UNIVERSIDAD DE HOLGUÍN

Facultad de Informática y Matemática

Estudio experimental sobre algoritmos de clasificación supervisada basados en reglas en conjuntos de datos de alta dimensión

Trabajo de Diploma para optar por el título de Ingeniero en Informática

Autor: Ariam Rivas Méndez

Tutor: Ing. Adrian Pino Angulo

Holguín, Cuba

Junio, 2014

UNIVERSIDAD DE HOLGUIN

OSCAR LUCERO MOYA

Agradecimientos

A mi mamá y mi papá que no existe un momento en el cual no me apoyen.

A mi abuela y mi hermano que siempre de alguna manera se preocupó y me ayudó.

Gracias a mi tutor que me acogió con los brazos abiertos y siempre estuvo dándome ánimo y atendiéndome cada vez que lo necesitaba a pesar de su pequeño espacio libre.

A Ricardo por confiar y apoyarme desde primer año. Por los consejos y los conocimientos brindados en estos años de trabajo.

A los profesores que me enseñaron a lo largo de la carrera.

A todos mis compañeros de grupo y mis grandes amigos, que siempre me apoyaron.

Para todos muchas gracias, este trabajo es fruto del amor, conocimiento y la amistad brindada por todos ustedes.

Dedicatoria

A mis padres por su entrega

Resumen

Los algoritmos de clasificación basados en reglas se han destacado debido a su gran número de ventajas. Sin embargo, a pesar de todas las ventajas que poseen y la rica cantidad de algoritmos de clasificación basados en reglas descritos en la literatura científica revisada; existe escasa evidencia, de decidir cuál de estos algoritmos es más propicio para conjunto de datos de alta dimensión.

Por lo que se realiza un estudio experimental de los algoritmos de clasificación basados en cubrimiento secuencial para la determinación de reglas, para su elección y aplicación. También se confecciona una taxonomía de los algoritmos de clasificación basados en reglas para una mejor comprensión y análisis.

Luego se efectúa una implementación de dos nuevos algoritmos, los cuáles se caracterizan por su velocidad y una mejora al algoritmo RIPPERk. Esta consiste en reordenar las reglas del algoritmo sin afectar el tiempo de procesamiento.

Se evalúa la efectividad de los principales métodos de clasificación basados en reglas ante conjuntos de datos de alta dimensión. Tomando como parámetros de evaluación la precisión, tiempo de ejecución y número de condiciones de la regla. Se hace uso de métodos estadísticos-matemáticos como la prueba no paramétricas de Friedman y la post-hoc de Holm. Esta evaluación se muestra en una gráfica para exponer los mejores clasificadores en cuanto a los parámetros seleccionados.

Abstract

The classification algorithms based on rules have stood out due to their great number of advantages. However, in spite of all the advantages that possess and the rich quantity of classification algorithms based on rules described in the scientific revised literature; scarce evidence exists, of deciding which of these algorithms it is more favorable for dataset of high dimension.

For what is carried out an experimental study of the classification algorithms based on sequential covering for the determination of rules, for their election and application. A taxonomy of the classification algorithms is also made based on rules for a better understanding and analysis.

Then an implementation of two new algorithms is made, those which they are characterized by its speed and an improvement to the algorithm RIPPERk. This consists on reordering the rules of the algorithm without affecting the time of processing.

The effectiveness of the main classification methods is evaluated based on rules before datasets of high dimension. Taking as evaluation parameters the precision, time of execution and number of conditions of the rule. Use of statistical-mathematical methods is made as the non parametric test of Friedman and the post-hoc of Holm. This evaluation is shown in a graph to expose the best classifiers as for the selected parameters.

Índice de Contenido

Índice de Tablas

Índice de Figuras

Introducción

Actualmente se ha incrementado la cantidad de información disponible en formato digital, pues la tecnología ha facilitado el almacenamiento y publicación de volúmenes de datos cada vez más grandes y con costes cada vez menores. Sin embargo, dichos avances han traído como consecuencia una mayor dificultad de procesamiento y extracción de información útil a partir de dichos datos. Estos datos son la fuente principal y más evidente de experiencia empresarial, científica y técnica acumulada sobre diversos dominios.

En determinados dominios existe una gran cantidad de variables, que por su complejidad, los datos son generalmente imposibles de estudiar por un experto humano para ejecutar predicciones empíricas. El experto humano, en el mejor de los casos, solo llegará a manejar e interrelacionar en su mente una parte de las variables dejando gran parte de los datos disponibles sin explorar [1]. Si se utilizaran todos, sería demasiado complejo o intratable en el tiempo del que dispone para tomar la decisión necesaria. Estos tipos de datos son conocidos en la literatura especializada como datos de alta dimensión [2], pues desde el punto de vista algebraico son datos que requieren una gran cantidad de dimensiones para ser representados.

La Inteligencia Artificial desarrolla procesos que imitan la inteligencia humana y enmarca disciplinas que sí responden de un modo más exitoso a la dificultad de la dimensión de los datos y la necesidad de aprender de ellos. El Descubrimiento de Conocimiento en Bases de Datos (KDD) pretende procesar grandes cantidades de información con el fin de descubrir conocimiento relevante. La Minería de Datos, constituye un subproceso de KDD, que recoge un grupo de técnicas como el Agrupamiento, la Regresión, la Asociación y la Clasificación; para la extracción de conocimiento de las bases de datos[3],[4]. Para ello, se apoya en el Aprendizaje Automático, que persigue que las computadoras sean capaces de inducir dicho conocimiento.

La clasificación es uno de los modelos más estudiados, su objetivo es asignar una clase a una instancia[1] descrita por un vector de características. Entre las técnicas clásicas para la construcción de un clasificador se encuentran las redes bayesianas[2], los árboles de decisión[3], las redes neuronales[4] y las reglas[5]. Esta investigación se enmarca principalmente en el estudio de las reglas.

Los algoritmos de clasificación basados en reglas se han destacado debido a su fácil comprensión por la estructura tan simple que poseen. Estos algoritmos son tan explícitos como los árboles de decisión, además son fáciles de interpretar y de generar, pueden clasificar nuevas instancias rápidamente y pueden manejar fácilmente valores faltantes y atributos numéricos. Su rendimiento es comparable con los árboles de decisión y muchos de ellos están disponibles en distintas herramientas de minería de datos como Weka[6].

A pesar de todas las ventajas mencionadas anteriormente y la rica cantidad de algoritmos de clasificación basados en reglas descritos en la literatura científica revisada; existe escasa evidencia, de decidir cuál de estos algoritmos es más propicio para conjunto de datos de alta dimensión. Esto generalmente conlleva a que tengan que invertir mucho tiempo de experimentación y puede llevar a la selección inadecuada de un algoritmo de aprendizaje.

De la situación problémica antes descrita surge el **problema científico**: ¿Cómo determinar la efectividad de los clasificadores basados en reglas en conjuntos de datos de alta dimensión? El mismo queda enmarcado en el **objeto de estudio**: el proceso de minería de datos a partir de clasificadores basados en reglas.

---

[1] Un par (X, Y), donde X representa los atributos de los datos de entrada e Y su clase correspondiente.
[2] Es un grafo dirigido acíclico conexo más una distribución de probabilidad sobre sus variables.
[3] Clasificadores supervisados, que a partir de un conjunto de objetos previamente clasificados se construye un árbol de decisión.
[4] Se basan en un modelo simplificado del cerebro humano
[5] Constituyen una técnica capaz de ser procesadas por un ordenador y entendibles por un ser humano.
[6] Sitio oficial: http://www.cs.waikato.ac.nz/~ml/Weka/

Ante el referido problema se traza como **objetivo:** Evaluar la efectividad de los algoritmos de clasificación basados en reglas, para su elección y aplicación en conjuntos de datos de alta dimensión.

El objetivo de la investigación delimita el **campo de acción:** clasificadores basados en reglas ante conjuntos de datos de alta dimensión.

Las siguientes **preguntas científicas** resultan una guía del trabajo de investigación:

- ¿Cuáles son las bases teóricas que sustentan los algoritmos de clasificación basados en reglas?
- ¿En qué radican los fenómenos que afectan la minería en datos de alta dimensión?
- ¿Qué características identifican a las principales técnicas de clasificación basadas en reglas?
- ¿En qué grado están disponibles los algoritmos de aprendizaje basados en reglas más destacados?
- ¿Cómo realizar un estudio experimental que ofrezca la evidencia necesaria para enfrentar el problema?
- ¿Cuán efectivos resultan los principales métodos de clasificación basados en reglas ante conjuntos de datos de alta dimensión?

Para responder las preguntas científicas y dar con ello cumplimiento al objetivo, fueron planteadas las siguientes **tareas de la investigación**:

1. Estudiar las bases teóricas que sustentan el aprendizaje automático para la construcción de modelos de clasificación basados en reglas.
2. Detallar los fenómenos que inciden en la minería de datos de alta dimensión y las estrategias para enfrentarlos.
3. Confeccionar una taxonomía de los algoritmos de clasificación.
4. Describir los algoritmos de aprendizaje basados en reglas para los cuales se reportan los resultados más destacados.
5. Seleccionar una herramienta de minería de datos e incluir los algoritmos destacados que no se encuentren en la misma.

6. Realizar un estudio experimental comparativo entre las técnicas más destacadas para hacer este tipo de minería de datos.
7. Evaluar la efectividad de los principales métodos de clasificación basados en reglas ante conjuntos de datos de alta dimensión.

Para ejecutar dichas tareas se utilizaron los siguientes métodos teóricos de investigación:

- Análisis y síntesis: en apoyo a la realización del estudio detallado del marco teórico de los algoritmos de aprendizaje basados en reglas.
- Histórico y lógico: para ver la evolución en el tiempo y facilitar la comprensión e ilustración de los algoritmos de aprendizaje basados en reglas.
- Enfoque sistémico-estructural: empleado en el diseño de una estructura adecuada para la implementación de los algoritmos seleccionados que no estaban disponibles en la herramienta de minería escogida para realizar la experimentación.

Por otra parte, entre los métodos empíricos que se usaron se encuentran:

- Revisión de documentos: se utiliza para la comprensión de las técnicas de aprendizaje automático, su estado actual a nivel mundial, los principales algoritmos existentes y las proyecciones futuras.
- Experimentación: empleada durante la comparación de los algoritmos de aprendizaje automático abordados en este estudio.

Así mismo, se hizo uso de métodos estadísticos-matemáticos.

- Prueba no paramétricas de Friedman: para comparar el rendimiento de todos los algoritmos utilizados y observar si existen diferencias significativas de manera general.
- Prueba post-hoc de Bergmann-Hommel: se utiliza para detectar diferencias significativas en el rendimiento de pares de algoritmos de aprendizaje automático.

Este trabajo de investigación tiene sus bases en el aprendizaje automático, con particular énfasis en las técnicas de clasificación supervisada basadas en reglas. Sé realizó una comparación analítica experimental para detectar diferencias significativas entre los algoritmos de aprendizaje basados en reglas en conjuntos de datos de alta dimensión.

El documento está estructurado en introducción, tres capítulos, conclusiones, recomendaciones, referencias bibliografía, glosario de términos y anexos.

En el capítulo I se introduce el tema del aprendizaje automático basado en reglas con el fin de ilustrar el marco teórico de esta técnica a través de la descripción detallada de conceptos, definiciones y temas afines. Además, se propone una taxonomía para destacar sus principales diferencias y semejanzas.

En el capítulo II se realiza el análisis y descripción de los algoritmos más actuales y populares de esta técnica para conjuntos de datos de alta dimensión. Se presenta la implementación de dos nuevos algoritmos y una mejora al algoritmo RIPPERk.

En el capítulo III se realiza una evaluación experimental en conjuntos de datos reales y artificiales de alta dimensión, con el fin de analizar las ventajas y deficiencias de los algoritmos seleccionados.

Capítulo I: Marco teórico de los clasificadores basados en reglas

En este capítulo se abordan las diferentes fases del proceso de descubrimiento de conocimiento en conjuntos de datos (KDD, por sus siglas en inglés), haciendo énfasis en los principales conceptos y definiciones pertenecientes a esta amplia rama de la inteligencia artificial. Dentro de este proceso se aborda con mayor interés el aprendizaje automático supervisado como soporte fundamental para la extracción de conocimiento útil en grandes volúmenes de datos. En este sentido se muestra una taxonomía para identificar semejanzas y diferencias entre los algoritmos de aprendizaje automático supervisado y se analizan de manera breve sus ventajas y desventajas.

1.1 Proceso de descubrimiento de conocimiento en base de datos

Con el desarrollo de la tecnología y los dispositivos electrónicos el hombre ha logrado almacenar de manera digital información obtenida del mundo real, la misma ha sido utilizada para la observación y mejoramiento de diversos proceso. Esto ha traído como consecuencia la necesidad de desarrollar técnicas y herramientas que permitan asistir al hombre a extraer información útil (conocimiento, patrones) de los datos almacenados. Para suplir esta necesidad surge el proceso de KDD. El mismo es definido por [5] como:

“El proceso no trivial de identificación, en los datos, de patrones válidos, novedosos, comprensibles y potencialmente útiles”.

Este proceso se nutre de varias disciplinas tales como: estadísticas, base de datos, aprendizaje automático e inteligencia artificial (IA) [6] [7] [4][8]. Pretende además visualizar el conocimiento encontrado de una manera que le sea entendible y útil al usuario, también dicho conocimiento ha de ser interesante y de alta calidad.

El KDD es un área de la computación que intenta explotar la enorme cantidad de información mediante el descubrimiento de patrones representativos y útiles,

extrayendo conocimiento que pueda asistir a un humano para llevar a cabo tareas de forma más eficiente y satisfactoria.

En la Figura 1.1-1 se muestran de manera general las fases por las que atraviesa cualquier proceso *KDD* [7]. Las mismas, son detalladas de manera simplificada a continuación.

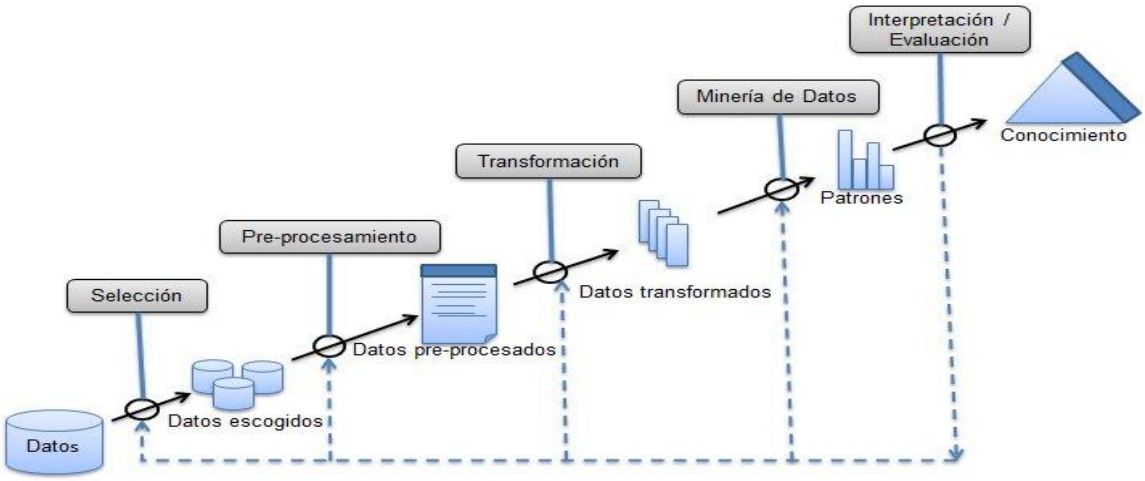

Figura 1.1-1 Esquema general de proceso KDD Fuente: [9]

1. Selección: desarrolla un entendimiento del dominio del problema y de los datos que serán utilizados en la tarea de descubrimiento de conocimiento.

2. Pre-procesamiento y transformación: cubre todas las actividades para la construcción de conjunto de datos final. Esta tarea incluye selección de registros, atributos, limpieza de los datos, tratamiento de los valores ausentes, entre otros. También se realiza la transformación de los datos en el formato requerido por la herramienta de minería de datos seleccionada. Esta tarea consume entre el 35% y 20% del tiempo [10].

3. Minería de datos (*MD*): es la determinación de la tarea de descubrimiento a realizar (clasificación, regresión, agrupamiento,…) y la aplicación de uno o varios algoritmos, de dicha tarea, con el fin de descubrir patrones ocultos en los datos. Esta tarea ocupa entre el 15% y 20% del tiempo de realización del proyecto[11].

4. Interpretación y evaluación: se interpretan y evalúan los patrones descubiertos, por lo que en ocasiones es necesario regresar a los pasos anteriores, lo que implica repetir el proceso, tal vez con otros datos, algoritmos, metas y estrategias. Este paso puede ser auxiliado por visualizaciones y contribuye a eliminar patrones redundantes o irrelevantes.

En ocasiones en la bibliografía se observa la fase aplicación, que hace referencia a como se ha de utilizar el conocimiento extraído, tomando provecho de él y realizando acciones, mediante su incorporación a un sistema de desempeño.

Los realizadores de este tipo de proyecto se apoyan en programas computacionales como Weka [12], Keel [13], Orange [14], RapidMiner entre otros, y en metodologías como la de Usama Fayyad y Piatetsky- Shapiro [9], *CRISP-DM*(*CRoss-Industry Standard Processfor Data Mining)* [15] y Krzysztof J. Cios [10] que le posibilitan agilizar el desarrollo de los proyectos así como proporcionar una detallada documentación del mismo.

1.2 Adquisición y representación de datos

El proceso de adquisición de datos en la proceso de KDD involucra la captura de los mismos a partir de diferentes fuentes (manuales, libros, expertos, sensores, base de datos, etc.) y su representación de manera tal que puedan ser empleados por las computadoras para la extracción de conocimiento.

En una típica tarea de aprendizaje automático los datos son representados como una tabla de ejemplos, instancias u objetos, los mismos se describen a partir de las componentes de un vector [16].

Estas componentes son conocidas como **atributos**, características o rasgos, los cuales pueden ser de uno de los dos tipos, nominales (miembro de algún conjunto de valores finitos) conocidos también como categóricos y numéricos (valores reales o enteros).

A manera de ejemplo, en la Tabla 1.2-1, se muestra un conjunto de datos organizados para la realización de la tarea de *MD*, perteneciente a pacientes que se realizan el chequeo de hipertensión.

Este conjunto de datos está compuesto por cuatro atributos (Fumador, Colesterol, Cargo e Hipertensión), donde el último es la clase, o variable dependiente como se le conoce estadísticamente, y su valor dependerá en mayor o menor medida de los restantes atributos o lo que es lo mismo de su ejemplo correspondiente.

Tabla 1.2-1: Conjunto de datos para el pronóstico de la hipertensión Fuente: [17]

Paciente	Fumador	Colesterol()	Cargo	Hipertensión?
1	No	3,9	Bajo	No
2	No	4,1	Medio	No
3	Si	4,9	Bajo	No
4	No	5,3	Medio	No
5	Si	5,7	Bajo	No
6	No	6,1	Alto	No
7	No	6,3	Bajo	No
8	Si	6,8	Medio	Si
9	Si	7,2	Bajo	Si
10	No	7,3	Medio	No
11	Si	7,7	Alto	Si
12	No	7,9	Bajo	No
13	Si	8,0	Alto	Si
14	No	8,2	Medio	No
15	Si	8,3	Bajo	Si

En síntesis, un conjunto de datos está constituido por las relaciones existentes entre una serie de atributos que en ocasiones describen a un conjunto de clases.

1.3 Pre-procesamiento de los datos

La utilidad del conocimiento extraído a partir de los datos mediante técnicas de minería de datos depende en gran medida de la calidad de los datos [11]. Como se evidencia en las secciones anteriores en todo proceso de descubrimiento de conocimiento existe una o varias fases de preparación de los datos (pre-procesamiento) previas al análisis de los mismos. El pre-procesamiento pretende obtener un conjunto de datos tales que al aplicar

técnicas de *MD* sobre ellos se generen modelos representativos con mayores prestaciones [18].

La importancia de este proceso se evidenciada principalmente en que mucho de los datos almacenados están sin pre-procesar, a menudo contiene valores incompletos, inconsistentes y ruidosos. Estas limitantes han influenciado en el surgimiento de una diversa gama de métodos de pre-procesamiento, que son clasificados en cuatro departamentos [19] como se muestra en la Figura 1.3-1.

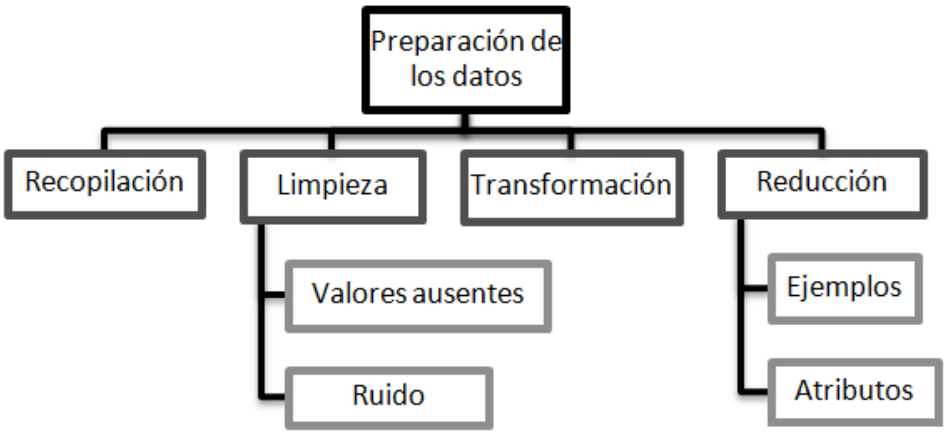

Figura 1.3-1 Esquema modular del pre-procesamiento Fuente: [19]

Para poder comenzar el análisis y extraer algo útil en los datos es preciso, en primer lugar, disponer de ellos. Esta tarea de recopilación en algunos casos puede parecer trivial, partiendo de que los datos sean extraídos de un simple archivo, sin embargo en los escenarios prácticos es una tarea compleja, donde deben de resolverse problemas de representación, codificación e integración de diferentes fuentes para crea un conjunto de datos homogéneo.

La tarea de limpieza de los datos persigue aumentar la calidad de los datos al nivel requerido mediante técnicas de análisis selectivo, como tratamiento del ruido, valores ausentes y *outliers (*valores cercanos al límite del rango del atributo que van en contra de la tendencia que siguen el resto de los datos)[11].

Por otra parte dentro de las técnicas de transformación de datos se encuentra la discretización, continuización, normalización o cambio de escala y la selección de atributos. Se consideran técnicas de transformación aquellas

destinadas a modificar los datos para mejorar el proceso de aprendizaje y no a corregir los errores presentes en los mismos[11], es por lo que la selección de atributos es considerara también como una de estas.

La reducción de datos consiste en decidir qué datos deben ser utilizados para el análisis [18]. Dentro de la misma se encuentra la Selección de Ejemplos [20] que reduce el número de instancias en el conjunto de datos y la Selección de Atributos [16], [17] que intenta obtener los atributos que más sintetiza la información contenida en el mismo.

Las estrategias anteriormente descritas no son mutuamente excluyentes. Existen técnicas de pre-procesamiento que utilizan dos o más de las estrategias previamente mencionadas y habría que catalogarlas como una combinación de las mismas.

1.4 Minería de datos

La Minería de datos (MD) comprende un grupo de métodos para la extracción de conocimiento procesable a partir de datos previamente acumulados, ofreciendo soluciones a problemas de predicción, clasificación, segmentación, entre otros. Para ello se nutre de diversas esferas científicas como la estadística, la computación gráfica, las bases de datos y el aprendizaje automático.

En la Figura 1.4-1 se puede observar, la interrelación y el agrupamiento entre diversos métodos de MD, mediante la ilustración de su taxonomía [21]. Es importante destacar entre dos principales tipos de minería: **verificación** (el sistema verifica las hipótesis del usuario) y **descubrimiento** (el sistema encuentra de manera automática nuevas reglas y patrones).

Los métodos de verificación son menos asociados con MD que los de descubrimiento, dado que los problemas de MD son más enfocados en el descubrimiento de nuevo conocimiento oculto en los datos, en lugar de probar el conocimiento brindado por alguna fuente externa (expertos).

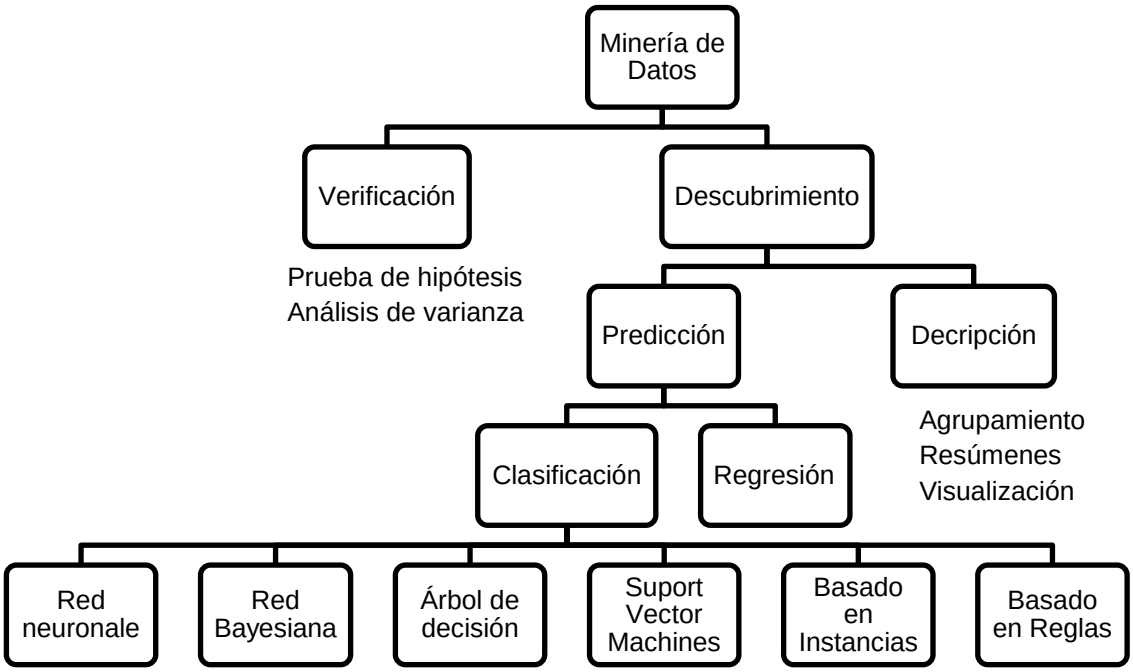

Figura 1.4-1 Taxonomía de la minería de datos Fuente: [21]

Los métodos de descubrimiento engloban las tareas de predicción y de descripción. Esta última está orientada a la interpretación de los datos, enfocándose en su entendimiento y las relaciones existentes en los mismos.

Las tareas de predicción construyen un modelo a partir del conjunto de instancias, el cual es utilizado para predecir el valor del atributo clase de una nueva instancia. En dependencia de si el atributo clase es discreto o continuo la tarea de predicción puede ser de clasificación o de predicción numérica respectivamente. Esta última tarea, correlaciona al espacio de entrada con el dominio de valores reales. Mientras que, los clasificadores correlacionan el espacio de entrada con clases predeterminadas.

La clasificación es uno de los modelos más estudiados, posiblemente él de mayor relevancia práctica. Los beneficios potenciales del progreso en la clasificación son inmensos, ya que esta técnica tiene un gran impacto sobre otras áreas, estas se encuentran dentro de la MD y de sus aplicaciones [21].

El objetivo de la clasificación es asignar una clase a un objeto descrito por un vector de características. La construcción de un sistema de clasificación tiene

dos pasos: el entrenamiento y la prueba del sistema. Cuando el sistema está en fase de entrenamiento, aprende a clasificar un conjunto de patrones cuya correcta clasificación es conocida previamente, llamados patrones de entrenamiento. Finalizado este paso el sistema entra en la fase de prueba, en la cual ha de usar el conocimiento aprendido en el entrenamiento para clasificar patrones que no ha tratado anteriormente [22]. Las técnicas clásicas de construcción de clasificadores son las redes bayesianas, los árboles de decisión, las redes neuronales y los basados en reglas.

Estos métodos de descubrimiento han constituido el principal objeto de estudio de la disciplina de aprendizaje automatizado, los cuales, de acuerdo con varios autores especializados en el tema, persiguen adquirir nuevo conocimiento mediante la obtención de un conjunto de leyes generales a partir de la observación de casos particulares (Aprender) [3], [4], [23]. Lo cual, se refiere a la construcción de un modelo abstracto a partir de los casos que describen un determinado dominio. Estableciendo que un modelo aprende cuando exhibe una mayor habilidad en la ejecución de la tarea para la cual fue construido o entrenado [24]. Este enfoque apunta claramente al carácter cuantificable del proceso de aprendizaje partiendo que uno o varios indicadores de desempeño permitan medir la habilidad en cuestión.

La construcción del modelo ocurre a partir de la experiencia acumulada sobre el dominio de un conjunto de datos. Por lo general, dichos datos deben ser suministrados a los algoritmos de aprendizaje estructurados en forma de “casos conocidos” (instancias de entrenamiento). De este modo, cada instancia almacenada es un caso, una ocurrencia, de un suceso independiente recogido en las primeras fases del proceso KDD. Como resultado se obtiene un conjunto de datos estructurados, que reciben la denominación de “datos de entrenamiento” o de “aprendizaje”. En la Figura 1.1.4-2 se muestra lo antes expuesto lo cual es conocido como el paradigma del aprendizaje inductivo.

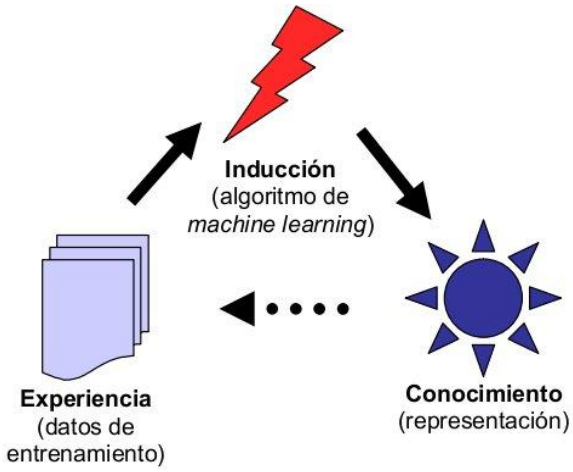

Figura 1.1.4-2: Paradigma del aprendizaje inductivo automatizado Fuente: [24]

La comunidad del aprendizaje automático para referirse a las tareas de predicción utiliza los términos: aprendizaje supervisado y aprendizaje no supervisado.

El modelo construido a partir de técnicas de aprendizaje supervisado es capaz de etiquetar o de asignarle un valor numérico, al atributo clase de una nueva instancia nunca antes observada, en dependencia de si el modelo es de clasificación o regresión.

Por otra parte, las técnicas de aprendizaje no supervisado no disponen del atributo clase para la confección de un modelo, en su lugar intenta extraer el conocimiento a partir del descubrimiento de interrelaciones existentes entre los elementos de datos. Los principales tipos de tareas del aprendizaje no supervisado son: agrupamiento y asociación.

En este documento se trata específicamente el aprendizaje supervisado ya que existe un interés un su utilización en conjuntos de datos de entrenamiento reales de nuestra provincia con el fin de predicciones futuras en estos dominios.

1.4.1 Aprendizaje automático supervisado

El aprender de ejemplos es un problema conocido como aprendizaje supervisado, ya que existe una serie de clases o categorías diseñadas a priori,

en las cuales hay que distribuir a cada uno de los nuevos datos. La clasificación supervisada también conocida como categorización [25] parte de la existencia de un conjunto de clases pre-establecidas, cuyo objetivo es colocar cada información en la clase que le corresponda.

La mayoría de los algoritmos elaboran un modelo o patrón para cada clase, esta fase se conoce como entrenamiento, la cual necesita un conjunto de datos ya clasificados manualmente. Estos algoritmos requieren intervención humana para la clasificación del conjunto de entrenamiento y para la revisión y refinamiento de resultado, de ahí viene el término “supervisado” [25].

El aprendizaje automático, también llamado aprendizaje artificial [26], es un área de interés muy desarrollada en la IA.

El aprendizaje es un término muy general que denota la forma, o formas, en la cual una máquina aumenta su conocimiento y mejora su desempeño (performance) en un entorno. De esta manera, el proceso de aprendizaje puede ser visto como un generador de cambios en el sistema que aprende y que pueden ser revocados o ampliados. Estos cambios se refieren no solo a la mejora de las capacidades y habilidades para realizar tareas sino que también implican modificaciones en la representación de hechos conocidos [27].

En este contexto, se dice que un sistema que aprende de forma automatizada es un conjunto de algoritmos que, mejora su actuación a la hora de resolver problemas de toma decisiones basadas en la experiencia acumulada. Estos sistemas deben ser capaces de trabajar con un rango muy amplio de tipos de datos de entrada, que pueden incluir datos incompletos, inciertos, ruido, inconsistencias, etc.

En la Figura 1.4-3 se muestra una taxonomía para los algoritmos de aprendizaje automático supervisado donde se agrupan sus principales métodos. Los algoritmos de aprendizaje basados en reglas definen una serie de reglas para cada una de las clases, los basados en reglas de inducción tienen un procedimiento distinto porque cuando se está construyendo la regla, en cada etapa, añade a la condición en curso atributos que optimizan la evaluación de la regla. Los basados en teoría de probabilidad se caracterizan

por tener una base probabilística, calcula la probabilidad de que una instancia encaje en una clase a partir de la probabilidad de que, instancias que contengan determinados términos pertenezcan a esa clase. Los basados en decisión se caracterizan por extraer reglas de un árbol de decisión sin podar. Por otra parte, los multiclasificadores, que se encuentran divididos en: para trabajo offline y online [28], se caracterizan por tener un conjunto de expertos, cada uno de ellos es un método de aprendizaje que para clasificar a una instancia y se toma una decisión teniendo en cuenta todas las propuestas. El trabajo offline no es más que una técnica perteneciente al aprendizaje por lote, el proceso de aprendizaje consiste en procesar repetidas veces el conjunto de entrenamiento hasta obtener un modelo final que describa al mayor número de ejemplos de entrenamiento posible, este es un proceso estático, y en el trabajo online el dominio va a hacer incremental pero sigue el mismo esquema de las técnicas de aprendizaje por lote, basándose en heurísticas y medidas de interés para eliminar, reactivar o añadir dinámicamente nuevos algoritmos de aprendizaje en respuesta a las variaciones ocurridas, este proceso es dinámico.

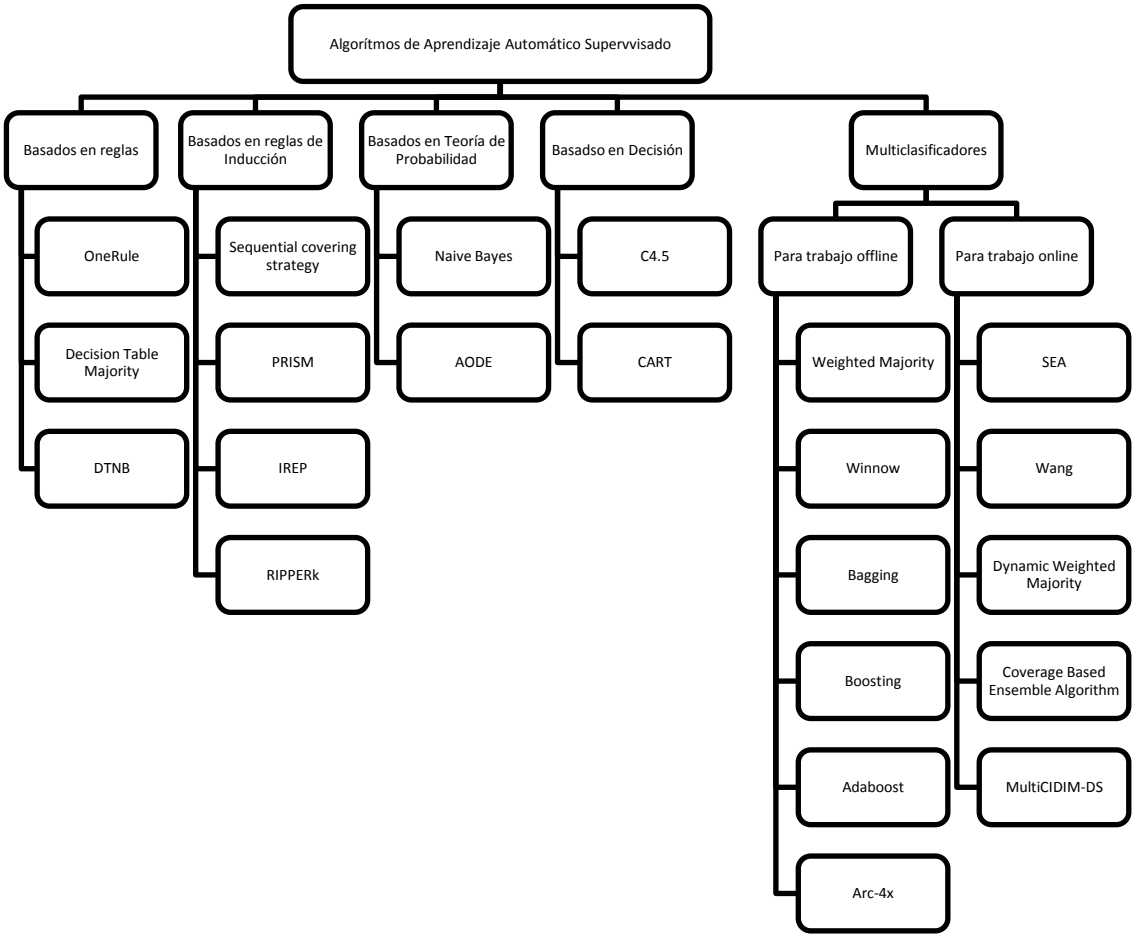

Figura 1.4-3: Taxonomía de los algoritmos de aprendizaje automático supervisado

Los clasificadores basados en reglas son unos de los modelos más utilizados en aplicaciones comerciales desde el comienzo de 1980 y han sido especialmente beneficiosos en la construcción de sistemas expertos y MD [29][30][31]. Los bancos modernos están usando este modelo dado que anteriormente con un software tradicional determinar qué solicitante de préstamo incumpliría o pagaría en el momento oportuno su deuda, desafortunadamente tenía una tasa de acierto menor del 50%. Con los clasificadores basados en reglas la tasa de acierto está por encima del 70% [32], American Express UK es una de las instituciones financieras que lo utiliza.

En general la valoración de riesgo por inversión es un área donde las reglas de inducción se destacan, especialmente ya que muchas compañías tendrán abundantes registros que pueden servir como datos de entrenamiento. IBM

comercializa el software usando reglas de inducción para la valoración de riesgo a compañías de seguros [31].

Los clasificadores basados en reglas poseen algunas características atractivas, ellas son explicitas, comprensibles y comprobable por los expertos de dominio, y puede ser modificada y extendida [33]. Otra ventaja que proporciona estos clasificadores es la habilidad de encontrar complicadas relaciones interconectadas en grandes cantidades de datos.

1.5 Datos de alta dimensión

En los estudios iniciales de los algoritmos para la estimación y la predicción se tenían en cuenta una cantidad de variables que resultaba de fácil comprensión humana. Los modelos de 2 y 3 variables son fáciles de representar y analizar en un espacio de 3 dimensiones, asignando una dimensión del espacio a cada variable y representando los objetos o casos como puntos en dicho espacio. Pero 4, 5 o más variables no son tan fáciles de representar en un espacio. En el año 1948, un modelo de 4 o 5 variables se calificaba “de alta dimensión” [34] dado que su representación requiere de un espacio vectorial de más dimensiones de las que es posible analizar intuitivamente.

A partir de 1961 inició a estilarse la frase de “maldición de la alta dimensión” [35] refiriéndose al efecto que ejerce el incremento de dimensiones en el espacio, sobre el proceso de búsqueda. La búsqueda exhaustiva al necesitar $2^{\mathbf{d}}$ evaluaciones para optimizar una función de **d** variables binarias, se vuelve intratable con el aumento del número de dimensiones. Esta situación se ve afectada en dominios que contienen conjuntos de datos definidos por miles de variables [36].

En las investigaciones de los efectos provocados por el aumento de variables que representan los datos, se ha descubierto ciertos fenómenos en los espacios de alta dimensión [2]. Estos fenómenos se resumen a continuación:

- La concentración de las distancias. Las métricas tradicionales utilizadas para mediciones de distancia entre dos puntos en un espacio vectorial, por ejemplo la norma Euclidiana, arrojan mediciones menos intuitivas

cuando son utilizadas en espacios de cientos o miles de dimensiones. Para dimensiones muy altas, todas las mediciones de distancia entre dos puntos cualesquiera de un conjunto de datos son tan grandes (o pequeñas) en magnitud, que la noción de “cerca” o “lejos” se distorsiona. Ocurre que la distancia de un punto a su vecino más próximo se asemeja a la distancia al punto más lejano. Por ello se dice que las distancias entre puntos vecinos se concentran alrededor de un valor, mostrando una varianza muy pequeña. Lógicamente, una métrica de distancia que arroja siempre aproximadamente el mismo valor, sean cuales sean los puntos, es inútil como criterio de similitud entre objetos de un conjunto de datos, afectando a muchos algoritmos geométricos y basados en la búsqueda de vecinos.

- El fenómeno del espacio vacío. Con el crecimiento de la cantidad de dimensiones, se necesitan muchos más puntos de datos para cubrir de forma densa y uniforme una región del híper-espacio. Si en un espacio de 2 dimensiones se considera que 100 puntos de datos son suficientes para cubrir densamente una unidad cuadrada, al pasar a 3 dimensiones se necesitarán 1000 puntos. En un espacio de **n** dimensiones se necesitarán 10^n puntos para cubrir densamente la misma región. Sin embargo, por lo general no se cuenta con tantos datos, por tanto la región bajo estudio estará mucho menos densamente cubierta. Para miles de dimensiones, el espacio parecerá prácticamente vacío, a pesar de que se cuente con muchos casos en el conjunto de datos, pues no serán suficientes de ninguna manera. Esto afecta severamente la estimación de la distribución probabilística de los datos, lo cual es directa o indirectamente la base de muchas técnicas de minería de datos, como las estadísticas.
- El fenómeno de Hughes. Bautizado con el nombre de Greg Hughes, el primero en describirlo [37], este fenómeno consiste en una variación drástica en la eficacia de un algoritmo de predicción al aumentar la cantidad de dimensiones. Entrenar un modelo con muy pocas variables puede resultar en un desempeño pobre, pues podrían resultar insuficientes para discriminar entre las clases del problema. A medida

que se añaden variables, alimentando el modelo con más información, la exactitud se incrementa. Pero al continuar adicionando variables y, por tanto, incrementando la cantidad de dimensiones, la exactitud alcanzará un máximo y caerá abruptamente, pues las nuevas variables solo traerán ruido y redundancia al entrenamiento, ante lo cual prácticamente todas las técnicas son sensibles.

En el Capítulo II se abordan los algoritmos más populares basados en reglas dentro del proceso de clasificación, entre ellos está PRISM, Incremental Reduced Error Pruning (IREP), RIPPERk, Ripple down Rules, OneRule y Decision Table Majority (DTM). Además se presenta un nuevo algoritmo de clasificación basado en reglas de cubrimiento para conjunto de datos de alta dimensión y una mejora al algoritmo RIPPERk.

Conclusiones Parciales

Este capítulo recoge los fundamentos teóricos los cuales sirven de base a la presente investigación, a través de los conceptos y definiciones esenciales enmarcadas en el plano teórico del aprendizaje automático supervisado, incluyendo la clasificación basada en reglas como una de las alternativas más populares. Además, se mencionan algunas de las principales aplicaciones que incluyen técnicas de este tipo.

Capítulo II. Clasificadores basados en reglas

En este capítulo es realizada una descripción detallada de los algoritmos de clasificación basados en reglas más destacados. Se presentan las ventajas y deficiencias de cada uno de ellos y su pseudocódigo. Por otro lado se presenta un nuevo algoritmo de clasificación para el trabajo en conjunto de datos de alta dimensión, este algoritmo se caracteriza por la rapidez y precisión.

2.1 One-Rule

One-Rule [38] es un simple y muy efectivo clasificador basado en reglas que genera un nivel en árboles de decisión. OneRule puede inducir de un conjunto de ejemplos reglas de clasificación simple y exacta. Los estudios exhaustivos del rendimiento de OneRule han mostrado que produce reglas ligeramente menos exactas que los esquemas de aprendizaje de punta, mientras produce reglas que son simples de interpretar para los seres humanos [39]. También puede manejar valores faltantes y atributos numéricos, mostrando la capacidad de adaptación a pesar de su sencillez.

En el algoritmo OneRule son creadas varias reglas para cada atributo en el conjunto de entrenamiento, una regla para cada valor del atributo. De esta manera el resultado final es un conjunto de reglas (una regla para cada clase) donde todas estas tendrán el mismo atributo en la parte condicional pero con valores diferentes, y en el consecuente, la clase más frecuente para cada valor respectivo. Luego seleccionar la regla del conjunto de reglas que tenga mayor accuracy en el conjunto de entrenamiento, si hay varias elegir una aleatoriamente.

En caso de tener algún atributo numérico el conjunto de dato, crear una versión nominal del atributo para definir un número finito de intervalos de valores. Por ejemplo si los valores numéricos del atributo A son particionados en tres intervalos la versión nominal de A tendrá tres valores “intervalo 1”, “intervalo 2”, “intervalo 3”.

A pesar de la baja complejidad computacional de este algoritmo, ha sido demostrado empíricamente que resultados similares a los obtenidos por los

algoritmos de aprendizaje de punta pueden ser obtenidos a través de su aplicación en los conjunto de datos más comúnmente usados.

Entrada

E: Conjunto de instancias

Sea N: Conjunto de reglas

1: **Para cada** atributo numérico A

2: Crear una versión nominal de A

3: **Para cada** atributo A

4: **Para cada** valor V_a

5: Contar cuán a menudo aparece cada clase

6: Encontrar la clase más frecuente $C_{f;}$

7: Crear una regla R cuando A= V_a; clase atributo valor = C_f;

8: Añadir R a N;

9: Seleccionar R de N que tenga más alto accuracy p/t en el conjunto de entrenamiento, (Romper empate eligiendo una aleatoriamente)

Figura 2.1-1 Pseudocódigo Algoritmo One-Rule

2.2 Tablas de Decisión (DT)

Las Tablas de Decisión (DT) son una de las formas más simples de representación del conocimiento dentro del ámbito de la clasificación.

Su forma más básica de utilización es el almacenamiento de las ocurrencias de los atributos más relevantes sobre cada una de las clases. Como se muestra en la tabla2.2.1, estas tablas están formadas por un esquema (schema) y un cuerpo (body). El primero se refiere a los valores de los atributos que mejor representen a las clases; mientras que el cuerpo está representado por varios arreglos que contienen conteos de correspondencia entre cada valor de atributo y cada clase. Así que, cada arreglo de conteo será de longitud *t*; donde *t* es el número de clases del conjunto de datos.

Tabla 2.2-1 Tabla de Decisión

Llave (Valores de Atributo)	Valor (Clase)	
	Play?=no	Play?=yes
Outlook=sunny	3	2
Outlook=overcast	0	4
Outlook=rainy	2	3
Humidity=high	4	3
Humidity=normal	1	6

Existen varias formas de clasificar una instancia con clase desconocida una vez construido el modelo o tabla de decisión. Una de estas es el algoritmo DTM (Decision Table Majority) [40] utiliza una tabla de decisión para clasificar instancias con clase desconocida.

Como se muestra en la Figura 2.2-1, en un primer paso (línea 1) realiza la selección del subconjunto de atributos que más distinga a las clases a través de una técnica de selección de atributos definida por el usuario. Luego se crea la tabla (línea 2) realizando el conteo de ocurrencias de cada valor de los atributos seleccionados con cada clase; como se mostró en la sección anterior. Finalmente se clasifica a la instancia con clase desconocida*s* etiquetándola con la clase que más se repite en los valores de atributos análogos en la tabla.

Algoritmo: DTM

Entrada:

$D(A_1, A_2, ..., C)$ // conjunto de datos de entrenamiento

I // instancia con clase desconocida a clasificar

F// función de evaluación para la selección de atributos

S// estrategia de búsqueda para la selección de atributos

Salida: C_c// clase para etiquetar a la instancia I.

»Etapa1: Selección del conjunto de atributos relevantes (body)

1: *Set = Seleccionar_Atributos(F, S, D(A_1, A_2, ...,C))*

»Etapa2: Construcción del modelo (tabla de decisión)

2: Construir una tabla hash donde las llaves sean los valores de atributos de los atributos contenidos en *Set*. Los valores de la tabla hash serán arreglos que contengan la cantidad de ocurrencias que tiene el valor de atributo correspondiente en cada clase

Sea $Table[A_i^j,\ C_k]$ la cantidad de veces que el valor *j* del atributo A_i ocurre en la clase C_k

»Etapa 3: Clasificación

3: Sea Q_k un contador de ocurrencias de varios valores de atributos con la clase *k*

Inicialmente $Q_k = 0$

4: Para cada clase *k*:

5: Para cada atributo A_i en *Set*:

6: Sea A_i^a el valor de atributo en el atributo A_i de la instancia *I*

7: $Q_k = Q_k + Table[A_i^j,\ C_k]$

8: Sea C_c el valor de *k* tal que: $\max_{\forall k}\{Q_k\}$

9: Devolver: C_c

Figura 2.2-1 Algoritmo Decision Table Majority

Es válido notar que la precisión de este clasificador depende en gran medida del proceso de selección de atributos que se realiza en su primera etapa. Es por esto que generalmente se utiliza como función de evaluación para la selección de atributos a la precisión de la propia tabla de decisión utilizando el proceso de validación cruzada. En grandes conjuntos de datos realizar la selección mediante este procedimiento es bastante lento; sin embargo se han ideado técnicas para mejorar la complejidad computacional del proceso de validación cruzada [41][40].

2.3 DTNB

Este algoritmo combina Naive Bayes y las tablas de decisión (DTNB) [42]. Naive Bayes y tablas de decisión pueden ser ambas entrenadas eficientemente. Este divide el conjunto de atributos en dos grupos: un grupo asigna las probabilidades de clase sobre la base de Naive Bayes, el otro sobre la base de tablas de decisión y los resultados de probabilidad estimada son combinados.

Las tablas de decisión: hacen primero un proceso de selección de atributos al conjunto de datos para quedarse con los datos más relevantes

Naive Bayes: clasificador probabilístico que emplea la formula de Bayes para clasificar instancias con clases desconocidas.

$$p(C_k \,/\, v_1, v_2, \ldots, v_n) = \frac{p(C_k)\prod_{i=1}^{n} p(v_i/C_k)}{p(v_1, v_2, \ldots, v_n)}$$

2.4 Algoritmos basados en estrategia de cubrimiento secuencial

La estrategia de cubrimiento secuencial (también conocida como divide y vencerás) es indudablemente la estrategia más explorada y usada para las reglas de inducción de datos. Dado un conjunto de datos conformado por un conjunto de instancias con una clase deseada y un conjunto de instancias con otras clases, el algoritmo de cubrimiento funciona añadiendo secuencialmente, a la regla que está bajo construcción, la prueba atributo-valor que maximice la probabilidad de la clase deseada y minimice la probabilidad de las otras clases.

La calidad de una regla es generalmente evaluada teniendo en cuenta la cobertura, el accuracy y la longitud de la regla. La cobertura de una regla es definida por la fracción (t/t_t) donde t son las instancias que satisfacen el antecedente de la regla y t_t el número de instancias del conjunto de datos, mientras que el accuracy de la regla es la fracción (p/t) donde p representa las instancias cubiertas que pertenecen a la clase consecuente de la regla. Las instancias cubiertas con la clase deseada son instancias positivas p y las otras cubiertas son instancias negativas n.

2.4.1 PRISM

El algoritmo PRISM [43] es para formular reglas, este solamente genera reglas perfectas. Mide el éxito de una regla por la fórmula de accuracy (p/t). Cualquier regla con un accuracy menor que 100% es incorrecta.

Para cada clase, secuencialmente añade a la regla bajo construcción la prueba atributo-valor con más alto accuracy, hasta que el accuracy de la regla ha

alcanzado un valor predeterminado (ejemplo: accuracy igual a uno). Entonces la regla es añadida al conjunto de reglas de la clase y todas las instancias cubiertas por la regla son eliminadas temporalmente. Este proceso es repetido hasta que no haya más instancias con la clase deseada, o hasta que la regla encontrada tenga una tasa de error grande, inaceptable. Lo mismo es hecho para cada una del resto de las clases teniendo en cuenta otra vez el conjunto completo de instancias.

La Figura 2.4-1 Espacio de instancias durante la operación del algoritmo de cubrimiento*[43]*muestra el espacio que contiene todas las instancias, una regla parcialmente construida y la misma regla después que ha sido añadido un nuevo término. El nuevo término restringe la cobertura de la regla: la idea es incluir cuanto más instancias de la clase deseada como sea posible y excluir cuanto más instancias de otras clases como sea posible.

Figura 2.4-1 Espacio de instancias durante la operación del algoritmo de cubrimiento[43]

Entrada

E: Conjunto de instancias

Sea V: valor de un atributo

Para cada clase C

Repetir

Crear una regla R con el antecedente vacio que prediga la clase C

Repetir

Para cada atributo A no mencionado en R y cada valor V

Considerar añadir la condición A=V al antecedente de R

Seleccionar A y V que maximicen el accuracy p/t

(Romper empate eligiendo la condición de mayor p)

Añadir A=V a R

Hasta que R sea perfecta (o no haya más atributos para usar)

Eliminar las instancias cubiertas por R de E

Hasta que E no contenga instancias de C

Figura 2.4-2 Pseudocódigo del algoritmo PRISM

Este algoritmo tiende a un sobre ajuste (overfitting) en los datos de pruebas debido a la función usada para la evaluación de una prueba, por lo que puede producir reglas con una cobertura muy baja. Esta desventaja fue perfeccionada usando un procedimiento Incremental Reduced Error Pruning (IREP).

2.4.2 IREP

El algoritmo de aprendizaje basado en reglas Incremental Reduced Error Pruning (IREP) [44] [43] es una mejora del algoritmo PRISM. En IREP se divide el conjunto de entrenamiento en dos partes: el conjunto de crecimiento y el conjunto de poda con el objetivo de evitar el overfitting en los datos de prueba. El conjunto de crecimiento es utilizado para construir una regla usando el algoritmo de cubrimiento básico. Luego la última prueba agregada es eliminada de la regla, y el efecto es evaluado probando la regla truncada sobre el conjunto de poda para determinar si se desempeña mejor que la regla original. Este proceso de poda se repite iterativamente desde la última hasta la primera prueba. Todo el procedimiento es repetido para cada clase, obteniendo una mejor regla para el conjunto de poda. Esta regla es añadida al conjunto de reglas, las instancias cubiertas son eliminadas del conjunto de entrenamiento (ambos conjunto crecimiento y poda) y el proceso es repetido construyendo y podando una nueva regla.

Por lo general el conjunto de entrenamiento es dividido en tres partes: dos tercios de las instancias son usadas para el crecimiento y un tercio para podar.

La idea de usar un conjunto de poda diferente para podar es llamada Reduced-Error Pruning. La variante descrita anteriormente poda una regla inmediatamente que ha sido crecida y es llamado Incremental Reduced-Error Pruning.

Existen muchas formas distintas de evaluar la utilidad de una regla sobre el conjunto de poda. Una medida simple es considerar cuán bien la regla haría en diferenciar la clase pronosticada de otras clases.

Una regla tiene una tasa total de éxito de $\frac{[\mathrm{p}+(\mathrm{N}-\mathrm{n})]}{T}$ y esta cantidad, evaluado en el conjunto de prueba, es usado para evaluar el éxito de una regla usando reduced-error pruning. Donde p son las instancias positivas, t instancias cubiertas por la regla, n es el número de instancias negativas que la regla cubre, donde $\mathrm{n} = \mathrm{t} - \mathrm{p}$. P es el número de instancias de la clase, T es el número de instancias del conjunto de datos, N es el número total de instancias negativas, $\mathrm{N} = \mathrm{T} - \mathrm{P}$, y n' son instancias negativas no cubiertas por la regla $n' = (\mathrm{N} - \mathrm{n})$.

Esta medida está abierta a la crítica porque trata la no cobertura de instancias negativas tan equitativamente importante como la cobertura de las positivas. Por ejemplo una regla que obtiene $\mathrm{p} = 2000$, $\mathrm{t} = 3000$ y $\mathrm{n} = 1000$, es estimado como más exitosa que una con $\mathrm{p} = 1000$, $\mathrm{t} = 1001$ y $\mathrm{n} = 1$, porque $\frac{[\mathrm{p}+(\mathrm{N}-\mathrm{n})]}{T}$ es $\frac{[1000+\mathrm{N}]}{T}$ en el primer caso, pero solo $\frac{[1+\mathrm{N}]}{T}$ en el segundo. La primera regla es claramente menos predictiva que la segunda, porque esta tiene 33,0% a diferencia de 0,1% de riesgo a ser incorrecto.

Usando la tasa de éxito p/t como medida no es una solución perfecta tampoco, porque esta preferiría una regla que obtuvo una sola instancia correcta ($p = 1$), $\mathrm{t} = 1$ y entonces $\mathrm{n} = 0$, que una regla más útil que consiguió 1000 instancias correctas, de 1001 cubiertas. Otra heurística que ha sido usada es $(p - n)/t$, pero padece exactamente del mismo problema que la anterior porque $\frac{(\mathrm{p}-\mathrm{n})}{t} = 2\,p/t - 1$.

Pseudocódigo del algoritmo IREP

Entrada

E: Conjunto de instancias

Sea N: Conjunto de reglas

Repetir

Dividir E en conjunto de crecimiento y poda a la razón 2:1

Para cada clase C, la cual crecimiento y poda ambas contienen al menos una instancia

Usar algoritmo de cubrimiento básico para crear la mejor regla para la clase C

Calcular la utilidad de la regla U(R) dentro de poda

Calcular la utilidad U(R) con la condición final omitida U(R-) dentro de poda

Repetir

Eliminar la condición final de la regla y repetir paso anterior

Hasta que U(R-) > U(R)

De las reglas generadas, seleccionar la de mayor utilidad U(R)

Añadir R a N

Eliminar las instancias cubiertas por la regla de E

Hasta que E < 3

2.4.3 RIPPERk

Para el aprovechamiento del desempeño de IREP varias modificaciones fueron introducidas a este, y por lo tanto se obtuvo el algoritmo RIPPERk**.**

RIPPERk es la consecuencia de tres principales cambios introducidos al algoritmo IREP[45]:

- Uso de una nueva función de evaluación para valorar una regla en la fase de poda
- Una nueva heurística para determinar cuándo dejar de añadir reglas al conjunto de reglas
- Un procedimiento de optimización global en el conjunto de reglas inducidas

La función usada en la fase de poda es $(p - n)/t$. Esta función busca favorecer sobre las reglas que maximiza las instancias positivas cubiertas y minimiza las instancias negativas cubiertas.

El algoritmo IREP deja de añadir reglas al conjunto de reglas cuando la última regla construida tiene un error que excede al 50% en el conjunto de poda[45]. Esto tiende a dejar de añadir reglas prematuramente cuando el aprende una clase que contiene muchas reglas de baja cobertura. En RIPPERk este problema es enfrentado manteniendo un balance entre el accuracy de las reglas y su description length. Para esto es usado el principio de Minimum Description Length (MDL).

El principio de MDL toma la postura de que la mejor teoría para un cuerpo de datos es una que minimice el tamaño de la teoría más la cantidad de información necesaria para especificar las excepciones en comparación con la teoría[43]. Es aplicable al aprendizaje de máquina de la siguiente manera: dado un conjunto de instancias y un método de aprendizaje que infiera una teoría del conjunto.

Cuando una regla es añadida al conjunto de reglas el *total description length* del conjunto de reglas es calculado, si el *total description length* es más grande en d bits (el tamaño de las estructuras que un esquema aprende puede ser medido en bits de la información) que la más pequeña *description length* obtenida hasta ahora, o no hay más instancias positivas entonces RIPPERk deja de añadir reglas para la clase objetivo. El autor de este algoritmo (William W. Cohen) sugiere $d = 64$ bits. En la ecuación 2.1 se muestra la función empleada para calcular el *total description length* [45]. Observe que k es el número total de condiciones en el conjunto de reglas.

$$S(n, k, p) = k \log_2 \frac{1}{p} + (n - k) \log_2 \frac{1}{1 - p} \qquad (2.1)$$

En aras de mejorar la predicción de IREP, en RIPPERk se llevó a cabo un procedimiento de optimización global sobre el conjunto de reglas inducidas. Este procedimiento reexamina las reglas en el orden en el cual ellas fueron aprendidas. Para cada regla, son creadas dos reglas alternativas: reemplazo y revisión de reglas. La primera alternativa es una regla vacía, cual es crecida y podada de una manera que minimiza el error del conjunto de regla modificado. La segunda es creada del mismo modo, excepto que empieza de la regla

original en lugar de la regla vacía. Para determinar cuál de las tres versiones de la regla conservar, es usado el criterio de MDL. Después los restantes positivos son cubiertos usando el algoritmo IREP.

El algoritmo RIPPERk itera la optimización del conjunto de regla y la cobertura de las instancias positivas restantes con RIPPER k veces.

Pseudocódigo del algoritmo RIPPERk

Entrada

E: Conjunto de instancias

Sea N: Conjunto de reglas

Para cada clase C, de la más pequeña a la más grande
Construir:
Dividir E en conjunto de crecimiento y poda a la razón 2:1
Repetir
Fase de crecimiento: crecer una regla añadiendo condiciones vorazmente hasta que la regla sea 100% precisa probando todos los posibles valores de cada atributo y seleccionar la condición con mayor ganancia de información G
Fase de poda: podar las condiciones en orden de la última a la primera. Continuar mientras la utilidad U de la regla aumente
Hasta que (a) no haya más instancias descubiertas de C; o (b) la *description length* (DL) del conjunto de reglas e instancias es 64 bits mayor que la menor DL encontrada hasta ahora, o (c) la tasa de error excede el 50%
Optimizar:
Generar variantes:
Para cada regla R de la clase C
Dividir de nuevo E en conjunto de crecimiento y poda
Eliminar todas las instancias del conjunto de poda que son cubiertas por otras reglas para C
Usar crecimiento y poda para generar y podar dos reglas compitiendo en la nueva división de datos:
R1 es una nueva regla, reconstruir desde el principio
R2 es generada por añadir antecedentes a R vorazmente
Podar usando la métrica A (en lugar de W) sobre estos datos reducidos
Selección representativa:
Remplazar R por cualquiera de R, R1 y R2 que tenga menor DL
Limpiar:
Si: quedan instancias no cubiertas de la clase C, retornar a la etapa de construcción para generar más reglas basadas en estas instancias
Limpiar a fondo:

Calcular DL para la totalidad del conjunto de regla y para el conjunto de regla con cada regla en turno omitida; eliminar cualquier regla que incremente el DL
Eliminar instancias cubiertas solo por las reglas generadas

Accuracy para esta regla $A = \frac{p+n'}{T}$

2.4.4 Ripple down rules

Ripple down rules es un procedimiento para generar reglas con excepciones [46]. En un experimento realizado con una amplia base de casos médica descubrieron que las personas pueden comprender grandes sistemas de reglas con excepciones más fácilmente que sistemas equivalentes de reglas regulares [43] porque esa es la forma en que ellos piensan sobre los diagnósticos médicos complejos.

El primer paso de este método es seleccionar del conjunto de datos una clase por defecto, por lo general se selecciona la clase que aparece más frecuentemente en los datos de prueba. Luego encontrar una regla para cualquier otra clase que no sea la escogida por defecto. Esta regla debe ser la de mayor poder discriminatorio, ejemplo una con la mejor evaluación en el conjunto de prueba. Esta es usada para dividir los datos de prueba en dos subconjuntos: uno conteniendo todas las instancias para la cual la condición de la regla es “true” y la otra conteniendo los “false”. Si uno y otro subconjunto contiene más de una clase, el algoritmo es invocado recursivamente en el subconjunto.

Para el subconjunto en el cual la condición es “true” la clase por defecto es la nueva clase especificada por la regla, y para el conjunto en el cual la condición es “false” la clase que queda por defecto es la misma de antes. Para el conjunto de instancias que no satisfacen su condición, pertenecen a la clase definida por defecto.

Las reglas que son producidas en este algoritmo tienen como propiedad que la mayoría de los ejemplos son cubiertos por las reglas de mejor evaluación y las de menor poder discriminatorio representan las excepciones. Por lo que una

persona puede solo mirar los primeros niveles e ignorar toda la estructura más profunda, eso es lo especial de las reglas con excepciones.

Entrada

bd: Base de datos a clasificar

1: default_clase: Seleccionar la clase que más se repite;

2: Eliminar default_clase de db;

3: **RippleDownRules** (bd, default_clase);

4: **Si:** $bd == \emptyset$

5: **Terminar;**

6: Aplicar algoritmo de inducción de reglas;

7: Seleccionar la regla con mayor precisión de una clase;

8: Dividir el conjunto de datos en true y false;

9: Eliminar el subconjunto true de db;

10: **Si:** el conjunto de true tiene más de una clase

11: RippleDownRules (true, default_clase);

12: **De otro modo**

13: RippleDownRules (false, default_clase);

Figura 2.4-3 Algoritmo: Ripple down rules

Conclusiones Parciales

En este capítulo se realizó un estudio del estado del arte de los clasificadores basados en reglas. Mostrando sus ventajas y deficiencias lo cual sirve como base para trabajos posteriores. Además se describen dos nuevos algoritmos de clasificación basados en reglas de cubrimiento los cuales se proponen para conjunto de datos de alta dimensión.

Capítulo III. Experimentación

En este Capítulo se efectúa un análisis comparativo de los resultados experimentales obtenidos por los algoritmos de clasificación basados en reglas empleados ante distintos conjuntos de datos. Varios de los conjuntos de datos utilizados son de alta dimensión, se presentan los conjuntos de datos, los algoritmos y cada uno de los resultados obtenidos. De esta forma se pretende alcanzar una mayor visión de los algoritmos en este ámbito.

3.1 Diseño del estudio experimental

En esta investigación se ha preparado un estudio experimental para explorar el comportamiento de los algoritmos de clasificación escogidos, en conjuntos de datos de alta dimensión, y así también probar los nuevos algoritmos de clasificación propuestos en este documento. Todo con el fin de proporcionarle mayor profundidad al estudio del estado del arte de esta investigación.

Se ha utilizado para la ejecución de los experimentos la herramienta Weka [43], desarrollada en el lenguaje de programación Java por un equipo de desarrolladores de la Universidad de Waikato (Nueva Zelanda) bajo licencia GNU (General Public License) y se caracteriza por la independencia de arquitectura, ya que funciona en cualquier plataforma sobre la que haya una Máquina Virtual Java disponible. Permite aplicar, analizar y evaluar algunas de las técnicas más relevantes del análisis de datos, dentro de las que se enmarca: el pre-procesamiento de datos, clasificación, predicción numérica, agrupamiento, reglas de asociación y visualización. Está orientado a la extensibilidad, por lo que es posible añadir nuevas funcionalidades de forma sencilla. Todo lo anterior justifica el porqué se ha convertido una de las herramientas más utilizada en el área.

Es utilizado un esquema de experimentación basado en validación cruzada que garantiza una mayor robustez estadística. Esta propuesta consiste en un procedimiento de validación cruzada con diez particiones con una corrida como lo propone Demsar [47].

Las técnicas estadísticas utilizadas en el procesamiento de los resultados son de tipo no paramétricas. *Friedman* para detectar diferencias significativas globalmente y la *prueba post-hoc de Holm* para detectar diferencias entre pares de algoritmos, como lo propone García y Herrera [48].

Las pruebas no paramétricas utilizadas son:

- Prueba de Friedman [49], para detectar la presencia de diferencias significativas de forma global en todos los algoritmos y obtener un ranking entre los mismos.

- Prueba post-hoc de Holm: se utiliza para detectar diferencias significativas de pares de algoritmos de aprendizaje automático.

Como parámetro de evaluación se emplea:

- Accuracy: Se obtiene del promedio de las diez evaluaciones realizadas al conjunto de datos.

- Tiempo: Tiempo de construcción del modelo de la primera evaluación.

- Número de condiciones de las reglas: Numero de condiciones de las reglas de la primera construcción del modelo del conjunto de datos en cuestión.

Conjunto de datos

Se proponen en calidad de conjuntos de entrenamiento, las bases de datos representadas en la Tabla 3.1-1.

Tabla 3.1-1 Conjunto de datos

Conjunto de datos	Siglas	Atributos	Instancias	Clases
HEPATITIS	HEP	20	155	2
DERMATOLOGY	DER	35	366	6
ARRHYTHMIA	ARR	280	452	13
ADA	ADA	49	4147	2
OPTDIGITS	OPT	65	5620	10
SECOM	SEC	591	1567	2
SYLVA	SYL	217	13086	2
MULTIPLE FEATURES	MFE	650	2000	10
GINA	GIN	970	3153	2
ADS	ADS	1559	3279	2
HIVA	HIV	1618	3845	2
HEPATITISC	HPC	22278	123	4
ARCENE	ARC	10001	100	2
DOROTHEA	DOR	100001	800	2
DEXTER	DEX	20001	300	2
BURKITTLYMPHOMA	BLY	22284	220	3
ISOLET	ISO	618	6238	26

Los conjuntos de datos [50] sobre los cuales se comprueba el funcionamiento de los algoritmos, fueron tomados del repositorio de la Universidad de California en Irvine (UCI), los que se han convertido en estándares, por su utilidad a la hora de comparar resultados entre técnicas de la Minería de Datos.

3.2 Discusión de los resultados

Accuracy

Tabla 3.2-1 Accuracy de los algoritmos de clasificación

DataSet	OneR	Ridor	DT	JRip	JRipO	PART	PAVICD	ZigZag
HEP	83,2258	78,7097	76,1290	78,0645	78,0645	84,5161	81,9355	81,2903
DER	49,7268	93,1694	86,8852	86,8852	87,9781	94,5355	90,7104	91,2568
ARR	57,5221	68,3628	65,7080	70,7965	71,4602	63,9381	70,1327	73,6726
ADA	79,4550	82,8068	84,3019	83,8679	83,8679	83,6991	80,5884	79,8650
OPT	26,9929	90,2313	62,2420	90,7829	90,9786	92,5445	51,8861	54,3594
SEC	92,9802	93,3631	93,2355	92,2782	92,2782	91,6401	93,4907	93,4907
SYL	94,7578	98,8155	99,1212	98,8461	98,8461	99,0448	98,5404	97,4247
MFE	48,3500	93,0500	82,7000	91,8000	92,0500	94,6500	86,4500	83,7500
GIN	71,7412	87,3771	83,4761	88,9312	88,9312	87,9480	78,9724	77,2280
ADS	92,8637	96,7978	95,8829	97,0418	97,0418	96,8588	96,8283	94,3275
HIV	96,6710	96,1769	95,8908	96,4109	96,4109	95,6827	96,4109	96,4889
HPC	65,8537	79,6748	69,9187	67,4797	68,2927	78,8618	80,4878	74,7967
ARC	48,7179	58,9744	58,9744	46,1538	46,1538	43,5897	89,7436	84,6154
DOR	89,0000	90,2500	82,0000	90,0000	90,0000	86,3750	84,8750	88,5000
DEX	71,3333	85,0000	81,1667	85,3333	85,3333	88,1667	90,0000	76,6667
BLY	83,1818	77,2727	76,8182	78,1818	78,1818	80,9091	85,9091	78,6364

ISO	15,2613	53,2500	47,0183	77,2363	77,6531	84,9471	46,6015	48,4450

En la Tabla 3.2-1 se muestran los resultados en cuanto a acurracy de los algoritmos empleados. Siendo PAVICD el algoritmo de mejor resultado con respecto a mejor accuracy por cada conjunto de datos. De manera general (promedio de accuracy en todos los conjuntos de datos) PART, JRipO y PAVICD son los tres mejores respectivamente, por lo que se puede decir que frecuentemente siempre está entre las mejores soluciones.

Tiempo

Tabla 3.2-2 Tiempo de los algoritmos de clasificación

DataSet	OneR	Ridor	DT	JRip	PART	PAVICD	ZigZag
HEP	0,009	0,007	0,097	0,056	0,061	0,03	0,003
DER	0,001	0,038	0,083	0,02	0,01	0,011	0,003
ARR	0,01	2,923	1,32	0,436	0,844	0,027	0,007
ADA	0,009	0,294	0,982	0,352	0,999	0,005	0,005
OPT	0,045	61,868	4,25	4,365	1,508	0,013	0,015
SEC	0,139	0,806	2,83	1,604	3,815	0,051	0,05
SYL	0,252	2,218	76,61	8,14	3,794	0,084	0,085
MFE	0,167	27,888	8,299	7,34	3,723	0,087	0,081
GIN	0,544	9,929	41,426	18,712	22,392	0,19	0,193
ADS	0,346	5,629	133,288	9,033	48,827	0,317	0,316
HIV	0,397	3,473	66,027	6,766	17,337	0,382	0,384
HPC	0,209	4,308	25,881	7,593	3,239	0,253	0,224
ARC	0,023	0,114	3,481	0,194	0,149	0,037	0,042
DOR	62,453	375,757	4849,316	618,373	1026,868	7,409	7,682
DEX	1,299	12,198	141,404	29,917	23,352	0,791	0,802
BLY	0,527	5,225	39,027	13,733	7,931	0,335	0,318
ISO	0,935	345600	72,665	254,462	230,251	0,24	0,239

En la Tabla 3.2-2 se presentan los resultados de los algoritmos de clasificación utilizados en el proceso de experimentación. En el cuál se puede observar como prácticamente los algoritmos PAVICD y ZigZag se reparten los mejores tiempos en todos los conjuntos de datos y con una amplia diferencia de los otros clasificadores, por lo que se puede señalar que son los algoritmos más rápidos en este ámbito de las reglas.

Tabla 3.2-3 Número de condiciones de las reglas de cada algoritmo de clasificación

Algoritmos	OneR	Ridor	DT	JRip	PART	PAVICD	ZigZag
HEP	3	1	135	7	20	5	4
DER	4	11	380	20	19	29	25
ARR	5	534	396	22	156	76	54
ADA	6	144	984	26	1038	10	4
OPT	11	19750	2500	305	423	54	36
SEC	4	1	10	8	47	6	4
SYL	25	20	5317	27	226	40	6
MFE	62	268	804	64	61	43	33
GIN	12	37	4407	77	227	31	9
ADS	23	18	1653	25	97	43	6
HIV	2	5	1800	20	231	17	8
HPC	4	8	84	9	8	11	9
ARC	3	3	18	2	4	5	10
DOR	2	1	1587	1	44	98	6
DEX	4	11	414	25	35	58	8
BLY	3	4	96	7	10	24	18
ISO	139	72	4782	434	633	79	86

En la Tabla 3.2-3 se presenta el número de condiciones de cada regla creada para los distintos conjuntos de datos por los clasificadores empleados. Es utilizado como parámetro de evaluación el número de condiciones de las reglas, y no el número de reglas ya que este nos permite tener una idea de cuán grande pueden ser las reglas creadas por el clasificador. El algoritmo ZigZag y PAVICD ocupan el segundo y tercer lugar respectivamente de manera general. Estos se encuentran muy cercanos al primer lugar que es OneR y los otros clasificadores se hallan bastante alejados de ellos, lo que nos indica que en número de condiciones de las reglas son pocas, es decir su modelo es simple.

Para detectar diferencias significativas en el grupo de algoritmos a evaluar se realizó la prueba de Friedman sobre el accuracy, tiempo y número de condiciones de las reglas alcanzada por cada algoritmo de aprendizaje en cada conjunto de datos.

En la Tabla 3.2-4 se muestra el ranking promedio que alcanzó cada algoritmo. En la Tabla 3.2-4 a) se observa que PART ocupa el primer lugar en el ranking,

luego JRipO y después PAVICD, en último lugar está OneR. En la Tabla 3.2-4 b) se presenta ZigZag, PAVICD y OneR en los tres primeros lugares respectivamente. Los restantes algoritmos están bastante alejados de ellos quedando en último lugar DT. En la Tabla 3.2-4 c) como mejores ubicados se obtienen a OneR, ZigZag y Ridor respectivamente y como último lugar está DT. El p-valor arrojado por la prueba de Friedman para los parámetros de evaluación accuracy, tiempo y número de condiciones fue de 0.0172, 5.107E-11 y 5.453E-10 respectivamente; por lo que se concluye que existen diferencias significativas en los tres parámetros a evaluar para un nivel de confianza de 0.05.

Tabla 3.2-4 Ranking de Friedman para cada parámetro de evaluación

Algorithm	Ranking
PART	3,647059
JRipO	3,764706
PAVICD	4,029412
Ridor	4,029412
JRip	4,147059
ZigZag	4,617647
DT	5,470588
OneR	6,294118

a) Accuracy

Algorithm	Ranking
ZigZag	1,647059
PAVICD	1,970588
OneR	2,500000
Ridor	4,794118
PART	5,235294
JRip	5,382353
DT	6,470588

b) Tiempo

Algorithm	Ranking
OneR	2,05882
ZigZag	2,82353
Ridor	3,02941
JRip	3,88235
PAVICD	4,35294
PART	5,08824
DT	6,76471

c) Número de condiciones

Al presentar diferencias significativas se realiza una prueba post-hoc para detectar diferencias significativas entre todos los pares de algoritmos de clasificación. La prueba utilizada en este estudio es la de *Holm,* es intensivamente estudiada y recomendada en (García y Herrera, 2008).

Tabla 3.2-5.P-valores ajustados que brinda la prueba post-hoc Holm para cada algoritmo en las hipótesis de comparación. Marcado en negrita los p-valores que permiten rechazar sus hipótesis correspondientes con un nivel de significación de 0.05

hypothesis	p_{Holm}
PART vs .OneR	**4,562E-02**
OneR vs .JRipO	7,040E-02
OneR vs .PAVICD	1,827E-01
OneR vs .Ridor	1,827E-01
OneR vs .JRip	2,545E-01
DT vs .PART	6,894E-01
DT vs .JRipO	9,309E-01
OneR vs .ZigZag	9,660E-01

hypothesis	p_{Holm}
ZigZag vs .DT	**1,580E-09**
PAVICD vs .DT	**2,508E-08**
OneR vs .DT	**1,592E-06**
ZigZag vs .JRip	**8,329E-06**
ZigZag vs .PART	**2,178E-05**
PAVICD vs .JRip	**6,614E-05**
PAVICD vsPART	**1,579E-04**
ZigZag vs .Ridor	**3,029E-04**

hypothesis	p_{Holm}
OneR vs .DT	**4,490E-09**
ZigZag vs .DT	**2,087E-06**
Ridor vs .DT	**8,791E-06**
OneR vs .PART	**7,815E-04**
JRip vs.DT	**1,704E-03**
PAVICD vs .DT	**1,815E-02**
OneR vs .PAVICD	**2,941E-02**
ZigZag vs .PART	**3,136E-02**

DT vs .PAVICD	1,726E+00
Ridor vs .DT	1,726E+00
DT vs .JRip	2,073E+00
PART vs .ZigZag	4,216E+00
DT vs .ZigZag	4,960E+00
ZigZag vs .JRipO	4,960E+00
OneR vs .DT	4,960E+00
PAVICD vs ZigZag	6,290E+00
Ridor vs .ZigZag	6,290E+00
JRip vs .PART	6,290E+00
JRip vs .ZigZag	6,290E+00
Ridor vs .PART	6,290E+00
JRip vs .JRipO	6,290E+00
PART vs .PAVICD	6,290E+00
Ridor vs .JRipO	6,290E+00
PAVICD vs .JRipO	6,290E+00
JRip vs .PAVICD	6,290E+00
Ridor vs .JRip	6,290E+00
PART vs .JRipO	6,290E+00
Ridor vs .PAVICD	6,290E+00

a) Accuracy

OneR vs .JRip	**1,303E-03**
PAVICD vs .Ridor	**1,663E-03**
OneR vs .PART	**2,452E-03**
OneR vs .Ridor	**1,961E-02**
Ridor vs .DT	2,130E-01
DT vs .PART	7,639E-01
DT vs .JRip	9,934E-01
ZigZag vs .OneR	1,498E+00
Ridor vs .JRip	2,136E+00
PAVICD vs .OneR	2,136E+00
Ridor vs .PART	2,136E+00
ZigZag vs .PAVICD	2,136E+00
JRip vs .PART	2,136E+00

b) Tiempo

Ridor vs .PART	7,097E-02
OneR vs .JRip	1,662E-01
DT vs .PART	2,603E-01
ZigZag vs PAVICD	3,901E-01
Ridor vs .PAVICD	6,665E-01
JRip vs .PART	8,291E-01
ZigZag vs .JRip	1,071E+00
OneR vs .Ridor	1,141E+00
Ridor vs .JRip	1,248E+00
OneR vs .ZigZag	1,248E+00
PAVICD vs .PART	1,248E+00
JRip vs .PAVICD	1,248E+00
ZigZag vs .Ridor	1,248E+00

c) Número de condiciones

Como se puede apreciar en la Tabla 3.2-5 a) utilizando como parámetro el accuracy, solo se detecta diferencias significativas entre PART y OneR, donde en la Tabla 3.2-4 a) que se presentan el ranking, PART aparece como primero y OneR como último. En la Tabla 3.2-5 b) tomando como parámetro al tiempo, es detectada diferencias significativas de ZigZag, PAVICD y OneR con el resto de los algoritmos, mientras que entre ellos no se detecta diferencias significativas. Finalmente en la Tabla 3.2-5 c) utilizando como parámetro el número de condiciones de cada regla es detectada diferencias significativas entre OneR y DT, PART y PAVICD; también ZigZAg con DT y PART. Luego Ridor, JRip y PAVICD presentan diferencias significativas con DT.

En la Figura 3.2-1 se muestra una gráfica con los clasificadores evaluados y se comparan con los parámetros seleccionados anteriormente (precisión, tiempo, número de condiciones de las reglas) a través del test de Nemenyi. Se puede observar como los algoritmos implementados no poseen diferencias significativas con el resto de los clasificadores evaluados, en cuanto a los parámetros tomados.

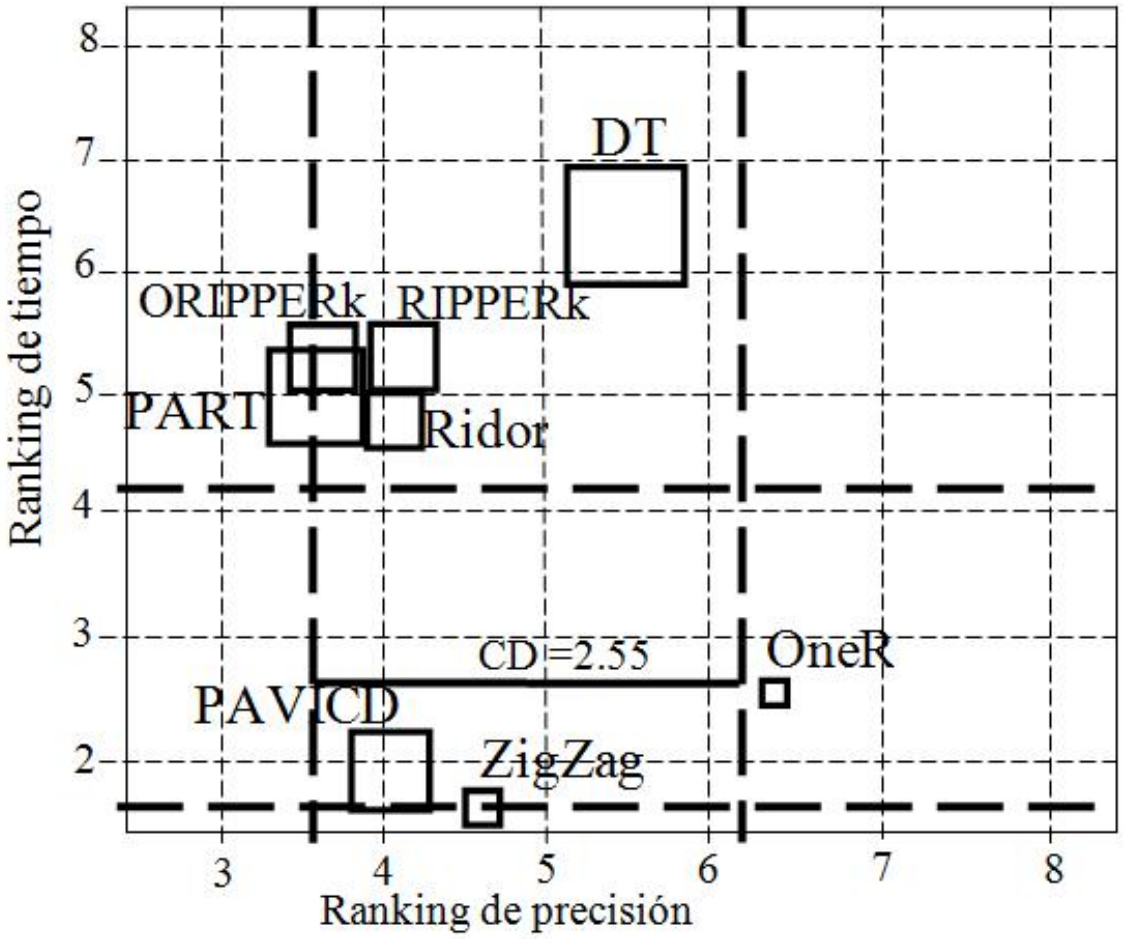

Figura 3.2-1 Ranking de precisión, tiempo y número de condiciones (Test Nemenyi)

Conclusiones Parciales

Definitivamente, para los conjuntos de datos y algoritmos de aprendizaje utilizados en este estudio, el clasificador *PART* es el que presenta mejor accuracy pero solo mostrando diferencias significativas con OneR. Mientras que en segundo lugar quedó JripO que es una mejora al orden de las reglas de Jrip y luego PAVICD. Sin embargo, en cuanto a tiempo ZigZag, PAVICD y OneR fueron los de mejor tiempo respectivamente y presentaron diferencias significativas con el resto de los clasificadores. En cuanto al número de condiciones de las reglas OneR, ZigZag y Ridor fueron los mejores reduciendo.

Por lo que se puede decir que PAVICD no es el mejor en accuracy pero si está entre los mejores en tiempo y condiciones de reglas, al contrario de PART. Convirtiéndose PAVICD en un algoritmo competente para conjunto de datos de alta dimensión.

Conclusiones

En la actualidad la creciente dimensión de los datos se ha convertido en un reto para los investigadores del campo de la minería de datos. Por lo que el rendimiento de los algoritmos de aprendizaje es afectado por el ruido, la redundancia intrínseca en estos conjuntos de datos y la inconsistencia.

- Se describieron los algoritmos más reconocidos en el área de estudio de los clasificadores basados en cubrimiento secuencial.

- Se implementó el algoritmo PAVICD y ZigZag y se mejoró el algoritmo RIPPERk.

- Se realizó una experimentación de los algoritmos de clasificación detallados en este estudio.

- Los algoritmos más rápidos resultaron ser PAVICD Y ZigZag y mostraron su competitividad en cuanto a la precisión alcanzada.

- Se constató que el algoritmo PART supera significativamente a OneR en cuanto a precisión.

- El algoritmo Decision Table genera significativamente mayor número de condiciones con respecto al resto de los algoritmos (exceptuando PART).

Recomendaciones

- Estudiar y describir otros algoritmos basados en cubrimiento secuencial.
- Extender este estudio experimental utilizando otros conjuntos de datos.
- Mejorar los algoritmos PAVICD y ZigZag de manera que obtengan reglas con mayor precisión.

Referencias bibliográficas

[1] A. Konar, “Artificial Intelligence and Soft Computing. Behavioral and Cognitive Modeling of the Human Brain,” *CRC Press LLC*, 2000.

[2] D. François, “High-dimensional data analysis: optimal metrics and feature selection,” *Université Catholique de Louvain. Faculté des Sciences Apliquées*, 2007.

[3] E. Frank and M. A. Hall, *Data mining. Practical Machine Learning Tools and Techniques*. 2011.

[4] T. M. Mitchell, *Machine Learning*. 1997.

[5] U. Fayyad, “The KDD Process for Extracting Useful Knowledge from Volumes of Data,” vol. 39, no. 11, pp. 27–34, 1996.

[6] R. M. Gray and L. D. Davisson, *An Introduction to Statistical Signal Processing*, vol. 1. 2000.

[7] S. Sumathi, *Fundamentals of Relational Database*. 2006.

[8] S. J. Russel and P. Norvig, *Artificial Intelligence. A modern Approach*. 2003.

[9] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, “From Data Mining to Knowledge Discovery in Databases,” vol. 17, 1996.

[10] K. J. Cios and L. A. Kurgan, “1 . Trends in Data Mining and Knowledge Discovery,” no. Dm, pp. 1–26.

[11] D. T. Larose, *An Introduction to Data Mining*. .

[12] R. R. Bouckaert, E. Frank, M. Hall, R. Kirkby, P. Reutemann, A. Seewald, and D. Scuse, “WEKA Manual for Version 3-7-4,” 2011.

[13] “keel,” *http://sci2s.ugr.es/keel.* .

[14] “Orange – Data Mining Fruitful & Fun,” *http://www.ailab.si/orange*. .

[15] P. C. Ncr, J. C. Spss, R. K. Ncr, T. K. Spss, T. R. Daimlerchrysler, C. S. Spss, and R. W. Daimlerchrysler, “Step-by-step data mining guide,” pp. 1–78.

[16] M. A. Hall, “Correlation-based Feature Selection for Machine Learning,” no. April, 1999.

[17] A. Pino, “Estudio experimental de algoritmos de Selección de Atributos para la Minería de Datos,” 2009.

[18] F. Herrera and J. . Cano, “Algoritmos Evolutivos para la Selección de Instancias.”

[19] R. Ruiz Sanchez, “Selección de Atributos mediante proyecciones,” Universidad de Sevilla, 2005.

[20] A. L. Bluma and P. Langley, “Artificial Intelligence Selection of relevant features and examples in machine,” vol. 97, no. 97, pp. 245–271, 1997.

[21] M. Oded and R. Lior, *DATA MINING AND KNOWLEDGE DISCOVERY HANDBOOK*. 2010.

[22] J. M. Font Fernández, “GENERACIÓN DE SISTEMAS BASADOS EN REGLAS MEDIANTE PROGRAMACIÓN GENÉTICA,” 2008.

[23] A. Webb, *Statistical Partten Recognition*. 2002.

[24] Y. Campos Hidalgo, “Minería de datos para la predicción en entornos de gestión que manejan datos de alta dimensión : un estudio experimental.”

[25] C. G. Figuerola, J. L. A. Berrocal, and A. F. Zazo, “Clasificación Automática de Documentos,” in *Universidad de Salamanca*, pp. 1–23.

[26] E. Plaza, “Tendencias en Inteligencia Arti cial hacia la cuarta década.,” *In A. del Moral*, no. Nuevas tendencias en Inteligencia Artificial, pp. 379–415.

[27] A. Moreno, E. Armengol, J. Béjar, L. Belanche, U. Cortés, R. Gavaldà, M. Gimeno Juan, B. López, M. Martín, and M. Sànchez, *Aprendizaje automático*, Edicions U. 2006.

[28] I. F. Blanco, A. O. Díaz, G. R. Jiménez, R. M. Bueno, and Y. C. Mota, “CLASIFICADORES Y MULTICLASIFICADORES CON CAMBIO DE CONCEPTO BASADOS EN ÁRBOLES DE DECISIÓN,” vol. 45, pp. 32–43, 2010.

[29] P. Langley and H. A. Simon, “Applications of Machine Learning and Rule Induction,” *Communications of the ACM*, vol. 38, no. 11, p. 55, 1995.

[30] “Data Mining Server. Rule Induction Methods,” *http://dms.irb.hr/tutorial/tut_rinduct_meth.php*, 2007. .

[31] C. Apte, E. Grossman, E. Pednault, B. Rosen, F. Tipu, and B. White, “Insurance Risk Modeling Using Data Mining Technology,” 1999.

[32] S. Johnston, “Applications of Rule Induction,” 2007.

[33] H. Liu and C. Science, “Efficient Rule Induction from Noise Data,” vol. 10, p. 2, 1996.

[34] C. Rao, “The utilisation of multiple measurements in problems of biological classification,” *Journal of the Royal Statistical Society*, vol. 10, pp. 159–203, 1948.

[35] R. Bellman, “Adaptive control processes: A guided tour,” *Princeton University Press*, 1961.

[36] H. Ahn, H. Moon, M. J. Fazzari, N. Lim, J. J. Chen, and R. L. Kodell, “Classification by ensembles from random partitions of high-dimensional data,” *Computational Statistics & Data Analysis*, vol. 51, p. 6166, 2007.

[37] G. Hughes, “On the mean accuracy of statistical pattern recognizers,” *Information Theory, IEEE Transactions on*, vol. 14, no. 1, pp. 55–63, 1968.

[38] R. C. Holte, “Very Simple Classification Rules Perform Well on Most Commonly Used Datasets,” no. 1988, pp. 63–91, 1993.

[39] G. Buddhinath and D. Derry, “A Simple Enhancement to One Rule Classification.”

[40] R. Kohavi, “The Power of Decision Tables,” *European Conference on Machine Learning (ECML)*, 1995.

[41] A. Moore and M. Lee, “Efficient algorithms for minimizing cross validation error,” *Procedings of the Eleventh International Conference, Morgan Kaufmann*, 1994.

[42] M. Hall and E. Frank, “Combining Naive Bayes and Decision Tables,” pp. 2–3, 2008.

[43] I. H. Witten and E. Frank, *Practical Machine Learning Tools and Techniques*. 2005.

[44] J. Furnkranz and G. Widmer, “Incremental Reduced Error Pruning,” 1993.

[45] W. W. Cohen, “Fast Effective Rule Induction,” *Machine Learning: Proceedings of the Twelfth International Conference*, 1995.

[46] B. R. Gaines and P. Compton, “Induction of Ripple-Down Rules Applied to Modeling Large Databases.”

[47] J. Demsar, “Comparison of Classifiers over Multiple Data Sets,” *Journal of Machine Learning Research*, vol. 7: 31, 2006.

[48] S. García and F. Herrera, “An Extension on ‘Statistical Comparisons of Classifiers over Multiple Data Sets’ for all Pairwise Comparisons,” *Journal of Machine Learning Research*, pp. 2677–2694, 2008.

[49] Friedman and Milton, “The use of ranks to avoid the assumption of normality implicit in the analysis of variance,” *American Statistical Association*, 2006.

[50] D. Newman, S. Hettich, C. Blake, and M. C. J, “UCI REPOSITORY OF MACHINE LEARNING DATABASES. University of California, Irvine, Department of Information and Computer Sciences,” 1998. [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html .