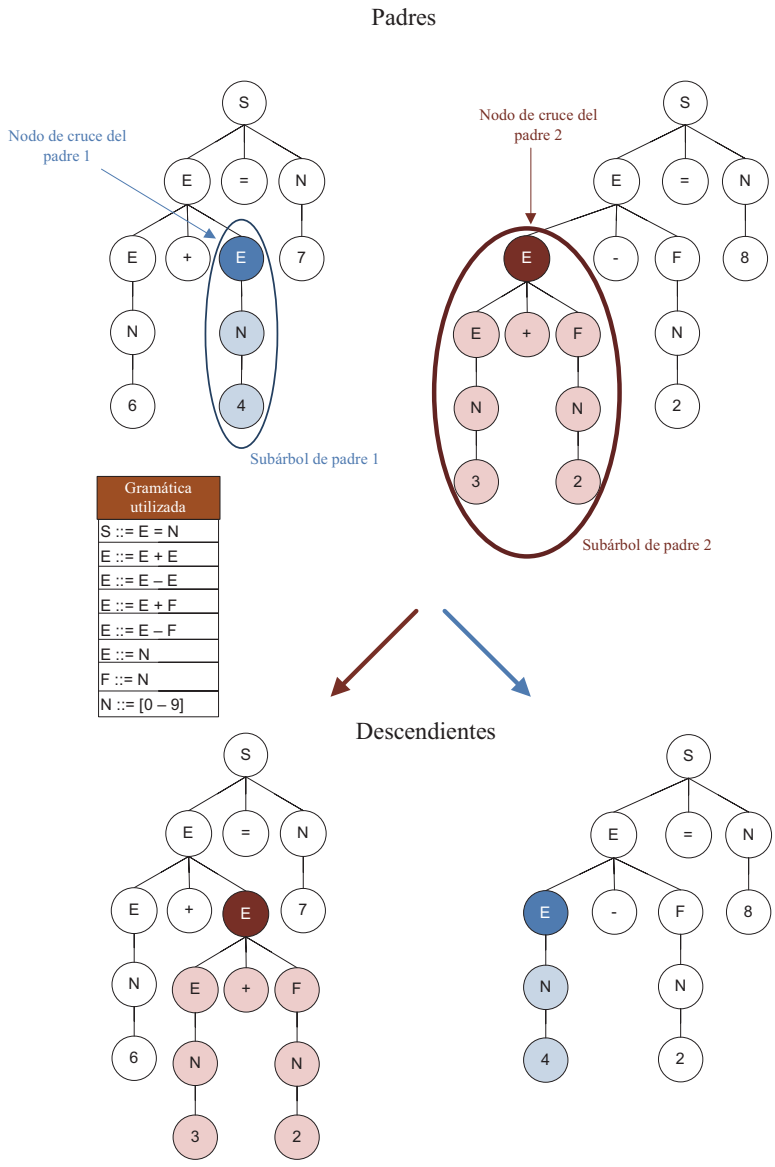


Figura 4.7: Ejemplo de funcionamiento del operador de cruce

5. Se calcula el conjunto NC (símbolos candidatos), formado por todos aquellos símbolos no terminales del $padre_2$ que coincidan o sean compatibles con el símbolo seleccionado en el $padre_1$.

   a) Si $NC \neq \oslash$, se elige aleatoriamente algún símbolo (SE_2, símbolo elegido) que pertenezca al conjunto NC.

   b) Si $NC = \oslash$, entonces no existen nodos de cruce del padre que satisfacen los requisitos de cruce para los símbolos del conjunto NC, por lo que se vuelve al paso 2 (hasta un máximo de 3 veces, tras los cuales si no se consigue compatibilidad, los descendientes son copias de los padres y terminamos la operación de cruce).

6. Se busca en el $padre_2$ un nodo que contenga el símbolo SE_2, (LN_2, localización del nodo en el árbol del $padre_2$). Este nodo puede verse como la raíz del subárbol que se desea intercambiar.

7. Se calcula la longitud de la derivación del subárbol establecido a partir del símbolo seleccionado en el $padre_2$ (LD_2, longitud derivación del subárbol del $padre_2$), como el número de símbolos terminales y no terminales incluidos en el subárbol.

8. Se calcula TH_1 (tamaño del $hijo_1$) como la longitud de la derivación del árbol completo del $padre_1$ (TP_1, tamaño del $padre_1$), a este tamaño le restamos la longitud del subárbol cuya raíz es el nodo seleccionado del $padre_1$ (SE_1) y que se va a pasar al $padre_2$ (LD_1) y le sumamos el subárbol que recibirá del $padre_2$ (LD_2). De igual manera se realiza el mismo procedimiento para obtener el valor TH_2 (tamaño del $hijo_2$).

   a) Si TH_1 o TH_2 superan el valor de la profundidad máxima (D) definida para un individuo, entonces se pasa a un procedimiento que evalúa las opciones de seleccionar el mismo símbolo pero a distinta profundidad en alguno de los padres. Si no hay una combinación posible teniendo en cuenta todos los símbolos iguales que contienen en el árbol los descendientes son copias de los padres y terminamos la operación, sino continuamos con el paso 8.*b*).

b) En caso contrario, se intercambian los dos subárboles cuyas raíces coinciden con los nodos de cruce previamente calculados (SE_1 y SE_2).

La figura 4.7 muestra el resultado de aplicar este método de cruce. Las características fundamentales de este operador de cruce se resumen en los siguientes puntos:

- La descendencia obtenida como resultado de aplicar el operador de cruce siempre da lugar a individuos válidos, es decir, sintácticamente correctos de acuerdo a la gramática dada. Este factor es sumamente importante para la eficiencia del proceso de evolución, ya que todos los individuos generados son posibles soluciones al problema.
- Evita una desaceleración del proceso de selección de nodos en los padres, debido a que una vez elegido el nodo de cruce para el primer padre, el operador tiene en cuenta solamente los nodos del segundo padre que pueden generar individuos válidos.
- Resulta sencillo incorporar al algoritmo un mecanismo de control eficiente que limita el crecimiento en exceso de los árboles, descartando todos aquellos posibles nodos de cruce que generan árboles descendientes de una profundidad superior a una profundidad máxima permitida (D) previamente establecida. Esto conlleva un ahorro de recursos computacionales (tiempo y memoria), dado que los individuos generados son de menor profundidad.

Operador de mutación

La mutación es otro de los operadores más influyentes en el proceso de evolución. Este operador es el responsable de prevenir la pérdida de diversidad genética en la población para evitar una convergencia genética prematura. El operador de mutación produce cambios aleatorios en un individuo para engendrar otro nuevo y continuar el proceso de búsqueda.

El operador de mutación diseñado está basado en el operador de mutación especificado por Whigham [206] y comentado en la sección 3.2.3 . Este operador es de propósito general para la resolución de problemas con el paradigma de G3P y presenta tres características reseñables:

1. Evita el crecimiento desmesurado del tamaño de los árboles de derivación que representan a los individuos [42], fenómeno conocido como explosión de código [146].

2. Proporciona modificaciones en los individuos cambiando símbolos no terminales por otros símbolos no terminales que sean compatibles al tener el mismo antecedente en las reglas de producción de las gramática. Esto nos permite realizar pequeños cambios cuando la población ya ha convergido.

3. Permite configurar una lista de símbolos seleccionables que pueden ser utilizados para la mutación, a la vez que se puede incrementar la probabilidad de mutación para ciertos símbolos y decrementar la probabilidad para otros.

El operador de mutación selectiva parte de un individuo seleccionado y consta de los pasos que se detallan a continuación:

1. Se crea un conjunto NTT (de símbolos no terminales y terminales) compuesto por los símbolos terminales y no terminales del individuo que se encuentren dentro de los símbolos posibles que se han especificado para poder ser utilizados en la operación de cruce.

2. Se evalúa el conjunto de símbolos de NTT.

   - Si $NTT \neq \oslash$, entonces se elige un elemento de dicho conjunto de manera aleatoria, el cual se denomina símbolo de mutación (SE, símbolo elegido).
   - Si $NTT = \oslash$, entonces la mutación del individuo no es posible y el descendiente pasa a ser igual que el padre.

3. Se evalúa el símbolo seleccionado para la mutación (SE).

   - Si SE es un símbolo terminal, se selecciona en el individuo un nodo que contenga el símbolo SE y se reemplaza por otro símbolo terminal compatible. Más precisamente, dos símbolos son compatibles si son terminales que derivan del mismo no terminal.
   - Si SE es un símbolo no terminal, se selecciona en el individuo un nodo que contenga el símbolo SE y el subárbol bajo este nodo seleccionado

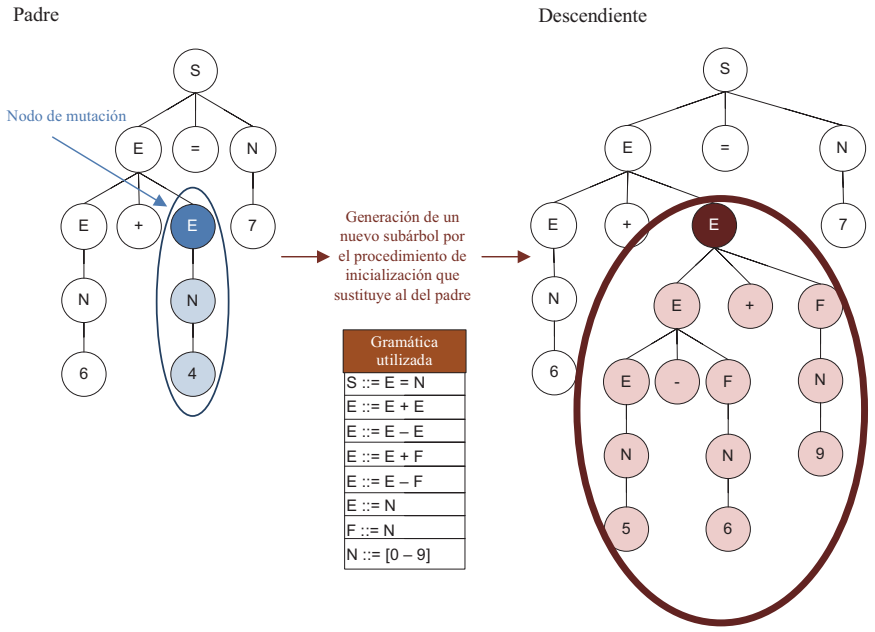


Figura 4.8: Ejemplo de funcionamiento del operador de mutación

será sustituido por cualquier otro subárbol de derivación válido. Para ello seguiremos los siguientes pasos:

a) Se calcula la longitud de la derivación del subárbol establecido en el individuo (LD, longitud derivación del subárbol), como el número de símbolos terminales y no terminales incluidos en el subárbol.

b) Se calcula la longitud de derivación del árbol completo del individuo (TI, tamaño del individuo), a este tamaño le restamos la longitud del subárbol cuya raíz es el nodo seleccionado (SE) y que se va a sustituir por un nuevo subárbol (LD). El valor resultante lo llamamos PA (profundidad actual del árbol del individuo).

c) El procedimiento usado para generar el nuevo subárbol consiste en invocar al mismo procedimiento de inicializar la población pero pasándole como argumentos el símbolo desde el que tiene que generar las producciones (SE) y el valor de profundidad máxima con el que cuenta para generar el subárbol $P = (D - PA)$, siendo D, el valor de profundidad máximo que se ha establecido para todos

los individuos. Este método garantiza la generación de un individuo válido que no exceda un tamaño máximo.

La figura 4.8 muestra el funcionamiento de aplicar este método de mutación selectiva.

4.2.4. Evaluación de los individuos

Cada individuo representa un clasificador, por tanto la función de evaluación de los individuos será escogida de acuerdo a las métricas existentes para evaluar el desempeño de las reglas. Hay diferentes medidas para evaluar diferentes componentes del clasificador y determinar la calidad de cada regla. Nuestra función de ajuste combina dos indicadores comúnmente usados, llamados sensibilidad (Se) y especificidad (Es) [18; 189]. La sensibilidad es la proporción de casos correctamente identificados como miembros de una cierta condición y la especificidad es la proporción de casos correctamente identificados como no miembros de una cierta condición. Se pueden representar por medio de:

$$sensibilidad = \frac{t_n}{t_n + f_p} \tag{4.3}$$

$$especificidad = \frac{t_p}{t_p + f_n} \tag{4.4}$$

donde t_p (*true positive*) representa el número de ejemplos que la regla predice correctamente, f_p (*false positive*) es el número de ejemplos negativos que la regla predice incorrectamente, t_n (*true negative*) es el número de ejemplos negativos que la regla predice correctamente y finalmente, f_n (*false negative*) es el número de ejemplos positivos que la regla predice incorrectamente.

La finalidad de G3P-MI es maximizar tanto la sensibilidad como la especificidad. Estas dos medidas evalúan características conflictivas en el proceso de clasificación. Así, una sensibilidad del 100 % significa que la regla reconoce todos los casos positivos como tales, pero si solamente conocemos esta medida, no sabemos nada acerca de cómo la regla predice las otras clases (en nuestro caso, que trabajamos con dos clases, no conoceríamos cómo de bien clasifica los casos negativos). De este modo,

para tener información de la clase negativa acudimos a la medida de la especificidad. Una especificidad del 100 % significa que la regla reconoce correctamente todos los casos negativos, pero por sí sola, no nos dice nada acerca de la clasificación de los ejemplos positivos. Necesitaríamos, por tanto, conocer tanto la sensibilidad como la especificidad de la regla si queremos evaluar el funcionamiento general de nuestro clasificador. Cuanto mayor sean los porcentajes de ambas medidas, mejor clasificaremos las diferentes clases y por tanto más exacto será nuestro clasificador.

Nuestra evaluación involucra una optimización simultánea de estos dos objetivos conflictivos donde un valor de uno en ambas medidas representa una clasificación perfecta. Este es un punto importante, ya que sería relativamente trivial maximizar el valor de uno de estos dos indicadores a expensas de reducir significativamente el valor del otro. La función de evaluación combina ambas medidas en una única forma de optimizarlas al mismo tiempo. De todas las combinaciones analizadas, el producto de ambas con igual ponderación ha sido el que producía los mejores resultados ya que asigna la misma importancia a todas las medidas y penaliza aquellos individuos con un valor de cero en cualquier de ellas. Además, esta función tiene la ventaja de ser simple y devolver un rango significativo y normalizado entre [0,1].

$$función\ evaluación = sensibilidad \cdot especificidad \tag{4.5}$$

No se ha considerado ninguna medida acerca de la complejidad del problema porque la restricción impuesta al tamaño máximo de los individuos tanto en la inicialización como en los operadores de reproducción garantiza la suficiente sencillez en las reglas para no tener que considerar su inclusión en la evaluación. De este modo, su inclusión incrementaba el coste computacional en la evaluación no aportando una mejoría significativa en la simplicidad de las reglas.

4.2.5. Proceso evolutivo

Nuestro algoritmo está basado en un algoritmo evolutivo generacional clásico y elitista. Inicialmente, se genera una población de reglas de clasificación siguiendo

el procedimiento descrito en la sección 4.2.2. Una vez que los individuos son evaluados con respecto a su habilidad para resolver el problema, el bucle principal del algoritmo es compuesto de las siguiente operaciones. El primer paso representa la fase de selección de padres donde los individuos son seleccionados por medio del torneo binario. Después, se llevan a cabo los procesos de recombinación y mutación con una cierta probabilidad. Una vez que los descendientes son obtenidos, ellos son evaluados. En la última etapa, la población es actualizada por un reemplazamiento que garantiza que el mejor individuo en la población actual no se pierde durante el proceso de actualización. De este modo, el proceso de reemplazamiento determina que la nueva población que se empleará en la siguiente generación estará formada por la población de descendientes obtenida, y según el tamaño de dicha población, ésta será aumentada o reducida, de acuerdo a los siguientes casos:

- Si el tamaño de la población nueva es igual al tamaño de la población actual, el peor individuo de la población nueva será sustituido por el mejor individuo de la población actual.
- Si el tamaño de la población nueva es menor que el tamaño de la población actual, los n mejores individuos de la población actual serán incorporados a la nueva población. Siendo n, el número de individuos necesario para completar el tamaño de la nueva población al tamaño fijado para una población en cada iteración.
- Si el tamaño de la población nueva es mayor que el tamaño de la población actual, los *n-1* peores individuos de la población nueva serán eliminados y se incorporará el mejor individuo de la población actual. En esta caso n es el exceso de individuos que se ha producido en la nueva población de acuerdo al tamaño fijado que debe tener una población en cada iteración.

Finalmente, hay dos condiciones para salir del bucle principal. Por un lado, el algoritmo finaliza si el número máximo de generaciones definido por el usuario es alcanzado y por otro lado, finaliza si el mejor individuo en la población logra la calidad de los objetivos indicados por el usuario.

La figura 4.9 muestra el diagrama general con los principales pasos del algoritmo G3P-MI a alto nivel sin profundizar en los detalles internos del algoritmo.

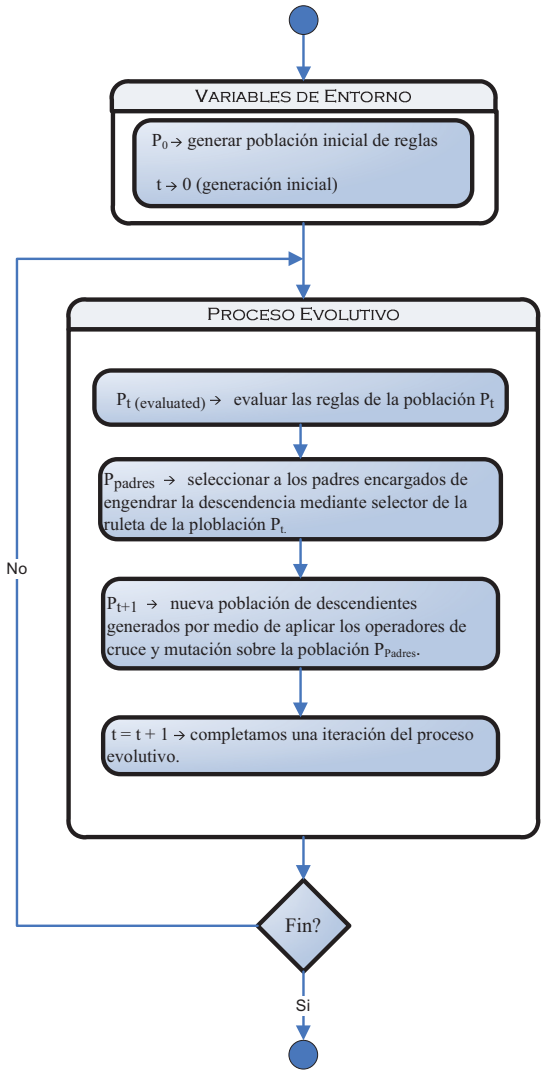


Figura 4.9: Proceso evolutivo del algoritmo G3P-MI

4.3. Experimentación y análisis de resultados

4.3.1. Introducción

En esta sección se pretende evaluar nuestra propuesta frente a otras que han sido publicadas en la bibliografía. Como se comentó en la introducción, el proceso de comparar experimentos, si bien siempre es una tarea difícil para considerar los algoritmos y aplicaciones más relevantes, en MIL todavía se complica más al no existir ningún estudio que compare diferentes propuestas con las aplicaciones más representativas. Los trabajos se limitan a comparar algunas propuestas con algunos conjuntos de datos, pero no existe un estudio comparativo que considere los paradigmas más relevantes. En este trabajo, uno de nuestros objetivos era realizar esta tarea y poder determinar si nuestro modelo funciona de manera correcta mediante una comparación lo más equitativa posible con otras técnicas.

Con esta finalidad, tras una evaluación de las aplicaciones que han sido más ampliamente utilizadas, se han seleccionado 10 de ellas, que incluyen tres dominios de aplicación: la predicción de actividad en los fármacos [126; 242], la recuperación de imágenes basada en contenido [148; 219; 220] y el problema del reto del Este-Oeste [81]. Una descripción de estos problemas puede verse en el apéndice A. Además, se han empleado en la experimentación un total de 15 algoritmos. Para realizar la comparación lo más equitativa posible, se realizará, previamente a la comparación general entre los métodos, un ajuste de los parámetros de configuración de todos algoritmos.

La sección quedará por tanto estructurada de la siguiente manera, comentaremos primeramente los algoritmos que se van a considerar para la experimentación, evaluaremos (con los mismos conjuntos de datos que después se utilizarán en la comparativa general) diferentes configuraciones de estos algoritmos para determinar la mejor propuesta. Una vez determinadas las configuraciones que se emplearán, se llevará a cabo un estudio comparativo entre estas propuestas con nuestro modelo, G3P-MI y se analizarán los resultados que se obtienen.

4.3.2. Algoritmos de aprendizaje usados en la experimentación

Esta sección describe los algoritmos utilizados en la experimentación, que tienen en cuenta tanto los métodos más populares que fueron diseñados especificamente para la resolución de problemas MIL, como extensiones de los paradigmas clásicos del aprendizaje tradicional.

4.3.2.1. Métodos basados en diverse density

Diverse Density (DD), propuesto por Maron y Lozano Pérez en 1998 [126] es uno de los algoritmos de aprendizaje multi-instancia más populares. El algoritmo se basa en aprender un concepto cuyo espacio de características esté suficientemente cercano al menos a una instancia de cada objeto positivo y significativamente lejos de todas las instancias de los objetos negativos. Para este fin, los autores definen el concepto de densidad de diversidad, una medida que determina la cercanía o lejanía de las instancias de los objetos positivos y negativos al punto estimado. La clave de este algoritmo reside en la elección del punto que maximice la densidad de la diversidad, que se obtiene adaptando un clasificador bayesiano estándar que considere bolsas con conjunto de instancias en lugar de instancias individuales. La experimentación considera tres algoritmos basados en el algoritmo clásico de DD para MI:

- *MIDD* es una implementación del algoritmo estándar DD que maximiza la probabilidad a nivel de bolsa usando un modelo de ruido en el entrenamiento. La etiqueta de la clase de una nueva bolsa es asignada basada en la clase que recibe la probabilidad máxima.
- *MIEMDD* este algoritmo combina el algoritmo DD con un enfoque iterativo inspirado por el algoritmo de maximización de la esperanza (EM), EM-DD [233]. En cada iteración se utiliza, el modelo de causa más probable (*most-likely-cause*) [126] para encontrar los puntos finales con más probabilidad sobre el modelo DD que ha sido aprendido.
- *MDD* es una variante del algoritmo DD en el que la probabilidad de la clase a nivel de bolsa se calcula como la media de las probabilidades de las clases

a nivel de instancia [215]. Esta probabilidad es usada en el entrenamiento, cuando la probabilidad es maximizada por el gradiente, y en el test, cuando una etiqueta de la clase es asignada basada en la clase con la probabilidad máxima.

4.3.2.2. Métodos basados en regresión logística

La regresión logística es un método de aprendizaje automático ampliamente utilizado en el aprendizaje con instancias simples [158]. Los experimentos consideran un algoritmo de regresión logística para múltiples instancias, *MILR*, que adapta la regresión logística estándar al aprendizaje de múltiples instancias asumiendo un modelo logístico de instancias simples y usando sus probabilidades de clases para calcular las probabilidad de las clases a nivel de bolsa usando el modelo de ruido empleado por DD.

4.3.2.3. Métodos basados en máquinas de soporte vectorial

Las Máquinas de Soporte Vectorial (SVM) se han desarrollado recientemente en la comunidad del aprendizaje automático y minería de datos y han sido acogidas con mucha popularidad. Hay un gran número de adaptaciones de este enfoque al marco MIL [4; 191; 233] cuyos resultados muestran un buen desarrollo en diferentes aplicaciones. Los experimentos consideran el algoritmo *MISMO* [211], que reemplaza la función del núcleo basada en instancias simples por un núcleo con múltiples instancias (es decir, utiliza una función de similitud apropiada que está basada en bolsas de instancias). MISMO usa el algoritmo SMO [152] para aprendizaje SVM en conjunción con un núcleo con múltiples instancias [80].

4.3.2.4. Métodos basados en distancia

Los algoritmos de los k vecinos más cercanos (kNN) fueron adaptados a MIL por Wang y Zucker [201]. Fundamentalmente necesitan para su adaptación al aprendizaje multi-instancia modificar la definición de la métrica de la distancia. Los diferentes algoritmos en esta propuesta se basan en utilizar diferentes métricas para medir la distancia entre bolsas. Dos esquemas extensivamente empleados son la distancia mínima de Hausdorff y la distancia de Kullback-Leibler. La experimentación considera los siguientes algoritmos:

- *CitationKNN*, se trata del algoritmo del vecino más cercano para MIL propuesto por Wang y Zucker [201]. La métrica que utiliza para medir la distancia entre bolsas está basada en la distancia de Hausdorff. En contraste con el aprendizaje estándar del vecino más cercano, donde solamente se consideran los vecinos más cercanos de un ejemplo para ser clasificados, CitationKNN considera en el proceso de clasificación aquellos ejemplos en el conjunto de entrenamiento para los que el ejemplo a ser clasificado es el más cercano tanto en las referencias como en las citas.

- *MIOptimalBall*, está basado en el método *optimal ball* [7]. Este método implementa la idea de clasificación basada en la distancia a un punto de referencia. Más específicamente, intenta encontrar una esfera en el espacio de instancias que deje fuera de dicha esfera todas las instancias de todas las bolsas negativas y al menos una instancia de cada bolsa positiva esté dentro.

4.3.2.5. Otros algoritmos

También se han considerado una serie de algoritmos que se han extendido de forma diferente del aprendizaje supervisado tradicional a MIL y que incluye a métodos de *ensembles*, SVMs, métodos basados en reglas y otros sistemas de aprendizaje. Concretamente, los clasificaremos en función de su adaptación en tres métodos distintos:

1. *MIWrapper* es un método que asigna la etiqueta de la clase de la bolsa a todas sus instancias y entonces entrena una única instancia sobre los datos resultantes [71]. A la hora de hacer la predicción, la probabilidad de la clase a nivel de bolsa se estima a partir del modelo de instancia simple y las predicciones obtenidas son entonces combinadas (basadas en la medida aritmética o geométrica, o en el valor máximo).

2. *MISimple* es un método que calcula un resumen estadístico de una bolsa para formar instancias simples desde ella. Dependiendo de la opción elegida, calcula la media geométrica analítica, media aritmética o el mínimo y máximo (la última estrategia es también utilizada por el núcleo minimax del algoritmo MISMO discutido anteriormente).

3. *Boosting* es un método popular para mejorar el funcionamiento de los llamados sistemas de aprendizaje débiles [74]. Concretamente, MIBoost [217] es un algoritmo inspirado en el método AdaBoost que construye una serie de clasificadores débiles usando un sistema de aprendizaje de instancias simples para las que las instancias reciben las etiquetas de la bolsa a la que pertenecen. Diferentes hipótesis son consideradas para obtener la etiquetas de la bolsas a partir de las etiquetas de sus instancias asignadas por cada uno de los clasificadores, entre las que se consideran la media geométrica y aritmética y el valor máximo y mínimo.

4.3.3. Parámetros de configuración

G3P-MI ha sido implementado en el software de JCLEC [199] y sus principales parámetros de configuración se muestran en la tabla 4.1. Todos nuestros experimentos son repetidos con 10 semillas diferentes y los resultados medios son los considerados en las comparaciones con el resto de técnicas.

Tabla 4.1: Parámetros de configuración del algoritmo G3P-MI

Algoritmo G3P-MI	
Parámetros	Valores
Tamaño de la Población	1000
Número de Generaciones	100
Probabilidad de Cruce	95 %
Probabilidad de Mutación	30 %
Selector de Padres	*Ruleta*
Profundidad máxima del árbol de derivación	50

4.3.4. Resultados obtenidos

En esta sección, se realiza un estudio comparativo entre algunas de las propuestas más relevantes de MIL con nuestro método. La primera sección se dedicada a ajustar la configuración de los parámetros de los algoritmos que se emplean en la

comparación. Realizado dicho estudio, las propuestas con los parámetros optimizados se evaluarán con los resultados de nuestro modelo.

Todos los conjuntos de datos son particionados usando validación cruzada con diez particiones [162; 208]. Las particiones se construyen sobre bolsas, en la que cada instancia en una bolsa aparece en la misma partición, esta información se encuentra disponible en la dirección *http://www.uco.es/grupos/ayrna/mil*.

4.3.4.1. Configuración de los algoritmos utilizados en la experimentación

En esta primera sección de la parte experimental se lleva a cabo un estudio sobre las diferentes configuraciones de cada algoritmo. Se ajustarán los atributos más significativos y la configuración que produce los mejores resultados será la configuración del algoritmo que se utilice en la comparación global entre todos los modelos.

Entre las configuraciones consideradas, es importante resaltar el uso de diferentes tipos de preprocesado de datos y distintos escenarios multi-instancia. Las dos principales hipótesis consideradas para la conexión de la etiquetas de la clase con las instancias en la bolsa (etiquetas desconocidas) son la hipótesis estándar [57] y la hipótesis colectiva [216]. La primera es definida solamente para problemas de aprendizaje de dos clases, la clase positiva y negativa. Este modelo es asimétrico porque se asume que una bolsa es etiquetada positiva sí y solo sí contiene al menos una instancia positiva, y negativa de otro modo. En contraste, con la segunda hipótesis, se asume que todas las instancias individuales dentro de una bolsa contribuyen igualmente e independientemente a la etiqueta de la clase de la bolsa.

Todas las configuraciones son evaluadas utilizando validación cruzada 10-fold en los diez conjuntos de datos que se han utilizado [162]. Las particiones se construyen sobre bolsas, en las que cada instancia en una bolsa aparece en la misma partición. El test de Friedman [56] se usa para comparar las diferentes alternativas de cada algoritmo sobre todos los conjuntos de datos considerados. Se trata de un test no paramétrico que compara los rangos medios de los algoritmos. La asignación de los rangos se realiza de la siguiente manera: el algoritmo con la precisión más alta para un conjunto de datos se le asigna el rango 1 para dicho conjunto de datos, el siguiente algoritmo con la precisión más alta se le asigna el valor 2 y así hasta asignar posiciones a cada uno de los algoritmo. Este procedimiento se realiza para

cada conjunto de datos y se calcula el rango medio obtenido por cada algoritmo para todos los conjuntos de datos. Estos rangos nos permiten conocer el algoritmo que obtiene los mejores resultados considerando todos los conjuntos de datos. De este modo, el algoritmo con el valor más cercano a 1 indica el mejor algoritmo sobre la mayoría de los conjuntos de datos, en el apéndice C se muestra información acerca de los estudios estadísticos utilizados para comparar los algoritmos.

A continuación, se discuten las mejores configuraciones de cada algoritmo, para ello en las tablas se muestra información de las variantes ejecutadas para cada algoritmo, los valores medios de exactitud (Ex), sensibilidad (Se) y especificidad (Es) obtenidos y los valores de rango. También se adjuntará información de los resultados de test de Friedman.

Algoritmo MILR, se han considerado dos variantes de la hipótesis colectiva, que consideran la media geométrica y aritmética, los resultados obtenidos podemos verlos en la tabla 4.2. El test de Friedman, mostrado en la tabla 4.3, determina que no hay diferencias significativas entre ellas. Sin embargo, los mejores valores medios se obtienen por el uso de una media aritmética, por lo que ha sido la configuración seleccionada.

Tabla 4.2: Configuración de los métodos de regresión logística

Algoritmos basados en Regresión Logística					
Algoritmo	Configuración		Rangos[1]		
	Hipótesis MI	*Tipos de medias*	*Ex*	*Se*	*Es*
MILR *(Type I)*	Collective Assumption	Geometric Mean	1.7499	1.4000	1.7000
MILR *(Type II)* •	Collective Assumption	Arithmetic Mean	1.2500	1.6000	1.3000

[1] Rangos obtenidos por la ejecución del algoritmo en los diez conjuntos de datos

Tabla 4.3: Test de Friedman para los métodos de regresión logística

Test de Friedman				
Algoritmo	*Medida*	*Valor de Friedman*	*Valor de* χ_2 $(\nu = 1, \rho = 0{,}01)$	*Conclusión*
Algoritmo MILR	Ex	2.4999	6.6349	Aceptar la hipótesis nula
	Se	0.4000	6.6349	Aceptar la hipótesis nula
	Es	1.6000	6.6349	Aceptar la hipótesis nula

Algoritmo MIOptimalBall, se han considerado tres variantes con diferentes tipos de filtros para transformación de los datos de entrenamiento, los resultados obtenidos podemos verlos en la tabla 4.4. El test de Friedman, cuyos resultados se pueden ver en la tabla 4.5, determina que no hay diferencias significativas entre ellas. No obstante, observando los rangos medios de las diferentes medidas, se puede ver que los mejores valores se obtienen con la propuesta que emplea los datos de entrenamiento sin ningún tipo de filtrado.

Tabla 4.4: Configuración de los métodos basados en distancia

Algoritmos basados en Distancia				
Algoritmo	Configuración	Rangos[1]		
	Tipos de filtro	*Ex*	*Se*	*Es*
MIOptimalBall *(Type I)* •	Sin filtro	1.7500	1.9500	2.0000
MIOptimalBall *(Type II)*	Datos estandarizados	1.9000	2.1000	1.9500
MIOptimalBall *(Type III)*	Datos normalizados	2.3500	1.9500	2.0500

[1] Rangos obtenidos por la ejecución del algoritmo en los diez conjuntos de datos

Tabla 4.5: Test de Friedman para los métodos basados en distancia

Test de Friedman				
Algoritmo	*Medida*	*Valor de Friedman*	*Valor de* χ_2 $(\nu = 2, \rho = 0{,}01)$	*Conclusión*
MIOptimalBall	Ex	1.9500	9.2104	Aceptar la hipótesis nula
	Se	0.1500	9.2104	Aceptar la hipótesis nula
	Es	0.0499	9.2104	Aceptar la hipótesis nula

Algoritmo MISMO, se han considerado cuatro variantes, en función del núcleo y de si se emplea o no el espacio de característica *minimax*. Los resultados obtenidos podemos verlos en la tabla 4.6. El test de Friedman, cuyos resultados están en la tabla 4.7, vuelve a determinar que no hay diferencias significativas entre ellos. Esto muestra que es un método independiente del tipo de núcleo y de la utilización de *minimax*. Finalmente, el núcleo RBF sin *minimax* es la opción que alcanza los mejores valores y por lo tanto es la configuración seleccionada.

Tabla 4.6: Configuración de los métodos basados en máquinas de soporte vectorial

Algoritmos basados en Máquinas de Soporte Vectorial					
Algoritmo	Configuración		Rangos[1]		
	Núcleo	*Empleo de minimax*	*Ex*	*Se*	*Es*
MISMO *(Type I)* •	RBF Kernel	No emplea minimax	1.7500	2.1999	2.1500
MISMO *(Type II)*	Polynomial Kernel	No emplea minimax	2.5000	1.7500	3.2500
MISMO *(Type III)*	RBF Kernell	Emplea minimax	3.3500	3.2500	2.4500
MISMO *(Type IV)*	Polynomial Kernel	Emplea minimax	2.4000	2.8000	2.1500

[1] Rangos obtenidos por la ejecución del algoritmo en los diez conjuntos de datos

Tabla 4.7: Test de Friedman para los métodos basados en máquinas de soporte vectorial

Test de Friedman				
Algoritmo	*Medida*	*Valor de Friedman*	*Valor de* χ_2 $(\nu = 3, \rho = 0{,}01)$	*Conclusión*
MISMO	Ex	7.7700	11.3449	Aceptar la hipótesis nula
	Se	7.8299	11.3449	Aceptar la hipótesis nula
	Es	4.8599	11.3449	Aceptar la hipótesis nula

Algoritmos MIWrapper, se han considerado cinco sistemas de aprendizaje, incluyendo sistemas basados en reglas (PART), máquinas de soporte vectorial, dos variantes de *ensembles* y clasificadores bayesianos. Cada método se ha evaluado por tres métodos de predicción diferentes considerando las medias aritméticas y geométricas y las probabilidades de clase máximas a nivel de instancia. Los resultados obtenidos podemos verlos en la tabla 4.8. En esta propuesta se seleccionará la mejor opción considerando cada sistema de aprendizaje por separado. De acuerdo al test de Friedman, mostrado en la tabla 4.9, hay diferencias significativas entre los diferentes métodos de predicción con lo que ha seleccionado el algoritmo con los valores de rango más bajos. Normalmente, los mejores métodos han resultado ser la media geométrica y aritmética. El método seleccionado para cada sistema de aprendizaje es señalado en la Tabla 4.8.

Tabla 4.8: Configuración de los métodos basados en aprendizaje supervisado (Wrapper)

Algoritmos basados en Aprendizaje Supervisado (Wrapper)					
Algoritmo	Configuración		Rangos[1]		
	Clasificador	*Método de prueba*	*Ex*	*Se*	*Es*
PART *(Tipo I)*	PART	Media aritmética[2]	1.7000	1.4999	2.6500
PART *(Tipo II)* •	PART	Media geométrica[3]	1.4000	1.5000	2.3500
PART *(Tipo III)*	PART	Valor máxima[4]	2.9000	2.9999	0.9999
Bagging&PART *(Tipo I)*	Bagging con PART	Media aritmética[2]	1.5000	1.5499	2.5000
Bagging&PART *(Tipo II)* •	Bagging con PART	Media geométrica[3]	1.5000	1.4500	2.5000
Bagging&PART *(Tipo III)*	Bagging con PART	Valor máxima[4]	2.9999	2.9999	0.9999
AdaBoost&PART *(Tipo I)* •	AdaBoost con PART	Media aritmética[2]	1.3499	1.4500	2.5000
AdaBoost&PART *(Tipo II)*	AdaBoost con PART	Media geométrica[3]	1.7000	1.5499	2.5000
AdaBoost&PART *(Tipo III)*	AdaBoost con PART	Valor máxima[4]	2.9499	2.9999	0.9999
SMO *(Tipo I)* •	SMO con núcleo PolyKernel	Media aritmética[2]	1.6500	1.4500	2.5000
SMO *(Tipo II)*	SMO con núcleo PolyKernel	Media geométrica[3]	1.7999	1.7499	2.2000
SMO *(Tipo III)*	SMO con núcleo PolyKernel	Valor máxima[4]	2.5500	2.8000	1.3000
NaiveBayes *(Tipo I)*	NaiveBayes	Media aritmética[2]	1.9500	1.7000	2.2000
NaiveBayes *(Tipo II)* •	NaiveBayes	Media geométrica[3]	1.9500	1.4000	2.6999
NaiveBayes *(Tipo III)*	NaiveBayes	Valor máxima[4]	2.1000	2.8999	1.0999

[1] Rangos obtenidos por la ejecución del algoritmo en los diez conjuntos de datos
[2] Media aritmética de las probabilidades de la clase a nivel de instancia
[3] Media geométrica de las probabilidades de la clase a nivel de instancia
[4] Valor máximo de las probabilidades de la bolsa positiva

Tabla 4.9: Test de Friedman para los métodos basados en aprendizaje supervisado (Wrapper)

Test de Friedman				
Algoritmo	*Medida*	*Valor de Friedman*	*Valor de* χ_2 $(\nu = 2, \rho = 0{,}01)$	*Conclusión*
PART (Wrapper)	Ex	12.5999	9.2104	Rechazar la hipótesis nula
	Se	15.0499	9.2104	Rechazar la hipótesis nula
	Es	15.4500	9.2104	Rechazar la hipótesis nula
Bagging&PART (Wrapper)	Ex	14.9999	9.2104	Rechazar la hipótesis nula
	Se	15.0499	9.2104	Rechazar la hipótesis nula
	Es	15.0000	9.2104	Rechazar la hipótesis nula
AdaBoost&PART (Wrapper)	Exa	14.1499	9.2104	Rechazar la hipótesis nula
	Se	15.0499	9.2104	Rechazar la hipótesis nula
	Es	15.0000	9.2104	Rechazar la hipótesis nula
SMO (Wrapper)	Exa	4.6499	9.2104	Aceptar la hipótesis nula
	Se	10.049	9.2104	Rechazar la hipótesis nula
	Es	7.8000	9.2104	Aceptar la hipótesis nula
NaiveBayes (Wrapper)	Exa	0.1500	9.2104	Accept null hypothesis
	Se	12.5999	9.2104	Rechazar la hipótesis nula
	Es	13.3999	9.2104	Rechazar la hipótesis nula

Algoritmos MISimple, se ha evaluado el efecto de usar diferentes sistemas de aprendizaje base considerando los sistemas AdaBost y PART. Cada método se ha evaluado con tres métodos diferentes de etiquetado de bolsa usando la media aritmética, geométrica o la media del mínimo y máximo de cada atributo. Los resultados obtenidos podemos verlos en la tabla 4.10. El test de Friedman, en la tabla 4.11, determina que no hay diferencias significativas entre los algoritmos. Esto se traduce en que los métodos de transformación con múltiples instancias son independientes. Sin embargo, la media aritmética obtiene los mejores resultados en los dos métodos de aprendizaje. Por lo tanto, este método será el seleccionado para la comparación.

Tabla 4.10: Configuración de los métodos basados en aprendizaje supervisado (Simple)

Algoritmos basados en Aprendizaje Supervisado (Simple)					
Algoritmo	Configuración		Rangos[1]		
	Clasificador	*Método de Transformación*	*Ex*	*Se*	*Es*
AdaBoost&PART *(Tipo I)* •	AdaBoost con PART	Media aritmética[2]	1.7500	1.7000	1.9000
AdaBoost&PART *(Tipo II)*	AdaBoost con PART	Media geométrica[3]	2.5500	2.2500	2.3000
AdaBoost&PART *(Tipo III)*	AdaBoost con PART	Valor máximo[4]	1.7000	2.0500	1.8000
PART *(Tipo I)* •	PART	Media aritmética[2]	1.7000	1.8500	1.7500
PART *(Tipo II)*	PART	Media geométrica[3]	2.4500	2.1999	2.2500
PART *(Tipo III)*	PART	Valor máximo[4]	1.8499	1.9499	2.0000

[1] Rangos obtenidos por la ejecución del algoritmo en los diez conjuntos de datos
[2] Media aritmética por bolsa de cada atributo
[3] Media geométrica por bolsa de cada atributo
[4] Valor máximo por bolsa de cada atributo

Tabla 4.11: Test de Friedman para los métodos basados en aprendizaje supervisado (Simple)

Test de Friedman				
Algoritmo	*Medida*	*Valor de Friedman*	*Valor de* χ_2 $(\nu = 2, \rho = 0{,}01)$	*Conclusión*
AdaBoost&PART (Simple)	Ex	3.1500	9.2104	Aceptar la hipótesis nula
	Se	0.6499	9.2104	Aceptar la hipótesis nula
	Es	1.2500	9.2104	Aceptar la hipótesis nula
PART (Simple)	Ex	4.5500	9.2104	Aceptar la hipótesis nula
	Se	1.5500	9.2104	Aceptar la hipótesis nula
	Es	1.4000	9.2104	Aceptar la hipótesis nula

Algorimto MIBOOST, se han considerado cuatro variantes con diferentes combinaciones de los sistemas de aprendizaje base y un número máximo de iteraciones del proceso; los resultados obtenidos podemos verlo en la tabla 4.12. El test de Friedman, cuyos resultados se pueden ver en la tabla 4.13, determina que este algoritmo no está relacionado con el clasificador ni con el incremento en el número de iteraciones. La configuración que emplea diez iteraciones y el método de aprendizaje RepTree [211] es la que logra los mejores rangos medios para todas las medidas y por tanto será la seleccionada.

Tabla 4.12: Configuración de los métodos basados en aprendizaje supervisado (Boost)

Algoritmos basados en Aprendizaje Supervisado (Boost)					
Algoritmo	Configuración		Rangos[1]		
	Clasificador	*Número de iteraciones*	*Ex*	*Se*	*Es*
MIBoost *(Tipo I)*	RepTree	50 iteraciones	2.0500	2.5500	2.5000
MIBoost *(Tipo II)* •	RepTree	10 iteraciones	2.0000	2.1500	2.8000
MIBoost *(Tipo III)*	DecisionStump	50 iteraciones	2.6500	2.4000	2.3000
MIBoost *(Tipo IV)*	DecisionStump	10 iteraciones	3.3000	2.9000	2.4000

[1] Rangos obtenidos por la ejecución del algoritmo en los diez conjuntos de datos

Tabla 4.13: Test de Friedman para los métodos basados en aprendizaje supervisado (Boost)

Test de Friedman				
Algoritmo	*Medida*	*Valor de Friedman*	*Valor de* χ_2 $(\nu = 3, \rho = 0{,}01)$	*Conclusión*
MIBOOST	Ex	6.6900	11.3449	Aceptar la hipótesis nula
	Se	1.7700	11.3449	Aceptar la hipótesis nula
	Es	0.8400	11.3449	Aceptar la hipótesis nula

Los algoritmos seleccionados de cada paradigma serán las configuraciones que utilizaremos en la comparación con nuestra propuesta, tanto en los casos que no existían diferencias significativas entre los métodos como en los casos en los que si existían se han seleccionado las configuraciones que mejores valores han obtenido en las diferentes medidas.

4.3.4.2. Resultados experimentales con G3P-MI

Esta sección compara nuestro método con las mejores configuraciones de los métodos comentados en la sección previa. Las Tablas 4.14 y 4.15 nos informan de los resultados medios de exactitud, sensibilidad y especificidad para todos los algoritmos tenidos en cuenta en cada conjunto de datos considerado. Se utilizará el test de Friedman [56] para evaluar la diferencia entre los algoritmos, en aquellos casos que dicho test nos determine que existan diferencias significativas en los métodos, se llevará a cabo un test a posteriori que será el de Bonferroni-Dunn [60]. Los rangos obtenidos por cada algoritmo en la evaluación de los 10 conjuntos de datos para todas las medidas se muestran en la Tabla 4.16 y los resultados del test de Friedman se indican en la Tabla 4.17.

Tabla 4.14: Resultados experimentales en la comparación de G3P-MI con otras técnicas de MIL para el problema de predicción de actividad en los fármacos

Resultados Globales						
	***D**atos*	***M**.Atoms*	***M**.Bonds*	***M**.Chains*	***M**usk1*	***M**usk2*
MISMO	Ex	0.7000	0.8421	0.7842	0.8778	0.8400
	Se	0.3833	0.8833	0.7167	0.8250	0.8667
	Es	0.8462	0.8231	0.8154	0.9200	0.8000
MIOptimalBall	Ex	0.7263	0.7421	0.7158	0.8000	0.7700
	Se	0.5500	0.6833	0.4500	0.9250	0.8333
	Es	0.8077	0.7692	0.8385	0.7000	0.8050
MILR	Ex	0.7000	0.7105	0.7684	0.8778	0.8500
	Se	0.2000	0.2167	0.4333	0.8500	0.9167
	Es	0.9308	0.9385	0.9231	0.9000	0.7500
PART	Ex	0.8105	0.7895	0.8684	0.8444	0.8500
(Wrapper)	Se	0.5667	0.7000	0.7500	0.8750	0.8833
	Es	0.9231	0.8308	0.9231	0.8200	0.8000
Bagging&PART	Ex	0.8105	0.7789	0.8684	0.9000	0.8500
(Wrapper)	Se	0.5833	0.7167	0.7833	0.8500	0.8667
	Es	0.9154	0.8077	0.9077	0.9400	0.8250
AdaBoost&PART	Ex	0.8053	0.7737	0.8737	0.8778	0.9000
(Wrapper)	Se	0.5333	0.7000	0.7667	0.8750	0.9167
	Es	0.9308	0.8077	0.9231	0.8800	0.8750
SMO	Ex	0.6842	0.6842	0.6895	0.8444	0.8900
(Wrapper)	Se	0.0000	0.0000	0.0167	0.8250	0.9333
	Es	1.0000	1.0000	1.0000	0.8600	0.8250
NaiveBayes	Ex	0.6842	0.6263	0.5263	0.8000	0.7700
(Wrapper)	Se	0.0000	0.1500	0.9667	0.8000	0.7667
	Es	1.0000	0.8462	0.3231	0.8000	0.7750
PART	Ex	0.7000	0.8263	0.7737	0.8111	0.7400
(Simple)	Se	0.4833	0.8333	0.5833	0.8000	0.7833

Continúa en la siguiente página

Continuación de la página anterior

Resultados Globales						
	***D**atos*	***M**.Atoms*	***M**.Bonds*	***M**.Chains*	***M**usk1*	***M**usk2*
	Es	0.8000	0.8231	0.8615	0.8200	0.6750
AdaBoost&PART	Ex	0.7526	0.8474	0.7947	0.8333	0.7900
(Simple)	Se	0.5167	0.8333	0.6667	0.8250	0.8500
	Es	0.8615	0.8538	0.8538	0.8400	0.7000
MIBoost	Ex	0.8316	0.7737	0.8474	0.8778	0.8700
	Se	0.7333	0.7333	0.7333	0.8000	0.9000
	Es	0.8769	0.7923	0.9000	0.9400	0.8250
MIEMDD	Ex	0.7316	0.7211	0.7368	0.8889	0.9000
	Se	0.5000	0.4333	0.5000	0.8750	0.8833
	Es	0.8385	0.8538	0.8462	0.9000	0.9250
MIDD	Ex	0.7263	0.7526	0.7684	0.9222	0.7300
	Se	0.4167	0.5593	0.5500	0.9500	0.9500
	Es	0.8692	0.8615	0.8692	0.9000	0.4000
MDD	Ex	0.7211	0.7316	0.7684	0.7889	0.7400
	Se	0.1667	0.4000	0.5500	0.7750	0.8500
	Es	0.9769	0.8846	0.8692	0.8000	0.5750
CitationKNN	Ex	0.7526	0.7316	0.7474	0.9444	0.8500
	Se	0.5833	0.4333	0.5167	0.9750	0.8667
	Es	0.8308	0.8692	0.8538	0.9200	0.8250
G3P-MI	Ex	0.8526	0.8210	0.8105	0.9445	0.8800
	Se	0.8462	0.8462	0.9231	1.0000	0.7500
	Es	0.8167	0.7833	0.7333	0.9000	0.9180

Tabla 4.15: Resultados experimentales en la comparación de G3P-MI con otras técnicas de MIL para el problema de clasificación de imágenes y de *east-west*

Resultados Globales						
***D**atos*		***F**ox*	***T**iger*	***E**lephant*	***E**astWest*	***W**eastEast*
MISMO	Ex	0.5800	0.8250	0.7900	0.7500	0.6500
	Se	0.4100	0.7800	0.7500	0.7000	0.6000
	Es	0.7500	0.8700	0.8300	0.8000	0.7000
MIOptimalBall	Ex	0.5300	0.6700	0.7750	0.7000	0.3000
	Se	0.7400	0.6500	0.8800	0.7000	0.3000
	Es	0.3200	0.6900	0.6700	0.7000	0.3000
MILR	Ex	0.5600	0.7700	0.7800	0.6000	0.6000
	Se	0.4900	0.7400	0.6900	0.5000	0.7000
	Es	0.6300	0.8000	0.8700	0.7000	0.5000
PART	Ex	0.6500	0.8150	0.8050	0.5000	0.5500
(Wrapper)	Se	0.6100	0.8300	0.7300	0.5000	0.7000
	Es	0.6900	0.8000	0.8800	0.5000	0.4000
Bagging&PART	Ex	0.6900	0.8100	0.8600	0.6000	0.5500
(Wrapper)	Se	0.6400	0.8000	0.7900	0.5000	0.6000
	Es	0.7400	0.8200	0.9300	0.7000	0.5000
AdaBoost&PART	Ex	0.6900	0.8000	0.8400	0.5000	0.6000
(Wrapper)	Se	0.6100	0.8000	0.7800	0.4000	0.6000
	Es	0.7700	0.8000	0.9000	0.6000	0.6000
SMO	Ex	0.6350	0.8000	0.8550	0.5500	0.5000
(Wrapper)	Se	0.4800	0.7700	0.8000	0.6000	0.8000
	Es	0.7900	0.8300	0.9100	0.5000	0.2000
NaiveBayes	Ex	0.5900	0.7650	0.8400	0.5000	0.6000
(Wrapper)	Se	0.2700	0.8800	0.7700	0.5000	0.6000
	Es	0.9100	0.6500	0.9100	0.5000	0.6000
PART	Ex	0.5800	0.7650	0.7800	1.0000	1.0000
(Simple)	Se	0.6200	0.8100	0.7600	1.0000	1.0000

Continúa en la siguiente página

Continuación de la página anterior

Resultados Globales						
***D**atos*		***F**ox*	***T**iger*	***E**lephant*	***E**astWest*	***W**eastEast*
	Es	0.5400	0.7200	0.8000	1.0000	1.0000
AdaBoost&PART	Ex	0.6250	0.7950	0.8100	0.9500	0.8500
(Simple)	Se	0.6200	0.8100	0.8200	0.9000	0.8000
	Es	0.6300	0.7800	0.8000	1.0000	0.9000
MIBoost	Ex	0.6950	0.8200	0.8150	0.6000	0.5000
	Se	0.6300	0.8000	0.7400	0.5000	0.6000
	Es	0.7600	0.8400	0.8900	0.7000	0.4000
MIEMDD	Ex	0.5900	0.7450	0.7300	0.4500	0.3000
	Se	0.3900	0.7100	0.7100	0.7000	0.3000
	Es	0.7900	0.7800	0.7500	0.2000	0.3000
MIDD	Ex	0.6550	0.7400	0.8300	0.5000	0.2500
	Se	0.6200	0.7200	0.8300	0.7000	0.3000
	Es	0.6900	0.7600	0.8300	0.3000	0.2000
MDD	Ex	0.7000	0.7550	0.8000	0.5500	0.5500
	Se	0.6300	0.7700	0.8200	0.8000	0.7000
	Es	0.7700	0.7400	0.7800	0.3000	0.4000
CitationKNN	Ex	0.5000	0.5000	0.5000	0.5500	0.5500
	Se	1.0000	1.0000	1.0000	0.5000	0.7000
	Es	0.0000	0.0000	0.0000	0.6000	0.4000
G3P-MI	Ex	0.7050	0.8700	0.8800	1.0000	0.8500
	Se	0.7900	0.9400	0.9300	1.0000	0.7000
	Es	0.6200	0.8000	0.8300	1.0000	1.0000

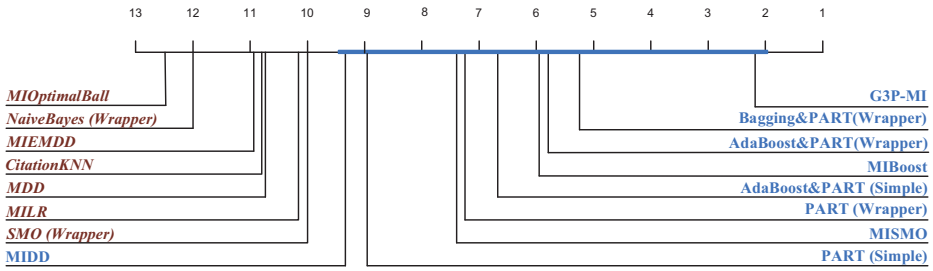
Con respecto a los valores de exactitud, nuestro algoritmo logra los mejores valores, obteniendo un valor 2.20, los valores del siguiente algoritmo están muy alejados de estos valores. Concretamente, el segundo algoritmo se basa en los métodos de wrapper, Bagging&PART, toma un valor de rango de 5.25. Estos resultados confirman que nuestros resultados, considerando todos los conjuntos de datos, superan a todos los algoritmos.

Los resultados de aplicar el test de Friedman son mostrados en la Tabla 4.17. Este test indica que los resultados son significativamente diferentes para los valores

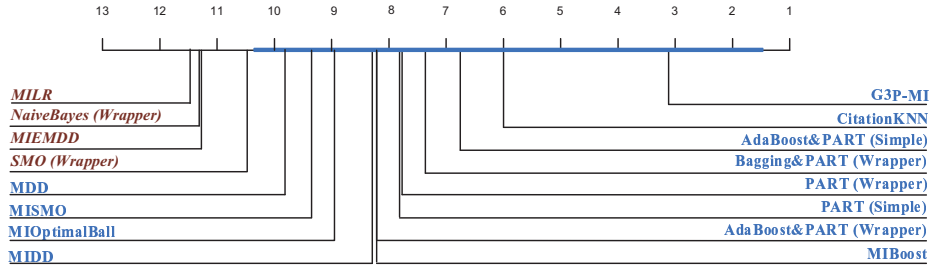
Tabla 4.16: Rangos medios de los algoritmos (comparativa de G3P-MI para todos los conjuntos de datos)

Rangos Medios			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especifidad***
MISMO	7.40	9.35	7.35
MIOptimalBall	12.50	8.95	13.20
MILR	10.15	11.45	6.70
MIEMDD	10.95	11.25	9.35
MIDD	9.35	8.30	10.35
MDD	10.75	9.80	9.75
CitationKNN	10.80	6.00	10.45
G3P-MI	2.20	3.10	8.45
PART (Wrapper)	7.25	7.75	8.45
Bagging&PART (Wrapper)	5.25	7.35	5.85
AdaBoost&PART (Wrapper)	5.80	8.20	6.20
SMO (Wrapper)	10.00	10.45	5.45
NaiveBayes (Wrapper)	12.00	11.30	8.70
PART (Simple)	8.95	7.80	10.40
AdaBoost&PART (Simple)	6.70	6.75	8.85
BOOST	5.95	8.20	6.50

de exactitud, por lo que es necesario realizar el test de Bonferroni-Dunn [60], para encontrar diferencias significativas ocurridas entre los algoritmos. La figura 4.10(a), muestra la aplicación de este test sobre esta medida. Este gráfico representa un gráfico de barras, cuyos valores son proporcionales a los rangos medios obtenidos de cada algoritmo. Existe un umbral, calculado como la suma del valor de rango más bajo de todos los algoritmos más el valor de Diferencia Crítica (*Critical Diference*, CD) que nos determina este test. Este umbral es representado en el gráfico por una línea horizontal más gruesa y de color verde (que para esta medida se fija en 9.445), todos aquellos algoritmos que tengan un valor de rango que exceda este límite son considerados algoritmos significativamente peores que el algoritmo de control (el algoritmo de control es el algoritmo que obtiene los valores de rango más bajos). El algoritmo de control en este caso es nuestra propuesta, G3P-MI, que es la que obtiene el valor más bajo de rango.



(a) Resultados para Exactitud



(b) Resultados para Sensibilidad

Figura 4.10: Resultados del test de Bonferroni-Dunn ($p < 0{,}01$) en la comparativa del modelo G3P-MI

Observando la figura 4.10(a), los algoritmos que no exceden el umbral determinado por el test de Bonferroni-Dunn son G3P-MI, algunas versiones de los algoritmos *wrapper* y *simple*, las máquinas de soporte vectorial (MISMO), método basado en boosting (MIBoost) y Diverse Density (MIDD), todos ellos representados en color azul. El resto de algoritmos, representados en la figura 4.10(a) en color rojo, son considerados algoritmos que obtienen estadísticamente peores resultados. En este estudio, nuestra propuesta vuelve a obtener los mejores resultados en diferentes conjuntos de datos. Los algoritmos más cercanos son los métodos de wrapper basado en diferentes sistemas de aprendizaje.

Con respecto a la medida de sensibilidad, nuestra propuesta logra de nuevo el mejor valor de rango medio para esta medida, obteniendo un valor muy próximo al 1, concretamente 1.50. De nuevo, los valores del siguiente algoritmo están bastante alejados, se trata de un algoritmo basado en distancias (CitationKNN) y obtiene un valor de rango de 5.45. Para comprobar estadísticamente si existen diferencias

Tabla 4.17: Resultados del test de Friedman (comparativa de G3P-MI con todos los conjuntos de datos)

Test de Friedman			
Medida	Valor de Friedman	$\chi^2(\rho = 0{,}01)$	Conclusión
Exactitud	53.1882	30.5780	Rechazar hipótesis nula
Sensibilidad	31.5838	30.5780	Rechazar hipótesis nula
Especificidad	28.9213	30.5780	Aceptar hipótesis nula

significativas entre los algoritmos para esta medida, realizamos el test de Friedman, cuyos resultados se muestran en la tabla 4.17. El test de Friedman indica que sí existen diferencias significativas, por lo que es necesario realizar el test de Bonferroni-Dunn para determinar cuales algoritmos pueden ser considerados peores propuestas.

La figura 4.10(b) muestra la aplicación del test de Bonferroni-Dunn sobre la medida de sensibilidad y fija el valor del umbral en 10.345. Observando esta figura, los algoritmos que no exceden este umbral, representados por color azul, son principalmente versiones de wrapper y simple, algoritmos basados en distancia (CitationkNN y OptimallBall), MIBoost y Diverse Density (MIDD, MDD). El resto de algoritmos, representados en color rojo, son consideradas peores propuestas con respecto a estos valores.

Con respecto a la medida de especificidad, el test de Friedman, en la Tabla 4.17, nos determina que no existen diferencias significativas entre los algoritmos para esta medida, por ello no será necesario realizar un test a posteriori.

Podemos concluir que nuestra propuesta obtiene los mejores valores considerando las diferentes medidas de evaluación de los clasificadores. Si nos fijamos en una medida concreta, los test estadísticos nos determinen que existen varios algoritmos entre los que no se puede indicar que haya diferencias entre sus resultados, sin embargo, si consideramos en la evaluación todas las medidas que se desean optimizar, podemos ver, que muchos de ellos, cuando en una de las medidas son considerados dentro de los algoritmos relevantes, para las otras medidas son considerados irrelevantes. Lo que conlleva a clasificadores erróneos por no clasificar alguna de las clases correctamente. Por ejemplo, si observamos el algoritmo *CitationKNN* que es el algoritmo de control para la medida de sensibilidad, si nos fijamos en su posición

con respecto a la medida de precisión podemos ver que se encuentra en las últimas posiciones, siendo considerado un algoritmo que estadísticamente presenta diferencias significativas con el algoritmo de control para precisión. Esto demuestra que el algoritmo clasifica muy bien una clase, pero muy mal la otra, con lo que finalmente la precisión no es aceptable. Si observamos nuestro algoritmo, vemos que mantiene una buena posición para todas las medidas, siendo el mejor en la medida de precisión, este hecho resalta que nuestra propuesta es una técnica fiable, presentando un clasificador con las mejores garantías de hacer clasificaciones correctas tanto para clasificar la clase positiva como la negativa.

4.4. Conclusiones

En este capítulo se presenta G3P-MI, un algoritmo de programación genética gramatical para aprendizaje con múltiples instancias. Nuestra propuesta se compara con los paradigmas más importantes que se han diseñado hasta la fecha para resolver estos problemas, considerando tanto algoritmos clásicos diseñados exclusivamente para este aprendizaje (algoritmo DD y sus variantes) así como adaptaciones de los paradigmas clásicos (considerando algoritmos basados en distancia, en máquinas de soporte vectorial, en métodos de regresión logística, en reglas, en *ensembles* y en *boosting*) y todas las propuestas se evalúan con diez conjuntos de datos que contemplan tres dominios de aplicación donde MIL ha sido ampliamente utilizado.

Los resultados experimentales han demostrado que nuestro algoritmo obtiene mejores resultados que los otros algoritmos en las diferentes aplicaciones con respecto a la exactitud, sensibilidad y especificidad. El test de Friedman confirma que se pueden establecer diferencias significativas entre los resultados de los diferentes métodos con respecto a la sensibilidad y exactitud. El test *a posteriori* llevado a cabo para medir diferentes medidas presentadas, el test de Bonferroni-Dunn, determina que nuestra propuesta es la que obtiene los mejores resultados para ambas medidas. Además, las demás técnicas, si bien mantienen una buena posición en una de las medidas (sensibilidad o especificidad), normalmente suelen tener una de los peores valores en la otra; este hecho indica que no son unos clasificadores aprpiados, debido a que clasifican erróneamente alguna de las clases. Otra de las

ventajas más destacadas de G3P-MI es que genera un sistema basado en reglas, generando reglas con características tan deseables como que son simples, intuitivas y modulares.

Estudios experimentales de estas características son una de la mayor deficiencia que podemos encontrar en MIL, donde la mayoría de los estudios que se han realizado han comparado de forma empírica solamente algunos enfoques o han usando algunos conjuntos de datos. Por ello, toda la información sobre los conjuntos de datos y las particiones se ha puesto disponible, para que los resultados puedan ser reproducidos fácilmente y nuevos estudios experimentales se pueden llevar a cabo eficiente y eficazmente bajo las mismas condiciones.

5

Modelos basados en Programación Genética Gramatical Multi-Objetivo para Aprendizaje Multi-Instancia

5.1. Introducción

El problema de evaluar la calidad de los clasificadores, ya sea bien desde una perspectiva de aprendizaje con múltiples instancias o con instancias simples, posee una naturaleza multi-objetivo, donde se desean optimizar diferentes métricas contradictorias para medir su rendimiento. Si nosotros intentamos optimizar el valor de cualquiera de las medidas, el valor de las otras se puede reducir significativamente. Las meta-heurísticas multi-objetivo, en este contexto, pueden usarse para producir diferentes balances entre la confianza, la cobertura y la complejidad de los resultados de un clasificador. En los últimos años se han publicado multitud de trabajos relacionados con la optimización multi-objetivo, entre ellos los algoritmos evolutivos multi-objetivo (*Multi-Objective Evolutionary Algorithms*, MOEAS) han sido profundamente usados para enfrentarse a estos problemas.

Son muchas las referencias sobre MOEAs en temas de clasificación, donde se han logrado grandes avances en los resultados [25; 48; 175; 188; 204]. Específicamente, si evaluamos su utilización en GP, paradigma en el que ha quedado demostrada su eficacia en la resolución de problemas mono-objetivo, podemos ver que su transferencia al dominio multi-objetivo obtiene soluciones comparables o mejores a las obtenidas usando GP estándar y con menor coste computacional [13; 54; 140; 149].

Las propuestas existentes para resolver los problemas desde una perspectiva MIL no tienen en cuenta el problema multi-objetivo y obtienen únicamente una solución que combina los diferentes objetivos para obtener un clasificador de alta calidad. Sin embargo, este enfoque no es satisfactorio de acuerdo al principio de optimalidad que se impone a las soluciones cuando se trabaja con múltiples objetivos contradictorios. Es bien conocido que en la presencia de múltiples y conflictivos objetivos, el problema de optimización ofrece un conjunto de soluciones óptimas, en lugar de dar una única solución. Esto es así porque ninguna de esas soluciones puede considerarse que mejor que la otra teniendo en cuenta todos los objetivos.

Por otro lado, las ventajas de los sistemas basados en G3P ya han sido detalladas en el capítulo anterior, con lo que no las repetiremos. Como resultado de todo lo discutido, se tiene por objetivo un primer estudio de los algoritmos multi-objetivo en el escenario de MIL. Estos algoritmos utilizan G3P, un paradigma de clasificación robusto capaz de generar reglas comprensibles y estarán basados en los enfoques multi-objetivo más ampliamente utilizados. Así, diseñamos e implementamos tres enfoques, el *Strength Pareto Grammar-Guided Genetic Programming* para MIL (SPG3P-MI) basado en *Strength Pareto Evolutionary Algorithm* (SPEA2) [245], Non-dominated Sorting Grammar-Guided Genetic Programming para MIL (NSG3P-MI) basado en *Non-dominated Sorting Genetic Algorithm* (NSGA2)[51] y *Multi-objective Genetic Local Search with Grammar-Guided Genetic Programming* para MIL (MOGLSG3P-MI) basado en *Multi-objective Genetic Local Search* (MOGLS)[104].

5.2. Algoritmos desarrollados

En esta sección trataremos los principales aspectos en los que se basan nuestras propuestas. Comenzaremos describiendo las métricas empleadas en la evaluación

de los clasificadores para centrarnos después en el desarrollo de los tres modelos desarrollados. Con respecto a la representación de los individuos, el método de inicialización y los operadores genéticos, se trata de la misma propuesta especificada en el capítulo 4, cuando se desarrolló el algoritmo G3P-MI.

5.2.1. Evaluación de los individuos

La función de evaluación debe medir la efectividad de los clasificadores, que es lo que representa cada individuo de la población.

En la sección 3.3.3 se han visto distintas medidas, tanto objetivas como subjetivas, para realizar una evaluación de la calidad de las reglas obtenidas. Nosotros consideramos dos medidas objetivas predictivas ampliamente utilizadas en problemas de clasificación, que son la sensibilidad (Se) y la especificidad (Es) [18; 189]. Ambas medidas han sido especificadas en la versión mono-objetivo en la sección 4.2.4, donde se realizaba una combinación de las mismas que consistía en el producto de ambas. En el caso multi-objetivo que nos ocupa serán tratadas como objetivos diferentes y contrapuestos.

Por ello, nuestros algoritmos va a intentar maximizar tanto la *sensibilidad* como la *especificidad*. Recordemos que existe una compensación entre ambas medidas debido a que evalúan características contradictorias, con lo que el aumento de cualquiera de ellas produce un decremento en la otra. Así, la sensibilidad nos dice cómo de bien predice la clase positiva y la especificidad hace lo equivalente con la clase negativa. De este modo, cuando incrementamos el valor de sensibilidad, la regla predice un mayor número de ejemplos positivos a expensas de clasificar como positivos ejemplos que pertenecen a la clase negativa. Un escenario similar nos encontramos cuando lo que incrementamos son los valores de especificidad a expensas de los de sensibilidad. Un valor de uno en ambas medidas representa una clasificación perfecta de cada una de las clases.

Otra medida muy interesante en clasificadores tiene que ver con la evaluación de la comprensibilidad, especificando si la información que revela el sistema es simple y sencilla de evaluar, aportando conocimiento del mismo. Hoy en día los requerimientos de precisión son tan importantes como los de comprensibilidad, con lo que está muy extendido que en el proceso de optimización se evalúe la precisión junto

con la sencillez de las reglas. En nuestra función de evaluación, no se ha incluido ninguna medida de simplicidad del modelo. El motivo para tomar esta decisión se debe a que nuestras propuestas consideran en el proceso evolutivo una restricción con respecto al tamaño de las soluciones, tanto en la inicialización como en las operaciones de recombinación generando reglas simples sin la consideración adicional de ninguna otra restricción en la evaluación de las mismas.

5.2.2. Algoritmo NSG3P-MI

La propuesta desarrollada, *Non-dominated Sorting Grammar-Guided Genetic Programming para MIL* (NSG3P-MI), está basada en el popular algoritmo (NSGA2)[51]. Se trata de un algoritmo elitista que implementa el concepto de frente del Pareto. Daremos una descripción del algoritmo para comprender su funcionamiento y luego detallaremos un esquema general del proceso evolutivo que sigue.

5.2.2.1. Descripción del funcionamiento

Para describir nuestro algoritmo, nos centraremos en los aspectos relacionados con la identificación de los frentes no dominados, mantenimiento de la diversidad y el proceso de evolución, basadas todas ellas del algoritmo NSGA2.

1. **Identificación de los frentes no dominados**

   El algoritmo se basa en la identificación de las soluciones por frentes de individuos no dominados, de manera que un frente f está formado por soluciones no dominadas por ninguno de ellas y dominadas por los frentes de $[0, f-1]$. El frente 0 será el que contenga las soluciones no dominadas por ningún otro individuo ni frente, formando la aproximación del frente de Pareto óptimo. Para realizar este procedimiento, para cada solución se calcula:

   a) Un contador n_p, que nos indica el número de soluciones que dominan a la solución p.

   b) Una lista S_p, que contiene el conjunto de soluciones que son dominadas por la solución p.

Todas las soluciones en el primer frente no dominado tendrán su contador de individuos que lo dominan a cero, $n_p = 0$. Para cada solución p con $n_p = 0$, se visita cada miembro q de su conjunto S_p, y estos individuos reducen su contador de individuos que lo dominan en uno, $n_p = n_p - 1$. De este modo, si alguno de estos individuos alcanzan en su contador de individuos que lo dominan el valor cero, $n_p = 0$, se pondrá en una lista separada, Q. Estos miembros pertenecen al segundo frente de no dominados. Ahora, el siguiente proceso es continuar con cada uno de los miembro de Q e identificar el tercer frente. Este proceso continúa hasta que se identifican todos los frentes de la población.

Para cada solución p en el segundo nivel o en un nivel superior de no dominancia, el valor de n_p debe ser menor de $N - 1$, siendo N el tamaño de la población. Así, cada solución p será visitada al menos $N - 1$ veces antes de que su contador de dominación llegue a ser cero. En este punto, la solución es asignada a un nivel de no dominancia y no será visitada de nuevo.

2. **Mantener la diversidad**

   Para evitar una convergencia prematura del algoritmo es deseable que se mantenga una buena diversidad de soluciones en el conjunto de soluciones obtenido. Para ello se utiliza una aproximación de la comparación cruzada (*crowded-comparison*) que soluciona dos dificultades en la que caen otros modelos:

   a) Mantener la diversidad de las soluciones depende del valor que se escoja para la métrica de distancia que se haya elegido para calcular la aproximación entre dos miembros de la población. Normalmente suele estar parametrizado el valor más grande que proporciona la métrica de las distancia entre cualesquiera dos soluciones.

   b) Debido a que cada solución debe ser comparada con todas las otras soluciones en la población, la complejidad de la función es $O(n^2)$.

   Este método no necesita ningún parámetro definido por el usuario para mantener la diversidad entre los miembros de la población, además de tener una mejor complejidad computación. Para describir esta mejora, primero definiremos una medida de *estimación de la densidad* y después se presentará el

operador de comparación cruzada (*crowded-comparison*). Para referirnos a él mantendremos la acepción en inglés.

a) Estimación de la densidad: para conseguir una estimación de la densidad de las soluciones circundantes a una solución concreta en la población, calculamos la distancia promedio entre las dos soluciones más próximas, la inmediatamente anterior y la posterior, dentro del frente de individuos ordenados, y se calcula las diferencias a lo largo de cada objetivo. Esta medida de distancia sirve como una estimación del perímetro del cubo formado por los vecinos más cercanos que actúan como los vértices (y es denominada, *crowding distance*).
La *crowding-distance* requiere almacenar la población en orden ascendente de magnitud para cada función objetivo. Las soluciones extremo (soluciones con los valores de la función más pequeños y más grandes) tendrá asignado un valor de distancia infinito. A todas las soluciones intermedias se les asignará un valor de distancia igual a la diferencia normalizada en valor absoluto de los valores de la función objetivo de las dos soluciones adyacentes. El valor final se calcula como la suma de la distancia calculada para cada uno de los objetivos.

b) El operador *crowded-comparison*: este operador guía el proceso de selección hacia un frente uniformemente expandido del Pareto. Se asume que cada individuo i de la población tiene 2 atributos:

1) frente de no dominados al que pertenece (i_{rank})
2) *crowding distance* ($i_{distance}$)

Ahora definimos un orden parcial $\prec_n$

$$
\begin{aligned}
& i \prec_n j, \; if(i_{rank} < j_{rank}) \\
& or((i_{rank} = j_{rank}) \\
& and(i_{distance} > j_{distance})
\end{aligned}
$$

esto es, entre dos soluciones en frentes diferentes, se prefiere la solución con un valor más bajo del frente. Por otro lado, si ambas soluciones

pertenecen al mismo frente, entonces nosotros preferimos la solución que se localiza en una región menos poblada (aquella cuya *crowding distance* sea mayor.

5.2.2.2. Proceso evolutivo

Definidos los conceptos principales en los que se basa el algoritmo, en este apartado se describirá el proceso evolutivo de NSG3P-MI. En la figura 5.1, se puede ver un diagrama de la secuencia de eventos que sigue el algoritmo. A continuación, se detallarán los principales pasos que sigue en algoritmo.

Inicialmente se crea una población aleatoria, P_0, dicha población se ordena en base a la no dominancia, a cada solución se le asigna un *fitness* igual a su nivel de no dominancia, siendo 1 el mejor nivel, 2 el siguiente mejor nivel, etc. De este modo, se busca la minimización del *fitness*. En la primera iteración se emplea la selección por torneo binario y los operadores de recombinación y mutación para crear una población de descendientes, Q_0, de tamaño N. A continuación, se introduce la población actual con las mejores soluciones no dominadas de la población previamente encontrada, el procedimiento es diferente después de la generación inicial. Describiremos una de las iteraciones intermedias del algoritmo:

- *Primero* se genera una población combinada de la población actual y de los descendientes $R_t = P_t \bigcup Q_t$. La población R_t será de tamaño $2xN$. La población R_t es ordenada de acuerdo al factor de dominancia de los individuos. Ya que todos los miembros de la población están incluidos en R_t, el elitismo está asegurado.

- *Segundo*, las soluciones pertenecientes al conjunto de no dominadas F_1, son las mejores soluciones en la población combinada y deben ser más enfatizadas que cualquier otra solución de la población combinada. Si el tamaño de F_1 es más pequeño que N, se elegirán todos los elementos de F_1 para formar la nueva población P_{t+1}, los miembros de la población P_{t+1} se eligen desde los frentes no-dominados en orden de pertenencia al frente. Así, las soluciones del conjunto F_2 son elegidas las siguientes, seguidas de las soluciones de F_3, y así. Este procedimiento continúa hasta que ningún conjunto más pueda unirse a la población (ya que se ha alcanzado la dimensión N).

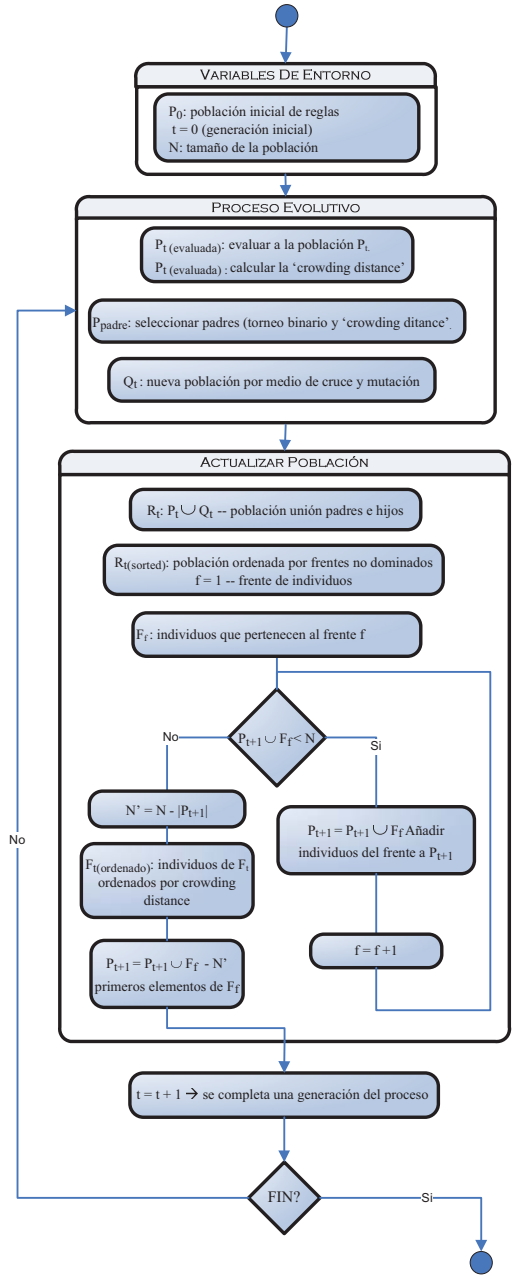


Figura 5.1: Proceso evolutivo del algoritmo NSG3P-MI

En general, la cantidad de soluciones en todo el conjunto de F_1 a F_l suele ser más grande que el tamaño de la población, es decir, el último frente que se incluye normalmente no se ajusta al tamaño de la población sino que tiene más elementos. Para elegir exactamente N miembros de la población, se ordenan las soluciones del último frente F_l usando el operador de *crowded-comparison* en orden decreciente y elegimos las mejores soluciones necesarias hasta completar los N individuos de la población.

La nueva población P_{t+1} de tamaño N es ahora usada por la selección, cruce y mutación para crear la nueva población Q_{t+1} de tamaño N, es importante darse cuenta que usamos el operador de selección por torneos pero el criterio de selección ahora está basado en el operador de *crowded-comparison*.

Para optimizar el algoritmo, podemos ver que sólo es necesario ordenar la población por frentes hasta que se alcance el frente con el que se supere o iguale el tamaño de la nueva población, N. De esta forma, tan pronto como el procedimiento de ordenación haya encontrado el número suficiente de frentes para tener N miembros en P_{t+1}, no tenemos porque continuar con el procedimiento de ordenación.

Para introducir diversidad entre las soluciones no dominadas se emplea el procedimiento de *crowded-comparison*, que se usa en la selección por torneos y durante la fase de reducción de la población, la diversidad es introducida ya que las soluciones compiten con su *crowding-distance* que se trata de una medida de densidad de las soluciones en su vecindario, de forma que las soluciones pertenecientes al mismo frente, nos interesarán aquellas que tengan menos vecinos próximos. Además esta técnica no necesita ningún parámetro externo. Aunque la *crowding-distance* se calcula en el espacio de las funciones objetivo, puede también ser implementado si se desea en el espacio de los parámetros.

5.2.3. Algoritmo SPG3P-MI

Strength Pareto Grammar-Guided Genetic Programming para MIL (SPG3P-MI) está basado en el algoritmo SPEA2 [245]. Se trata también de un algoritmo elitista que emplea una población externa. Describiremos las principales características del algoritmo para comprender su funcionamiento y luego detallaremos un esquema general del proceso evolutivo que sigue.

5.2.3.1. Descripción del funcionamiento

Las principales particularidades que tiene el algoritmo son:

- Usa una estrategia de asignación de *fitness* que incorpora la densidad de información.
- El tamaño de la población externa es fijo, es decir, si el número de individuos no dominados es menor que el tamaño predeterminado de esta población se rellena con individuos dominados.
- La técnica de *clustering* que se emplea cuando el frente de individuos no dominados excede al límite del archivo, consiste en un método de truncamiento que garantiza la selección de soluciones que estén lo más dispersas posibles sin perder los puntos extremos.
- Sólo los miembros de la población externa participan en el proceso de selección de padres.

Para describir el algoritmo SG3P-MI nos centraremos en los aspectos relacionados con la asignación del *fitness* y la actualización de la población externa.

1. **Asignación del fitness**

   En el cálculo del *fitness* se tienen en cuenta tanto las soluciones dominadas como las no dominadas. Concretamente, a cada individuo de la población externa, $\bar{P}_t$, y de la población actual, P_t, se le asigna un valor de proporción, $S_{(i)}$, que representa el número de soluciones dominadas:

   $$S_{(i)} = |j \mid j \in (P_t + \bar{P}_t) \ \wedge (i \succ j)|$$

   Donde $|\cdot|$ denota la cardinalidad del conjunto, $+$ indica el símbolo de unión y el símbolo $\succ$ corresponde a la relación de dominancia del Pareto. Bajo la bases de los valores de S, el rango del *fitness* del individuo, R_i, del individuo i se calcula:

   $$R_i = \sum_{j \in \bar{P}_t + P_t,\ j \succ i} S_{(j)}$$

El rango del $fitness$ se determina por las proporción de los individuos que le dominan de la población externa y de la población actual. El $fitness$ se debe minimizar, de forma que $R_i = 0$ corresponde a individuos no dominados, mientras un valor alto de R_i significa que i es dominado por muchos individuos.

Aunque la asignación del rango del $fitness$ proporciona un mecanismo de ordenación basado en el concepto de dominancia de Pareto, no es apropiado cuando hay muchos individuos que no son dominados por otros. Para solucionar esa situación, se incorpora al $fitness$ un valor adicional que nos informa sobre la densidad, de este modo podemos discriminar entre individuos que tienen idénticos valores del rango del $fitness$. La técnica de estimación de la densidad que se ha empleado es una adaptación del método de los k-vecinos más cercanos, donde la densidad a cualquier punto es una función decreciente de la distancia al punto del *k-ésimo* vecino más cercano. En concreto, se ha tomado la inversa de la distancia a los k-vecinos más cercanos. El procedimiento sería: para cada individuo i se calcula, en el espacio de objetivos, la distancia a todos los individuos j de la población externa y de la población actual, y se almacenan en una lista. Después de ordenar la lista en orden creciente, se selecciona el k-ésimo elemento que nos dará la distancia buscada, esta distancia la denotaremos por σ_i^k.

Normalmente usamos k igual a la raíz cuadrada del tamaño de la población actual y de la población externa, así $k = \sqrt{N + \bar{N}}$. La densidad, D_i, se corresponde con el individuo i-ésimo y se define como

$$D_i = \frac{1}{\sigma_i^k + 2}$$

De este modo, se cumple que $D_i > 0$ y que $D_i < 1$. Finalmente, el *fitness* F_i se genera añadiendo a D_i el valor del rango del *fitness*, R_i:

$$F_i = R_i + D_i$$

2. **Actualización de la población externa**

La operación de actualización de la población externa se caracteriza porque:

a) El número de individuos contenidos en el archivo es constante todo el tiempo.

b) El operador de truncamiento previene que las soluciones extremas sean eliminadas.

Durante el proceso de selección, el primer paso es copiar todos los individuos no dominados, aquellos que tienen un *fitness* menor de 1, de la población externa y de la población actual a la población externa de la siguiente generación.

$$\bar{P}_{t+1} = i \mid i \in (P_t + \bar{P}_{t+1}) \ \wedge \ (F_i < 1)$$

Donde: P_i que representa la población de la generación i, $\bar{P}_i$ representa la población externa de la generación i, N es el tamaño de la población y $\bar{N}$ es el tamaño de la población externa:

Si los individuos no dominados se ajustan exactamente al archivo ($\bar{P}_{t+1} = \bar{N}$) el proceso de selección termina. En caso contrario, pueden darse dos situaciones:

- El archivo es demasiado pequeño, ($\bar{P}_{t+1} < \bar{N}$). En este caso los mejores $\bar{N} - \bar{P}_{t+1}$ individuos dominados por la población unión formada por la población externa anterior, $\bar{P}_t$ y la población actual P_t se copian a la nueva población externa. Esto se puede implementar ordenando el conjunto $P_t + \bar{P}_t$ de acuerdo al valor de *fitness* y copiando los primeros $\bar{N} - \bar{P}_{t+1}$ individuos con $F_i \geq 1$ de la lista ordenada a $\bar{P}_{t+1}$.
- El archivo es demasiado grande, ($\bar{P}_{t+1} > \bar{N}$). En este caso, cuando el tamaño de la población externa excede de $\bar{N}$, se invoca a un procedimiento de truncamiento que iterativamente borra los individuos de $\bar{P}_{t+1}$ hasta que $\bar{P}_{t+1} = \bar{N}$. En cada iteración el individuo i que se elimina de la población es el que cumple que $i \leq_d j$ para todo $j \in \bar{P}_{t+1}$ donde:

$$\begin{aligned} i \leq_d j :\Leftrightarrow \forall\ 0 < k < |\bar{P}_{t+1}| \ &: \sigma_i^k = \sigma_j^k \ \vee \\ \exists\ 0 < k < |\bar{P}_{t+1}| \ &: \left[(\forall\ 0 < l < k \ : \sigma_i^l = \sigma_j^l) \wedge \sigma_i^k < \sigma_j^k\right] \end{aligned}$$

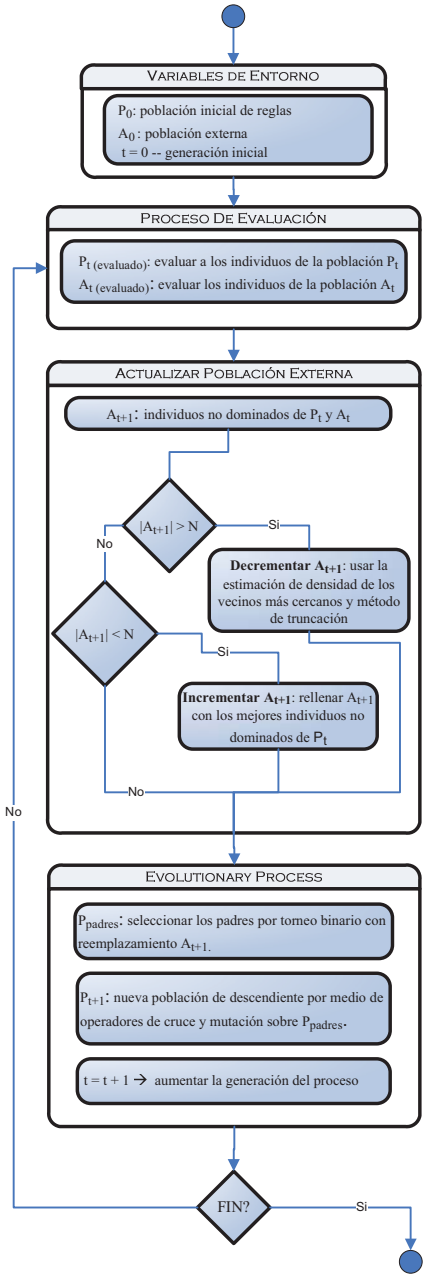


Figura 5.2: Proceso evolutivo del algoritmo SPG3P-MI

siendo σ_i^k la distancia de i a su k-*ésimo* vecino más cercano en $\bar{P}_{t+1}$. En otras palabras, en cada iteración, se elige el individuo que tiene la mínima distancia a otro individuo. Si hay varios individuos con distancia mínima, se considerará la segunda distancia más pequeña y así sucesivamente.

5.2.3.2. Proceso evolutivo

En este apartado se describirá mediante un diagrama la secuencia de eventos del algoritmo SPG3P-MI (se puede ver en la figura 5.2). A continuación se mostrarán los principales pasos de los que consta una evolución de este algoritmo.

Este algoritmo sigue un esquema evolutivo donde inicialmente se genera la población inicial y se crea un archivo externo vacío. Esta población inicial se evalúa de acuerdo a las funciones objetivo que se desean optimizar. Tras la evaluación se seleccionan todos los individuos no dominados y se copian en la población externa. Si el tamaño para la población externa excede del tamaño fijado, se reduce por medio del operador de truncamiento. Si por el contrario es menor que el tamaño fijado de la población externa, entonces se rellena con los mejores individuos dominados. La nueva población es creada por medio de un torneo binario con reemplazamiento y operadores de cruce y mutación sobre la población que es unión de la población actual y la externa. Este proceso de repite hasta que se alcance un número máximo de generaciones o se alcance una solución deseada.

5.2.4. Algoritmo MOGLSG3P-MI

Multi-objective genetic local search with Grammar-Guided Genetic Programming para MIL (MOGLSG3P-MI) está basado en el algoritmo MOGLS [104]. Se trata de un algoritmo clasificado dentro de los algoritmos de segunda generación, elitista que emplea una población externa. Utiliza una hibridación de metaheurísticas con elementos de varios métodos, combinando los AEs con procesos de búsqueda local que se centran en una explotación de determinadas áreas.

5.2.4.1. Definiciones previas

Antes de las explicar el funcionamiento del algoritmo son necesarias unas definiciones previas que presentamos a continuación.

Consideraremos el vector $z = [z_1, ..., z_n]$ como el valor de las funciones objetivo que se desean evaluar $f_1(x) = z_1 ... f_j(x) = z_j, x \epsilon D$, donde $x = [x_1 .... x_j]$ es un vector de variables de decisión y D es el conjunto de soluciones factibles. Podemos definir los siguiente conceptos que emplearemos en este algoritmo:

- *La imagen de una solución x en el espacio de objetivos* es un conjunto de puntos $z^x = [z_1^x, ...z_j^x] = f(x)$, tales como $z_j^x = f_j(x), j = 1.....J$. La imagen del conjunto D en el espacio de objetivos es un conjunto Z compuesto de puntos alcanzables, es decir puntos que son imágenes de soluciones factibles.

- Una solución $x \epsilon D$ es eficiente (Pareto optimo) si no hay ningún $x^{'} \epsilon D$ que domine a x. El punto que es una imagen de una solución eficiente se llama no dominado. El conjunto de todas las soluciones eficientes se llama conjunto eficiente. La imagen del conjunto eficiente en el espacio objetivo se llama conjunto no dominado o frente del Pareto.

- Una aproximación al conjunto no dominado es un conjunto A de puntos y soluciones correspondientes tales que $\neg \exists\ z_1,\ z_2 \epsilon A$ tales que $z^1 \succ z^2$, es decir el conjunto A está compuesto de puntos no dominados mutuamente.

- El punto z^* que está compuesto de los mejores valores de las funciones objetivo disponibles se llama *punto ideal*.

$$z_j^* = max\ f_j(x)\ /\ x \in\ A\ j = 1, ..., J$$

- El punto z^{**}(A) representa una aproximación del *punto ideal* sobre el conjunto A, es decir:

$$z_j^{**} = max z_j / z \in A\ j = 1, ..., J$$

- El punto z_* está compuesto de los peores valores de las funciones objetivo disponibles, se llama *punto nadir*.

- Los factores del rango de ecualización se define del siguiente modo:

$$\Pi_j = \frac{1}{R_j}, j = 1, ..., J$$

Donde R_j es el rango del objetivo j en el conjunto N y D y A, los valores de la función objetivo multiplicado por el factor de rango de ecualización se llama valores de la función objetivo normalizados.

- Una función de utilidad $u : \Re^J \rightarrow \Re$, mapea cada punto en el espacio de objetivos dentro de un valor de utilidad.
  Una función de utilidad, u, es compatible con la relación de dominación si y solo si $\forall z^1, z^2 \epsilon R^J z^1 \succ z^2 \rightarrow u(z^1) \geq u(z^2)$. El conjunto de todas las funciones de utilidad que son compatibles con la relación de dominancia se denota por U_c. Dos tipos de funciones empleados son:

  - Funciones de utilidad con pesos Tchebycheff se definen del siguiente modo:

    $$u_\infty(z, z^*, \wedge) = -max\ \lambda_i\ (z_j^* - z_j)$$

    Donde $\wedge = [\wedge_1 \ldots \wedge_J]$ es un vector de pesos tales que $\wedge_j \geq 0\ \forall j$.

    La minimización de la función escalar de pesos Tchebycheff corresponde a un problema min-max. El problema puede ser, sin embargo, transformado a uno del siguiente modo:
    minimizar α s.t. $\alpha \geq \lambda_i(z_j^0 - z_j) = \lambda_i(z_j^0 - f_j(x))\ j = 1, ..., J,\ x \in D$
    Donde cada función escalar de pesos Tchebycheff tiene al menos un óptimo global que pertenece al conjunto de soluciones eficientes. Para cada solución eficiente x existe una función escalar de pesos Tchebycheff tal que x es un óptimo global de s.

  - Funciones de utilidad con pesos lineales se definen del siguiente modo:

    $$u_1(z, \wedge) = \sum_{j-1}^{J} \lambda_i z_j.$$

    El vector de pesos con el que trabajamos tiene las siguiente condiciones:

    $$\forall\ j\ \lambda_j \geq 0,\ \sum_{j-1}^{J} \lambda_j = 1.$$

    Es llamado vector de pesos normalizados.

5.2.4.2. Descripción del funcionamiento

El algoritmo MOGLSG3P-MI se basa en la idea de optimización simultánea de todas las posibles funciones de utilidad [103]. El mismo proceso realizado en la descripción de los algoritmos anteriores se aplicará a este algoritmo, así detallaremos las características del algoritmo para describir posteriormente su proceso evolutivo.

- Filosofía del algoritmo. Inicialmente se genera un conjunto de soluciones y en cada iteración del método se emplean aleatoriamente distintas funciones de pesos Tchebycheff o funciones de pesos lineales.

  En cada etapa, se selecciona una muestra de las mejores soluciones del conjunto de soluciones generadas anteriormente, que se trata como una población temporal. De esta población temporal se seleccionan un par de soluciones aleatoriamente y se recombinan. Después de la recombinación se realiza una búsqueda local.

- Actualizar la población externa. Actualizar el conjunto de soluciones no dominadas, PE, con una nueva solución x, consiste en:

  - añadir x a PE si ninguna solución en PE domina a x,
  - borrar de PE todas las soluciones dominadas por x.

- Establecer los valores de S y N. Para establecer el parámetro S, el número de soluciones iniciales y N, el tamaño de una población temporal dada, se pueden emplear diferentes heurísticas. La idea es encontrar un conjunto de soluciones que para cualquier función de utilidad pueda contener N mejores soluciones de la misma calidad.

- Funciones de utilidad. El algoritmo usa funciones de utilidad con pesos lineales o pesos Tchebycheff, y aleatoriamente selecciona un vector de pesos del conjunto de vectores de pesos normalizados que será por el que se optimice la soluciones en esa iteración. Los vectores peso se indican con una distribución de probabilidad uniforme $p(\wedge)$, es decir una distribución para la cual:

$$\forall\ \Psi' \subseteq \Psi \int_{\wedge \in \Psi'} p(\wedge) d \wedge \ / \ \int_{\wedge \in \Psi} p(\wedge) d \wedge = \vee(\Psi') / \vee(\Psi)$$

Donde Ψ y Ψ' denotan el conjunto de todos los pesos normalizados y subconjuntos de ellos, respectivamente; $\vee(\Psi)$ y $\vee(\Psi')$ son el hiper-volumen euclídeo de Ψ y Ψ', respectivamente. En otras palabras, la probabilidad de que un vector pesos pertenezca a Ψ' es proporcional al hiper-volumen de Ψ'.
Se utiliza como punto de referencia una aproximación del punto ideal, compuesto de los mejores valores conocidos de cada objetivo. Este punto de referencia cambia (mejora) durante la ejecución del metodo. Para seleccionar aleatoriamente la función escalar con la que se va a optimizar las soluciones, el vector de pesos normalizados se elige aleatoriamente mediante el procedimiento que se muestra a continuación, que garantiza que el proceso se realice con una distribución uniforme aleatoria $p(\wedge)$:

$$\lambda_1 = 1 - \sqrt[J-1]{rand()}$$
$$\cdots$$
$$\lambda_j = (1 - \sum_{l-1}^{j-1} \lambda_l)(1 - \sqrt[J-1-j]{rand()})$$
$$\cdots$$
$$\lambda_j = 1 - \sum_{l-1}^{J-1} \lambda_l$$

5.2.4.3. Proceso evolutivo

En esta sección se describe el proceso evolutivo de MOGLSG3P-MI, para ello se mostrará un diagrama con la secuencia de eventos del algoritmo (mostrado en la figura 5.3). A continuación se detalla los principales pasos del algoritmo, el cual se describe dividiéndolo en dos fases:

- **Fase I**
  - Seleccionar el número de funciones de utilidad (por ejemplo 20).
  - Para cada función de utilidad.
    - Generar un número de óptimos locales de la función de utilidad (por ejemplo 10).
    - Calcular el valor medio del óptimo local sobre otros óptimos locales generados.

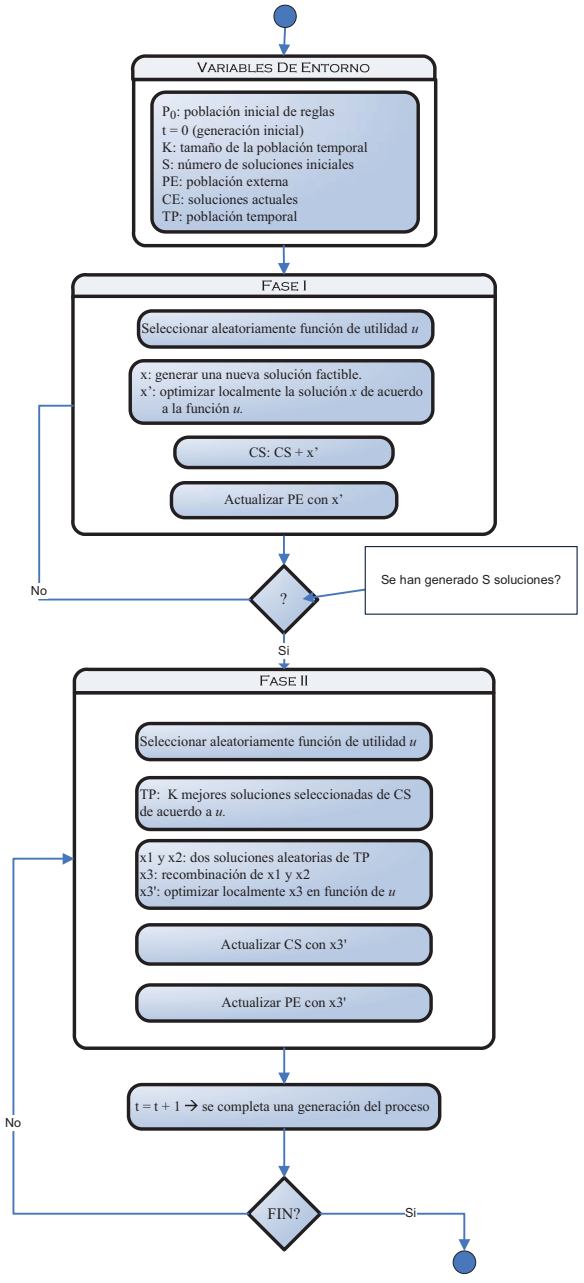


Figura 5.3: Proceso evolutivo del algoritmo MOGLSG3P-MI

- Calcular AV_1, valor medio del óptimo local, sobre las funciones de utilidad seleccionadas.

- **Fase II**

  - Para el conjunto actual de soluciones $CS = 0$, repetir:
    - Elegir aleatoriamente una función de utilidad desde el conjunto de todas las posibles funciones.
    - Encontrar el óptimo local de la función de utilidad y añadirlo a CS.
    - Para cada función de utilidad seleccionada en la Fase I encontrar las N mejores soluciones de CS.
    - Calcular AV_2, valor medio de las N mejores soluciones sobre la función de utilidad seleccionadas.
    - Todas las optimizaciones locales sobre el algoritmo comienzan desde soluciones aleatorias.

La primera fase es equivalente a la generación de un número de poblaciones de partida de un único objetivo empleando búsqueda local genética (*Genetic Local Search*, GLS). En la segunda fase, se optimiza aleatoriamente un conjunto de óptimo locales para contener las N mejores soluciones de la misma calidad media sobre las funciones de utilidad seleccionadas.
En cada iteración, se construye una población temporal compuesta de K soluciones, que son las mejores entre las soluciones conocidas sobre la función escalar actual. Sobre esta población temporal, se seleccionan dos soluciones diferentes con probabilidad uniforme y se recombinan, realizándose de este modo la recombinación de dos soluciones que son buenas en la función escalar con la que se está trabajando. La salida de MOGLS es una aproximación al conjunto no dominado (población externa), compuesto de todas los puntos potencialmente no dominados.

5.3. Experimentación y análisis de resultados

En esta sección, se realizan experimentos sobre distintos dominios de aplicación que consideran ocho conjuntos de datos y diecinueve algoritmos. Los algoritmos con los

Tabla 5.1: Parámetros de configuración del algoritmo NSG3P-MI

Algoritmo NSG3P-MI	
Parámetros	Valores
Tamaño de la población	1000
Tamaño de la población externa	100
Número de generaciones	150
Probabilidad de cruce	95 %
Probabilidad de mutación	80 %
Profundidad máxima del árbol	50
Método de selección de padres	*Selector por torneos (toreno=2)*

que se realiza la comparación se encuentran definidos en el framework de WEKA [211] que considera algunas de las propuestas más significativas en este campo de investigación.

Primero se llevará a cabo un estudio de las diferentes propuestas multi-objetivo comparándolas mediante las principales medidas que evalúan las características de los frentes de Pareto. A continuación, las propuestas multi-objetivo son comparadas con otros algoritmos previos de MIL que consideran el problema como un problema mono-objetivo y se analizarán los resultados obtenidos.

5.3.1. Parámetros de configuración

Los modelos multi-objetivo para MIL que se han propuesto han sido implementados en el software de JCLEC [199] y sus principales parámetros de configuración se muestran en las tablas 5.1, 5.2 y 5.3. Todos los experimentos con AEs son repetidos diez veces con semillas diferentes y los resultados medios son los considerados en las comparaciones con el resto de técnicas.

Con respecto a los algoritmos utilizados en la comparación, se puede ver la configuración utilizada de los mismos en la sección 4.3.2. En este no caso no llevamos a cabo un estudio de las diferentes configuraciones utilizadas en los algoritmos. Al tratarse de los mismos conjuntos de datos y algoritmos utilizados con la versión

Tabla 5.2: Parámetros de configuración del algoritmo SPG3P-MI

Algoritmo SPG3P-MI	
Parámetros	Valores
Tamaño de la población	1000
Tamaño de la población externa	100
Número de generaciones	150
Probabilidad de cruce	95 %
Probabilidad de mutación	80 %
Profundidad máxima del árbol	50
Método de selección de padres	*Selector por torneos (torneo=2)*

Tabla 5.3: Parámetros de configuración del algoritmo MOGLSG3P-MI

Algoritmo MOGLSG3P-MI	
Parámetros	Valores
Tamaño de la población	80
Tamaño de la población externa	50
Tamaño de la población temporal	50
Número de generaciones	7500
Probabilidad de cruce	100 %
Profundidad máxima del árbol	50
Método de selección de padres	*Selector Aleatorio*
Método de búsqued local	*Método de Solis & West*

mono-objetivo, se utilizará directamente la misma configuración más beneficiosa obtenida en el estudio previo y que se puede consultar en la sección 4.3.4.1.

5.3.2. Comparación de las estrategias multi-objectivo

En esta sección, comparamos las diferentes técnicas multi-objetivo implementadas para trabajar en un escenario de MIL. Primero, realizamos una comparación cuantitativa del funcionamiento de los algoritmos multi-objetivo para evaluar el POF

obtenido por cada técnica. Después se mostrarán los resultados de precisión, sensibilidad y especificidad para cada conjunto de datos y un test no paramétrico nos determinará si hay diferencias significativas entre los algoritmos.

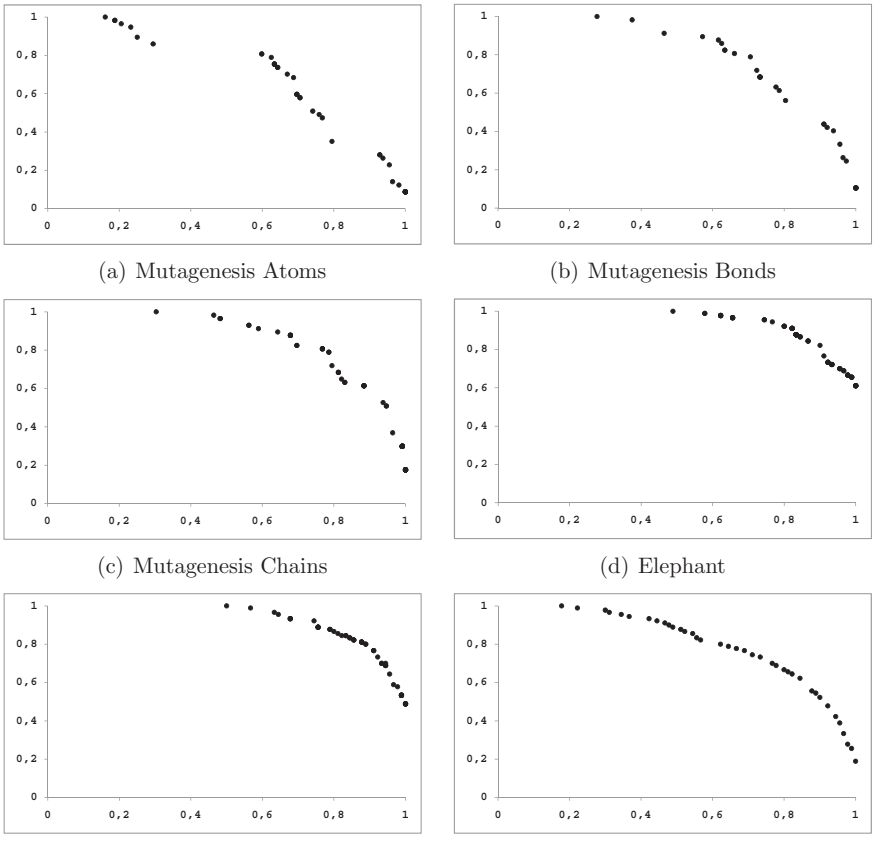
Todos los conjuntos de datos utilizados han sido particionados usando validación cruzada con diez particiones [162]. Las particiones se construyen sobre bolsas, en las que cada instancia en una bolsa aparece en la misma partición, se trata de las mismas particiones utilizadas en la experimentación realizada con el modelo G3P-MI.

5.3.2.1. Análisis de los POF obtenidos por las estrategias multi-objetivo

Una muestra de los diferentes Paretos obtenidos en algunos de los problemas con los que se ha trabajado puede verse en las figuras 5.4 (para el método SPG3P-MI), 5.5 (para el método NSG3P-MI) y 5.6 (para el método MOGLSG3P-MI). Observando estas figuras se puede apreciar que se obtienen varias soluciones con diferente balance entre sensibilidad y especificidad en una única ejecución, uno de los principales objetivos que se pretendía alcanzar con estos modelos. A simple vista, los frentes de los Paretos se encuentran bien distribuidos y con las soluciones equiespaciadas. No obstante, se realizará un análisis empleando diferentes métricas que nos permitan evaluar con mayor precisión los POFs que se obtiene.

Se han propuesto muchas medidas que evalúan diferentes características de los POFs, tal y como se ha visto en la sección 3.4.4. De las diferentes métricas comentadas, en este trabajo consideraremos tanto medidas individuales, que sirven para medir la calidad de un conjunto Pareto, como métricas cuyo fin es la comparación entre algoritmos y que permiten comparar el Pareto obtenido por un algoritmo con el generado por otro. Para el primer propósito se seleccionará el *espaciado* y el *hiper-volumen* que nos permite medir la distribución del POF así como el área que cubre. Para el segundo fin, emplearemos el *cubrimiento de dos conjuntos* para evaluar los POF de las diferentes técnicas y poder así comparar las soluciones que pertenecen a los distintos frentes.

Los resultados medios de estas métricas aplicados sobre los diferentes conjuntos de datos estudiados se muestran en la Tabla 5.4. Las métricas utilizadas nos permiten considerar distintas características deseables de los POF obtenidos.



(a) Mutagenesis Atoms

(b) Mutagenesis Bonds

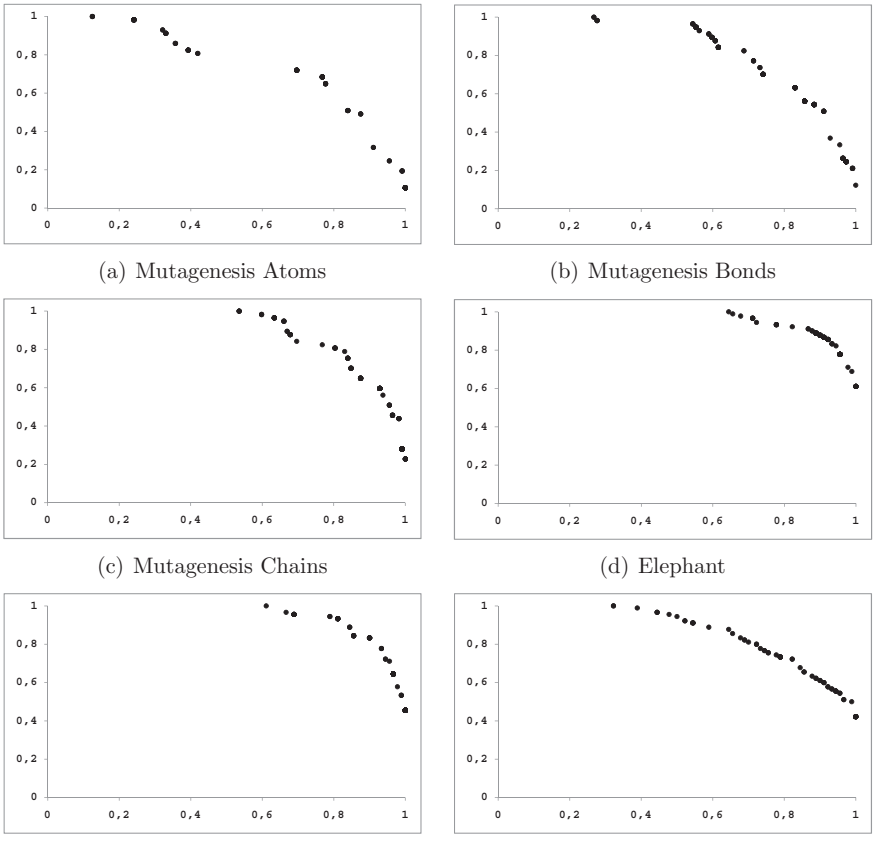
(c) Mutagenesis Chains

(d) Elephant

(e) Tiger

(f) Fox

Figura 5.4: POF generado por el algoritmo SPG3P-MI



(a) Mutagenesis Atoms

(b) Mutagenesis Bonds

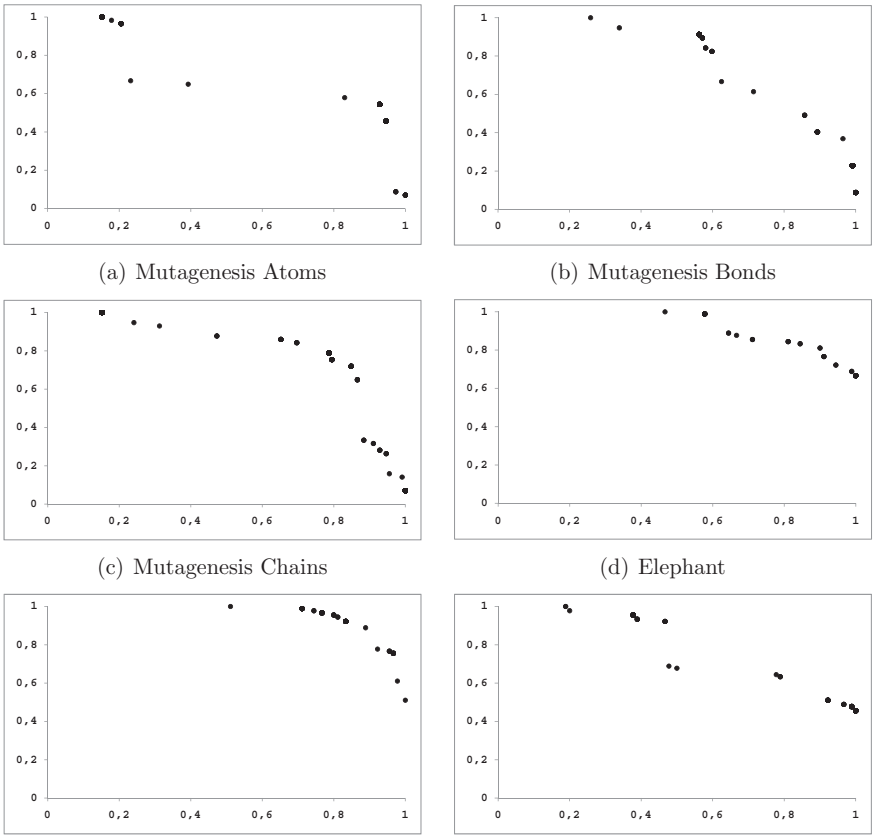
(c) Mutagenesis Chains

(d) Elephant

(e) Tiger

(f) Fox

Figura 5.5: POF generado por el algoritmo NSG3P-MI



(a) Mutagenesis Atoms

(b) Mutagenesis Bonds

(c) Mutagenesis Chains

(d) Elephant

(e) Tiger

(f) Fox

Figura 5.6: POF generado por el algoritmo MOGLSG3P-MI

Tabla 5.4: Análisis de la calidad de los POFs considerando los valores medios para todos los conjuntos de datos estudiados

Comparación POFs				
Algoritmo	Hipervolumen (HV)	Espaciado (S)	Cubrimiento De 2 Conjuntos (CS)	
MOGLSG3P-MI	0.844516	0.016428	CS(MOGLSG3P-MI,NSG3P-MI)	0.357052
			CS(MOGLSG3P-MI,SPG3P-MI)	0.430090
NSG3P-MI	0.890730	0.007682	CS(NSG3P-MI,MOGLSG3P-MI)	0.722344
			CS(NSG3P-MI,SPG3P-MI)	0.776600
SPG3P-MI	0.872553	0.012290	CS(SPG3P-MI,MOGLSG3P-MI)	0.508293
			CS(SPG3P-MI,NSG3P-MI)	0.235222

La métrica del *espaciado* [39] describe la difusión del conjunto de no dominados. De acuerdo a los resultados obtenidos, el POF del algoritmo NSG3P-MI tiene las soluciones espacias de forma más homogénea que los otros algoritmos.

Otra métrica considerada es el indicador del hiper-volumen [39] definido como el área de cubrimiento del conjunto de no dominados con respecto al espacio objetivo. Los resultados muestran que las soluciones no dominadas de NSG3P-MI tienen el valor más alto, lo que indica que cubren más área que las otras técnicas.

Finalmente, se evalúa el cubrimiento de dos conjuntos [39]. Esta métrica nos permite comparar las soluciones que contienen dos POFs, utilizando para comparar las diferentes soluciones el concepto de no dominancia. Los resultados nos muestran que NSG3P-MI obtiene los valores más altos cuando es comparada con las otras técnicas. Esto indica que las soluciones del POF generado por NSG3P-MI dominan en mayor porcentaje a las soluciones no dominadas de los POFs de las otras propuetas.

Teniendo en cuenta todos los resultados obtenidos en las diferentes métricas, se puede decir que NSG3P-MI consigue una mejor aproximación del POF que las otras técnicas.

Una vez realizado este estudio, en la siguiente sección analizaremos los valores medios de exactitud, sensibilidad y especificidad para los diferentes conjuntos de

Tabla 5.5: Resultados experimentales de la comparación de las propuestas multi-objetivo

Algoritmo	MOGLSG3P-MI			NSG3P-MI			SPG3P-MI		
	Acc	*Se*	*Sp*	*Acc*	*Se*	*Sp*	*Acc*	*Se*	*Sp*
Elephant	0.8900	0.8700	0.9100	0.9400	0.9400	0.9400	0.9250	0.9400	0.9100
Tiger	0.8850	0.9400	0.8300	0.9350	0.9200	0.9500	0.9200	0.9200	0.9200
Fox	0.7600	0.7800	0.7400	0.7800	0.8600	0.7000	0.8350	0.8900	0.7800
MutAtoms	0.8421	0.9385	0.6333	0.9158	0.9462	0.8500	0.8790	0.9308	0.7667
MutBonds	0.8421	0.9077	0.7000	0.8737	0.9231	0.7667	0.8684	0.9308	0.7333
MutChains	0.8737	0.9462	0.7167	0.9211	0.9462	0.8667	0.9053	0.9000	0.9167
Musk1	0.9778	0.9600	1.0000	1.0000	1.0000	1.0000	0.9667	0.9800	0.9500
Musk2	0.9400	0.9500	0.9333	0.9301	0.9607	0.9095	0.9400	0.9750	0.9167
RANKING	2.6875	2.4375	2.3750	1.3750	1.6875	1.6875	1.9375	1.8750	1.9375

datos que obtienen estas propuestas. Para ello, la solución que contiene mejor equilibrio en las medidas de sensibilidad y especificidad es la solución seleccionada del Pareto para ser aplicada a los test de evaluación. Estas soluciones son mostradas y evaluadas en la siguiente sección 5.3.2.2.

5.3.2.2. Análisis de la calidad de las soluciones de las estrategias multi-objetivo

En esta sección se realizará una comparativa entre las calidad de los clasificadores obtenidos por las estrategias multi-objetivo. La tabla 5.5 muestra los resultados medios de exactitud, sensibilidad y especificidad junto con los valores de *ranking* para cada una de las medidas considerando los diferentes conjuntos de datos. Los valores de *ranking* nos permiten conocer qué algoritmos obtienen los mejores resultados considerando todos los conjuntos de datos. De este modo, el algoritmo con el valor más cercano al 1 indica el mejor algoritmo sobre la mayoría de los conjuntos de datos.

Los resultados del test de Friedman, en la Tabla 5.6, determinan que no hay diferencias significativas para las medidas de sensibilidad y especificidad, y sí las hay para la precisión. Un test a posteriori, el test de Bonferroni-Dunn, se usó para determinar cuáles de los algoritmos puede ser considerado significativamente peor. Los resultados muestran, con un nivel de confianza del 95 %, que todos los algoritmos

Tabla 5.6: Resultados del test de Friedman para comparación de las propuestas multi-objetivo

Test de Friedman			
Medida	Valor de Friedman	$\chi^2(\rho = 0{,}1)$	Conclusión
Exactitud	6.9375	3.219	Rechazar la hipótesis nula
Sensibilidad	2.4375	3.219	Aceptar la hipótesis nula
Especificidad	1.9375	3.219	Aceptar la hipótesis nula

con un *ranking* mayor de 2.496 para precisión obtienen resultados significativamente peores que el algoritmo de control (el algoritmo de control es el algoritmo con el valor de *ranking* más bajo, en este caso NSG3P-MI).

Este estudio estadístico nos determina que MOGLSG3P-MI puede ser considerado peor que las otras propuestas con respecto a la precisión, aunque la diferencia es muy baja e incluso si aumentamos el nivel de confianza no habría tal diferencia.

Como conclusión, el algoritmo NSG3P-MI obtiene los mejores valores en todos los problemas para todas las medidas ya que obtiene los valores de rango más bajos. Además logra los mejores Paretos con respecto a las diferentes medidas consideradas.

5.3.3. Comparación con otras propuestas

En esta sección, mostraremos una comparación general entre las técnicas multi-objetivo desarrolladas con otras técnicas que se han desarrollado anteriormente, que consideran el problema como un problema mono-objetivo. En la comparación consideraremos ejemplos de los diferentes paradigmas, que se han comentado en la sección 4.3.2 y cuya configuración se estableció en la sección 4.3.4.1 para los mismos problemas. Los algoritmos considerados son: *Métodos basados en Diverse Density*: MIDD [126], MIEMDD [233] y MDD [215]; *Métodos basados en regresión logística*: MILR [158]; *Métodos basados en Máquinas de Soporte Vectorial*: MISMO que emplea el algoritmo SMO [152] para aprendizaje con SVM en conjunción con un núcleo MI [80]; *Enfoques basados en distancia*: CitationKNN [201] y MIOptimalBall [7]; *Métodos basados en Aprendizaje Supervisado*: *MIWrapper* [71] usando diferentes sistemas de aprendizaje; *MISimple* [211]; *Boosting*: MIBoost, [217] y *Algoritmos Evolutivos*: G3P-MI, la propuesta mono-objetivo que se ha desarollado.

Tabla 5.7: Resultados experimentales en la comparación de las técnicas multi-objetivo con otras propuestas para todos los conjuntos de datos considerados

Resultados Globales									
Datos		***Fox***	***Tiger***	***Elephant***	***Mut. Atoms***	***Mut. Bonds***	***Mut. Chains***	***Musk2***	***Musk2***
MOGLSG3P-MI	*Ex*	0.7600	0.8850	0.8900	0.8421	0.8421	0.8737	0.9778	0.9400
	Se	0.7800	0.9400	0.8700	0.9385	0.9077	0.9462	0.9600	0.9500
	Es	0.7400	0.8300	0.9100	0.6333	0.7000	0.7167	1.0000	0.9333
NSG3P-MI	*Ex*	0.7800	0.9350	0.9400	0.9158	0.8737	0.9211	1.0000	0.9301
	Se	0.8600	0.9200	0.9400	0.9462	0.9231	0.9462	1.0000	0.9607
	Es	0.7000	0.9500	0.9400	0.8501	0.7666	0.8667	1.0000	0.9095
SPG3P-MI	*Ex*	0.8350	0.9200	0.9250	0.8790	0.8684	0.9053	0.9667	0.9400
	Se	0.8900	0.9200	0.9400	0.9308	0.9308	0.9000	0.9800	0.9750
	Es	0.7800	0.9200	0.9100	0.7667	0.7333	0.9167	0.9500	0.9167
G3P-MI	Ex	0.7050	0.8700	0.8800	0.8526	0.8210	0.8105	0.9445	0.8800
	Se	0.7900	0.9400	0.9300	0.8462	0.8462	0.9231	1.0000	0.7500
	Es	0.6200	0.8000	0.8300	0.8167	0.7833	0.7333	0.9000	0.9180
MISMO	Ex	0.5800	0.8250	0.7900	0.7000	0.8421	0.7842	0.8778	0.8400
	Se	0.4100	0.7800	0.7500	0.3833	0.8833	0.7167	0.8250	0.8667
	Es	0.7500	0.8700	0.8300	0.8462	0.8231	0.8154	0.9200	0.8000
MIOptimalBall	Ex	0.5300	0.6700	0.7750	0.7263	0.7421	0.7158	0.8000	0.7700
	Se	0.7400	0.6500	0.8800	0.5500	0.6833	0.4500	0.9250	0.8333
	Es	0.3200	0.6900	0.6700	0.8077	0.7692	0.8385	0.7000	0.8050
MILR	Ex	0.5600	0.7700	0.7800	0.7000	0.7105	0.7684	0.8778	0.8500
	Se	0.4900	0.7400	0.6900	0.2000	0.2167	0.4333	0.8500	0.9167
	Es	0.6300	0.8000	0.8700	0.9308	0.9385	0.9231	0.9000	0.7500
MIEMDD	Ex	.5900	0.7450	0.7300	0.7316	0.7211	0.7368	0.8889	0.9000
	Se	0.3900	0.7100	0.7100	0.5000	0.4333	0.5000	0.8750	0.8833
	Es	0.7900	0.7800	0.7500	0.8385	0.8538	0.8462	0.9000	0.9250
MIDD	Ex	0.6550	0.7400	0.8300	0.7263	0.7526	0.7684	0.9222	0.7300
	Se	0.6200	0.7200	0.8300	0.4167	0.5593	0.5500	0.9500	0.9500
	Es	0.6900	0.7600	0.8300	0.8692	0.8615	0.8692	0.9000	0.4000
MDD	Ex	0.7000	0.7550	0.8000	0.7211	0.7316	0.7684	0.7889	0.7400
	Se	0.6300	0.7700	0.8200	0.1667	0.4000	0.5500	0.7750	0.8500
	Es	0.7700	0.7400	0.7800	0.9769	0.8846	0.8692	0.8000	0.5750
CitationKNN	Ex	0.5000	0.5000	0.5000	0.7526	0.7316	0.7474	0.9444	0.8500
	Se	1.0000	1.0000	1.0000	0.5833	0.4333	0.5167	0.9750	0.8667
	Es	0.0000	0.0000	0.0000	0.8308	0.8692	0.8538	0.9200	0.8250

Continúa en la siguiente página

Continuación de la página anterior

Resultados Globales									
Datos		*Fox*	*Tiger*	*Elephant*	*Mut. Atoms*	*Mut. Bonds*	*Mut. Chains*	*Musk2*	*Musk2*
PART	Ex	0.6500	0.8150	0.8050	0.8105	0.7895	0.8684	0.8444	0.8500
(Wrapper)	Se	0.6100	0.8300	0.7300	0.5667	0.7000	0.7500	0.8750	0.8833
	Es	0.6900	0.8000	0.8800	0.9231	0.8308	0.9231	0.8200	0.8000
Bagging&PART	Ex	0.6900	0.8100	0.8600	0.8105	0.7789	0.8684	0.9000	0.8500
(Wrapper)	Se	0.6400	0.8000	0.7900	0.5833	0.7167	0.7833	0.8500	0.8667
	Es	0.7400	0.8200	0.9300	0.9154	0.8077	0.9077	0.9400	0.8250
AdaBoost&PART	Ex	0.6900	0.8000	0.8400	0.8053	0.7737	0.8737	0.8778	0.9000
(Wrapper)	Se	0.6100	0.8000	0.7800	0.5333	0.7000	0.7667	0.8750	0.9167
	Es	0.7700	0.8000	0.9000	0.9308	0.8077	0.9231	0.8800	0.8750
SMO	Ex	0.6350	0.8000	0.8550	0.6842	0.6842	0.6895	0.8444	0.8900
(Wrapper)	Se	0.4800	0.7700	0.8000	0.0000	0.0000	0.0167	0.8250	0.9333
	Es	0.7900	0.8300	0.9100	1.0000	1.0000	1.0000	0.8600	0.8250
NaiveBayes	Ex	0.5900	0.7650	0.8400	0.6842	0.6263	0.5263	0.8000	0.7700
(Wrapper)	Se	0.2700	0.8800	0.7700	0.0000	0.1500	0.9667	0.8000	0.7667
	Es	0.9100	0.6500	0.9100	1.0000	0.8462	0.3231	0.8000	0.7750
PART	Ex	0.5800	0.7650	0.7800	0.7000	0.8263	0.7737	0.8111	0.7400
(Simple)	Se	0.6200	0.8100	0.7600	0.4833	0.8333	0.5833	0.8000	0.7833
	Es	0.5400	0.7200	0.8000	0.8000	0.8231	0.8615	0.8200	0.6750
AdaBoost&PART	Ex	0.6250	0.7950	0.8100	0.7526	0.8474	0.7947	0.8333	0.7900
(Simple)	Se	0.6200	0.8100	0.8200	0.5167	0.8333	0.6667	0.8250	0.8500
	Es	0.6300	0.7800	0.8000	0.8615	0.8538	0.8538	0.8400	0.7000
MIBoost	Ex	0.6950	0.8200	0.8150	0.8316	0.7737	0.8474	0.8778	0.8700
	Se	0.6300	0.8000	0.7400	0.7333	0.7333	0.7333	0.8000	0.9000
	Es	0.7600	0.8400	0.8900	0.8769	0.7923	0.9000	0.9400	0.8250

La Tabla 5.7 muestra los resultados medios de exactitud (Ex), sensibilidad (Se) y especificidad (Es) para todos los algoritmos en cada uno de los conjuntos de datos. Para evaluar la diferencia entre los algoritmos se ha realizado el test de Friedman. El test de Friedman se muestra en la Tabla 5.9 y los rangos de los algoritmos en la tabla 5.8. Este test indica que se encuentran diferencias significativas en los resultados de las medidas de exactitud, sensibilidad y especificidad. Un test a posteriori, el test de Bonferroni-Dunn [60] se ha realizado para encontrar los algoritmos que presentan diferencias significativas y pueden ser considerados peores propuestas.

La figura 5.7(a), muestra la aplicación de éste test sobre la exactitud. Este gráfico representa un diagrama de barras, cuyos valores son proporcionales a los rangos

Tabla 5.8: Rangos medios de los Algoritmos (comparativa para todos los conjuntos de datos)

Rangos Medios			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especifidad***
MOGLSG3P-MI	3.06	3.69	9.75
NSG3P-MI	1.38	2.38	7.19
SPG3P-MI	1.94	2.75	7.31
MISMO	11.06	12.50	9.94
MIOptimalBall	16.06	11.38	16.94
MILR	13.94	14.88	8.38
MIEMDD	12.88	14.38	9.63
MIDD	12.19	10.63	11.19
MDD	13.88	13.94	10.81
CitationKNN	13.94	6.88	12.81
G3P-MI	5.13	5.50	12.13
MIWrapper (*PART*)	9.13	10.31	9.50
MIWrapper (*Bagging&PART*)	7.38	9.25	7.13
MIWrapper (*AdaBoost&PART*)	7.69	9.94	7.50
MIWrapper (*SMO*)	12.56	14.50	4.69
MIWrapper (*NaiveBayes*)	15.50	13.81	10.31
SimpleMI (*PART*)	13.88	12.38	14.75
SimpleMI (*AdaBoost&PART*)	10.56	10.63	12.44
MIBOOST	7.88	10.31	7.63

medios obtenidos de cada algoritmo. Como ya hemos comentado en otras secciones de experimentación, para generar la gráfica se suma al valor más bajo de todos (el algoritmo de control) el valor de la diferencia crítica (valor CD) de este test y obtenemos un rango que es representado por una línea horizontal más gruesa en color azul que denotaremos como *umbral*. Aquellos valores que excedan de esta línea son algoritmos con resultados significativamente peores que el algoritmo de control (asociado con uno de nuestras propuestas multi-objetivo, NSG3P-MI, porque alcanza los valores más bajos). El umbral para esta medida se establece en 9.7935 con ($\alpha = 0{,}05$). Observando esta figura, los algoritmos que no exceden el umbral determinado por Bonferroni-Dunn (representados en color azul) son las propuestas multi-objetivo, G3P-MI y alguna versiones de algoritmos *wrapper* y

Tabla 5.9: Resultados del test de Friedman (comparativa para todos los conjuntos de datos)

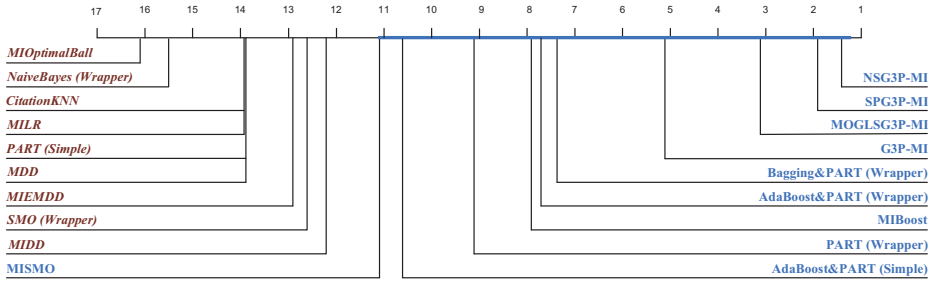
Test de Friedman			
Medida	Valor de Friedman	$\chi^2(\rho = 0{,}01)$	Conclusión
Exactitud	95.4730	28.87	Rechazar la hipótesis nula
Sensibilidad	73.0322	28.87	Rechazar la hipótesis nula
Especifidad	39.8743	28.87	Rechazar la hipótesis nula

boost. Los resultados de test de Bonferroni consideran el resto de los algoritmos con resultados lo suficientemente alejados para ser consideradas peores propuestas. Se puede ver que la mejor propuesta es NSG3P-MI, seguida del resto de estrategias multi-objetivo, demostrando que estas técnicas son las que resultan más precisas.

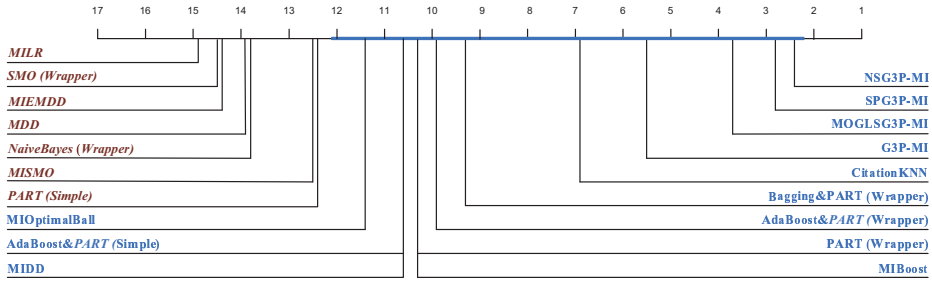
Con respecto a la medida de sensibilidad, los resultados del test de Bonferroni-Dunn se muestra en la Figura 5.7(b). El umbral en este caso es 11.0435 ($\alpha = 0{,}05$). Observando esta figura, los algoritmos que no exceden el umbral determinado por el test de Bonferroni-Dunn son principalmente las propuestas multi-objetivo (de nuevo son los algoritmos más cercanos a la mejor propuesta, NSG3P-MI), G3P-MI, algunas versiones de los métodos *wrapper* y *simple*, métodos basados en distancia (CitationkNN y OptimallBall), Boost y diverse density (MIDD). Los resultados del test de Bonferroni-Dunn consideran que los resultados del resto de algoritmos están suficientemente lejos para presentar diferencias significativas y poder ser consideradas peores propuestas.

Podemos concluir que considerando los valores de sensibilidad, de nuevo las propuestas multi-objetivo obtienen los mejores resultados (los algoritmos con los valores de rango más bajos) presentando una considerable diferencia con respecto a las otras propuestas.

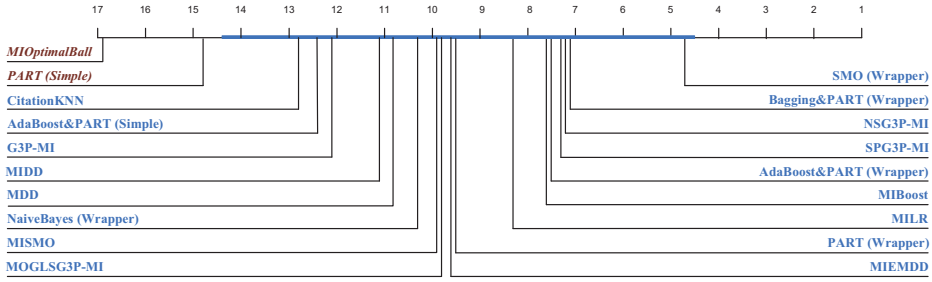
Con respecto a la medida de especificidad, la figura 5.7(c) muestra los resultados del test de Bonferroni-Dunn. El umbral en este caso se fija en el valor de 13.2310 ($\alpha = 0{,}05$). Observando esta figura, los algoritmos que no exceden el umbral determinado por el test de Bonferroni-Dunn son principalmente algunas versiones wrapper o simple, las propuestas multi-objetivo, algoritmos basados en SVM, boosting y versiones de diverse density. Los resultados del test de Bonferroni-Dunn consideran el resto de técnicas obtienen unos resultados suficientemente lejos para mostrar



(a) Resultados para Exactitud



(b) Resultados para Sensibilidad



(c) Resultados para Especificidad

Figura 5.7: Resultados del test de Bonferroni Dunn ($p < 0{,}01$) en la comparativa del modelo MOG3P-MI

diferencias significativas con los demás. En los valores de especificidad, el algoritmo de control es un método *wrapper* que usa máquinas de soporte vectorial, aunque ninguna de las propuestas multi-objetivo es el algoritmo de control en esta medida, ellas mantienen unos resultados competitivos siendo las más cercanas al algoritmo de control.

Podemos concluir que las técnicas multi-objetivo implementadas obtienen los mejores resultados en los valores de exactitud y sensibilidad y con respecto a la medida de especificidad mantiene unos resultados competitivos. En este sentido, debemos resaltar un aspecto importante, y es que a diferencia del resto de algoritmos que, cuando optimizan una medida, lo hacen a expensas de disminuir otra de las medidas enormemente. De este modo, los que mantienen una posición buena en la medida de especificidad, son consideradas propuestas que obtienen peores resultados en sensibilidad. Esto es indicativo de que la calidad de las reglas no es la adecuada porque una de las clases de los ejemplos no los clasifica de forma adecuada, ya sean los positivos o los negativos, dependiendo de que optimicemos más la sensibilidad que la especificidad. Así, por ejemplo, el mejor algoritmo en especificidad *MIWrapper con SMO*, es la peor propuesta en la medida de sensibilidad. Esto resulta en clasificadores que no clasifican de manera adecuada los ejemplos que pertenecen a la clase positiva. Las propuestas multi-objetivo consiguen solucionar este problema, obteniendo los mejores resultados en sensibilidad y exactitud y manteniendo una buena posición en la especificidad donde son una de las propuestas más cercanas a la mejor propuesta no presentando diferencias significativas con dicha propuesta.

5.4. Conclusiones

En este capítulo se lleva a cabo un primer estudio comparativo de algoritmos evolutivos multi-objetivo para el aprendizaje con múltiples instancias. Para ello, se han desarrollado tres propuestas multi-objetivo, NSG3P-MI, SPG3P-MI y MOGLSG3P-MI, basadas en tres algoritmos ampliamente utilizados y comparados en el escenario de aprendizaje tradicional (NSGA2, SPEA2 y MOGLS) y adaptados al paradigma de G3P y a un entorno MIL.

Para comprobar el funcionamiento de éstas técnicas se ha llevado a cabo dos estudios experimentales. El primero de ellos consiste en un estudio comparativo entre las diferentes propuestas multi-objetivo desarrolladas que pone de manifiesto que NSG3P-MI es el mejor algoritmo considerando tanto la evaluación de los POFs, como los resultados finales de exactitud, sensibilidad y especificidad que se alcanzan. El segundo estudio compara las técnicas multi-objetivo con algunos de los paradigmas más importantes que se han desarrollado hasta la fecha para resolver

problemas multi-instancia teniendo en cuenta un total de dieciséis propuestas y ocho conjuntos de datos en la comparación. Los resultados experimentales han demostrado que las propuestas multi-objetivo obtienen los mejores valores con respecto a las medidas de exactitud y sensibilidad y que alcanzan uno de resultados competitivos en la medida de especificidad. El test de Friedman confirma que existen diferencias significativas entre los resultados de los métodos para las medidas de exactitud, sensibilidad y especificidad. Con respecto a las medidas de exactitud y sensibilidad, el test de Bonferroni-Dunn determina que NSG3P-MI es el algoritmo de control tanto en exactitud como en sensibilidad y en general todas las propuestas multi-objetivo logran mejores resultados que el resto de las técnicas. Con respecto a los valores de especificidad, dos de las técnicas multi-objetivo obtienen uno de los resultados más próximos a los obtenidos por el algoritmo de control. Uno de los logros más representativo de las técnicas multi-objetivo es el equilibrio que alcanza en las diferentes medidas, de forma que a pesar de obtener los mejores resultados de sensibilidad son capaces de obtener unos resultados muy competitivos en la medida de especificidad. Además, nuestras propuestas proporcionan reglas en el formato SI-ENTONCES que son fáciles de interpretar y comprender.

A la vista de los resultados obtenidos, nos decantaremos por un modelo multi-objetivo de G3P para MIL, que denominaremos a partir de ahora MOG3P-MI. El algoritmo seleccionado es NSG3P-MI por dos razones principalmente, la primera es que se trata del modelo que ha obtenido mejor evaluación en las características de los POFs y en segundo lugar, si bien, entre los resultados de SGP3-MI y NSG3P-MI no se pueden obtener diferencias significativas en sus resultados, según el estudio realizado, NSG3P-MI es el que obtiene los mejores valores en las medidas de exactitud, sensibilidad y especificidad considerando los diferentes conjuntos de datos empleados.

6

Recomendación de Páginas Web

Índice

6.1. Introducción

Durante las últimas décadas, la cantidad de información disponible en Internet ha crecido tan rápidamente que excede la posibilidad de poder procesarla manualmente. Los usuarios se sienten desbordados con tanta información, siendo realmente complicado localizar información que se adapta a sus necesidades en un tiempo razonable. En esta situación, las herramientas que predicen las preferencias de los usuarios y proporcionan recomendaciones sobre si un artículo, producto o servicio concreto será de interés o no, aparecen como sistemas indispensables. Estos sistemas son referenciados en la literatura como sistemas de recomendación [130], diferenciándose principalmente dos tipos de sistemas: los basados en filtrado colaborativo [167] que intentan identificar grupos de personas con gustos similares a los del usuario y realizar la recomendación en base a los intereses de ese grupo de personas y los basados en contenido [150] que almacenan información sobre los artículos que le interesaron al usuario en el pasado para recomendarle elementos de acuerdo a esas preferencias.

Los sistemas de recomendación mantienen características similares con respecto a los enfoques de recuperación de información tradicional, pero difieren de ellos, especialmente, en el uso de perfiles o modelos que contienen información sobre los gustos de los usuarios, sus preferencias y sus necesidades. La información mantenida en estos sistemas difiere de acuerdo al tipo de procesamiento realizado. En sistemas de recomendación basados en filtrado colaborativo, este modelo refleja las preferencias y necesidades de los usuarios, mientras en los sistemas basados en contenido, esta información comprueba la relación entre los artículos a recomendar y las preferencias previas dadas por el usuario. En cualquier caso, la construcción de estos modelos de usuario se convierte así la tarea más importante para el funcionamiento exitoso de los sistemas de recomendación.

Un modelo de usuario (*User Model*, UM) se compone principalmente de información sobre las preferencias individuales del usuario. La calidad de las recomendaciones depende, en gran medida, de las características de este modelo, por ejemplo, cómo de preciso es, qué cantidad de información almacena, y si esta información está actualizada. Por esta razón, la construcción de perfiles exactos es una tarea fundamental para garantizar el éxito del sistema de recomendación. Sin embargo, el modelado de las preferencias de los usuarios es un trabajo costoso, es complicado obtener información de los intereses de lo usuarios. En términos de tiempo y esfuerzo, se requiere una información inicial sobre sus preferencias para poder modelar su perfil y ofrecerle recomendaciones de alta calidad. Las primeras etapas de recomendación son las más complicadas ya que normalmente hay muy poca información disponible sobre los gustos del usuario.

Dentro de este contexto, nos encontramos con el problema de clasificar un conjunto de páginas web índice en interesantes o no interesantes para un usuario dado. La dificultad de este problema, clasificado en la categoría de los sistemas basados en contenido, estriba en la representación de los ejemplos de entrenamiento, dado que el número de enlaces que presenta una página es variable y que, además, no tenemos conocimiento sobre qué enlaces han motivado que un individuo considere o no una página como interesante. Recientemente, Zhou et al. [239] han abordado el problema desde la perspectiva del aprendizaje multi-instancia adaptando el algoritmo de los k vecinos más cercanos (k *Nearest Neightbour*, KNN) a este nuevo

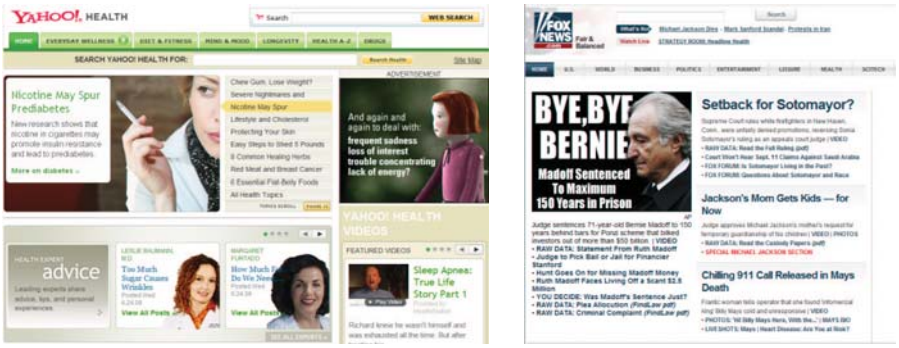
paradigma de aprendizaje, demostrando que este enfoque mejora ampliamente los resultados producidos por los algoritmos de aprendizaje supervisado.

A pesar de los resultados obtenidos, el enfoque de [239] presenta los siguientes inconvenientes: en primer lugar, como el algoritmo KNN realiza cálculos que crecen linealmente con el número de items, este algoritmo es difícil de escalar manteniendo exactitud en la predicción cuando el número de items aumenta. El problema es que en cada nueva clasificación requiere de los ejemplos que se poseen para realizar la nueva clasificación. En segundo lugar, los métodos KNN es un algoritmo de caja negra, es decir, se limita a clasificar las páginas web en interesantes o no interesantes, sin proporcionar información adicional que pueda ayudar a comprender la preferencias del usuario. Esta es una propiedad no deseada en el caso de sistemas de recomendación, donde cualquier información que nos permita comprender el modelo de usuario para ver las preferencias de los mismos resulta de gran utilidad.

Para solucionar estas deficiencias, en este capítulo se estudiará la ideonidad de nuestros modelos para la solución del problema. Particulamente evaluaremos tanto la versión mono-objetivo, G3P-MI, como la versión multi-objetivo, MOG3P-MI. Ambos sistemas generan clasificadores basados en reglas simples y fácilmente interpretables que incrementan la capacidad de generalización, obteniendo reglas que proporcionan información sobre los intereses de los usuarios y por tanto pueden ser usadas como motor de los sistemas de recomendación. Además, los nuevos ejemplos (paginas webs) pueden ser clasificados rápidamente, basándonos en los perfiles de usuario que se han generado.

6.2. Descripción del problema

Las páginas índice son páginas web cuyo contenido presenta múltiples enlaces que referencian a otras páginas. Estas páginas presentan mucha información a través de dichos enlaces, aunque en ellas solamente encontramos resúmenes de dicha información y debemos acceder a las páginas referenciadas para obtener una descripción en mayor profundidad del tema en cuestión. Podemos encontrar un gran número de estas páginas en Internet, por ejemplo, la entrada de salud de Yahoo, en la dirección web *http://health.yahoo.com*, muestra diferentes temas relacionados con la salud (en la figura 6.1(a) se puede ver el contenido de dicha página). Otro ejemplo



(a) http://health.yahoo.com (b) http://www.foxnews.com

Figura 6.1: Ejemplos de páginas web índice

lo podemos encontrar en la página *http://www.foxnews.com* donde la temática se centra en diferentes noticias de actualidad (en la figura 6.1(b) podemos ver esta página).

Algunas de estas páginas pueden contener temas interesantes para el usuario mientras que otras no. Sería interesante poder analizarlas automáticamente y mostrar al usuario sólo aquellas que pueden resultarle interesantes. Para alcanzar este fin, es necesario identificar los intereses del usuario partiendo de páginas índice que previamente haya consultado y determinado si contenían algún tema de su interés o no.

El problema de la recomendación de páginas web índice consiste en construir un modelo que establezca qué páginas índice interesan a un usuario dado, a partir del contenido de un conjunto de páginas web índice que han sido previamente etiquetadas como "interesantes" o "no interesantes" por dicho usuario. Este problema tiene una resolución más compleja que otros problemas análogos, relacionados con la construcción de modelos de usuario en sistemas de recomendación basados en contenido [139; 150]. La principal dificultad recae en que el usuario especifica si está interesado en la página o no, en lugar de especificar los enlaces concretos en los que realmente está interesado. De este modo, conocemos las preferencias del usuario de acuerdo a la información general que muestra la página, pero no conocemos la información específica que realmente le interesa de dicha página (es decir, el enlace concreto o el conjunto de ellos que realmente son de su interés).

Esta información general hace el problema más complejo y ha provocado que los algoritmos de aprendizaje supervisado hayan encontrado problemas en su resolución, obteniendo recomendaciones que no poseen la calidad de predicción esperada [239]. El aprendizaje con múltiples instancias se presenta como una opción más beneficiosa cuya especificación se dará en la siguiente sección.

6.2.1. Descripción del problema mediante el empleo de múltiples instancias

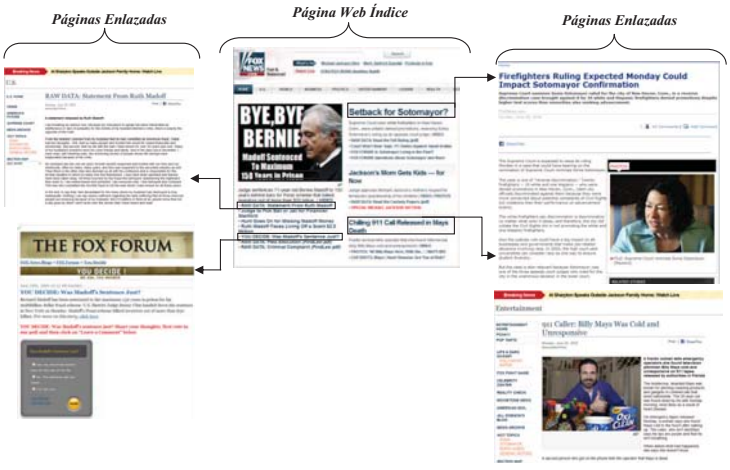
El problema de la recomendación de páginas web índice se ajusta perfectamente a una representación multi-instancia. Cada página web índice podría representarse mediante una bolsa, y cada una de las páginas conectadas mediante enlaces sería una instancia de la bolsa. Además, la hipótesis de Dietterich [57] parece también apropiada para la resolución de este problema. Una página web índice será considerada interesante, si al menos uno de los enlaces que contiene referencia a una página que resulta interesante al usuario, mientras que una página será considerada no interesante si ninguna de las páginas a las que referencia interesan al usuario.

De este modo, el problema multi-instancia se puede plantear como un problema donde la meta es etiquetar nuevas páginas índice como positivas o negativas. Una página índice es positiva si el usuario está interesado al menos en uno de sus enlaces. Una página índice es negativa si ninguno de sus enlaces es de interés para el usuario. De acuerdo a la representación especificada, cada página índice será considerada como una bolsa mientras que sus páginas enlazadas serán consideradas como instancias en la bolsa. Esta configuración de bolsas e instancias se puede apreciar en las figuras 6.2(a) y 6.2(b) con respecto a las páginas índice que se indicaron anteriormente.

Para la representación de las instancias, que representan cada una de las páginas enlazadas, hemos optado por dos esquemas de codificación distintos, usados habitualmente en categorización de documentos [170]:

- Un primer esquema consiste en representar una página como un vector de términos, cada uno de los cuáles representa la presencia de dicho término

(a) Representación de *http://health.yahoo.com*



(b) Representación de *http://www.foxnews.com*

Figura 6.2: Representación multi-instancia de las páginas web índice

en el documento. Formalmente, cada instancia será representada por n componentes, $< term_1 >; < term_2 >; \ldots; < term_n >$ donde $term_i$ es uno de los n términos más frecuentes que aparecen en las correspondientes páginas enlazadas.

- El segundo esquema representa la página como un vector de términos junto con su frecuencia. Formalmente cada instancia es representada por n componentes, $< term_1, freq_{term_1} >; < term_2, freq_{term_2} >; \ldots; < term_n, freq_{term_n} >$ donde $term_i$ es uno de los n términos más frecuentes en las correspondiente página enlazada y $freq_{term_i}$ es el número de ocurrencias de $term_i$.

Finalmente, una bolsa tendrá tantas instancias como enlaces, siendo diferente el número de instancias de una bolsa a otra. En la figura 6.3 se puede ver una representación de una página web enlazada a m páginas de acuerdo a la segunda representación.

$\langle term_{11}, freq_{term11} \rangle$, $\langle term_{12}, freq_{term12} \rangle$, ... , $\langle term_{1n}, freq_{term1n} \rangle$
$\langle term_{21}, freq_{term21} \rangle$, $\langle term_{22}, freq_{term22} \rangle$, ... , $\langle term_{2n}, freq_{term2n} \rangle$
...
$\langle term_{m1}, freq_{termm1} \rangle$, $\langle term_{m2}, freq_{termm2} \rangle$, ... , $\langle term_{mn}, freq_{termmn} \rangle$

Figura 6.3: Representación de una página web índice con m páginas enlazadas

6.2.2. Datos utilizados en la experimentación

Para evaluar el rendimiento de nuestro algoritmo en la tarea de recomendación de páginas web índice y comparar con otros resultados disponibles en la bibliografía, hemos utilizado los conjuntos de datos preparados por el equipo del Prof. Z.H. Zhou y utilizados en su trabajo sobre recomendación de páginas web índice [239]. Esta colección de datos está compuesta por nueve conjuntos de datos, cada uno de los cuales contiene 113 páginas índice que fueron etiquetadas por un voluntario de acuerdo a sus intereses. De este número de páginas, 75 son seleccionadas aleatoriamente como páginas de entrenamiento mientras las restantes 38 son usadas para

Tabla 6.1: Propiedades del conjunto de datos del problema de recomendación

Conjunto de datos										
Partición	N^o de Páginas	V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8	V_9
Entrenamiento	Positivas	17	18	14	56	62	60	39	35	37
	Negativas	58	57	61	19	13	15	36	40	38
Test	Positivas	4	3	7	33	27	29	16	20	18
	Negativas	34	35	31	5	11	9	22	18	20

test. La tabla 6.1 muestra una descripción de los ejemplos que componen cada conjunto de datos[1].

Atendiendo a la distribución de los ejemplos en el conjunto, podemos encontrar tres grupos de usuarios. El primer grupo, formado por los conjuntos de datos V_1, V_2 y V_3, contiene más bolsas negativas que positivas (este grupo lo denominaremos usuarios selectivos, representando un conjunto mayoritario de ejemplos de la clase negativa). El segundo grupo está formado por los conjuntos de datos V_4, V_5 y V_6 que contienen más bolsas positivas que negativas (este grupo lo denominaremos usuarios permisivos y representan un conjunto mayoritario de ejemplos que pertenecen a la clase positiva) y el tercer grupo formado por los conjuntos de datos V_7, V_8 y V_9 contiene el mismo número de bolsas positivas que bolsas negativas (este grupo lo denominaremos usuarios balanceados y representan un conjunto balanceado de ejemplos que pertenecen a la clase positiva y negativa).

Para poder utilizar los conjuntos de datos originales, ha sido necesario realizar un preprocesado de la información, de forma que éstos se adapten al formato de representación que utilizan nuestros algoritmos. En efecto, aunque Zhou et al. presentan las páginas web índice como bolsas con una o varias instancias, representan cada instancia mediante una lista que contiene los N términos que más aparecen en la página (donde N toma distintos valores entre 5 y 15) y realizan diferentes experimentos considerando diferente número de términos más frecuentes.

[1]Estos conjuntos de datos pueden descargarse de la siguiente dirección web: http://cs.nju.edu.cn/zhouzh/zhouzh.files/publication/annex/milweb-datafile.htm

En nuestro caso, se ha representado cada instancia en formato vectorial, donde cada componente del vector está relacionada con un término del corpus de documentos. El valor concreto de cada componente puede ser binario (si optamos por la representación booleana) o entero si optamos por una representación basada en frecuencias absolutas. Esta información es adaptada a los dos formatos multi-instancia con los que se trabaja, el de KEEL y WEKA, que son comentados en el apéndice B.

6.3. Aplicación de G3P-MI para la resolución del problema de recomendación de páginas índice

Tal y como se ha comentado, en la evaluación que llevaremos a cabo emplearemos dos esquemas de representación diferentes, uno basado en la aparición de los términos y otro basado en la frecuencia de dichos términos. Ambas representaciones serán evaluadas en la experimentación con el resto de algoritmos que han sido utilizados hasta la fecha en la resolución de este problema. Además, previamente llevaremos a cabo un estudio de nuestro algoritmo, G3P-MI, para comprobar el efecto de la reducción del conjunto de datos en nuestra propuesta.

6.3.1. Adaptación del modelo G3P-MI

En esta sección presentaremos algunos detalles de la adaptación del algoritmo G3P-MI a la tarea de recomendación de páginas web índice y algunas variaciones que se ha realizado al conjunto de datos para evaluar cuál de los dos esquemas de representación utilizados produce mejores resultados.

En primer lugar, explicaremos las variantes del algoritmo que se han definido para resolver este problema utilizando, respectivamente, las representaciones booleana y entera. Posteriormente pasaremos a explicar la función de evaluación utilizada. Finalizaremos exponiendo brevemente los parámetros de configuración utilizados en la experimentación.

6.3.1.1. Variantes G3P-MI

Se han desarrollado dos variantes del algoritmo G3P-MI, especializadas para los dos tipos de representaciones que se han desarrollado. Dichas variantes sólo se diferencian en el formato de las reglas aprendidas en el proceso evolutivo. En el caso de la representación booleana, solamente se trabajará con atributos categóricos que nos permiten determinar reglas que introducen en su comparación si la página contiene un determinado concepto o no, de acuerdo a la gramática especificada en la figura 6.4 *a)*. Los operadores empleados son "EQ" (*equal*, igual) o "NO_EQ" (*no equal*, diferente); en el caso de que un determinado término sea "EQ" a *true* o "NO_EQ" a *false*, se identifica que contiene dicho término y en el caso de que sea "NO_EQ" a *true* o "EQ" a *false*, se considera que no contiene dicho término.

Cuando la representación de las páginas es numérica (basada en frecuencias) las reglas generadas informan sobre si un determinado término aparece en una determinada página más o menos veces que un umbral definido. De acuerdo a la gramática mostrada en la figura 6.4 *b)*, los operadores empleados son "GT" (*greater than*, mayor que), "GE" (*greater and equals that*, mayor o igual que), "LT" (*less than*, menor que) and "LE" (*less and equals than*, menor y igual que), que comparan la frecuencia de aparición de un término en el documento con el umbral (un valor entero).

6.3.1.2. Función de Evaluación

El problema de desarrollar buenas métricas para medir la efectividad de las recomendaciones no es una tarea trivial [90; 221; 91]. Al estar trabajando con tareas de clasificación podríamos pensar en medidas como *exactitud* (es decir, el porcentaje de páginas web índice que son correctamente clasificadas por el clasificador).

$$exactitud = \frac{\#recomendaciones\ correctas}{\#paginas\ recomendadas\ \ y\ \ no\ recomendadas} \tag{6.1}$$

Sin embargo, en sistemas de recomendación es muy común trabajar con datos no balanceados, donde el número de artículos interesantes es mucho menor que el número de artículos no interesantes o viceversa. En este caso, tal y como se estudió en la sección 3.3.3 referentes a las métricas de evaluación, si usamos solamente

$\sum N = \langle cond_I \rangle$, $\langle$term$\rangle$, $\langle$valor-term$\rangle$, $\langle$cmp$\rangle$, $\langle$cmp-cat$\rangle$, $\langle$op-cat$\rangle$
$\sum T =$ EQ, NOT-EQ, OR, AND

$\langle cond_I \rangle$:= $\langle$cmp$\rangle$
| **OR** $\langle$cmp$\rangle$ $\langle$antecedente$\rangle$
| **AND** $\langle$antecedente$\rangle$
$\langle$cmp$\rangle$:= $\langle$op-cat$\rangle$ $\langle$cmp-cat$\rangle$
$\langle$op-cat$\rangle$:= **EQ**
| **NOT EQ**
$\langle$cmp-cat$\rangle$:= $\langle$term$\rangle\langle$valor-term$\rangle$
$\langle$term$\rangle$:= *Cualquier término valido en el conjunto de datos*
$\langle$valor-term$\rangle$:= *Cualquier valor válido que puede tomar el término, en este caso true o false*

a) Representación booleana

$\sum N = \langle cond_I \rangle$, $\langle$term$\rangle$, $\langle$freq-term$\rangle$, $\langle$cmp$\rangle$, $\langle$cmp-cat$\rangle$, $\langle$op-cat$\rangle$
$\sum T =$ GE, LT, LE, GT, OR, AND

$\langle cond_I \rangle$:= $\langle$cmp$\rangle$
| **OR** $\langle$cmp$\rangle$ $\langle$antecedente$\rangle$
| **AND** $\langle$antecedente$\rangle$
$\langle$cmp$\rangle$:= $\langle$op-num$\rangle$ $\langle$cmp-num$\rangle$
$\langle$op-num$\rangle$:= **GE**
| **LT**
| **LE**
| **GT**
$\langle$cmp-num$\rangle$:= $\langle$term$\rangle\langle$freq-term$\rangle$
$\langle$term$\rangle$:= *Cualquier término válido en el conjunto de datos*
$\langle$freq-term$\rangle$:= *Cualquier valor válido de frecuencia de aparición del término*

b) Representación numérica

Figura 6.4: Gramática empleada para representa el problema de recomendación de páginas web índice

como heurística la exactitud en la predicción, el clasificador obtendría mejores resultados si ningún ejemplo de la clase minoritaria es clasificado y determinando siempre que todo objeto pertenece a la clase mayoritaria. Por esta razón, se han considerado otras medidas como la *precision*, que representa el porcentaje de documentos aceptados que son realmente relevantes para el usuarios:

$$precisión = \frac{\#interesantes\ y\ recomendadas}{\#recomendadas} \tag{6.2}$$

En otros casos, el problema es que hay muchos más ejemplos interesantes que no interesantes. En este caso, la precisión no es la mejor medida para tener en cuenta, y sería necesario considerar otras medidas, como por ejemplo el *alcance* o *recall*[2]:

$$alcance = \frac{\#interesantes\ y\ recommendadas}{\#interesantes} \tag{6.3}$$

Por las razones anteriores, hemos intentado construir una función de evaluación combinando la exactitud, la precisión y el alcance. De todas las combinaciones ensayadas, el producto ha sido la que ha producido los mejores resultados, ya que pondera por igual las tres métricas y, además penaliza aquellos individuos que tienen alguna medida con valor cero. De este modo la función de evaluación utilizada en los experimentos sería:

$$fitness = exactitud * alcance * precisión \tag{6.4}$$

6.3.1.3. Parámetros de configuración

La tabla 6.2 muestra los parámetros de configuración empleados en los experimentos realizados con las diferentes versiones del algoritmo G3P-MI. Realmente la única diferencia que hay entre ellas es el conjunto de datos con el que se está trabajando, el resto de parámetros, en lo referente a tamaño de la población, número de generaciones, operadores de cruce y mutación están configurados de igual manera.

[2]la traducción al castellano de la palabra *recall* ha recibido varias acepciones, y es complicado elegir la que mejor se adapta al contenido de lo que representa, en este caso nos decantaremos por la traducción dada en [93], aunque seguramente existan otras traducciones más acertadas.

Tabla 6.2: Configuración de los parámetros de G3P-MI

Algoritmo G3P-MI	
Parámetros	Valores
Tamaño de la población	1000
Número de generaciones	100
Probabilidad de cruce	95 %
Probabilidad de mutación	5 %
Porcentaje de elistismo	1 %
Método de selección de padres	*Ruleta*
Profundidad máxima del árbol	15

6.3.2. Resultados experimentales

Llevaremos a cabo dos tipos de experimentos. En el primero compararemos el funcionamiento de nuestras propuestas con respecto al problema de recomendación de páginas web índice y en la segunda parte se comparan nuestras mejores propuestas con las otras técnicas empleadas en la resolución de este problema.

6.3.2.1. Comparación de diferentes versiones de G3P-MI

En el trabajo de Zhou et al. [239] experimentando considerando diferente frecuencias de los términos, en concreto usando 5, 8, 10 y 15 términos. La información que se muestra en estos conjuntos considera los 20 términos más frecuentes de cada página. Los conjuntos de datos representados, tal y como se han especificado en la sección 6.2.2, se caracterizan por su elevada dimensionalidad y su alta dispersión. Con el objeto de reducir ambos efectos y, probablemente, mejorar el rendimiento de los algoritmos, en esta sección se analizará el desarrollo de un preprocesado de los conjuntos originales. Para ello, eliminaremos los términos que aparecen en muy pocos documentos y términos que aparecen en la mayoría de éstos. En ambos casos, la capacidad de discriminación de dicho término es tan reducida, que su eliminación no debería reducir la información relevante. Tras una serie de pruebas preliminares, decidimos eliminar los términos que aparecían en 3 o menos documentos y los que

aparecían en más de 40. De este modo el número total de atributos pasó de 5763 a 600.

De este modo, para evaluar nuestra propuesta se han empleado 4 versiones diferentes de los conjuntos de datos:

- Boolean All. Estos 9 conjuntos de datos usan la representación booleana (presencia o ausencia de un término en el documento), y cada vector que representa un documento presenta un tamaño de 5763 componentes, considerando todos los términos del corpus de documentos original.
- Boolean Filtered. Estos 9 conjuntos de datos usan la representación booleana (presencia o ausencia de un término en el documento), y cada vector que representa un documento presenta un tamaño de 600 componentes, los términos del corpus que aparecen en más de 3 documentos y menos de 40.
- Frequency All. Estos 9 conjuntos de datos usan la representación numérica (frecuencia de aparición de un término en el documento), y cada vector que representa un documento presenta un tamaño de 5763 componentes, todos considerando todos los términos del corpus de documentos original.
- Frequency Filtered. Estos 9 conjuntos de datos usan la representación numérica (frecuencia de aparición de un término en el documento), y cada vector que representa un documento presenta un tamaño de 600 componentes, los términos del corpus que aparecen en más de 3 documentos y menos de 40.

Para su evaluación, se ha ejecutado nuestro algoritmo 5 veces (con diferentes semillas aleatorias) sobre cada uno de los nueve conjuntos de datos disponibles y los valores medios obtenidos para exactitud, precisión y alcance son los mostrados. La tabla 6.3 muestra los valores medios obtenidos para exactitud, precisión y alcance para los diferentes versiones de los conjuntos de datos que se han considerado. Realizaremos a continuación un estudio para determinar cuáles de ellas producen mejores resultados en la tarea de recomendación de páginas web índice. Para establecer si existen diferencias significativas entre los resultados obtenidos en la Tabla 6.3, hemos realizado un test de Friedman [56]. La comparación considera 4 algoritmos y 9 conjuntos de datos. En la tabla 6.4 se muestran los rangos medios

Tabla 6.3: Resultados experimentales en la comparación de las diferentes versiones de G3P-MI

Exactitud									
Algoritmo	V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8	V_9
Boolean All	0.816	0.868	0.842	0.868	0.763	0.790	0.763	0.684	0.632
Boolean Filtered	0.842	0.737	0.842	0.842	0.842	0.790	0.842	0.605	0.684
Frequency All	0.921	0.737	0.868	0.895	0.790	0.763	0.421	0.526	0.474
Frequency Filtered	0.868	0.921	0.868	0.895	0.763	0.816	0.475	0.553	0.553
Precisión									
Algoritmo	V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8	V_9
Boolean All	0.283	0.250	0.571	0.868	0.750	0.784	0.652	0.643	0.567
Boolean Filtered	0.333	0.231	0.571	0.936	0.889	0.862	0.917	0.576	0.688
Frequency All	1.000	0.231	0.750	0.892	0.771	0.763	0.421	0.526	0.474
Frequency Filtered	0.429	0.500	0.625	0.892	0.750	0.806	0.444	0.541	0.514
Alcance									
Algoritmo	V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8	V_9
Boolean All	0.500	0.333	0.571	1.000	1.000	1.000	0.938	0.900	0.944
Boolean Filtered	0.500	1.000	0.571	0.879	0.889	0.862	0.688	0.950	0.611
Frequency All	0.250	1.000	0.429	1.000	1.000	1.000	1.000	1.000	1.000
Frequency Filtered	0.750	1.000	0.714	1.000	1.000	1.000	1.000	1.000	1.000

para los distintos conjuntos de datos con los que se ha trabajado. El resultado del test, acepta la hipótesis nula, con lo que se puede determinar que no existen diferencias significativas entre las distintas variantes analizadas.

A pesar del los resultados proporcionados por el test de Friedman, un análisis de la Tabla 6.4 muestra que las versiones que usan la operación de filtrado (*boolean filtered* and *frequency filtered*) producen resultados ligeramente mejores que sus homólogos sin preprocesamiento (por ejemplo, *frequency filtered* es siempre mejor que *frequency all*). Este resultado es consistente con el hecho de que reducir el espacio de búsqueda hace el problema más sencillo y, en consecuencia, los resultados obtenidos son mejores.

Otra cuestión interesante que puede deducirse del análisis de la tabla 6.4 es que, en general, las versiones booleanas de G3P-MI parecen presentar resultados más precisos, mientras que las versiones numéricas (que emplean las frecuencias de los términos) son mejores en términos de alcance. Esto nos lleva a pensar que, a

Tabla 6.4: Rangos medios de los algoritmos (comparativa para las versiones de G3P-MI)

Rangos Medios			
Algoritmo	Exactitud	Alcance	Precisión
Boolean All	2.6111	2.7777	2.7777
Boolean Filtered	2.3888	3.3333	1.8888
Frequency All	2.8333	2.2777	2.8888
Frequency Filtered	2.1666	1.6111	2.4444

pesar del estudio realizado sobre la función de evaluación a emplear, el algoritmo G3P-MI no ha sido capaz de alcanzar un equilibrio entre las distintas medidas, con lo que sería interesante la aplicación de un algoritmo multi-objetivo que nos permita encontrar las soluciones que componen el POF y analizar soluciones con un equilibrio entre los valores de las diferentes medidas. En la siguiente sección veremos la propuesta multi-objetivo de nuestro modelo aplicada a este problema para estudiar cómo lo resuelve.

6.3.2.2. Comparación de G3P-MI con otros paradigmas aplicados al problema

En esta sección, compararemos los resultados de nuestro algoritmo con el resto de propuestas que han abordado este problema. Estas propuestas son Fretcit-KNN, Citation-KNN y Txt-KNN, todas descritas en [237]. Txt-KNN es una adaptación del algoritmo KNN estándar para objetos textuales. Citation-KNN es similar a Txt-KNN pero considera tanto referencias como citas para predecir los nuevos documentos. Finalmente, Fretcit-KNN es similar a Citation-KNN, pero adapta la distancia mínima de Hausdorff a la medición de la frecuencia de los términos para medir la distancia de los vecinos considerando bolsas y analiza tanto las citas como las referencias de las nuevas bolsas para determinar su etiqueta.

La Tabla 6.5 muestra un resumen de los valores medios obtenidos por los diferentes algoritmos y conjuntos de datos; los resultados de nuestro algoritmo corresponden a la versión *filtered*, que es la seleccionada para comparar con el resto de técnicas. En todos los casos se han considerado tanto la representación booleana y la basada en frecuencias.

Tabla 6.5: Resultados experimentales en la comparación de G3P-MI con otras técnicas de MIL para el problema de recomendación de páginas web índice

	Exactitud									
	Algoritmo	V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8	V_9
Booleana	Txt-kNN	0.863	0.795	0.726	0.889	0.732	0.795	0.552	0.590	0.569
	Citation-kNN	0.879	0.768	0.763	0.816	0.758	0.816	0.684	0.684	0.653
	Fretcit-kNN	0.911	0.868	0.858	0.847	0.853	0.863	0.758	0.690	0.648
	G3P-MI	0.842	0.737	0.842	0.842	0.842	0.790	0.842	0.605	0.684
Frecuencia	Txt-kNN	0.874	0.742	0.768	0.895	0.737	0.805	0.647	0.600	0.553
	Citation-kNN	0.853	0.810	0.837	0.863	0.669	0.816	0.679	0.705	0.637
	Fretcit-kNN	0,900	0,837	0,873	0.826	0.805	0.800	0.774	0.716	0.705
	G3P-MI	0.868	0.921	0.868	0.895	0.763	0.816	0.474	0.553	0.553
	Precisión									
	Algoritmo	V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8	V_9
Booleana	Txt-kNN	0.370	0.271	0.365	0.905	0.735	0.789	0.478	0.597	0.539
	Citation-kNN	0.400	0.200	0.409	0.946	0.830	0.849	0.613	0.732	0.758
	Fretcit-kNN	0.573	0.348	0.620	0.954	0.855	0.879	0.728	0.739	0.687
	G3P-MI	0.333	0.231	0.571	0.936	0.889	0.862	0.917	0.576	0.688
Frecuencia	Txt-kNN	0.424	0.200	0.354	0.905	0.761	0.797	0.556	0.601	0.529
	Citation-kNN	0.343	0.276	0.670	0.921	0.821	0.832	0.616	0.732	0.737
	Fretcit-kNN	0.500	0.667	0.714	0.935	0.810	0.824	0.722	0.788	0.895
	G3P-MI	0.429	0.500	0.625	0.892	0.750	0.806	0.444	0.541	0.514
	Alcance									
	Algoritmo	V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8	V_9
Booleana	Txt-kNN	0.450	0.933	0.571	0.976	0.978	1.000	0.713	0.730	0.722
	Citation-kNN	0.300	0.600	0.400	0.836	0.830	0.924	0.663	0.630	0.389
	Fretcit-kNN	0.600	0.667	0.600	0.866	0.956	0.952	0.700	0.641	0.467
	G3P-MI	0.500	1.000	0.571	0.879	0.889	0.862	0.688	0.950	0.611
Frecuencia	Txt-kNN	0.550	0.733	0.315	0.982	0.919	1.000	0.813	0.710	0.700
	Citation-kNN	0.350	0.734	0.286	0.921	0.682	0.952	0.650	0.700	0.433
	Fretcit-kNN	0.533	0.283	0.643	0.860	0.948	0.938	0.750	0.631	0.433
	G3P-MI	0.750	1.000	0.714	1.000	1.000	1.000	1.000	1.000	1.000

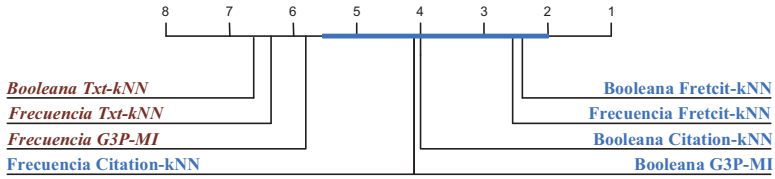
Tabla 6.6: Resultados del test de Friedman (comparativa con la propuesta G3P-MI)

Test de Friedman			
Medida	Valor de Friedman	$\chi^2(\rho = 0{,}1)$	Conclusión
Exactitud	15.7314	18.475	Aceptar hipótesis nula
Alcance	28.5185	18.475	Rechazar hipótesis nula
Precisión	37.1110	18.475	Rechazar hipótesis nula

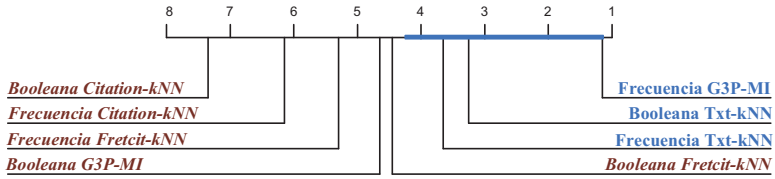
Para hacer una comparación empírica entre los diferentes métodos, usamos el test de Friedman [56] sobre los resultados mostrados en la Tabla 6.5. Los resultados de este test podemos verlos en la tabla 6.6, para las medidas de precisión, alcance y exactitud. El test acepta la hipótesis nula para los valores de exactitud, esto significa que se puede determinar estadísticamente que no existen diferencias significativas para esta medida con respecto a los resultados obtenidos por los diferentes algoritmos. Para las medidas de precisión y alcance, el test rechaza la hipótesis nula, indicando la existencia de diferencias significativa entre los resultados de los diferentes algoritmos. Debido a estos resultados se ha realizado un test a posteriori, el test de Bonferroni Dunn.

En las figuras 6.5(a) y 6.5(b), podemos ver la aplicación de este test. El intervalo de la diferencia crítica (CD) especificado por este test es indicado por una línea más gruesa en las figuras 6.5. Esta línea representa los valores de los rangos entre los que no se determina que estadísticamente muestren peores resultados. Cualquier rango del algoritmo fuera de esta línea puede ser considerado significativamente diferente del algoritmo de control (este algoritmo, es el algoritmo con los valores más bajos del *ranking*). Podemos ver que con el valor de precision, nuestra propuesta obtiene los mejores valores y podemos afirmar que cinco algoritmos son peores que nuestra propuesta porque ellos obtienen un rango fuera del umbral establecido por el test. En el caso de los valores de alcance, Fretcit-KNN obtiene el mejor valor, y en este caso, el test no considera diferencias significativas con respecto a nuestra propuesta booleana.

Resumiendo, de acuerdo a los test estadísticos, Fretcit-kNN es el algoritmo de control para la medida de alcance, pero nuestra propuesta basada en una representación booleana no es significativamente diferente de acuerdo al test estadístico de Bonferroni-Dunn. Por otro lado, nuestra propuesta basada en una representación



(a) Comparando la medida de Alcance



(b) Comparando la medida de Precisión

Figura 6.5: Resultados del test de Bonferroni-Dunn ($p < 0{,}05$) para resolver el problema de recomendación

de las frecuencias para las medidas de precisión es considerada el algoritmo de control, y en este caso, los resultados del test de Bonferroni-Dunn muestran que las dos versiones Fretcit-KNN son significativamente diferentes con respecto a la nuestra. Por lo tanto, desde un punto de vista estadístico, podemos indicar que los resultados de G3P-MI son ligeramente mejores que los resultados de Fretcit-KNN. Sin embargo, estos resultados no son de gran relevancia, pues ambos métodos funcionan mejor en una medida que en otra y finalmente en la medida de exactitud no existen diferencias entre los métodos, la principal contribución de nuestro modelo es que añade comprensibilidad y claridad al conocimiento que aporta. Este punto será tratado en profundidad en la siguiente sección.

6.3.2.3. Reglas obtenidas por G3P-MI

En muchos casos la utilidad del sistema de clasificación está íntimamente relacionada con la comprensibilidad del modelo inferido, son muchas las técnicas que se están construyendo para facilitar la tarea de que un experto entienda bien las salidas de los sistemas de clasificación, siempre que estos no actúen como cajas negras que no reportan ninguna información. Algunas técnicas que se están perfeccionando son usar representaciones gráficas, convertir los patrones a lenguaje natural o utilizar técnicas de visualización de los datos.

La comprensibilidad del modelo es una cuestión muy subjetiva, ya que un modelo puede ser poco comprensible para un usuario y muy comprensible para otro. A pesar de ello, podemos encontrar algunas medidas generales. Así, cuando el modelo se expresa como un conjunto de reglas, cuanto más cortas sean (es decir, cuanto menor sea el número condiciones) más comprensible suele ser el modelo. Sin embargo, la longitud no es el único factor que influye en la comprensibilidad del modelo. Deberían también tenerse en cuenta las preferencias de los usuarios y otras cuestiones semánticas, como por ejemplo, el nivel de abstracción usado para expresar los valores de los parámetros en el modelo puede influir en su comprensibilidad. En general, cuanto más alto es el nivel de abstracción mayor será la comprensibilidad.

En este sentido, una de las ventajas de nuestro sistema es que genera un clasificador formado por reglas SI-ENTONCES, que proporcionan un modelo que nos aporta conocimiento tras la clasificación mediante las reglas. Estas reglas se caracterizan por varias propiedades deseables: son simples, intuitivas, fáciles de comprender y proporcionan información representativa. Estas cualidades representan una ventaja con respecto a los otros algoritmos que actúan como cajas negras dando una salida incomprensible que no permite obtener información sobre los intereses de los usuarios. En este problema es de gran trascendencia la comprensibilidad para poder generar modelos de usuario que nos permitan aprender por un lado sus intereses y por otro actuar con más eficiencia para nuevas recomendaciones al estar el modelo de usuario creado.

Nuestro sistema añade comprensibilidad y claridad a la información descubierta, siguiendo un lenguaje muy intuitivo y fácil de comprender, de acuerdo a las gramáticas establecidas en la sección 4.2.1 y haciendo posible acelerar la velocidad del proceso de predecir nuevos ejemplos debido a que solamente tienen que aplicar el clasificador para ver si unen con su antecedente.

Mostraremos un ejemplo de regla obtenida para cada uno de los usuarios. En primer lugar mostraremos la representación booleana.

SI (*(contiene* ***planet*** *AND contiene* ***forecast****) OR*
(contiene ***atmospheric****) OR (no_contiene* ***football****)*)
ENTONCES *Recomendar la página al usuario* V_1.
SINO *No recomendar la página al usuario* V_1.

Por medio de esta regla podemos aprender los temas que pueden ser recomendados al usuario. Así, el usuario 1 está interesado en temas como ecología y predicción del entorno, que podemos pronosticar por medio de palabras como *planeta*, *previsión* y *atmosférico* y no está interesado en el fútbol.

SI ((*contiene* ***science***) *OR (contiene* ***any***) *OR (contiene* ***nations***)
OR (contiene ***online***) *OR (contiene* ***technology***) OR
(contiene ***economic***))
ENTONCES *Recomendar la página al usuario* V_2.
SINO *No recomendar la página al usuario* V_2.

En el caso del usuario 2, la regla que se obtiene nos informa que entre las preferencias del usuario se encuentra temas relacionados con tecnología, economía y ciencia.

SI ((*contiene* ***sydney***) *OR (contiene* ***rat***) *OR (contiene* ***demand***)
OR (contiene ***northern***) *OR (contiene* ***science***) *OR*
(contiene ***monkey***))
ENTONCES *Recomendar la página al usuario* V_3.
SINO *No recomendar la página al usuario* V_3.

El usuario 3 está interesado en páginas web índice que contengan asuntos relacionados con demandas, sydney, ciencia y monos.

SI ((*contiene* ***french***) *OR (contiene* ***security***) *OR (contiene* ***car***)
OR (contiene ***senate***) *OR (contain* ***shooting***) *OR (contiene* ***web***)
OR (contiene ***government***) *OR (contiene* ***harris***))
ENTONCES *Recomendar la página al usuario* V_4.
SINO *No recomendar la página al usuario* V_4.

En el caso del usuario 4, las páginas que le interesan son las que contienen temas relacionados entre otros con el francés, la seguridad, los coches, el gobierno y el senado.

SI (*(contiene* ***web****) OR (contiene* ***former****) OR (contiene* ***steward****)*
OR (contiene ***security****) OR (contain* ***officials****) OR*
(contiene ***soccer****)*)
ENTONCES *Recomendar la página al usuario* V_5.
SINO *No recomendar la página al usuario* V_5.

Las preferencias del usuario 5 giran en torno a contenidos que versan sobre web, seguridad, oficiales, el fútbol y camareros o auxiliares.

SI (*(contiene* ***korea****) OR (contiene* ***federal****) OR (contiene* ***web****) OR*
(contiene ***afghanistan****) OR (contain* ***taylor****) OR (contiene* ***july****) OR*
(contiene ***russian****) OR (contiene* ***driver****) OR (contiene* ***france****)*)
ENTONCES *Recomendar la página al usuario* V_6.
SINO *No recomendar la página al usuario* V_6.

El usuario 6 estaría interesado en páginas que tratan sobre temas relacionados con Corea, web, Afganistán, Francia, Rusia y los conductores.

SI ((*contiene* ***car****) OR (contiene* ***data****) OR (contiene* ***software****) OR*
(contiene ***afghanistan****) OR (contain* ***fbi****) OR (contiene* ***england****)*)
ENTONCES *Recomendar la página al usuario* V_7.
SINO *No recomendar la página al usuario* V_7.

Las preferencias del usuario 7 se centran en contenidos relacionados con los coches, el software, el FBI, Aghanistán e Inglaterra.

SI ((*contiene* ***law****) OR (contiene* ***government****) OR (contiene* ***software****)*
OR ((no_contiene ***microsoft****) AND (contain* ***car****)) OR*
((contiene ***government****) AND (no_contiene* ***next****) AND*
(contiene ***manager****) AND (no_contiene* ***law****)*)
ENTONCES *Recomendar la página al usuario* V_8.
SINO *No recomendar la página al usuario* V_8.

El usuario 8 muestra interés por el software, las leyes, el gobierno y el software, los coches y no muestra interés por temas relacionados con microsoft.

SI (((*contiene* ***months***) *AND (contiene* ***china***)) *OR ((contiene* ***army***)
AND (contiene ***french***)) *OR ((no_contain* ***korea***) *AND*
(contain ***search***) *AND (contiene* ***full***)) *OR (contiene* ***korea***)
OR (contiene ***germany***) *OR (contiene* ***car***))
ENTONCES *Recomendar la página al usuario* V_9.
SINO *No recomendar la página al usuario* V_9.

Finalmente, el usuario 9 presenta preferencias relacionados con China, el ejército, los franceses, las búsquedas y Alemania.
A continuación, mostraremos ejemplos de reglas utilizando la representación numérica.

SI ((***french*** $>$ ***16***) *OR (****house*** $>$ *11) OR (****science*** $>$ *2)*
*OR ((****aol*** $\leq$ *20) AND* ***on-line*** $>$ *4))* **)**
ENTONCES *Recomendar la página al usuario* V_1.
SINO *No recomendar la página al usuario* V_1.

En este caso las reglas indican no sólo si el término debe aparecer o no, sino también indican la frecuencia en la que el término debe aparecer. En el caso del usuario 1 se identifican intereses en los franceses, las casas y la ciencia, debiendo los dos primeros términos aparecer con mayor frecuencia en la página.

SI ((***festival*** $>$ *22) OR (****government*** $>$ *11) OR (****experts*** $>$ *5) OR*
*(****reuters*** $>$ *8) OR ((****august*** $>$ *34) AND (****government*** $>$ *11))*
*OR (****pm*** $>$ *37) OR (****stone*** $>$ *0) OR (****august*** $>$ *34)* **)**
ENTONCES *Recomendar la página al usuario* V_2.
SINO *No recomendar la página al usuario* V_2.

En el caso del usuario 2, una de las palabras que debe aparecer con más frecuencia es Agosto y festival, otros intereses se encontrarían en temas relacionados con el gobierno y los expertos.

SI (*((**bin** > 19) AND (**florida** ≤ 5)) OR (**australian** > 8) OR*
*(**company** > 8) OR (**wireless** > 7) OR (**violence** > 2)*
*OR (**routers** > 11)*)
ENTONCES *Recomendar la página al usuario* V_3.
SINO *No recomendar la página al usuario* V_3.

El usuario 3 considera interesantes temas relacionados con los routers, la tecnología inalámbrica y los australianos, con menor frecuencia de aparición también le interesan páginas que contengan temas relacionados con Florida y la violencia.

SI (*(**hall** ≤ 1) OR ((**information** ≤ 5) AND (**study** ≤ 0) AND*
*(**stocks** ≤ 37) AND (**black** ≤ 27) AND (**brazil** > 10) AND*
*(**site** > 5) AND (**study** ≤ 0))*)
ENTONCES *Recomendar la página al usuario* V_4.
SINO *No recomendar la página al usuario* V_4.

En el caso del usuario 4, las páginas que más le interesan versan sobre Brasil, tiendas, información y estudio.

SI (*(**health** ≤ 0) OR ((**center** ≤ 38) AND (**fight** > 14) AND*
*(**party** > 36) AND (**financial** > 11) AND (**stories** > 37) AND*
*(**show** ≤ 38) AND (**week** ≤ 6) AND (**cut** ≤ 37))*)
ENTONCES *Recomendar la página al usuario* V_5.
SINO *No recomendar la página al usuario* V_5.

Las preferencias del usuario 5 giran en torno a contenidos que versan sobre salud, el centro, las anécdotas, las financias, las fiestas y las peleas.

SI (*(**five** > 0) OR ((**health** ≤ 5) AND (**inspectors** ≤ 35) AND*
*(**golf** ≤ 1) AND (**way** ≤ 1)) OR ((**attacks** > 7) AND (**right** ≤ 33)*
*AND (**d** > 29) AND (**site** ≤ 3) AND (**information** ≤ 4)) OR*
*((**blood** > 20) AND (**vs** ≤ 27) AND (**support** > 33))*)
ENTONCES *Recomendar la página al usuario* V_6.
SINO *No recomendar la página al usuario* V_6.

El usuario 6 estaría interesado en páginas que tratan sobre temas relacionados con inspectores, golf, información y salud.

SI (*(**decision** > 3) OR ((**fox** ≤ 0) AND (**football** > 0) AND (**even** > 26)) OR ((**award** ≤ 5) AND (**peace** > 12)) OR ((**fox** ≤ 0) AND (**medical** ≤ 0))*)
ENTONCES *Recomendar la página al usuario* V_7.
SINO *No recomendar la página al usuario* V_7.

Las preferencias del usuario 7 se centran en contenidos relacionados con la fox, el fútbol, indemnizaciones y medicina.

SI (*(**trade** > 2) OR ((**guidelines** > 2) AND (**season** ≤ 11)) OR ((**day** > 7) AND (**g** ≤ 25)) OR (**day** > 32) OR (**bin** > 4) OR (**search** > 0) OR (**engine** > 0) OR ((**fox** ≤ 14) AND (**korea** > 7)) OR (**quarter** > 25) OR ((**before** ≤ 6) AND (**manager** > 3))*)
ENTONCES *Recomendar la página al usuario* V_8.
SINO *No recomendar la página al usuario* V_8.

El usuario 8 muestra interés por el comercio, las estaciones, los callejeros, Corea y los gerentes.

SI (*((**guard** > 26) AND (**used** ≤ 27) AND (**record** ≤ 19) AND (**never** ≤ 36) AND (**football** ≤ 39))*)
ENTONCES *Recomendar la página al usuario* V_9.
SINO *No recomendar la página al usuario* V_9.

Finalmente, el usuario 9 presenta preferencias relacionados con la vigilancia, los registros y el fútbol.

En general, después de mostrar ejemplos de los dos tipos de representación que se han considerado en la generación de las reglas, podemos ver que la representación numérica resulta en algunos casos más compleja para identificar las preferencias de los usuarios, debido a que los intereses de los mismos se encuentran limitados por

la aparición de un término un número mínimo o máximo de veces más que simplemente delimitados por si el término aparece o no como ocurre con la representación booleana. Por esto, aunque ambas representaciones obtienen resultados similares, podemos concluir que la representación numérica resulta menos efectiva debido a que obtiene reglas menos comprensibles.

6.4. Aplicación del modelo MOG3P-MI

6.4.1. Adaptación del modelo MOG3P-MI

Para la utilización del modelo MOG3P-MI hemos seleccionado la configuración que mejores resultados produjo en la versión mono-objetivo. Esto es, la versión que elimina las palabras menos significativas será la utilizada para llevar a cabo la comparativa. Con respecto a las gramáticas empleadas, serán las mismas que se especificaron en la sección 6.3.1.1, una para la representación booleana y otra para la numérica.

6.4.1.1. Función de evaluación

Como ya se ha estudiado la evaluación de la efectividad de la recomendación no es una tarea trivial, debido a que normalmente los datos no están balanceados, por ello la exactitud no es suficiente por sí sola para evaluar la calidad de los clasificadores, ya que beneficiaría siempre la clasificación de la clase mayoritaria. Por esta razón, se han considerado otras medidas, tales como, la sensibilidad y la especificidad. La sensibilidad nos indica el porcentaje de documentos aceptados que son de hecho relevantes para el usuario, es decir:

$$sensibilidad = \frac{\#interesantes\ y\ recomendados}{\#interesantes} \tag{6.5}$$

y la *especificidad* es el porcentaje de documentos no aceptados, que son de hecho no relevantes para el usuario, es decir:

$$especificidad = \frac{\#no\ interesantes\ y\ no\ recomendados}{\#no\ interesantes} \tag{6.6}$$

Estas medidas son contrapuestas, por ello con este algoritmo intentaremos obtener el conjunto de soluciones no dominadas con respecto a ambas. No se ha introducido ninguna otra medida relacionada con la comprensibilidad, debido a que las reglas obtenidas son suficientemente simples para añadir mayor complejidad con otro objetivo.

6.4.1.2. Parámetros de configuración

Los parámetros más relevantes del algoritmo con respecto al tamaño de la población, número de generaciones, probabilidad de cruce y mutación y método de selección, se muestran en la tabla 6.7.

Tabla 6.7: Parámetros de ejecución en Recomendación de Páginas Índice

Algoritmo MOG3P-MI	
Parámetros	Valores
Tamaño de la población externa	100
Tamaño de la población	1000
Número de generaciones	100
Probabilidad de cruce	95 %
Probabilidad de mutación	60 %
Porcentaje de elistismo	5 %
Método de selección de padres	*Torneo Binario*
Profundidad máxima del árbol	50

6.4.2. Resultados experimentales

En esta sección analizaremos los resultados obtenidos desde varias perspectivas. En primer lugar, se analizará el frente de Pareto obtenido por nuestra propuesta para cada usuario, no realizaremos comparaciones, pues en este caso es la única propuesta multi-objetivo que se ha aplicado al problema. En segundo lugar, se realizará una comparación de los resultados globales de exactitud, sensibilidad y especificidad considerando las diferentes técnicas que se han aplicado al problema hasta la fecha. Después, se estudiarán los resultados obtenidos, considerando los

diferentes grupos de usuario de forma separada. Finalmente, evaluamos la calidad y comprensibilidad de los modelos de usuario obtenidos por nuestra propuesta.

Para la experimentación realizada se procedió a la ejecución de nuestros modelos, G3P-MI y MOG3P-MI, para cada conjunto de datos, con 5 semillas diferentes y los valores medios de *exactitud*, *sensibilidad* y *especificidad* serán los valores que se indiquen.

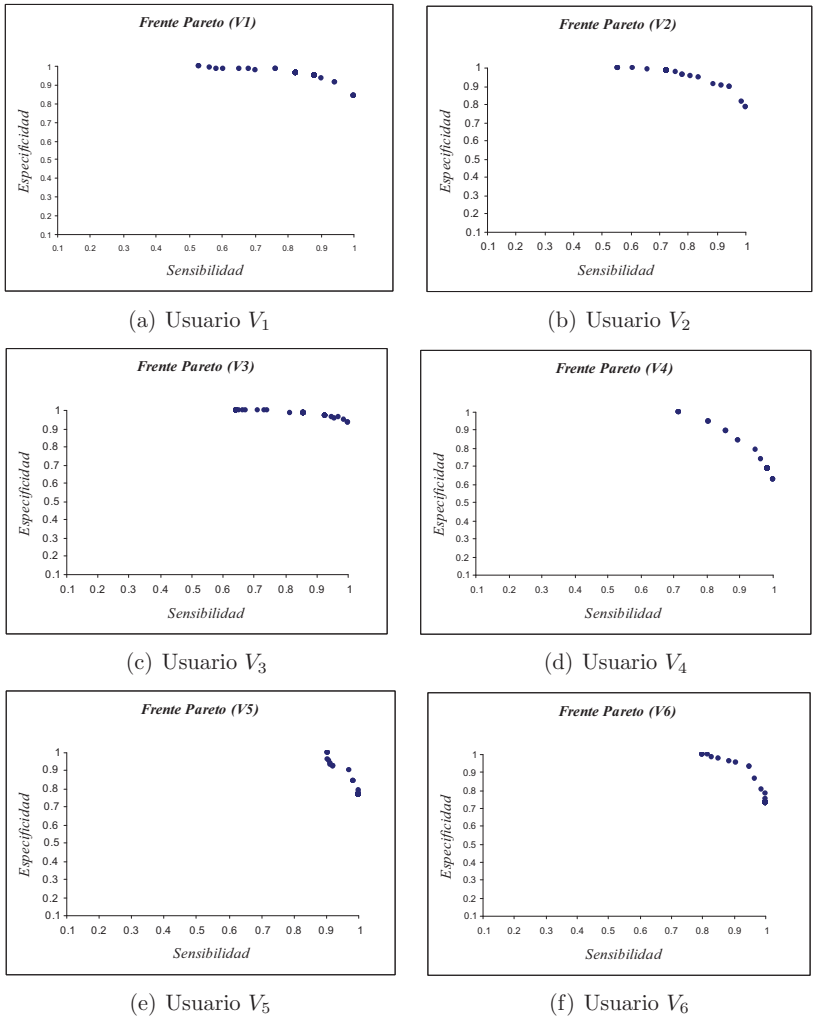
6.4.2.1. Evaluación de los resultados obtenidos por MOG3P-MI

Nuestra propuesta aplica un enfoque multi-objetivo caracterizado por la generación de un POF. La optimalidad del Pareto, como se ha estudiado, es un concepto clave en la optimización multi-objetivo. Una solución se dice que pertenece al POF si no es dominada por ninguna otra solución del espacio de búsqueda, es decir, el POF está formado por todas las conclusiones óptimas obtenidas. Las figuras 6.6 y 6.7 muestran los POF obtenidos para los diferentes conjunto de datos de entrenamiento con los que se ha trabajado. En estas figuras podemos ver que las soluciones oscilan entre una optimización total de sensibilidad y especificidad hasta soluciones que mantienen un equilibrio entre ambos valores. La solución con mayor equilibrio entre ambas medidas, será el clasificador seleccionado por nuestra metodología para comparar sus resultados en las otras técnicas que se han desarrollado para solucionar este problema.

6.4.2.2. Comparación de resultados globales con otras propuestas aplicadas al problema

En esta sección, comparamos los valores globales de exactitud, sensibilidad y especificidad de todos los algoritmos considerados en este estudio. La Tabla 6.8 muestra los resultados medios obtenidos para todos los conjuntos de datos disponibles. Esta tabla está dividida en dos secciones. La primera corresponde a los resultados obtenidos con la representación booleana de las páginas, mientras que la segunda sección corresponde a los resultados utilizando una representación basada en la frecuencias de los términos.

Podemos ver, que MOG3P-MI logra los resultados más precisos, obteniendo los modelos más exactos tanto con representación numérica como booleana. Con respecto al algoritmo G3P-MI (la versión mono-objetivo), este algoritmo consigue peores

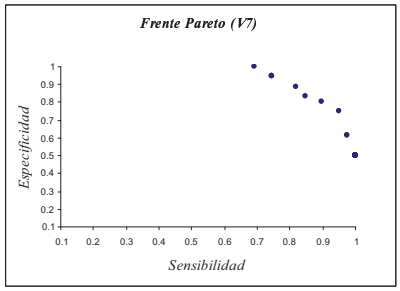


(a) Usuario V_1 (b) Usuario V_2

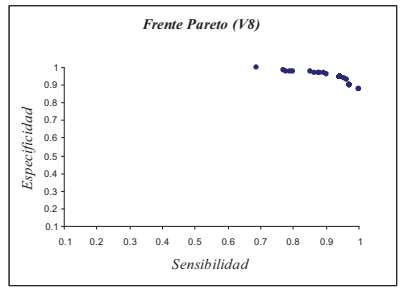
(c) Usuario V_3 (d) Usuario V_4

(e) Usuario V_5 (f) Usuario V_6

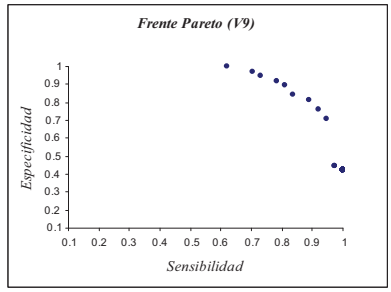
Figura 6.6: POF generado por MOG3P-MI para el problema de recomendación de páginas web índice (Usuario1 - Usuario6)



(a) Usuario V_7



(b) Usuario V_8



(c) Usuario V_9

Figura 6.7: POF generado por MOG3P-MI para el problema de recomendación de páginas web índice (Usuario7 - Usuario9)

resultados con una representación booleana. Y aunque la representación numérica obtiene valores de sensibilidad ligeramente más altos, los valores de especificidad se ven decrementados radicalmente. Esto significa que sus modelos no identifican correctamente lo que no les interesa a los usuarios. Con respecto a las otras técnicas (variantes KNN), todas ellas muestran peores resultados en valores de exactitud.

Se ha llevado a cabo un estudio estadístico para comparar los resultados obtenidos. Primero el test de Friedman [56] se ha aplicado para determinar si hay diferencias significativas entre los algoritmos teniendo en cuenta las tres medidas de exactitud, sensibilidad y especificidad. Los rangos medios de todos los algoritmos se pueden consultar en la tabla 6.9. De acuerdo a los resultados del test de Friedman mostrados en la tabla 6.10, la hipótesis nula es rechazada para todas las medidas; es decir, hay diferencias significativas en los resultados de todas las métricas para los diferentes algoritmo considerados. En segundo lugar, hemos usado un test a posteriori, el test de Bonferroni-Dunn para determinar qué algoritmos pueden ser considerados peores

Tabla 6.8: Resultados Globales MOG3P-MI

Resultados Globales				
	Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
Booleana	Txt-KNN	0.723	0.738	0.485
	Citation-KNN	0.758	0.607	0.741
	Fretcit-kNN	0.810	0.701	0.780
	G3P-MI	0.781	0.772	0.730
	MOG3P-MI	0.848	0.779	0.757
Frecuencias	Txt-KNN	0.763	0.613	0.721
	Citation-KNN	0.736	0.702	0.528
	Fretcit-kNN	0.804	0.712	0.742
	G3P-MI	0.731	0.940	0.401
	MOG3P-MI	0.842	0.873	0.704

propuestas con respecto a los resultados que ofrecen. Los resultados de este test se muestran en la figura 6.8 que muestra que todos los algoritmos con valores fuera del intervalo representado por la línea horizontal más gruesa obtienen resultados peores que el algoritmo de control.

Con respecto a la medida de exactitud, (Figura 6.8(a)), MOG3P-MI es el mejor algoritmo debido a que tiene los valores de rangos más bajos. Además, Fretcit-kNN, Frequency Fretcit-kNN y Frequency MOG3P-MI no son significativamente diferentes del algoritmo base que es MOG3P-MI.

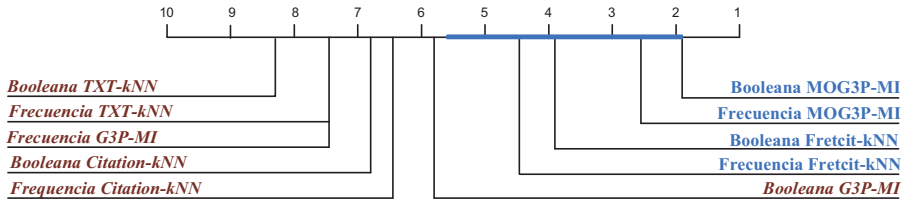
Se ha realizado un estudio similar para las otras métricas, como se puede ver en las Figuras 6.8(b) y 6.8(c). G3P-MI utilizando representación de las frecuencias de los términos consigue los mejores resultados para sensibilidad a expensas de ser uno de los peores algoritmos en especificidad. Por otro lado, Fretcit-kNN logra los resultados mejores para especificidad, pero su valor de sensibilidad es uno de los valores más bajos. En contraste con estos algoritmos, nuestra propuesta produce un sistema de clasificación con los valores más altos tanto en sensibilidad como especificidad. Concretamente, en las dos medidas es un algoritmo cercano al algoritmo de control (de acuerdo al test de Bonferroni-Dunn). Además, este test estadístico determina que no hay diferencias significativas entre nuestro algoritmo y el de control para dicha medida. En conclusión, nuestro algoritmo es más fiable y logra alcanzar un compromiso entre las dos medidas, características que le hacen

Tabla 6.9: Rangos medios de los algoritmos (comparativa con las propuesta MOG3P-MI)

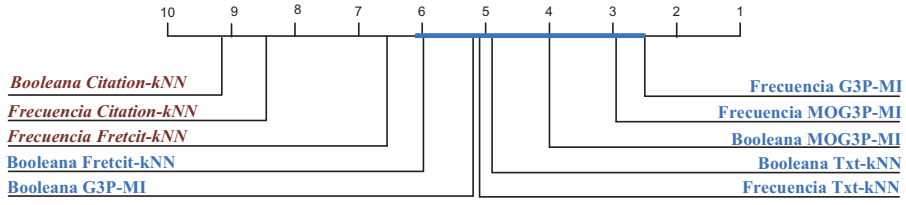
Rangos Medios				
	Algoritmo	Exactitud	Sensibilidad	Especificidad
Booleana	Txt-kNN	8.334	4.889	8.889
	Citation-kNN	6.778	9.332	4.666
	Fretcit-kNN	3.889	5.999	3.111
	G3P-MI	5.778	5.222	5.333
	MOG3P-MI	1.888	4.00	3.665
Frecuencias	Txt-kNN	7.445	5.112	8.000
	Citation-kNN	6.445	8.444	4.444
	Fretcit-kNN	4.445	6.555	3.772
	G3P-MI	7.443	2.500	8.777
	MOG3P-MI	2.556	2.946	4.333

Tabla 6.10: Resultados del test de Friedman (comparativa con la propuesta MOG3P-MI)

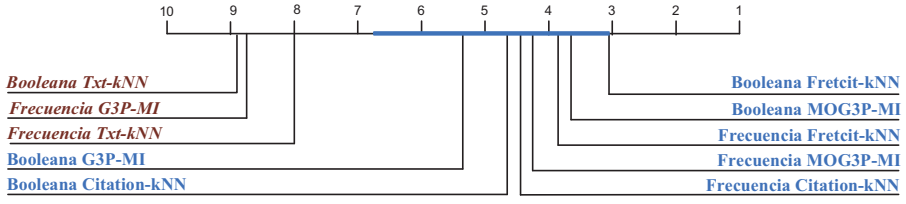
Test de Friedman			
Medida	Valor de Friedman	$\chi^2(\rho = 0{,}1)$	Conclusión
Exactitud	21.666	42.818	Rechazar la hipótesis nula
Sensibilidad	21.666	42.327	Rechazar la hipótesis nula
Especificidad	21.666	42.915	Rechazar la hipótesis nula



(a) Bonferroni-Dunn para exactitud



(b) Bonferroni-Dunn para sensibilidad



(c) Bonferroni-Dunn para especificidad

Figura 6.8: Test de Bonferroni Dunn ($p < 0{,}1$)

finalmente producir las mejores recomendaciones con mayor porcentaje de páginas mostradas que son de su preferencia.

6.4.2.3. Comparación de los resultados en función del tipo de usuario

La tabla 6.11 muestra los resultados agrupados de acuerdo a los diferentes tipos de usuarios. Como puede verse en la primera columna, MOG3P-MI consigue resultados competitivos en el caso de usuarios selectivos, con perfiles muy exactos y específicos (los mejores valores de exactitud y especificidad) sin una pérdida significativa de los valores de sensibilidad. Este resultado es especialmente importante, debido a que aún sin tener suficiente información sobre los intereses de los usuarios, el modelo es capaz de identificar sus preferencias correctamente.

La segunda columna muestra los resultados en el caso de usuarios permisivos. Como puede verse, MOG3P-MI obtiene los mejores resultados con respecto a la medida

Tabla 6.11: Comparación de resultados de acuerdo a los diferentes tipos de usuarios

Algoritmo		Usuarios Selectivos			Usuarios Permisivos			Usuarios Balanceados		
		Exa	*Se*	*Sp*	*Exa*	*Se*	*Sp*	*Exa*	*Se*	*Sp*
Booleana	Txt-KNN	0.795	0.636	0.822	0.805	0.863	0.194	0.570	0.715	0.438
	Citation-KNN	0.803	0.397	0.868	0.796	0.863	0.577	0.674	0.562	0.777
	Fretcit-kNN	0.879	0.579	0.919	0.854	0.924	0.634	0.698	0.599	0.788
	G3P-MI	0.807	0.690	0.919	0.825	0.877	0.628	0.711	0.750	0.642
	MOG3P-MI	0.904	0.579	0.950	0.868	0.975	0.557	0.772	0.784	0.763
Frecuencias	Txt-KNN	0.795	0.519	0.843	0.812	0.851	0.264	0.600	0.736	0.478
	Citation-KNN	0.833	0.402	0.907	0.782	0.851	0.498	0.674	0.586	0.757
	Fretcit-kNN	0.870	0.615	0.904	0.811	0.916	0.470	0.732	0.604	0.852
	G3P-MI	0.845	0.821	0.904	0.823	1.000	0.201	0.526	1.000	0.099
	MOG3P-MI	0.895	0.774	0.919	0.860	1.000	0.466	0.771	0.844	0.726

de sensibilidad y similares resultados para la medida de especificidad. En este caso, trabajando en un escenario MIL, es más difícil porque aunque se tiene suficiente información sobre los intereses de los usuarios, no se conocen los enlaces específicos que son de su interés; solamente conocemos que la página contiene al menos un enlace de interés para el usuario pero no hay más información. Incluso así, nuestro algoritmo obtiene resultados competitivos, mejorando la exactitud obtenida con respecto a los otros algoritmos.

Finalmente, la última columna muestra los resultados para los usuarios balanceados. En este caso, nuestro algoritmo se mantiene fiable, proporcionando los mejores resultados tanto en especificidad como en sensibilidad y exactitud, obteniendo las preferencias de los usuarios de forma acertada.

Podemos concluir que nuestro algoritmo realiza recomendaciones correctas para los diferentes tipos de usuarios, por lo que puede ser considerada una herramienta fiable, logrando resultados más balanceados con respeto a todas las medidas cuando la información sobre el usuario no está balanceada y obteniendo los mejores valores en todas las medidas cuando esta información está balanceada.

6.4.2.4. Reglas obtenidas por el modelo G3P-MI multi-objetivo

Nuestro modelo genera reglas en formato SI-ENTONCES. Estas reglas están caracterizadas por ser simples, intuitivas y fáciles de comprender, que nos aportan información sobre los intereses de los usuarios. Como ya se ha estudiado, esta información es cada vez más valiosa para que los expertos puedan extraer información a partir de ella y aprender las preferencias de los usuarios; ésta característica es la principal ventaja que aporta nuestra propuesta para la resolución de este problema.

A continuación mostraremos un ejemplo de regla obtenida para cada uno de los usuarios. En primer lugar mostraremos la representación booleana.

SI (*(no_contiene* ***financial****) OR ((contiene* ***violence****) AND*
(no_contiene ***science****) OR ((no_contiene* ***services****) AND*
(no_contiene ***web****))*)
ENTONCES *Recomendar la página al usuario* V_1.
SINO *No recomendar la página al usuario* V_1.

Por medio de esta regla podemos aprender los temas que pueden ser recomendados al usuario. Así, el usuario V_1 está interesado en términos relacionados con violencia y no tiene interés en finanzas, ni en servicios en la Web.

SI (*((contiene* ***reuters****) AND (contiene* ***mobile****)) OR ((contiene* ***way****) AND*
(contiene ***another****)) OR ((contiene* ***users****) AND (contiene* ***mobile****)) OR*
((contiene ***race****) AND (contiene* ***high****)) OR ((contiene* ***between****)*
AND (contiene ***sri****))*)
ENTONCES *Recomendar la página al usuario* V_2.
SINO *No recomendar la página al usuario* V_2.

En el caso del usuario 2, la regla que se obtiene nos informa que entre las preferencias del usuario se encuentra temas relacionados con móviles, carreras y usuarios.

SI (*((contiene* ***software****) AND ((contiene* ***web****) OR (contiene* ***friends****))*
OR ((contiene ***oil****) AND (contiene* ***analysts****)) OR (contiene* ***mobile****)*
OR ((contiene ***web****) AND (contiene* ***oil****))*)
ENTONCES *Recomendar la página al usuario* V_3.
SINO *No recomendar la página al usuario* V_3.

El usuario 3 está interesado en páginas web índice que contengan asuntos relacionados con software, web, amigos, analista, móviles y petróleo.

SI (*(contiene* ***never****) OR ((contiene* ***health****) AND (contiene* ***car****))*
OR (contiene ***cnn****) OR (contiene* ***french****) OR (contiene* ***best****)*
OR ((contiene ***health****) AND (contiene* ***slideshows****)) OR*
(contiene ***white****)* **)**
ENTONCES *Recomendar la página al usuario* V_4.
SINO *No recomendar la página al usuario* V_4.

En el caso del usuario 4, las páginas que son de su interés son las que contienen temas relacionados entre otros con la salud, los coches, la cnn, el francés y exposición de diapositivas.

SI (*(contiene* ***korea****) OR ((contiene* ***work****) AND (contiene* ***information****))*
OR (contiene ***related****) OR (contiene* ***germany****) OR (contiene* ***street****)*
OR (contiene ***french****) OR (contiene* ***ford****) OR (contiene* ***stories****)*
OR (contiene ***title****)* **)**
ENTONCES *Recomendar la página al usuario* V_5.
SINO *No recomendar la página al usuario* V_5.

Las preferencias del usuario 5 giran en torno a contenidos que versan sobre Corea, trabajo, información, Alemania, francés y las anécdotas.

SI (*(contiene* ***russian****) OR (contiene* ***french****) OR (contiene* ***services****)*
OR (contiene ***germany****) OR (contiene* ***web****) OR (contiene* ***basketball****)*
OR (contiene ***son****) OR (contiene* ***car****) OR (contiene* ***death****) OR*
(contiene ***call****)* **)**
ENTONCES *Recomendar la página al usuario* V_6.
SINO *No recomendar la página al usuario* V_6.

El usuario 6 estaría interesado en páginas que tratan sobre temas relacionados con ruso, francés, servicios, Alemania, web, coches, baloncesto y muerte.

SI (*(contiene* ***english****) OR ((contiene* ***story****) AND (contiene* ***web****))*
OR (contiene ***car****) OR ((contiene* ***university****) AND (contiene* ***last****))*
OR ((contiene ***technology****) AND (contiene* ***ireland****)) OR*
(contiene ***machines****)* **)**
ENTONCES *Recomendar la página al usuario* V_7.
SINO *No recomendar la página al usuario* V_7.

Las preferencias del usuario 7 se centran en contenidos relacionados con los ingleses, las anécdotas, la web, los coches, la universidad, la tecnologías y las máquinas.

SI (*(contiene* ***england****) OR ((contiene* ***heart****) AND (contiene* ***june****))*
OR (contiene ***engines****) OR (contiene* ***internet****) OR ((contiene* ***web****)*
AND (no_contiene ***prison****) AND (no_contiene* ***internet****) AND*
(contiene ***security****) AND (contiene* ***england****))* **)**
ENTONCES *Recomendar la página al usuario* V_8.
SINO *No recomendar la página al usuario* V_8.

El usuario 8 muestra interés por Inglaterra, el corazón, los motores, internet, la web, las prisiones y la seguridad.

SI (*(contiene* ***officials****) OR (contiene* ***arrested****) OR (contiene* ***england****)*
OR (contiene ***cnn****) OR (contiene* ***google****) OR ((contiene* ***crash****)* **)**
ENTONCES *Recomendar la página al usuario* V_9.
SINO *No recomendar la página al usuario* V_9.

Finalmente, el usuario 9 presenta preferencias relacionados con los oficiales, las detenciones, Inglaterra, la cnn, google y los accidentes.

A continuación, mostraremos ejemplos de reglas obtenidas con la representación numérica para cada uno de los usuarios.

SI (*(****french*** > 16*) OR (****house*** > 11*) OR (****online*** > 6*)*
*OR ((****science*** > 2*) AND (****edt*** > 20 *)) OR (****aol*** > 7*)* **)**
ENTONCES *Recomendar la página al usuario* V_1.
SINO *No recomendar la página al usuario* V_1.

El usuario 1 estaría interesado en los franceses, las casas y la ciencia. En este caso los diferentes temas deben tener una frecuencia de aparición mínima en la página para que sean considerados significativos, así por ejemplo, los dos primeros términos deben aparecer 16 u 11 veces al menos para que llegue a ser interesante, mientra el tema de ciencia se requiere con una menor frecuencia para llegar a intersarse por esa página.

SI (*((**yahoo** > 1) AND (**general** > 0)) OR (**al** > 28) OR (**island** > 13)*
*OR ((**yahoo** > **1**) AND (**lower** > 1)) OR ((**since** > **2**) AND*
*(**yahoo** > 1)) OR ((**yahoo** > **1**) AND ((**wireless** > **8**) OR*
*OR (**general** > 0)))*)
ENTONCES *Recomendar la página al usuario V_2.*
SINO *No recomendar la página al usuario V_2.*

El usuario 2 tiene interés en temas relacionados con yahoo, islas y la tecnología inalámbrica.

SI (*(**sales** > 10) OR (**iraq's** > 0) OR (**online** > 8) OR (**german** > 10)*
*OR (**edt** > **37**)*)
ENTONCES *Recomendar la página al usuario V_3.*
SINO *No recomendar la página al usuario V_3.*

El usuario 3 está interesado en páginas web índice que contengan asuntos relacionados con las ventas y los alemanes.

SI (*((**stories** $\leq$ 1) AND (**stories** > 1)) OR (**health** $\leq$ 1) OR*
*(**microsoft** > 6) OR ((**health** $\leq$ 1) AND (**health** > 1)*
*AND (**china** $\leq$ 10))*)
ENTONCES *Recomendar la página al usuario V_4.*
SINO *No recomendar la página al usuario V_4.*

En el caso del usuario 4, las páginas interesantes son las que contienen temas relacionados entre otros con las anécdotas, la salud y China.

SI (*((**health** ≤ 0) OR ((**market** ≤ 0) AND (**states** > 2) AND (**championship** ≤ 22) AND (**points** ≤ 0) AND (**brown** > 5))*)
ENTONCES *Recomendar la página al usuario* V_5.
SINO *No recomendar la página al usuario* V_5.

Las preferencias del usuario 5 giran en torno a contenidos que versan sobre la salud, el mercado, los estados y los campeonatos.

SI (*((**health** ≤ 2) AND (**yahoo** ≤ 2)) OR (**father** > 3) OR (**cnn** > 9) OR (**attacks** > 2)*)
ENTONCES *Recomendar la página al usuario* V_6.
SINO *No recomendar la página al usuario* V_6.

El usuario 6 estaría interesado en páginas que tratan sobre temas relacionados con la salud, yahoo, la cnn y los atentados.

SI (*(**university** > 9) OR (**qaeda** > 4) OR (**wireless** > 0) OR (**free** > 5) OR (**cell** > 5) OR (**airport** > 0) OR (**english** > 8) OR (**car** > 0)*)
ENTONCES *Recomendar la página al usuario* V_7.
SINO *No recomendar la página al usuario* V_7.

Las preferencias del usuario 7 se centran en contenidos relacionados con la universidad, los aeropuertos, los ingleses y la tecnología inalámbrica.

SI (*(**congress** > 0) OR (**federal** > 13) OR (**international** > 8) OR (**internet** > 11) OR (**message** > 1) OR (**ap** > 12) OR (**engines** > 0) OR (**korean** > 2) OR ((**yahoo** > 3) AND (**football** > 6))*)
ENTONCES *Recomendar la página al usuario* V_8.
SINO *No recomendar la página al usuario* V_8.

El usuario 8 muestra interés por los congresos, la internacionalidad, los mensajes, los motores, los coreanos y el fútbol.

SI (*((**four** ≤ 21) AND (**foreign** > 9) AND (**got** ≤ 32) AND (**half** ≤ 9))*)
ENTONCES *Recomendar la página al usuario* V_9.
SINO *No recomendar la página al usuario* V_9.

Finalmente, el usuario 9 presenta preferencias relacionados con los extranjeros.

Una conclusión similar a la que hemos obtenido en el estudio de las reglas obtenidas por el modelo G3P-MI podemos ver con este modelo. Siendo las reglas con representación booleana más sencillas y fáciles de interpretar.

7

Predicción del Rendimiento Académico de los Estudiantes

7.1. Introducción

El avance de la tecnología junto con el impacto que ha producido Internet en los últimos años ha tenido consecuencias en todas las áreas de nuestra vida. Concretamente, las implicaciones en el ámbito educativo han sido de una magnitud incalculable, caracterizando de forma cada vez más indiscutible la manera de impartir los contenidos académicos. La relación entre tecnología y educación se hace cada vez más patente motivada por la urgente necesidad que tienen las universidades y centros de formación por un lado, de ampliar su oferta educativa y hacerla accesible a un mayor número de estudiante y por otro, para romper con las restricciones establecidas por la docencia tradicional basada en las clases magistrales y llevar a cabo un nuevo aprendizaje basado en la colaboración y cooperación.

Las plataformas virtuales o plataformas e-learning [166] aparecen como medios que promueven este aprendizaje potenciando el desarrollo de actividades cooperativas y colaborativas. Los Entornos Virtuales permiten un mayor acceso a la educación, eliminan las barreras geográficas y temporales proporcionando más flexibilidad,

actualización de materiales, aprendizaje individualizado y realimentación sobre las clases tradicionales, a la vez que facilitan y fomentan el aprendizaje colaborativo y el trabajo en grupo [37].

El uso de las plataformas e-learning acumula una gran cantidad de información al almacenar todas las acciones e interacciones de los estudiantes en distintos medios de almacenamiento, como pueden ser ficheros log o bases de datos. Esta información almacenada ha resultado muy valiosa para detectar posibles errores, deficiencias y mejoras en el rendimiento de los estudiantes y descubrir cómo la motivación del estudiante afecta al modo en el que él o ella interactúa con el software. Lo cierto es que desde que este problema fue identificado, ha habido un interés creciente en analizar esta información almacenada y un considerable número de herramientas automáticas que hacen posible trabajar con grandes cantidades de datos han aparecido. Fausett y Elwasif [66] predijeron las notas de los estudiante clasificadas en cinco clases: A, B, C, D y E o F a partir de las puntuaciones de los test usando redes neuronales; Martínez [128] predijo el éxito académico de los estudiantes (clases que son exitosas o no)usando análisis de funciones discriminantes; Minaei-Bidgoli y Punch [135] clasificaron estudiantes usando algoritmos genéticos para predecir su grado final; Superby, Vandamme y Meskens [187] predijeron el éxito académico de los estudiantes (clasificados en bajo, medio y alto) usando para ello diferentes métodos de minería de datos; Kotsiantis y Pintelas [116] predijeron las notas de los estudiantes (aprobar o suspender) usando técnicas de regresión con datos de Universidad de Open Hellenic, Delgado et al. [55] usaron modelos de redes neuronales a partir de los ficheros log del Moodle.

Este problema siempre ha sido representado y tratado desde una perspectiva del aprendizaje supervisado tradicional. Sin embargo, la particularidad esencial cuando nos enfrentamos al problema es que la información es incompleta debido a que cada curso tiene diferente tipos y número de actividades, y cada estudiante lleva a cabo el número y tipo de actividades que considera oportuno, dedicando más o menos tiempo a resolverlas. MIL permite una representación más apropiada almacenando la información general que poseen todos los alumnos por medio de un número fijo de atributos que pertenecen al ejemplo y la información específica que depende del trabajo particular que ha realizado cada alumno por medio de un número variable de instancias. En este capítulo proponemos tanto una representación mediante

aprendizaje supervisado tradicional y una primera propuesta para trabajar en un escenario MIL, de forma que se nos permita comparar ambas representaciones con los paradigmas más representativos y comprobar si la representación con múltiples instancias resulta más efectiva.

7.2. Descripción del problema

Predecir el rendimiento de los estudiantes de acuerdo a su trabajo realizado en una determinada plataforma virtual nos permite encontrar relaciones entre las actividades y los recursos que determinen cuáles de ellas resultan ser más interesantes para favorecer y mejorar el proceso de aprendizaje. Así, se puede determinar si todo el material adicional proporcionado a los estudiantes (tareas basadas en la web) les ayuda a fortalecer los conceptos y temas desarrollados en clase o si algunas actividades son más indicadas para mejorar los resultados finales. El problema puede formularse como sigue, un estudiante podría hacer diferentes actividades en un curso dedicadas a fortalecer los conceptos adquiridos en clase. Más tarde, al final del curso, los estudiantes se enfrentan a un examen final. Un estudiante que obtiene una nota mayor o igual a un mínimo fijado pasará el módulo o lección mientras un estudiante con un nota menor que este mínimo no aprobará ese módulo. Con esta premisa, el problema consiste en predecir si el estudiante pasará o no un modulo considerando el número, tiempo y tipo de actividades que el/ella ha realizado durante un periodo previo a la finalización del curso. A continuación, se especificará la información disponible y se describirá una representación basada tanto en el aprendizaje supervisado tradicional como en el aprendizaje con múltiples instancias.

7.2.1. Actividades consideradas en las plataformas de aprendizaje virtual

Hay una enorme variedad de plataformas e-learning, la mayoría de ellas tienen características y servicios comunes. Hoy en día, una de las más empleadas es Moodle (Modular Object Oriented Developmental Learning Environment), un sistema de

aprendizaje capaz de la creación de cursos y experiencias en línea muy potente y flexible [161], que además es de software libre.

El sistema Moodle almacena en una base de datos relacional una gran cantidad de información sobre el contenido del curso, tanto datos personales de los usuarios como del trabajo que han realizado en el sistema. En este trabajo emplearemos la información almacenada sobre tres actividades: cuestionarios, tareas y foros.

- Los cuestionarios son una herramienta muy potente y extremadamente flexible que permite diseñar diferentes estrategias de evaluación. Se puede utilizar en evaluaciones iniciales (para tener una primera idea del grado de conocimientos y habilidades por parte de los estudiantes) o en exámenes tipo test (con la ventaja de que el cuestionario se puede generar aleatoriamente y que su corrección sea inmediata). En todos los casos de autoevaluación, facilita a los estudiantes la monitorización de su propio rendimiento y como instrumento de refuerzo y repaso.

- Las tareas permiten recoger los trabajos y actividades del estudiante. Es una herramienta útil para permitir a los estudiantes subir contenido digital que se quedará almacenados para su posterior evaluación en la que se podrá añadir un comentario que llegará de forma independiente al estudiante mediante correo electrónico. Se les puede solicitar subir redacciones, hojas de cálculo, presentaciones, páginas web, fotografías o fragmentos de video o audio.

- Finalmente, los foros son una herramienta de comunicación muy potente. La comunicación provee de una gran cantidad de beneficios, como son disminuir la sensación de aislamiento, proporcionar mayor flexibilidad, dar independencia tanto geográfica como horaria y permitir la interactividad. Los foros permiten realizar una reflexión antes de plantear un tema o responder una pregunta y no requiere la conexión simultánea de los usuarios para poder participar y recibir los mensajes. De esta manera, cada participante puede expresar su propia visión sobre un determinado tema y fomentar la participación de aquellos con más problemas de interactuar de forma inmediata. Los foros proporcionan muchas posibilidades para su utilización, se puede emplear para repetir las conversaciones tenidas en clase si se desea recalcar

algún aspecto, formular proyectos para discutir entre grupos de estudiantes o llevar las mejores ideas y preguntas al foro para que las conozca toda la clase.

Un resumen de la información considerada para cada actividad en nuestro estudio se muestra en la tabla 7.1.

Tabla 7.1: Actividades consideradas en el trabajo de los estudiantes

Información de las Actividades		
Tipo	***Atributo***	***Descripción***
ASSIGNMENTS (Tareas)	numberAssignment	Número de tareas/prácticas hechas por el alumno en el curso.
	timeAssignment	Tiempo total en segundos que el usuario ha dedicado a realizar tareas.
FORUMS (Foros)	numberPosts	Número de mensajes enviados al foro por el usuario.
	numberRead	Número de mensajes leídos en el foro por el usuario
	timeForum	Tiempo total en segundos que el usuario ha estado en la sección de foros.
QUIZZES (Cuestinarios)	numberQuiz	Número de cuestionarios vistos por el usuario.
	numberQuiz_a	Número de cuestionarios realizados y aprobados por el usuario.
	numberQuiz_s	Número de cuestionarios realizados y suspendidos por el usuario.
	timeQuiz	Tiempo total en segundos que el usuario ha estado en la sección de cuestionarios.

7.2.2. Representación del problema

Una vez definido el problema y la información de las actividades que se tienen en cuenta en este estudio, se van a presentar dos representaciones que nos permitan abordar el problema mediante algoritmos de aprendizaje automático. Una representación está basada en el aprendizaje tradicional tal y como se ha abordado hasta ahora este problema y en la segunda representación se introduce una nueva forma

de interpretar la información especificando cada concepto por medio de múltiples instancias.

7.2.2.1. Representación del problema desde una perspectiva tradicional

Para resolver el problema con aprendizaje supervisado tradicional se emplea una representación que considera una única instancia por ejemplo que se dispone. Cada estudiante matriculado en un curso representará un ejemplo del conjunto de datos, donde se especificarán todas las actividades que el estudiante podría llevar a cabo, independientemente de que el estudiante las haya realizado o no o éstas estén disponibles en un curso concreto o no.

En este caso, cada estudiante puede realizar un número diferente de actividades: algunos estudiantes pueden haber trabajado duro y haber realizado todas las actividades disponibles en un curso, mientras que otros podrían no haber llegado a realizar ninguna de las actividades. Además, como se considera información de diferentes cursos, se puede encontrar diferencias sustanciales entre los mismos con respecto al número y tipo de actividades que ponen a disposición de los alumnos. De esta forma, nos podemos encontrar con cursos que tienen disponibles una gran variedad y número de actividades y con otros que el número y tipo de actividades es más reducido (información detallada se ha mostrado en la tabla 7.1).

Con esta representación, a pesar de la información variable de cada estudiante y curso, todas las instancias comparten la misma información lo que produce que la mayoría de los ejemplos tengan atributos vacíos, bien porque el estudiante no lo haya realizado o simplemente porque ese tipo de actividad no esté contemplada en dicho curso. Normalmente, un escenario con múltiples atributos vacíos tiene un efecto negativo en los algoritmos de aprendizaje que no funcionan de forma adecuada, reduciendo notablemente su rendimiento.

El principal problema es que la información no se muestra de acuerdo a las características del problema; por lo tanto la representación no se ajusta a la información disponible en cada ejemplo.

7.2.2.2. Representación del problema con múltiples instancias

Este problema puede verse como un problema multi-instancia donde cada patrón es compuesto de un estudiante matriculado en un curso. En este sentido, cada estudiante es contemplado como una bolsa que representa el trabajo llevado a cabo. Cada bolsa está compuesta de diferentes tipos de actividades realizadas por el estudiante. Esta representación se ajusta al problema perfectamente porque la información general de un estudiante, que es mantenida por todos los alumnos matriculados son considerados atributos de la bolsa o patrón, que tenemos la garantía que todos los alumnos disponen como mínimo de esa información. Con respecto a la información variable de cada uno de ellos, es almacenada en cada instancia, de este modo cada alumno tendrá un número de instancias en función del trabajo que ha desarrollado, no existiendo información vacía. Cada instancia es dividida en tres atributos: tipo de actividad, número de ejercicios en esa actividad y tiempo dedicado para completarla. Ocho tipos de actividades son consideradas, por tanto un número máximo de ocho instancias será la que puede tener cada alumno. Las actividades consideradas son: *ASSIGNMENT_S*, número de tareas que el estudiante a realizado, *ASSIGNMENT* se refiere al número de veces que el estudiante ha consultado la tarea pero finalmente no subió ningún archivo para su calificación. *QUIZ_P*, número de cuestionarios aprobados por el estudiante, *QUIZ_F* número de cuestionarios suspendidos por el estudiante, *QUIZ* se refiere a las veces que el estudiante ha visitado un cuestionario sin llegar a contestarlo con lo que no se dispone de calificación, *FORUM_POST* número de mensajes que el estudiante ha escrito ene l foro, *FORUM_READ* número de mensajes que el estudiante ha leído del foro *FORUM* se refiere al número de veces que el estudiante ha visto los foros que hay pero no ha llegado ni a escribir un mensaje ni a leer alguna de las conversaciones del foro.

Los atributos de la bolsa serían la identificación de estudiante, la identificación del curso y la nota final obtenida por el estudiante del curso. Un resumen de los atributos que pertenecen a una bolsa y la información que corresponde a las instancias se presentan en la figura 7.1.

Esta información puede ser representada de un modo muy natural en MIL. Se trata de una representación flexible donde nuevas actividades pueden ser añadidas sin afectar a los patrones que no consideran este nuevo tipo de actividades y solamente

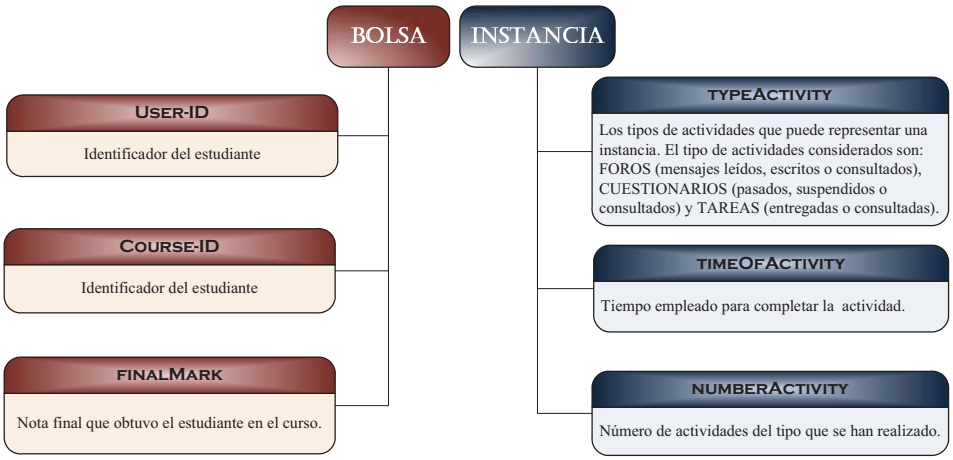


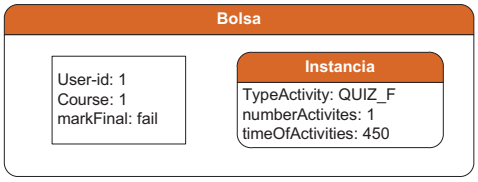
Figura 7.1: Información de las bolsas y las instancias

se indica la información disponible por cada estudiante, pudiendo ser diferente de une estudiante a otro. Esta información variables es almacenada como instancias, siendo el número de instancias que representa cada estudiante variable.

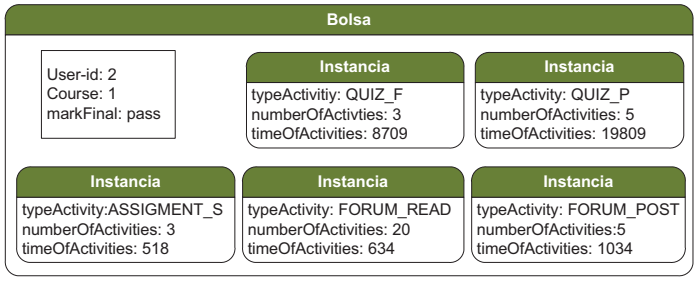
La figura 7.2 muestra la información disponible sobre dos estudiantes. La figura 7.2(a) muestra la información de acuerdo al aprendizaje supervisado tradicional; cada estudiante es un patrón que contiene toda la información considerada, aunque dicho estudiante no haya realizado ninguna actividad. Así, puede verse que el estudiante 1 tiene muchos campos vacíos de contenido. La figura 7.2(b) y 7.2(c) muestran la información de acuerdo a la representación MIL. La figura 7.2(b) muestra la información del estudiante 1 con representación MIL, este estudiante solamente realiza una actividad por tanto solamente dispondrá de la información general perteneciente a la bolsa (identificador del usuario, identificador del curso y calificación que obtuvo) y una instancia perteneciente al tipo de actividad que ha llevado a cabo. Se puede ver que esta representación solamente indica la información de las tareas realizadas por el estudiante sin incluir ningún campo vacío para las actividades no realizadas, ya que estas simplemente no son instancias del estudiante. La figura 7.2(c) muestra la información del estudiante 2, este estudiante ha llevado a cabo 5 tipos de actividades diferentes realizando a su vez diferente número de actividades de cada tipo, con lo que además de la información general, que es

Atributos	Estudiante1	Estudiante2
User_id	1	2
Course	1	1
n_assigment	0	3
total_time_assigment	0	8709
n_posts	0	5
n_read	0	20
total_time_forum	0	1034
n_quiz	1	8
n_quiz_a	0	5
n_quiz_s	1	3
total_time_quiz	450	19809
Final_mark	Fail	Pass

(a) Representación tradicional del estudiante 1 y estudiante 2



(b) Representación multi-instancia del estudiante 1



(c) Representación multi-instancia del estudiante 2

Figura 7.2: Information about two students

considerada por cada bolsa (identificador del usuario, identificador del curso y calificación que obtuvo), este estudiante tendrá cinco instancias de acuerdo al trabajo realizado.

7.2.3. Datos utilizados en la experimentación

Este estudio emplea los datos de los estudiantes del entorno virtual de aprendizaje implantado en la Universidad de Córdoba. Actualmente, la Universidad de Córdoba utiliza como herramienta e-learning la plataforma Moodle que cuenta con unos 580 cursos y unos 19000 estudiantes matriculados en alguno de ellos.

De todos los cursos disponibles, se hace una selección de acuerdo al número de actividades que ofertan, de modo que los cursos seleccionados oferten un número significativo de tareas que contengan al menos dos tipos de actividades de las consideradas en el estudio. Así, este estudio finalmente considera 7 cursos con un total de 419 estudiantes. Información acerca del número de estudiantes matriculado, el número de tareas, foros y cuestionario disponibles para cada curso de los considerados se muestra en la Tabla 7.2. Esta información se corresponde con los datos recogidos durante un año académico (de Septiembre a Junio) Toda la información sobre cada estudiante para ambas representaciones es exportada a un fichero de texto en formato ARFF [211].

Tabla 7.2: Información general de los cursos utilizados

Identificador del Curso	*Número de Estudiantes*	*Número de Tareas*	*Número de Foros*	*Número de Cuestionarios*
ICT-29	118	11	2	0
ICT-46	9	0	3	6
ICT-88	72	12	2	0
ICT-94	66	2	3	31
ICT-110	62	7	9	12
ICT-111	13	19	4	0
ICT-218	79	4	5	30

7.3. Comparación de la representación tradicional y multi-instancia para resolver el problema

En esta sección se llevará a cabo una comparación de las dos representaciones desarrolladas que nos permitan determina si la representación con múltiples instancias resulta ser más adecuada, permitiendo obtener mejoras significativas en la resolución del mismo.

Dos tipos de experimentos son llevados a cabo. El primer experimento emplea una representación con instancias simples y algunos de los algoritmos más representativos del aprendizaje supervisado tradicional son aplicados. El segundo experimento emplea una representación con múltiples instancias para resolver el problema y los paradigmas más representativos de este aprendizaje son aplicados. Finalmente, se realiza una comparación entre los resultados obtenidos por ambas representaciones.

Todos los experimentos son llevados a cabo usando validación cruzada con 10-fold [208] y los resultados medios de precisión, sensibilidad y especificidad son mostrados en las tablas de resultados que se muestran.

7.3.1. Resultados experimentales empleando la representación tradicional

Para evaluar la representación con instancias simples se ha considerado los paradigmas más representativos del aprendizaje supervisado tradicional. Estos métodos han logrado eficientes resultados en otros dominios de aplicación. Entre los paradigmas considerados, se han tenido en cuenta métodos basados en árboles, reglas, redes neuronales, máquinas de soporte vectorial y regresión logística. Un breve resumen de ellos se describen a continuación:

- Métodos basados en árboles: estos métodos han sido utilizados con éxito en muchas áreas. Se basa en generar árboles de decisión que clasifican instancias de acuerdo a los valores de sus características. Cada nodo en un árbol representa una característica de la instancia y cada rama representa un valor que

ese nodo puede tomar. Las instancias son clasificadas comenzando en el nodo raíz y seleccionando cada hijo en función de los valores de sus características, hasta finalmente llegar a un nodo hoja que determine la clasificación. Los algoritmos considerados en este paradigma son: ADTree[73], BFTree [174], DecisionStump [211], RandomForest [21], RandomTree [211], SimpleCart [22], C4.5 (J48) [155] and grafted C4.5 (J48graft) [203].

- Métodos basados en reglas: estos métodos generan reglas de decisión que son interpretables por el usuario. Las reglas de clasificación se componen de un antecedente y un consecuente. El consecuente representa la clase y el antecedente representa un número de condiciones unidas por conjunciones y disyunciones que evalúan el cubrimiento de un conjunto de instancias. Los algoritmos considerados en este paradigma son: NNge [127], Ridor [77], ZeroR [211] and OneR [100].

- Métodos basados en redes neuronales: las redes neuronales son consideradas una herramienta potente en temas de clasificación. La ventaja de las redes neuronales la podemos encontrar en que son métodos adaptativos que se pueden ajustar a los datos con independencia de su distribución. Además, son aproximadores universales que pueden aproximar casi cualquier función. El algoritmo considerado en este paradigma es: *RBFNetwork* [211].

- Métodos basados en Máquinas de Soporte Vectorial: las máquinas de soporte vectorial son una de las metodologías más populares para el diseño de sistemas de clasificación con fundamentos teóricos y una alta generalización. SVMs implementan el principio de minimización de riesgo estructural para construir clasificadores con un amplio margen. El algoritmo considerado en este paradigma es SMO [108] que implementa el algoritmo de optimización mínimo de John Platt's para entrenar clasificadores de máquinas de soporte vectorial.

- Métodos basados en regresión logística: la regresión logística (*Logistic Regression*, LR) es un modelo estadístico utilizado en clasificación probabilística. El algoritmo considerado en este paradigma es SimpleLogistic [186] que construye modelos lineales de regresión logística.

Tabla 7.3: Resultados experimentales de los métodos basados en el aprendizaje supervisado tradicional

Algoritmos Basados en Árboles			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
ADTree	0.7000	0.8042	0.5569
BFTree	0.6976	0.8368	0.5059
DecisionStump	0.6690	0.8889	0.3651
RandomForest	0.6667	0.7573	0.5426
RandomTree	0.6476	0.6996	0.5755
SimpleCart	**0.7071**	**0.7958**	**0.5867**
J48graft	0.6881	0.7950	0.5408
J48	0.6857	0.7950	0.5345
Algoritmos Basados en Reglas			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
PART	0.7023	0.8641	0.4786
NNge	0.6952	0.7329	0.6434
Ridor	0.6810	0.8648	0.4310
OneR	0.6476	0.7665	0.4835
ZeroR	0.5810	1.0000	0.0000
Algoritmos Basados en Naive Bayes			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
NB (Naive Bayes)	0.6857	0.8232	0.4944
NB Multinomial	0.6929	0.7662	0.5918
NB MultinomialUpdateable	0.6929	0.7662	0.5918
NB Simple	0.6810	0.8232	0.4832
NB Updateable	0.6857	0.8232	0.4944
Algoritmos Basados en Redes Neuronales			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
RBFNetwork	0.6929	0.8227	0.5114
Algoritmos Basados en Máquinas de Soporte Vectorial			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
SMO	0.6976	0.8842	0.4374
Algoritmos de Regresión Logística			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
SimpleLogistic	0.7048	0.8312	0.5296

Los resultados medios de exactitud, sensibilidad y especificidad son mostrados en la tabla 7.3. En general, se puede ver que los diferentes métodos optimizan los valores de sensibilidad a expensas de un decremento en los valores de especificidad que obtienen unos valores de clasificación más bajos. Esto demuestra que los modelos no clasifican correctamente los ejemplos negativos, en nuestro caso implica que tienen problemas en identificar a los estudiantes que finalmente no pasan el curso, los cuales son clasificados erróneamente como que si pasan dicho curso.

Observando los resultados, se puede apreciar que el algoritmo SimpleCart obtiene la exactitud más alta con un porcentaje del 70.71 %. Sin embargo, diferentes paradigmas usados en las experimentaciones produce resultados similares; así, los diferentes paradigmas contienen algún algoritmo con un resultado muy próximo a este porcentaje.

Este problema tiene casos especialmente difíciles de clasificar debido a que hay estudiantes muy trabajadores que hacen todas las actividades pero que finalmente no aprueban la asignatura y del mismo modo, también nos podemos encontrar los casos en el otro extremo, es decir estudiantes que no realizan ningún ejercicio pero que finalmente aprueban el curso (aunque estos casos nos son los habitual). Además, este problema tiene otra complejidad añadida, que consiste en que los cursos tienen diferente número y tipo de actividades con lo que todavía se dificulta más la resolución del mismo. Un estudiante puede que no haga determinado tipo de actividades porque no deseaba hacerlas o porque directamente ese curso no las tenía disponibles.

7.3.2. Resultados experimentales empleando la representación con múltiples instancias

Para evaluar los resultados con la representación del problema con múltiples instancias, de igual forma que se ha hecho con el estudio del problema con una representación tradicional, se han utilizado diferentes paradigmas para su resolución. Los principales paradigmas considerados incluye métodos basados en *diverse density*, regresión logística, máquinas de soporte vectorial, en distancias y *ensembles*, que considera las propuestas más relevantes que se han realizado en MIL.

- Métodos basados en DD: el algoritmo DD, propuesto por Maron y Lozano-Perez [126], es quizás una de los algoritmos más conocidos en MIL. Dado un conjunto de bolsas positivas y negativas, la idea detrás de este enfoque es aprender un concepto que esté cercano al menos a una instancia en cada bolsa positiva y lo suficientemente lejos de todas las instancias negativas. Así, el concepto debe describir regiones densas del espacio de instancia de las bolsas positivas y estar dispersa de la que describe cada bolsa negativa. Los algoritmos considerados en este paradigma son: MIDD [126], MIEMDD [233] y MDD [211].

- Métodos basados en regresión logística: la regresión logística es un método de aprendizaje muy popular en ML. El algoritmo considerado es MILR [215] que adapta la regresión logística estándar a MIL asumiendo un modelo de regresión logístico a nivel de instancia y usando sus probabilidad de clase para calcular la clase a nivel de bolsa.

- Métodos basados en Máquinas de Soporte Vectorial: las máquinas de soporte vectorial es un desarrollo reciente dentro de las comunidades de ML y DM. Se han realizado muchas extensiones de este modelo a MIL cuyos resultados han sido bastante satisfactorios. El algoritmo considerado en este paradigma es MISMO que usa el algoirtmo SMO [108] para el aprendizaje de SVM en conjunto con un núcleo MI [80].

- Métodos basados en distancia: el algoritmo de los k vecinos más cercanos en MIL fue introducido por Wang y Zucker [201]. La principal diferencia entre las propuestas para los métodos del vecino más cercano recae en la definición de las métricas de las distancias, que debe considera la distancia entre bolsas. Dos esquemas extensivamente utilizados son la distancia de Hausdorff mínima y la distancia de Kullback-Leibler. Los algoritmos considerados de este paradigma son: CitationKNN [201] y MIOptimalBall [7].

- Otros métodos propuestos: en este apartado consideraremos el uso de otra serie de algoritmos extendidos de diferente forma del aprendizaje supervisado tradicional para trabajar en MIL. Tres diferentes modelos se emplearán: MI-Wrapper [211] es un método que asigna la etiqueta de la clase de una bolsa a todas las instancias y entonces entrena un algoritmo de instancia simple

Tabla 7.4: Resultados experimentales de los métodos basados en MIL

Algoritmos basados en Aprendizaje Supervisado (Simple)			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
PART	0.7357	0.8387	0.5920
AdaBoostM1&PART	0.7262	0.8187	0.5992
Algoritmos basados en Aprendizaje Supervisado (Wrapper)			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
Bagging&PART	0.7167	0.7733	0.6361
AdaBoostM1&PART	0.7071	0.7735	0.6136
PART	0.7024	0.7857	0.5842
SMO	0.6810	0.8644	0.4270
NaiveBayes	0.6786	0.8515	0.4371
Algoritmos basados en Múltiples Distancia			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
MIOptimalBall	0.7071	0.7218	0.6877
CitationKNN	0.7000	0.7977	0.5631
Algoritmos basados en Clasificadores Débiles (Boosting)			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
DecisionStump	0.6762	0.7820	0.5277
RepTree	0.6595	0.7127	0.5866
Algoritmos basados en Regresión Logística)			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
MILR	0.6952	0.8183	0.5218
Algoritmos basados en Diverse Density			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
MIDD	0.6976	0.8552	0.4783
MIEMDD	0.6762	0.8549	0.4250
MDD	0.6571	0.7864	0.4757

esos datos. Diferentes sistemas de aprendizaje han sido utilizados, tales como Bagging, PART, SMO, AdaBoost y NaiveBayes; MISimple [211] reúne una serie de métodos que calculan un resumen estadístico para una bolsa y forman instancias simples a partir de ellas. PART y AdaBoost son utilizados como sistemas de aprendizaje y finalmente, MIBoost [217] es un algoritmo inspirado en AdaBoost que construye una serie de clasificadores débiles usando un sistema de aprendizaje tradicional y se basa en establecer de forma apropiada diferentes pesos a los datos de entrada, recibiendo todas las instancias las etiquetas de la bolsa.

Los resultados medios de exactitud, sensibilidad y especificidad son mostrados en la tabla 7.4. Si observamos los resultados de los distintos paradigmas, ocurre una situación similar a la comentada en el caso del aprendizaje tradicional donde se optimiza más la medida de sensibilidad a costa de un decremento del valor de especificidad. De nuevo, esto se traduce en una mala predicción de los estudiantes que no aprueban la asignatura. Se produce una mejor clasificación de los estudiantes que aprueban (mayor sensibilidad) a expensas de una peor clasificación de los que suspenden (menor especificidad).

El mejor algoritmo se trata de un sistema de clasificación basado en reglas, PART, utilizando la adaptación que hemos denominado *wrapper* que obtiene la exactitud más alta. No obstante resultados similares son obtenidos por otras técnicas combinadas con esta propuesta como son AdaBoost&PART y Bagging&PART.

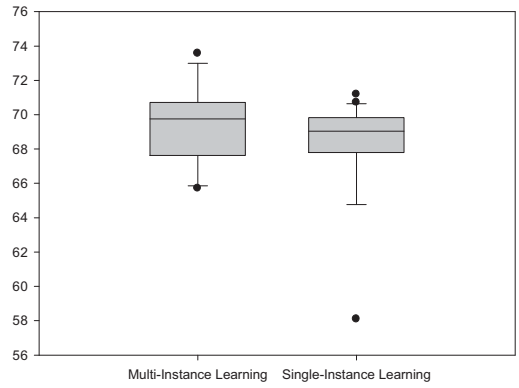
El principal problema de clasificación que nos encontramos en esta aplicación, como se ha comentado en el caso de aprendizaje supervisado, es que además de que cada estudiante realizar un número diferente de actividades en cada curso, los propios cursos tienen actividades diferentes en número y tipo con lo que es más costoso establecer relaciones generales entre ellas.

7.3.3. Comparación de la representación tradicional y con múltiples instancias

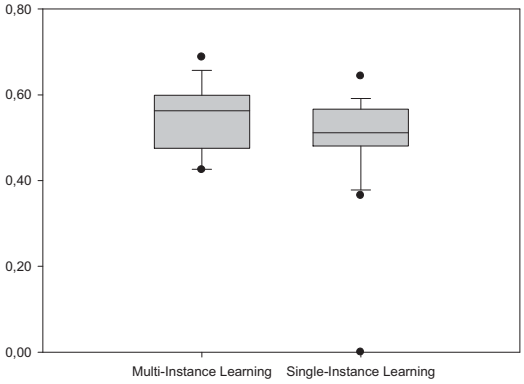
Para poder determinar si una representación resulta más adecuada que otra, se van a comparar los resultados obtenidos de las diferentes propuestas que se han aplicado, habiéndose considerado en ambas propuestas los paradigmas más relevantes.

Para ver cuales de estos métodos funciona mejor en general, se va a llevar a cabo un estudio sobre los valores de exactitud, sensibilidad y especificidad considerando los resultados de los algoritmos para cada tipo de representación. La figura 7.3 muestra un diagrama de cajas (box plot) con los resultados de las distintas medidas consideradas para los métodos que emplean la representación con múltiples instancias y la representación tradicional.

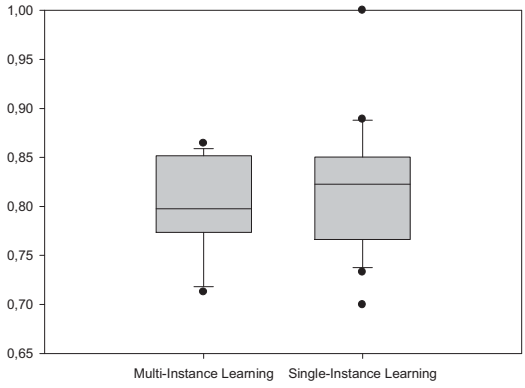
La figura 7.3(a) revela que los resultados obtenidos por los métodos con representación con múltiples instancias alcanzan mejores valores en los extremos, así como



(a) Exactitud



(b) Especificidad



(c) Sensibilidad

Figura 7.3: Comparativa de las representaciones tradicional y con múltiples instancias en el problema de educación (Box plot)

en los cuartiles medios, bajo y alto. Con respecto a la especificidad, figura 7.3(b), se puede apreciar que MIL obtiene mejores valores también en los extremos y en media aunque el primer y tercer cuartil son valores parecidos. Finalmente, si evaluamos la sensibilidad 7.3(c) se puede apreciar que en este caso los métodos que emplean una representación con instancias simples obtienen mejores resultados medios y en los extremos, aunque los valores del primer y tercer cuartil son ligeramente mejores en los métodos que emplean una representación con múltiples instancias.

Estos resultados ponen de manifiesto que las diferencias son menos acusadas en sensibilidad en comparación con las mejoras producidas en los valores de especificidad y exactitud. Lo que nos permite confirmar que la representación más adecuada toma lugar en un escenario con MIL, donde finalmente se obtienen los modelos más exactos.

Además, los resultados numéricos mostrados en las tablas 7.3 y 7.6 indican que los algoritmos que pertenecen al mismo paradigma pero en diferente modelo de aprendizaje consiguen mejores resultados en una representación con múltiples instancias. Por ejemplo, los algoritmos clásicos que siempre trabajan bien, como C4.5, PART y redes neuronales, alcanzan resultados peores a los que obtienen en un modelo de aprendizaje con múltiples instancias.

7.4. Aplicación de G3P-MI para la resolución del problema de predicción del rendimiento académico de un estudiante

Demostrada que la representación con múltiples instancias obtiene métodos más exactos en general que la representación tradicional, en esta sección se realizará un estudio de nuestro algoritmo, G3P-MI, para resolver el problema de predecir el rendimiento académico de un estudiante y compararlo con el resto de propuestas existentes.

$\sum N = \langle cond_I \rangle$, $\langle$variable-cat$\rangle$, $\langle$variable-num$\rangle$, $\langle$valores-cat$\rangle$, $\langle$valores-num$\rangle$, $\langle$cmp$\rangle$, $\langle$cmp-num$\rangle$, $\langle$cmp-int$\rangle$, $\langle$cmp-cat$\rangle$, $\langle$op-cat$\rangle$, $\langle$op-num$\rangle$, $\langle$op-int$\rangle$
$\sum T$ = GE, LT, EQ, NOT-EQ, IN, OUT, OR, AND
$\langle cond_I \rangle$:= $\langle$cmp$\rangle$
| **OR** $\langle$cmp$\rangle$ $\langle$antecedente$\rangle$
| **AND** $\langle$antecedente$\rangle$
$\langle$cmp$\rangle$:= $\langle$op-num$\rangle$ $\langle$cmp-num$\rangle$
| $\langle$op-cat$\rangle$ $\langle$cmp-cat$\rangle$
| $\langle$op-int$\rangle$ $\langle$cmp-int$\rangle$
$\langle$op-cat$\rangle$:= **EQ**
| **NOT EQ**
$\langle$op-num$\rangle$:= **GE**
| **LT**
$\langle$op-int$\rangle$:= **IN**
| **OUT**
$\langle$cmp-cat$\rangle$:= $\langle$variable-cat$\rangle\langle$value-cat$\rangle$
$\langle$cmp-int$\rangle$:= $\langle$variable-num$\rangle\langle$value-int$\rangle\langle$value-int$\rangle$
$\langle$cmp-num$\rangle$:= $\langle$variable-num$\rangle\langle$value-num$\rangle$
$\langle$variable-cat$\rangle$:= *Any valid attribute in dataset*
$\langle$variable-num$\rangle$:= *Any valid attribute in dataset*
$\langle$valores-cat$\rangle$:= *Any valid value*
$\langle$valores-num$\rangle$:= *Any valid value*

Figura 7.4: Gramática para representar el problema de predicción del rendimiento académico

7.4.1. Adaptación del modelo G3P-MI

En esta sección presentaremos algunos detalles de la adaptación del algoritmo G3P-MI al problema concreto que se desea abordar. En primer lugar comentaremos la gramática concreta que se utiliza en este problema. A continuación se comentarán los parámetros de configuración utilizados en la experimentación, para finalmente comentar los resultados obtenidos.

7.4.1.1. Gramática empleada

La gramática que utiliza G3P-MI se muestra en la figura 7.4. En este problema, se incorporan dos operadores numéricos con respecto a la gramática de la aplicación vista en el capítulo anterior, que son el operador IN y el operador OUT para

indicar intervalos de valores en el caso de atributos numéricos, el primero de ellos indica intervalos cerrados mientras que el segundo indica intervalos abiertos.

7.4.1.2. Función de evaluación

Para evaluar los clasificadores de G3P-MI se emplearán dos medidas, la sensibilidad (Se) y la especificidad (Es) [18; 189]. La sensibilidad, en este problema, nos evaluará el porcentaje de aciertos que se ha realizado en la clasificación con respecto a los estudiantes que pasan el curso. Por el contrario, con la especificidad evaluamos el funcionamiento del clasificador considerando el número de predicciones que se ha realizado correctamente para el caso de estudiantes que suspenden el curso. Se pueden definir como

$$sensibilidad = \frac{\#aprobados\ bien\ clasificados}{\#todos\ los\ aprobados} \tag{7.1}$$

$$especificidad = \frac{\#suspensos\ bien\ clasificados}{\#todos\ los\ suspensos} \tag{7.2}$$

Para obtener un clasificador con gran exactitud tendremos que conseguir predecir de manera correcta ambos tipos de estudiantes (aprobados y suspensos). Recordemos que se tratan de medidas contradictorias, si aumentamos al precisión en la clasificación de una de ellas, será a costa de reducir la otra medida. Por ejemplo, si optimizamos muy bien a los aprobados, teniendo por tanto, una sensibilidad muy elevada, normalmente irá asociado a una generalización de la regla que hará que se consideren también a alumnos suspensos como aprobados, lo que directamente implica una reducción de la especificidad y consecuentemente la clasificación correcta con respecto a la exactitud también podrá verse disminuida, en función de como se encuentren balanceado los datos.

Para generar la función de evaluación se combinan ambas medidas, se utilizará el producto de las mismas para penalizar los clasificadores que contengan cero en alguna de ellas, como además queremos que tengan la misma importancia en la clasificación, no habrá ningún tipo de ponderación. Quedando la función de evaluación definida como:

$$función\ de\ evaluación = Sensibilidad \cdot Especificidad$$

Un valor de 1 en la función, será representativo de que se ha obtenido una clasificación perfecta de todos los estudiantes.

7.4.1.3. Parámetros de configuración de G3P-MI

La Tabla 7.5 muestra los parámetros de configuración relativos al tamaño de la población, el número de generaciones, la probabilidades de los operadores de cruce y mutación y el método de selección de padres, que han sido empleados en los experimentos realizados con nuestro algoritmo G3P-MI.

Tabla 7.5: Parámetros de configuración del algoritmo G3P-MI

Algoritmo G3P-MI	
Parámetros	Valores
Tamaño de la población	1000
Número de generaciones	100
Probabilidad de cruce	95 %
Probabilidad de mutación	5 %
Porcentaje de elistismo	1 %
Método de selección de padres	*Ruleta*
Profundidad máxima del árbol	15

7.4.2. Comparación de G3P-MI con otros paradigmas aplicados al problema

En esta sección comprobaremos los resultados de nuestra propuesta con respecto a otras técnicas utilizadas en MIL. Los algoritmos que se considerarán en esta experimentación son: *Métodos basados en Diverse Density*: MIDD [126], MIEMDD [233] y MDD [215]; *Métodos basados en regresión logística*: MILR [158]; *Métodos basados en Máquinas de Soporte Vectorial*: MISMO que emplea el algoritmo SMO [152] para aprendizaje con SVM en conjunción con un núcleo MI [80]; *Enfoques*

basados en distancia: CitationKNN [201] y MIOptimalBall [7]; *Métodos basados en Aprendizaje Supervisado*: *MIWrapper* [71] usando diferentes sistemas de aprendizaje; *MISimple* [211]; *Boosting*: MIBoost, [217] y nuestra propuesta G3P-MI, que estaría clasificado dentro de métodos basados en algoritmo evolutivos.

Para la evaluación con este conjunto de datos, el paradigma evolutivo se ejecutó con 5 semillas diferentes y los valores medios de *exactitud*, *sensibilidad* y *especificidad* son mostrados. Todos los algoritmos utilizaron para la partición del conjunto de datos validación cruzada *10-fold* y Los resultados medios de exactitud, sensibilidad y especificidad se muestran en la tabla 7.6. Se puede ver que G3P-MI obtiene los modelos con resultados más exactos. Como principal ventaja que nos encontramos con este modelo es que logra un equilibrio entre las medidas contradictorias de sensibilidad y especificidad.

Si observamos los resultados de los distintos paradigmas, tal y como se ha comentado en la sección 7.3.2 se produce una optimización más exacta de la medida de sensibilidad a costa de un decremento del valor de especificidad, con los inconvenientes que este hecho acarrea con respecto a la clasificación de los estudiantes que no superan la asignatura y que ha sido comentado en las comparaciones que se han realizado anteriormente. No obstante, G3P-MI en este sentido balancea ambas medidas, obteniendo los valores más altos de sensibilidad sin apenas decrementar los de especificidad. Además, otra de las ventajas aportada por G3P-MI es la obtención de reglas interpretables que permiten obtener relaciones interesantes para determinar si ciertas actividades son influyentes para que el alumno apruebe o si es importante una implicación del alumno con respecto al tiempo dedicado en la plataforma o cualquier otra relación entre el trabajo realizado por el alumno y los resultados que finalmente se puede prever que obtiene.

Tabla 7.6: Resultados experimentales en la comparación de G3P-MI con otros métodos basados en MIL

Algoritmos basados en Aprendizaje Supervisado (Simple)			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
PART	0.7357	0.8387	0.5920
AdaBoostM1&PART	0.7262	0.8187	0.5992
Algoritmos basados en Aprendizaje Supervisado (Wrapper)			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
Bagging&PART	0.7167	0.7733	0.6361
AdaBoostM1&PART	0.7071	0.7735	0.6136
PART	0.7024	0.7857	0.5842
SMO	0.6810	0.8644	0.4270
NaiveBayes	0.6786	0.8515	0.4371
Algoritmos basados en Múltiples Distancia (Boosting)			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
MIOptimalBall (TypeI)	0.7071	0.7218	0.6877
CitationKNN	0.7000	0.7977	0.5631
Algoritmos basados en Clasificadores Débiles			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
DecisionStump	0.6762	0.7820	0.5277
RepTree	0.6595	0.7127	0.5866
Algoritmos basados en Regresión Logística)			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
MILR	0.6952	0.8183	0.5218
Algoritmos basados en Diverse Density			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
MIDD	0.6976	0.8552	0.4783
MIEMDD	0.6762	0.8549	0.4250
MDD	0.6571	0.7864	0.4757
Algoritmos basados en Algoritmos Evolutivos			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
G3P-MI	**0.7429**	**0,7020**	**0,7750**

7.4.3. Reglas obtenidas por G3P-MI

G3P-MI obtiene reglas del tipo SI-ENTONCES, de acuerdo a la gramática especificada, que son fáciles de comprender y aportan información acerca del dominio del problema, en este caso nos muestra el trabajo necesario para que el alumno supere el curso. Las reglas obtenidas son simple y contienen pocos elementos en la comparación. A continuación mostramos algunos ejemplos de reglas que se obtienen.

SI [$((NumberOfActivity \geq 3) \wedge (TypeOfActivity = \text{QUIZ\_P})) \vee$
$((NumberOfActivity \in [3, 8]) \wedge (TimeOfActivity \in [2554, 11602]))$
$\vee (NumberOfActivity \in [6, 8])$]
ENTONCES *El estudiante está apto en el curso.*
SINO *El estudiante no está apto en el curso.*

De acuerdo a esta regla para que un estudiante pase el curso debe realizar al menos tres cuestionario de manera correcta o realizar entre 3 y 8 actividades (bien foros, tareas o cuestionarios) dedicándoles un tiempo comprendido entre los 2554 y 11602 minutos o bien realizar más de 6 tareas de alguna de las actividades. La actividad más predominante son los cuestionarios, las demás requieren un mayor número de tareas entregadas y un tiempo mínimo de dedicación para ser significativa en el resultado final que se obtenga en la asignatura.

SI [$((TimeOfActivity \geq 2984) \vee (TypeOfActivity = \text{QUIZ\_P})) \wedge$
$((NumberOfActivity > 5)$]
ENTONCES *El estudiante está apto en el curso.*
SINO *El estudiante no está apto en el curso.*

Esta regla nos determina también como actividad más interesante los cuestionarios siendo necesario realizar más de 5 cuestionarios correctos o dedicar un tiempo en la plataforma consultando actividades y recursos de al menos 2984 minutos.

SI [$((NumberOfActivity > 4) \wedge (TypeOfActivity = \text{QUIZ\_P})) \vee$
$((TimeOfActivity \in [2736, 4886])]$
ENTONCES *El estudiante está apto en el curso.*
SINO *El estudiante no está apto en el curso.*

Otra de las reglas que se obtienen indica que el número de actividades realizada debe ser mayor de 4 y estás actividades debe tratarse de cuestionarios realizados correctamente o haber dedicado un tiempo a la plataforma realizando actividades comprendido entre 2736 y 4886 minutos.

SI [$((TimeOfActivity \in [754, 11813]) \wedge ((NumberOfActivity \geq 7)$
$\vee ((TypeOfActivity\ != \text{ASSIGMENTS}) \wedge (NumberOfActivity > 2)))$]
ENTONCES *El estudiante está apto en el curso.*
SINO *El estudiante no está apto en el curso.*

En esta regla se determina que para que el alumno pase el curso es necesario dedicar un tiempo a las actividades entre 754 y 11813 minutos y realizar al menos 7 actividades o bien realizar más de dos cuestionarios.

Las conclusiones que se pueden obtener es que en el proceso de pasar el curso se ven involucrados los tres factores principales que se han considerado, que son el tiempo que se dedica a las actividades, así como el número y tipo de actividades que se realizan. De las reglas anteriores e puede extraer que las actividades que aparecen como más relevantes son los cuestionarios, los cuales requieren un número menor de ellos realizados para poder pasar el curso y apareciendo en casi todas las reglas como actividad específica a realizar (al contrario que ocurre con las otras actividades, tareas y foros). Otro de los recursos interesantes del que se ha obtenido información es que los alumnos que han dedicado más de un tiempo determinado en la plataforma consultando materiales también ha resultado ser ventajoso para aprobar un curso. Finalmente, se ha podido concluir que dependiendo del tipo de actividad se ha requerido un mayor número de ellas desempeñadas para finalizar el curso con éxito.

Hay que tener en cuenta que este problema, es un problema con una alta complejidad debido a que no todos los cursos tienen todos los tipos de actividades considerados, ni la misma cantidad, lo que implica una mayor dificultad para establecer relaciones entre los estudiantes y las actividades más relevantes.

7.5. Aplicación del modelo MOG3P-MI

7.5.1. Adaptación del modelo MOG3P-MI

En esta sección aplicaremos la versión multi-objetivo de G3P-MI, MOG3P-MI para resolver el problema que se está abordando. La gramática que se utilizará es la misma que la mostrada en la figura 7.4, especificada cuando se empleó la versión mono-objetivo. Del mismo modo, la función de evaluación considerada es la sensibilidad y especificidad, como dos valores a optimizar simultáneamente, también han sido comentado en la sección 7.4.1.2 con lo que no se volverá a comentar aquí.

7.5.1.1. Parámetros de configuración

Los parámetros más relevantes del algoritmo con respecto al tamaño de la población, número de generaciones, probabilidad de cruce y mutación y método de selección, se muestran en la tabla 7.7.

Tabla 7.7: Parámetros de ejecución en Predicción del Rendimiento Académico

Algoritmo MOG3P-MI	
Parámetros	Valores
Tamaño de la población externa	100
Tamaño de la población	1000
Número de generaciones	100
Probabilidad de cruce	95 %
Probabilidad de mutación	60 %
Porcentaje de elistismo	5 %
Método de selección de padres	*Torneo Binario*
Profundidad máxima del árbol	50

7.5.2. Resultados experimentales

Los resultados experimentales de esta sección se llevarán a cabo desde diferentes perspectivas. En primer lugar, se analizará el frente de Pareto obtenido por nuestra

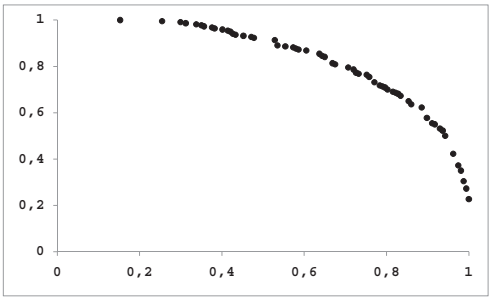


Figura 7.5: POF generado por MOG3P-MI para el problema de predicción del rendimiento académico

propuesta para cada usuario, no realizaremos comparaciones, pues en este caso es la única propuesta multi-objetivo que se ha aplicado al problema. En segundo lugar, se realizará una comparación de los resultados globales de exactitud, sensibilidad y especificidad considerando las diferentes técnicas que se han aplicado al problema hasta la fecha. Finalmente, evaluamos la calidad y comprensibilidad de los modelos de usuario obtenidos por nuestra propuesta.

7.5.2.1. Evaluación de los resultados obtenidos por MOG3P-MI

Nuestra propuesta aplica un enfoque multi-objetivo caracterizado por la generación de un POF. La optimalidad del pareto, como se ha estudiado, es un concepto clave en la optimización multi-objetivo. La Figura 7.5 muestra el POF obtenido para el conjunto de datos de entrenamiento con los que se ha trabajado. En esta figura podemos ver que las soluciones oscilan entre una optimización total de sensibilidad y especificidad, así como soluciones con un equilibrio entre ambos valores. Los clasificador con los resultados más balanceados de cada ejecución en el conjunto de entrenamiento será el clasificador considerado en el conjunto de test por nuestra metodología para comparar sus resultados con el resto de técnicas.

7.5.2.2. Comparación de los resultados

En esta sección comprobaremos los resultados de MOG3P-MI con respecto a otras técnicas utilizadas en MIL y la versión mono-objetivo de este algoritmo G3P-MI. Concretamente, los algoritmos que se considerarán en esta experimentación son:

Tabla 7.8: Resultados experimentales en la comparación de MOG3P-MI con otros métodos basados en MIL

Algoritmos basados en Aprendizaje Supervisado (Simple)			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
PART	0.7357	0.8387	0.5920
AdaBoostM1&PART	0.7262	0.8187	0.5992
Algoritmos basados en Aprendizaje Supervisado (Wrapper)			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
Bagging&PART	0.7167	0.7733	0.6361
AdaBoostM1&PART	0.7071	0.7735	0.6136
PART	0.7024	0.7857	0.5842
SMO	0.6810	0.8644	0.4270
NaiveBayes	0.6786	0.8515	0.4371
Algoritmos basados en Múltiples Distancia (Boosting)			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
MIOptimalBall (TypeI)	0.7071	0.7218	0.6877
CitationKNN	0.7000	0.7977	0.5631
Algoritmos basados en Clasificadores Débiles			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
DecisionStump	0.6762	0.7820	0.5277
RepTree	0.6595	0.7127	0.5866
Algoritmos basados en Regresión Logística)			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
MILR	0.6952	0.8183	0.5218
Algoritmos basados en Diverse Density			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
MIDD	0.6976	0.8552	0.4783
MIEMDD	0.6762	0.8549	0.4250
MDD	0.6571	0.7864	0.4757
Algoritmos basados en Algoritmos Evolutivos			
Algoritmo	***Exactitud***	***Sensibilidad***	***Especificidad***
G3P-MI	0.7429	0.7020	0,7750
MOG3P-MI	**0.7952**	**0.7209**	**0.8500**

Métodos basados en Diverse Density: MIDD [126], MIEMDD [233] y MDD [215]; *Métodos basados en regresión logística*: MILR [158]; *Métodos basados en Máquinas de Soporte Vectorial*: MISMO que emplea el algoritmo SMO [152] para aprendizaje con SVM en conjunción con un núcleo MI [80]; *Enfoques basados en distancia*: CitationKNN [201] y MIOptimalBall [7]; *Métodos basados en Aprendizaje Supervisado*: *MIWrapper* [71] usando diferentes sistemas de aprendizaje; *MISimple* [211]; *Boosting*: MIBoost, [217] y la propuesta G3P-MI mono-objetivo, que estaría clasificado dentro de métodos basados en algoritmo evolutivos, junto con esta versión multi-objetivo.

Para la evaluación con este conjunto de datos, los paradigmas evolutivos se han ejecutado con 5 semillas diferentes y los valores medios de *exactitud*, *sensibilidad* y *especificidad* son mostrados. Todos los algoritmos utilizaron para la partición del conjunto de datos validación cruzada *10-fold* y Los resultados medios de exactitud, sensibilidad y especificidad se muestran en la tabla 7.8.

Se puede ver que la mejor exactitud es obtenida por el modelo MOG3P-MI, quien además consigue un mejor equilibrio en los resultados de sensibilidad y especificidad, poniendo de manifiesto que la búsqueda del POF nos permite trabajar con el conjunto de soluciones óptimas y finalmente se obtienen mejores resultados en las tres medidas consideradas.

7.5.2.3. Reglas obtenidas por el modelo MOG3P-MI

Como ya se ha comentado anteriormente, una de las ventajas de nuestro sistema es que genera un clasificador formado por reglas SI-ENTONCES, que proporcionan un modelo que nos aporta conocimiento tras la clasificación mediante las reglas. Además, estas reglas se caracterizan por varias propiedades deseables: son simples, intuitivas, fáciles de comprender y proporcionan información representativa acerca del tipo, número y tiempo que el alumno debe dedicar al curso para poder pasarlo que puede ser utilizado por los docentes para conseguir dirigir a los alumnos hacia las actividades que resultan más eficientes para adquirir conocimiento. Mediante los siguientes ejemplos, se demuestra ejemplos obtenidos por estas reglas, a través de ellos un experto puede comprender aquellos atributos que son relevantes y el intervalo al que pertenecen.

SI [((*TimeOfActivity* $<$ 1508) $\wedge$ (*NumberOfActivity* $=$ 7)) $\vee$
(((*TypeOfActivity* $=$ QUIZ_P) $\wedge$ (*NumberOfActivity* $\geq$ 6)) $\vee$
(*TimeOfActivity* $\in$ [3007, 9448])]
ENTONCES *El estudiante está apto en el curso.*
SINO *El estudiante no está apto en el curso.*

De acuerdo a esta regla para que un estudiante pase el curso debe realizar al menos seis cuestionarios de manera correcta o realizar 7 actividades de algún tipo dedicando al menos 1508 minutos o bien dedicar entre 3007 y 9448 minutos a realizar diferente tipo de actividades en la plataforma o consultar los recursos disponibles.

SI [((*TimeOfActivity* $\geq$ 1769) $\wedge$ (*NumberOfActivity* $\geq$ 7)) $\vee$
((*TimeOfActivity geq* 3294) $\vee$ (*NumberOfActivity* $=$ 7) $\vee$
((*TypeOfActivity* $=$ QUIZ_P) $\wedge$ (*NumberOfActivity* $\geq$ 1) $\wedge$
(*TimeOfActivity* $\in$ [217, 1768]))]
ENTONCES *El estudiante está apto en el curso.*
SINO *El estudiante no está apto en el curso.*

Esta regla nos determina que para superar el curso se debe realizar al menos 1 cuestionario correcto empleando un tiempo entre los 217 y 1768 minutos o bien realizar 7 actividades de algún tipo o bien dedicar al menos 3294 minutos a realizar diferentes actividades y consulta de recursos.

SI [((*NumberOfActivity* $\in$ [3, 12]) $\wedge$ (*TypeOfActivity* $=$ QUIZ_P) $\wedge$
(*TimeOfActivity* $\in$ [516, 1727])) $\vee$ (*TimeOfActivity* $\in$ [2998, 7143]) $\vee$
((*TypeOfActivity* $=$ QUIZ_P) $\wedge$ (*NumberOfActivity* $\geq$ 7))]
ENTONCES *El estudiante está apto en el curso.*
SINO *El estudiante no está apto en el curso.*

Otra de las reglas que se obtienen indica que para pasar el curso se requiere al menos 3 cuestionarios correctos dedicando un tiempo de uso de la plataforma entre 516 y 1727 minutos o bien realizar más de 7 cuestionarios correctos sin un tiempo mínimo de dedicación o bien haber empleado en la realización de actividades y consulta de materiales un tiempo entre 2998 y 7143 minutos.

SI [$((NumberOfActivity \geq 8) \wedge (TypeOfActivity = \text{QUIZ\_P})) \vee$
$(TimeOfActivity \in [3009, 6503])$]
ENTONCES *El estudiante está apto en el curso.*
SINO *El estudiante no está apto en el curso.*

En esta regla, de nuevo se pone de manifiesto la importancia de desarrollar cuestionarios de forma correcta para confirmar los conocimientos del alumno, de este modo un estudiante que haya realizado al menos 8 cuestionarios correctos o dedicado un tiempo a la realización de actividades entre 3009 y 6503 minutos pasará el curso.

Comentarios Finales

Dedicamos esta sección a resumir brevemente los resultados alcanzados y a destacar las principales conclusiones obtenidas en este trabajo. Presentaremos las publicaciones asociadas al trabajo desarrollado y comentaremos algunos aspectos sobre trabajos futuros que siguen la línea aquí expuesta, y otras líneas de investigación que se puedan derivar de ella.

A Conclusiones

En esta memoria se han desarrollado dos modelos de aprendizaje multi-instancia basados en programación genética gramatical. El primero de ellos, denominado G3P-MI, representa el primer modelo de aprendizaje evolutivo que se ha aplicado a la resolución de este problema. El segundo, denominado MO G3P-MI, es una versión multi-objetivo del anterior, y obtiene soluciones en las que existe un equilibrio entre sensibilidad y especificidad.

Ambos algoritmos se han evaluado utilizando diez conjuntos de datos de prueba elaborados durante el desarrollo de esta memoria. Asimismo, se ha analizado el uso de ambos algoritmos en la resolución de dos aplicaciones reales: la recomendación de páginas web índice y la predicción del rendimiento académico de los estudiantes de acuerdo al trabajo realizado en las plataformas virtuales.

Los siguientes apartados resumen brevemente los resultados obtenidos y presentan las conclusiones más relevantes.

A.1 Algoritmo G3P-MI

Como ya se ha comentado, el primer objetivo de este trabajo ha sido el desarrollo de un modelo de programación genética gramatical para aprendizaje con multi-instancias. El modelo propuesto, denominado G3P-MI, representa el conocimiento por medio de reglas SI-ENTONCES cuyos antecedentes son codificados por el genotipo de cada individuo. El fenotipo se corresponde con la representación de la regla completa aplicada sobre las bolsas, considerando la hipótesis estándar para evaluar si dicha bolsa ha sido o no cubierta por la regla.

Para comprobar el buen funcionamiento de G3P-MI se ha realizado un estudio comparativo utilizando diez bases de datos de prueba sobre las que se han aplicado tanto nuestro algoritmo como otros que representan el estado del arte en lo que a MIL se refiere. Los resultados experimentales muestran que G3P-MI obtiene mejor rendimiento considerando las diferentes medidas de exactitud, sensibilidad y especificidad que todos los algoritmos incluidos en la comparación, aunque no en todos los casos existen diferentes significativas entre los resultados obtenidos.

Por último, indicar que una de las grandes ventajas de G3P-MI es el tipo de conocimiento descubierto. Como se ha comentado, se obtienen reglas simples (contienen muy pocos términos en el antecedente), fáciles de comprender y muy intuitivas, lo que garantiza su aplicabilidad a distintos problemas reales.

A.2 Algoritmo MOG3P-MI

El siguiente objetivo de esta memoria ha sido el desarrollo de un modelo multi-objetivo para aprendizaje multi-instancia, que produzca soluciones en las que haya un equilibrio entre exactitud y precisión. Dicho modelo estaría basado en el algoritmo G3P-MI y alguna de las propuestas más populares de MOEAs. Por esta razón, para decidir cuál era la propuesta que mejor se ajusta a nuestro propósito, hemos analizado el comportamiento de tres de los algoritmos evolutivos multi-objetivo más conocidos (NSGA2, SPEA2 y MOGLS) en la resolución de problemas de MIL.

El estudio desarrollado determina que no hay diferencias significativas entre los resultados obtenidos por los modelos multi-objetivo ensayados a nivel de exactitud y especificidad. Con respecto a sensibilidad, las versiones basadas en SPEA2 y NSGA2 presentan un mejor rendimiento que la versión MOGLS.

Además del estudio anterior, se ha realizado una comparación de los frentes de Pareto que producen los tres modelos propuestos. Utilizando tres diferentes métricas de comparación (hipervolumen, espaciado y cubrimiento) se determina que NSG3P-MI produce los frentes de Pareto con las mejores características de dispersión y cubrimiento de sus soluciones, además de obtener los mejores resultados con respecto a exactitud, sensibilidad y especificidad en la comparativa.

Finalmente, se ha realizado un estudio de rendimiento de los tres modelos multi-objetivo comparándolos con otros algoritmos de aprendizaje multi-instancia (incluido G3P-MI). Este estudio muestra que las técnicas multi-objetivo son las que obtienen un mejor rendimiento, obteniendo además el mejor equilibrio entre sensibilidad y especificidad, que finalmente conlleva en una mejor exactitud en la clasificación, donde son las técnicas que obtienen mejores resultados.

Además, de nuevo tenemos presente la interpretabilidad que introducen estos modelos, al generar reglas simples y fáciles de comprender con la información mostrada.

A.3 Problema de recomendación de páginas web índice

El primer problema real sobre el que se ha evaluado la calidad de los modelos desarrollados ha sido el de la recomendación de páginas web índice. Dicho problema consiste en predecir si una página web índice resultará o no interesante a un usuario dado en función de su contenido, sin ningún conocimiento sobre cuál de los enlaces contenidos en la página es el que resulta de interés para éste.

Para resolver dicho problema, cada página web índice se ha representado mediante una bolsa (multi-instancia), cuyas instancias son las páginas web a las que apunta dicha página índice. Se han desarrollado dos variantes de este sistema de representación: una booleana, en la que cada instancia representa la presencia o ausencia de un determinado término (palabra) en la página y una numérica, en la que la instancia contiene valores numéricos correspondientes a la frecuencia absoluta de aparición de dicho término en el documento.

Los modelos G3P-MI y MOG3P-MI se han aplicado sobre nueve conjuntos de datos y los resultados obtenidos se han comparado a los descritos en la bibliografía, obteniéndose los mejores resultados en cuanto a exactitud, precisión y alcance. Además de ésto, nuestros modelos presentan como ventaja más importante la generación de reglas que identifican las preferencias de los usuarios, a diferencias de las técnicas previas que actúan como cajas negras que no generan ningún tipo de conocimiento.

A.4 Problema de predicción del rendimiento académico de los estudiantes

El segundo problema real sobre el que se ha comprobado el buen comportamiento de los algoritmos de aprendizaje multi-instancia desarrollados es el de predicción del rendimiento académico de estudiantes. En este caso, proponemos el uso de una representación multi-instancia para obviar el problema de la información perdida que se presenta cuando utilizando una representación convencional.

Estudios experimentales entre ambas representaciones que consideran diferentes sistemas de clasificación demuestran que los métodos obtienen mejores resultados utilizando una representación con múltiples instancias, donde se alcanzan modelos más exactos.

Tras la valoración positiva de este tipo de representación, nuestro modelo, G3P-MI, es aplicado para su resolución comparándolo con el resto de técnicas de MIL. Los resultados experimentales nos permiten concluir que G3P-MI consigue un mejor equilibrio entre las medidas de sensibilidad y especificidad. Mientras que la mayoría de las otras técnicas optimizan los valores de sensibilidad a costa de obtener resultados con valores de especificidad muy decrementados, esto se traduce en que los estudiantes que no superan el curso no son clasificados de manera correcta. En este sentido, G3P-MI obtiene un equilibrio de ambas medidas, que finalmente le lleva a obtener valores más exactos que otras técnicas.

B Publicaciones Asociadas a la Tesis

A continuación, se muestran un listado de las publicaciones realizadas con este trabajo de investigación, clasificadas en congresos nacionales, internacionales, capítulos de libros y revistas.

B.1 Publicaciones en Congresos Nacionales

1. A. Zafra, S. Ventura, E. Herrera-Viedma. *Aprendizaje Multi-Instancia con Programación Genética para Web Mining*, in JAEM'07: I Jornadas de Algoritmos Evolutivos y Metaheurísticas, Zaragoza, SPAIN, 2007, pp. 285-292.

B.2 Publicaciones en Congresos Internacionales

1. A. Zafra, S. Ventura, C. Romero, and E. Herrera-Viedma, *Multiple instance learning with genetic programming for web mining*, in IWANN'07: The 9th International Work-Conference on Artificial Neural Networks, Lecture Notes in Computer Science, vol. 4507, 2007, pp. 919-927.

2. A. Zafra and S. Ventura, *Multi-objective genetic programming for multiple instance learning*, in EMCL'07: Proceedings of the 18th European Conference on Machine Learning, LNAI 4701, Warsaw, Poland, 2007, pp. 790-797.

3. A. Zafra, S. Ventura, *Modelling User Preferences with Multi-Instance Genetic Programming*, in IPMU'08: Proceedings of the 12th International Conference on Information Processing and Management of Uncertainly in Knowledge-Based Systems, Malaga, SPAIN, 2008, pp. 1113-1120.

4. A. Zafra, E. Gibaja and S. Ventura, *Multiple Instance Learning with Multi-Objective Genetic Programming for Web Mining.* in HIS'08: Proceedings of the 8th International Conference on Hybrid Intelligent Systems, Barcelona, SPAIN, 2008, pp. 513-518.

5. A. Zafra, S. Ventura, *A Comparison of Multi-Objective Grammar-Guided Genetic Programming methods to Multiple Instance Learning*, in HAIS'09: Proceedings of the 4th International Conference on Hybrid Artificial Intelligence Systems, LNAI 5572, Salamanca, SPAIN, 2009, pp. 450458.

6. A. Zafra, S. Ventura, *Predicting Student Grades in Learning Management Systems with Multiple Instance Genetic Programming*, in EDM'09: Proceedings of the 2sd International Conference on Educational Data Mining, Cordoba, SPAIN, 2009, pp. 309-314.

7. A. Zafra, C. Romero, S. Ventura, *Predicting Academic Achievement Using Multiple Instance Genetic Programming*, in ISDA'09: Proceedings of the 9th International Conference on Intelligent Systems Design and Applications, Pisa, ITALY, 2009. Aceptado.

B.3 Publicaciones en Capítulos de Libro

1. A. Zafra, S. Ventura, *Multiple Instance Learning with MultiObjective Genetic Programming*, in John Wang (Ed) Encyclopedia of Data Warehousing and Mining (ISBN: 978-1-60566-010-3), Information Science Reference, 2008, pp. 1372-1379.

2. A. Zafra, C. Romero, S. Ventura. *Multi-Instance Learning for students classifying in Learning Management Systems*, in C. Romero, S. Ventura, M. Pecheniskyi and R.S.d.J. Baker (Eds) Handbook Educational Data Mining.

B.4 Publicaciones en Revistas Internacionales

1. A. Zafra, S. Ventura, C. Romero, E. Herrera-Viedma, *Multi-Instance Genetic Programming for Web Index Recommendation*, Expert Systems with Applications, 36(9), 2009, pp. 11470–11479.

2. A. Zafra, Eva L. Gibaja, S. Ventura. *Multiple Instance Learning with Multi-Objective Genetic Programming for Web Mining*, Applied Soft Computing. Enviado.

3. A. Zafra, S. Ventura. *G3P-MI: A Genetic Programming Algorithm for Multiple Instance Learning*, Information Sciences. Enviado.

4. A. Zafra, S. Ventura. *A Comparison of Multi-Objective Genetic Programming methods to Multiple Instance Learning*, Pattern Recognition. Enviado

5. A. Zafra, S. Ventura. *Predicting Student Grades in Learning Management Systems with Multiple Instance Learning*, Computers & Education. Enviado.

C Trabajo Futuro

Tras el estudio realizado pensamos que todavía queda trabajo pendiente en esta línea, el cual se sale del ámbito de esta memoria. En nuestra opinión pensamos que existen fundamentalmente tres líneas de trabajo que presentamos a continuación:

C.1 Diseño de técnicas de selección de características para Aprendizaje Multi-instancia

Muchos de los problemas de aprendizaje multi-instancia con los que se ha trabajado contienen un número elevado de atributos, lo que supone un crecimiento del espacio de búsqueda de soluciones y, por tanto, dificulta la tarea de encontrar modelos adecuados. El preprocesado de la información y concretamente los métodos de selección de características han sido un tema ampliamente estudiado en el aprendizaje supervisado tradicional, existiendo una gran cantidad de propuestas al respecto [122; 16]. En el caso de MIL, este problema ha sido abordado en muy pocos trabajos, donde se diseñan algunos sistemas de selección de características incorporados al propio algoritmo de aprendizaje [229; 223; 160]. El inconveniente de estos métodos, denominados genéricamente *wrapper*, es que son costosos computacionalmente. Por esta razón, nuestra intención es desarrollar métodos de selección de características independientes del algoritmo de aprendizaje. Dichos métodos, denominados genéricamente métodos de filtrado, utilizan métricas de ganancia de información, distancia o consistencia, entre el atributo y la clase y son mucho más eficientes al no requerir que el algoritmo sea aplicado numerosas veces, variando en cada ejecución el número de atributos. Nuestro objetivo es comprobar si las

métricas empleadas tradicionalmente son válidas en un entorno multi-instancia y, en caso de que no lo sean, establecer qué métricas utilizar en esta tarea.

En concreto, el plan de trabajo a realizar consistiría en las siguientes actividades:

- Estudio de las principales técnicas de selección de características basadas en filtrado.
- Análisis de utilidad de las técnicas actuales sobre datos procedentes de un entorno multi-instancia.
- Diseño de nuevas métricas para el filtrado de características adaptadas al problema del aprendizaje multi-instancia.

C.2 Realización de clasificación con múltiples instancias y múltiples etiquetas

El problema del aprendizaje con múltiples etiquetas [76] es un problema relativamente reciente, que se plantea en respuesta a la necesidad de categorizar objetos que pertenecen simultáneamente a varias categorías. En este paradigma, el conjunto de etiquetas que definen el concepto que pretendemos aprender no se corresponde con conjuntos disjuntos de ejemplos, sino que pueden existir ejemplos pertenecientes simultáneamente a varias categorías. Por ejemplo, en categorización de imágenes [20], podemos asociar una fotografía con la etiqueta A si la persona A aparece en la imagen o B si es la persona B la que aparece, pero también podríamos categorizar la imagen como A y B si aparecen ambos personajes en la imagen. Existen otros muchos problemas que se han formulado desde esta perspectiva, tales como la categorización de documentos [121] o música [120] y diagnóstico médico [144].

En los últimos años han aparecido algunas propuestas en las que combinan las ideas de aprendizaje multi-etiqueta con el aprendizaje multi-instancia [228; 225]. La idea es, por una parte, aprovechar la potencia que tiene la representación mediante multi-instancias, que evita ambigüedades en el espacio de entrada (donde un objeto puede quedar representado mediante múltiples descripciones), y por otro lado, aprovechar la ambigüedad en el espacio de salida que confiere el aprendizaje con múltiples etiquetas, donde un objeto puede tener múltiples descripciones de salida.

Nuestro grupo de investigación ha desarrollado un modelo de programación genética para clasificación multi-etiqueta [8]. Este modelo, basado también en un esquema de codificación *individuo = regla de clasificación* se basa en ideas que son perfectamente asumibles por el modelo G3P-MI, por lo que consideramos que el desarrollo de una versión adaptada a clasificación con múltiples etiquetas es relativamente sencilla, y esperamos que produzca resultados de suficiente calidad en comparación con los publicados hasta la fecha.

C.3 Desarrollo de modelos de aprendizaje de reglas difusas para MIL

Las aplicaciones que se han estudiado de MIL están diseñadas como problemas de clasificación de dos clases. No obstante, es frecuente que las aplicaciones tengan varias clases de salida y, en algunos casos, que el conocimiento almacenado sea impreciso. En estos casos, un tratamiento difuso de las etiquetas ha demostrado una mayor flexibilidad en los sistemas [118; 177].

Dos aplicaciones consideradas en esta memoria, la recomendación de páginas web índice y la predicción del rendimiento académico de estudiantes, pueden ser modeladas directamente utilizando consideraciones de la lógica difusa. En la primera de ellas, sería interesante no hablar simplemente de interés o no interés, sino de introducir una medida de dicho interés (muy interesante, algo interesante, nada interesante), lo que permitiría obtener preferencias más precisas de los usuarios. En la segunda aplicación, también sería más interesante no considerar simplemente estudiantes aprobados o suspensos, sino considerar también una clasificación mayor con respecto a la nota obtenida por lo mismo, estableciendo diferentes grados (sobresaliente, notable, aprobado o suspenso).

Para el desarrollo de este tipo de modelos, habría que analizar si alguna de las hipótesis de trabajo alternativas (por ejemplo, la hipótesis generalizada) son más apropiadas para representar la relación existente entre el conocimiento almacenado en las bolsas y la variable lingüística que pretendemos modelar.

A

Aplicaciones

En este apéndice, describiremos las aplicaciones de prueba usadas en esta memoria. Nos centraremos en tres dominios de aplicaciones de MIL, la predicción de actividad de fármacos, la clasificación y categorización de imágenes y el problema del reto Este-Oeste. La primera de ellas es la aplicación más popular de MIL. La mayoría de los algoritmos desarrollados han sido aplicados sobre este problema, circunstancia por la que finalmente ha sido considerado como un benchmark que nos permite justificar los resultados de cualquier método nuevo con las demás técnicas existentes. La segunda, es una de las aplicaciones para la que se han propuestos más algoritmos, considerando diferente imágenes y formas de procesarlas y el tercer problema es considerado un problema que ha adquirido mucha popularidad en la programación lógica inductiva.

Todos los conjuntos de datos con los que se ha trabajado, junto con las particiones utilizadas están disponibles en *http://www.uco.es/grupos/ayrna/mil*.

A.1. Predicción de la actividad de fármacos

El problema de determinar la actividad de fármacos consiste en predecir si una determinada molécula presenta una cierta actividad investigada o no. Una molécula se caracteriza por presentar distintas *conformaciones espaciales*, producidas por la capacidad de rotación de sus enlaces. En la figura A.1 se muestran diferentes conformaciones de una molécula. Si una de sus conformaciones o configuraciones se puede enlazar a un centro activo de la actividad farmacológica que se está estudiando, la molécula presentará dicha actividad. En caso contrario, la molécula no presenta actividad alguna. Utilizando una representación con múltiples instancias, cada molécula representa un patrón y las diferentes conformaciones o fragmentos de dicha molécula es representado por medio de instancias de dicha molécula.

Aquí trabajaremos y comentaremos dos conjunto de datos, musk and mutagénesis:

1. El conjunto de datos *Musk* [57] está compuesto por dos subconjuntos *musk1* y *musk2*. El primero compuesto por 92 moléculas y el segundo por 102, ambos comparten algunas de las moléculas. Una molécula está formada por varias instancias, donde cada instancia representa una conformación de la molécula, que queda definida mediante 166 atributos numéricos. En este problema la propiedad que se investiga es el olor almizclado de un serie de sustancias aromáticas. Las características principales de este conjunto de datos se muestran en la tabla A.1.

2. El conjunto de datos *Mutagenesis* [182] está compuesto por tres subconjuntos diferentes en función de la representación utilizada, *mutagenesis-atoms* donde una bolsa contiene todos los átomos que componen la molécula, *mutagenesis-bonds* donde una bolsa contiene todos los átomos-enlaces que componen una molécula y *mutagenesis-chains* donde una bolsa contiene todos los enlaces

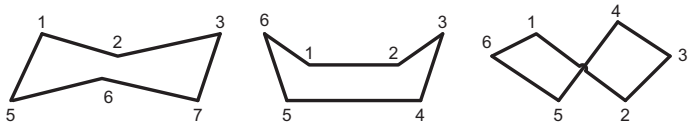


Figura A.1: Conformaciones de las moléculas

Tabla A.1: Conjunto de datos para la predicción de actividad de los fármacos

Conjunto de datos	Bolsas			Número Atributos	Número Instancias	Tamaño Medio de Bolsa
	Pos	*Neg*	*Total*			
Musk1	47	45	92	166	476	5.17
Musk2	39	63	102	166	6598	64.69
Mutagenesis-atoms	125	63	188	10	1618	8.61
Mutagenesis-bonds	125	63	188	16	3995	21.25
Mutagenesis-chains	125	63	188	24	5349	28.45

adyacentes que componen la molécula. Los tres conjuntos de datos están compuestos de 188 moléculas, 125 de las cuáles presentan la actividad y 63 no la presentan. Cada molécula queda definida por diferente número de atributos dependiendo de la representación utilizada. Las características principales de este conjunto de datos se muestran en la tabla A.1.

A.1.1. Clasificación y recuperación de imágenes basada en contenido

Este problema consiste en identificar un objeto en las imágenes. La principal dificultad de este problema recae en que una imagen puede contener múltiples objetos, lo que dificulta enormemente la tarea de clasificación y se trata de una aplicación compleja en el aprendizaje supervisado tradicional.

La clave para el procesamiento de imágenes basada en contenido es que solamente algunas partes de una imagen cuenta con el objeto que se describe. La representación de los ejemplos mediante múltiples instancias para este problema consiste en representar cada imagen es tratada como una bolsa y los diferentes regiones de la imagen son modeladas mediante instancias de esa bolsa. El modo de funcionamiento consiste en estudiar las distintas regiones de la imagen, determinando que la imagen contiene el objeto si al menos una de las regiones lo contiene y no lo contendría si en ningunas de sus regiones se identifica dicho objeto. Las regiones se identifican por un vector de características que muestran información sobre las propiedades del color y la textura de dicha región de la imagen, además de otra información como la posición que ocupa dicha región en la imagen. Existen distintas formas de

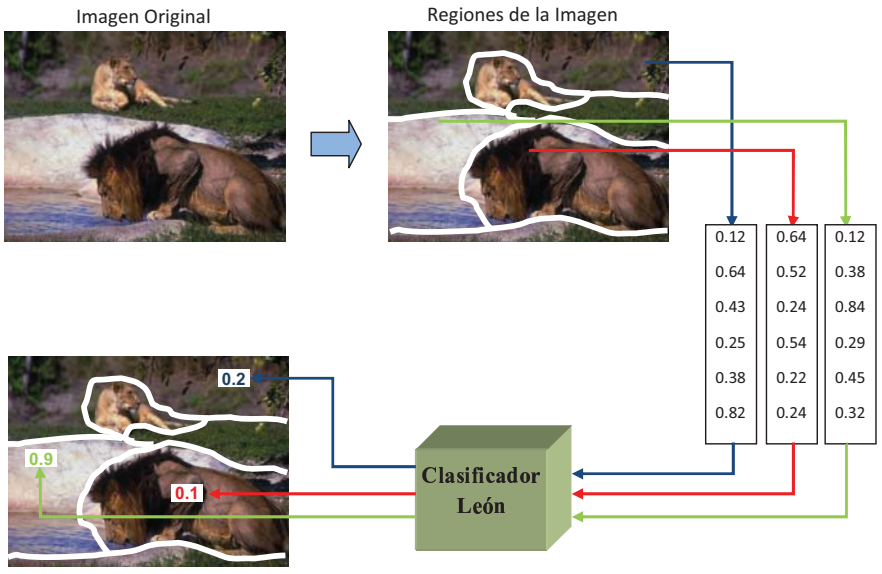


Figura A.2: Clasificación de las imágenes por regiones

fragmentar una imagen y dos métodos para medir los píxeles utilizados (bien mediante los valores de RGB o bien usando los valores de luminosidad y cromosidad). Un ejemplo de clasificación de las imágenes se puede ver en la figura A.2.

Los conjunto de datos que se emplean son *tiger*, *elephant* y *fox*. Todos ellos consisten en identificar si la imagen tiene el animal concreto del que se trata, en el primer caso tigres, en el segundo elefantes y el tercero zorros. Para ellos se cuenta con 200 imágenes clasificadas, de las que en cada caso 100 contienen el animal y otras 100 no lo contienen. La información de cada imagen se define por medio de 230 atributos que consideran la posición y características relevantes de la misma acerca de los píxeles utilizados. Las características principales de este conjunto de datos se muestran en la tabla A.2.

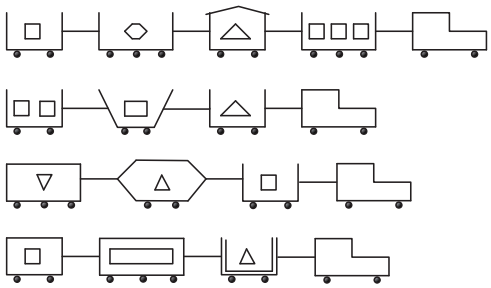
A.1.2. El reto Este-Oeste

El reto Este-Oeste [81] es un problema que tiene sus orígenes en la programación lógica inductiva. El problema consiste en predecir si un determinado tren va en dirección este o en dirección oeste. Para identificar el sentido de cada tren, cada

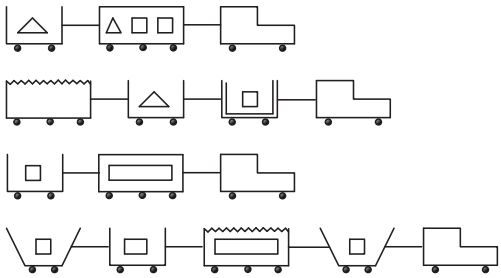
Tabla A.2: Conjunto de datos para la clasificación de imágenes por contenido

Conjunto de datos	Bolsas			Número Atributos	Número Instancias	Tamaño Medio de Bolsa
	Pos	*Neg*	*Total*			
Elephant	100	100	200	230	1391	6.96
Tiger	100	100	200	230	1220	6.10
Fox	100	100	200	230	1320	6.60

uno de ellos se caracteriza por diferentes vagones, cada uno de ellos con una forma y carga diferente. Un ejemplo de representaciones de los trenes puede verse en la figura A.3.



(a) Trenes con dirección este



(b) Trenes con dirección oeste

Figura A.3: Representación de los trenes

La representación de los ejemplos con múltiples instancias consiste en que cada tren es representado por una bolsa que contiene un número variable de vagones (instancias). Las instancias quedan identificadas por los atributos que indican las formas y las carga de cada vagón. Debido a que la hipótesis de Dietterich et al. que determina que una bolsa es positiva si va en la dirección este o en la dirección oeste es asimétrica, en los experimentos se considerarán dos conjuntos de datos. Uno es identificado como el problema Este-Oeste (*East-West*) para identificar como bolsa positiva los trenes que llevan una dirección hacia el este y el otro es Oeste-Este (*West-East*) para identificar como positivos los trenes que llevan una dirección hacia el oeste. Las características principales de este conjunto de datos se muestran en la tabla A.3.

Tabla A.3: Conjunto de datos para el problema de *Este-Oeste*

Conjunto	Bolsas			Número	Número	Tamaño
de datos	*Pos*	*Neg*	*Total*	Atributos	Instancias	Medio de Bolsa
EastWest	10	10	20	24	213	10.65
WestEast	10	10	20	24	213	10.65

B

Formatos

En esta sección mostraremos los formatos de representación de los ficheros que se han utilizado. La implementación que se ha realizado de nuestros modelos manejan dos tipos de formatos, uno de ellos es el formato establecido por WEKA para este problema y el otro es una adaptación que se ha realizado del formato KEEL para instancias simples, de forma que pueda soportar una representación con este aprendizaje.

B.1. Formato KEEL multi-instancia

El formato KEEL multi-instancia está basado en el formato KEEL utilizado en el aprendizaje con instancias simples [1]. Este formato está caracterizado por lo siguiente parámetros:

- Nombre del conjunto de datos *(@relation)*: es el primer campo que debe aparecer en el fichero, su formato es *@relation* <*nombre*>, donde *nombre* es una cadena representada entre comillas dobles.
- Atributos *(@attribute)*: los diferentes atributos son declaradas por diferentes secuencias ordenadas. Cada atributo en el conjunto de datos tiene su sentencia

@attribute que solamente puede define el nombre y tipo de datos de ese atributo. El orden en el que aparecen los atributos indica la posición de la columna que representa en el conjunto de datos. Por ejemplo, si un atributo es la tercera declaración, sus valores se corresponderán con los que se encuentren en la tercera columna, cada columna se encuentra separada por una coma. El formato para la definición de los atributos es:

@attribute <nombre> integer [mínimo,máximo]
@attribute <nombre> real [mínimo,máximo]
@attribute <nombre> <valor 1>, <valor 2>, $\cdots$, <valor N>

- Tipo de atributos*(@input, @output)*, se trata de un campo opcional y representa los atributos que actúan como entrada y los que actuán como salida. El formato sería:

  @inputs <nombre 1>, <nombre 2>, $\cdots$, <nombre N>
  @outputs <nombre 1>, <nombre 2>, $\cdots$, <nombre M>
  donde <nombre> es una cadena. La cadena debe ser igual al nombre especificado previamente en uno de los atributos.

- Datos *(@data)*, representa los datos de los ejemplos que se dispone. El orden de las columnas debe ser el mismo orden en el que los atributos se han definido. La declaración *(@data)* es una sentencia que indica el comiendo de los datos en el fichero. El formato es:

  @data
  x_{11}, x_{12}, $\cdots$, x_{1N}
  x_{21}, x_{22}, $\cdots$, x_{2N}
  $\cdots$, $\cdots$, $\cdots$, $\cdots$
  x_{M1}, x_{M2}, $\cdots$, x_{MN}
  La cadena <null> se utilizará para indicar los valores nulos de un campo.

En la figura B.1 se muestra un ejemplo de fichero con la representación KEEL.

Siguiendo con el formato establecido en KEEL, para utilizar dicho formato en una representación con múltiples instancias, se diferencia entre atributos de bolsa y atributos de instancia. De este modo, cada instancia está representada por los atributos que la caracterizan pero se añaden dos atributos más correspondientes a

```
@relation Iris_Plants_Database

@attribute sepalLength real [4.3, 7.9]
@attribute sepalWidth real [2.0, 4.4]
@attribute petalLength real [1.0, 6.9]
@attribute petalWidth real [0.1, 2.5]
@attribute class {Iris-setosa, Iris-versicolor, Iris-virginica}
@inputs sepalLength, sepalWidth, petalLength, petalWidth

@outputs class
@data
5.1, 3.5, 1.4, 0.2, Iris-setosa
4.9, 3.0, 1.4, 0.2, Iris-setosa
4.7, 3.2, 1.3, 0.2, Iris-setosa
4.6, 3.1, 1.5, 0.2, Iris-setosa
...
6.3, 2.5, 5.0, 1.9, Iris-virginica
6.5, 3.0, 5.2, 2.0, Iris-virginica
6.2, 3.4, 5.4, 2.3, Iris-virginica
5.9, 3.0, 5.1, 1.8, Iris-virginica
```

Figura B.1: Formato single-instancia de KEEL

información sobre la bolsa. Uno de estos atributos es el identificador de una bolsa, todas aquellas instancias que contengan el mismo identificador de bolsa indicará que son instancias de un mismo ejemplo. El otro atributo representa la clase del objeto, todas las instancias que pertenezcan a una misma bolsa tendrán la misma clase que la bolsa. El resto de parámetros de formato siguen la misma estructura que se ha comentado en el caso de instancias simples.

En la figura B.2 se muestra un ejemplo de fichero, que es un ejemplo que representa el primer ejemplo del conjunto de datos Musk. El nombre de la relación es precedido por la etiqueta *@relation*, la declaración de los atributos se introduce por la etiqueta *@attribute*. Hay dos atributos nominales que son *attribute molecule_name* y *class* que representan el nombre de la molécula y la etiqueta que representa la clase a la que pertenece respectivamente. Los datos se introducen por la etiqueta *@data*. Hay 166 atributos numéricos que describen las conformaciones de una molécula. Cada fila en la sección de datos es una instancia, introducida por la misma etiqueta

```
% This data was obtained from the UCI repository of machine learning datasets.
@relation musk1

@attribute molecule_name {MUSK-jf78,MUSK-jf67,MUSK-jf59,MUSK-jf58,MUSK-jf47,MUSK-
jf46,MUSK-jf17, …, NON-MUSK-232,NON-MUSK-226,NON-MUSK-220,NON-MUSK-208,NON-
MUSK-200,NON-MUSK-199}

@attribute f1 real [-9,130]
@attribute f2 real  [-199,98]
@attribute f3 real [-166,83]
…
@attribute f164 real [-132,24]
@attribute f165 real [-258,82]
@attribute f166 real [-71,235]
@attribute class {0,1}

@data
MUSK-jf58,50,-92,-24,29,-117,-86,61,-24,…,-90,-237,-160,70,-31,-58,7,93,-33,-9,86,1
MUSK-jf58,28,-118,-22,27,-117,54,-162,34,…,87,-244,-87,28,-32,14,24,61,-51,-31,-12,1
MUSK-jf58,58,-108,-20,19,-117,25,-164,72,…,-87,-244,-87,28,-32,14,24,61,-51,-31,-12,1
MUSK-jf58,24,-116,-29,32,-117,-94,56,15,…,-78,-234,-269,-208,-14,-51,1,80,-30,7,74,1
MUSK-jf58,23,-125,-30,31,-117,-91,62,-7,…,-84,-237,-221,-210,-16,-50,0,83,-30,6,77,1
….
NON-MUSK-319,53,-193,-146,137,-117,-39,48,-4,…,-242,-213,-107,4,64,205,-19,-194,62,0
NON-MUSK-319,59,-193,-146,32,-117,120,49,-2,…,-247,-213,-104,9,64,207,-18,-196,64,0
NON-MUSK-319,121,-189,-124,119,-117,-146,47,7,…,-232,-206,-103,32,66,209,-15,-192,81,0
NON-MUSK-319,112,-189,-125,17,-117,72,47,8,…,-223,-118,-105,33,67,207,-15,-190,82,0
```

Figura B.2: Formato multi-instancia de KEEL

del atributo *molecule_name*. De este modo cada fila en la sección de datos es una instancia que consiste de 168 valores de atributos separados por comas. el primer valor en cada fila es el nombre de la molécula. Como puede verse, las primeras cuatro instancias son conformaciones de la molécula *MUSK-188*. El último valor en cada fila es la etiqueta de la clase de la molécula, como *MUSK-188* es positiva, la clase de la etiqueta es codificada con un 1. El problema con esta representación es que cada fila es una instancia y el ejemplo completo requiere de varia filas, por lo que el proceso de evaluación deberá tenerlo en cuenta.

B.2. Formato WEKA multi-instancia

El formato WEKA multi-instancia está basado en el formato WEKA utilizado en el aprendizaje con instancias simples [1]. El formato de instancias simples está caracterizado por lo siguiente parámetros:

- Nombre del conjunto de datos *@relation*, el nombre del conjunto de datos se define en la primera línea. El formato es: @relation <nombre>, donde nombres es una cadena delimitada por comillas dobles.

- Declaración de atributos *@attribute*, la declaración de atributos tiene la forma de una especificación para cada uno de los atributos considerados. Cada atributo tiene su propia definición de nombre y tipo de datos. El orden en el que son especificados los atributos indica la columna en la que aparecen en los ejemplos. Por ejemplo, si un atributo está definido en tercer lugar, la tercera columna del conjunto de datos representaría los valores de este atributo. El formato sería:

  @attribute <nombre> <tipo de datos>

  donde nombre debe comenzar por un carácter del alfabeto, si hay espacios en el nombre, éste se debe representar entre comillas. El tipo de datos puede ser cualquiera de los tipos soportados por Weka: NUMERIC or REAL: que indica atributos con números reales, INTEGER: indica atributos con números enteros, DATE: para los atributos que representan fechas, el formato especificado es el aceptado en ISO-8601, que es ”yyyy-MM-dd'T'HH:mm:ss”, STRING: atributos que contienen cadenas de caracteres y ENUMERATE: atributos que pueden tomar un conjunto de posibles valores (categorías).

- Datos *@data*, esta sección contiene una línea por cada instancia en el conjunto de datos. La sentencia *@data* indica el comienzo del conjunto de datos. El formato es:

  @ data

  $x_{11}, x_{12}, \cdots, x_{1N}$

  $x_{21}, x_{22}, \cdots, x_{2N}$

  $\cdots, \cdots, \cdots, \cdots$

```
@relation Iris_Plants_Database

@attribute sepalLength real [4.3, 7.9]
@attribute sepalWidth real [2.0, 4.4]
@attribute petalLength real [1.0, 6.9]
@attribute petalWidth real [0.1, 2.5]
@attribute class {Iris-setosa, Iris-versicolor, Iris-virginica}
@inputs sepalLength, sepalWidth, petalLength, petalWidth

@outputs class
@data
5.1, 3.5, 1.4, 0.2, Iris-setosa
4.9, 3.0, 1.4, 0.2, Iris-setosa
4.7, 3.2, 1.3, 0.2, Iris-setosa
4.6, 3.1, 1.5, 0.2, Iris-setosa
...
6.3, 2.5, 5.0, 1.9, Iris-virginica
6.5, 3.0, 5.2, 2.0, Iris-virginica
6.2, 3.4, 5.4, 2.3, Iris-virginica
5.9, 3.0, 5.1, 1.8, Iris-virginica
```

Figura B.3: Formato single-instancia de WEKA

x_{M1}, x_{M2}, $\cdots$, x_{MN}
Los valores de los atributos de cada instancia son separados por comas y deben aparecer en el mismo orden en el que se especificaron. Los valores perdidos se representan por un símbolo de interrogación.

En la figura B.3 se muestra un ejemplo de fichero con la representación WEKA con instancias simples.

Siguiendo con el formato establecido en WEKA, se hace una adaptación para trabajar con una representación que emplea múltiples instancias. Para ello se realiza una adaptación para que cada fila se corresponda realmente con un ejemplo que contenga las distintas instancias. De este modo, se añade un nuevo atributo especificador que engloba la especificación de todos los atributos que pertenecen a una instancia. El delimitador empleado es *@attribute bag relational* indicando el inicio de la representación y *@end bag* indicando el final de la declaración de los atributos. Esta representación permite establecer otras relaciones anidadas, aunque para una

representación con MIL no nos son necesarias, ya que solamente se requiere una atributo para relacionar los datos que representas las instancias de las bolsas. Los otros atributos que se emplean fuera de estos atributos relacionales es el identificador de bolsa y la clase de la bolsa. En el caso de la representación de los ejemplos en el campo *@data*, todos los atributos que pertenecen a las instancias van entre comillas y una instancia de otra es separada por el operador
n.

En la figura B.4 se muestra un ejemplo de fichero, que es un ejemplo que representa el primer ejemplo del conjunto de datos Musk. En este caso, cada valor del atributo relacional consiste en una relación con 166 atributos numéricos que describen la forma de una conformación. También se añaden tres atributos a nivel de bolsa, el nombre de la molécula, el atributo de valor relacional *bag*, y la clase de la bolsa. Los atributos numéricos de la conformación de las moléculas son llamados atributos a nivel de instancia y se tienen que localizar entre el atributo relacional que se ha declarado.

También podemos ver en la figura B.4 la sección de datos representada por la etiqueta *@data*. Cada bolsa ahora se corresponde con una única fila en la sección de datos. Los valores del atributo relacional son puestos entre comillas, y, como ellos presentan bolsas de instancias completas, se separan por la secuencia
n. Cada instancia consiste de 166 valores numéricos y pueden haber un número arbitrario de ellas en cada bolsa. Por ejemplo, la molécula *MUSK-188* tiene un total de 4 instancias. La etiqueta de la clase de este bolsa es 1, que se especifica al final de la final que representa la instancia, indicando que es positiva en este caso.

```
% This data was obtained from the UCI repository of machine learning datasets.

@relation musk1

@attribute molecule_name {MUSK-jf78,MUSK-jf67,MUSK-jf59,MUSK-jf58,MUSK-jf47,MUSK-
jf46,MUSK-jf17, …, NON-MUSK-232,NON-MUSK-226,NON-MUSK-220,NON-MUSK-208,NON-
MUSK-200,NON-MUSK-199}

@attribute bag relational
@attribute f1 NUMERIC
@attribute f2 NUMERIC
@attribute f3 NUMERIC
…
@attribute f164 NUMERIC
@attribute f165 NUMERIC
@attribute f166 NUMERIC
@end bag

@attribute class {0,1}
@data

MUSK-jf58,"50.0,-92.0,-24.0,29.0,-117.0,…,22.0,50.0,-46.0,-15.0,-13.0,\n28.0,-118.0,-
22.0,27.0,-117.0,…,7.0,93.0,-33.0,-9.0,86.0,\n58.0,-108.0,-20.0,19.0,-117.0,…,24.0,61.0,-51.0,-
31.0,-12.0,\n24.0,-116.0,-29.0,32.0,-117.0,…,1.0,80.0,-30.0,7.0,74.0,\n23.0,-125.0,-30.0,31.0,-
117.0,…,0.0,83.0,-30.0,6.0,77.0,",1
….
NON-MUSK-319,"53.0,-193.0,-146.0,137.0,-117.0,…,64.0,205.0,-19.0,-194.0,62.0,\n59.0,-
193.0,-146.0,32.0,-117.0,…,64.0,207.0,-18.0,-196.0,64.0,\n121.0,-189.0,-124.0,119.0,-
117.0,…,66.0,209.0,-15.0,-192.0,81.0,\n112.0,-189.0,-125.0,17.0,-117.0,…,67.0,207.0,-15.0,-
190.0,82.0,",0
```

Figura B.4: Formato multi-instancia para WEKA

C

Análisis Estadísticos

En esta sección se describen los fundamentos para la comparación de heurísticas mediante técnicas estadísticas incluyendo los contrastes disponibles y las características que los hacen apropiados para las circunstancias más usuales. Se explica el uso de unos contrastes no paramétricos sencillos, seguros y robustos para la comparación estadística de metaheurísticas sobre un conjunto de instancias: el test de los rangos con signo de Wilcoxon para la comparación de dos heurísticas y, para la comparación de más de dos heurísticas, el test de Friedman de comparaciones múltiples con los correspondientes test a posteriori de Namenyi y de Bonferroni-Dunn para establecer las conclusiones.

C.1. Introducción

En los últimos años, la comunidad de Metaheurística ha alcanzado un claro convencimiento de la necesidad de aplicar técnicas estadísticas para validar los avances conseguidos. Esto es reflejo de cierta madurez del área, del crecimiento continuo de la capacidad de cómputo, del incremento de las aplicaciones reales y de la disponibilidad de cada vez más metaheurísticas. Las características del campo facilitan el desarrollo e implementación de nuevas metaheurísticas o la modificación de las

existentes, y la realización de experimentos para establecer comparaciones entre ellas.

En un artículo típico sobre metaheurísticas, se propone un nuevo algoritmo metaheurístico (frecuentemente un híbrido) o una versión de una metaheurística ya conocida; o se introduce una componente, o un nuevo paso (de pre-procesamiento, de post-optimización o intermedio); y se realiza, explícita o implícitamente, la hipótesis de que tal desarrollo mejora el rendimiento de los algoritmos previamente existentes. En otros artículos, se proponen diversas soluciones heurísticas alternativas para un problema y el propósito es analizar cuál resuelve el problema con éxito y cuál fracasa. Para la parte experimental del estudio se selecciona un conjunto de instancias, reales o generadas aleatoriamente, y sobre ellas se ejecutan los algoritmos y se mide el rendimiento. El trabajo suele incluir un conjunto de tablas con los indicadores del rendimiento alcanzados por los algoritmos participantes en el estudio y, con algún procedimiento de sentido común sobre la visualización de esos indicadores o con la aplicación de alguna técnica estadística, se determina si las diferencias observadas pueden atribuirse al azar o son evidencia suficiente de una diferencia real en el rendimiento de los algoritmos. A partir de este análisis se realizan conclusiones en el sentido del objetivo propuesto, que se suele concretar en que la propuesta realizada mejora significativamente las anteriores. En otros estudios se justifica la contribución del método propuesto con una clara mejora en otras propiedades menos cuantificables (simplicidad en el algoritmo o su implementación, menor número de parámetros a ajustar, inspiración en la naturaleza) sin un empeoramiento significativo en el indicador del rendimiento. Todas estas conclusiones deben estar sustentadas en contrastes estadísticos rigurosos aplicados con imparcialidad en lugar de en la mera observación de tablas numéricas de cierto tamaño.

Este tipo de situación, con sus propias particularidades, se viene presentando desde hace muchísimo tiempo en las investigaciones de laboratorio de las ciencias experimentales clásicas y en las pruebas de los diseños prácticos realizados desde la ingeniería. Por esto empezó a desarrollarse a principios del siglo XX la teoría de los contrastes de hipótesis estadísticos para realizar un planteamiento objetivo y fiable de las investigaciones.

En las siguientes secciones se analizan algunas cuestiones de la elección del indicador del rendimiento y del planteamiento general de los contrastes de hipótesis estadísticos para comparar heurísticas que se están utilizando.

C.2. Rendimiento de las metaheurísticas

La comparación entre las metaheurísticas debe hacerse en términos del mayor o menor cumplimiento de las propiedades deseables; aquellas que favorecen el interés práctico y teórico de las metaheurísticas [131]. Indican direcciones a las que dirigir los esfuerzos para contribuir al desarrollo del área, pero no será posible mejorar todas las propiedades a la vez.

Las propiedades cuantificables que pueden intervenir en la evaluación del rendimiento son la eficiencia, la eficacia, la efectividad, y tal vez la robustez. Estas propiedades están muy relacionadas y aunque frecuentemente no se establece una separación clara conviene distinguirlas y establecer precisamente qué miden. La eficiencia hace referencia a la cantidad de recursos empleados (espacio y, principalmente, tiempo) al actuar, la eficacia se relaciona con la probabilidad de alcanzar una solución óptima y la efectividad con la calidad de las soluciones propuestas. La robustez refleja la variabilidad de comportamiento al modificarse las características de las instancias sobre las que se ejecuta.

Las magnitudes asociadas a estas propiedades pueden ser fácilmente evaluadas en una ejecución de la metaheurística y permiten establecer una fórmula de computar un índice del rendimiento sobre una serie de casos para, en base a ellos, establecer comparaciones. Las cantidades usadas pueden ser el valor de la función objetivo alcanzada o, si se conoce el valor óptimo, el tiempo empleado en alcanzarlo, o si no se alcanza siempre, el número de veces que se alcanza en un tiempo razonable, o la razón con respecto al óptimo de la solución aportada. En el indicador del rendimiento adoptado se puede hacer intervenir una de ellas, o una combinación razonable de varias.

Para aplicar los contrastes estadísticos usuales en las ciencias experimentales sólo se requiere que los indicadores sean apropiados para comparar el rendimiento de

los algoritmos sobre cada instancia, en el sentido de que un mayor valor de dicho indicador es señal de un mejor rendimiento del algoritmo.

C.3. Contrastes de hipótesis

La teoría y la práctica del contraste de hipótesis estadístico surge a finales del siglo XIX gracias fundamentalmente a los trabajos de R.A. Fisher con el siguiente planteamiento general [67]. En la situación de partida de un estudio experimental se plantea una hipótesis nula, denotada H_0, que, como su nombre indica, debe representar la situación previamente establecida o aquella que implica que las propuestas contempladas en el estudio no suponen novedad o mejora significativa, frente a la hipótesis contraria o hipótesis alternativa, denotada H_1. Esta hipótesis alternativa puede ser tan general como el no cumplimiento completo de toda la hipótesis nula, pero, apoyada o no en argumentos teóricos en la dirección deseada, viene a representar de forma muy concreta, los planteamientos o conclusiones que se persigue sustentar en el estudio estadístico. En el contexto del estudio experimental del rendimiento de metaheurísticas, la hipótesis nula corresponde con la idea de que la nueva heurística que se propone no supone mejora con respecto a la solución estándar o las ya conocidas con las que se trata de comparar, o, en el caso de que se esté realizando un estudio para comparar entre sí varias propuestas heurísticas, corresponde con que las diferencias observadas en su comportamiento se pueden atribuir al efecto del azar. En el primero de estos casos, la hipótesis alternativa sería que la nueva propuesta heurística realizada supone una mejora, pudiéndose llegar a concretar una medida de dicha mejora, y en el segundo caso, que existe alguna diferencia, pudiéndose también concretar dónde (entre qué par de propuestas) radica exactamente la diferencia e incluso alguna medida de dicha diferencia.

Para contrastar estadísticamente una hipótesis nula H_0, frente a la hipótesis alternativa H_1, se construye un test de hipótesis para, a partir de los datos obtenidos, optar por rechazar o no rechazar la hipótesis nula, aceptando la alternativa. En la práctica los tests de hipótesis se diseñan utilizando una o más variables aleatorias cuyo comportamiento dependa de que sea cierta la hipótesis nula o la alternativa.

Estas variables se obtienen de alguna fórmula aplicada a los datos obtenidos directamente de la experimentación, como son la media, la desviación típica o fórmulas similares, que se denominan estadísticos. Estos estadísticos deben resumir los datos pero construidos de forma que se extraiga la información contenida en los datos que afecte a las hipótesis nula y alternativa, para lo que existe una parte de la estadística que se ocupa de ello. Una vez seleccionado el estadístico T en el que basar el test se determina un conjunto R de posibles valores del estadístico, llamado región crítica, para que, en el caso de que el valor del estadístico T esté en la región crítica se opte por rechazar la hipótesis nula en favor de la hipótesis alternativa, que es aceptada. En caso contrario, no se rechaza la hipótesis nula y por tanto no se acepta la alternativa. Nótese que por la diferente naturaleza de ambas hipótesis con respecto a los objetivos de la investigación, se habla de rechazar o no la hipótesis nula y de aceptar o no la hipótesis alternativa, evitando otras expresiones. Conociendo cual debe ser el comportamiento del estadístico T dependiendo de si es cierta la hipótesis nula o la alternativa, la región crítica se construye con aquellos valores que son muy verosímiles para el estadístico si es cierta la hipótesis alternativa y muy poco verosímiles si es cierta la hipótesis nula. Teniendo en cuenta el planteamiento original del contraste de hipótesis se pueden cometer básicamente dos errores: el denominado error de tipo I consistente en rechazar la hipótesis nula siendo cierta y el error de tipo II consistente en no rechazarla siendo cierta la hipótesis alternativa. Con un test de hipótesis basado en el estadístico T y la región crítica R, la probabilidad de cometer el error de tipo I es $Pr(T \in R/H_0)$ y la probabilidad de cometer el error de tipo II es $Pr(T \in R/H_1)$.

El nivel de confianza del test establece una cota para la probabilidad, $Pr(T \in R/H_0)$, de cometer el error de tipo I y la probabilidad, $Pr(T \in R/H_1)$, de no cometer el error de tipo II se denomina poder o potencia del test. El nivel de confianza se fija mediante unos porcentajes próximos al 100 % y se denota por $1 - \alpha$. Los valores usuales para α denominado nivel de significación, suelen ser de 0,01, 0,05 o 0,10 que expresados en porcentajes son el 1 %, el 5 % y el 10 % y corresponden, respectivamente a niveles de confianza del 99 %, del 95 % y del 90 %. El propósito ideal con el diseño del test mediante la elección de T y R es minimizar las dos probabilidades de error; sin embargo, dado que esto no es posible en general, se fija el nivel de significación α en un valor apropiado y se trata de que la potencia, denotada por β sea máxima.

En muchos casos, las particularidades del problema permiten garantizar que la región crítica apropiada consiste en los valores del estadístico que sobrepasan un valor determinado, el valor crítico. Por tanto, la construcción del test consiste en, dado el valor de α, obtener el valor crítico t_α, tal que $Pr(T > t_\alpha/H_0) = \alpha$. Una vez realizado el experimento, evaluando el estadístico T en los datos se obtiene el valor práctico t, entonces si $t > t_\alpha$ se rechaza la hipótesis nula porque $Pr(T > t/H_0) \leq \alpha$. Sin embargo, una información más precisa del nivel de confianza con el que se rechaza la hipótesis nula, nos la da el cálculo preciso de esta probabilidad, $Pr(T > t/H_0)$, que se denomina el p-valor de los datos.

En general, la hipótesis nula no comprende una única situación concreta sino un conjunto de ellas por lo que estas probabilidades calculadas en el supuesto de que sea cierta la hipótesis nula deben calcularse en aquella situación de las comprendidas dentro de la hipótesis nula más cercana a la alternativa, en la que esta probabilidad que pretendemos hacer tan pequeña como sea posible es máxima para garantizar que la probabilidad de cometer el error de tipo I nunca es superior al calculado.

Para realizar un análisis estadístico sobre el rendimiento, se supone que disponemos de los resultados de la ejecución de k algoritmos heurísticos para n instancias. Sea c_{ji} una medida del rendimiento del $j-ésimo$ algoritmo sobre la $i-ésima$ instancia. La tarea consiste en contrastar estadísticamente si, basado en los valores c_{ji}, se puede afirmar que el rendimiento de los algoritmos es significativamente diferente, y en caso de que el número de algoritmos sea mayor que dos, cuales son los algoritmos cuyo rendimiento es realmente diferente. De forma mucho más ambiciosa se trata incluso de proporcionar una medida o evaluación de esas diferencias. Sin embargo, estas afirmaciones no pueden realizarse de forma aislada sin indicadores de las garantías que aportan los datos de que sean ciertas o del riesgo de que alguna sea incorrecta.

Frente a los datos de evaluación del rendimiento de algoritmos en una serie de instancias se pueden plantear diversos contrastes de hipótesis para realizar comparaciones objetivas y fiables. Los contrastes paramétricos son más conocidos que los no paramétricos pero con la creciente divulgación de éstos [98; 180] la elección en cada circunstancia debe realizarse teniendo en cuenta, además de las características

propias de los contrastes, lo que realmente miden los estadísticos y las garantías de que sean ciertas las suposiciones que se asumen sobre los experimentos.

C.4. Comparación de dos algoritmos

Para empezar con la situación más sencilla, supongamos que se ejecutan dos algoritmos A y A' sobre un conjunto de n instancias y el rendimiento en la $i-ésima$ instancia es evaluado por los indicadores c_{1i} y c_{2i}. Frente al test de la t de student basado en la diferencia de los promedios, se proponen dos tests no paramétricos: el test de los rangos con signos de Wilcoxon [210] basado en la ordenación de las diferencias y el test de los signos que sólo tienen en cuenta el número de las diferencias positivas y las negativas. Aunque estos tests son más débiles (menos potentes) miden diferencias entre los algoritmos desde otro punto de vista.

C.4.1. El test t para casos emparejados

Una forma usual de contrastar si las diferencias entre dos variables (los indicadores del rendimiento de dos algoritmos) en una muestra de casos (las instancias) es atribuible al azar, es calcular el test t para datos emparejados, que chequea si la diferencia promedio es significativamente diferente de cero. La diferencia en el rendimiento entre los dos algoritmos en la i-ésima instancia es $d_i = c_{2i} - c_{1i}$. Si

$$\bar{d} = \frac{1}{n}\sum_{i=1}^{n} d_i \quad y \quad \sigma_{\bar{d}}^2 = \frac{1}{n-1}\sum_{i=1}^{n}(d_i - \bar{d})^2$$

el estadístico $T = \bar{d}/\sigma_{\bar{d}}$, se distribuye según la t de Student con $n-1$ grados de libertad. Para contrastar si existe diferencia entre el rendimiento de los dos algoritmos se realiza un contraste bilateral y, fijado el nivel α y encontrado en la tabla el valor crítico $t_{n,\alpha/2}$, se rechaza la hipótesis nula si $|T| > t_{n,\alpha/2}$. Para contrastar que, por ejemplo, el segundo algoritmo tiene mejor rendimiento que el primero se aplica un contraste unilateral y, a partir del valor crítico es $t_{n,\alpha}$, se rechaza la hipótesis nula si $T > t_{n,\alpha}$. Un primer inconveniente importante es que el test t requiere que las diferencias sobre las distintas instancias sean conmensurables; por tanto, es tan cuestionable como el cálculo de promedios entre instancias. Además,

a menos que el tamaño muestral sea suficientemente grande (30 instancias), se requiere también que la diferencia de las dos variables comparadas tenga distribución normal. No existe argumento suficiente a favor de ésto en nuestro contexto y los tests de normalidad existentes, como el de Kolmogorv-Smirnov, son poco potentes para muestras pequeñas; es decir son incapaces de encontrar anormalidades. Otro inconveniente es que el test t se ve gravemente afectado por los casos patológicos que sesgan el test quitándole potencia.

C.4.2. El Test de los rangos con signo de Wilcoxon

El test de los rangos con signos de Wilcoxon es la alternativa no paramétrica al test de la t de Student para casos emparejados. Para aplicar el test se calculan los valores absolutos de las diferencias en el rendimiento de dos algoritmos para cada instancia y se ordenan de mayor a menor y se comparan los lugares que ocupan las diferencias a favor de uno y otro algoritmo. Formalmente, sea $rang(d_i)$ la posición que ocupa $|d_i|$ en esta ordenación, denominado rango de d_i. Sea R^+ *y* R^-. las sumas respectivas de los rangos de las diferencias positivas y negativas. Los rangos de las diferencias nulas ($d_i = 0$) se reparten entre ambas sumas; si hay un número impar de ellas, se ignora una. Por tanto:

$$R^+ = \sum_{d_i>0} rang(d_i) + \frac{1}{2}\sum_{d_i=0} rang(d_i)$$

$$R^- = \sum_{d_i<0} rang(d_i) + \frac{1}{2}\sum_{d_i=0} rang(d_i)$$

Sea $T = min(R^+, R^-)$ la menor de las sumas. Si la hipótesis nula es cierta T debe aproximarse a $n(n+1)/4$ y cuanto más diferente sea el rendimiento de ambos algoritmos, menor se será T. Muchos textos de estadística incluyen valores críticos exactos para T para n hasta 25. Para un número mayor de instancias, el estadístico

$$z = \frac{T - n(n+1)/4}{\sqrt{n(n+1)(2n+1)/24}}$$

tiene distribución aproximadamente normal. Por tanto, para $\alpha = 0{,}05$, la hipótesis nula del contraste bilateral se puede rechazar si z es menor -1,96.

El test de los rangos con signo de Wilcoxon es más sensible que el test t, requiere conmensurabilidad de las diferencias, pero sólo cualitativamente: mayores diferencias cuentan más, lo que puede ser deseable, pero se ignoran las magnitudes absolutas. Desde el punto de vista estadístico es más seguro porque no se asume distribución normal. Los outliers tienen menos efecto que en el test de la t. El test de Wilcoxon contempla diferencias continuas, por tanto los índices de rendimiento no deben ser redondeados ya que ello disminuiría la potencia del test debido al aumento artificial del número de empates. Cuando se dan las condiciones del test de la t, el test de Wilcoxon es menos potente, pero cuando se incumplen puede ser más potente que el test t.

C.4.3. El test de los signos

Una forma sencilla y usual de comparar dos algoritmos es contar el número de veces en las que uno de ellos tiene mejor comportamiento que el otro. Bajo la hipótesis nula de que ambos tienen el mismo rendimiento, este número se aproximaría a la mitad de las veces, más concretamente, obedecería a una distribuciòn binomial con probabilidad de éxito 0.5 y número de pruebas n. Los valores críticos de esta distribución se suelen encontrar tabulados en cualquier libro de estadística. Cuando n es suficientemente grande, la distribución es aproximadamente normal con media $n/2$ y varianza $n/4$. Por tanto, al nivel α, la hipótesis nula se puede rechazar, aceptando que $A^{'}$ es una mejora de A si el número de veces que $A^{'}$ gana al algoritmo A es al menos $n/2 + z_\alpha\sqrt{n}/2$. Para $\alpha = 0{,}05$, es $z_\alpha = 1,96$, lo que ha dado origen a la regla rápida de $n/2 + \sqrt{n}$.

Este test no requiere ninguna conmensurabilidad entre las diferencias ni normalidad en la distribución y por tanto es aplicable a cualquier conjunto de instancias. Es más débil que el test de Wilcoxon. Algunos autores argumentan además que sólo se deben tener en cuenta las diferencias significativas, y el umbral de significación se fija por algún otro criterio; el contraste es similar y se puede aplicar un test del mismo tipo.

C.5. Comparaciones múltiples

Los tests de comparaciones por pares no han sido diseñados para obtener conclusiones sobre varias variables. Para un estudio experimental con 7 algoritmos, no parece razonable realizar las 21 comparaciones por pares posibles. Cuando se realiza un alto número de contrastes, aunque todas las hipótesis nulas sean ciertas y el nivel de confianza en cada uno de ellos sea razonable, la probabilidad de que alguna sea rechazada por efectos del azar crece, siendo relativamente fácil llegar a la conclusión de que existen diferencias, sin haberlas.

La realización simultánea de múltiples comparaciones es un tema importante en el contraste de hipótesis estadística [95; 171]. Aunque recientemente se tiende también a controlar la tasa de descubrimientos falsos (porcentaje de hipótesis nulas erróneamente rechazadas), el objetivo tradicional es controlar el error global, la probabilidad de cometer al menos un error de tipo I en todas las comparaciones. Existen procedimientos estadísticos especializados como el bien conocido Análisis de la Varianza (ANOVA) y la alternativa no parametrica; el test de Friedman. Este último, es quizás menos conocido e infrecuente, aunque más recomendable.

C.5.1. ANOVA

El método corriente para contrastar la existencia de diferencias entre varias variables en un mismo conjunto de casos es el ANOVA de bloques [67]. En este contexto, la hipótesis nula a contrastar es que todos los algoritmos rinden igual y que las diferencias observadas entre los rendimientos mostrados por los algoritmos son debidas al azar. El ANOVA divide la variabilidad total del indicador del rendimiento en tres sumandos: la variabilidad entre algoritmos (tratamientos en la terminología usual en ANOVA), la variabilidad entre instancias (bloques en la terminología ANOVA) y la variabilidad residual (o de error). Si la variabilidad entre algoritmos es significativamente mayor que la variabilidad residual, podemos rechazar la hipótesis nula y concluir que hay alguna diferencia entre los algoritmos. Este contraste se realiza con la distribución F de Fisher-Snedecor y en caso de rechazar la igualdad en el rendimiento de los algoritmos, se puede proceder con un contraste a posteriori para encontrar entre que pares de algoritmos se presentan esas diferencias. Entre los muchos tests para realizar esto en ANOVA, los dos más corrientes son el test

de Tuckey para comparar todos los algoritmos entre si y el test de Dunnett para compararlos con un algoritmo control (por ejemplo, una heurística básica y algunas mejoras propuestas, o comparar los algoritmos novedosos propuestos con métodos existentes). Ambos métodos usan la estimación de la desviación típica de la diferencia entre los rendimientos de dos algoritmos dividiendo la variabilidad residual por el número de instancias menos uno. Para comparar pares de algoritmos, las correspondientes diferencias en rendimiento se dividen por esta estimación y se comparan con el valor crítico.

Los tests usuales en ANOVA están basados en suposiciones que son previsiblemente violadas cuando se analiza el rendimiento de algoritmos heurísticos de optimización. Tales requerimientos del ANOVA se refieren a la normalidad de las distribuciones, la igualdad de varianzas y la independencia de las medidas. La violación de estas suposiciones perjudica aún más a los contrastes a posteriori.

C.5.2. Test de Friedman

El test de Friedman [75] es un test no paramétrico correspondiente al test F en un ANOVA de bloques. Se ordenan los k algoritmos separadamente para cada instancia según el rendimiento alcanzado, al mejor algoritmo tiene rango 1, el segundo mejor rango 2 y así sucesivamente. En caso de empate se asignan rangos intermedios. Sea r_j^i el rango del j-ésimo algoritmo sobre la i-ésima instancia. El test de Friedman compara los rangos medios de los algoritmos calculados por

$$R_j = \frac{1}{n} \sum_{i=1}^{n} r_j^i$$

Bajo la hipótesis nula, el rendimiento de todos los algoritmos es el mismo y los rangos R_j deben ser similares. El estadístico

$$\chi_F^2 = \frac{12n}{k(k+1)} \left[\sum_{j=1}^{k} R_j^2 - \frac{k(k+1)^2}{4} \right]$$

propuesto por Friedman, se distribuye de acuerdo a una χ^2 con $k-1$ grados de libertad, cuando n y k son suficientemente grandes (como regla experimental práctica,

$n > 10$ y $k > 5$). Para un número menor de algoritmos y casos se han calculado los valores críticos exactos [173]. Pero en [102] se propone un estadístico mejor, $F_F = \frac{(n-1)\chi_F^2}{n(k-1)-\chi_F^2}$ que se distribuye de acuerdo a la distribución F de Fisher-Snedecor con $k-1$ y $(k-1)(n-1)$ grados de libertad. La tabla con los valores críticos puede encontrase en cualquier libro de estadística. Como en las comparaciones por pares, el test no paramétrico de Friedman tiene teóricamente menos potencia que el test F, cuando las suposiciones de ANOVA se verifican, pero no ocurre así en caso contrario.

C.5.3. Contrastes a posteriori

Si se rechaza la hipótesis nula de equivalencia de los rendimientos de los algoritmos, se puede proceder a realizar contrastes a posteriori. El test de Nemenyi [143] es similar al de Tukey para ANOVA y se usa para comparar todos los algoritmos entre si. El rendimiento de dos algoritmos es significativamente diferente si los rangos promedios difieren, al menos en el valor crítico $CD = q_\alpha \sqrt{\frac{k(k+1)}{6n}}$ donde los valores críticos q_α son los del estadístico de rangos estudentizados dividido por $\sqrt{2}$.

Para comparar los algoritmos con un algoritmo de control se puede usar los métodos generales para controlar los errores comunes como la corrección de Bonferroni u otra similar que resulte más potente. El estadístico para comparar los rendimientos de los algoritmos i-ésimo y j-ésimo es: $Z = \frac{R_i - R_j}{\sqrt{\frac{k(k+1)}{6n}}}$

A partir del valor práctico z se encuentra el correspondiente p-valor usando la distribución normal, que es comparado con el nivel apropiado. Los tests se diferencian en la forma en que ajustan el valor de α para compensar las comparaciones múltiples.

El test de Bonferroni-Dunn [60] divide el valor de α por el número de comparaciones realizadas. El método alternativo para realizar los mismos contrastes es calcular la diferencia crítica usando la misma fórmula que para el test de Nemenyi, pero usando los valores críticos para $\alpha/(k-1)$. La comparación entre las tablas de los test de Nemenyi y de Dunn muestran que la potencia del test a posteriori es mucho mayor cuando todos los algoritmos se comparan sólo con uno de control y no entre ellos. Por tanto no se deben hacer todas las comparaciones por pares cuando sólo se trata de ver si un método novedoso propuesto es mejor que los existentes.

A diferencia del procedimiento de Bonferroni-Dunn que hace los contrastes de una vez, los procedimientos secuenciales hacia arriba (*step-up*) y hacia abajo (*step-down*) contrastan las hipótesis ordenadamente según el nivel. Denotaremos los p-valores ordenados por $p_1, p_2, \ldots.$ de forma que $p_1 \leq p_2 \ldots \leq p_{k-1}$. Dos métodos simples usuales son el método descendente de Holm [99] y el ascendente de Hochberg [94]. Ambos comparan cada p_i con $\alpha/(k-i)$, pero difieren en el orden de los tests. El procedimiento del test hacia abajo de Holm empieza con el p-valor más significativo. Si p_1 está por debajo de $\alpha/(k-1)$, se rechaza la correspondiente hipótesis nula y se procede a comparar p_2 con $\alpha/(k-2)$. Si se rechaza la segunda hipótesis se procede con la tercera, y así sucesivamente. Cuando una hipótesis nula no pueda ser rechazada, no se rechaza ninguna de las restantes. El procedimiento del test hacia arriba de Hochberg trabaja en la dirección contraria, comparando el mayor p-valor con α, el siguiente más largo con $\alpha/2$ y así hasta que encuentre una hipótesis nula que pueda rechazar. Entonces todas las hipótesis con p-valor menor se rechazan también.

Algunas veces el test de Friedman encuentra unas diferencias significativas pero los test a posteriori no llegan a detectarla. Esto se debe a la menor potencia de estos últimos. En este caso no se puede sacar ninguna otra conclusión de que el comportamiento de algunos algoritmos es diferente. En los experimentos, esto sólo ocurre en unos pocos casos entre miles.

El rendimiento del algoritmo A12 es significativamente mejor que el algoritmo $A(3,14-1,96 > 1,17)$ pero A2 no es significativamente mejor que el algoritmo $A(3,14-2,89 < 1,17)$, mientras que el algoritmo A1 está debajo de la diferencia crítica, aunque cerca de ella $(3,14-2,00 < 1,17)$. Se puede concluir que los experimentos muestran que la mejora"1"parece que ayuda mientras que no se detecta ninguna mejora significativa con la mejora "2".

Bibliografía

[1] Alcalá-Fernández, J., Sánchez, L., García, S., del Jesus, M. J., Ventura, S., Garrell, J. M., Otero, J., Romero, C., Bacardit, J., Rivas, V. M., Fernández, J. C., and Herrera, F. Keel: A software tool to assess evolutionary algorithms for data mining problems. *Soft Computing 13*, 3 (2009), 307–318.

[2] Alphonse, E., and Matwin, S. Filtering multi-instance problems to reduce dimensionality in relational learning. *Journal of Intelligent Information Systems 22*, 1 (2004), 23–40.

[3] Amar, R. A., Dooly, D. R., Goldman, S. A., and Zhang, Q. Multiple-instance learning of real-valued data. In *ICML'01: Proceedings of 18th International Conference on Machine Learning* (Williams College, Williamstown, MA, USA, 2001), Morgan Kaufmann, San Francisco, CA, pp. 3–10.

[4] Andrews, S., Hofmann, T., and Tsochantaridis, I. Multiple instance learning with generalized support vector machines. In *AAAI'02: Proceedings of the 18th National Conference on Artificial Intelligence* (Edmonton, Alta, 2002), pp. 943–944.

[5] Auer, P. On learning from multi-instance examples: Empirical evaluation of a theoretical approach. In *ICML'97: Proceedings of the Fourteenth International Conference on Machine Learning* (1997), pp. 21–29.

[6] Auer, P., Long, P. M., and Srinivasan, A. Approximating hyper-rectangles: learning and pseudo-random sets. In *STOC'97: Proceedings of the 29th annual ACM symposium on Theory of computing* (New York, NY, USA, 1997), ACM, pp. 314–323.

[7] Auer, P., and Ortner, R. A boosting approach to multiple instance learning. In *ECML'04: Proceedings of the 15th European Conference on Machine Learning* (Pisa, Italy, 2004), B. J.-F., E. F., P. D., and G. F., Eds., vol. 3201, pp. 63–74.

[8] Ávila, J. L., Gibaja, E. L., and Ventura, S. Multi-label classification with gene expression programming. In *Proceedings of the 4th International Conference on Hybrid Artificial Intelligence Systems* (Salamanca, SPAIN, 2009), pp. 629–637.

[9] Baker, J. E. Reducing bias and inefficiency in the selection algorithm. In *ICML'87: Proceedings of the 2nd International Conference on Genetic Algorithms* (1987), pp. 14–21.

[10] Banzhaf, W., Nordin, P., Keller, R. E., and Francone, F. D. *Genetic Programming – An Introduction; On the Automatic Evolution of Computer Programs and its Applications*. Morgan Kaufmann, San Francisco, CA, USA, Jan. 1998.

[11] Bentley, P. J. Evolutionary, my dear watson - investigating committee-based evolution of fuzzy rules for the detection of suspicious insurance claims. In *Proceedings of the Conference on Genetic and Evolutionary Computation* (2000), D. Whitley, D. Goldberg, L. S. E. Cantu-Paz, I. Parmee, and H.-G. Beyer, Eds., pp. 702–709.

[12] Berlanga, F. J., del Jesus, M. J., and Herrera, F. A novel genetic cooperative-competitive fuzzy rule based learning method using genetic programming for high dimensional problems. In *Proceedings of the 3rd International Workshop on Genetic and Evolving Fuzzy Systems* (Witten-Bommerholz, Germany, 2008), pp. 101–106.

[13] Bernstein, Y., Li, X., Ciesielski, V., and Song, A. Improving generalisation performance through multiobjective parsimony enforcement. In *GECCO'04: Proceedings of the Conference on Genetic and Evolutionary Computation* (June 2004), vol. 3103, pp. 702–703.

[14] Beume, N., Naujoks, B., and Emmerich, M. Sms-emoa: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research 181* (2007), 1653–1669.

[15] Beyer, H.-G., and Schwefel, H.-P. Evolution strategies: A comprehensive introduction. *Natural Computing 1*, 1 (2002), 3–52.

[16] Blum, A., and Langley, P. Selection of relevant features and examples in machine learning. *Artificial Intelligence 97*, 1 (1997), 245–271.

[17] Böhm, W., and Geyer-Schulz, A. Exact uniform initialization for genetic programming. In *FOGA'96: Proceedings of the 4th Workshop on Foundations of Genetic Algorithms* (San Diego, CA, USA, 1996), Morgan Kaufmann, pp. 379–407.

[18] Bojarczuk, C. C., Lopes, H. S., and Freitas, A. A. Genetic programming for knowledge discovery in chest-pain diagnosis. *IEEE Engineering in Medicine and Biology Magazine 19*, 4 (July-Aug. 2000), 38–44.

[19] Bojarczuk, C. C., Lopes, H. S., Freitas, A. A., and Michalkiewicz, E. L. A constrained-syntax genetic programming system for discovering classification rules: application to medical data sets. *Artificial Intelligence in Medicine 30*, 1 (2004), 27–48.

[20] Boutell, M. R., Luo, J., Shen, X., and Brown, C. M. Learning multi-label scene classification. *Pattern Recognition 37*, 9 (September 2004), 1757–1771.

[21] Breiman, L. Random forests. *Machine Learning 45*, 1 (2001), 5–32.

[22] Breiman, L., Friedman, J., Olshen, R., and Stone, C. *Classification and Regression Trees*. Wadsworth International Group, Belmont, California, 1984.

[23] Brindle, A. *Genetic Algorithms for Function Optimization*. PhD thesis, University of Alberta, 1991.

[24] Caballero, R., González, M., Guerrero, F., Molina, J., and Paralera, C. Solving a multiobjective location routing problem with a metaheuristic based on tabu search. application to a real case in andalusia. *European Journal of Operational Research 177* (2007), 1751–1763.

[25] Cabrera, J. A., Nadal, F., Muñoz, J. P., and Simon, A. Multiobjective constrained optimal synthesis of planar mechanisms using a new evolutionary algorithm. *Mechanism and Machine Theory 42*, 7 (2007), 791–806.

[26] Carreno, E., Leguizamon, G., and Wagner, N. Evolution of classification rules for comprehensible knowledge discovery. In *Proceedings of the IEEE Congress on Evolutionary Computation* (Singapore, 2007), D. Srinivasan and L. Wang, Eds., pp. 1261–1268.

[27] Cattral, R., Oppacher, F., and Deugo, D. Supervised and unsupervised data mining with an evolutionary algorithm. In *Proceedings of the IEEE Congress on Evolutionary Computation* (2001), vol. 2, pp. 767–774.

[28] Chai, Y.-M., and Yang, Z.-W. A multi-instance learning algorithm based on normalized radial basisfunction network. In *ISSN'07: Proceedings of the 4th International Symposium on Neural Networks* (Nanjing, China, 2007), vol. 4491 of *Lecture Notes in Computer Science*, pp. 1162–1172.

[29] Chapelle, O., Schölkopf, B., and Zien, A. *Semi-Supervised Learning*. MIT Press, 2006.

[30] Chellapilla, K. Evolving computer programs without subtree crossover. *IEEE Transactions on Evolutionary Computation 1*, 3 (Sept. 1997), 209–216.

[31] Chen, S.-C., Rubin, S., Shyu, M.-L., and Zhang, C. A dynamic user concept pattern learning framework for content-based image retrieval. *IEEE Transaction on System Man Cybernetics Part C Applications and Reviews 36*, 6 (2006), 772–783.

[32] Chen, X., Zhang, C., Chen, S. ., and Rubin, S. A human-centered multiple instance learning framework for semantic video retrieval. *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews 39*, 2 (2009), 228–233.

[33] Chen, Y., and Wang, J. Z. Image categorization by learning and reasoning with regions. *Journal of Machine Learning Research 5* (2004), 913–939.

[34] Chevaleyre, Y., and Zucker, J. D. Solving multiple-instance and multiple-part learning problems with decision trees and rule sets. application to the mutagenesis problem. In *AI'01: Proceedings of the 14th Biennial Conference of the Canadian Society on Computational Studies of Intelligence* (London, UK, 2001), E. Stroulia and S. Matwin, Eds., vol. 2056 of *Lecture Notes in Artificial Intelligence*, pp. 204–214.

[35] Chiang, J. Y., and Cheng, S. Multiple-instance content-based image retrieval employing isometric embedded similarity measure. *Pattern Recognition 42*, 1 (2009), 158–166.

[36] Chien, B.-C., Lin, J. Y., and Hong, T.-P. Learning discriminant functions with fuzzy attributes for classification using genetic programming. *Expert Systems with Applications 23*, 1 (2002), 31–37.

[37] Chou, S., and Liu, S. Learning effectiveness in web-based technology-mediated virtual learning environment. In *HICSS'05: Proceedings of the 38th Hawaii International Conference on System Sciences* (Washington, USA, 2005).

[38] Coello, C. A. A comprehensive survey of evolutionary-based multiobjective optimization techniques. *Knowledge and Information Systems 3* (August 1999), 269–308.

[39] Coello, C. A., Lamont, G. B., and Van Veldhuizen, D. A. *Evolutionary Algorithms for Solving Multi-Objective Problems. Genetic and Evolutionary Computation. Second Edition.* Springer-Verlag New York., 2007.

[40] Coello, C. A. C., Lamont, G. B., and Veldhuizen, D. A. V. *Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[41] Corne, D. W., Jerram, N. R., Knowles, J. D., and Oates, M. J. Pesa-ii: Region-based selection in evolutionary multiobjective optimization. In *GECCO'01: Proceedings of the Conference on Genetic and Evolutionary Computation Conference* (San Francisco, California, 2001), L. Spector, E. Goodman, A. Wu, W. B. Langdon, H.-M. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. H. Garzon, and E. Burke, Eds., pp. 283–290.

[42] Couchet, J., Manrique, D., Ríos, J., and Rodríguez-Patón, A. Crossover operators for grammar-guided genetic programming. *Soft Computing: A Fusion of Foundations, Methodologies and Applicactions 11*, 10 (2006), 943–955.

[43] Crow, J., and Kimura, M. *An Introduction to Population Genetics Theory*. New York. Harper and Row, 1970.

[44] Czyzak, P., and Jaszkiewicz, A. Pareto simulated annealing—a metaheuristic technique for multiple-objective combinatorial optimization. *Journal of Multi-Criteria Decision Analysis 7* (1998), 34–47.

[45] Davis, L. *The Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York., 1991.

[46] Davis, R. A., Charlton, A. J., Oehlschlager, S., and Wilson, J. C. Novel feature selection method for genetic programming using metabolomic 1H NMR data. *Chemometrics and Intelligent Laboratory Systems 81*, 1 (Mar. 2006), 50–59.

[47] de Falco, I., della Cioppa, A., and Tarantino, E. Discovering interesting classification rules with genetic programming. *Applied Soft Computing Journal 1*, 4 (2002), 257–269.

[48] de la Iglesia, B., Philpott, M. S., Bagnall, A. J., and Rayward-Smith, V. J. Data mining rules using multi-objective evolutionary algorithms. In *CEC'03: Congress on Evolutionary Computation* (December 2003), vol. 3, pp. 1552–1559.

[49] de Raedt, L. Attribute-value learning versus inductive logic programming: The missing links. In *ILP'98: Proceedings of the 8th International Conference on Inductive Logic Programming* (Wisconsin, USA, 1998), vol. 1446 of *Lecture Notes in Computer Science*, pp. 1–8.

[50] Deb, K. Evolutionary algorithms for multi-criterion optimization in engineering design. In *Evolutionary Algorithms in Engineering and Computer Science* (Chichester,UK, 1999), P. N. K. Miettinen, M. M. Mäkelä and J. Periaux, Eds., no. 8, John Wiley & Sons, pp. 135–161.

[51] Deb, K., Agrawal, S., Pratap, A., and Meyarivan, T. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimisation: NSGA-II. In *PPSN'00: Proceedings of the 6th International Conference on Parallel Problem Solving from Nature* (London, UK, 2000), Springer-Verlag, pp. 849–858.

[52] Deb, K., Mohan, M., and Mishra, S. A fast multi-objective evolutionary algorithm for finding well-spread pareto-optimal solutions. Technical report, Indian Institute of Technology, Kanpur, India, September 2003.

[53] Deb, K., and Reddy, A. Reliable classification of two-class cancer data using evolutionary algorithms. *Biosystems 72* (2003), 111–129.

[54] Dehuri, S., and Cho, S.-B. Multi-objective classification rule mining using gene expression programming. In *ICCIT '08: Proceedings of the 2008 Third International Conference on Convergence and Hybrid Information Technology* (Washington, DC, USA, 2008), IEEE Computer Society, pp. 754–760.

[55] Delgado, M., and Pegalajar, M. A multiobjective genetic algorithm for obtaining the optimal size of a recurrent neural network for grammatical inference. *Pattern Recognition 38* (2005), 1444–1456.

[56] Demšar, J. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research 7* (2006), 1–30.

[57] Dietterich, T. G., Lathrop, R. H., and Lozano-Perez, T. Solving the multiple instance problem with axis-parallel rectangles. *Artifical Intelligence 89*, 1 (1997), 31–71.

[58] Dooly, D., Zhang, Q., Goldman, S., and Amar, R. Multiple-instance learning of real-valued data. *Journal of Machine Learning Research 3*, 4-5 (2003), 651–678.

[59] Dundar, M. M., Fung, G., Krishnapuram, B., and Rao, R. B. Multiple-instance learning algorithms for computer-aided detection. *IEEE Transactions on biomedical engineering 55*, 3 (2008), 1015–1021.

[60] Dunn, O. J. Multiple comparisons among means. *Journal of the American Statistical Association 56*, 293 (1961), 52–64.

[61] Eggermont, J., Eiben, A. E., and van Hemert, J. I. Adapting the fitness function in gp for data mining. In *EuroGP'99: Second European Workshop on Genetic Programming* (Springer-Verlag, 1999), R. Poli, P. Nordin, W. B. Langdon, and T. C. Fogarty, Eds., vol. 1598 of *Lecture Notes in Computer Science*, pp. 193–202.

[62] Eggermont, J., Eiben, A. E., and van Hemert, J. I. A comparison of genetic programming variants for data classification. In *IDA'99: Third International Symposium on Advances in Intelligent Data Analysis* (Springer-Verlag, 1999), D. J. Hand, J. N. Kok, and M. R. Berthold, Eds., vol. 1642 of *Lecture Notes in Computer Science*, pp. 281–290.

[63] Elisseeff, A., and Weston, J. A kernel method for multi-labelled classification. In *Advances in Neural Information Processing Systems 14* (2002).

[64] Espejo, P. G., Romero, C., Ventura, S., and Hervás, C. Induction of classification rules with grammar-based genetic programming. In *ML'05: Proceedings of the 2nd International Conference on Machine Intelligence* (Tozeur, Tunisia, 2005), pp. 596–601.

[65] Espejo, P. G., Ventura, S., and Herrera, F. A survey on the application of genetic programming to classification. *IEEE Transactions on System, Man, and Cybernetics. Part C. Submitted*.

[66] Fausett, L., and Elwasif, W. Predicting performance from test scores using backpropagation and counterpropagation. In *WCCI'94: IEEE World Congress on Computational Intelligence* (Washington, USA, 1994), pp. 3398–3402.

[67] Fisher, R. *Statistical methods and scientific inference*. Hafner Publishing Co, 1959.

[68] Fonlupt, C. Solving the ocean color problem using a genetic programming approach. *Applied Soft Computing Journal 1*, 1 (June 2001), 63–72.

[69] Fonseca, C. M. *Multiobjective Genetic Algorithms with Applications to Control Engineering Problems*. PhD thesis, Department of Automatic Control and Systems Engineering, University of Sheffield, Sheffield, UK, September 1995.

[70] Fonseca, C. M., and Fleming, P. J. *An Overview of Evolutionary Algorithms in Multiobjective Optimization*. 1995.

[71] Frank, E., and Xu, X. Applying propositional learning algorithms to multi-instance data. Tech. rep., Department of Computer Science, University of Waikato, 2003.

[72] Freitas, A. Understanding the crucial differences between classification and discovery of association rules. *ACM SIGKDD Explorations 2*, 1 (2000), 65–69.

[73] Freund, Y., and Mason, L. The alternating decision tree learning algorithm. In *ICML'99: Proceeding of the 16th International Conference on Machine Learning* (1999), pp. 124–133.

[74] Freund, Y., and Schapire, R. E. Experiments with a new boosting algorithm. In *ICML'96: Proceedings of the 13th International Conference on Machine Learning* (Garda, Italy, 1996), pp. 148–156.

[75] Friedman, M. The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association 32*, 200 (1937), 675–701.

[76] G. Tsoumakas, I. Katakis, I. V. *Data Mining and Knowledge Discovery Handbook*. Springer, 2009, ch. Mining Multi-label Data.

[77] Gaines, B. R., C. P. Induction of ripple-down rules applied to modeling large databases. *Journal of Intelligence Information System 5*, 3 (1995), 211–228.

[78] García, S., González, F., and Sánchez, L. Evolving fuzzy rule based classifiers with ga-p: a grammatical approach. In *EuroGP'99: Second European Workshop in Genetic Programming* (1999), R. Poli, P. Nordin, W. B. Langdon, and T. C. Fogarty, Eds., vol. 1598 of *Lecture Notes in Computer Science*, pp. 203–210.

[79] García, S., and Herrera, F. An extension on "statistical comparisons of classifiers over multiple data sets"for all pairwise comparisons. *Journal of Machine Learning Research 9* (2008), 2677–2694.

[80] Gartner, T., Flach, P. A., Kowalczyk, A., and Smola, A. J. Multi-instance kernels. In *Proceedings of the 19th International Conference on Machine Learning* (2002), Morgan Kaufmann, pp. 179–186.

[81] Gartner, T., Lloyd, J. W., and Flach, P. A. Kernels and distances for structured data. *Machine Learning 57*, 3 (2004), 205–232.

[82] Geng, L., and Hamilton, H. J. Interestingness measures for data mining: A survey. *ACM Comput. Surv. 38*, 3 (2006), 9.

[83] Geyer-Schulz, A. *Fuzzy Rule-Based Expert Systems and Genetic Machine Learning*, first ed. Physica Verlag, Heidelberg, Germany, 1995.

[84] Gilbert, R. J., Rowland, J. J., and Kell, D. B. Genomic computing: explanatory modelling for functional genomics. In *Proceedings of the Genetic and Evolutionary Computation Conference* (2000), D. Whitley, D. Goldberg, E. Cantú-Paz, L. Spector, I. Parmee, and H.-G. Beyer, Eds., pp. 551–557.

[85] Goldberg, D. E. Zen and the art of genetic algorithms. In *Proceedings of the 3rd International Conference on Genetic Algorithms* (1989), pp. 80–85.

[86] Goldberg, D. E. *The design of innovation: Lessons from and for competent genetic algorithms*. Kluwer Academic Publishers, 2002.

[87] Goldberg, D. E., Sastry, K., and Lorá, X. Toward routine billion-variable optimization using genetic algorithms: Short communication. *Complexity 12*, 3 (2007), 27–29.

[88] Gu, Z., Mei, T., Tang, J., Wu, X., and Hua, X. MILC2: A multi-layer multi-instance learning approach to video concept detection. In *MMM'08: Proceedings of the 14th International Conference of Multimedia Modeling* (Kyoto, Japan, 2008), pp. 24–34.

[89] Hakanen, J., Miettinen, K., Mäkelä, M., and Manninen, J. On interactive multiobjective optimization with nimbus in chemical process design. *Journal of Multi-Criteria Decision Analysis 13* (2005), 125–134.

[90] Herlocker, J. L., Konstan, J. A., Borchers, A., and Riedl, J. An algorithmic framework for performing collaborative filtering. In *SIGIR'99: Proceedings of the 22nd Annual International ACM Conference on Research and Development in Information Retrieval* (Berkeley, California, United States, 1999), pp. 230–237.

[91] Herlocker, J. L., Konstan, J. A., Terveen, L. G., and Riedl, J. T. Evaluating collaborative filtering recommender systems. *ACM Transaction Information Systems 22*, 1 (2004), 5–53.

[92] Herman, G., Ye, G., Xu, J., and Zhang, B. Region-based image categorization with reduced feature set. In *IEEE 10th Workshop on Multimedia Signal Processing* (2008), pp. 586–591.

[93] Hernández, J., Ramírez, J., M., and Ferri, C. *Introducción a la Minería de Datos*. Pearson Prentice Hall, 2004.

[94] Hochberg, Y. A sharper bonferroni procedure for multiple tests of significance. *Biometrika 75*, 4 (1988), 800–802.

[95] Hochberg, Y., and Tamhane, A. C. *Multiple Comparison Procedures*. Wiley, 1987.

[96] Holland, J. H. Outline for a logical theory of adaptive systems. *Journal of the Association for Computing Machinery 3* (1962), 297314.

[97] Holland, J. H. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.

[98] Hollander, M., and Wolfe, D. *Nonparametric Statistical Methods*. Wiley, 1999.

[99] Holm, S. A simple sequentially rejective multiple test procedure. *Scand. J. Stat 6* (1979), 65–70.

[100] Holte, R. C. Very simple classification rules perform well on most commonly used datasets. *Machine Learning 11* (1993), 63–91.

[101] Howard, L. M., and D'Angelo, D. J. The GA-P: a genetic algorithm and genetic programming hybrid. *IEEE Expert 10*, 3 (1995), 11–15.

[102] Iman, R. L. Davenport, J. M. Approximations of the critical region of the friedman statistic. *Communications in Statistics 6* (1980), 571–595.

[103] Jaszkiewicz, A. Genetic local search for multiple combinatorial optimization. In *Working paper* (Piotrowo, 60-965Poznan, Poland, 1998).

[104] Jaszkiewicz, A., and Kominek, P. Genetic local search with distance preserving recombination operator for a vehicle routing problem. *European Journal of Operational Research 151*, 2 (2003), 352–364.

[105] Johnson, H. E., Gilbert, R. J., Winson, M. K., Goodacre, R., Smith, A. R., Rowland, J. J., Hall, M. A., and Kell, D. B. Explanatory analysis of the metabalone using genetic programming of simple, interpretable rules. *Genetic Programming and Evolvable Machines 1*, 3 (2000), 243–258.

[106] Jurado, F., and Valverde, M. Enhancing the electrical performance of a solid oxide fuel cell using multiobjective genetic algorithms. *Renewable Energy 30* (2005), 881–902.

[107] Karmakar, S., and Mujumdar, P. An inexact optimization approach for river waterquality management. *Journal of Environmental Management 81* (2006), 233–248.

[108] Keerthi, S., Shevade, S., Bhattacharyya, C., and K.R.K., M. Improvements to platt's smo algorithm for svm classifier design. *Neural Computation 13*, 3 (2001), 637–649.

[109] Kishore, J. K., Patnaik, L. M., Mani, V., and Agrawal, V. K. Application of genetic programming for multicategory pattern classification. *Evolutionary Computation, IEEE Transactions on 4*, 3 (September 2000), 242–258.

[110] Klosgen, W. Explora: a multipattern and multistrategy discovery assistant. 249–271.

[111] Knowles, J. D. ParEGO: A hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. Technical report, University of Manchester, UK, September 2004.

[112] Knowles, J. D., and Corne, D. W. Approximating the non-dominated front using the pareto archived evolution strategy. *Evolutionary Computation 8*, 2 (2000), 149–172.

[113] Knowles, J. D., and Corne, D. W. On Metrics for Comparing Nondominated Sets. In *Congress on Evolutionary Computation (CEC'2002)* (Piscataway, New Jersey, May 2002), vol. 1, IEEE Service Center, pp. 711–716.

[114] Knowles, J. D., Thiele, L., and Zitzler, E. A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers. 214, Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Switzerland, Feb. 2006. revised version.

[115] Knuth, D. E. Backus normal form vs. backus naur form. *Communications of the ACM 7*, 12 (1964), 735–736.

[116] Kotsiantis, S., and Pintelas, P. Predicting students marks in hellenic openuniversity. In *ICALT'05: The 5th International Conference on Advanced Learning Technologies* (Kaohsiung, Taiwan, 2005), pp. 664–668.

[117] Koza, J. R. *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* MIT Press, December 1992.

[118] Lee, K. H. *First Course On Fuzzy Theory And Applications.* SpringerVerlag, 2005.

[119] Lensberg, T., Eilifsen, A., and McKee, T. E. Bankruptcy theory development and classification via genetic programming. *European Journal of Operational Research 169*, 2 (1 Mar. 2006), 677–697.

[120] Li, T., and Ogihara, M. Detecting emotion in music. In *Proceedings of the 14th intern. conference on music information retrieval (ISMIR03)* (Baltimore, USA, 2003).

[121] Li, T., Zhang, C., and Zhu, S. Empirical studies on multi-label classification. In *ICTAI '06: Proceedings of the 18th IEEE International Conference on Tools with Artificial Intelligence* (Washington, DC, USA, 2006), IEEE Computer Society, pp. 86–92.

[122] Liu, H., and Motoda, H. *Feature Selection for Knowledge Discovery and Data Mining.* Kluwer Academic Publishers, Norwell, MA, USA, 1998.

[123] Long, P. M., and Tan, L. PAC learning axis-aligned rectangles with respect to product distributions from multiple-instance examples. *Machine Learning 30*, 1 (1998), 7–21.

[124] Luke, S., and Panait, L. A survey and comparison of tree generation algorithms. In *GECCO'01: Proceedings of the Genetic and Evolutionary Computation Conference* (San Francisco, California, 2001), Morgan Kaufmann, pp. 81–88.

[125] Mangasarian, O. L., and Wild, E. W. Multiple instance classification via successive linear programming. *Journal of Optimization Theory and Applications 137*, 3 (2008), 555–568.

[126] Maron, O., and Lozano-Pérez, T. A framework for multiple-instance learning. In *NIPS'97: Proceedings of Neural Information Processing System 10* (Denver, Colorado, USA, 1997), MIT Press, pp. 570–576.

[127] Martin, B. *Instance-Based learning: Nearest Neighbor With Generalization.* PhD thesis, Department of Computer Science. University of Waikato.

[128] Martínez, D. Predicting student outcomes using discriminant function analysis. In *Annual Meeting of the Research and Planning Group* (California, USA, 2001), pp. 163–173.

[129] Mayr, E. *Toward a New Philosophy of Biology: Observations o fan Evolutionist.* Belknap, 1988.

[130] M.D. Mulvenna, S.S. Anand, A. B. Personalization on the net using web mining. *Commum. ACM 43*, 8 (2000), 123–125.

[131] Melian Batista, B., Moreno Perez, J., and J.M., M. V. Metaheurísticas: una vision global. *Inteligencia Artificial 2*, 19 (2003), 7–28.

[132] Mendes, R. R. F., de B. Voznika, F., Nievola, J. C., and Freitas, A. A. Discovering fuzzy classification rules with genetic programming and co-evolution. In *PKDD'01: Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery* (London, UK, 2001), Springer-Verlag, pp. 314–325.

[133] Michalski, R. S. On the quasi-minimal solution of the general covering problem. In *FCIP'69: Proceedings of the 5th International Symposium on Information Processing* (Bled, Yugoslavia, 1969), vol. 3, pp. 125–128.

[134] Miettinen, K., Mäkelä, M., and Männikkö, T. Optimal control of continuous casting by nondifferentiable multiobjective optimization. *Computational Optimization and Applications 11* (1998), 177–194.

[135] Minaei-Bidgoli, B., and Punch, W. Using genetic algorithms for data mining optimization in an educational web-based system. *Genetic and Evolutionary Computation 2* (2003), 2252–2263.

[136] Mitchell, T. *Machine Learning.* McGraw-Hill Education (ISE Editions), October 1997.

[137] Mohanty, S. Multiobjective optimization of synthesis gas production using non-dominated sorting genetic algorithm. *Computers & Chemical Engineering 30* (2006), 1019–1025.

[138] Montana, D. J. Strongly typed genetic programming. *Evolutionary Computation 3*, 2 (1995), 199–230.

[139] Mooney, J. R., and Loriene, R. Content-based book recommending using learning for text categorization. In *Proceedings of the ACM International Conference on Digital Libraries* (2000), pp. 195–204.

[140] Mugambi, E. M., and Hunter, A. Multi-objective genetic programming optimization of decision trees for classifying medical data. In *KES'03: Knowledge-Based Intelligent Information and Engineering Systems* (2003), vol. 2773 of *Lecture Notes in Computer Science*, pp. 293–299.

[141] Muharram, M. Evolutionary constructive induction. *IEEE Transactions on Knowledge and Data Engineering 17*, 11 (2005), 1518–1528.

[142] Muni, D. P., Pal, N. R., and Das, J. A novel approach to design classifiers using genetic programming. *IEEE Transactions on Evolutionary Computation 8*, 2 (2004), 183–196.

[143] Nemenyi, P. B. *Distribution-free multiple comparisons*. PhD thesis, Princeton University, 1963.

[144] Ngan, P. S., Wong, M. L., Lam, W., Leung, K., and Cheng, J. Medical data mining using evolutionary computation. *Artificial Intelligence in Medicine 16*, 1 (1999), 73–96.

[145] Nilsson, N. J. *Introduction to Machine Learning*. Stanford University, 1996.

[146] Panait, L., and Luke, S. Alternative bloat control methods. In *GECCO'04: Proceedings of the 2004 Conference on Genetic and Evolutionary Computation* (2004), pp. 630–641.

[147] Pang, J., Huang, Q., and Jiang, S. Multiple instance boost using graph embedding based decision stump for pedestrian detection. In *ECCV'08: Proceedings of the 10th European Conference on Computer Vision* (Berlin, Heidelberg, 2008), vol. 5305 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 541–552.

[148] Pao, H. T., Chuang, S. C., Xu, Y. Y., and Fu, H. An EM based multiple instance learning method for image classification. *Expert Systems with Applications 35*, 3 (2008), 1468–1472.

[149] Parrott, D., Xiaodong, L., and Ciesielski, V. Multi-objective techniques in genetic programming for evolving classifiers. In *IEEE Congress on Evolutionary Computation* (September 2005), vol. 2, pp. 1141– 1148.

[150] Pazzani, M. J., and Billsus, D. Content-based recommendation systems. In *The Adaptive Web: Methods and Strategies of Web Personalization* (2007), P. Brusilovsky, A. Kobsa, and W. Nejdl, Eds., vol. 4321 of *Lecture Notes in Computer Science*, Springer, pp. 325–341.

[151] Pelikan, M. Hierarchical bayesian optimization algorithm: Toward a new generation of evolutionary algorithms. In *Studies in Computational Intelligence* (2005), vol. 170, Springer, pp. 27–29.

[152] Platt, J. C. Fast training of support vector machines using sequential minimal optimization. *Advances in Kernel Methods: Support Vector Learning* (1999), 185–208.

[153] Qi, X., and Han, Y. Incorporating multiple svms for automatic image annotation. *Pattern Recognition 40*, 2 (2007), 728–741.

[154] Qing-Shan, C., De-Fu, Z., Li-Jun, W., and Huo-Wang, C. A modified genetic programming for behavior scoring problem. In *Proceedings of the IEEE Symposium on Computational Intelligence and Data Mining* (Honolulu, Hawaii, USA, 2007), pp. 535–539.

[155] Quinlan, J. R. *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, San Mateo, CA, 1993.

[156] Ramon, J., and de Raedt, L. Multi instance neural networks. In *Proceedings of IMCL Workshop on Attribute Value and Relational Learning* (2000).

[157] Ratan, A., Maron, O., Grimson, W., and Lozano-Perez, T. Framework for learning query concepts in image classification. *Proc IEEE Comput Soc Conf Comput Vision Pattern Recognit 1* (1999), 423–429.

[158] Ray, S., and Craven, M. Supervised versus multiple instance learning: An empirical comparison. In *ICML 2005: 22nd International Conference on Machine Learning* (Bonn, 2005), R. L. and W. S., Eds., pp. 697–704.

[159] Ray, T. *An Approach to the Synthesis of Artificial Life II.* Addison-Wesley, 1991.

[160] Raykar, V. C., Krishnapuram, B., Bi, J., Dundar, M., and Rao, R. B. Bayesian multiple instance learning: Automatic feature selection and inductive transfer. In *ML'08: Proceedings of the 25 th International Conference on Machine Learning* (2008), pp. 808–815.

[161] Rice, W. H. *Moodle e-learning course development.* Pack Publishing, 2006.

[162] Ron, K. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *IJCAI'95: International Joint Conference on Artificial Intelligence* (Montreal, Canada, 1995), pp. 1137–1145.

[163] Ruffo, G. *Learning single and multiple instance decision tree for computer security applications.* PhD thesis, Department of Computer Science. University of Turin, Torino, Italy, 2000.

[164] Russell, S. *Rationality and Intelligence.* Common sense, reasoning, and rationality. Oxford University Press, 2002.

[165] Sakprasat, S., and Sinclair, M. C. Classification rule mining for automatic credit approval using genetic programming. In *Proceedings of the IEEE Congress on Evolutionary Computation* (2007), D. Srinivasan and L. Wang, Eds., pp. 548–555.

[166] Sanz, C., Madoz, M., Gorca, G., Zangara, A., Gonzalez, A., Ibañez, E., Ricci, G., Iglesias, L., and Martorelli, S. E-learning. In *WICC'06: Proceedings of 8th Workshop de Investigadores en Ciencias de la Computación* (2006).

[167] Schafer, J. B., Herlocker, D. F. J., and Sen, S. Collaborative filtering recommender systems. In *The Adaptive Web: Methods and Strategies of Web Personalization* (2007), P. Brusilovsky, A. Kobsa, and W. Nejdl, Eds., vol. 4321 of *Lecture Notes in Computer Science*, Springer, pp. 291–324.

[168] Schott, J. R. Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization. Master's thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts, May 1995.

[169] Scott, S. D., Zhang, J., and Brown, J. On generalized multiple-instance learning. *International Journal of Computational Intelligence and Applications 5*, 1 (March 2005), 21–35.

[170] Sebastiani, F. Machine learning in automated text categorization. *ACM Computing Surveys 34*, 1 (2002), 1–47.

[171] Shaser, J. P. Multiple hypothesis testing. *Annual Review of Psychology 46* (1995), 561–584.

[172] Shen, S., Sandham, W., Granat, M. H., Dempsey, M. F., and Patterson, J. A new approach to brain tumour diagnosis using fuzzy logic based genetic programming. In *Proceedings of the 25th Annual International Conference of the IEEE on Engineering in Medicine and Biology Society* (2003), pp. 870–873.

[173] Sheskin, D. J. *Handbook of parametric and nonparametric statistical procedures*. Chapman and Hall, 2000.

[174] Shi, H. *Best-first decision tree learnings*. PhD thesis, Department of Computer Science. University of Waikato, Hamilton, New Zealand, 2007.

[175] Shukla, P. K., and Deb, K. On finding multiple pareto-optimal solutions using classical and evolutionary generating methods. *European Journal of Operational Research 181*, 3 (2007), 1630–1652.

[176] Silberschatz, A., and Tuzhilin, A. On subjective measures of interestingness in knowledge discovery. In *KDD'95: Proceedings of the 1st International Conference on Knowledge Discovery and Data Mining* (Menlo Park, 1995).

[177] Siler, W., and Buckley, J. J. *Fuzzy Expert Systems and Fuzzy Reasoning*. John Wiley & Sons, 2005.

[178] Simon, H. A. Artificial intelligence: an empirical science. *Artificial Intelligence 77* (1995), 95–127.

[179] Song, A., Ciesielski, V., and Williams, H. Texture classifiers generated by genetic programming. In *CEC'02: Proceedings of Congress on Evolutionary Computation* (2002), pp. 243–248.

[180] Sprent, P. Smeeton, N. *Applied Nonparametric Statistical Methods*. Chapman and Hall, 2001.

[181] Srinivas, N., and Deb, K. Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. *Evolutionary Computation 2*, 3 (1994), 221–248.

[182] Srinivasan, A., Muggleton, S. H., King, R. D., and Sternberg, M. J. E. Mutagenesis: ILP experiments in a non-determinate biological domain. In *ILP'94: Proceedings of the Fourth International Workshop on Inductive Logic Programming* (1994), vol. 3, pp. 217–232.

[183] Stanhope, S. A., and Daida, J. M. Genetic programming for automatic target classification and recognition in synthetic aperture radar imagery. In *Proceedings of the 7th Annual Conference on Evolutionary Programming* (Mission Valley Marriott, San Diego, California, USA, 1998), pp. 735–744.

[184] Stefano, C. D., Cioppa, A. D., and Marcelli, A. Character preclassification based on genetic programming. *Pattern Recognition Letters 23*, 12 (2002), 1439–1448.

[185] Stone, M. Cross-validation and multinomial prediction. *Biometrika 61* (1974), 509–515.

[186] Sumner, M., Frank, E., and Hall, M. Speeding up logistic model tree induction. In *PKDD'05: Proceedings of the 9th European Conference on Principles and Practice of Knowledge Discovery in Databases* (2005), pp. 675–683.

[187] Superby, J., Vandamme, J., and Meskens, N. Determination of factors influencing the achievement of the first-year university students using data mining methods. In *EDM'06: Proceedings of the 1st Workshop on Educational Data Mining* (Hong Kong, China, 2006), pp. 37–44.

[188] Tan, K. C., Chiam, S. C., Mamun, A. A., and Goh, C. K. Balancing exploration and exploitation with adaptive variation for evolutionary multi-objective optimization. *European Journal of Operational Research 197*, 2 (2009), 701–713.

[189] Tan, K. C., Tay, A., Lee, T. H., and Heng, C. M. Mining multiple comprehensible classification rules using genetic programming. In *CEC'02: Proceedings of the IEEE Congress on Evolutionary Computation* (Honolulu, HI, USA, May 2002), vol. 2, pp. 1302–1307.

[190] Tan, K. C., Yu, Q., Heng, C. M., and Lee, T. H. Evolutionary computing for knowledge discovery in medical diagnosis. *Artificial Intelligence in Medicine 27*, 2 (2003), 129–154.

[191] Tao, Q., and Scott, S. A faster algorithm for generalized multiple-instance learning. In *FLAIRS'04: Proceedings of the 17th International Florida Artificial Intelligence Research Society Conference* (Miami Beach, FL, 2004), B. V. and M. Z., Eds., vol. 2, pp. 550–555.

[192] Tao, Q., Scott, S., Vinodchandran, N. V., and Osugi, T. T. SVM-based generalized multiple-instance learning via approximate box counting. In *ICML'04: Proceedings, 21st International Conference on Machine Learning* (Banff, Alta, 2004), pp. 799–806.

[193] Tao, Q., Scott, S., Vinodchandran, N. V., Osugi, T. T., and Mueller, B. An extended kernel for generalized multiple-instance learning. In *ICTAI'04: Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence* (Boca Raton, FL, 2004), K. T.M., Ed., pp. 272–277.

[194] Tsakonas, A. A comparison of classification accuracy of four genetic programming-evolved intelligent structures. *Information Sciences 176*, 6 (2006), 691–724.

[195] Veldhuizen, A. V. *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*. PhD thesis, Department of Electrical and Computer Engineering. Graduate School of Engineering, Air Force Institute of Technology, Wright- Patterson AFB, Ohio, May 1999.

[196] Veldhuizen, D. A. V., and Lamont, G. B. Evolutionary computation and convergence to a pareto front. In *Late Breaking Papers at the genetic programming* (Stanford University, California, 1998), J. Koza, Ed., Morgan Kaufmann, pp. 221–228.

[197] Veldhuizen, D. A. V., and Lamont, G. B. Multiobjective Evolutionary Algorithm Test Suites. In *Proceedings of the ACM Symposium on Applied Computing* (San Antonio, Texas, 1999), J. Carroll, H. Haddad, D. Oppenheim, B. Bryant, and G. B. Lamont, Eds., ACM, pp. 351–357.

[198] Veldhuizen, D. A. V., and Lamont, G. B. Multiobjective Evolutionary Algorithms: Analyzing the State-of-the-Art. *Evolutionary Computation 7*, 3 (2000), 1–26.

[199] Ventura, S., Romero, C., Zafra, A., Delgado, J. A., and Hervás, C. JCLEC: A java framework for evolutionary computation. *Soft Computing 12*, 4 (2008), 381–392.

[200] Venturini, G. Sia: A supervised inductive algorithm with genetic search for learning attributes based concepts. In *ECML'93: Proceedings of the European Conference on Machine Learning* (1993), Springer-Verlag, Berlin, Heidelberg, pp. 280–296.

[201] Wang, J., and Zucker, J.-D. Solving the multiple-instance problem: A lazy learning approach. In *ICML'00: Proceedings of the 17th International Conference on Machine Learning* (San Francisco, CA, USA, 2000), Morgan Kaufmann Publishers, pp. 1119–1126.

[202] Wang, S. X., and Lichodzijewski, P. Boolean genetic programming for promoter recognition in eukaryotes. In *Proceedings of the IEEE Congress on Evolutionary Computation* (Edimburgh, UK, 2005), pp. 683–690.

[203] Webb, G. Decision tree grafting from the all-tests-but-one partition. In *IJCAI'99: Proceedings of the 16th International Joint Conference on Artificial Intelligence* (1999), pp. 702–707.

[204] Wee, H., Lo, C., and Hsu, P. A multi-objective joint replenishment inventory model of deteriorated items in a fuzzy environment. *European Journal of Operational Research 197*, 2 (2009), 620–631.

[205] Weidmann, N., Frank, E., and Pfahringer, B. A two-level learning method for generalized multi-instance problems. In *ECML'03: 14th European Conference on Machine Learning* (2003), pp. 468–479.

[206] Whigham, P. A. Grammatically-based genetic programming. In *Proceedings of the Workshop on Genetic Programming: From Theory to Real-World Applications* (California, USA, 1995), pp. 33–41.

[207] Whigham, P. A. *Grammatical Bias for Evolutionary Learning*. PhD thesis, School of Computer Science. University College, University of New South Wales, Australia, 1996.

[208] Wiens, T. S., Dale, B. C., Boyce, M. S., and Kershaw, P. G. Three way k-fold cross-validation of resource selection functions. *Ecological Modelling 212*, 3-4 (2008), 244–255.

[209] Wilcox, J. R. *Organizational learning within a learning classifier system*. PhD thesis, University of Illinois at Urbana Champaign, 1995.

[210] Wilcoxon, F. Individual comparisons by ranking methods. 80–83.

[211] Witten, I. H., and Frank, E. *Data Mining: Practical Machine Learning Tools and Techniques*, 2 ed. Morgan Kaufmann, 2005.

[212] Wolpert, D., and Macready, W. No free lunch theorems for search. In *Technical Report* (Santa Fe Institute, 1996).

[213] Wong, M. L., Lam, W., Leung, K. S., Ngan, P. S., and Cheng, J. C. Y. Discovering knowledge from medical databases using evolutionary algorithms. *IEEE Engineering in Medicine and Biology Magazine 19*, 4 (2000), 45–55.

[214] Wong, M. L., and Leung, K. S. *Data Mining Using Grammar-Based Genetic Programming and Applications*. Kluwer Academic Publishers, Norwell, MA, USA, 2000.

[215] Xu, X. *Statistical learning in multiple instance problems.* PhD thesis, Department of Computer Science. University of Waikato, 2003.

[216] Xu, X. *Statistical learning in multiple instance problems.* PhD thesis, Department of Computer Science. University of Waikato, Tauranga, New Zealand, 2003.

[217] Xu, X., and Frank, E. Logistic regression and boosting for labeled bags of instances. In *PAKDD'04: Proceedings of the 8th Conference of Pacific-Asia* (Sydney, Australia, 2004), vol. 3056 of *Lecture Notes in Computer Science*, pp. 272–281.

[218] Xue, X., Han, J., Jiang, Y., and Zhou, Z. H. Link recommendation in web index page based on multi-instance leaning techniques. *Computer Research and Development 44*, 3 (2007), 106–411.

[219] Yang, C., Dong, M., and Fotouhi, F. Region based image annotation through multiple-instance learning. In *Multimedia'05: Proceedings of the 13th Annual ACM International Conference on Multimedia* (New York, USA, 2005), pp. 435–438.

[220] Yang, C., and Lozano-Perez, T. Image database retrieval with multiple-instance learning techniques. *Proceedings of the International Conference on Data Engineering* (2000), 233–243.

[221] Yang, Y., and Padmanabhan, B. Evaluation of online personalization systems: A survey of evaluation schemes and a knowledge-based approach. *Journal of Electronic Commerce Research 6*, 2 (2005), 112–122.

[222] Yu, J., Almal, A. A., Dhanasekaran, S. M., Ghosh, D., Worzel, W. P., and Chinnaiyan, A. M. Feature selection and molecular classification of cancer using genetic programming. *Neoplasia 9*, 4 (2007), 292–303.

[223] Yuan, X., Hua, X.-S., Wang, M., Qi, G.-J., and Wu, X.-Q. A novel multiple instance learning approach for image retrieval based on adaboost feature selection. In *Proceedings of the International Conference on Multimedia and Expo* (2007), pp. 1491–1494.

[224] Zafra, A., Ventura, S., Romero, C., and Herrera-Viedma, E. Multi-instance genetic programming for web index recommendation. *Expert System and Applications 36*, 9 (2009), 11470–11479.

[225] Zha, Z.-J., Hua, X.-S., Mei, T., Wang, J., Qi, G.-J., and Wang, Z. Joint multi-label multi-instance learning for image classification. In *CVPR'08: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition* (2008), pp. 1–8.

[226] Zhang, C., and Chen, X. Region-based image clustering and retrieval using multiple instance learning. In *CIVR'05: Proceedings of the 4th International Conference on Image and Video Retrieval* (2005), vol. 3568, pp. 194–204.

[227] Zhang, M., and Smart, W. Using gaussian distribution to construct fitness functions in genetic programming for multiclass object classification. *Pattern Recognition Letters 27*, 11 (Aug. 2006), 1266–1274.

[228] Zhang, M.-L. M3miml: A maximum margin method for multi-instance multi-label learning. In *ICDM'08: Proceedings of the 8th IEEE International Conference on Data Mining* (2008), pp. 688–697.

[229] Zhang, M.-L., and Zhou, Z.-H. Improve multi-instance neural networks through feature selection. *Neural Processing Letters 19*, 1 (2004), 1–10.

[230] Zhang, M.-L., and Zhou, Z.-H. Adapting RBF Neural Networks to multi-instance learning. *Neural Processing Letters 23*, 1 (2006), 1–26.

[231] Zhang, M.-L., and Zhou, Z.-H. Ml-knn: A lazy learning approach to multi-label learning. *Pattern Recognition 40*, 7 (2007), 2038 – 2048.

[232] Zhang, M.-L., and Zhou, Z.-H. Multi-instance clustering with applications to multi-instance prediction. *Applied Intelligence 31*, 1 (2009), 47–68.

[233] Zhang, Q., and Goldman, S. EM-DD: An improved multiple-instance learning technique. In *NIPS'01: Proceedings of Neural Information Processing System* (2001), vol. 14, pp. 1073–1080.

[234] Zhang, Q., Goldman, S., Yu, W., and Fritts, J. Content-based image retrieval using multiple-instance learning. In *ML'02: Proceedings of the 9th of Conference on Machine Learning* (New South Wales, Sydney, Australia, 2002), pp. 682–689.

[235] Zhou, Z.-H. Multi-instance learning from supervised view. *Journal of Computer Science and Technology 21*, 5 (2006), 800–809.

[236] Zhou, Z.-H., Jiang, K., and Li, M. Multi-instance learning based web mining. *Applied Intelligence 22*, 2 (2005), 135–147.

[237] Zhou, Z.-H., Jiang, K., and Li, M. Multi-instance learning based web mining. *Applied Intelligence 22*, 2 (2005), 135–147.

[238] Zhou, Z.-H., and Xu, J.-M. On the relation between multi-instance learning and semi-supervised learning. In *ICML'07: Proceedings of the 24th international conference on Machine learning* (Corvalis, Oregon, 2007), ACM, pp. 1167–1174.

[239] Zhou, Z.-H., Xue, X.-B., and Jiang, Y. Locating regions of interest in cbir with multi-instance learning techniques. In *AI'05: Proceedings of the 18th Australian Joint Conference on Artificial Intelligence* (Sydney, Australia, 2005), vol. 3809 of *Lecture Notes in Artificial Intelligence*, pp. 92–101.

[240] Zhou, Z.-H., and Zhang, M.-L. Neural networks for multi-instance learning. Technical report, AI Lab, Computer Science and Technology Department. Nanjing University, Nanjing, China, August 2002.

[241] Zhou, Z.-H., and Zhang, M.-L. Ensembles of multi-instance learners. In *ECML'03: Proceedings of the 14th European Conference on Machine Learning* (Cavtat-Dubrovnik, 2003), vol. 2837, pp. 492–502.

[242] Zhou, Z.-H., and Zhang, M.-L. Solving multi-instance problems with classifier ensemble based on constructive clustering. *Knowledge and Information Systems 11*, 2 (2007), 155–170.

[243] Zitzler, E. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. PhD thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, November 1999.

[244] Zitzler, E., Deb, K., and Thiele, L. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation 8*, 2 (Summer 2000), 173–195.

[245] Zitzler, E., Laumanns, M., and Thiele, L. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Tech. Rep. 103, Gloriastrasse 35, CH-8092 Zurich, Switzerland, 2001.

[246] Zitzler, E., and Thiele, L. Multiobjective Optimization Using Evolutionary Algorithms. A Comparative Study. In *Proceedings of the 5th International Conference on Parallel Problem Solving from Nature* (Amsterdam, September 1998), Springer-Verlag, pp. 292–301.

[247] Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C. M., and da Fonseca, V. G. Performance Assessment of Multiobjective Optimizers: An Analysis and Review. *IEEE Transactions on Evolutionary Computation 7*, 2 (April 2003), 117–132.

[248] Zucker, J. D., and Ganascia, J. Changes of representation for efficient learning in structural domains. In *ICML'96: Proceedings of the 13th International Conference on Machine Learning* (Bary, Italy, 1996), pp. 543–551.

[249] Zucker, J. D., and Ganascia, J. Learning structurally indeterminate clauses. In *ILP'98: Proceedings of the 8th International Workshop on Inductive Logic Programming* (Berlin, 1998), vol. 1446 of *Lecture Notes in Artificial Intelligence*, pp. 235–244.

[250] Zydallis, J. B. *Explicit Building-Block Multiobjective Genetic Algorithms: Theory, Analysis, and Development*. PhD thesis, Air Force Institute of Technology, Department of the Air Force, Air University, Wright-Patterson, Airforce Base, Ohio, USA, March 2003.