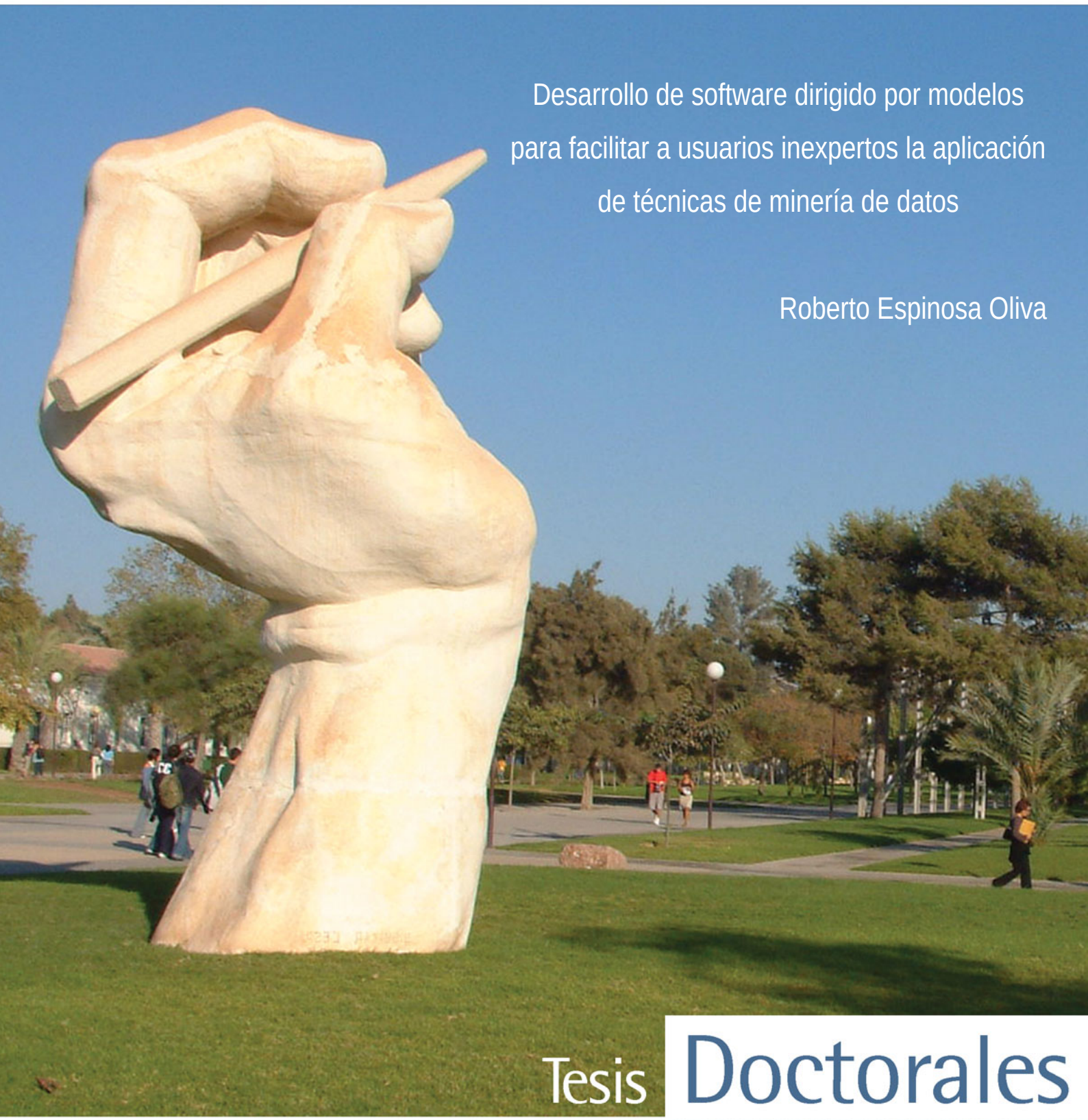
Universitat d'Alacant
Universidad de Alicante



Desarrollo de software dirigido por modelos para facilitar a usuarios inexpertos la aplicación de técnicas de minería de datos

Roberto Espinosa Oliva

Tesis Doctorales

www.eltallerdigital.com

UNIVERSIDAD de ALICANTE

Universidad de Alicante

Instituto Universitario de Investigación Informática

Tesis Doctoral

Desarrollo de software dirigido por modelos para facilitar a usuarios inexpertos la aplicación de técnicas de minería de datos

---

Autor: Roberto Espinosa Oliva

Directores: Jose Norberto Mazón López, José Jacobo Zubcoff Vallejo

Tesis presentada para optar

al grado de Doctor en Informática

Grupo de Investigación WaKe (Web and Knowledge)

Departamento de Lenguajes y Sistemas Informáticos

Universidad de Alicante

noviembre 2014

Resumen

La sociedad en la que vivimos nos ha convertido en entes inseparables de la tecnología. Usamos a diario multitud de dispositivos como teléfonos móviles inteligentes y ordenadores portátiles, así como infinidad de aplicaciones como redes sociales, buscadores, sistemas de comercio electrónico, etc. Esta interacción con la tecnología hace que, en nuestra vida diaria, produzcamos y consumamos cantidades ingentes de datos (por cierto, no sólo en nuestras tareas profesionales sino también cotidianas). Valga expresar un ejemplo concreto:

Un ciudadano desea tomar disminuir el consumo eléctrico en su hogar. Si tuviera una aplicación que utilizara los datos de consumo energético, equipos funcionando, costo de kilo watts por día, humedad relativa, temperatura, en toda una población, o a nivel de país, pudiera llegar a saber que la lavadora-secadora genera un costo significativo, siempre que se utiliza durante el período de carga máxima. Por lo que debería encenderla al final de la noche [1]. Desafortunadamente, esta cantidad de datos no se aprovecha para realizar una toma de decisiones informada en nuestra vida diaria (es decir, fundamentadas en conocimiento extraído de los datos disponibles).

El problema está precisamente en que la explotación de los datos para conseguir extraer conocimiento de los mismos no es una tarea tan sencilla para cualquier persona, más bien resulta una tarea bastante complicada, y ya que se requiere tener experiencia en conceptos estadísticos y en algoritmos de minería de datos, lo que está reservado a personas expertas (los llamados científicos de datos o, en inglés, *"data scientists"*). Este hecho que establece la causa de la brecha entre los datos y las acciones a tomar por los usuarios inexpertos, es lo que se conoce como *"Big Data Divide"*.

[1] http://es.slideshare.net/apsheth/smart-data-how-you-and-i-will-exploit-big-data-for-personalized-digital-health-and-many-other-activities

En el marco de esta tesis doctoral, se plantea desarrollar una propuesta para lograr facilitar el uso de técnicas de minería de datos (o análisis de datos), específicamente técnicas de clasificación, a usuarios inexpertos. El objetivo es posibilitar a estos usuarios la explotación de los datos que tengan disponibles para que puedan extraer conocimiento de ellas de forma fácil y rápida, sin la presencia de un experto.

Esta propuesta usa técnicas de desarrollo de software dirigido por modelos con el fin de homogeneizar y automatizar el proceso de aplicación de técnicas de minería de datos por parte de usuarios inexpertos. Las contribuciones de nuestra propuesta se muestran a continuación:

- Se ha diseñado una base de conocimiento que permite almacenar toda la información que se genera en el proceso de extracción de conocimiento por usuarios expertos.
- El modelo de minería que se obtiene como respuesta al usuario inexperto es obtenido teniendo en cuenta la calidad de sus datos, al ser demostrado su incidencia en los resultados cuando se aplican técnicas de minería.
- Este resultado es obtenido al aplicar el recomendador construido con vistas a obtener el mejor algoritmo a aplicar sobre las fuentes de datos de entrada del usuario inexperto.
- El recomendador construido utiliza los datos almacenados en la base de conocimiento.
- Como elemento importante se ha tenido en cuenta los requerimientos de los usuarios inexpertos para brindarle la solución que mejor satisfaga sus expectativas.
- Un conjunto de experimentos han sido realizados para validar la viabilidad de nuestra propuesta.

En definitiva, en un mundo *"Big Data"* es necesario contar con mecanismos que nos permitan sacar provecho de la cantidad de datos disponibles. Nuestra propuesta pretende ser uno de estos mecanismos, orientada a la democratización en el uso de la minería de datos, facilitando la obtención de conocimiento y, por ende, una toma de decisiones más informada a todas las personas por igual, independientemente de su nivel de experiencia.

Agradecimientos

Luego de tantos años de esfuerzo y perseverancia son muchas las personas que me han apoyado para poder alcanzar esta codiciada meta. Pretenderé agradecer mediante estas líneas a todas ellas, aunque el espacio no me permita mencionar todos sus nombres. En primer lugar agradecer a mi familia, su apoyo incodicional bajo cualquier circunstancia me ha dado las fuerzas necesarias para seguir adelante. Durante todo el proceso de mi formación fueron muchos los profesores que aportaron su grano de arena para convertirme en la persona que actualmente soy, debo agradecer especialmente a dos personas importantes: Josefina Rabaza e Ismael Castillo. En el plano personal, me siento satisfecho de contar con los amigos que tengo, su apoyo desinteresado ha contribuido a la obtención de estos resultados.

Esta investigación nunca hubiera sido lograda sin la ayuda y la orientación que desde el inicio me brindaron mis dos tutores, incluso sin conocerme, a ellos mi agradecimiento infinito por ser tan comprensibles y pacientes en todo momento. Finalmente, agradecer a mis compañeros de trabajo del Departamento de Informática de la Universidad de Matanzas, al personal del Departamento de Lenguajes y Sistemas Informáticos de la Universidad de Alicante, y a los miembros del Grupo de Investigación Wake. Esta tesis esta dedicada a Yuniesky Zamora Galbán, un doctor en potencia a quien la vida lo privó de cumplir sus sueños....

Índice general

Lista de Figuras

Lista de Tablas

A mis Padres.....

Capítulo 1

Introducción

En los últimos años el aumento del uso de las tecnologías de la información y la comunicación (aplicaciones Web, dispositivos móviles, redes sociales, etc.) ha propiciado un crecimiento exponencial de los datos que se generan [1]. Además, muchas instituciones públicas proveen acceso fácil y libre a muchos de estos datos para potenciar su reutilización dentro de la filosofía de datos abiertos [1]. Esta gran disponibilidad de datos está haciendo posible que la ciudadanía se interese por la posibilidad de realizar un análisis de los mismos para llevar a cabo una toma de decisiones informada, mejorando así su vida diaria en diversos ámbitos (profesional, vida familiar o momentos de ocio) [2].

El objetivo es que la ciudadanía pueda tomar decisiones mejores fundamentadas en su vida diaria. A continuación se muestran varios ejemplos:

- Un profesor universitario pudiera preguntarse, teniendo los datos de la interacción de sus alumnos con los entornos de aprendizaje virtuales, como se corresponde la actividad durante el curso con la evaluación final de cada estudiante
- Biólogos que estudian el hábitat marino, quisieran conocer que variables tienen mayor incidencia en las poblaciones de cada especie
- Un gerente de un hotel quisiera estudiar el comportamiento de sus clientes de acuerdo a sus gastos, para identificar clientes potenciales

[1] http://www.opendatafoundation.org/

Sin embargo, la mayoría de estas situaciones no son realistas para la ciudadanía (de manera general), ya que no posee los conocimientos ni las habilidades para analizar los datos. Esta situación ya ha sido planteada por varios expertos [3], declarando explícitamente la necesidad de implementar mecanismos que permitan transformar datos crudos en conocimiento, debido a la gran cantidad de tipos de usuarios que cada día están relacionados con el proceso de generar, procesar y consumir los datos.

De hecho, se puede llegar a provocar un serio problema que ya ha sido denominado como "Big Data Divide" [4, 5]. Este término aborda precisamente la dificultad de contar con grandes cantidades de datos, pero sólo un selecto grupo de personas (por ejemplo: expertos en minería de datos, estadísticos, etc.), por sus habilidades (vasto conocimiento de técnicas de extracción de conocimientos, algoritmos de minería de datos, etc.), pueden extraer conocimiento de ellos.

Llegados a este punto cabe diferenciar bien tres términos que se usan en muchas ocasiones como sinónimos, aunque realmente no lo son: datos, información y conocimiento. El uso de estos tres términos no es consistente y resulta a menudo, contradictorio. Los datos y la información son frecuentemente intercambiables en la informática (por ejemplo, procesamiento de datos y procesamiento de la información o la gestión de datos y la gestión de información).

Existen varias definiciones en la literatura para cada uno de estos términos, teniendo en cuenta que los utilizaremos frecuentemente durante este documento, es conveniente introducir sus definiciones, tomadas de Bellinger [6]:

- Los datos representan un hecho o evento sin relación con otras cosas. Ejemplo: Está lloviendo.
- La información encarna la comprensión de una relación de algún orden, posiblemente causa y efecto. Ejemplo: La temperatura descendió 15 grados y luego empezó a llover.
- El conocimiento representa un patrón que conecta y generalmente proporciona un alto nivel de previsibilidad en cuanto a lo que se describe o lo que va a ocurrir a continuación. Ejemplo: Si la humedad es muy alta y la temperatura cae sustancialmente, es poco probable que la atmósfera sea capaz de mantener la humedad, de modo que llueve.

Es una realidad que los datos, pueden ser públicos, a la vista de todos, pero en la mayoría de los casos el conocimiento no está explícito, y requiere de un procesamiento adicional para extraerlo.

La obtención de conocimiento es posible a partir de la aplicación del denominado, proceso de extracción de conocimiento, del inglés *Knowledge Discovery in Databases (KDD)*. Este proceso está formado por un conjunto de etapas, entre las cuales resalta la minería de datos, ya que provee los mecanismos para la búsqueda de patrones que no son perceptibles a simple vista. El uso de técnicas de minería de datos es sumamente importante para que los usuarios inexpertos puedan descubrir conocimiento con menos esfuerzo. La realización de este proceso es sumamente complejo, siendo habitualmente ejecutado solamente por aquellos usuarios expertos en técnicas de análisis de datos.

De hecho, aquellos ciudadanos inexpertos, pero ansiosos de conocimiento, están abrumados porque no son conscientes de las técnicas que existen para analizar los datos y obtener conocimiento. Es evidente que la enorme cantidad de datos disponibles es inmanejable para la gran mayoría de personas, y la necesidad de contar con mecanismos que les permitan generar conocimiento al analizar los datos sería altamente beneficioso.

Los datos disponibles deben ser analizados no sólo por estadísticos o mineros de datos que trabajan en grandes compañías, sino también por ciudadanos comunes.

Existe una brecha entre la gran cantidad de datos disponibles y la cantidad de datos que se usan en la toma de decisiones. Estadísticas del año 2012, exponen que sólo el 23 % de los datos pudieran ser útiles, si estos estuvieran etiquetados y analizados. La realidad expone que sólo son analizados el 0,5 % del total de datos existentes [1]. Otros estudios afirman que la mayoría de las empresas estiman que están analizando sólo el 12 % de los datos que poseen [7].

El propósito de este tesis doctoral es, a grandes rasgos, reducir la brecha existente entre las dificultades que presentan los usuarios inexpertos al interactuar con los datos existentes, y la posibilidad de extraer conocimiento de ellos en aras de tomar decisiones mejores fundamentadas.

A continuación se describirá la situación problemática existente, y luego se hace necesario la introducción de un conjunto de conceptos, para que el lector pueda

seguir el hilo conductor de esta investigación. Finalmente, se presenta la hipótesis de partida y los objetivos planteados.

1.1. Situación problemática

La minería de datos se erige como una solución prominente para descubrir patrones de conocimiento, y de esta forma explotar mejor los datos que se poseen. Sin embargo, tradicionalmente la aplicación de técnicas de minería se ha considerado un proceso intrínsicamente complejo [8], [9] en el cual (i) se pueden aplicar un gran número de algoritmos para resolver el mismo problema con diferentes resultados, y (ii) la aplicación correcta de técnicas de minería de datos siempre requiere un gran esfuerzo manual para la preparación de los conjuntos de datos de acuerdo a su calidad. Generalmente es necesaria la presencia de un experto con los conocimientos básicos para culminar estas tareas.

Ante la gran disponibilidad de datos anteriormente mencionada, resulta imprescindible democratizar el análisis de los datos, o lo que es lo mismo lograr una minería de datos para todos para evitar el *"Big Data Divide"*. La democratización de la minería de datos, requiere confiar en los conocimientos acerca de que técnicas de minería de datos y parámetros de configuración son adecuados para ser aplicados a las fuentes de datos en función de su calidad. La minería de datos amigable [10] es un paso para lograr dicha democratización, ya que fomenta el descubrimiento de conocimiento sin la necesidad de poseer dominio de los conceptos fundamentales de minería de datos.

Actualmente el proceso de obtención de conocimiento es llevado a cabo fundamentalmente por expertos en esta rama. Muchos son los factores que inciden para lograr que un usuario inexperto pueda utilizar de manera fácil y eficiente mecanismos para extraer conocimiento, siendo éste un problema aún sin resolver. Algunos de los temas más importantes, para que se tenga una idea de la complejidad del proceso, son:

- definir objetivos.
- encontrar la técnica de minería para dar respuesta a un objetivo.
- tener en cuenta la diversidad de formatos de datos existentes.

- hacer una limpieza de los datos.
- elegir los atributos que participarán en el análisis.
- configurar los parámetros.
- tener en cuenta la influencia de criterios de calidad y meta-características de los datos para la obtención de conocimiento fiable.
- aplicar el algoritmo adecuado.
- saber interpretar los resultados.

En general existen varios aspectos que deben mejorarse para lograr la aplicación de técnicas de minería por usuarios inexpertos. Para lograr que un usuario inexperto pueda obtener conocimiento fiable, es necesario implementar un mecanismo que permita fácilmente obtener las expectativas del usuario al desear analizar un conjunto de datos, e interpretarlas, en aras de poder aplicar el mejor algoritmo de minería de datos posible, teniendo en cuenta la calidad de las fuentes de datos analizadas. De manera general, no se encuentran propuestas que integren esos aspectos enfocadas a estos tipos de usuarios.

1.2. Proceso de descubrimiento de conocimiento

El descubrimiento de conocimiento en bases de datos fue definido por Fayyad et al en 1996 como "el proceso no trivial de identificación de patrones válidos, noveles, potencialmente útiles y, en definitiva, comprensible en los datos" [11]. En la Fig. 1.1 se muestra el ciclo de desarrollo del proceso de descubrimiento de conocimiento. Este proceso se puede agrupar en tres grandes etapas: integración y preprocesado de los datos en un repositorio único, selección de atributos y algoritmos para la minería de datos, y finaliza con el análisis e interpretación de los patrones resultantes.

Todas las fases de este proceso son altamente dependientes de la anterior. El éxito de la etapa de análisis radicará en que los atributos y los algoritmos sean seleccionados de una manera adecuada. A su vez, esta fase depende de la fase de integración de los datos, donde también debe procurarse eliminar cualquier problema que afecte la calidad de los datos. Como se puede observar, este proceso

es cíclico, cualquier fase puede aportar más información que se puede incorporar al repositorio de datos. A continuación se describe brevemente cada una de las fases que componen este proceso.

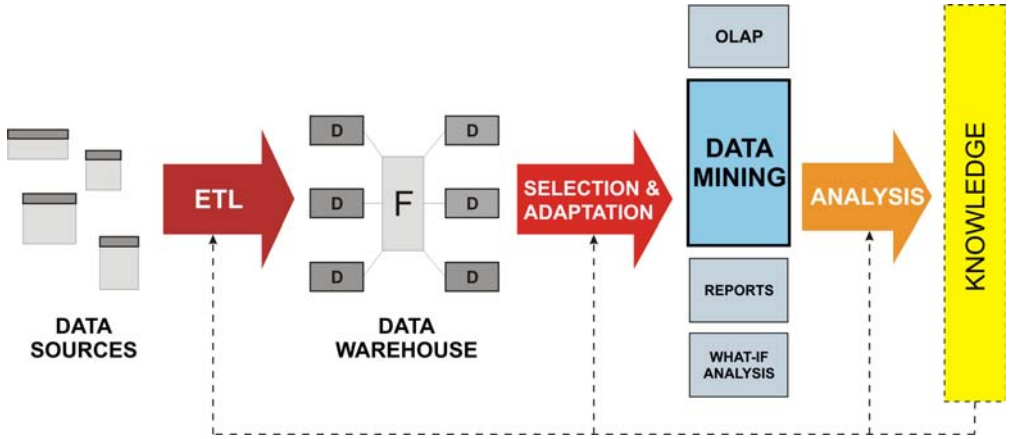


FIGURA 1.1: Proceso de descubrimiento de conocimiento

1.2.1. Integración y preprocesado de los datos

Siendo la etapa inicial dentro del proceso KDD, tiene como objetivo integrar de manera única toda la información necesaria para analizarse posteriormente. En muchas ocasiones la información a analizar se encuentra en múltiples fuentes de datos, por lo que esta etapa debe dar lugar a la creación de un único repositorio (o almacén de datos) de información que contenga los datos limpios y preparados (entre las operaciones que se le aplican a los datos en esta etapa se encuentran: detección de duplicados, eliminación de los mismos, entre otros) para el análisis. Esta fase es la que más tiempo consume de todas las fases del proceso de búsqueda de conocimiento, entre el 70 % y el 90 % [10], [12], [13].

1.2.2. Minería de datos

Esta fase se considera como el núcleo del descubrimiento del conocimiento. Su objetivo es producir nuevo conocimiento que pueda utilizarse por los usuarios a partir de los datos preprocesados por la fase anterior. Esta fase agrupa varias tareas a realizar:

1. Selección de los datos que participarán en el análisis. Un vago conocimiento de los datos que participan puede llevar a resultados erróneos o incomprensibles.

2. Tipo de tarea más apropiada de acuerdo a los patrones que se generan. A continuación, se muestra una breve descripción de las principales tareas:

   a) *Agrupamiento:* son las técnicas que permiten identificar un grupo finito de categorías o *clusters* para definir los datos [14]. Este tipo de técnica puede ser utilizada por ejemplo para caracterizar una enfermedad a partir de un grupo de síntomas.

   b) *Asociación:* las reglas de asociación permiten descubrir relaciones o correlaciones interesantes en grandes cantidades de datos. El objetivo es descubrir patrones en forma de reglas que representen la asociación encontrada [15]. El ejemplo más citado es la utilización de reglas de asociación en el análisis de la cesta de compra.

   c) *Análisis de series temporales:* los datos que siguen una secuencia a lo largo del tiempo, y por lo tanto, pueden ser ordenados cronológicamente, constituyen una serie temporal [16]. A partir de analizar el rendimiento en determinadas temporadas de un jugador de baloncesto pudiera estimarse los parámetros que podría tener en la próxima temporada.

   d) *Clasificación:* Teniendo en cuenta que durante la investigación nos centraremos específicamente en este tipo de técnica, haremos énfasis en su descripción.

   Las técnicas de clasificación son una de las más utilizadas en el mundo de la minería de datos. Estas técnicas analizan un conjunto de datos de entrada y a partir de ellos construyen las correspondientes clases basado en las características de los datos [17]. La idea es establecer una regla donde el analista pueda clasificar cada nueva observación en una clase existente. Por ejemplo, clasificar la posición de un jugador de baloncesto de acuerdo a su estatura.

   Dado un conjunto de datos D con las instancias I representadas cada una en un archivo, con un conjunto de atributos A (A1, A2, ...) y un atributo objetivo C (ambos A y C representados como columnas), también conocido como atributo de clase o etiqueta de clase, con un

conjunto de posibles valores discretos (c1, c2, ...) nombrados clases, el objetivo de un proceso de clasificación es, utilizando un algoritmo de clasificación de minería de datos, relacionar los valores de los atributos A1, A2, etc y las clases c1, c2, etc. Esto significa que, utilizando un algoritmo de clasificación con un conjunto de datos D se obtendrá un modelo de predicción o clasificación que infiere o predice el valor correcto de la etiqueta de la clase C, debido a los valores del resto de los atributos A [18].

Una vez que el modelo de clasificación es obtenido, tiene que ser evaluado y validado con el fin de determinar a qué nivel este modelo es bueno para hacer predicciones con nuevas instancias acerca de la clase. Esto significa, cuan bueno es este modelo con el fin de predecir el valor correcto de la clase C de un nuevo conjunto de instancias, llamado conjunto de datos de prueba o conjunto de pruebas, dado el valor de sus atributos A1, A2, etc.

Hay un cúmulo de medidas para validar un modelo de clasificación. Nos centramos en una de las medidas más utilizada y fácil de entender, la precisión, es un porcentaje calculado mediante el número de casos que han sido correctamente clasificados (las instancias con el modelo de clasificación predice el valor de la clase correcta), dividido por el número total de instancias en el conjunto de prueba. Relacionado con el proceso de validación de un modelo de clasificación, existen muchos enfoques diferentes, como la validación cruzada[19] o muestreo aleatorio múltiple[20]. Utilizamos el enfoque *Holdout Set*[20]. Consiste en dividir el conjunto de datos original D en dos conjuntos de datos diferentes, el conjunto de entrenamiento y el conjunto de prueba. El algoritmo de clasificación construye el modelo de predicción con las instancias de D del conjunto de entrenamiento, y lo evalúa y valida con las instancias del conjunto D de prueba. Diferentes propuestas se han hecho para establecer el porcentaje de casos del conjunto de datos original en formar parte del conjunto de entrenamiento y el conjunto de prueba falta, pero un acercamiento común es asignar un 67 % de las instancias, para el conjunto de entrenamiento y el resto de 33 % para el conjunto de prueba.

A continuación se mencionan sus diferentes formas de presentación [21]:

- Funciones: Grupo de métodos que utilizan diferentes técnicas estadísticas para obtener la clasificación. Entre los algoritmos más conocidos están: *Multilayer Perceptron* (crea una red neuronal de retropropagación), *RBFNetwork* (red de función radio base), y *SMO* (clasificación basada en vectores de soporte).
- Árboles de decisión: Representan decisiones anidadas que sirven para clasificar los datos. Aplicando un árbol de decisión sobre los datos se obtendrán las reglas que permiten clasificarlos, a través de un modelo predictivo en el cual una instancia es clasificada siguiendo un camino de condiciones satisfechas desde la raíz del árbol hasta alcanzar una hoja, que corresponde a una clase etiquetada [22].
- Reglas de inducción: Es un área dentro del aprendizaje automático en el que reglas de producción del tipo *if-then* (si-entonces) se extraen de un conjunto de observaciones [23]. Los algoritmos incluidos en este paradigma pueden considerarse como una búsqueda heurística de espacio de estado. En la inducción de reglas, un estado corresponde a una regla candidata y los operadores corresponden a las operaciones de la generalización y la especialización que transforman una regla candidata en otra.
- Redes neuronales: Definen un modelo predictivo que se configura de forma iterativa mediante ejemplos que se emplean a modo de aprendizaje para la red neuronal. Técnicamente, emplean modelos estadísticos como la regresión múltiple para funcionar.

3. Elegir el algoritmo de minería que resuelva la tarea. En esta fase resulta fundamental el profundo conocimiento de los datos. La información acerca de su estructura y las relaciones existentes en los mismos ayuda a comprender el dominio de aplicación de las técnicas de minería de datos.

Como se muestra en la Fig. 1.1, la minería de datos se concibe como una técnica de explotación sobre fuentes de datos (ficheros de texto estructurados, bases de datos o almacenes de datos) para la extracción de conocimiento. Concretamente, la minería de datos se muestra como el proceso que extrae patrones de conocimiento de las fuentes de datos, que serán evaluados posteriormente por los analistas para finalmente extraer el conocimiento inicialmente oculto en esos datos.

Con el fin de ejecutar los algoritmos de minería de datos, se utiliza el API (*Application Programming Interface*) de Weka [24]. Aunque existen varias herramientas que permiten aplicar algoritmos de minería de datos para obtener conocimiento: sistemas propietarios (*IBM* [25], *Microsoft SQL Server Data Mining tools* [26], *Oracle* [27], *SAS* [28], etc.), y otras variantes de interfaces de código abierto como Rattle [29], interfaz para minería de datos con la herramienta estadística R.

Weka es una colección de algoritmos de aprendizaje automático para tareas de minería de datos. Los algoritmos pueden ser aplicados directamente a un conjunto de datos o llamados desde código Java. Contiene herramientas para pre-procesamiento y visualización de datos, y algoritmos para la construcción de modelos de predicción y descripción. También es muy adecuado para el desarrollo de nuevos esquemas de sistemas de aprendizaje automático. Además, es un software de código abierto publicado bajo la licencia GNU[2].

1.2.3. Evaluación de los patrones resultantes

El proceso KDD finaliza con la interpretación correcta de los resultados de la minería de datos. El descubrimiento de conocimiento se obtiene a partir del análisis de los patrones obtenidos en la fase anterior. Este análisis es el que genera conocimiento, y es la etapa final en todo proceso KDD.

Aunque ya se ha empezado a trabajar en esta línea de investigación tal y como se especifica en el capítulo 7 del presente documento, debe quedar claro que nuestro centro de atención está en determinar el mejor algoritmo de minería de datos a ejecutar.

1.3. Influencia de las meta-características en los procesos de minería de datos

Como se ha comentado anteriormente, el preprocesado de los datos es una etapa clave para la obtención de resultados óptimos en la minería de datos, ya que es ahí donde se trata con la calidad de los datos implicados [30]. Errores de introducción, aspectos estructurales, valores inadecuados, atípicos, perdidos, y otros aspectos

[2] http://www.gnu.org/licenses/gpl.html

relacionados con la limpieza de los datos pueden hacer que el patrón resultante no sea correcto. Por otro lado, según [31], las meta-características se pueden clasificar en tres grupos: general, basadas en características teóricas y relacionadas con la calidad. Existen varios trabajos que plantean la incidencia de los dos primeros grupos de meta-características en los resultados obtenidos al aplicar técnicas de minería de datos.Sin embargo, con las meta-características relacionadas con la calidad de los datos, específicamente no encontramos trabajos relacionados. Por lo que nos planteamos utilizar las meta-características que ya han sido demostradas que tienen influencia en los resultados de minería y estudiaremos la viabilidad de proponer otras relacionadas con la calidad de los datos. Una experimentación sera realizada para demostrar que verdaderamente influyen en los resultados obtenidos al aplicar técnicas de minería.

En varias propuestas se establece que el meta-aprendizaje es un mecanismo para hacer frente a los problemas de selección del algoritmo [32, 33]. En [34] los autores plantean que la complejidad de las tareas de minería de datos están relacionadas con las características de los conjuntos de datos y el sesgo inductivo de algoritmos de aprendizaje.

Existen varios metadatos que son frecuentemente extraídos de las fuentes de datos con vistas a analizar la incidencia de sus valores en los algoritmos de minería aplicados, por ejemplo: número de atributos de la fuente de datos, cantidad de instancias, porcentaje de atributos nominales, etc [35, 36].

Dentro de la comunidad científica, este tema ha sido abordado en varias ocasiones, quedando clara la evidencia de la relación existente entre el meta-aprendizaje y la selección de los algoritmos de minería. El campo de aplicación es muy extenso aunque, en general, los modelos son implementados para cada dominio en específico. Por ejemplo, encontramos su uso en la selección de modelos de previsión de series temporales [37], en la educación [38] y en la bioinformática [39, 40].

A partir del análisis realizado, se ha observado que existen propuestas que utilizan las meta-características generales de los datos y las basadas en características teóricas para ser usadas en el meta-aprendizaje, fundamentalmente para estudiar el comportamiento de los algoritmos de minería de datos.

En nuestra propuesta pretendemos determinar aquellas meta-características relacionadas con la calidad que puedan afectar el resultado de las técnicas de minería de datos, pudiéndose detectar y corregir en etapas tempranas de diseño. Además,

estudiar el comportamiento de los algoritmos de minería, teniendo en cuenta los valores de las meta-características para predecir el mejor resultado al analizar una nueva fuente de datos. Para ello utilizaremos la técnica de recomendación basada en el conocimiento (*KBR: Knowledge-Based Recommendation* [41]), dado que intenta sugerir objetos basados en inferencias sobre las preferencias y necesidades del usuario. Además tiene un conocimiento previo funcional sobre cómo un elemento en particular puede satisfacer la necesidad de un usuario y por tanto puede razonar sobre la relación entre esta necesidad y una posible recomendación [42].

1.4. Flujos de trabajos científicos

Los flujos de trabajo científicos son ampliamente reconocidos como un paradigma útil para describir, gestionar y compartir análisis científicos complejos. Han surgido para abordar el problema de la excesiva complejidad en los experimentos y aplicaciones. Proveen una forma declarativa de alto nivel de especificar cada detalle en los experimentos, enfocado siempre a lo que se quiere lograr, y no a cómo se ejecutará. Existen varias ramas que están utilizando los flujos de trabajos para resolver la capacidad de procesamiento, por ejemplo en la bioinformática para el caso de procesamiento de procesamiento de proteínas.

Varios tipos de tareas que se pueden realizar dentro de un flujo de trabajo pueden ser ejecutado por servicios locales, servicios web remotos, *scripts* y sub-flujos de trabajo. Cada componente sólo es responsable de un pequeño fragmento de funcionalidad, por lo tanto, muchos de los componentes deben ser encadenados con el fin de realizar una tarea útil. Los flujos de trabajo científicos son una alternativa viable para la aplicación de técnicas de minería de datos. Sería conveniente además que los usuarios pudieran utilizarlo fácilmente desde cualquier lugar, permitiendo de esta manera el análisis de los datos sin importar la ubicación del usuario. Para lograrlo, es preciso diseñar una base de conocimientos utilizando el desarrollo dirigidos por modelos para asegurar la homogeneidad al manejar los datos. En el próximo apartado se describe las características de una de las soluciones existentes para la implementación de flujos de trabajo científicos que es adecuada para la implementación de nuestra propuesta.

1.4.1. Taverna Workbench

Taverna[3] es un Sistema de Gestión de Flujos de Trabajo Científicos independiente del dominio, y de código abierto. Forma parte del proyecto myGrid[4], que tiene como objetivo producir y utilizar un conjunto de herramientas diseñadas para permitir a la comunidad internacional publicar y compartir información. Esta herramienta nos permitirá implementar flujos de trabajo para dar solución a las problemáticas de nuestro enfoque. La lógica de un flujo de trabajo Taverna es la siguiente: (i) definir la entrada y salida de los elementos del flujo de trabajo, y (ii) enlazar un conjunto de componentes previstos para la ejecución de las diferentes tareas, por ejemplo:

- **Beanshell**: servicio que permite ejecutar un código Java. Permite referenciar librerías existentes.
- **Nestled workflow**: un servicio que permite tener un flujo de trabajo anidado con otro.
- **IO components**: para realizar operaciones con archivos (concatenar archivos, ejecutar una línea de comando, listar archivos por su extensión).
- **UI components**: para realizar operaciones comunes como selección, opción, etc.
- **REST Services**: permite utilizar servicios genéricos REST que puedan manipular todos los métodos HTTP.

Para implementar flujos de trabajo que nos permitan la entrada y salida de datos, por ejemplo utilizar algoritmos de minería implementados en librerías externas, se debe utilizar algún mecanismo de transporte. Dada la utilidad que ofrece Taverna para utilizar llamadas a servicios REST, vamos a abordar sus fundamentos teóricos en la siguiente subsección.

1.4.2. Servicios Web RESTful

La Transferencia de Estado Representacional (REST) [43] es un estilo arquitectónico, desarrollado como un modelo abstracto de la arquitectura Web para guiar el

[3] http://www.taverna.org.uk/
[4] http://www.myGrid.org.uk

rediseño y definición del protocolo de transferencia de hipertexto (http) y identificadores uniformes de recursos (URI). Al ser aplicado a un servicio web introduce propiedades deseables, tales como el rendimiento, la escalabilidad y la modificabilidad, que permiten a los servicios trabajar mejor en la Web.

En el estilo arquitectónico REST, los datos y la funcionalidad se consideran recursos y se acceden usando URIs, por lo general vínculos en la Web. Los recursos son accionados mediante el uso de una serie de operaciones simples y bien definidas. El estilo arquitectónico REST restringe su arquitectura a una arquitectura cliente/servidor y está diseñado para utilizar un protocolo de comunicación, típicamente el HTTP. En el estilo de arquitectura REST, los clientes y los servidores intercambian recursos usando una interfaz normalizada y el protocolo.

Todas estas tecnologías se han utilizado con el fin de crear una infraestructura transparente para el usuario inexperto, permitiendo obtener resultados confiables en línea con lo esperado en un plazo razonable, sin contar con un conocimiento profundo sobre el tema. Los servicios web REST permiten a nuestro enfoque ejecutar fácilmente cada funcionalidad dentro del flujo de trabajo científico.

1.5. Desarrollo de software dirigido por modelos

Para lograr una solución que permita almacenar en una fuente de datos resultados de minería, y demás tener en cuenta los posibles formatos de datos que pueden analizarse, hemos decidido utilizar el desarrollo de software dirigido por modelos. La pregunta que surge es: ¿Qué es un modelo?

Según varias de las acepciones del diccionario de La Real Academia Española[5] un modelo es, entre otras cosas, lo siguiente:

1. Arquetipo o punto de referencia para imitarlo o reproducirlo.
2. Representación en pequeño de alguna cosa.
3. Esquema teórico, generalmente en forma matemática, de un sistema o de una realidad compleja, como la evolución económica de un país, que se elabora para facilitar su comprensión y el estudio de su comportamiento.

[5] http://www.rae.es

4. Figura de barro, yeso o cera, que se ha de reproducir en madera, mármol o metal.

5. En empresas, usado en aposición para indicar que lo designado por el nombre anterior ha sido creado como ejemplar o se considera que puede serlo. Empresa modelo. Granjas modelo.

6. Objeto, aparato, construcción, etc., o conjunto de ellos realizados con arreglo a un mismo diseño. Auto modelo 1976. Lavadora último modelo.

7. Vestido con características únicas, creado por determinado modista, y, en general, cualquier prenda de vestir que esté de moda.

8. Persona u objeto que copia el artista.

Todas estas definiciones tienen en común que un modelo (i) es una abstracción de algo que existe en la realidad, (ii) se diferencia en algo de la "cosa real" que se modela (no se tienen en cuenta todos y cada uno de los detalles o cambia el tamaño, etc.) y (iii) puede usarse como ejemplo para producir algo que existe en la realidad. A partir de estas tres características, resulta necesario para la definición de un modelo el poder determinar qué es ese "algo que existe en la realidad". Para contestar a esto se debe resaltar que un modelo debe centrarse en aquellas partes importantes de la realidad representada, desechando aspectos superfluos, con el fin de poder predecir su calidad, razonar acerca de sus propiedades específicas, comunicar sus características, etc. La realidad representada, indudablemente, depende del contexto en el que nos encontremos, por ejemplo, un edificio en arquitectura o un automóvil en ingeniería industrial.

En concreto, en ingeniería del software, los modelos deben ser, ciertamente, precursores de la implementación de un sistema software, o bien pueden derivarse de un sistema software existente con el fin de comprenderlo mejor y poder adaptarlo a nuevas necesidades [44]. Dentro del contexto de la ingeniería del software, son varias las definiciones de modelo, si bien una de las más extendidas es la propuesta en [45] donde se define un modelo como "una descripción de (parte de) un sistema escrito en un lenguaje bien definido". Por tanto, un modelo siempre está escrito en un lenguaje, ya sea lenguaje natural, un lenguaje de programación o cualquier otro. Sin embargo, con el fin de poder comprender y manejar los modelos convenientemente sin ningún tipo de ambigüedad se deben usar "lenguajes bien definidos".

Los mismos autores describen un lenguaje bien definido como "un lenguaje con una forma (sintaxis) y significado (semántica) bien definidos, el cual se puede interpretar de manera automática por una computadora". Pero, ¿cómo se puede definir dicho lenguaje? Tradicionalmente, si estos lenguajes eran textuales se definían mediante una gramática, por ejemplo en BNF[6] [46] para lenguajes de programación o XML Schemas[7] o DTD[8] para XML[9] [47], lo que cumple con el requisito de que sean interpretables de manera automática (en este caso mediante un compilador o intérprete). Sin embargo, con el fin de elevar el nivel de abstracción de los modelos maximizando su utilidad en ingeniería del software, estos suelen utilizar una sintaxis gráfica, por lo que se necesita un mecanismo diferente para definir dichos lenguajes. Este mecanismo se llama metamodelado.

Con el fin de lidiar con estos aspectos, se dispone de una estructura jerárquica en cuatro niveles [48], en la cual, el nivel inferior es "una instancia" del nivel superior (excepto el nivel superior que es reflexivo por lo que consiste en "una instancia" de sí mismo). El nivel más bajo se denomina M0 y se corresponde con el sistema software real. En el nivel M1 se encuentra el modelo que representa el sistema software, mientras que el nivel M2 contiene el metamodelo al cuál se ajusta el modelo. Por último el nivel M3 se corresponde con el metametamodelo al cuál se ajusta el metamodelo del nivel M2. Este último nivel tiene su fundamento en la reflexividad ampliamente utilizada en informática, por ejemplo en bases de datos, sistemas operativos o lenguajes que contienen descripciones de sí mismos llamadas esquemas, metadatos o metaclases [49].

[6]BNF o notación de Backus-Naur: siglas en inglés de Backus Naur Form, es una metasintaxis usada para expresar gramáticas libres de contexto, es decir, una manera de describir lenguajes formales. Se utiliza extensamente como notación para las gramáticas de los lenguajes de programación, de los sistemas de comando y de los protocolos de comunicación, etc.

[7]XML Schema: es un lenguaje de esquema utilizado para describir la estructura y las restricciones de los contenidos de los documentos XML de una forma muy precisa, más allá de las normas sintácticas impuestas por el propio lenguaje XML. Se consigue así una percepción del tipo de documento con un nivel alto de abstracción. Fue desarrollado por el World Wide Web Consortium (W3C).

[8]DTD: siglas en inglés de *Document Type Definition*, es una descripción (a través de la especificación de restricciones) de la estructura y sintaxis de un documento XML o SGML. Su función básica es la descripción del formato de datos, para usar un formato común y mantener la consistencia entre todos los documentos que utilicen la misma DTD.

[9]XML: siglas en inglés de Extensible Markup Language, es un metalenguaje extensible de etiquetas desarrollado por el W3C. Es una simplificación y adaptación del SGML y permite definir la gramática de lenguajes específicos (de la misma manera que HTML es a su vez un lenguaje definido por SGML). Por lo tanto XML no es realmente un lenguaje en particular, sino una manera de definir lenguajes para diferentes necesidades. Algunos de estos lenguajes que usan XML para su definición son XHTML, SVG, MathML.

Si se ejemplifica esta estructura jerárquica mediante un programa en Pascal, su ejecución estaría en el nivel M0, mientras que el propio programa se representaría en el nivel M1 y la gramática BNF que permitiera la definición de programas en Pascal sintácticamente correctos estaría en el nivel M2. EL nivel M3 estaría ocupado mediante la gramática de BNF definida según el propio BNF.

Uno de los metamodelos más usados en ingeniería del software es UML (Unified Modeling Language) [50]. UML contiene facilidades para visualizar, especificar, construir y documentar un sistema software, de tal manera que se puedan modelar todos los aspectos del sistema, tales como procesos de negocio y funciones del sistema, expresiones de lenguajes de programación o esquemas de bases de datos. Por lo tanto UML se situaría como un metamodelo a nivel M2 dentro de MDD. Se debe resaltar que UML es muy adecuado para el desarrollo de software de propósito general como aplicaciones de gestión o telecomunicaciones, pero sin embargo es difícil de aplicar cuando se desarrolla software para contextos más específicos, por ejemplo para medicina, cuyo principal interés es modelar directamente el dominio de aplicación en lugar de realizar una compleja adaptación de UML al contexto específico [49].

Con el fin de desarrollar metamodelos útiles para dominios concretos se usa MOF (Meta Object Facility) [51]. MOF es un estándar creado para definir lenguajes de modelado de manera formal, es decir, se situaría en el nivel M3, teniendo la capacidad de definirse a sí mismo. De hecho, MOF es el lenguaje a partir del cual se define UML (o mejor dicho, el metamodelo de UML). MOF suministra los conceptos y notación gráfica necesaria para crear metamodelos, así como funcionalidades para la definición de identificadores, tipos primitivos de datos, etc. Un ejemplo de la estructura jerárquica para el modelado se muestra en la Fig. 1.2.

El desarrollo dirigido por modelos en un nuevo paradigma de desarrollo de software. Teniendo como principal elemento del proceso de desarrollo la posibilidad de definir metamodelos y modelos, y sus posibles transformaciones para obtener el código correspondiente de manera automática. Este enfoque nos permitirá mantener la uniformidad en la información que se manipule, así como la no dependencia de sistemas específicos de bases de datos para gestionar la información.

Bajo el paraguas dirigido por modelos, y de acuerdo con [52], un modelo es una “descripción de (parte de) un sistema escrito en un idioma bien definido", mientras que un lenguaje bien definido es un lenguaje con la forma bien definida (sintaxis) y

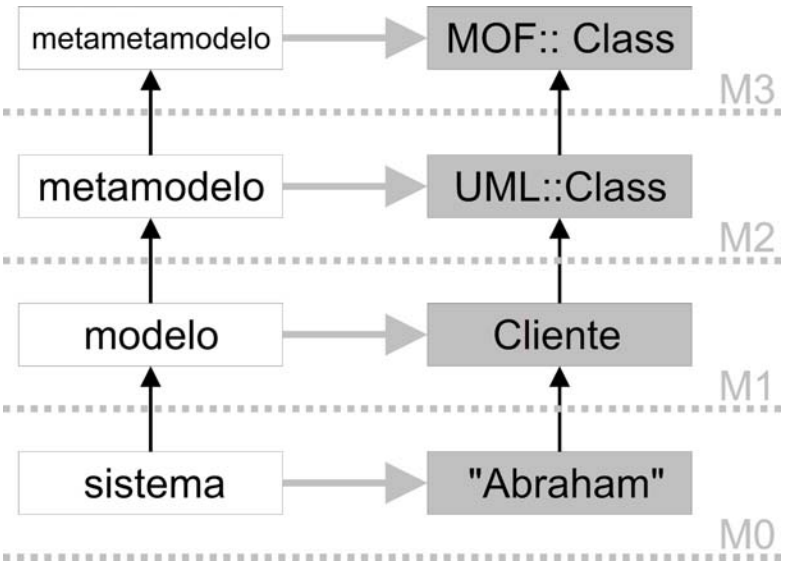


FIGURA 1.2: Estructura jerárquica de MOF.

significado (semántica), que es adecuado para la interpretación automatizada por un ordenador". Por lo tanto, por un lado, un modelo debe centrarse en las partes importantes de un sistema, evitando así los detalles supérfluos, por otro lado, idiomas bien definidos pueden ser diseñados por medio del metamodelado [53], que proporciona la base para la creación de modelos de una manera significativa, precisa y consistente.

Con vistas de implementar esta visión basada en modelos, herramientas como el marco de modelado Eclipse (EMF)[10] es profundamente utilizado. El proyecto EMF es un marco de modelado con facilidad para la generación de código para la construcción de herramientas y otras aplicaciones basadas en un modelo de datos estructurado. Como Eclipse ha sido concebido como una plataforma modular, puede ser extendido por medio de *plugins* para agregar más características. Los metamodelos creados pueden ser incluidos en un *plugin* que contiene toda la nueva funcionalidad necesaria para crear modelos conformes al metamodelo. Eclipse también proporciona funcionalidades para realizar transformaciones: entre modelos o transformaciones de texto, con el fin de generar nuevos modelos refinados o algún código necesario a partir de modelos, respectivamente.

[10]http://www.eclipse.org/modeling/emf/

1.6. Hipótesis de partida

La hipótesis de partida de este trabajo es que el análisis de la calidad de las fuentes de datos, a partir de sus meta-características, puede ayudar a seleccionar aquellos algoritmos de minería más apropiados. En la Fig. 1.3 se esboza a grandes rasgos nuestra propuesta. Para ello, es preciso contar con una base de conocimiento que permita almacenar todos los resultados obtenidos de la aplicación de experimentos de minería en fuentes de datos introducidas por usuarios expertos. Esta información podrá ser después utilizada para la construcción de un sistema recomendador para determinar la mejor opción de algoritmo para cada nueva fuente de datos introducida por los usuarios inexpertos. En la medida que se logre identificar las expectativas de los usuarios inexpertos, estas serán convertidas en conocimiento al aplicar el modelo de minería recomendado.

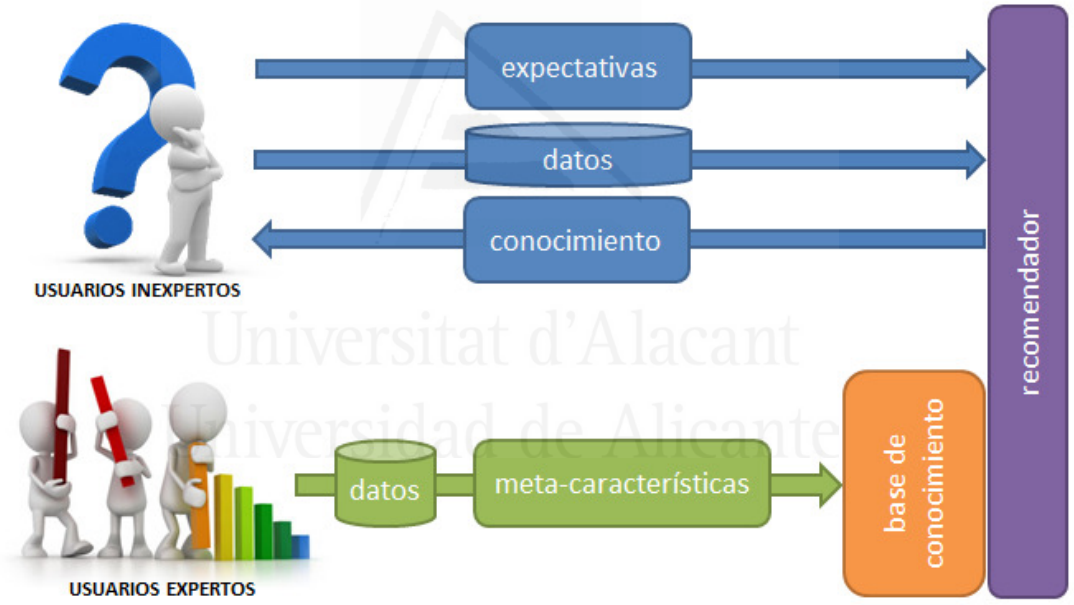


FIGURA 1.3: Propuesta general de la solución.

Existen varios aspectos adicionales que deben ser resaltados en aras de lograr un producto con la mejor calidad posible. Por un lado, la enorme heterogeneidad de formatos en que se presentan los datos, y por otro, la necesidad de contar con un proceso sistemático a la hora de diseñar la base de conocimiento y la construcción del recomendador. Estos requisitos necesariamente imponen la utilización de recursos que nos permitan transformar los posibles datos de entrada de una manera precisa teniendo en cuenta nuestro propósito, y diseñar correctamente los pasos

necesarios hasta lograr que un usuario inexperto pueda obtener conocimiento de sus datos.

Para comprobar esta hipótesis se desarrollará un sistema basado en el desarrollo de software dirigido por modelos conducido por flujos de trabajos científicos que permita tener en cuenta las meta-características de los datos para crear una base de conocimiento con resultados de experimentos de minería de datos, que será utilizada por un sistema recomendador para recomendar el mejor algoritmo posible para una fuente de datos introducida al sistema por un usuario inexperto.

Se debe resaltar que el objetivo de nuestra propuesta es democratizar el uso de la minería de datos, y por supuesto, los resultados que se pueden obtener por medio semi-automáticos nunca serán capaces de alcanzar la precisión lograda usando procedimientos manuales, pero preferimos esta variante teniendo en cuenta el alto coste y tiempo que consume un experto para realizar esta tarea, además de la ventaja de no tener que contar con su presencia.

1.7. Objetivos.

En resumen, esta tesis doctoral plantea como cuestión principal desarrollar una propuesta que permita a usuarios inexpertos facilitar la obtención de conocimiento al aplicar técnicas de minería. Específicamente, los objetivos de este trabajo de tesis doctoral se pueden sintetizar en:

- Identificar criterios de calidad de datos que influyan en los resultados obtenidos al aplicar técnicas de minería de datos, específicamente, clasificación. En nuestra aproximación no identificamos meta-características pero, se pretende usar algunas de las que existen en el estado de la cuestión.
- Proponer una aproximación que integre los pasos necesarios para lograr la aplicación de técnicas de minería de datos por usuarios inexpertos.
- Demostrar la validación práctica de la propuesta mediante el desarrollo de varios experimentos en casos de estudio reales.

Para conseguir los objetivos propuestos se propone realizar las tareas que a continuación se describen:

- Se analizarán los principales trabajos existentes que tengan puntos en común con la problemática planteada. De esta forma se realizará un análisis crítico y objetivo para plantearse los elementos a incluir, y el mecanismo de solución que se aportará.

- Definición de metamodelos para la representación de la información de minería. Teniendo claro que optamos por la utilización de técnicas de desarrollo de software dirigido por modelos, se hace necesario implementar aquellos metamodelos que permitirán modelar todos los recursos pertenecientes a la solución propuesta.

- Definición de los criterios de calidad a controlar en el proceso de minería. Es necesario definir cada uno de los criterios de calidad que intervendrán posteriormente en el proceso que se diseñe.

- Creación de una base de conocimiento para almacenar los resultados de minería. Este elemento es fundamental para el desarrollo de nuestra propuesta. Aquí tienen que incluirse los conceptos relacionados con la aplicación de técnicas de minería que serán utilizados posteriormente.

- Construcción de un recomendador para sugerir al usuario inexperto el mejor algoritmo de minería a utilizar. Se hace necesario construir un mecanismo que permita estudiar la información almacenada para predecir el algoritmo que mejor corresponda con las características de la nueva fuente de datos a analizar.

- Ejecutar la propuesta del recomendador. Luego de obtener el algoritmo recomendado se debe obtener el modelo de minería con vistas a ofrecerle una salida factible al usuario inexperto.

1.8. Estructura del documento.

El presente documento queda estructurado de la siguiente manera: En el capítulo 2 se presenta un análisis crítico de las propuestas existentes relacionadas con la aplicación de técnicas de minería. Un análisis de la influencia de la calidad de los datos en los resultados de minería es presentado en el capítulo 3. En el siguiente apartado se expone la propuesta creada para la obtención de conocimiento por

usuarios inexpertos 4. En los capítulos 5 y 6 varios casos de estudios son presentados para validar la propuesta teórica presentada. En 7 se exponen algunas líneas de investigación en las que se ha comenzado a trabajar con algunos resultados obtenidos. Las conclusiones finales de la investigación se exponen en el capítulo 8, además de presentar algunas líneas a donde puede encaminarse el trabajo futuro.

Capítulo 2

Estado de la cuestión

En este capítulo se describen las principales propuestas encontradas que abordan temas relacionados con el desarrollo de soluciones de minería de datos para usuarios inexpertos. La descripción de propuestas se agrupa en las ramas más significativas relacionadas con nuestro trabajo, como son: calidad de los datos, minería de datos amigable, ontologías para minería de datos, meta-aprendizaje, así como otras propuestas basadas en ingeniería del sofware relacionadas con la minería de datos.

2.1. Calidad de datos en tareas de minería

En [54], Batini et al ofrecen un estudio comparativo de las metodologías existentes para medir la calidad de los datos de manera general, describiendo algunas de las definiciones dadas para las medidas de calidad más utilizadas.

Como se ha mencionado anteriormente 1.2.1, la fase de preprocesado es sumamente importante dentro del proceso KDD. Existen numerosos trabajos en la literatura que abordan el problema de calidad de los datos desde el punto de vista de la limpieza de los mismos en esta etapa:

1. Para la detección y eliminación de duplicados [55–57], reconocimiento de instancias bajo distintas etiquetas [58, 59], comparaciones de cadenas de caracteres, etc.

2. Resolución de conflictos en instancias [60, 61] usando técnicas específicas de limpieza [62, 63].

3. Valores atípicos [64], perdidos o incompletos [62, 65], entre otros.

La estandarización de los datos también ha sido considerada por varias técnicas de limpieza para resolver problemas como la estructura heterogénea de los datos (i.e. representación estándar de fechas) [66, 67]. Existen otros acercamientos que consideran la calidad de los datos durante la segunda fase del proceso KDD, selección de atributos y algoritmos para la minería de datos. Por ejemplo, proponiendo el trabajo con los metadatos en almacenes de datos, almacenando medidas de calidad relacionadas con el estado de las fuentes de datos. En [68], Chiang et al proponen una variante para proveer a los usuarios todos los detalles para realizar correctas decisiones. Plantean que además de los informes tradicionales es también esencial brindarle información a los usuarios acerca de la calidad, por ejemplo, la calidad de los metadatos, por la gran importancia que tiene conocer el estado de los metadatos y su implicación en la toma de decisiones.

Jarke and Vassiliou en [69], utilizan los metadatos como recurso para almacenar los resultados al medir algunas medidas de calidad, pero estas medidas están basadas en el preprocesamiento de los datos, es decir, considerando la calidad de datos como la limpieza de los mismos. Sin embargo, la calidad de los datos no sólo se refiere a los procedimientos de limpieza, según Zhu et al [70], existe un amplio espectro de criterios relacionados con la calidad de los datos que deben ser considerados. Es importante, incluso luego de concluir el proceso de limpieza de los datos, analizar otros criterios de calidad que pueden afectar a la obtención de resultados no fiables cuando se aplican técnicas de minería de datos [71].

Cuando se buscan propuestas asociadas a la calidad intrínseca de los datos, una de las principales propuestas presentadas es [72], donde Berti-Equille abordó un tema novedoso, definiendo un método para medir la calidad de las reglas de asociación obtenidas, con el objetivo de definir cuales en realidad eran correctas. La autora utilizó CWM (Common Warehouse Metamodel) para la inclusión de las medidas que propuso para que pudieran ser aplicadas posteriormente, creando una extensión del metamodelo de instancias de datos, que denominó *QoD*. Este es el principal trabajo de calidad de datos en técnicas de minería en la literatura. Como se puede apreciar esta propuesta tiene puntos en común con la nuestra, considerando la calidad de los datos para minería, sin embargo, consideramos la calidad de los datos a partir de una perspectiva más amplia, suponiendo que no sólo la limpieza de datos es importante para la minería de datos y obtener conocimiento

fiable, sino también analizar otros criterios de calidad de datos existentes en las primeras etapas del proceso de descubrimiento de conocimiento. Estos criterios de calidad de datos los hemos incluidos dentro de las meta-características existentes en las fuentes de datos, como hemos dicho en la sección 1.3.

Dentro del ámbito de la calidad de datos para procesos de minería de datos, y hasta donde hemos podido comprobar en las fuentes bibliográficas, no se establecen mecanismos con el objetivo de guiar la selección del algoritmo de minería de datos apropiado.

2.2. Minería de datos amigable

Cada vez son más necesarios mecanismos que permitan a los usuarios inexpertos, poder analizar fácilmente sus datos. El gran desafío está en lograr la aplicación de técnicas de minería de datos de manera amigable. Algunos enfoques se centran en la realización de sistemas de minería de datos interactivos, al considerar una comunicación adaptativa y eficaz entre los usuarios humanos y los sistemas informáticos, tal como expone Zhao [73], donde el usuario es guiado a través del proceso de minería de datos.

FIU-Miner [74], es un sistema integrado que facilita a los usuarios llevar a cabo tareas de minería de datos *ad-hoc*. Proporciona una interfaz gráfica de usuario amigable para permitir a los usuarios configurar rápidamente tareas de minería de datos complejas.

Dimitropoulos et al [75] han propuesto una plataforma de minería de datos, diseñada para el análisis de grandes conjuntos de datos heterogéneos de manera escalable, amigable, e interactiva. El principal inconveniente de este sistema es que requiere conocimiento sobre el proceso KDD, ya que se centra en la aplicación de manera sencilla, de diferentes técnicas en cada paso del proceso. Por lo tanto, los usuarios sin conocimiento previo de minería de datos no se pueden aprovechar de este enfoque.

Adicionalmente, hay algunas propuestas que tienen por objeto asistir a usuarios inexpertos en la aplicación de minería de datos centrado básicamente en un dominio de aplicación específico. Por ejemplo, Camiolo and Porceddu [76], proponen una alternativa para generar secuencias genómicas, lo que permite la extracción

de características de genes a partir de un archivo de anotación mientras controla varios filtros de calidad y el mantenimiento de un entorno gráfico de usuario amigable.

De Bodt et al [77] proponen una herramienta específica amigable para la extracción e integración de datos en biología. En concreto, los autores proporcionan a los biólogos un mecanismo para investigar las asociaciones entre los genes y las proteínas codificadas.

En el campo educativo, Salcines et al [42], presentan un sistema recomendador colaborativo que utiliza minería de datos distribuida para la mejora continua de cursos de e-learning, utilizando una medida de evaluación de las reglas descubiertas basada en pesos, y que tiene en cuenta la opinión de los expertos y de los propios profesores, para producir recomendaciones cada vez más efectivas.

Zorrilla y García-Saiz [78] proponen una herramienta web denominada ElWM, con el objetivo de ayudar a los instructores que participan en la educación a distancia a descubrir perfiles y modelos de comportamiento de sus alumnos acerca de cómo navegan y trabajan en sus cursos virtuales ofrecidos en sistemas de gestión de contenidos y aprendizaje, tales como *Blackboard* o *Moodle*. Una versión ampliada de esta herramienta se describe en [79].

Hasta donde hemos podido investigar no existen propuestas de minería de datos enfocadas a usuarios inexpertos de manera general, es decir, se proponen soluciones pero enfocadas a un dominio concreto. En nuestro caso se pretende analizar cualquier fuente de datos, sin restricción a un dominio específico. La idea es poder configurar y construir un recomendador de manera dinámica por un usuario experto, dada la complejidad de los pasos a realizar en este proceso, teniendo entre las posibles variables a configurar, el dominio de los datos.

2.3. Ontologías para minería de datos

La selección de un algoritmo de minería es el núcleo del proceso de descubrimiento de conocimiento [14]. Existen varias ontologías enfocadas en representar los conceptos relacionados con las técnicas de minería de datos. A continuación se descrbirán algunas de ellas. Por ejemplo, OntoDM [80] es una ontología de nivel

superior para los conceptos de minería de datos que describe las entidades destinadas a cubrir todo su dominio, mientras que la ontología EXPO [81] esta enfocada en el modelado de experimentos científicos. Una ontología más completa es *DMOP*, propuesta por Hilario et al [82], la cual no solo describe algoritmos de aprendizaje (incluyendo sus mecanismos internos y modelos), sino tambiém los flujos de trabajo. Un amplio conjunto de operadores de minería de datos se describen en las ontologías [83] y en eProPlan [84].

En cuanto a los flujos de trabajo de minería de datos, la ontología KDDONTO de Diamantini et al [85], apunta al descubrimiento de algoritmos de descubrimiento de conocimientos adecuados y la descripción de los flujos de trabajo de este proceso. Está centrada principalmente en los conceptos relacionados con las entradas y salidas de los algoritmos y cualquier pre y post condiciones para su uso.

La ontología basada en meta-minería de datos de flujos de trabajo para el descubrimiento de conocimiento [86] tiene como objetivo apoyar la construcción de flujos de trabajo para el proceso de descubrimiento de conocimiento.

Por otra parte, Vanschoren y Soldatova [87] proponen una ontología específica para describir los experimentos de aprendizaje automático de una forma estandarizada para apoyar un enfoque colaborativo para el análisis de los algoritmos de aprendizaje (desarrollado aún más en [88]).

Hay algunos proyectos que permiten a la comunidad científica contribuir con su experimentación en la mejora del proceso de descubrimiento de conocimiento. La base de datos de experimentos de aprendizaje automático desarrollada por la Universidad de Leuven [89] ofrece una herramienta web para almacenar los experimentos realizados en una base de datos y poder consultarlos. El proyecto e-LICO [90] ha desarrollado un asistente de minería de datos basado en el conocimiento que se apoya en una ontología de minería de datos para planificar el proceso y proponer flujos de trabajo clasificados para un problema concreto dado [86].

Estas propuestas han sido estudiadas con el objetivo de identificar los elementos de minería de datos que utilizan en sus respectivos procesos. A partir del estudio realizado, estos conceptos han sido utilizados para el diseño de nuestra base de conocimiento. A diferencia de nuestra propuesta, estos proyectos están orientados a apoyar a los mineros de datos expertos, siendo necesario algún conocimiento de conceptos básicos y términos específicos de la materia en cuestión. Por otra parte, a pesar de que las ontologías mencionadas son muy útiles para proporcionar la

semántica entre elementos, carecen de mecanismos para la automatización de la gestión, e intercambio de los metadatos, como el metamodelado [91]. Las ontologías son aplicables generalmente a dominios específicos, mientras que en este caso se debe estar preparado en un futuro para el análisis de datos, teniendo formatos disímiles.

2.4. Meta-aprendizaje

La influencia de meta-características para comprobar el comportamiento de los algoritmos de minería en fuentes de datos es una temática que ha sido tratada en varias propuestas. El campo de aplicación es amplio, aunque, en general, el meta-aprendizaje se implementa para cada dominio específico, por ejemplo, en series temporales para modelos de pronósticos [92], o en bioinformática [39]. En el ámbito educativo encontramos también varias propuestas que apuestan por abordar esta problemática [38, 93, 94].

Existen diferentes enfoques acerca de cuales meta-características pueden utilizarse para analizar su influencia en las fuentes de datos. En la mayoría de los casos se eligen propiedades medibles de las fuentes de datos y algoritmos.

Algunos autores [95, 96] utilizan medidas generales, estadísticas y de información teórica, mientras que otros utilizan puntos de referencia [97]. Por otro lado, hay trabajos que utilizan características del modelo como metadatos, tales como la relación media de sesgo, o su sensibilidad al ruido [98], o la forma estructural y el tamaño del modelo como en [99].

En [97], Abdelmessih et al proponen un enfoque denominado plantillas para meta-aprendizaje, propuesto con el fin de recomendar una combinación jerárquica de algoritmos, en lugar de sólo uno. En [100], Brazdil et al plantean que una de las variantes recomendadas para asistir al usuario en la selección de un modelo de minería adecuado es el uso del meta-aprendizaje, ya que este mecanismo puede ser implementado de manera automática, y es sistemático.

La tarea de elegir un algoritmo adecuado para un determinado conjunto de datos es muy importante. Varios son los trabajos de investigación que han sido propuestos [35, 101–104] que abordan esta línea de trabajo. Fundamentalmente utilizando algunas medidas teóricas estadísticas y de información para caracterizar las fuentes

de datos y, a continuación, tratar de captar la relación entre las características del conjunto de datos medidos y el rendimiento del algoritmo de clasificación por árboles de decisión, métodos de reglas de inducción basados en instancias o modelos de regresión.

En [105], Song et al proponen un método de recomendación para algoritmos de clasificación, basados en el estudio de características de las fuentes de datos, con la particularidad de utilizar un método que establece la cercanía entre fuentes de datos. Aplicaron un conjunto de experimentos sobre varias fuentes de datos para demostrar la validez del método de recomendación propuesto.

Recientemente, Gore y Pise [36] aportaron otra solución para recomendar algoritmos de clasificación, basada en el método de Selección de Algoritmos Dinámica, fundamentalmente comparando la información de una fuente de datos de entrada con la información histórica almacenada y sus meta-características. Esta propuesta sólo hace una selección de los principales algoritmos de clasificación, utilizando sólo 8 de ellos, y no está enfocada al uso por usuarios inexpertos.

En esta sección se han analizado varias propuestas relacionadas con la aplicación del meta-aprendizaje y el uso de meta-características que inciden en el comportamiento de los algoritmos de minería. En nuestra propuesta pretendemos utilizar algunas de las meta-características frecuentemente utilizadas, así como otras relacionadas específicamente con la calidad de los datos, en la construcción de un sistema recomendador para encontrar el algoritmo que obtenga el posible mejor rendimiento ante una fuente de datos introducida por un usuario inexperto.

2.5. Propuestas basadas en ingeniería de software relacionadas con la minería de datos

En esta sección se presenta una descripción de las aproximaciones creadas para aplicar técnicas de ingeniería de software en las distintas fases del proceso KDD. Se realiza una revisión detallada de las aproximaciones más importantes desarrolladas en la actualidad que tienen puntos de contacto con nuestra propuesta.

2.5.1. Propuestas desarrolladas para las distintas etapas del KDD.

El objetivo de esta sección es presentar el enfoque de las soluciones actuales a las distintas fases del proceso KDD. Esta tarea describe un proceso que puede resumirse en tres fases: i) la integración de los datos en un repositorio único, ii) la fase de selección de atributos y algoritmos para la minería de datos (el núcleo del proceso KDD), y iii) finaliza con el análisis e interpretación de los patrones resultantes.

Para la primera fase del proceso KDD, denominada *integración* existen propuestas para todos los niveles de abstracción. Para el modelado conceptual [106], presentan DWEP (Data Warehouse Engineering Process) una metodología que permite realizar correctamente el completo desarrollo de un almacén de datos, basándose en los estándares UML (Unified Modeling Language) y UP (Unified Software Development Process, también conocido como Unified Process).

Para el modelado lógico tenemos la propuesta de Muñoz et al [107], y para el nivel físico la de Tziovara et al [108]. En sentido general estas propuestas se basan en definir mecanismos que permiten obtener los modelos del almacén de datos para las distintas etapas del proceso KDD (conceptual, lógico y físico). En el caso de las propuestas de la etapa de preprocesado tienen en cuenta solamente criterios de calidad relacionados con la limpieza de los datos. Por lo que no tienen en cuenta otros criterios de calidad de datos que pueden estar presentes en las fuentes de datos, centro de atención de nuestra propuesta.

Para la segunda fase del proceso KDD, hay varias propuestas [109–114] que abarcan el modelado conceptual, el lógico y el físico.

En [109] Luján-Mora presenta un perfil UML que permite lograr la representación de las características multidimensionales más relevantes de los almacenes de datos a nivel conceptual. Implementando un conjunto de transformaciones que permiten automáticamente generar la correspondiente implementación en otra plataforma destino. Sin embargo, no se han estudiado los diferentes criterios de calidad de datos que pueden influir en la segunda fase de KDD con el fin de poder formalizar la selección del algoritmo adecuado en el almacén de datos.

En la tercera fase del proceso KDD existen propuestas para el modelado conceptual de técnicas de minería de datos, entre ellas encontramos [115–119]. En dichas

propuestas, el diseño de modelos conceptuales de minería de datos se integra como parte del proceso global de descubrimiento de conocimiento KDD, tomando como base (y aprovechando) los elementos de los modelos multidimensionales de los almacenes de datos.

2.5.2. Aproximaciones existentes relacionadas con el modelado de técnicas de minería

Existen varias aproximaciones existentes que abordan temas relacionados con el modelado de técnicas de minería desde distintos puntos de vista:

- CRISP-DM [120] (*CRoss Industry Standard Process for Data Mining*), es una guía de referencia para el desarrollo sistemático de proyectos de Data Mining. Según Marbán [121], no representa un proceso maduro que pueda calificarse como una metodología sólida, es decir, que si bien establece un conjunto de tareas y actividades que deben ser llevadas a cabo en el proyecto, no establece con que técnicas o modelos debe implementarse cada actividad.
- PMML (*Predictive Model Markup Language*) [122] provee un formato para definir algoritmos de aprendizaje independiente de plataforma. Facilita el intercambio de modelos usando el estándar XML (*eXtensible Markup Language*), utilizado principalmente para intercambiar modelos de minería de datos entre herramientas.
- PBMS (*Pattern Base Management System*) [123] define como una plataforma que almacena y gestiona patrones, por lo que su objetivo no es diseñar modelos de minería de datos sino modelar los resultados de este proceso.

2.5.3. Otras propuestas

En los últimos años existen pocas propuestas relacionadas con la aplicación de técnicas de ingeniería del software a minería, o incluso metodologías que dirigan el proceso de minería, o permitan reutilizarlo. En [124] se propone un método para la evaluación e integración de los procesos de minería de datos en los procesos de negocios. El autor plantea reducir el esfuerzo reutilizando las soluciones de minería exitosas, pero la propuesta todavía se encuentra en un nivel incipiente,

lejos de ser un acercamiento sólido. Por otro lado, en [125] se propone un modelo de costos genérico para proyectos de minería de datos. Luego de realizar un profundo estudio de los factores que inciden en todo el proceso, los autores propusieron una ecuación para determinar el coste de un proyecto de minería, identificando los factores asociados.

Marbán et al [126], presentan un modelo de proceso de minería de datos e ingeniería, haciendo distinción entre lo que es un modelo de proceso, y lo que es una metodología y su ciclo de vida.

Aunque existen algunas propuestas que utilizan las técnicas de minería de datos para extraer conocimiento en campos como la educación, o para la obtención del mejor algoritmo de minería, hasta donde se ha podido investigar, no existe ningún acercamiento que incluya la posibilidad de aplicación de técnicas de minería de datos fácil de usar para usuarios inexpertos.

Capítulo 3

Base de conocimiento para almacenar resultados de minería

La calidad de los datos es un elemento a tener en cuenta en cualquier sistema de información. La aplicación de técnicas de minería de datos no está exenta de esta situación, teniendo en cuenta que su razón de ser, es el análisis de datos.

Un análisis de las meta-características relacionadas con la calidad de los datos es aquí presentado, para evidenciar la necesidad de creación de una base de conocimiento que permita gestionar la información generada en experimentos de minería.

Como hemos introducido anteriormente, las meta-características se pueden agrupar en tres grupos (ver sección 1.3). En este capítulo se expone un análisis para demostrar que existen meta-características relacionadas con la calidad de datos que afectan considerablemente el resultado de las técnicas de minería. Además pretendemos utilizar las meta-características de caracter general, y basadas en medidas teóricas, ya que estas sí tienen influencia en los resultados de minería de datos.

Con el objetivo de medir la calidad de los datos en bases de datos relacionales, en etapas previas del proceso de minería, una serie de heurísticas utilizando CWM han sido definidas. En este capítulo se presentan los resultados de la experimentación realizada.

Los resultados obtenidos nos permitieron diseñar un proceso que permitiese gestionar los criterios de calidad sobre fuentes de datos para la realización de técnicas de minería. Teniendo como antecedentes los experimentos aquí presentados y la

relación de los procesos de minería con la calidad de los datos, se define una base de conocimiento para almacenar de forma homogénea los elementos que inciden en el proceso de aplicación de técnicas de minería de datos.

3.1. Calidad de datos

Un proceso típico de minería de datos [127] comienza a partir de un conjunto de datos, del cual el analista selecciona un subconjunto que va a formar parte del análisis. Además, debe seleccionar la técnica y más concretamente el algoritmo a aplicar a ese subconjunto de datos. Éste se aplica con unos parámetros en función del objetivo buscado o bien con los parámetros por defecto. Finalmente, se obtienen los patrones de comportamiento común en los datos y el análisis de estos patrones es lo que permite descubrir conocimiento útil [14]. Un proceso de minería de datos exitoso depende de la calidad de las fuentes de datos con el fin de obtener un conocimiento fiable.

Existen un buen número de dimensiones de calidad de datos (ver [128], [72], y el estándar ISO [129]) que se podrían tener en cuenta en el proceso de minería de datos como por ejemplo, la selección adecuada de los atributos que formarán el modelo de minería de datos, o la correcta selección de parámetros y/o algoritmos a utilizar en el proceso. Cuando se trabaja con grandes cantidades de datos, un minero de datos inexperto y no consciente del contexto, puede construir un modelo seleccionando un conjunto de atributos y una técnica determinada, pero esto no asegura que el patrón que obtenga sea correcto aunque los datos estén limpios y libres de errores.

3.1.1. Determinación de criterios de calidad de datos

Para establecer los criterios de calidad que influyen en la obtención de conocimiento al aplicar técnicas de minería de datos, debemos iniciar el análisis describiendo el concepto Calidad de Datos. La definición más comúnmente aceptada es la basada en la "adecuación al uso", dada por Deming en [130].

Esta definición expresa que un usuario sólo puede evaluar el nivel de calidad de un conjunto de datos usados en una determinada tarea y en un contexto específico,

acorde a un grupo de criterios, determinando de esta forma si los datos pueden ser usados para el propósito previsto [60, 131].

En la aplicación de técnicas de minería, significa que los datos que participen en el proceso tengan en cuenta el contexto, es decir, el conocimiento del problema que se está tratando para que el usuario pueda obtener conocimiento útil al aplicar técnicas de minería de datos. De esta forma, se podría evitar que la aplicación de dichas técnicas resulte en conocimiento superfluo, contradictorio o incluso erróneo. Por consiguiente, enfocaremos la atención en analizar los problemas de calidad de datos que afectan negativamente los resultados obtenidos al aplicar técnicas de minería, específicamente abordaremos las referentes a las técnicas de clasificación.

Tradicionalmente, varios criterios de calidad relacionados con la limpieza de los datos se han sido considerados en la fase de preprocesamiento del proceso KDD [30]. Sin embargo, otros criterios de calidad de datos deben ser analizados en la fase de minería de datos. Para determinar cuales criterios de calidad deben ser tenidos en cuenta, nuestro análisis está basado en tres aspectos fundamentales:

1. Las definiciones de cada una de las dimensiones de calidad propuestas por el éstandar ISO/IEC 25012 [129], siendo un modelo de calidad de datos para la gestión de los datos en los sistemas de información.
2. Algunos indicios dados en el artículo [11], así como la consulta de trabajos anteriores relacionados [132, 133].
3. La experiencia derivada del trabajo previo realizado sobre calidad de datos.

3.1.2. Dimensiones de calidad propuestas por el éstandar ISO/IEC 25012

En este apartado se presenta las dimensiones de calidad de datos definidas en el éstandar ISO/IEC 25012 [129], siendo un modelo de calidad de datos para gestionar los datos en los sistemas de información de manera general.

En las tablas 3.1, 3.2, 3.3 se muestran cada una de las definiciones de las dimensiones de calidad propuestas. Este modelo considera quince dimensiones o características que se agrupan según dos puntos de vista:

- Inherente: se refiere a características de calidad de datos que tienen el potencial intrínseco para satisfacer las necesidades, cuando los datos son usados en condiciones específicas. La calidad inherente de los datos se refiere al grado en que los valores de los datos pertenecen a un dominio específico, cumplen con ciertas restricciones (por ejemplo, reglas del negocio), y respetan ciertas relaciones de los datos (por ejemplo, consistencia), entre otros [134].

- Dependiente del sistema: se refiere al grado en el cual la calidad de datos es enriquecida y preservada dentro de un sistema computacional cuando es usado bajo condiciones específicas. La calidad de datos depende del dominio tecnológico bajo el cuál se usan los datos. Dependen de las capacidades de los componentes de los sistemas computacionales como: hardware (por ejemplo para proveer el acceso adecuado a los datos), software de sistemas (por ejemplo para el respaldo de los datos y proveer la Recuperabilidad de los mismos), y otros tipos de software (por ejemplo en herramientas de migración para proveer Portabilidad) [134].

Dimensión	Descripción
Exactitud	El grado en el cual el dato tiene atributos que representan el valor correcto de un concepto o evento en un contexto específico de uso.
Completitud	El grado en el cual el dato asociado a una entidad tiene valores para todos los atributos esperados e instancias de entidad relacionadas, de acuerdo a un contexto específico de uso.
Consistencia	El grado en el cual el dato tiene atributos libres de contradicción y son coherentes con otros datos en un contexto específico de uso.
Credibilidad	El grado en el cual el dato tiene atributos considerados como verdaderos y creíbles por usuarios en un contexto específico de uso.

Sigue en la página siguiente.

Dimensión	Descripción
Actualidad	El grado en el cual el dato tiene los atributos que son del período correcto en un contexto específico de uso.

Tabla 3.1: Dimensiones de calidad de datos propuestas por el éstandar ISO/IEC 25012. Grupo Inherente.

Dimensión	Descripción
Accesibilidad	El grado en el cual se puede acceder al dato en un contexto específico de uso.
Conformidad	El grado en el cual el dato tiene atributos que se adhieren a normas, convenciones o regulaciones vigentes y reglas relacionadas con la calidad de datos en un contexto específico de uso.
Confidencialidad	El grado en el cual el dato tiene los atributos que aseguran que éste es sólo accesible e interpretable por usuarios autorizados en un contexto específico de uso.
Eficiencia	El grado en el cual el dato tiene los atributos que pueden ser procesados y proporciona los niveles esperados de funcionamiento (desempeño) usando las cantidades y los tipos de recursos apropiados en un contexto específico de uso.

Sigue en la página siguiente.

Dimensión	Descripción
Precisión	El grado en el cual el dato tiene atributos que son exactos o que proporcionan su discriminación en un contexto específico de uso.
Trazabilidad	El grado en el cual el dato tiene atributos que proporcionan un rastro de auditoría de acceso a los datos y de cualquier cambio hecho a los datos en un contexto específico de uso.
Comprensibilidad	El grado en el cual el dato tiene atributos que le permiten ser leído e interpretado por usuarios, y es expresado en lenguajes apropiados, símbolos y unidades en un contexto específico de uso.

Tabla 3.2: Dimensiones de calidad de datos propuestas por el éstandar ISO/IEC 25012. Grupo Inherente y Dependiente del sistema.

En nuestro contexto de minería de datos, los criterios de calidad definidos en el éstandar en el grupo Dependiente del Sistema no tienen influencia, ya que los usuarios son los que suministran los datos cuando desean analizarlos, y estos son independientes de cualquier sistema. En principio, no existe dependencia de ningún sistema informático para la aplicación de los criterios de calidad, por lo que nos centraremos en los criterios de calidad mostrados en el grupo Inherente.

Dimensión	Descripción
Disponibilidad	El grado en el cual el dato tiene atributos que le permiten ser recuperados por usuarios autorizados y/o aplicaciones en un contexto específico de uso.

Sigue en la página siguiente.

Dimensión	Descripción
Portabilidad	El grado en el cual los datos tienen atributos que les permiten ser instalados, substituidos o movidos de un sistema a otro conservando la calidad existente, en un contexto específico de uso.
Recuperabilidad	El grado en el cual el dato tiene atributos que le permiten mantener y conservar un nivel especificado de operaciones y calidad, aún en caso de falla, en un contexto específico de uso.

TABLA 3.3: Dimensiones de calidad de datos propuestas por el éstandar ISO/IEC 25012. Grupo Dependientes del sistema.

Una vez que los criterios de calidad fueron analizados podemos asegurar que ninguno de ellos se ajusta a nuestro contexto. Esta afirmación está justificada ya que analizaremos los datos que los mismos usuarios introduzcan en el proceso, dado que esas fuentes de datos son generadas en sistemas externos, no hay modo de comprobar su Credibilidad o Actualidad, por mencionar dos de ellos. El que pudiera generar más duda sería el caso del criterio Completitud, pero la diferencia se mostrará detalladamente más adelante.

Luego de haber finalizado el análisis de los criterios propuestos por el éstandar ISO/IEC 25012, se concluye que estos criterios no son adecuados para su aplicación en técnicas de minería de datos. Se propone entonces, un estudio que permita identificar criterios de calidad de datos específicos a nuestro contexto, en este caso, la minería de datos.

3.1.3. Criterios de calidad para minería de datos

Teniendo en cuenta algunos de los indicios plasmados entre los retos de investigación y aplicación en [11], hemos detectados algunos criterios de calidad que deben ser considerados. Concretamente, creemos que cuando los datos no son conocidos

estamos ante un criterio de calidad denominado **completitud de datos**. Según Fayyad, las relaciones complejas entre los datos deben ser detectadas, por lo que este grupo incluye el **grado de correlación entre los atributos** y el **balance entre los datos**.

Todos estos criterios de calidad encontrados han sido medidos a partir de la relación entre los mismos datos, y deben situarse al mismo nivel de importancia, por lo que no hay ningún subnivel de organización entre ellos. Debe anotarse que otros criterios de calidad pueden perfectamente ser adicionados, siempre teniendo en cuenta la definición inicial de "adecuación al uso" que marca nuestra línea de trabajo.

Al analizar nuestro contexto de aplicación, y la idea de Fayyad de tener en cuenta los problemas que existen en las relaciones entre los atributos que componen las fuentes de datos, hemos identificado dos grupos de problemas fundamentales que agrupan a los criterios de calidad de datos para minería:

1. Problemas entre los datos de una columna:

   a) **Datos Balanceados**: Los datos almacenados en una columna estan balanceados si el número de valores diferentes que representan cada instancia diferente es significativamente igual. Esto significa que se esperan cantidades de instancias similares para cada valor que pueda tomar la columna. El valor esperado con el que se comparará, se obtendrá considerando que los datos existentes tienen una distribución uniforme, y se calculará a partir del valor medio de las cantidades de las diferentes clases presentes en el citado atributo. ¿Cómo saber si son parecidos los valores existentes en las fuentes de datos? De manera general, lo que se quiere es determinar un mecanismo que nos permita definir si un conjunto de observaciones están distribuidas uniformemente respecto a un total. Si es así, se puede aplicar efectivamente técnicas de minería teniendo a esos atributos presentes en los datos de entrada o predicción. Existe una prueba estadística para determinar la significatividad de la diferencia en un conjunto de frecuencias observadas para atributos discretos, es la prueba llamada Chi Cuadrado (O^2). Teniendo en cuenta que para nuestro caso, siempre que se necesite aplicar técnicas de minería se tendrán observaciones de las respectivas cantidades de los atributos

seleccionados, este método se ajusta perfectamente. La expresión del estadístico de Chi Cuadrado se puede observar a continuación:

$$\chi^2_{obs} = \sum_{i=1}^{n} \frac{(f_i - np_i)^2}{np_i}$$

donde:

- f_i: número de frecuencias observadas.
- p_i: número de frecuencias esperadas.
- n es el número de categorías a ser consideradas.

Al aplicar la fórmula para obtener Chi cuadrado se resta al número de frecuencias observadas, el número de frecuencias esperadas; se eleva esta diferencia al cuadrado, lo que hace que todos los valores asuman un valor positivo, y luego se divide el cuadrado obtenido entre las frecuencias esperadas. Esto se hace de manera independiente para cada una de las categorías. Una vez terminado este paso, se suman los resultados obtenidos en cada categoría y ese valor resultante de la suma es el valor Chi cuadrado observado, el cual deberá ser comparado con el valor Chi cuadrado crítico tabulado según el nivel alpha de significatividad escogido y los grados de libertad correspondientes.

Nivel de significación alfa: Error que se comete al rechazar la Ho. Dado que se conoce a priori permite establecer el umbral de tolerancia para el cual se acepta Ha y se rechaza Ho. En general se considera un margen de error del 5 %, lo que significa que hay un margen de error del 0.05. Los grados de libertad (gl) se calculan a partir de la formula: $gl = k - 1$. Siendo k el número de categorías existentes.

La prueba O^2 es considerada como una prueba no paramétrica que mide la discrepancia entre una distribución observada y otra teórica (bondad de ajuste), indicando en qué medida las diferencias existentes entre ambas, de haberlas, se deben al azar en el contraste de hipótesis. El modelo experimental tiene una muestra y nuestro objetivo es la bondad del ajuste. Es decir evaluar el ajuste de una función de distribución a una muestra de variables.

Planteamiento de la hipótesis.

- Hipótesis nula (Ho). Las diferencias observadas entre los valores observados y los teóricos se deben al azar.

- Hipótesis alterna (Ha). Las frecuencias observadas difieren de las que corresponden a un modelo teórico (en nuestro caso la ley uniforme).

Para nuestro objetivo si rechazamos Ho, no podríamos afirmar que los datos sigan un modelo teórico (en este caso uniforme). Se acepta Ho si los datos están distribuidos según una ley uniforme. Así, se define la zona de rechazo de Ho como: todo valor del estadístico que sea mayor que el punto crítico tabulado para Chi cuadrado con sus correspondientes grados de libertad y nivel de significación. Si $\mid X^2 \mid > X^2_{k-1,alfa}$ rechazamos la hipótesis nula (Ho).

b) **Completitud**: Este criterio generalmente es usado en la fase de limpieza de datos, pero en este contexto se analizará otro tipo de valores nulos, los nulos estructurales [135–137]. Cuando existen valores vacíos en una fuente de datos necesariamente no es porque falte su valor, sino que debido a su contexto ese dato no puede tener valor. Por ejemplo: Si tenemos una fuente de datos con información de ciudades españolas, y existe un atributo que almacena la provincia a la que pertenece cada ciudad, habrá ciudades que tendrán su atributo provincia vacío porque existen algunas ciudades autónomas en España (como Melilla y Ceuta) que no pertenecen a ninguna provincia. Nótese que la existencia de un valor nulo en un atributo X no significa que el valor no sea conocido, sino que no es aplicable en su contexto.

2. Problemas entre los datos de diferentes columnas:

a) **Correlación entre atributos**: Dos columnas están correlacionadas si cambios en un valor de una columna está asociado con cambios en el otro atributo, existiendo una medida de asociación entre ambas columnas. Se implementó el coeficiente de correlación de Pearson para determinar el grado de correlación entre dos columnas, a través de la siguiente fórmula:

$$r_{xy} = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{\sqrt{n\sum x_i^2 - (\sum x_i)^2}\sqrt{n\sum y_i^2 - (\sum y_i)^2}}$$

donde

- r_{xy}: es el coeficiente de correlación. Este valor siempre estará en un rango entre $+1$ and -1 ambos inclusive. Un valor de 1 implica que x

y y están perfectamente correlacionadas (a medida que incrementa x incrementará la y), mientras que un valor de -1 implica que y decrece al x decrecer. Un valor de 0 implica que no existe corelación entre x y y.

- x y y son las dos variables implicadas en el cálculo de la correlación.
- n es el número de valores a ser considerados.

Teniendo en cuenta que el método anterior nos permite obtener el coeficiente de correlación entre dos atributos númericos, se propone un método alternativo para determinar el grado de asociación entre atributos nominales. Para ello utilizamos el método de Cramer, ya que es el más popular entre las medidas basadas en Chi Cuadrado para la asociación nominal porque proporciona un valor entre 0 y 1 en dependencia del tamaño de la tabla. La fórmula de Cramer es la siguiente:

$$V = \sqrt{\chi^2_{obs} O++(m-1)}$$

donde

- chi^2_{obs}: es el valor de Chi Cuadrado.
- m: min(f,c).
- f: es el número de filas.
- c: es el número de columnas.
- $O++$: total de observaciones.

3.1.4. Formalización de los criterios encontrados usando CWM

A partir de tener identificados ciertos criterios de calidad de datos (completitud correlación, balanceo), en esta sección nos proponemos formalizar el proceso de medición de la calidad de datos mediante un proceso de ingeniería inversa de las fuentes de datos existentes. Este paso fue realizado para validar que los criterios de calidad afectan a los resultados de minería, con el fin de realizar a posteriori la base de conocimiento.

Las fuentes de datos de nuestro caso de estudio corresponden a una base de datos relacional implementada en el gestor de bases de datos SQL Server 2000. Como nuestra propuesta se basa en obtener un modelo de datos relacional enriquecido

con los criterios de calidad de datos implícitos, utilizaremos el metamodelo CWM para dar solución a la problemática planteada ya que contiene varios metamodelos que sirven para especificar modelos dependientes de varias tecnologías aplicables al diseño de bases de datos.

La implementación de nuestra propuesta se ha llevado a cabo utilizando como entorno de desarrollo integrado (Integrated Development Enviroment, IDE) Eclipse. Teniendo en cuenta que a continuación se abordará la formalización de los criterios de calidad para técnicas de minería de datos, específicamente para técnicas de clasificación, y que entre los paquetes que tiene implementado CWM consta uno para minería, pudiera pensarse por qué no utilizarlo en el desarrollo de nuestra propuesta. Hay que tener en cuenta que lo que buscamos es representar la estructura de los datos en un esquema relacional, por lo que sólo nos hace falta en principio el paquete relacional de CWM. Este paquete contiene un metamodelo con los elementos necesarios para modelar cada uno de los aspectos de las bases de datos relacionales (ver Fig. 3.1). Con este metamodelo podemos representar tablas, columnas, claves primarias, claves extranjeras, etc.

3.1.4.1. Descripción de los pasos a realizar

Asumiremos que tenemos un modelo relacional con los datos obtenidos del diccionario de datos, y es en este diccionario donde vamos a hacer las consultas para obtener el modelo de datos y, junto con el propio esquema de la base de datos, el valor de los diferentes criterios de calidad de datos.

Aplicaremos un paso de ingeniería inversa, de manera automática, en el cual partiendo de las fuentes de datos y después de implementar los mecanismos que permitirán saber los valores de los criterios encontrados, obtendremos un modelo CWM del esquema de la base de datos junto con los diferentes criterios de calidad definidos en este trabajo y sus respectivos valores.

A continuación detallaremos los pasos necesarios para obtener un metamodelo CWM enriquecido con los valores de los criterios de calidad encontrados, básicamente nuestra propuesta consta de las siguientes tareas:

1. Establecer la conexión con la base de datos: Este es el paso inicial, donde según el gestor de bases de datos se crea el adaptador necesario para establecer

la conexión. En este caso como la fuente de datos está alojada en un servidor Sql Server 2000, se utilizó la librería de clases java.sql.DriverManager para establecer la conexión.

2. Obtener el diccionario de datos: Un Diccionario de datos es un repositorio de todos los metadatos relevantes de los elementos almacenados en la base de datos. Cada Sistema de Gestión de Bases de Datos (*DBMS, Database Management System*) tiene su propia estructura para almacenar las definiciones y representaciones de cada elemento, pero también contienen información acerca de las restricciones de integridad, asignación de espacios en memoria y no la estructura general de la base de datos. Dentro de todos los DBMS toda la información de las estructuras de datos está almacenada en el diccionario de datos, por lo que la información de cada base de datos implementada puede ser especificada a través de éste.

3. Agregar los métodos y consultas necesarios para añadir al modelo de las fuentes de datos los criterios de calidad definidos en el capítulo anterior y sus respectivos valores: En este paso se implementarán las consultas SQL que permitan obtener los valores de los criterios de calidad encontrados.

   a) Para el caso de la completitud, se debe determinar para cada columna la cantidad de valores nulos que aparecen. En concreto se desea saber qué porcentaje representan los valores nulos del total de elementos de cada columna para cada tabla de la base de datos.

   b) Para el caso del desbalance se implementó un método que devuelve el valor Chi Cuadrado para cada columna de cada tabla. Con el objetivo de saber si la distribución de los elementos dentro de cada columna se comporta de manera uniforme.

   c) Para el caso de la correlación se implementó un método que devuelve el grado de correlación de esa columna respecto al resto de las columnas de las tablas de la base de datos.

4. Obtener el metamodelo CWM enriquecido: Luego de haber finalizado los pasos anteriores se genera el modelo CWM enriquecido.

3.1.4.2. Implementación

En este epígrafe se abordarán los detalles del proceso de implementación de las consultas y métodos con el objetivo de obtener el metamodelo CWM enriquecido con los criterios de calidad encontrados. Para lograr el objetivo planteado es indispensable conocer como CWM define los principales elementos (ver Fig. 3.1. Fuente: http://www.omg.org/spec/CWM/1.1), y por otro lado, tener presente que el diccionario de datos de SQL Server consiste en un grupo de tablas en las cuales los metadatos son almacenados. Por lo que la información de los metadatos que se requiera puede ser obtenida consultando los siguientes objetos: *information_schema.tables* (datos acerca de las tablas), *information_schema.columns* (datos acerca de las columnas de cada tabla), y *information_schema.table_constraints* (datos acerca de las restricciones existentes en cada columna). A partir de tener recopilada toda esta información pasamos a imple-

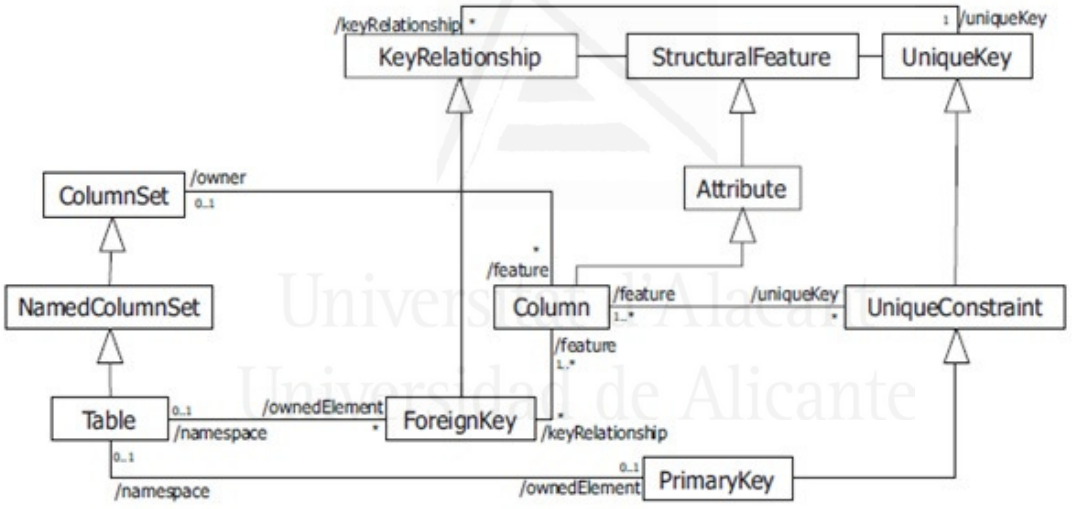


FIGURA 3.1: Parte del metamodelo CWM Relacional usado.

mentar las consultas necesarias:

1. Para el caso de la completitud se implementó una consulta que devuelve el porcentaje que representa la cantidad de valores nulos existentes respecto al total de atributos:

```
stmtNullValues.executeQuery("select
count('"+auxCol.getName()+"') * 100 / count(*)" + "
from information_schema.columns " + "where table_name = '"
+ tab.get(i).Name + "'");
```

2. Para calcular el valor Chi cuadrado, se implementaron los siguientes métodos para obtener el Balance de los atributos en una columna de una tabla: Para este caso se implementaron dos métodos:

   a) *getFrequencies*: Permite obtener para cada valor diferente que exista en la columna almacenar su frecuencia de aparición. El parámetro *nums* es el arreglo con todos los valores de la columna. En este método también se calcula el valor esperado, que va a ser el valor promedio de todos los valores presentes en la columna que se esté procesando.

   b) Balanced: Es el método que devuelve si una columna está balanceada o no respecto a la distribución de sus valores. Utiliza como parámetro los valores de la columna que se está analizando y se auxilia del método *getFrequencies* para calcular su valor de Chi cuadrado. Si al comparar el valor Chi cuadrado con el valor almacenado en la variable *FunctDistribChiCuad005* es menor, los datos están balanceados, sino no.

```
private static Map<Double,Integer> getFrequencies(double[]
nums)
{
  double cantidadtotal = 0;
  Map<Double,Integer> freqs = new
  HashMap<Double,Integer>();
  for (double x : nums)
   {
    cantidadtotal = cantidadtotal + x;
    if (freqs.containsKey(x))
     freqs.put(x, freqs.get(x) + 1);
    else
     freqs.put(x, 1);
   }
   valorEsperado = cantidadtotal/nums.length;
   return freqs;
}
public static boolean Balanced(double[] randomNums)
{
```

```
  Map<Double,Integer> ht =
  getFrequencies(randomNums);
  double chiSquare1 = 0;
  for (double v : ht.values())
   {
    double f1 = v - valorEsperado;
    chiSquare1 += f1* f1;
   }
  chiSquare1 /= valorEsperado;
  int GradosLibertad = randomNums.length-1;
  if (chiSquare1<FunctDistribChiCuad005[GradosLibertad+1])
   return true;
  else
   return false;
}
```

Además se implementaron consultas para devolver la frecuencia de aparición de cada elemento en cada columna de cada tabla con el propósito de aplicarle el método Chi cuadrado.

3. Para el caso de la correlación se implementó un método que devuelve el factor de correlación entre dos columnas de la base de datos:

```
public static double getmycorrelation(double[]
scores1,double[] scores2)
{
 double sum_x=0, sum_y=0, sum_square_x=0, sum_square_y=0,
 sum_xy=0,n=scores1.length;
 for (int i = 0; i < scores1.length; i++)
  {
   sum_x += scores1[i];
   sum_square_x += scores1[i] * scores1[i];
   sum_xy += scores1[i]* scores2[i];
  }
 for (int i = 0; i < scores2.length; i++)
  {
   sum_y += scores2[i];
   sum_square_y += scores2[i] * scores2[i];
  }
```

```
    return (sum_xy - ((sum_x*sum_y)/n))/
    math.sqrt((sum_square_x - ((sum_x *
    sum_x)/n))* (sum_square_y - ((sum_y*sum_y)/n)));
}
```

Utilizar CWM para extender nuestra propuesta a un nivel superior fuera bastante complicado. Un elemento importante es la gran diversidad de formatos de datos disponibles actualmente, por lo que se debe diseñar una propuesta que permita tener en cuenta esta diversidad.

Luego de haber sido formalizado como se calculan los criterios de calidad anteriormente mostrados, a continuación se procederá a realizar un conjunto de experimentos que permitan demostrar la viabilidad de la propuesta.

3.1.5. Experimentos para mostrar la adecuación de los criterios de calidad

Durante el desarrollo de esta sección mostraremos varios ejemplos, y situaciones creadas al interactuar con una fuente de datos real, para demostrar de esta forma la adecuación de los criterios calidad de datos planteados.

Hemos realizado varios experimentos usando un caso de estudio real, con vistas a comprobar si los criterios de calidad anteriormente presentados están relacionados con la obtención de conocimiento no útil, no relevante e incluso inconsistente cuando se aplican técnicas de clasificación en fuentes de datos con problemas de calidad.

3.1.5.1. Descripción del caso de estudio de baloncesto

Este caso de estudio contiene datos del comportamiento de jugadores en juegos de varios torneos de baloncesto de la Liga Cubana de Baloncesto, incluyendo datos de los "III Juegos Deportivos del ALBA" (http://www.juegosalba.cu), un torneo internacional, de ambos sexos efectuado en la provincia cubana de Ciego de Ávila en abril del 2009. Los datos de nuestro caso de estudio fueron almacenados en una base de datos conforme al esquema que se presenta en la Fig. 3.2. Los datos para cada juego y jugador contienen un conjunto de indicadores (ver Tabla 3.4).

Además, se tiene en cuenta datos básicos de los jugadores como son el sexo, estatura, peso, etc. Para cada jugador, su posición (ver Tabla 3.5), y una evaluación nominal global (Poco Integral, Integral y Muy Integral) es conocida. Esta evaluación es establecida por la actuación obtenida en cada juego, teniendo en cuenta su posición dentro de la cancha, y los indicadores ofensivos y defensivos logrados.

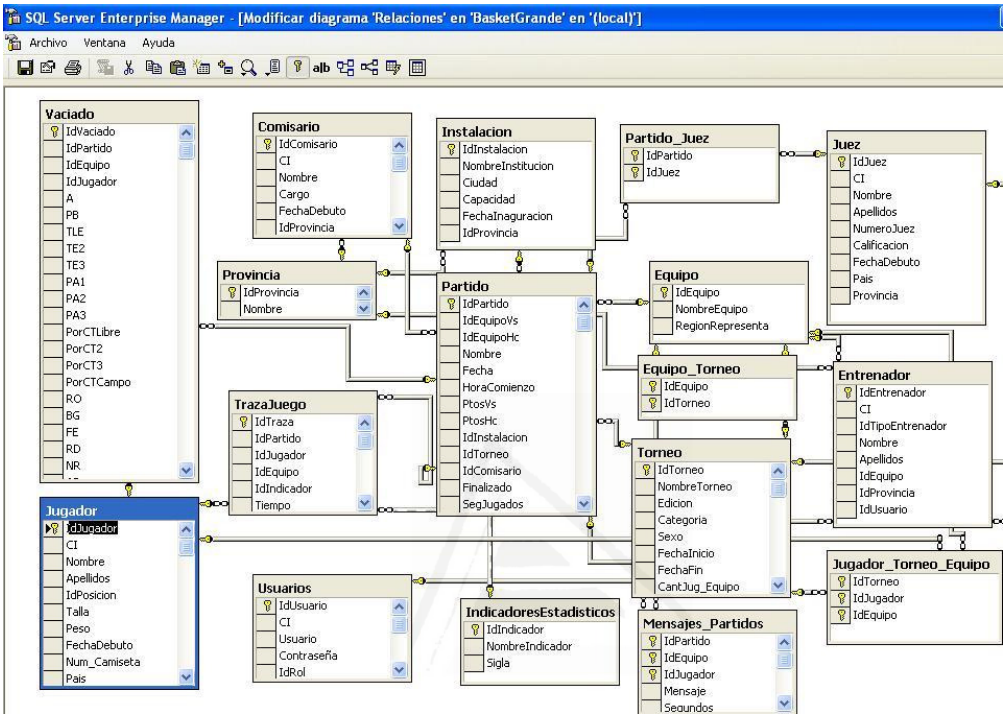


FIGURA 3.2: Resumen del esquema de la base de datos.

Aunque nuestra base de datos se puede considerar pequeña, es muy apropiada para generar varios modelos de clasificación, y mostrar como una selección de atributos poco adecuada puede corresponderse con la obtención de patrones no útiles, poco precisos o no válidos.

3.1.5.2. Descripción de los experimentos

En la Fig. 3.3 se muestra como se ha diseñado la realización de los experimentos para este caso de estudio. De manera general, primero se comprobará la incidencia de los criterios de calidad de manera individual, y posteriormente se realizarán experimentos para ver como incide la presencia de los criterios de calidad de manera combinada en las fuentes de datos.

Estos experimentos nos permitirán detectar si los datos de origen son adecuados para llevar a cabo las técnicas de minería de datos deseadas o si, por el contrario,

Tabla 3.4: Grupo de indicadores estadísticos.

Indicadores	Nombre	Abreviatura
Defensivos	Bolas Ganadas	BG
	Rebotes Defensivos	RD
	Fallar en el enfrentamiento a un adversario que penetra hacia el cesto	FE
	No recuperar el rebote tras el lanzamiento del equipo rival	NR
	Asistencias Defensivas	AD
Ofensivos	Asistencias	A
	Pérdidas del Balón	PB
	Tiros Libres Errados	TLE
	Tiros Errados 2 Puntos	TE2
	Tiros Errados 3 Puntos	TE3
	Rebotes Ofensivos	RO
	Tiros Libres Anotados	PA1
	Tiros Anotados 2 Puntos	PA2
	Tiros Anotados 3 Puntos	PA3
	Porcentaje de efectividad en tiros libres	PorCTLibre
	Porcentaje de efectividad en tiros de 2 puntos	PorCT2
	Porcentaje de efectividad en tiros de 3 puntos	PorCT3
	Porcentaje de efectividad en tiros de campo	PTCampo
	Permanencia en la cancha de juego	PC

Tabla 3.5: Posiciones de los Jugadores.

Posición	Abreviatura	Función
Base	B	Organizar el juego y ayudar a sus compañeros
Alero	A	Tener el peso ofensivo de su equipo
Pivot	P	Garantizar el juego en la pintura

existen restricciones marcadas por el contexto o los propios datos que deben ser tenidas en cuenta con el fin de extraer conocimiento útil.

3.1.5.3. Correlación de datos

En la Tabla 3.6 se puede observar como al aplicarse el algoritmo de regresión lineal entre los indicadores PA1, TLE y PorCTLibre, el factor de correlación obtenido fue de 0.7283. Este indica que los atributos están correlacionados. De manera similar ocurre al aplicar regresión lineal entre TE2, PA2 y PorCT2; TE3, PA3 y PorCT3; y TE2, TE3, PA2, PA3 y PorCTCampo. Estos resultados llevan a descartar los atributos PorCTLibre, PorCT2, PorCT3, PorCTCampo para utilizarse en técnicas de regresión si se combinan con atributos que se utilizan para su cálculo.

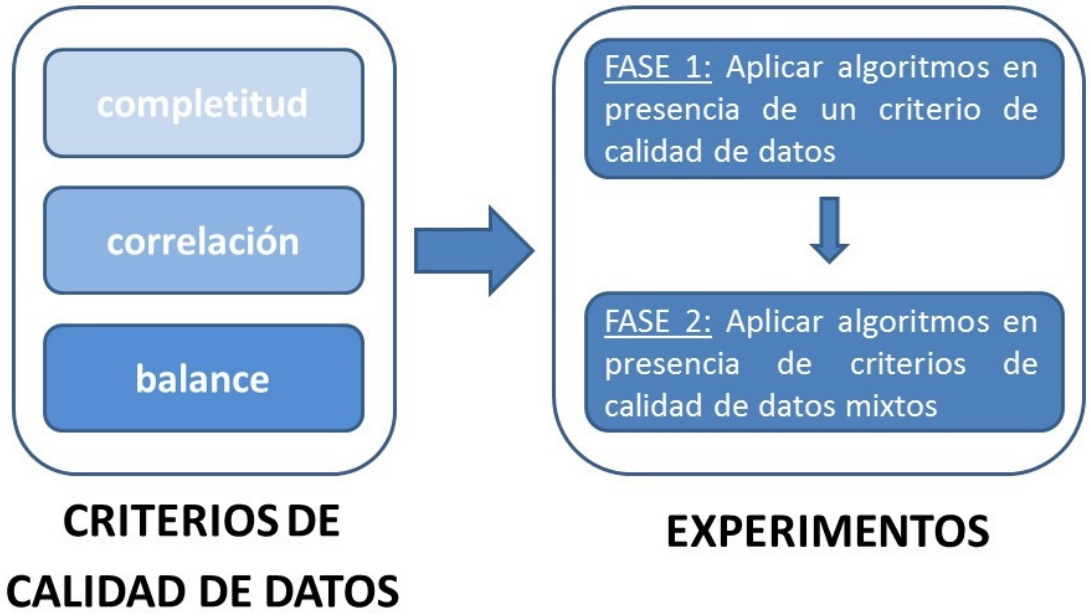


FIGURA 3.3: Visión general de nuestros experimentos para considerar criterios de calidad de datos en técnicas de clasificación.

Es de destacar que los atributos fueron seleccionaos a partir de criterios de los expertos, al saber que cada tipo de tiro tenía en la fuente de datos su correspondiente atributo que almacenaba el porcentaje de cada tipo de tiro. Evidentemente al hacer las respectivas pruebas se corroboró la idea inicial para este caso. Por lo que se reafirma lo dicho en [11], que el criterio de los expertos, además el conocimiento del dominio es un factor importante a la hora de seleccionar los atributos que estarán presentes en la aplicación de técnicas de minería. Si al aplicar técnicas

TABLA 3.6: Resultados obtenidos al aplicar regresion lineal entre TLE, PA1 and PorCTLibre.

$PorCTLibre = -6{,}1835 * TLE + 16{,}7723 * PA1 + 14{,}049$ Time taken to build model: 0.19 seconds === Cross-validaton === === Summary ===	
Correlation coefficient	0.7283
Mean absolute error	18.886
Root mean squared error	24.3515
Relative absolute error	59.2649 %
Root relative squared error	68.4085 %
Total Number of Instances	420

de clasificación, un atributo *entrada* depende, o esta relacionado con un atributo a *predecir*, y esto es conocido *a priori*, entonces el patrón resultante será inútil. La principal razón es que los patrones resultantes de una técnica de clasificación

tratan de agrupar los atributos de entrada dependiendo de su relación con el atributo seleccionado para predecir (variable dependiente). Aunque es relativamente fácil identificar la correlación entre los atributos en grupos de pequeñas dimensiones, es un problema complejo cuando tratamos con fuentes de datos de alta dimensionalidad, por lo que este fenómeno debería ser solucionado de una manera formal.

3.1.5.4. Completitud

Otro de los criterios de calidad de datos que analizamos fue la completitud de datos (es decir la presencia de valores nulos en las fuentes de datos). Nuestro objetivo era comprobar de qué manera influía la presencia de valores nulos en los atributos seleccionados para utilizarse en técnicas de clasificación. Para este ejemplo se seleccionó un conjunto de datos donde se tiene la estatura de todos los jugadores y sus respectivas posiciones (ver Tabla 3.7). Se trabajó con los registros de los jugadores masculinos solamente, para que el clasificador no tuviera en cuenta el atributo sexo. La idea fue aumentar la cantidad de valores nulos progresivamente en la fuente original y aplicar el mismo clasificador J48. Los resultados se muestran en la Tabla 3.8. La primera fuente de datos analizada (ver Tabla 3.8) fue con

Tabla 3.7: Datos estadísticos de los jugadores masculinos.

Posición	Instancias
Base	136
Alero	152
Pivot	132
Total	420

los datos originales, y el clasificador obtuvo un 84 % de instancias correctamente clasificadas. En el segundo y tercer caso se aumentó la cantidad de valores nulos en la fuentes a 10 % and 30 %, respectivamente. Los resultados alcanzados sugirieron el mismo comportamiento: mientras la presencia de los valores aumenta, los porcentajes de correcta clasificación decresen. Teniendo en cuenta el total de valores nulos adicionados a la fuente de datos original, se puede concluir que el aumento de valores nulos homogéneamente adicionados a la fuente de datos no afecta considerablemente los resultados del clasificador. El objetivo del cuarto caso fue aplicar el clasificador con una mayor presencia de valores nulos en una sola clase, y no homogéneamente repartidos como en los casos anteriores. Se aplicó el clasificador al conjunto de datos con el 50 % de valores nulos, perteneciendo todos a

TABLA 3.8: Resultados obtenidos al aplicar el algoritmo J48 de clasificación mientras la cantidad de valores nulos en la fuente de datos aumenta

	Instancias	1-Instancias Correctamente Clasificadas	2-Instancias Correctamente Clasificadas con 10 % valores nulos por cada clase	3-Instancias Correctamente Clasificadas con 30 % valores nulos por cada clase	4-Instancias Correctamente Clasificadas con 50 % valores nulos para la clase Base
Base	136	122 (89.7 %)	111 (81.6 %)	86 (63.2 %)	62 (45.6 %)
Alero	152	134 (88.2 %)	112 (73.7 %)	83 (54.6 %)	134 (88.2 %)
Pivot	132	97 (73.5 %)	87 (65.9 %)	68 (51.5 %)	97 (73.5 %)
Portero	0	-	-	-	-
Total instancias	420	-	-	-	-
Instancias Correctamente Clasificadas	-	353 (84 %)	310 (73.8 %)	237 (80.9 %)	293 (83.2 %)
Instancias Incorrectamente Clasificadas	-	67 (16 %)	68 (16.2 %)	56 (19.1 %)	59 (16.8 %)
Clases ignoradas instancias desconocidas	-	0	42	127	68

una misma clase, en este caso, la clase Base. Intuitivamente, mientras la cantidad de valores nulos es mayor, los porcentajes de correcta clasificación disminuyen. Debido a esto, fueron correctamente clasificados 62, 134 y 97 instancias de Base, Alero y Pivot representando el 45 %, 88 % y 73 %, respectivamente. El porcentaje mas bajo perteneció a la clase con mayor cantidad de instancias con valores nulos para el atributo Peso, como se puede observar en la Tabla 3.8.

Los porcentajes de instancias correctamente clasificadas con similar variación entre ellos pueden ser observados. Sin embargo, cuando observamos la clase que fue afectada, el impacto se hace más evidente.

Como se puede apreciar en la Tabla 3.8, existe una clase Portero con cero impacto en los resultados. Esto se debe a que en el Baloncesto no existe esa posición en el terreno de juego. Esa clase está en la fuentes de datos para demostrar que, las clases que no tienen presencia en la fuente de datos no inciden en los resultados, por lo que deben ser eliminadas del conjunto de entrada.

3.1.5.5. Datos Balanceados

En este apartado se desea analizar como inciden la presencia de clases desbalanceadas en los resultados al aplicar técnicas de clasificación. Para ello se seleccionó un grupo de datos desbalanceados, de manera tal que la clase *Base* tuviera mas instancias. Cuando el algoritmo *J48* fue aplicado los resultados (ver Fig. 3.4) pueden ser considerados buenos: el 73.4375 % de las instancias fueron correctamente clasificadas. Sin embargo, se puede apreciar en la matriz de confusión del resultado, que la alta mayoría de las instancias correctamente clasificadas pertenecen a la clase favorecida por el desbalance.

```
Correctly Classified Instances         235               73.4375 %
Incorrectly Classified Instances        85               26.5625 %
Kappa statistic                          0.5261
Mean absolute error                      0.2575
Root mean squared error                  0.3682
Relative absolute error                 64.34   %
Root relative squared error             82.3633 %
Total Number of Instances              320

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.943     0.336       0.77     0.943      0.848      0.853    Base
                 0.506     0.1         0.625    0.506      0.559      0.721    Alero
                 0.463     0.047       0.721    0.463      0.564      0.823    Pivot
Weighted Avg.    0.734     0.217       0.724    0.734      0.717      0.814

=== Confusion Matrix ===

   a   b   c   <-- classified as
 164   6   4 |   a = Base
  31  40   8 |   b = Alero
  18  18  31 |   c = Pivot
```

FIGURA 3.4: Resultados obtenidos al aplicar el clasificador a una fuente de datos desbalanceada.

3.1.5.6. Experimentos aplicando diferentes algoritmos de clasificación

En esta sección, se describen varios experimentos para la aplicación de diferentes algoritmos a diversas técnicas de clasificación, con el objetivo de determinar la influencia de los criterios de calidad de datos anteriormente expuestos en los

TABLA 3.9: Resultados obtenidos al clasificar los datos originales y cuando se adicionaron atributos correlacionados.

Técnicas	Algoritmos	Original (%)	Fuente Correlacionada (%)
Rules	ConjuntivesRules	61.19	7-62.50
	Decision Table	80.00	3-94.79
	DTNB	80.00	5-91.67
	JRip	80.00	4-95.83
	NNge	73.57	6-79.17
	OneR	73.33	8-70.83
	Part	80.48	1-96.88
	Ridor	78.81	2-93.75
	ZeroR	36.19	9-31.25
Functions	Logistic	76.67	5-76.04
	MultilayerPerceptron	78.81	4-82.29
	RBFNetwork	76.67	3-80.21
	SimpleLogistic	76.19	2-82.29
	SMO	76.67	1-83.33
Trees	BFTree	69.00	4-90.00
	FT	79.00	5-91.00
	J48	73.00	2-97.00
	J48graft	71.00	1-97.00
	Random Forest	75.00	3-98.00

resultados obtenidos. La herramienta utilizada para la obtención de estos resultados fue Weka[1]. A continuación describiremos el orden y el propósito de cada uno de los experimentos diseñados. Primero, se seleccionó un conjunto de datos ideal teniendo en cuenta el conocimiento de los expertos, es decir, sin ninguno de los problemas de calidad mencionados previamente, de tal manera que se obtenga la mejor clasificación posible. A continuación, fueron aplicados un conjunto de algoritmos (Tabla 3.9) a la fuente de datos ideal (fuente de datos sin ningún problema de calidad), y a otra fuente de datos alterada para lograr una alta correlación entre sus atributos. Se añadieron números consecutivos para definir el orden de los algoritmos correctamente clasificados en comparación con el original. La fila que contiene el número uno fue el algoritmo con el mejor resultado en relación con los resultados obtenidos en el archivo original. El siguiente paso fue obtener varias fuentes de datos a partir de variar la presencia de cada uno de los criterios de calidad en la fuente de datos original. La distribución fue la siguiente:

1. Completitud:

[1] http://www.waikato.ac.nz/ml/weka/

   a) 10 % de valores nulos para cada una de las clases existentes en el atributo a predecir

   b) 30 % de valores nulos para cada una de las clases existentes en el atributo a predecir

   c) 50 % de valores nulos para la clase *Base* (ver Tabla 3.8)

2. Correlación:

   a) Inclusión de atributos de *Entrada* que se conocen de antemano están correlaciondados. (Tabla 3.9)

3. Datos Balanceados y Desbalanceados: Se realizaron experimentos con tres fuentes de datos distintas, considerando que en nuestro caso de estudio existen tres clases del atributo a predecir, en este caso el atributo Posición:

   a) Fuentes de datos con un número desbalanceado de jugadores *Base* respecto a los *Alero* y *Pivot*

   b) Fuentes de datos con un número desbalanceado de jugadores *Alero* respecto a los *Base* y *Pivot*

   c) Fuentes de datos con un número desbalanceado de jugadores *Pivot* respecto a los *Base* y *Alero*

Para cada clase, la cantidad de elementos de la clase desbalanceada fue modificada, distribuyéndose la cantidad de registros de cada clase teniendo en cuenta los siguientes porcentajes en relación con el total: 40, 30 y 30; 50, 25 y 25; 60, 20 y 20 (ver Tabla 3.10).

1. Desbalance y valores nulos:

   a) Desbalance del 10 % de datos nulos para cada clase existente del atributo a predecir.

   b) Desbalance del 30 % de datos nulos para cada clase existente del atributo a predecir.

   c) Desbalance del 50 % de datos nulos para cada clase existente del atributo a predecir.

2. Desbalance y correlación: Utilizamos los mismos archivos para la variante de desbalance pero ahora incluyendo los atributos correlacionados.

TABLA 3.10: Resultados del clasificador para *Árboles* con datos desbalanceados.

Clase	Algoritmos	Original (%)	40, 30 y 30 %	50, 25 y 25 %	60, 20 y 20 %
Base	BFTree	80.70	2-80.50	5-69,80	5-67.90
	FT	77.10	1-80.70	1-69.30	1-69.80
	J48	81.40	5-80.00	2-71.90	2-71.70
	J48graft	81.40	5-80.00	2-71.90	2-71.70
	Random Forest	80.70	4-80.20	4-70.00	4-69.00
Alero	BFTree	80.70	4-78.30	4-73.57	1- 82.38
	FT	77.10	1-78.10	1-73.10	2-78.57
	J48	81.40	2-80.71	2-74.52	4-81.19
	J48graft	81.40	2-80.71	2-74.52	5-81.19
	Random Forest	80.70	4-78.33	5-71.90	3-81.67
Pivot	BFTree	80.70	3-80.95	5-76.43	2-74.76
	FT	77.10	1- 80.00	1-76.43	1-77.14
	J48	81.40	4-78.33	2-78.33	3-75.00
	J48graft	81.40	4-78.33	2-78.33	3-75.00
	Random Forest	80.70	2-82.14	4-77.14	5-73.81

3. Correlación y valores nulos: Utilizamos los mismos archivos para la variante de valores nulos pero ahora incluyendo los atributos correlacionados.

3.1.5.7. Resultados Obtenidos

A continuación serán abordadas las principales conclusiones obtenidas a partir de la experimentación. Como se puede observar en la Tabla 3.8, de manera general los porcentajes de correcta clasificación de las instancias se reducen significativamente debido a la presencia de atributos nulos en el conjunto de datos. Teniendo en cuenta que la modificación de los valores nulos en los casos 2 y 3 (10 y 30 %) fue homogénea en todas las clases del atributo a predecir; se puede concluir que la presencia de valores nulos homogéneamente añadidos a un conjunto de datos no afecta considerablemente los resultados del clasificador. Los porcentajes de instancias correctamente clasificadas apenas varían en la fuente de datos original, pero si el atributo es afectado en unas de las clases, el porcentaje de las instancias correctamente clasificadas disminuye considerablemente.

Tabla 3.11: Resultados del clasificador para *Funciones* con datos desbalanceados y valores nulos.

Clase	Algoritmos	Original (%)	Desb. y 10 % valores nulos	Desb. y 30 % valores nulos	Desb. y 50 % valores nulos
Base	Logistic	76.67	3-74.42	1-76.92	2-74.24
	Multilayer Perceptron	78.81	5-66.28	2-76.92	4-72.73
	RBFNetwork	76.67	4-70.93	4-69.23	5-65.15
	SimpleLogistic	76.19	2-74.42	5-66.67	1-74.24
	SMO	76.67	1-75.58	3-71.79	3-72.73
Alero	Logistic	70.49	1-83.64	4-78.82	5-78.69
	Multilayer Perceptron	70.49	1-83.64	1-82.35	2-83.61
	RBFNetwork	63.93	5-70.91	5-70.59	4-75.41
	SimpleLogistic	75.41	3-85.45	3-85.88	3-88.52
	SMO	72.13	4-81.82	2-83.53	1-91.80
Pivot	Logistic	71.05	1-73.91	4-60.78	4-55.26
	Multilayer Perceptron	84.21	4-78.26	2-76.47	3-71.05
	RBFNetwork	81.58	5-63.77	5-66.67	5-63.16
	SimpleLogistic	82.89	3-81.16	1-80.39	1-73.68
	SMO	84.21	2-82.61	2-76.47	2-73.68

Este análisis fue posible, ya que sabíamos la cantidad original de los atributos de cada clase en las fuentes de datos. Este conocimiento no se conoce previamente cuando nos enfrentamos a una fuente de datos real. Cuando no se conoce la cantidad de elementos de cada clase, sólo se conoce el porcentaje que representa los atributos con valores nulos del total de elementos, y múltiples combinaciones de los elementos no nulos, con el mismo porcentaje de valores nulos en relación con el total, podrían aparecer. Por ejemplo, en nuestra fuente de datos, cuando nosotros adicionamos el 30 % de valores nulos a una misma clase (*Base*), esto representa la misma cantidad que si se hubiera adicionado el 10 % de valores nulos para cada clase (*Base*, *Alero* y *Pivot*), y los resultados obtenidos son muy diferentes. Por lo que debe considerarse la cantidad de elementos nulos que presentan cada uno de los atributos presentes en la fuente de datos. Esto debe tenerse en cuenta debido a que cuando la cantidad de valores nulos crecen de forma no homogénea, otro problema es introducido: el desbalance entre las clases (ver caso 4 en la Tabla 3.8)). Finalmente, los resultados muestran un notable impacto debido a la presencia de clases desbalanceadas en relación con el resto de las clases. (ver Tabla 3.8).

Para el caso de experimentos con datos desbalanceados (ver Tabla 3.10), la situación depende de la cantidad de instancias que fueron correctamente clasificadas en las fuentes de datos. Debido a esto, los resultados son correctos si la distribución de los datos para cada clase es uniforme tanto como sea posible. Teniendo los resultados obtenidos en cuenta y la cantidad de combinaciones en las fuentes de datos, una posible solución es determinar si los datos están balanceados o no, de acuerdo a la cantidad de clases diferentes existentes. Los datos desbalanceados tienen un impacto a nivel global. En algunos casos, los cambios adicionados a las fuentes de datos implican un sobreajuste en el modelo de clasificación obtenido. Un análisis mas profundo puede confirmar que el algoritmo FT siempre tiene los mejores resultados respecto a los otros algoritmos. Esto permite confirmar que, para este caso de estudio, el uso del algoritmo FT puede ser sugerido cuando haya presencia de datos desbalanceados.

En el caso de la correlación (ver Tabla 3.9), los porcentajes de clasificación fueron obstensiblemente mejor, considerando que los atributos adicionados como *Entrada* en las fuentes de datos fueron los más influyentes. La solución debe ser la creación de un mecanismo que alerte la introducción de atributos fuertemente correlacionados como *Entrada/Salida*. Cuando se aplicó la clasificación a los datos desbalanceados y nulos (Tabla 3.11), se pudo notar que los porcentajes de instancias correctamente clasificadas disminuyeron en la medida que aumentaban las cantidades de valores nulos en la fuente de datos. La clasificación mejoró sus resultados en aquellos casos que los valores nulos fueron homogéneamente introducidos en la fuente de datos, confirmando que la cantidad de datos de la clase desbalanceada tiene gran influencia.

Hemos detectado tres aspectos relacionados con la calidad de datos para las técnicas de clasificación que deben ser abordados en etapas iniciales del proceso de descubrimiento de conocimiento. Los aspectos que afectan negativamente la calidad de los datos para minería de datos son: (i) datos correlacionados, (ii) datos altamente desbalanceados, y finalmente, (iii) seleccionar datos acordes al contexto del problema. Los datos correlacionados incrementan la complejidad de los patrones resultantes, pero no proveen información útil. Los datos desbalanceados proveen patrones sobre ajustados poco confiables. Finalmente, los atributos de *Entrada* que no hayan sido seleccionados siguiendo un criterio del dominio que se está tratando, pueden influir en la obtención de patrones más complejos, incluso con información no útil o novedosa.

Los experimentos realizados nos permitieron demostrar nuestra hipótesis: conocer el comportamiento de los diferentes subgrupos de técnicas de clasificación y los distintos algoritmos en diferentes fuentes de datos, ante la presencia de varios criterios de calidad. Para generalizar el diseño a un paso superior de desarrollo, se ha decidido crear una base de conocimiento que almacene todos los conceptos relacionados con la medición de la calidad de datos sobre técnicas de minería. Dicha base de conocimiento permitirá almacenar la información generada por experimentos de minería de datos realizados a lo largo del tiempo, permitiendo de esta manera obtener una homogenización en la recogida de la información para su posterior uso.

3.2. Meta-características a utilizar

Como ha sido descrito en la sección 1.3 las meta-características son elementos que permiten caracterizar una fuente de datos, fundamentalmente por la relación existente con la calidad de los resultados obtenidos al aplicar técnicas de minería de datos.

Múltiples son los trabajos que las han utilizado en varios escenarios 2.4, como también muchas son las que se han definido. En nuestra propuesta hemos seleccionado un conjunto de ellas, fundamentalmente las convencionales, en aras de comprobar que influyen en el proceso de recomendación y permitirnos la estandarización de su uso por el sistema recomendador. Aunque, en el proceso de diseño debe tenerse en cuenta la posibilidad de agregar nuevas meta-características en cualquier momento para su posterior utilización. De manera general utilizaremos las siguientes: número de atributos, número de instancias, número de clases del atributo objetivo, porcentaje de atributos nominales, porcentaje de atributos numéricos, y entropía.

A continuación, será descrito el proceso de diseño de la base de conocimiento para almacenar toda la información que se genera en los experimentos de minería.

3.3. Diseño de la base de conocimiento de minería de datos

En esta sección se propone un enfoque dirigido por modelos para la construcción de una base de conocimiento. El objetivo es almacenar la información generada en experimentos de minería, para poder usarla a posteriori para la minería de datos amigable. El uso del enfoque dirigido por modelos nos permite mantener la uniformidad de la información que se manipula, así como la no dependencia de un sistema específico de gestión de base de datos para gestionar toda la información.

Esta base de conocimiento podrá ser posteriormente enriquecida con la información generada por usuarios expertos al ejecutar experimentos de minería de datos. Permitiendo utilizar la información almacenada con vistas a asistir a usuarios inexpertos en la obtención del modelo de minería adecuado, considerando sus requisitos (ver sección 7.2) y siendo conscientes de las meta-características de los datos de entrada y la evaluación de la fiabilidad de los modelos obtenidos.

Nuestro enfoque se basa en la siguiente hipótesis: si se mantiene la calidad de la fuente de datos original, al aplicar diferentes algoritmos de minería de datos varían los resultados obtenidos. La selección de cierto algoritmo de minería de datos depende de la calidad de las fuentes de datos. Por lo que, se requiere mantener la información sobre el comportamiento de diferentes algoritmos de minería con respecto a la calidad de las fuentes de datos y en general, de sus meta-características.

La información recopilada de los experimentos de minería realizados puede ser útil para ayudar a los usuarios inexpertos en el proceso de toma de decisiones. Nuestra base de conocimiento tiene como objetivo representar de una manera estructurada y homogénea todos los elementos necesarios de minería de datos que se presentan en el proceso KDD. Para lograr este objetivo, nuestra base de conocimiento almacenará la siguiente información:

Información acerca de las fuentes de datos de entrada. Los metadatos de las fuentes de datos deben ser conocidos. Por ejemplo, número de atributos y de instancias, así como el tipo de datos de cada uno de los atributos que la componen.

Resultados al aplicar cada algoritmo de minería. Se almacenará la información relacionada con la ejecución de cada algoritmo de minería aplicado,

además del tipo de técnica ejecutada y el atributo a predecir (en caso de ser necesario).

Criterios de calidad de datos. Varios criterios de calidad deben ser medidos en las fuentes de datos, ya sean relacionados con la fuente de datos en general (por ejemplo, porcentaje de valores nulos), como de atributos en específico (por ejemplo, correlación entre atributos).

A continuación se describirá los mecanismos que sustentan la propuesta para su correcta ejecución.

3.3.1. Metamodelado para la creación de la base de conocimiento

En este apartado vamos a explicar cada uno de los elementos que conforman nuestro metamodelo de minería. Siguiendo el paradigma dirigido por modelos, nuestra base de conocimiento es uniforme y es creada automáticamente, siendo un repositorio de modelos que se ajustan al metamodelo diseñado para representar la información anteriormente descrita.

Todos los datos relacionados con la información mencionada sobre los experimentos de minería de datos (metadatos de las fuentes de datos, resultados de los algoritmos de minería de datos, y los valores de los criterios de calidad de datos) son adecuadamente representados en un modelo, con el objetivo de ser lo más genérico posible. Nuestros modelos no están restringidos a determinados criterios de calidad, ya que el metamodelo soporta la creación de nuevos criterios de calidad según sea necesario. Los elementos que componen nuestro metamodelo (ver Fig. 3.5) se basan en un análisis de varias ontologías (véase la sección 2.3).

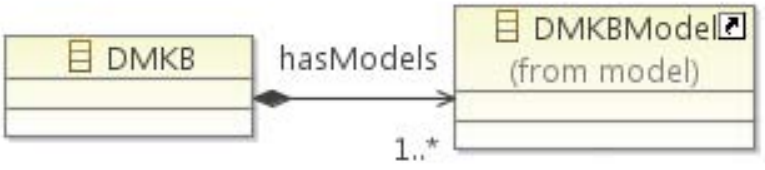


Figura 3.5: Metamodelo principal.

Para simplificar la propuesta con el fin de mejorar el rendimiento del procesamiento durante la ejecución, se decidió diseñar el metamodelo con una característica especial. Ser capaz de diseñar un metamodelo único que agrupe a todos los modelos generados, y a su vez, poder acceder a los modelos de cada fuente de datos analizada de manera independiente.

El metamodelo general (ver Fig. 3.5) representa la base de conocimiento. Contiene una colección de elementos *DMKBModel*, a través de la relación hasModels. El metamodelo *DMKBModel* (ver Fig. 3.6), es el que permite almacenar cada modelo creado para cada fuente de datos analizada. Entonces, el metamodelo general *DMKB* es una colección de los modelos individuales *DMKBModel*. (ver Fig 3.5 y Fig. 3.6)

DMKB. Es la clase principal que representa la base de conocimiento de minería de datos. A partir de la relación `hasModels` contiene elementos del tipo `DMKBModel`.

DMKBModel. Es la clase que contiene todos los elementos necesarios para representar la base de conocimiento de minería de datos (*DMKB*). Es la clase que reúne toda la información que se genera después de analizar una nueva fuente de datos. Permite la especificación de un modelo en el que la siguiente información puede ser almacenada: fuente de datos de entrada, metadatos, algoritmos de minería de datos, configuración de parámetros, resultados de minería de datos que se generan, y los valores de los criterios de calidad de datos evaluados.

DataSet. Describe las fuentes de datos usadas para generar la información que se incluirá en la base de conocimiento. Cada `DataSet` está compuesto por diferentes atributos. Además, cada fuente de datos contiene una categoría y un conjunto de metadatos.

Field. Representa la información de cada atributo contenido en el `DataSet`. Este segmento de dato es identificado por un nombre. Además, la categoría debe ser definida (a partir de una enumeración denominada *FieldKind*) y su tipo (por medio de una enumeración llamada *FieldType*). Esta clase contiene un conjunto de valores de calidad de datos que están relacionados con el atributo.

FieldKind. Es una clase de enumeración para definir el tipo general de los valores que las instancias de campos pueden tener (contínuo, categórico o mixto).

FieldType. Es una clase de enumeración para representar el tipo de cada `Field` (numérico, fecha, nominal o cadena).

DataMiningResults. Esta clase representa los valores de las medidas para cada conjunto de datos después de la ejecución de un algoritmo (por ejemplo la precisión).

Algorithm. Esta clase representa la información acerca de los algoritmos de minería ejecutados. Cada algoritmo pertenece a una técnica específica. (por ejemplo *NaiveBayes*, *J48*, *RandomTree* or *Adaboost*).

Parameter. Esta clase representa los valores de los parámetros iniciales, para la ejecución de un algoritmo. Esta clase contiene el nombre del parámetro y su valor.

Technique. Esta clase define un conjunto de técnicas de minería de datos existentes (Por ejemplo, un árbol, una matriz de probabilidad, etc). Contiene un atributo `subGroup` en el caso de que la técnica lo requiera.

ProblemKind. En él se definen los diferentes tipos de problemas con los que las necesidades del usuario pueden satisfacerse (por ejemplo, clasificación, predicción, clustering, etc.)

DataQualityCriteria. Es una clase abstracta que representa la información relacionada con los diferentes criterios que se pueden presentar, ya sea en un `DataSet` (*DatasetDataQualityValue*) o en cada *Field* (*FieldDataQualityValue*). Para cada uno de los criterios de calidad de datos, un *ComputationMode* es definido y descrito cómo es calculado (por ejemplo, método de correlación de Pearson), y una *MeasuringUnit* donde se representa la unidad de medida correspondiente.

DatasetDataQualityValue Esta clase hereda de la clase *DataQualityCriteria* y define el valor de un criterio de calidad para un *Dataset*.

FieldDataQualityValue Hereda de la clase *DataQualityCriteria* y representa un valor para su correspondiente clase *Field*.

PredictedField Esta clase hereda de *Field* y es diseñada para identificar aquellos atributos que son usados para predecir en un modelo de minería.

ClassMeasuresValues Representa para cada uno de las instancias del atributo a predecir la variable medida y su correspondiente valor.

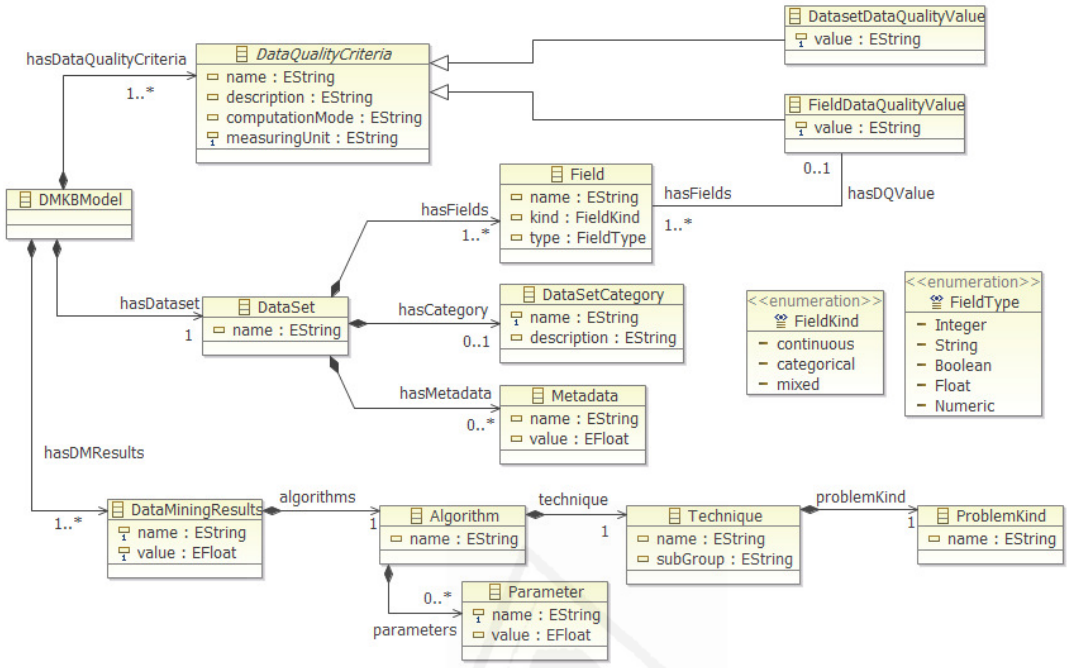


FIGURA 3.6: Metamodelo para una fuente de datos.

La base de conocimiento diseñada nos permitirá aportar la información necesaria para ser usada en la construcción de un sistema recomendador. Dicho recomendador será el responsable de determinar el algoritmo que se aplicará a los datos proporcionados por los usuarios inexpertos y ofrecer, de manera fácil y sencilla, la respuesta más adecuada (modelo de minería) a las necesidades de cada usuario. En el próximo capítulo se muestra el uso de la base de conocimiento para apoyar la toma de decisiones.

Capítulo 4

Propuesta para la obtención de conocimiento por parte de usuarios inexpertos

Ante la gran cantidad de datos y la necesidad de su análisis por personas inexpertas, tal y como se menciona en el Capítulo 1, se ha determinado que las metacaracterísticas juegan un papel importante en los resultados obtenidos al aplicar técnicas de minería [71, 138] (ver Capítulo 3). Ante estos antecedentes, se hace preciso diseñar una propuesta que permita a usuarios inexpertos, teniendo en cuenta la influencia de la calidad de los datos, obtener conocimiento fiable.

Con el fin de que la propuesta presentada tenga naturaleza colaborativa y sea replicable, todo el proceso será controlado por flujos de trabajos científicos. Los flujos de trabajo científicos son una formalización del proceso *ad-hoc* que un científico puede pasar para llegar de los datos "en bruto" a resultados publicables [139].

Se pretende con la interacción de los usuarios expertos con el flujo de trabajo, esta base de conocimientos vaya creciendo y permita obtener cada vez, mejores recomendadores con vistas a ser usados por usuarios inexpertos para tomar decisiones mejores fundamentadas.

Se detallarán cada uno de los flujos de trabajo que han sido implementados y la lógica del funcionamiento de nuestra propuesta. Específicamente, los flujos de trabajos que han sido diseñados son los siguientes:

- Flujo de trabajo científico que permite a los usuarios expertos alimentar la base de conocimiento a partir de procesar sus fuentes de datos.
- Flujo de trabajo científico que le permite al usuario experto configurar los parámetros y la construcción de un recomendador, para obtener el mejor modelo de minería de datos luego de ser aplicado a una fuente de datos inrtoducida por un usuario inexperto.
- Flujo de trabajo científico que permite a un usuario inexperto, introducir sus datos y obtener su resultado de minería, al ser evaluada la fuente de datos de entrada por el recomendador construido por el usuario experto.

La base de conocimiento se alimenta automáticamente de la información generada por la ejecución de los flujos de trabajo sobre los datos de entrada. Ésta contiene información sobre el comportamiento de los algoritmos de minería de datos en presencia de las meta-características intrínsecas de las fuentes de datos.

Este proceso se realiza de manera automática gracias al uso del desarrollo de software dirigido por modelos, permitiendo que un usuario inexperto pueda obtener conocimiento al analizar sus datos. Los métodos y funcionalidades que permiten manipular los conjuntos de datos de entrada por los flujos de trabajo están implementadas en un proyecto Eclipse EMF diseñado para ello. La relación entre los flujos de trabajo científicos, implementados en la Herramienta de Trabajo Taverna WorkBench y Eclipse EMF se establece mediante servicios web REST-FUL, encargados de transportar toda la información de un lado a otro.

Una visión general de nuestro enfoque es mostrada a través de sus principales contribuciones:

1. Un método para alimentar una base de conocimiento y recopilar toda la información que se considera pertinente para la aplicación de algoritmos de minería de datos (ver Fig. 4.1). Esto será posible gracias al flujo de trabajo científico presentado en la sección 4.2.
2. Mecanismo que proporciona al usuario experto poder configurar los parámetros necesarios para la construcción del sistema recomendador de algoritmos de minería de datos, basado en la base de conocimiento existente (ver Fig. 4.4). El flujo que permite realizar esta tarea es presentado en la sección 4.3.1.

3. Un método para permitir a usuarios inexpertos utilizar de forma transparente un recomendador con el fin de extraer conocimiento de las fuentes de datos disponibles, a través de la selección de un algoritmo de minería de datos concreto. (ver Fig. 4.6). El flujo de trabajo creado para cumplir esta tarea es definido en la sección 4.4.1.

En la sección 1 se presenta una descripción de los principales referentes teóricos utilizados en nuestra propuesta.

4.1. Uso de la base de conocimiento

En esta sección se pretende detallar como el usuario experto podrá alimentar la base de conocimientos. Además se presenta el flujo de trabajo diseñado para ser usado por el usuario experto, y se especifica cada uno de sus componentes.

La Fig. 4.1 muestra el procedimiento ejecutado por el usuario experto para alimentar la base de conocimiento. Este proceso es implementado a través dos procesos: uno que permite medir la calidad de los datos de entrada, y el otro para gestionar la ejecución de los algoritmos de minería.

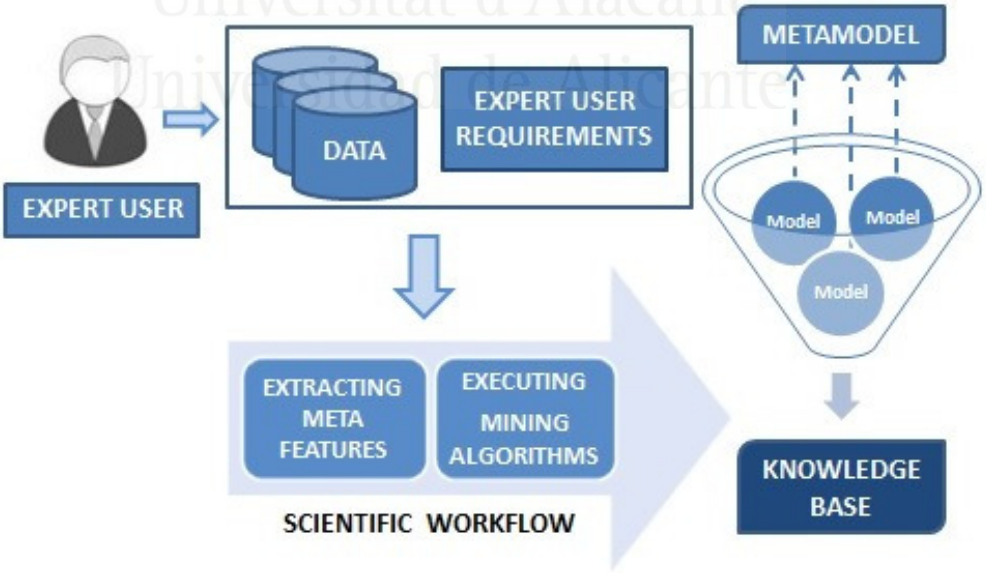


Figura 4.1: Obtención de la base de conocimiento.

El proceso de obtención de la base de conocimiento (Fig. 4.1) comienza cuando el usuario experto adiciona una fuente de datos para que sea analizada. Luego,

introduce sus requisitos (selección de técnicas de minería, atributo a predecir y parámetros generales).

Posteriormente se aplican los dos subflujos de trabajo (los algoritmos de minería y la medición de las meta-características). Toda la información generada por el flujo de trabajo es almacena en un modelo para cada fuente de datos analizada, conforme al metamodelo de minería diseñado, que reúne los conceptos necesarios para la realización de nuestra propuesta. El modelo obtenido se crea de manera automática mediante el uso del enfoque dirigido por modelos.

En resumen, el proceso para llenar la base de conocimientos está compuesto por los siguientes pasos:

1. Adición de la fuente de datos (por parte del experto)
2. Introducción de requisitos del experto
3. Medición de la calidad de datos
4. Ejecución de los algoritmos de minería de datos
5. Introducción de datos en la base de conocimiento

A continuación se introduce el flujo de trabajo que permite a los usuarios expertos enriquecer la base de conocimiento.

4.2. Flujo de trabajo para la creación de la base de conocimiento por usuarios expertos

La alimentación de la base de conocimiento de minería de datos es conducida por el desarrollo de un flujo de trabajo científico implementado en Taverna. Tiene como principal objetivo configurar algunos parámetros de minería y procesar las fuentes de datos con el fin de obtener modelos de minería que serán almacenados en la base de conocimiento. Fig. 4.2.

Para simplificar la comprensión de cada una de las funciones que abarca este flujo de trabajo, lo hemos descompuesto en dos subflujos de trabajos. Cada uno de ellos tiene una responsabilidad específica. A continuación se aborda en detalle cada uno de ellos.

4.2.1. Configuración del flujo de trabajo

El flujo de trabajo comienza con la selección de las fuentes de datos que serán analizadas (hasta el momento sólo archivos *.arff* [1]). Luego, el experto de acuerdo a su experiencia configurará algunos parámetros de minería:

- Técnica de minería que será utilizada.

  En la caja $Select\_DataMining\_Profile$ el usuario experto podrá seleccionar la técnica de minería a aplicar sobre sus ficheros de entrada.

- Existen varias métricas de rendimiento que se tienen en cuenta para evaluar los modelos de minería (ej. exactitud, f-medida, etc.). En la caja

  $Select\_Measure\_of\_Performance$ el usuario experto puede seleccionar la variable a tener en cuenta en los modelos de minería.

- Método para medir el rendimiento del clasificador de minería (ej. *cross validation*, *hold-out*, etc.). Esta selección puede realizarse mediante la caja $Select\_Method\_of\_Performance$.

Una vez que se ha terminado la fase de configuración de los parámetros en la caja $REST\_Get\_Files\_Information$ se extraen los valores de los metadatos de la fuente de entrada (número de atributos, de instancias, porcentajes de atributos nominales y numéricos,etc.). Estos metadatos formarán también parte de la base de conocimiento. Este flujo de trabajo incluye la llamada a dos subworkflows adicionales: *DataQualityWorkflow* y *AlgorithmsWorkflow*.

4.2.2. Subflujo para la aplicación de algoritmos de minería

En este subflujo se implementa la posibilidad de ejecutar los modelos de minería acordes a los parámetros condigurados por el experto

(ver $DataMiningAlgorithm\_NestedWorkflow$). La fuente de datos, el perfil de minería, la medida de rendimiento y el método para la evaluación de los modelos son las entradas del subflujo de trabajo, y la salida son los resultados de minería obtenidos. En la caja $REST\_DataMining\_Algorithms$, es donde la llamada a la

[1] Attribute-Relation File Format (ARFF), un formato de archivo utilizado por la herramienta de minería de datos Weka para almacenar datos.

librería de Weka es realizada. Por ejemplo, si el usuario seleccionó aplicar técnicas de clasificación, sólo se aplicarán los algoritmos pertenecientes a ese grupo.

4.2.3. Subflujo para la medición de criterios de calidad

En este subflujo la calidad de los datos es medida de acuerdo a un conjunto de criterios de calidad de datos previamente establecidos 3.1.1. La entrada será el fichero a procesar y la salida los resultados de los criterios de calidad medidos. Los resultados de los criterios de calidad, así como los valores de los metadatos obtenidos serán elementos importantes en la construcción del recomendador. Por último, toda la información generada en el flujo de trabajo se almacenará en un modelo *DMKBModel* acorde al metamodelo previamente presentado. Para cada conjunto de datos introducidos en el flujo de trabajo se construirán de forma automática su modelo correspondiente. La base de conocimiento estará formada por todos los modelos generados a partir de las fuentes de datos de entrada. Una versión del flujo de trabajo puede consultarse en la siguiente dirección.[2]

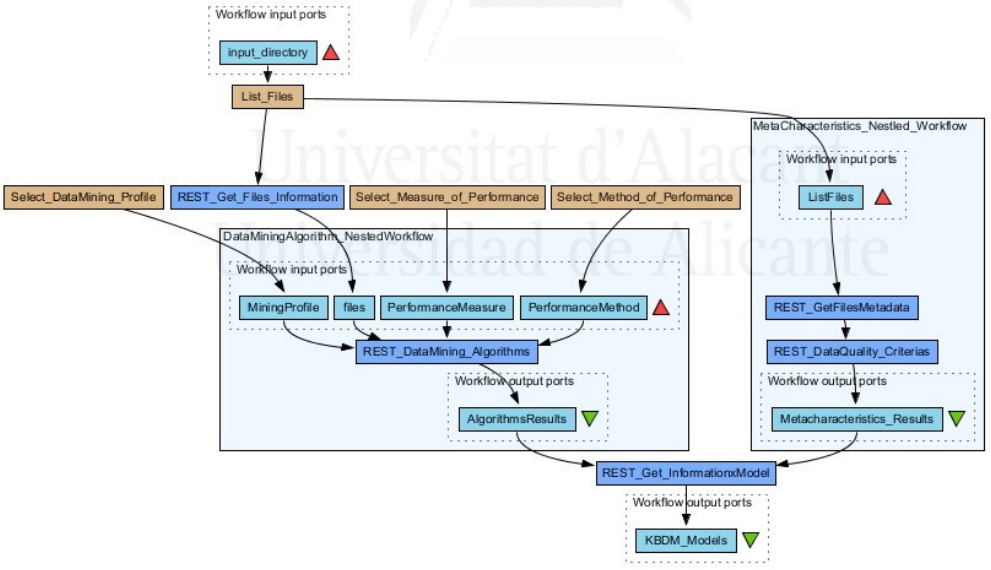


FIGURA 4.2: Flujo de trabajo para el uso por usuarios expertos.

[2]http://www.myexperiment.org/workflows/3843/download?version=3

4.2.4. Creación de los modelos que forman la base de conocimientos

Como se ha mencionado, este flujo de taverna es el encargado de manipular la alimentación de la base de conocimiento, utilizando el desarrollo de software dirigido por modelos a través de la información que se adquiere luego de ejecutar el proceso a las fuentes de datos de entrada.

Con ese objetivo, en esta sección se explicará como se crean los modelos que forman la base de conocimiento. Para ello, el flujo de trabajo interactúa con los métodos implementados a través de un servicio web para llevar a cabo las transformaciones necesarias. Las transformaciones para la generación de los modelos están soportadas por las facilidades que nos ofrece el lenguaje de programación Java, proporcionado por *Eclipse Modeling Framework*.

En el segmento de código 4.1 se muestra un extracto de la transformación implementada para obtener el modelo que formará parte de la base de conocimiento, luego de procesar una fuente de datos de entrada.

Para cada uno de los algoritmos de minería ejecutados las siguientes clases son generadas: `DataMiningResult`, `Algorithm`, `Technique`, y `ProblemKind`; así como las relaciones existentes requeridas entre ellas: `hasDMResults`, `algorithms`, `technique`, y `problemKind`. Finalmente, el modelo (representado por un fichero con extensión *xmi*) es creado. La Fig. 4.3 muestra un ejemplo de modelo `DMKBModel` generado usando nuestra propuesta.

```
1  for (int i = 0; i <= First.listaResAlg.size()-1; i++)
2   {
3     DataMiningResults dmr =kbf.createDataMiningResults();
4     dmr.setName(First.listaResAlg.get(i).requirementName);
5     dmr.setValue(First.listaResAlg.get(i).value);
6     Algorithm alg= kbf.createAlgorithm();
7     alg.setName(First.listaResAlg.get(i).algName);
8     Technique tec=kbf.createTechnique();
9     tec.setName(First.listaResAlg.get(i).technique);
10    tec.setSubGroup(First.listaResAlg.get(i).subgroup);
11    ProblemKind pk=kbf.createProblemKind();
12    pk.setName(probKind);
13    alg.setTechnique(tec);
14    tec.setProblemKind(pk);
15    dmr.setAlgorithms(alg);
16    model.getHasDMResults().add(dmr);
17  }
18 ResourceSet rs = new ResourceSetImpl();
19 rs.getResourceFactoryRegistry().getExtensionToFactoryMap().put("xmi",
20 new XMIResourceFactoryImpl());
21 Resource resource = rs.createResource(URI.createFileURI("ouput_generated/" +
22 ds.getName() + ".xmi"));
23 resource.getContents().add(model);
```

Código 4.1: Segmento de código Java para crear un modelo

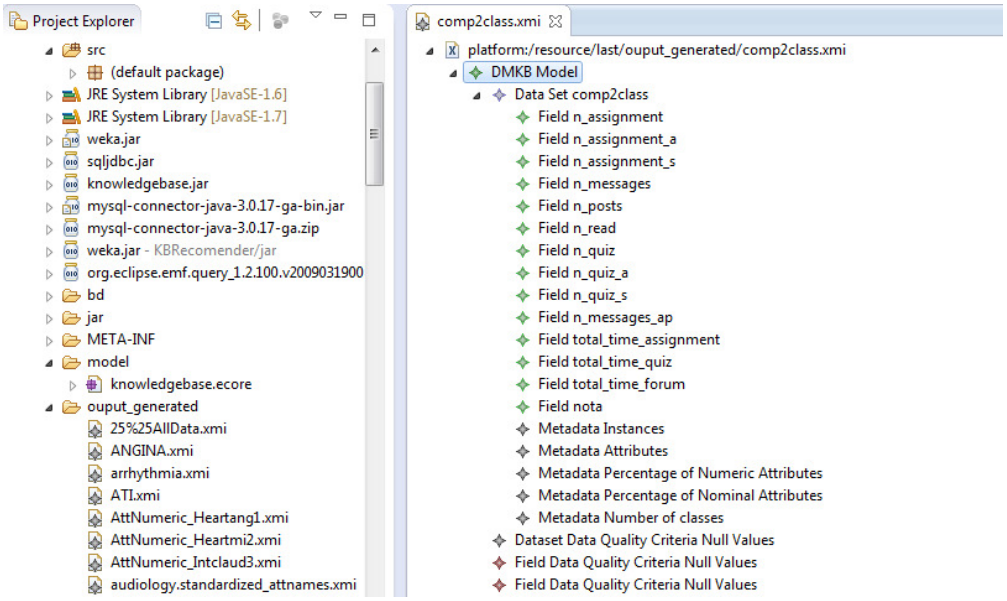


FIGURA 4.3: Ejemplo de un modelo *xmi* creado.

4.3. Construcción del recomendador

En nuestra propuesta el recomendador se nutre de los datos almacenados en la base de conocimiento. Esta tarea se ha delegado en un experto en minería de datos, ya que su precisión depende fuertemente de las instancias elegidas, el algoritmo utilizado y la configuración de sus parámetros.

Nos hemos basado en el meta-aprendizaje para construir el recomendador ya que esta técnica se demuestra adecuada para asistir a los usuarios en la elección del mejor algoritmo para el problema en questión [140, 141]. El recomendador toma como entrada una colección de casos etiquetados, cada uno perteneciente a una de las clases existentes, descrito por sus valores para cada uno de sus atributos, y la salida es un clasificador que puede predecir con exactitud la clase a la que pertenece una nueva instancia [142]. Estos acercamientos están enfocados a comparar algoritmos, solamente han hechos la experimentación desde el punto de vista de la selección de algoritmos. En nuestro caso utilizaremos el sistema recomendador para brindarle una respuesta a los usuarios inexpertos que deseen analizar sus datos y al mismo tiempo satisfacer sus espectativas, permitiéndole de esta forma la toma de decisiones.

El sistema recomendador es construido siguiendo la filosofía *Holdout Set* explicada en la sección 1.2.2. Este método necesita dos conjuntos de datos: uno de entrenamiento y el otro de prueba. El conjunto de entrenamiento es construido a partir de la información proveniente de una parte de los modelos de minería que conforman la base de conocimiento, y el conjunto de prueba es formado además con la información proveniente del análisis de los archivos de datos que el usuario inexperto desea analizar.

Para crear este recomendador, el experto debe seleccionar las instancias de la base de conocimiento más adecuadas para su propósito. La selección puede realizarla teniendo en cuenta algunas de las meta-características almacenadas. Se debe tener en cuenta el dominio de cada modelo de datos y los tipos de usuarios que lo consumirán, por ejemplo, los profesores en el campo de la educación. Inicialmente puede utilizar todas las meta-características almacenadas en la base de conocimiento, pero algunas podrían ser eliminadas durante el proceso, en el caso de no proporcionar información significativa.

El experto debe configurar algunos requisitos de minería antes de construir el recomendador. En concreto, debe establecer el atributo objetivo, el algoritmo de minería a utilizar, y establecer el método de evaluación del desempeño que utilizará. Cuando el experto ejecuta el algoritmo seleccionado en el archivo de datos de entrada, obtiene el modelo de minería.

Este proceso se puede realizar utilizando varios algoritmos y luego se comparan los resultados. Finalmente, se obtiene el modelo adecuado usando los tests estadísticos. El recomendador puede entonces evaluar una nueva fuente de datos, a partir de la información obtenida de la base de conocimiento, obteniendo de esta manera el algoritmo más preciso para ser ejecutado en ella.

La Fig. 4.4 nos muestra el proceso ejecutado por el usuario experto para configurar los parámetros necesarios para la construcción del recomendador.

Al ser la construcción del recomendador un proceso dinámico no descartamos que se puedan incluir otros atributos, incluso utilizar otros clasificadores para la obtención del algoritmo con el mejor rendimiento a ser utilizado. Al aplicar el recomendador con el conjunto de entrenamiento y luego evaluarlo con el conjunto de prueba se genera un tercer archivo donde se muestra para cada instancia del conjunto de prueba, que clase le asignó el clasificador ejecutado al atributo objetivo.

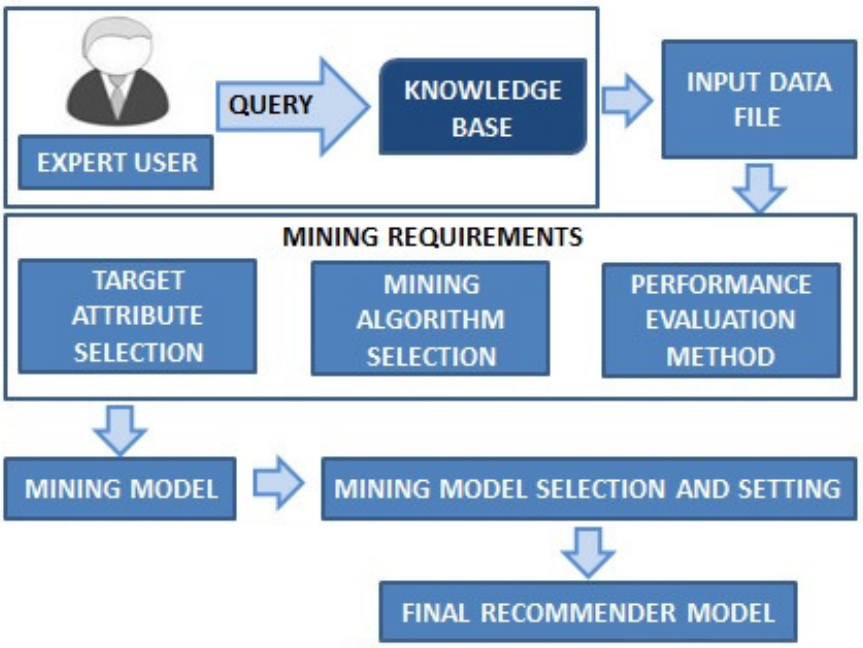


FIGURA 4.4: Pasos para la construcción del recomendador.

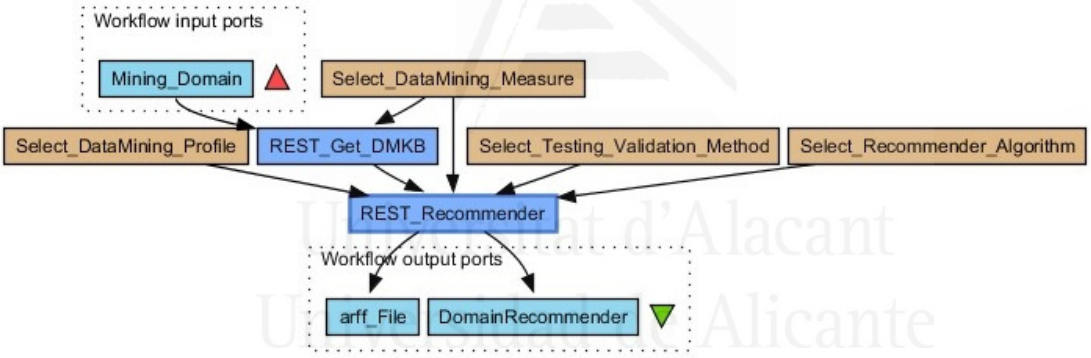


FIGURA 4.5: Fujo de trabajo para la construcción del recomendador.

4.3.1. Flujo de trabajo para la construcción del recomendador

Para construir un buen recomendador de algoritmos de minería el usuario final debe configurar varias acciones. Por esta razón, se ha creado un flujo científico, para que el usuario experto pueda configurar los parámetros y decidir cual debe ser el mejor recomendador, luego de aplicar varios experimentos (ver Fig. 4.5). Lo primero que debe hacer el usuario experto es consultar la base de conocimiento para obtener los datos que formarán parte de su recomendador, por ejemplo, teniendo en cuenta el dominio de los datos, medidas mínimas de exactitud, etc.

Luego, puede configurar cada uno de los siguientes parámetros: seleccionar el perfil de minería (ver caja $Select\_DataMining\_Profile$), medida de minería para evaluar el desempeño del recomendador ($Select\_DataMining\_Measure$), método de validación a utilizar ($Select\_Testing\_Validation\_Method$), y el algoritmo de minería a ejecutar ($Select\_Recommender\_Algorithm$).

La información necesaria para la construcción del recomendador es obtenida de la base de conocimiento. En la caja ($REST\_Recommender$), se ejecuta el recomendador de acuerdo al algoritmo seleccionado por el experto. Teniendo en cuenta que el uso de otro algoritmo pudiera dar mejores resultados, se debe aplicar un proceso de comparación entre varios resultados, para finalmente decidir cual debe ser el modelo a utilizar. El resultado se muestra en la caja $DomainRecommender$. Adicionalmente, el fichero *.arff* obtenido también se brinda como salida al usuario experto (caja $arff\_file$). Una versión del flujo de trabajo puede consultarse en la siguiente dirección del dominio público www.myexperiment.org.[3]

En resumen, el flujo de trabajo para construir el recomendador está compuesto por los siguientes pasos:

1. Selección del dominio para el que se construirá el recomendador
2. Selección de los modelos de la base de conocimiento que formarán parte del análisis
3. Configuración de los parámetros necesarios para la construcción del recomendador: técnica de minería, algoritmo que se utilizará, método de validación, medida de minería que se tendrá en cuenta para la evaluación
4. Aplicación del recomendador (Comparación con otros algoritmos)
5. Dar como salida el modelo de minería obtenido y la fuente de datos en formato *arff*, por si el experto desea realizar otro tipo de análisis

Aunque en este trabajo se utilizan técnicas de clasificación, los flujos están preparados para su futura extensión a otros tipos de técnicas.

[3]http://www.myexperiment.org/workflows/4522/download?version=1

4.4. Uso del recomendador por usuarios inexpertos

La Fig. 4.6 muestra cómo se establece la interacción del usuario inexperto con el flujo de trabajo implementado. Para lograrlo, se capturan los requisitos del usuario y se mide la calidad de los datos del fichero de entrada. Luego se ejecuta el recomendador utilizando la información que proporcionan los modelos que forman la base de conocimiento y en función de la calidad de los datos de entrada. El resultado del sistema recomendador es el modelo de minería que obtuvo el mejor resultado.

Se pretende con el uso de este flujo de trabajo que un usuario inexperto obtenga conocimiento en un tiempo considerablemente corto, respecto al tiempo que consume un experto al realizar este proceso, y permitir tomar decisiones mejor fundamentadas.

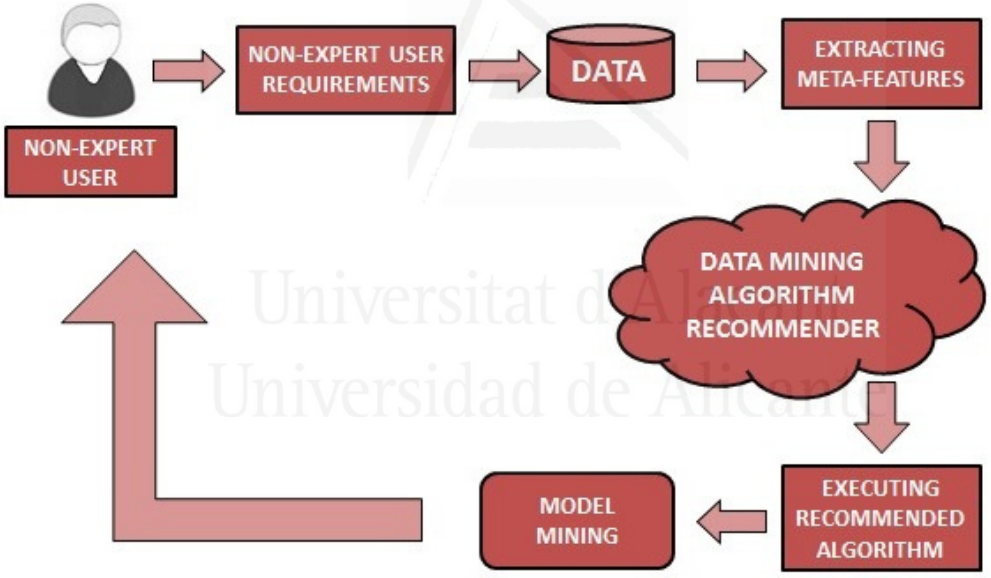


FIGURA 4.6: Uso de la base de conocimiento por el sistema recomendador.

4.4.1. Flujo de trabajo para el uso por usuarios inexpertos

Este flujo de trabajo (Fig. 4.7) es en realidad el que permite que un usuario inexperto sin ningún conocimiento de algoritmos y técnicas de minería de datos pueda analizar sus datos y obtener conocimiento fiable. Todo el proceso es transparente para el usuario, pero el flujo de trabajo es responsable de ejecutar el recomendador diseñado y devolver el mejor resultado de acuerdo a la calidad de la fuente de

datos y los requisitos iniciales del usuario. Una versión del flujo de trabajo puede consultarse en la siguiente dirección.[4]

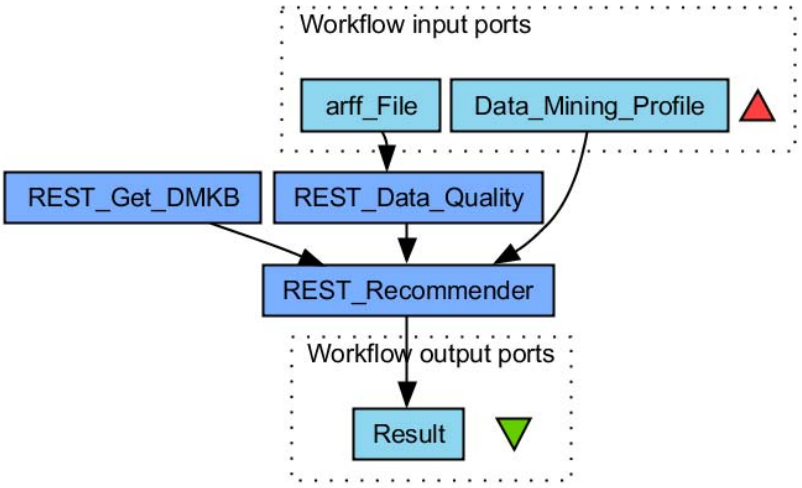


FIGURA 4.7: Flujo de trabajo en Taverna para el uso por usuarios inexpertos.

Partiendo de que asumimos que el usuario no tiene conocimiento de técnicas de minería de datos, cuando el usuario introduce la fuente de datos, con vistas a realizar el proceso de minería, es necesario conocer que desea obtener al analizar sus datos. Esta información adicional, conocida en nuestro medio como requisitos de usuario, será obtenida a partir de simples preguntas que nos guiarán a identificar sus expectativas, para de manera sencilla poder representar que es lo que quiere el usuario obtener de sus datos (ver sección 7.2).

A partir de las respuestas dadas por el usuario, se infiere la técnica específica que quiere aplicar a sus datos (ver la caja *Data_Mining_Profile* que se puede apreciar en la Fig. 4.7). Con estos requisitos obtenidos, se realizará el proceso de forma automática con la calidad adecuada.

Cuando el usuario inexperto carga las fuentes de datos, estas son analizadas previamente, con el fin de extraer la información de sus metadatos y la calidad que presenta, a través de la medición de varios criterios de calidad de datos y meta-características. Una vez que se tiene la información requerida, el recomendador es construido utilizando todos los datos existentes en la base de conocimiento y la información obtenida de la fuente de datos del usuario inexperto.

[4]http://www.myexperiment.org/workflows/3846/download?version=3

Como se puede ver en la Fig. 4.7, para la construcción del recomendador es necesario información que se obtiene a partir de varios componentes. Por un lado, los requisitos del usuario *Data_Mining_Profile*, en este caso, la técnica de minería de datos que el usuario desea aplicar y el atributo a predecir. Por otra parte, los valores obtenidos mediante la aplicación de los criterios de calidad de datos al conjunto de datos de entrada *REST_data_quality*, y, por último, la información proporcionada por la base de conocimiento *REST_Get_DMKB*.

4.4.2. Transformaciones modelo a texto

Como se ha explicado el uso del desarrollo de software dirigido por modelos permite tener almacenada toda la información de manera homogénea. Toda la información que se genera de los ficheros de prueba (aquellos introducidos al sistema por los usuarios inexpertos para poder ser analizados por el recomendador), así como de la base de conocimiento, está almacenada en modelos, específicamente en formato *xmi*. Las operaciones que se deben ejecutar por el recomendador están implementadas para ejecutarse sobre ficheros *arff* válidos, por lo que es necesario aplicar algún mecanismo que permita convertir los modelos *xmi* a ficheros *arff*.

Para implementar dicha transformación usamos Acceleo[5], un generador de código de Eclipse que permite utilizar el enfoque dirigido por modelos para la creación de aplicaciones. Nuestra transformación tomará como modelo de entrada, los modelos *xmi* que forman la base de conocimiento, y como salida, generará un archivo *.arff* válido. Acceleo funciona creando una plantilla donde los segmentos de código que acceden a los elementos del modelo de entrada, y generan el código de salida, son implementados.

```
1  @relation KnowledgeBase
2  @attribute numAtt numeric
3  @attribute numIns numeric
4  @attribute numClasses numeric
5  @attribute hasMissingValues {yes,no}
6  @attribute numMissing numeric
7  @attribute percMissing numeric
8  @attribute isbalanced{balanced,quite_unbalanced,unbalanced}
9  @attribute bestAcc{J48,NaiveBayes,BayesNet,OneR,JRip,Ridor,DecisionTable,NNge}
10 @data
11 25,537,4,yes,4680,0.348603,unbalanced,NNge
12 22,193,2,no,0,0,balanced,BayesNet
13 14,64,2,no,0,0,balanced,NNge
14 12,65,2,no,0,0,balanced,J48
15 21,185,3,yes,1443,0.371429,unbalanced,DecisionTable
16 21,185,4,yes,1443,0.371429,unbalanced,BayesNet
17 21,185,5,yes,1443,0.371429,unbalanced,Ridor
18 22,188,3,yes,1170,0.282882,unbalanced,Ridor
```

[5]http://www.eclipse.org/acceleo

Código 4.2: Segmento del fichero *.arff* creado

La estructura de un archivo *.arff* tiene dos secciones. La primera sección es la información del encabezado, que es seguido por la información de sus instancias. El encabezado del archivo *.arff* contiene el nombre de la relación, una lista de los atributos y sus tipos. Los atributos nominales deben especificarse en la cabecera, junto con sus posibles valores. Un ejemplo de un archivo *.arff* de la base de conocimiento se muestra en el código 4.2.

El código que permite crear el archivo *arff* está compuesto por los siguientes pasos:

1. Crear el encabezado del archivo *arff*
2. Crear la lista de los atributos
3. En el caso de cada modelo de minería, se debe seleccionar el valor del mejor algoritmo almacenado
4. Iterar por los modelos de minería y obtener los valores de los metadatos y de los criterios de calidad

```
[comment encoding = UTF-8 /]
[module generate('http://kb/1.0', 'http://dmkbmodel/1.0')]
[template public generateElement(aKB : DMKB)]
[file ('knowledgebase.arff', false, 'UTF-8')]
@relation ['KnowledgeBase'/]
@attribute numAtt numeric
@attribute numIns numeric
@attribute numClasses numeric
@attribute hasMissingValues {yes,no}
@attribute numMissing numeric
@attribute percMissing numeric
@attribute isbalanced {balanced,quite_unbalanced,unbalanced}
@attribute bestAcc {[for (dm: DMKBModel| aKB.hasModels) separator(',')]
                        [for (dmr: DataMiningResults | dm.hasDMResults)]
                          [if (maximoModelo(dm).toString() = dmr.value.toString())]
                            [dmr.algorithms.name/] [/if][/for][/for]}
@data
 [for (meta:Metadata | dm.hasDataset.hasMetadata) separator (',')]
    [if (meta.name = 'Attributes')]
         [setTotalInstances(meta.value)/] [meta.value/] [/if]
                [if (meta.name = 'Instances')]
         [setTotalInstances(meta.value)/] [meta.value/] [/if]
    [if (meta.name = 'Number of classes')]
         [setTotalInstances(meta.value)/] [meta.value/] [/if]
    [if (meta.name = 'Percentage of Numeric Attributes')]
         [setTotalInstances(meta.value)/] [meta.value/][/if]
    [if (meta.name = 'Percentage of Nominal Attributes')]
         [setTotalInstances(meta.value)/] [meta.value/][/if][/for]
  [for (dsqc: DatasetDataQualityCriteria | dm.hasDataQualityCriteria) separator (',')]
     [if (dsqc.name = 'Null Values')]
       [dsqc.value/]
     [elseif (dsqc.name = 'Average Entropy')]
       [dsqc.value/]
     [elseif (dsqc.name = 'UnbalanceColumns')]
```

```
35          [if (dsqc.value.toReal() >= 70)]
36            true
37          [else]
38            false [/if][/if][/for]
39      [for (dmr: DataMiningResults | dm.hasDMResults)]
40          [if (maximoModelo(dm).toString() = dmr.value.toString())],[dmr.algorithms.name/][/if][/for]
41            [/for]
42  [/file]
43  [/template]
44  [query public maximoModelo(admkb: DMKBModel): Real=
45      admkb.hasDMResults.value-> max()/]
46  [query public setTotalInstances(r1:Real): Boolean = invoke
47  ('org.eclipse.acceleo.module.sample2.files.Utility', 'setTotalInstances(java.lang.Double)',
48  Sequence{r1}) /]
```

Código 4.3: Segmento de código Java para crear la sección de encabezado

En el segmento de código 4.3 se recorre todos los modelos *xmi* que fueron previamente creados con los datos de cada fuente de datos analizada. Se crea una instancia en el archivo de texto *.arff* para cada modelo *xmi* que forma la base de conocimiento. Por ejemplo, en el caso del atributo *bestAcc*, cada modelo tiene los valores obtenidos al aplicar cada uno de los algoritmos que se ejecutaron en el flujo de trabajo. Para el archivo *.arff* utilizado por el recomendador sólo se seleccionó el mejor valor de todos los algoritmos.

El atributo *bestAcc* para cada fuente de datos muestra el algoritmo que obtiene el mejor rendimiento (en nuestro caso de estudio, basado en la precisión) para cada modelo. Para obtener los posibles valores de los algoritmos es necesario iterar sobre todos los modelos, y luego por todos los resultados de minería. Luego se busca el mayor valor de precisión entre los algoritmos ejecutados y se imprime el nombre del algoritmo correspondiente. Para devolver los valores de los atributos *numAtt*, *numIns*, *numClasses*, el mismo procedimiento es realizado. Se itera a través de los elementos *Metadata* y luego de comparar con el nombre del metadato correspondiente, se accede a su valor.

En el caso de los atributos *numMissing* y *isbalanced*, en el modelo son parte de los criterios de calidad de datos, de manera que sus valores son accedidos a través del elemento *DatasetDataQualityCriteria*, específicamente por la relación *hasDataQualityCriteria*, después de comparar con el nombre del criterio de calidad correspondiente.

Dado el carácter colaborativo que pretende tener la propuesta, hemos utilizado la plataforma *MyExperiments* para publicar todos los flujos de trabajos implementados para que sea posible su uso por la comunidad científica internacional. En el pie de página se indican las respectivas *Uniform Resource Locator (urls* de los flujos

de trabajo dadas por la plataforma luego de ser publicados. Los flujos de trabajo se pueden ejecutar manualmente o configurar mediante la línea de comandos.

Capítulo 5

Aplicación de la propuesta a un caso de estudio de e-learning

En este capítulo se presenta la experimentación realizada sobre un caso de estudio con el objetivo de validar la propuesta diseñada. El caso de estudio está centrado en el dominio del e-learning.

Específicamente, la experimentación ha sido aplicada para evaluar la base de conocimiento como recurso para los usuarios inexpertos en minería de datos en el contexto educativo en línea: profesores de los cursos de e-learning son inexpertos en minería de datos que necesitan descubrir quién y cómo sus cursos son utilizados con el fin de mejorarlos.

La minería de datos está siendo utilizada comunmente [143] en el ámbito educativo como consecuencia de la rápida expansión del uso de las tecnologías como apoyo al aprendizaje, no sólo en los contextos y plataformas institucionales establecidas, sino también en los ámbitos de aprendizaje libre y social en línea. Aunque hay herramientas como ElWM [144] que ayudan a los instructores a analizar sus cursos virtuales, la base de conocimiento que aquí se propone se convertirá en un recurso fundamental para el diseño de un sistema recomendador que ayude al docente (como inexperto en minería de datos) a aplicar los algoritmos correctos de minería de datos a sus conjuntos de datos y extraer conclusiones orientadas a mejorar el proceso de enseñanza-aprendizaje.

En esta sección, en primer lugar se describe un resumen de las principales características de las fuentes de datos utilizadas. A continuación, se expone el proceso

previsto para llevar a cabo este experimento y, por último, se muestran y discuten los resultados obtenidos.

5.1. Descripción de las fuentes de datos utilizadas para la experimentación

Con el objetivo de generar el recomendador y mostrar los beneficios de nuestra propuesta, nos centramos en los datos del área educativa. Concretamente, en los datos obtenidos de los cursos de e-learning impartidos en la Universidad de Cantabria. Para nuestra experimentación, hemos utilizado los datos de los cursos de e-learning impartidos por la Universidad de Cantabria con la plataforma *Moodle* en los cursos académicos 2011-2012 y 2012-2013. De cada uno de estos cursos hemos generado cuatro fuentes de datos que tienen la actividad de los alumnos en el curso y como variable de predicción la evaluación otorgada a los estudiantes en estos cursos.

La actividad de los estudiantes es medida por medio de los accesos totales de cada estudiante en cada módulo en *Moodle*; el número de mensajes leídos, escritos y respondidos, y las suscripciones en los foros (así como en el glosario, blogs, wikis y mensajes personales); el número de pruebas de rendimiento y el número de pruebas aprobadas y desaprobadas por separado; y el total de acciones ejecutadas en el curso. Las diferencias entre las fuentes de datos de cada curso es el número de clases a predecir. Para cada fuente de datos del curso tenemos dos, tres, cuatro o cinco clases a predecir, dada la siguiente descripción:

2 clases : *Pass*: nota entre 5 y 10. *Fail*: nota entre 0 y 5 (exclusivos).

3 clases : *Pass*: nota entre 5 y 10. *Fail*: nota entre 0 y 5 (ambos exclusivos). *Dropout*: nota de 0, significa que el estudiante abandonó el curso en algún punto.

4 clases *Highpass*: nota entre 8.5 y 10. *Remarkable*: nota entre 7 y 8.5 (exclusivos). *Pass*: nota entre 5 y 7 (exclusivos). *Fail*: nota entre 0 y 5 (exclusivos).

5 clases *Highpass*: nota entre 8.5 y 10. *Remarkable*: nota entre 7 y 8.5 (exclusivos). *Pass*: nota entre 5 y 7 (exclusivos). *Fail*: nota entre 0 y 5 (ambos exclusivos). *Dropout*: la nota es 0.

Un ejemplo de fuente de datos de un curso con 5 clases es mostrado a continuación:

```
@attribute userAnonimeId String
@attribute N_initiated_disussions_no_deleted numeric
@attribute N_written_post_no_deleted numeric
@attribute N_initiated_discussions numeric
@attribute N_written_posts numeric
@attribute N_subscriptions_forum numeric
@attribute N_updates_forum numeric
@attribute N_updates_posts numeric
@attribute N_readen_discussions numeric
@attribute N_views_forum numeric
@attribute N_views_resources numeric
@attribute N_views_data numeric
@attribute N_attempts_quizzes numeric
@attribute N_views_blog numeric
@attribute N_entrances_blog numeric
@attribute N_views_wiki numeric
@attribute N_entrances_wiki numeric
@attribute N_editions_wiki numeric
@attribute N_personal_messages_written numeric
@attribute N_written_entries_glossary numeric
@attribute N_updated_entries_glossary numeric
@attribute N_views_glossary numeric
@attribute N_comments_glossary numeric
@attribute N_passed_quizzes numeric
@attribute N_failed_quizzes numeric
@attribute N_passed_quizzes_attemps numeric
@attribute N_failed_quizzes_attemps numeric
@attribute N_actions numeric
@attribute qualification {dropout, fail, pass, highpass, remarkable}
@data
495 ,0,0,0,0,0,0,1,0,37,0,0,0,0,0,0,0,0,0,0,0,0,0,0,158,40.00000,fail
4981,0,0,0,0,0,0,2,0,32,0,0,0,0,0,0,0,0,0,0,0,0,0,0,110,?
2219,0,4,0,4,0,0,0,31,46,40,0,0,3,0,0,0,0,0,0,0,0,0,0,0,335,50.00000,pass
...
```

Código 5.1: Ejemplo de una fuente de datos extraída de un curso de Moodle

Teniendo en cuenta que estas fuentes de datos sólo pueden tener como número de instancias los estudiantes que estén matriculados en ellos, y este número usualmente es menor que 100, con el fin de tener una experimentación extensa con fuentes de datos que podrían tener un alto número de casos, a partir de estas 84 fuentes de datos iniciales (84 cursos * 4 combinaciones de valores de clase), generamos 336 nuevas fuentes de datos mediante la combinación de, al azar, las instancias iniciales. Por lo tanto, para los experimentos se trabajó con 336 fuentes de datos con diferentes características entre ellos, ya que tienen una gran variación en el número de instancias y atributos, número de valores perdidos, etc.

5.2. Proceso de experimentación

Para nuestra experimentación, el proceso fue dividido en los siguientes pasos:

Paso 1 Se extrajo la información relacionada con la actividad de los alumnos en los cursos de e-learning que se imparten en la Universidad de Cantabria y que se

ofrecen en *Moodle*, y luego generamos las fuentes de datos que se describen en la sección anterior.

Paso 2 Con las fuentes de datos mencionadas, hemos generado, utilizando Taverna (como se describe en la sección 4.2) una base de conocimiento, con la meta-información de cada fuente de datos, descrita en la sección 3.3, y como valor de la clase el algoritmo de clasificación que alcanzó la más alta precisión al aplicarle algoritmos de clasificación a la fuente de datos. Eso significa que, el valor de la clase será el clasificador que tiene la exactitud más alta entre ellos.

Paso 3 A partir de la base de conocimiento creada, se extrajo sus metadatos y se evaluó el valor para la clase base con vistas a obtener un modelo meta-clasificador o sistema recomendador, que contendrá la información acerca de cuál es el mejor algoritmo de clasificación a ser usado por una fuente de datos concreta dadas sus meta-características. Por ejemplo, hemos utilizado la mayoría de los instancias como conjunto de entrenamiento para generar este recomendador y separamos algunas de las instancias como conjunto de prueba con la meta-información de las fuentes de datos con el objetivo de probar el recomendador. En el código 4.3 se puede apreciar la transformación de modelo a texto que fue necesaria implementar para obtener los datos almacenados en la base de conocimiento hacia el fichero *arff* que será usado para el recomendador.

Paso 4 Debido al conocimiento que se tiene como experto en minería de datos y su aplicación en el campo educativo, conocemos el comportamiento del mejor modelo de clasificación para algunos de los cursos o conjuntos de datos utilizados como prueba en el paso anterior, a fin de establecer la diferencia que, en términos de precisión, podría existir entre el modelo generado por el clasificador seleccionado por nuestro recomendador y el modelo de clasificación que podemos lograr con nuestra experiencia, y finalmente concluir si el tiempo y recursos invertidos para ello son necesarios, o al contrario, el clasificador seleccionado por nuestro recomendador es lo suficientemente bueno para construir y mostrar la información requerida.

TABLA 5.1: Meta-características de las fuentes de datos para construir el meta-clasificador

Nombre	Descripción	Rango
N# of at.	Número de atributos (excluyendo el atributo clase)	4-25
N# of ins.	Número de instancias	25-543
N# of classes	Número de clases	2-4
% valores nulos	Porcentaje de valores nulos	0-30 %
Balanceo	Indica si la clase base está balanceada	balanced, quite_unbalanced, unbalanced

5.3. Recomendador

Con el fin de construir un recomendador, en nuestra experimentación utilizamos las meta-características de las fuentes de datos que se describen en la tabla 5.1, siguiendo el proceso expuesto en la sección 4.3.1 usando Taverna. El rango indica los valores máximos y mínimos que tienen las meta-características en las fuentes de datos, y si es numérico o la posible lista de valores si es nominal.

5.4. Discusión de los resultados obtenidos

Siguiendo el procedimiento descrito en la sección 5.2, una vez que se han generado las fuentes de datos que se describen en la sección 5.1, el siguiente paso consiste en construir una meta fuente de datos que contiene como instancias las meta-características que se describen en la sección 3.3.1 de cada una de las fuentes de datos mencionadas, y como atributo de clase el clasificador, entre los utilizados que ha devuelto el mejor modelo de predicción basado en la precisión, o lo que significa lo mismo, el clasificador que ha obtenido la más alta precisión para cada fuente de datos. A continuación se muestra una pequeña muestra de la meta fuente de datos, con 336 instancias (el mismo número de fuentes de datos que teníamos):

```
1  @relation MD
2  @attribute numAtt numeric
3  @attribute numIns numeric
4  @attribute numClasses numeric
5  @attribute bestAcc {J48,NaiveBayes,BayesNet,OneR,JRip,Ridor,DecisionTable,NNge}
6  @attribute hasMissingValues {yes,no}
7  @attribute numMissing numeric
8  @attribute percMissing numeric
9  @attribute isbalanced {balanced,quite_unbalanced,unbalanced}
10 @data
```

```
14,72,2,J48,no,0,0,quite_unbalanced
17,504,2,JRip,no,0,0,quite_unbalanced
18,163,2,DecisionTable,no,0,0,balanced
12,84,2,OneR,no,0,0,balanced
14,136,2,OneR,no,0,0,unbalanced
...
```

CÓDIGO 5.2: Ejemplo de fuente de datos extraída de los cursos de Moodle

Como se puede observar, por ejemplo, para el primer caso, que representa una fuente de datos que consta de 14 atributos, 72 instancias, 2 clases, no tiene valores nulos y presenta un ligero desbalance, el algoritmo de clasificación que alcanzó el mejor modelo en términos de exactitud o precisión fue el *C4.5* (*J48*). Una vez que hemos obtenido la fuente de datos con las meta-caracteríticas, el siguiente paso consiste en generar el recomendador que tiene que predecir e indicar qué algoritmo de clasificación se debe utilizar con el fin de predecir la evaluación de un alumno en un curso de e-learning, dadas las características de sus datos.

Para ese propósito, siguiendo el enfoque *Holdout Set* descrito en la sección 1.2.2 con el fin de validar el modelo de clasificación, hemos elegido como conjunto de prueba 8 casos que representan 8 fuentes de datos diferentes (cursos) con características diferentes entre ellas y distintos clasificadores como valores de clase. Estos casos del conjunto de prueba se han utilizado para evaluar la bondad del recomendador en términos de precisión. Los 328 casos restantes, que componen el conjunto de entrenamiento, se han utilizado para capacitar y generar el recomendador mencionado. Los casos de prueba establecidos son los siguientes:


```
@relation TestMD
@attribute numAtt numeric
@attribute numIns numeric
@attribute numClasses numeric
@attribute bestAcc {J48,NaiveBayes,BayesNet,OneR,JRip,Ridor,DecisionTable,NNge}
@attribute hasMissingValues {yes,no}
@attribute numMissing numeric
@attribute percMissing numeric
@attribute isbalanced {balanced,quite_unbalanced,unbalanced}
@data
12,126,2,J48,no,0,0,quite_unbalanced
22,1488,2,NNge,no,0,0,quite_unbalanced
22,1488,3,NNge,no,0,0,unbalanced
20,83,4,BayesNet,no,0,0,unbalanced
21,115,5,J48,no,0,0,unbalanced
22,217,3,Ridor,no,0,0,unbalanced
18,231,5,NNge,yes,1048,0.252044,unbalanced
19,836,2,J48,yes,4708,0.296399,balanced
```

CÓDIGO 5.3: Fuente de datos utilizada como conjunto de prueba

Se puede observar, como hemos comentado, las características son diferentes entre las instancias del conjunto de prueba. Por ejemplo, la primera instancia representa una fuente de datos que tiene 12 atributos, 126 archivos o instancias, 2 clases y

está ligeramente desbalanceada, siendo *J48* el mejor algoritmo de clasificación en términos de precisión; mientras que la tercera instancia representa una fuente de datos con 22 atributos, 1488 archivos, 3 clases y la fuente de datos está muy desbalanceada, siendo *NNge* el mejor algoritmo de clasificación. El conjunto de prueba tiene también instancias que representan conjuntos de datos con valores nulos (los últimos tres) con diferentes algoritmos de clasificación como los mejores para clasificar a las fuentes de datos con estas características.

En este punto, tuvimos que decidir qué algoritmo de clasificación debemos usar para construir el recomendador. Teniendo en cuenta que debe ser fácil de interpretar (los usuarios finales son inexpertos en temas de minería de datos) se consideró que se debía usar los algoritmos con modelos más simples. Una buena opción es usar algoritmos de clasificación basados en árboles y, entre ellos, uno de los más utilizados es el algoritmo *C4.5* (*J48*). En la Fig. 5.1 mostramos un segmento del árbol generado con los resultados del recomendador construido con *C4.5*. A partir de este recomendador se puede inferir predicciones futuras sobre cuál será el mejor algoritmo de clasificación para una fuente de datos con características específicas. Después, por ejemplo, la rama más justa del árbol, se puede observar que este recomendador nos indica que, si la fuente de datos contiene 137 o menos instancias, tiene 13 atributos o menos y la fuente de datos en general está balanceada, entonces el recomendador sugiere el algoritmo OneR, en lugar de otros.

Por otro lado, después de la siguiente ramificación, si el número de atributos de una fuente de datos es mayor que 13, el recomendador dependerá de nuevo del número de instancias: si es 96 o inferior, entonces se recomienda *NNge*, pero si es mayor que 96 el algoritmo recomendado es *J48* (y hay que recordar que en las ramas más altas tenemos otra condición que impone que el número de instancias es de 137 o menos, por lo que el recomendador sugiere usar el *J48* estando condicionado a un número de instancias entre 97 y 137). Haciendo uso de este recomendador, Taverna puede mostrar de una manera sencilla los resultados a un usuario inexperto en minería de datos y le recomendaría el algoritmo de clasificación para ser usado con sus datos, sin necesidad de hacer uso de un experto en minería de datos. Sin embargo, tenemos que saber la fiabilidad o rendimiento de este modelo, que es medido en nuestro caso de estudio, por la exactitud o precisión. El valor de precisión fue de un 75 %, lo que significa que, a partir de las 8 instancias iniciales del conjunto de prueba, el recomendador clasificó correctamente 6 instancias y recomendó un algoritmo de clasificación que no era el mejor, en 2 de ellos. En la tabla 5.2 podemos ver, para

```
=== Classifier model (full training set) ===

J48 pruned tree
------------------

numIns <= 358
|   numAtt <= 20
|   |   numIns <= 137
|   |   |   numAtt <= 18
|   |   |   |   isbalanced = balanced
|   |   |   |   |   numAtt <= 13: OneR (6.0/3.0)
|   |   |   |   |   numAtt > 13
|   |   |   |   |   |   numIns <= 96: NNge (3.0/1.0)
|   |   |   |   |   |   numIns > 96: J48 (2.0/1.0)
|   |   |   |   isbalanced = quite_unbalanced
|   |   |   |   |   numAtt <= 12
|   |   |   |   |   |   numClasses <= 2: J48 (3.0/1.0)
|   |   |   |   |   |   numClasses > 2
|   |   |   |   |   |   |   numAtt <= 8: J48 (3.0/1.0)
|   |   |   |   |   |   |   numAtt > 8: BayesNet (11.0/5.0)
|   |   |   |   |   numAtt > 12
|   |   |   |   |   |   numClasses <= 2
|   |   |   |   |   |   |   numIns <= 72: J48 (2.0/1.0)
|   |   |   |   |   |   |   numIns > 72: NaiveBayes (3.0/2.0)
|   |   |   |   |   |   numClasses > 2: JRip (8.0/5.0)
|   |   |   |   isbalanced = unbalanced
|   |   |   |   |   numClasses <= 2
|   |   |   |   |   |   numIns <= 73: J48 (2.0/1.0)
|   |   |   |   |   |   numIns > 73: OneR (3.0/1.0)
```

FIGURA 5.1: Segmento del árbol generado con los resultados del recomendador.

cada caso, que algoritmo de clasificación ha recomendado nuestro recomendador. Este resultado se puede obtener generando el fichero *arff* con los resultados del recomendador luego de ser aplicado con las instancias de prueba A. Como se puede observar, las instancias número 1, 2, 3, 5, 6 y 7 fueron clasificadas correctamente. La instancia número 4 tiene como clase *DecisionTable*, pero nuestro recomendador sugiere usar el algoritmo *BayesNet*, que no es el mejor clasificador, en términos de precisión, para esta fuente de datos. Algo similar ocurre con la instancia número 8. Pero, incluso cuando el recomendador no devolvió el mejor clasificador para los dos casos, su recomendación podemos ubicarla también entre las mejores.

Tabla 5.2: Recomendación para cada instancia del conjunto de prueba utilizando el algoritmo *J48* en el recomendador

Dataset	Best Classifier	Recommended Clasiffier
1	J48	J48
2	NNge	NNge
3	NNge	NNge
4	DecisionTable	BayesNet
5	J48	J48
6	Ridor	Ridor
7	NNge	NNge
8	DecisionTable	J48

Tabla 5.3: Recomendación para cada fuente de datos del conjunto de prueba con el recomendador *J48*

Dataset	Rank	Classifier	Accuracy (%)
4	1	BayesNet	49.3976
	2	DecisionTable	48.1228
	3	OneR	46.9880
	4	NNge	45.7811
	5	Ridor	44.5783
	6	NaïveBayes	44.5783
	7	J48	40.9639
	8	JRip	38.5542
8	1	J48	85.0048
	2	NNge	84.3301
	3	JRip	84.2105
	4	Ridor	82.2967
	5	DecisionTable	81.6986
	6	NaïveBayes	80.5024
	7	BayesNet	79.3062
	8	OneR	78.3493

En la tabla 5.3 se muestra, para el caso de dos de las fuentes de datos clasificadas por el recomendador, un ranking que indica en orden de arriba a abajo de la tabla, cual era el mejor clasificador para cada fuente de datos y cual fue el peor.

Como se puede observar, en la fuente de datos número 4, aunque nuestro recomendador no sugirió el mejor clasificador, *BayesNet*, se sugirió el segundo mejor clasificador, *DecisionTable*. De hecho, la diferencia en cuanto a la exactitud o la precisión entre *BayesNet* y *DecisionTable* es muy pequeña, apenas de un 1 %, mientras que la diferencia entre el clasificador seleccionado *DecisionTable* y el cuarto mejor clasificador, *NNge*, ya es mayor de un 2 %. Por lo tanto, el recomendador todavía logra su objetivo, incluso no recomendando el mejor clasificador,

recomienda el segundo mejor, teniendo como precisión un valor muy cercano a la del primero.

Algo similar ocurre con la otra fuente de datos mal clasificada, la número 8. En este caso, el mejor algoritmo de clasificación es *J48*, pero nuestro recomendador sugirió *DecisionTable*. Aunque en este caso la diferencia, en términos de precisión, entre *J48* y *DecisionTable* es mayor que en el caso anterior (la fuente de datos número 4), también se puede observar que la precisión de los peores clasificadores están por debajo del 80 %, y el clasificador propuesto por nuestro recomendador, *DecisionTable*, tiene precisión cerca de 82 %. Por lo que podemos concluir, que aún sabiendo que existen clasificadores con mejores resultados, nuestro recomendador sugiere al usuario un clasificador con un rendimiento medio, muy alejado de los peores clasificadores.

Con estos resultados, podemos concluir que nuestro recomendador cumple de forma más que notable con la intención de recomendar al usuario inexperto en minería de datos que clasificador debería usar para obtener predicciones y modelos sobre sus datos, sin necesidad de consultar con un experto en minería de datos. En este punto, surgen dos preguntas: ¿Hasta qué punto la exactitud o precisión de los clasificadores sugeridos por nuestro recomendador difieren con respecto a la exactitud o la precisión que un experto en minería de datos puede alcanzar? y, ¿Dada la cantidad de tiempo y recursos invertidos por un experto en minería de datos para lograr el mejor modelo de clasificación posible, merece la pena ahorrárselos en pos de usar directamente el recomendador con el que tenemos la opción de obtener los resultados en pocos segundos y sin coste alguno?

Con el fin de responder a estas preguntas, se tomó como referencia el primer conjunto de datos que representa una de las instancias que usamos en el conjunto de prueba para validar el recomendador. Para esta fuente de datos, el recomendador sugirió el uso del algoritmo de clasificación *J48*, con una exactitud de 88.8889 %. Las tareas que el experto de minería de datos ha realizado para esta experimentación en aras de obtener el mejor modelo posible para esta fuente de datos en términos de precisión, comprenden el estudio de los datos (su comportamiento, estadística, etc.), la discretización de los datos (utilizando en cada prueba diversos algoritmos y probando a discretizar unos u otros atributos por separado), la localización de *outliers* que puedan ser susceptibles de ser eliminados del conjunto de datos, así como otros procesos de manipulación de datos. El mejor modelo obtenido por parte del experto obtuvo una precisión de 90.4763 %. La diferencia, por tanto,

en precisión del modelo que recomienda nuestro recomendador y el mejor modelo obtenido por el experto es de 1.5873 % en favor del modelo del experto. Podemos ver que la diferencia no es muy alta, sobre todo sabiendo que estamos hablando de modelos que están en torno al 90 % de precisión. Sin embargo, mientras el tiempo que tarda nuestro recomendador en mostrar el resultado al usuario es de apenas unos segundos, el experto ha necesitado una gran cantidad de tiempo para llegar a su mejor modelo.

Podemos concluir, viendo el alto coste que le supone al experto el llegar a obtener el mejor modelo y el tiempo que tarda en obtenerlo y la ínfima diferencia que existe entre este modelo y el recomendado por nuestro recomendador, que nuestra propuesta es totalmente válida para los objetivos propuestos, retornando resultados casi tan buenos como los que podría obtener un experto y pudiendo ser obtenidos en sólo unos segundos.

Como se muestra en los experimentos realizados, nuestra base de conocimiento puede ser un recurso útil para la aplicación de algoritmos de minería de datos por usuarios inexpertos. Los mejores clasificadores fueron recomendados 3 de cada 4 veces y el resto de las veces la recomendación nunca estuvo entre los peores resultados. Se hizo un análisis también teniendo en cuenta las tareas que debe hacer un experto y el tiempo y esfuerzo que estas conllevan a la hora de hacer un análisis de minería, en comparación con el breve tiempo que demora el procesamiento de nuestro recomendador.

Con el objetivo de validar nuestra propuesta en diferentes casos de estudio y tener diferentes criterios de comparación al observar el funcionamiento de nuestra propuesta, se aplicará a otros casos de estudio que se presentarán a continuación.

Capítulo 6

Aplicación de la propuesta de minería a otros casos de estudio

En este capítulo se muestran los resultados obtenidos por nuestra propuesta al analizar datos pertenecientes a otros dos casos de estudio. Se pretende comparar los resultados obtenidos por los expertos en cada caso, con los obtenidos por nuestra propuesta, con el fin de poder validarla.

6.1. Caso de estudio con datos urbanísticos

Los datos analizados fueron obtenidos durante un estudio relacionado con la incidencia de las externalidades en la formación espacial del valor del suelo. Esta investigación tuvo lugar en algunos municipios de la provincia de Alicante [145].

La idea general es que los usuarios, puedan obtener patrones que describan cuáles son las variables que más influencia tienen en la formación del valor del suelo. Teniendo en cuenta que estos usuarios no tienen profundos conocimientos de técnicas de minería de datos, lograr este objetivo por sí mismos, les era bastante complejo.

En este contexto resulta viable aplicar la propuesta desarrollada en esta tesis doctoral, y analizar sus resultados. La diferencia fundamental en la aplicación de los experimentos respecto a los realizados en el Capítulo 5 radica en que para el caso de estudio de e-learning se tenía un conocimiento suficiente de las fuentes de datos y la aplicación del recomendador se restringuió solamente a los 8 mejores

algoritmos según su desempeño, mientras que para este caso se tendrán en cuenta todos los posibles algoritmos de clasificación disponibles en Weka.

6.1.1. Necesidad del análisis de los datos por nuestra propuesta

La vivienda es un bien multiatributo que satisface varias necesidades simultáneamente y presenta múltiples cualidades que son valorables. El precio que el consumidor estará dispuesto a pagar por un inmueble dependerá de estas cualidades y de la valoración económica que haga de ellas.

Considerando una tipología edificatoria concreta, el valor de mercado de un determinado bien se convierte en un referente económico del grado de preferencia de una localización frente a otras, así como de su nivel de competitividad dentro de un ámbito determinado. Por tanto, la cuantificación del grado de preferencia de una localización frente a otra y la predisposición a pagar por ella, no depende de un único atributo, sino de múltiples factores de diferentes naturalezas como son la económica, social, urbana y medioambiental. Por ello, el análisis del comportamiento de un determinado mercado inmobiliario implica la evaluación de dichas variables en el mercado final.

El experto en el dominio, se encuentra ante la necesidad de utilizar herramientas capaces de dar respuestas ágiles en la gestión de múltiples datos de diferente naturaleza. Son precisas técnicas de minería de datos para definir un algoritmo que establezca una estructura de relación entre los valores de mercado y el conjunto de atributos, cuantificando su incidencia y contribución al valor global de activo analizado [145]. La formación del experto en el dominio, en campos vinculados pero no específicos de su desarrollo profesional, como es el área del análisis de datos, supone en muchas ocasiones un handicap que dificulta sus análisis. A partir de este antecedente, se hace necesaria la utilización de nuestra propuesta en este caso de estudio.

6.1.2. Descripción y preparación de las fuentes de datos utilizadas para la experimentación

En el estudio realizado se extrajeron un número considerablemente grande de variables, alrededor de 200, por lo que en este caso fue necesario un análisis previo de los expertos para realizar una selección adecuada de los datos que tomarían parte en nuestro análisis. Para obtener cuál de estas variables tenían más incidencia en la variable objetivo, aplicamos técnicas de regresión. La cantidad inicial de atributos se acortó hasta aproximadamente 25 para cada una de las fuentes de datos. Los especialistas necesitaban analizar las fuentes de datos acorde a la distribución familiar presente en cada inmueble. Debido a esto, se distribuyeron los datos en 3 grupos:

- Unifamiliar: inmuebles donde habita una sola familia.
- Plurifamiliar: inmuebles donde están establecidos más de una familia.
- General: fuente de datos donde se incluyen todos los inmuebles incluidos en el estudio.

Para este caso de estudio, se pretende estudiar el comportamiento de un indicador específico, el valor del suelo, teniendo en cuenta el resto de las variables que se seleccionaron en el estudio realizado. A partir de los requerimientos de los usuarios la metodología solamente se enfocará en las técnicas de clasificación. Se decidió para un correcto uso de las técnicas de clasificación aplicarle a los datos un proceso de discretización, para convertir atributos numéricos a nominales. Esta decisión fue tomada debido a que el atributo que los usuarios querían predecir era el valor del suelo, y este atributo tenía valor numérico. Utilizamos el algoritmo *Discretize* perteneciente a la librería de código abierto de Weka para este propósito. Para discretizar el principal parámetro de configuración es el número de *bins* o grupos de clases en que se quiere distribuir los datos del atributo a predecir. Con el objetivo de obtener una mejor distribución de los datos, se decidió obtener dos fuentes de datos por cada fuente de datos original analizada (unifamiliar, plurifamiliar y general). La diferencia consiste en el número de grupos de clases para el atributo a predecir, configurada finalmente en grupos de 4 y 10 clases respectivamente para cada una de las fuentes de datos a analizar. Para todos los casos el atributo a predecir es el valor del suelo. Finalmente, se obtuvieron 6 fuentes de datos correctamente preparadas (*4binsPluriFam25Atrib*, *10binsPlurifamiliar25Atrib*,

4BinsUnifamiliar25Atrib, *10binsUnifamiliar25Atrib*, *4binsGeneral26Atrib*, *10binsGeneral26Atrib*). Un ejemplo de un fichero de la fuente de datos Plurifamiliar con 4 clases es mostrada en el código 6.1.

```
@relation '4binsPlurifamiliar25Atrib'
@attribute EntradaAutovia numeric
@attribute Playa numeric
@attribute Golf numeric
@attribute C2INTERIOR1 numeric
@attribute C4COSTAESPECIAL numeric
@attribute C5COSTA2 numeric
@attribute SA\_OCUPACIONTERRITORIO numeric
@attribute SA\_USODELAVIVIENDA numeric
@attribute SA\_TIPOLOGIADELAVIVIENDA numeric
@attribute SA\_PAISAJE numeric
@attribute P\_ESTRUCTURADELAPOBLACION numeric
@attribute P\_NACIONALIDADPOBLACION numeric
@attribute P\_TIPODENUCLEOFAMILIAR numeric
@attribute P\_ESTUDIOS numeric
@attribute P\_TIPODEPUESTODEPERSONADEREFERENCIA numeric
@attribute P\_SECTORDEREFERENCIADELHOGAR numeric
@attribute E\_POBLACIONPRODUCTIVA numeric
@attribute E\_SOCIOECONOMIA numeric
@attribute E\_PARO numeric
@attribute E\_ACTIVIDAD numeric
@attribute I\_EQUIPAMIENTOS numeric
@attribute I\_SERVICIOS numeric
@attribute I\_PROBLEMATICASURBANAS numeric
@attribute TURISMO numeric
@attribute ValorSuelo \{'\'(-inf-173.25]\'','\'(173.25-589.5]\'','\'(589.5-005.75]\'',
'\'(1005.75-inf)\''\}
```

Código 6.1: Segmento del fichero *.arff* creado

Con las fuentes de datos obtenidas un estudio fue realizado por expertos en minería de datos, dónde un grupo de algoritmos de clasificación fueron aplicados con el objetivo de obtener modelos de minería y determinar cuál fue el mejor. Esta experimentación fue realizada por expertos de forma manual. Debido a ellos gastaron un tiempo considerable para obtener los resultados.

La contribución de nuestra propuesta es la obtención de un mecanismo automático que permite a usuarios inexpertos aplicar el mejor algoritmo de minería de datos después de sea analizado la calidad de sus datos de entrada. Nuestra propuesta tiene como incentivo principal la posibilidad de substituir la presencia de un experto cuándo un usuario inexperto quiere analizar sus datos en tiempo real para obtener conocimiento fiable. En la siguiente sección se procederá a aplicar la metodología propuesta para el caso de estudio presentado. El objetivo es demostrar que un usuario inexperto, arquitectos en este caso, obtiene conocimiento con relativa buena calidad al compararse con los resultados obtenidos por los expertos de minería de datos.

6.1.3. Resultados obtenidos

En esta sección, se explica el proceso seguido para llevar a cabo la experimentación. Luego, se mostrará y discutirá los resultados obtenidos. Para evaluar los resultados obtenidos por el recomendador se compararán contra los obtenidos por los expertos en minería de datos, para ello se tienen los resultados obtenidos por los expertos de minería de datos después de ser aplicados los algoritmos de minería de datos. Estos algoritmos fueron ordenados por el porcentaje obtenido de instancias correctamente clasificadas. Para cada fuente de datos, los expertos seleccionaron el mejor algoritmo a aplicar.

Por otro lado, se tomaron los resultados obtenidos por el recomendador para cada una de las fuentes de datos de prueba. En la tabla 6.1 se expone el ranking de los resultados obtenidos cuando los expertos aplicaron los algoritmos de minería de datos. Los datos que son mostrados son los obtenidos para la fuente de datos *10binsPlurifamiliar*. Los resultados fueron ordenados de mayor a menor exactitud. El resultado sugerido por el recomendador es también mostrado en la tabla, señalado con asteriscos. En la Fig. 6.1 se muestra para cada uno de las 6 fuentes de datos analizadas los resultados obtenidos luego de aplicar un conjunto de criterios estadísticos. Estos valores son representados en una gráfica *Boxplot* con vistas a realizar un mejor análisis (Fig. 6.2) [146], [147]. Cada caja representa una de las fuentes de datos analizadas. En cada una de ellas han sido representado los rangos intercuartiles (valores que contienen el 25 % y 75 % de los casos). Las marcas y la altura de la caja nos informan de la variabilidad de los resultados obtenidos para cada fuente de datos. Cuando se observa los resultados obtenidos en una caja, se puede hacer una comparación con el resto de las fuentes de datos, para saber si hay homogeneidad de varianzas o por el contrario son muy heterogéneos.

Labels	4binsPluriFam25Atrib	10binsPlurifamiliar25Atrib	4BinsUnifamiliar25Atrib	10binsUnifamiliar25Atrib	4binsGeneral26Atrib	10binsGeneral26Atrib
Min	49.1646	41.0633	73.8386	90.4645	95.5965	81.6989
Q_1	75.0886	49.924	99.022	94.33575	99.7189	85.6964
Median	82.3291	69.5696	99.4295	94.458	99.7189	88.351
Q_3	85.6709	73.7721	99.5925	94.98775	99.8438	90.34975
Max	88.557	76.7089	99.674	95.925	99.8751	91.3179
IQR	10.5823	23.8481	0.5705	0.652	0.1249	4.65335
For the Box (IQR and Median)						
Q2-Q1	7.2405	19.6456	0.4075	0.12225	0	2.6546
Q3-Q2	3.3418	4.2025	0.163	0.52975	0.1249	1.99875
For the Whiskers						
Q_3+1.5*IQR	101.54435	109.54425	100.44825	95.96575	100.03115	97.329775
Q_1-1.5*IQR	59.21515	14.15185	98.16625	93.35775	99.53155	78.716375
Upper Whisker	88.557	76.7089	99.674	95.925	99.8751	91.3179
Lower Whisker	59.21515	41.0633	98.16625	93.35775	99.53155	81.6989
W_{upper}-Q_3	2.8861	2.9368	0.0815	0.93725	0.0313	0.96815
Q_1-W_{lower}	15.87345	8.8607	0.85575	0.978	0.18735	3.9975
For the Outliers						
Manual choice	86.9367	76.7089	99.674	95.925	99.8438	91.3179
Recommender choice	82.3291	72.0506	99.674	94.3765	99.8438	89.9438

FIGURA 6.1: Análisis estadístico de los resultados obtenidos.

TABLA 6.1: Resultados obtenidos después de ser aplicados los algoritmos de clasificación

Fuente de datos	Lugar	Clasificador	Accuracy (%)
10binsPlurifamiliar	1	JRip	76.7089
	2	ClassificationViaRegression	75.7468
	3	OneR	75.1392
	4	END	74.7342
	5	J48	73.8734
	6	RotationForest	73.7721
	7	Ridor	73.6709
	8	IBk	73.1139
	9	NNge	72.0506****
	10	DecisionTable	71.5443
	11	PART	69.5696
	12	KStar	68.1519
	13	LogitBoost	67.7975
	14	BayesNet	58.2785
	15	LWL	57.3671
	16	RacedIncrementalLogitBoost	49.924
	17	ConjunctiveRule	48.5063
	18	MultiBoostAB	45.3165
	19	AdaBoostM1	45.3165
	20	NaïveBayes	45.3165
	21	ZeroR	41.0633

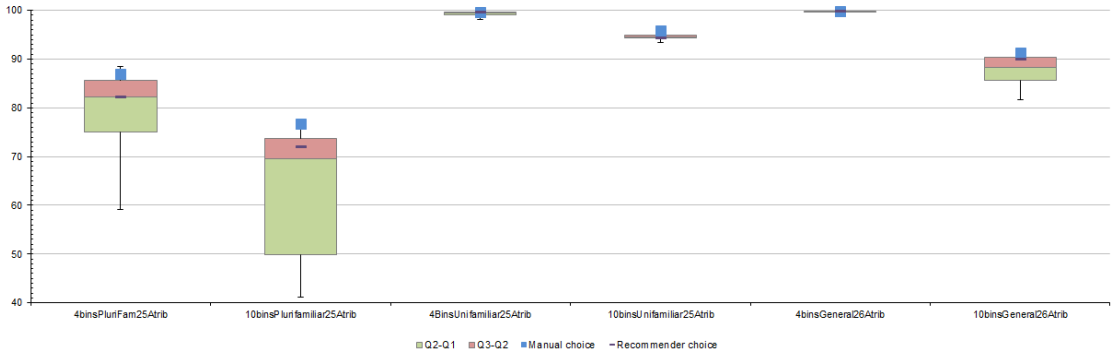


FIGURA 6.2: Representación visual de los resultados obtenidos.

Como se puede apreciar en la Fig. 6.2 están representados las seis fuentes de datos que formaron el conjunto de prueba evaluado por el recomendador. Para cada uno de las fuentes de datos es mostrado el rango de valores que obtuvo los algoritmos aplicados por los expertos en minería de datos. Indicamos el algoritmo decidido por el experto con un pequeño cuadrado en la parte superior de cada rango, como el mejor, y con la marca de un guión la posición obtenida dentro de los posibles algoritmos la opción sugerida por el recomendador.

La idea fue mostrar los resultados obtenidos por el recomendador y los obtenidos por los expertos en un gráfico *Boxplot*, para analizar en qué medida es bueno el resultado obtenido por el recomendador, teniendo en cuenta los valores que obtuvieron los expertos cuándo los algoritmos fueron aplicados.

Como se puede apreciar en la Fig. 6.2 todos los valores obtenidos por el recomendador están ubicados en el rango del primer cuartil de distribución de los datos. De esta manera se hace realidad la premisa inicial que el resultado del recomendador era considerablemente bueno si el valor obtenido se localizaba en el primer cuartil.

En los casos 1, 2 y 6, por su orden en la figura, se puede apreciar mejor debido a la variabilidad de los resultados de los algoritmos obtenidos aplicados por los expertos en minería. La fuente de datos *10binsPlurifamiliar25Atrib* tiene una mayor varianza en relación con el resto de las fuentes de datos. Sus resultados están muy dispersos en todo el rango, teniendo valores entre 41 % y 77 %. Sin embargo, en el resto de las fuentes de datos los resultados son más compactos, debido a que es menor la variabilidad. En el caso de la primera fuente de datos *4binsunifamiliar25Atrib*, fué el que obtuvo el resultado más discreto. La diferencia con la mejor exactitud fue 4.6076, un valor relativamente bueno, si se tiene en cuenta que los expertos ejecutaron más de veinte algoritmos. En los casos 3, 4 y 5 los valores de los algoritmos son muy similares y no divergen mucho. El recomendador dió una respuesta aceptada teniendo en cuenta el rango permisible. Esto es debido a la baja variabilidad en los resultados obtenidos por los expertos luego de aplicar los algoritmos.

Al observar la gráfica se pudiera pensar que en los casos 3,4 y 5 no era necesario utilizar el recomendador, ya que cualquier algoritmo daría un buen resultado. Se debe puntualizar que ese razonamiento se puede hacer después que los expertos obtuvieron el comportamiento de todos los algoritmos, en un caso de la realidad, normalmente no se tiene el resultado del experto, por lo que no se sabe como debe

funcionar cada algoritmo sobre una fuente de datos, por lo que se demuestra la validez de nuestra propuesta.

Del análisis comparado de los modelos para uso residencial de carácter unifamiliar y plurifamiliar podemos establecer los puntos de divergencia y encuentro (casos 1,2 y 3,4), justificados por las diferentes preferencias de la potencial demanda. La cuantificación de la influencia de cada variable permite establecer la distribución diferencial por ámbitos de caracterización de las variables incorporadas a los modelos.

En ambos casos son las variables vinculadas al sistema de asentamientos las que presentan una mayor influencia en la formación del valor, seguido de los rasgos poblacionales y económicos. Sin embargo se tratan de mercados diferenciados que matizan el perfil de demanda y, por tanto, presentan diferencias en los algoritmos planteados.

En el caso del plurifamiliar existen amplias diferencias en la caracterización de la muestra, con una parte de marcado carácter de primera residencia junto a los núcleos urbanos del interior, y uno de segunda residencia en el entorno suburbano y los núcleos urbanos de costa. Por ello, podemos hablar de que existen diferentes submercados o patrones, con una mayor dificultad de adaptación a un único algoritmo de predicción.

En este caso, existe una fuerte prevalencia en la incorporación de variables vinculadas al sistema de asentamiento en detrimento de las de carácter económico. Existe una mayor especialización del sector terciario vinculado al turismo, apreciándose de modo muy significativo los atributos ambientales que se posicionan de reclamo para el desarrollo de actividades de ocio y recreo, como son la cercanía a la línea de costa y calidad del entorno ambiental.

El caso del unifamiliar se trata de un mercado más compensado lo que conlleva a un reparto equilibrado entre los diferentes ámbitos de caracterización de las variables, reflejándose en una mayor capacidad de explicación de los modelos y adaptabilidad a diferentes algoritmos sin pérdida en la capacidad de explicación.

Teniendo este comportamiento en cuenta, se puede afirmar que el resultado del recomendador para estos casos no fue categórico, pero está en el rango permisible de los mejores algoritmos para ser usados.

6.2. Casos de estudio con datos de UCI

A los efectos de evaluar el desempeño y la eficacia de nuestro método de recomendación de algoritmo para técnicas de clasificación, verificar si el método es potencialmente útil en la práctica, y permitir a otros investigadores confirmar nuestros resultados, hemos utilizado varias fuentes de datos del repositorio de datos UCI [148]. Múltiples estudios por la comunidad de aprendizaje automático para el análisis empírico de los algoritmos de aprendizaje automático han sido realizados. Estas fuentes de datos han sido citadas en más de 1000 ocasiones (ver sección 2).

Nuestra propuesta ha sido aplicada a 64 de estas fuentes de datos. La tabla 6.2 presenta el identificador de cada fuente de datos, su nombre, el número de instancias que tiene, el número de atributos que permite describir a cada instancia (sin incluir el atributo objetivo), y el número de clases para cada fuente de datos.

Id	Nombre	Instancias	Atributos	NoClases
1	anneal	898	39	6
2	anneal.ORIG	898	39	6
3	arrhythmia	452	280	16
4	audiology	226	70	24
5	autos	205	26	7
6	balance-scale	625	5	3
7	breast-cancer	286	10	2
8	breast-w	699	10	2
9	bridges_version1	107	13	6
10	bridges_version2	107	13	6
11	car	1728	7	4
12	cmc	1473	10	3
13	colic	368	23	2
14	colic.ORIG	368	28	2
15	credit-a	690	16	2
16	credit-g	1000	21	2
17	cylinder-bands	540	40	2
18	dermatology	366	35	6
19	diabetes	768	9	2

Sigue en la página siguiente.

Id	Nombre	Instancias	Atributos	NoClases
20	ecoli	336	8	8
21	flags	194	30	8
22	glass	214	10	7
23	haberman	306	4	2
24	hayes-roth_test	28	5	4
25	hayes-roth_train	132	5	4
26	heart-c	303	14	5
27	heart-h	294	14	5
28	heart-statlog	270	14	2
29	hepatitis	155	20	2
30	hypothyroid	3772	30	4
31	ionosphere	351	35	2
32	iris	150	5	3
33	kdd_synthetic_control	600	62	6
34	kr-vs-kp	3196	37	2
35	labor	57	17	2
36	liver-disorders	345	7	2
37	lung-cancer	32	57	2
38	lymph	148	19	4
39	mfeat-factors	2000	217	10
40	mfeat-morphological	2000	7	10
41	mfeat-pixel	2000	241	10
42	molecular-biology_promoters	106	59	4
43	page-blocks	5473	11	5
44	pendigits	10992	17	10
45	postoperative-patient-data	90	9	3
46	primary-tumor	339	18	22
47	segment	2310	20	7
48	shuttle-landing-control	15	7	2
49	sick	3772	30	2
50	solar-flare_1	323	13	2
51	solar-flare_2	1066	13	3

Sigue en la página siguiente.

Id	Nombre	Instancias	Atributos	NoClases
52	sonar	208	61	2
53	soybean	683	36	19
54	spambase	4601	58	2
55	spect_test	187	23	2
56	spect_train	80	23	2
57	sponge	76	46	3
58	tae	151	6	3
59	tic-tac-toe	958	10	2
60	trains	10	33	2
61	vehicle	846	19	4
62	vote	435	17	2
63	vowel	990	14	11
64	zoo	101	18	7

Tabla 6.2: Descripción de las 64 fuentes de datos

A continuación se describirá como se diseñaron los experimentos realizados. Esta experimentación se dividió en tres etapas. Para este caso de estudio, a diferencia de los anteriores, como no contábamos con la información que asegurara que algoritmo tenía mejor desempeño, para cada fuente de datos, modificamos necesariamente los pasos para poder aplicar nuestra metodología, y comparar los resultados obtenidos por nuestro recomendador.

- En la primera etapa se introdujeron las fuentes de datos al flujo de trabajo diseñado para construir la base de conocimiento. Es decir, se aplicaron todos los algoritmos de clasificación a las 64 fuentes de datos. De esta manera se simuló el trabajo que haría un experto para determinar el algoritmo que mejor rendimiento consiguió para cada una de las fuentes de datos.

- En la segunda fase, se ejecutó el flujo de trabajo para usuarios inexpertos, en aras de ejecutar el recomendador diseñado, y obtener el algoritmo sugerido por nuestro recomendador. Al ser analizadas las fuentes de datos por el flujo de trabajo para los inexpertos se obtuvo la información de la calidad de cada fuente de datos. Debemos aclarar que la base de conocimiento utilizada para la construcción del recomendador en este caso no tuvo en cuenta los modelos obtenidos en la etapa 1.

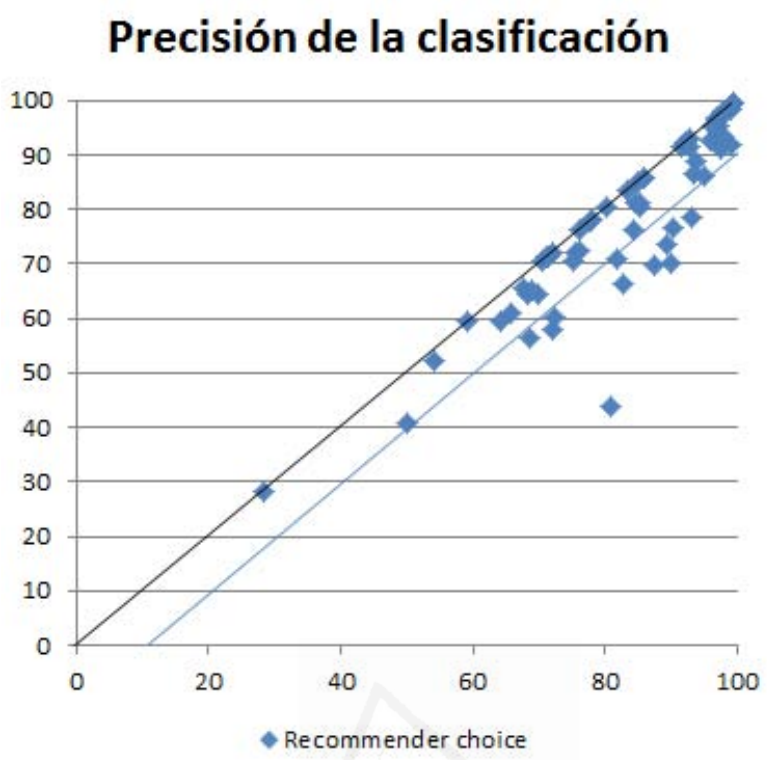


FIGURA 6.3: Precisión en la clasificación del mejor algoritmo real contra la precisión de la clasificación de los algoritmos obtenidos por el recomendador.

- Finalmente se realizó un análisis comparativo a partir de dos gráficas para comparar los resultados obtenidos por el mejor algoritmo en cada caso, y el algoritmo que sugirió el recomendador construido

En aras de proporcionar una imagen intuitiva del rendimiento de nuestro método de recomendación propuesto, se empleó un gráfico de dispersión 6.3. El objetivo es mostrar la precisión de los algoritmos recomendados frente a los resultados del mejor algoritmo en cada una de las 64 fuentes de datos analizadas. Una recta paralela a la diagonal fue dibujada sobre la gráfica para representar el rango que define el 90 % de confianza. Analizando la gráfica de esa forma podemos observar como una gran mayoría de los resultados caen en el área entre la diagonal y la línea paralela trazada, estando dentro del rango señalado más del 82 % de las fuentes de datos. Debemos aclarar que la mayoría de los expertos de minería de datos y estadística refieren que un buen es aquel que tiene un 80 % de precisión, por lo que si se trazara otra recta paralela a la diagonal que señalara ese rango, sólo una instancia caería fuera del rango señalado.

FIGURA 6.4: Resultados de los algoritmos obtenidos por el recomendador contra los mejores algoritmos.

Estos resultados son una clara evidencia de que el recomendador ha tenido un buen rendimiento. Además se debe tener en cuenta la gran variedad de algoritmos de clasificación y la heterogeneidad de las fuentes de datos analizadas. A partir de la Fig. 6.4 se puede observar que la mayoría de los puntos se encuentran en, o muy cerca de la diagonal. Esto indica que la mayoría de los algoritmos recomendados son tan buenos o, casi tan buenos como el mejor algoritmo.

Podemos concluir que se ha demostrado la factibilidad de usar nuestra propuesta como recurso para guiar a usuarios inexpertos que desean analizar sus datos. Por otra parte, se ha podido comparar el rendimiento que tienen los algoritmos recomendados contra el que tiene el mejor algoritmo en cada caso, estando siempre en el primer cuartil, demostrando el éxito en este sentido.

Al tener el proceso realizado por expertos para la obtención de los mejores modelos de minería un coste tan alto, en términos de tiempo y esfuerzo, comparado con el bajo coste de aplicar nuestro recomendador, podemos asegurar que nuestra propuesta no es sólo válida, sino que el recomendador sugiere algoritmos de clasificación que logran modelos de clasificación tan buenos como aquellos obtenidos por un experto, y en tan sólo algunos segundos.

Capítulo 7

Trabajos futuros

En este capítulo se exponen varias líneas de trabajo en las que ya se ha comenzado a ingadar y trabajar, fundamentalmente con el objetivo final de lograr la total automatización de las tareas relacionadas con la utilización de técnicas de minería de datos, y estas puedan ser utilizadas fácilmente por usuarios de todos los sectores de la sociedad. Específicamente se presentarán dos líneas de trabajo:

- Hacia el descubrimiento y reusabilidad del conocimiento obtenido a partir de explotar fuentes de datos abiertas. En esta línea de trabajo se pretende extender nuestra propuesta de aplicación de técnicas de minería, enfocada al uso sobre datos abiertos. Agregando la posibilidad de compartir el conocimiento adquirido durante el análisis de los datos para su posterior reutilización.
- Taxonomía de requisitos para la realización de técnicas de minería por usuarios inexpertos. Teniendo en cuenta que un usuario inexperto no tiene idea de que es un requisito de minería, se propone un mecanismo para identificarlos de la manera más fácil posible.

7.1. Reutilización del conocimiento obtenido a partir de explotar fuentes de datos abiertas

Actualmente los gobiernos de todo el mundo generan datos abiertos en aras de aumentar la transparencia hacia la sociedad. Además, la filosofía de datos abiertos alienta el valor de reusar los datos mediante la participación y la colaboración

entre los ciudadanos, las instituciones públicas y las organizaciones privadas. La promesa de los datos abiertos está promocionando la reutilización de mecanismos para descubrir nuevos conocimientos y mejorar la vida cotidiana de los ciudadanos a través del:

- desarrollo de aplicaciones (en la web, aplicaciones móviles, etc.) que permitan reusar y añadir valor a los datos abiertos existentes.
- el análisis de datos para obtener nuevas compenetraciones y se adquiera conocimiento que soporte el proceso de toma de decisiones en la vida cotidiana.

Sin embargo, el nuevo conocimiento adquirido a partir de los datos abiertos no se incorpora a las fuentes de datos abiertas nuevamente y de esta forma se obstaculiza su aprovechamiento. Con vistas a solucionar esta situación, los metadatos y cómo este conocimiento fue hallado en las fuentes de datos deben ser incorporados como nuevas fuentes de datos. La extracción de los metadatos pueden ser simples cuando se realiza un análisis sencillo de los datos, pero algunos análisis avanzados como los que se requieren al utilizar técnicas de minería de datos demandan técnicas específicas para ocuparse de los metadatos. Específicamente, se detectan dos problemas cuando son necesarias técnicas de minería de datos:

- los ciudadanos comunes no saben como extraer conocimiento de los datos abiertos disponibles.
- una vez que alguien ha hecho un análisis para descubrir algún conocimiento de los datos, este no puede ser reusado y reincorporado a la estructura *Linked Open Data (LOD)* como nuevo conocimiento para ser posteriormente reutilizado.

El primer punto ha sido tratado en [149], y se considera como el Big Data Divide (capítulo 1) la gran disponibilidad de datos pero la imposibilidad de poder analizarlos por parte de ciudadanos comunes. También se ha abordado en el capítulo 4. A continuación nos centraremos en el segundo punto. Nuestra hipótesis es la siguiente: los datos abiertos están aumentando cada vez más su disponibilidad, haciendo posible su reutilización mediante la aplicación de esquemas *Resource Description Framework (RDF)*) y el uso de *Linked Open Data (LOD)*, por lo que ficheros RDF pueden ser también utilizados para maximizar la reutilización del

conocimiento descubierto a partir de los resultados obtenidos al aplicar técnicas de minería de datos, siempre que el conocimiento sea incluido de nuevo en las fuentes de datos LOD como un nuevo recurso.

Cuando examinamos fuentes de datos abiertas en la Web, encontramos varios formatos heterogéneos, tales como CSV, JSON o RDF. Últimamente, RDF se considera como el formato más adecuado para la reutilización de datos a través del concepto LOD. Por lo tanto, cuando los datos son analizados y procesados, los metadatos y sus resultados deben ser también convertidos en RDF para continuar con la filosofía LOD y potenciar su reutilización. Para lograr tal objetivo, se propone utilizar el desarrollo dirigido por modelos, con el fin de obtener modelos homogéneos de cualquier tipo de formato de fuente de datos.

Nuestro aporte está basado en la definición de un proceso de descubrimiento de conocimiento sobre datos abiertos que permita que los metadatos y los resultados obtenidos sean etiquetados en ficheros RDF e incorporados como LOD. Debido a la naturaleza iterativa de este proceso lo hemos nombrado como *Knowledge Spring Process*, ó Proceso Resorte de Conocimiento (KSP). A continuación, ejemplificamos un posible escenario en el que nuestro enfoque podría ser útil: el dominio del periodismo de datos.

Un periodista desea realizar un análisis de datos mediante la aplicación de técnicas de minería para descubrir algunos patrones en los gastos realizados por los candidatos de varios partidos políticos en algunas campañas electorales. El conocimiento proporcionado por este estudio es útil, ya que pudiera ser accedido y reutilizado por otros actores, en combinación con otros datos abiertos existentes, por ejemplo, organizaciones de vigilancia electorales.

7.1.1. Habilitando a usuarios inexpertos para aplicar técnicas de minería de datos

En nuestros días el incremento del uso de las Tecnologías de la Información y las Comunicaciones (TICs) nos permiten acceder rápidamente a gran cantidad de información y apoyar nuestro proceso de toma de decisiones diarias. Por ejemplo, alguien que use un motor de búsqueda Web siempre espera la mejor respuesta en el menor tiempo con el fin de tomar algunas decisiones relacionadas con posibilidades

de viajes, compras, o cualquier otra tarea de la vida diaria. Sin embargo, para realizar análisis avanzados para obtener conocimiento de los datos disponibles (por ejemplo, descubrir patrones) requiere que un experto tome parte en el proceso. Como hemos mencionado anteriormente el proceso de descubrimiento de conocimiento ha sido vinculado con la presencia de expertos, pero desafortunadamente cambios radicales en este aspecto son necesarios, con el fin de crear mecanismos que permitan a usuarios inexpertos consumir la información disponible y descubrir conocimiento útil en ella. Con vistas a apoyar el concepto de *User Friendly Data Mining*, presentamos a continuación un mecanismo para permitir la extracción de conocimiento dentro de un escenario de datos abiertos. Con este objetivo varios desafíos deben ser abordados (ver Fig. 7.1):

1. Superar las dificultades que presentan los usuarios inexpertos con la diversidad de formatos de datos abiertos existentes.

2. Proponer un mecanismo que permita a los usuarios inexpertos identificar sus requerimientos al aplicar técnicas de minería para descubrir conocimiento.

3. Facilitar el descubrimiento de conocimiento a partir de fuentes de datos abiertos utilizando técnicas de minería de datos sin la presencia de un experto.

4. Crear un mecanismo que permita reusar el conocimiento previamente descubierto por usuarios inexpertos.

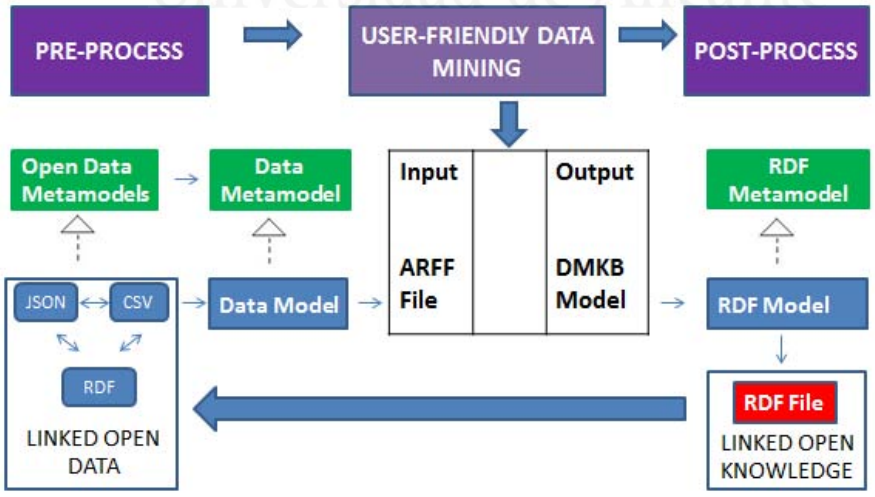


FIGURA 7.1: Propuesta para reusar el conocimiento.

El punto 2 será abordado en la sección 7.2, mientras que el punto 3 fue tratado en el capítulo 4. Específicamente aquí detallaremos los puntos 1 y 4.

7.1.2. Formatos de datos abiertos

Nuestro proceso KSP comienza cuando un usuario desea analizar alguna fuente de datos abiertas con el objetivo principal de descubrir conocimiento con vistas a tomar una decisión bien fundamentada. Los fuentes de datos abiertas están disponibles en muchos formatos: RDF, JSON o CSV. Cada uno de esos formatos presentan una estructura diferente para almacenar la información. La información se muestra de acuerdo a cada estructura. Para un usuario no experto se hace sumamente difícil entender cada una de ellas, y mucho menos tratar de encontrar patrones que le faciliten la toma de decisiones. A partir de esta problemática, luego de realizar un análisis, se propuso un método que permita integrar cada uno de los formatos que usualmente encontramos en los sitios de datos abiertos con el objetivo de contar con una estructura única para facilitar su explotación. Afortunadamente, también existen elementos comunes dentro de la diversidad de formatos de datos abiertos, por lo que se propuso un modelo homogéneo para representar datos de forma independiente de su formato original. Con este objetivo, se utilizó el enfoque conducido por modelos para representar la información extraída de las fuentes de datos abiertas en una forma estándar. El metamodelo de datos diseñado es presentado en la Fig. 7.2.

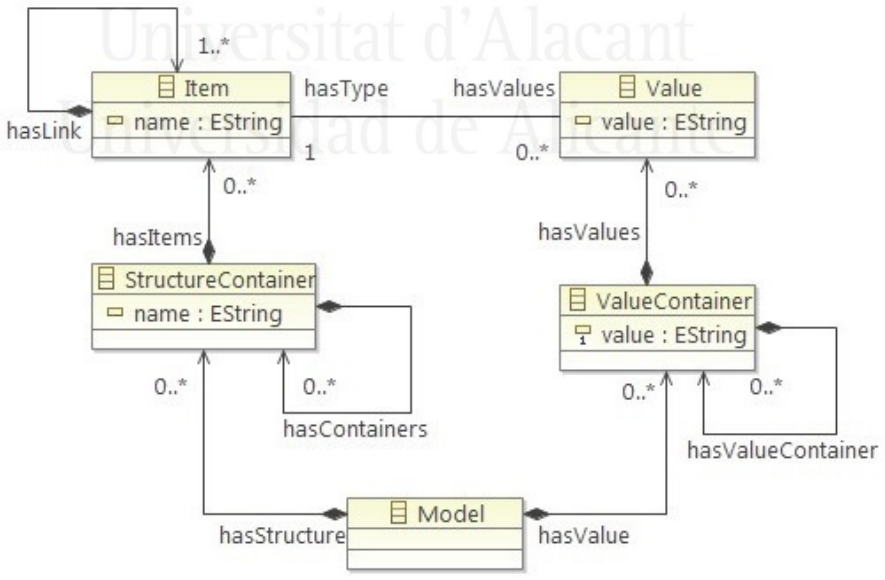


FIGURA 7.2: Metamodelo para representar la información de las diferentes fuentes de datos abiertas.

7.1.2.1. Descripción del metamodelo de datos

Después de examinar la forma en que los datos abiertos son estructurados, se ha definido un metamodelo con el fin de representar los conceptos de datos y sus relaciones de manera general. El objetivo es que este metamodelo pueda ser suficientemente simple para poder definir cualquier representación de los datos por el uso de modelos conformes a este metamodelo sin perder la semántica del esquema. En términos generales, cualquier fuente de datos puede convertirse en este tipo de modelo. A continuación, se explicará en detalle los conceptos incluidos en el metamodelo:

- Model: Esta es la clase contenedora principal del metamodelo, esta asociada con las clases `StructuralContainer` y `ValueContainer`. Estos elementos pueden ser accedidos por los atributos `hasStructural` y `hasValue`, respectivamente.
- StructuralContainer: Almacena los tipos de datos `hasItems`, que son elementos de tipo `Items`. Puede contener otros elementos `StructuralContainer`. El atributo `name` es la llave del contenedor.
- ValueContainer: Esta clase contiene elementos del tipo `Value`, a través de la relación `hasValues`. Pueden contener otros elementos `ValueContainer`. El atributo `value` es la llave del contenedor.
- Item: Almacena los tipos de datos en el atributo `name`. Un elemento `Item` puede contener algunos valores a través de la relación `hasValues`. Un `Item` puede tener vínculos con otros elementos de tipo `Item`, a través de la propiedad `hasLink`.
- Value: Contiene los valores en el atributo `value`. Estos valores pertenecen al tipo `Item` y ellos pueden ser accedidos a través de la propiedad `hastype`.

7.1.2.2. Obteniendo el modelo de datos

En este epígrafe se detallará cómo una fuente de datos abierta de entrada es transformada hacia el modelo de datos de acuerdo al metamodelo previamente presentado. Tomando en cuenta los posibles esquemas de los archivos que aparecen en los portales abiertos (RDF, JSON o CSV), se han implementado sus correspondientes

gramáticas. Las gramáticas fueron implementadas utilizando proyectos XText[1] de la plataforma Eclipse[2]. Una vez que las gramáticas son definidas se pueden obtener sus correspondientes metamodelos. A continuación se hará énfasis en ejemplos de ficheros RDF. El metamodelo de un fichero RDF es obtenido a partir de su correspondiente gramática, y es mostrado en la Fig. 7.3. Cuando un archivo se introduce

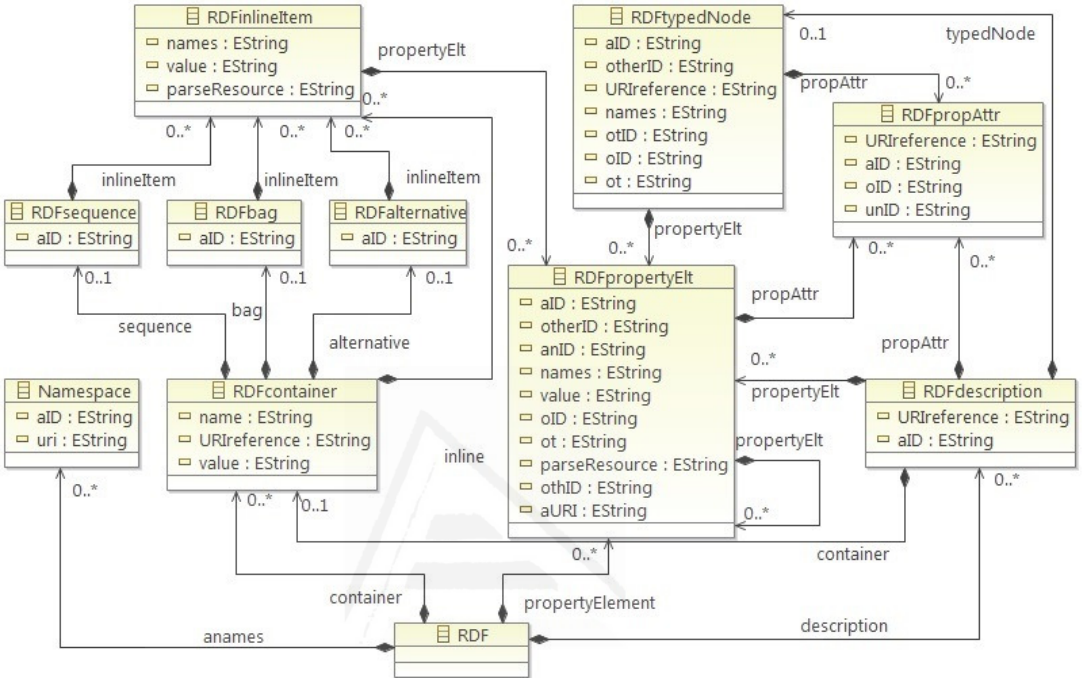


FIGURA 7.3: Metamodelo RDF.

en el sistema, este es validado por su correspondiente gramática en aras de garantizar una estructura correcta. Para cada uno de los posibles esquemas de archivos de entrada, transformaciones ATL (*Atlas Transformation Language*) [150] fueron implementadas para obtener el correspondiente modelo de acuerdo con el metamodelo *Datamodel* diseñado. A través de este proceso la uniformidad de los datos en el proceso de minería está garantizada. La definición formal de las transformaciones se lleva a cabo a través de reglas ATL. El segmento de código que a continuación se muestra permite transformar el elemento *rdfSequence* de un modelo RDF de entrada en el elemento *Item* del modelo destino 7.1.

```
1  rule rdfseq2item{
2   from rsq: RDF!RDFsequence(not(rsq.aID=OclUndefined))
3   to di:DM!Item(name <- rsq.aID,
4                 hasValues <- rsq.GetSeqValue())
5  }
```

CÓDIGO 7.1: Regla para convertir un elemento rdfseq a item

[1]http://www.eclipse.org/Xtext/
[2]http://www.eclipse.org

Después de obtener el modelo de datos, el proceso continúa la ejecución de la metodología de minería de datos amigable con el fin de descubrir conocimiento fiable [149]. En la próxima sección se discutirá qué tratamiento se le dará a la información que se genera después de aplicar el mencionado proceso.

7.1.3. Obteniendo conocimiento abierto

La principal ventaja de nuestra propuesta consiste en la potencialidad de la reutilización de los conocimientos adquiridos en el proceso de minería ejecutado. La contribución de nuestra propuesta está dada por dos resultados fundamentales al usuario. El primero es la solución al problema inicial presentado por el usuario inexperto, después de haber sido aplicado el modelo de minería correspondiente a los datos de entrada según la sugerencia del sistema recomendador y los requisitos del usuario. El otro, es la generación de un archivo RDF semánticamente etiquetado con todo el conocimiento obtenido en el proceso de minería de datos. Esta salida es muy innovadora porque de esta manera se comparte información útil y puede ser utilizada por otros usuarios. Ver Fig. 7.4. Los usuarios pueden utilizar los

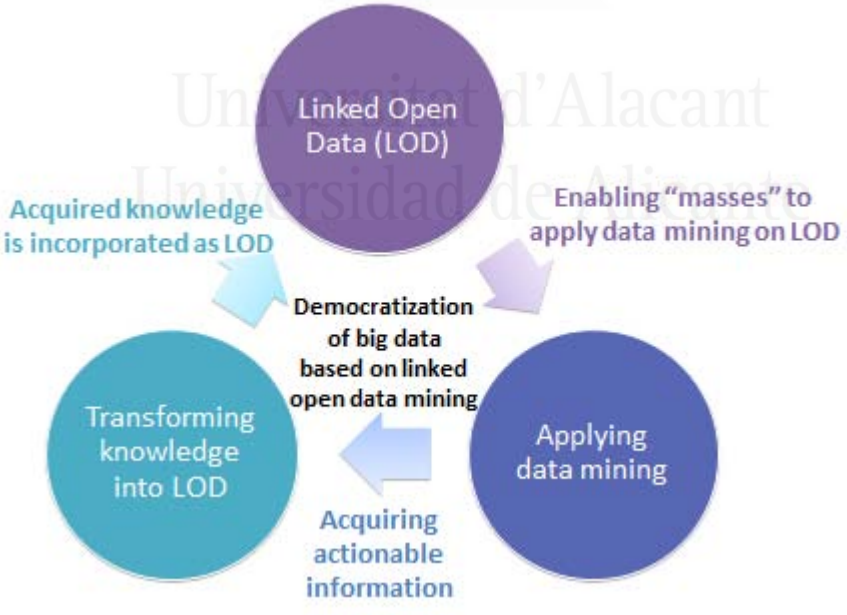


FIGURA 7.4: Generación de Conocimiento Etiquetado.

archivos LOD publicados en Internet. Aunque al estar semánticamente anotados éstos pueden estar asociados con otros archivos LOD. Cuando éstos son analizados por nuestra propuesta se convertirán en archivos *Linked Open Knowledge (LOK)*. Los archivos LOK también pueden estar asociados a otros archivos LOD o LOK.

7.1.3.1. Descripción del modelo RDF

Toda la información que se genera en el proceso de minería de datos aplicado [149] es almacenada en un modelo conforme al metamodelo presentado en la Fig. 3.5. La pregunta a resolver es: ¿Cómo podemos obtener un archivo RDF semánticamente anotado con la información incluida en el modelo de minería retornado? Para abordar este problema se presenta un metamodelo acordes a archivos RDF, con el fin de aplicar transformaciones ATL entre el metamodelo DMKB y la propuesta de metamodelo RDF. El modelo de datos RDF ha sido obtenido a partir de la gramática correspondiente[3] [4]. De esta manera, se define un modelo simple para describir las relaciones entre los recursos en términos de sus propiedades designadas y sus valores. El modelo básico de datos RDF definido consiste en 3 tipos de objetos:

- Recursos: Todo lo descrito por expresiones RDF son llamados recursos. Los recursos son siempre denotados por URIs más identificadores opcionales.
- Propiedades: Una propiedad es un aspecto específico, característica, atributo o relación utilizada para describir un recurso.
- Sentencias: Un recurso específico junto con una propiedad con nombre más el valor de esa propiedad para ese recurso es una declaración RDF.

Estas tres partes individuales de una sentencia son conocidas en la literatura como sujeto, predicado y objeto, respectivamente. El objeto de una oración (por ejemplo, el valor de la propiedad) puede ser otro recurso o podría ser un literal; es decir, un recurso (especificado por un URI) o una cadena sencilla o de otros tipos de datos primitivos definidos por XML. La principal ventaja de utilizar un metamodelo que representa archivos RDF es generar el resultado que se encuentra en un modelo DMKB en un fichero RDF para incluirlo como LOK.

7.1.3.2. Mapeo del modelo DMKB a RDF

La implementación del prototipo actual se basa en transformaciones modelo a modelo. Las transformaciones son el núcleo del desarrollo dirigido por modelos.

[3] Resource Description Framework (RDF) Model and Syntax, W3C Recommendation, 2.2. 1999

[4] http://www.w3.org/TR/1999/REC-rdf-syntax-19990222

En este epígrafe se describe el mapeo entre un modelo DMKB de entrada a un modelo RDF. La tranformación fue especificada usando el pluggin ATL de la plataforma Eclipse. En la Fig. 7.5 una definición informal es presentada. Cada elemento *DMKBModel* del metamodelo es mapeado en elementos RDF del metamodelo destino. En cada asignación entre clases, las propiedades de la clase origen se asignan a propiedades equivalentes del metamodelo destino. La transformación es automáticamente generada a partir de una transformación de orden superior de acuerdo con los enlaces semánticos establecidos en el modelo de mapeo [151]. En el Anexo B se muestra la transformaci'on implementada entre un modelo *DMKB* a su correspondiente modelo *RDF*

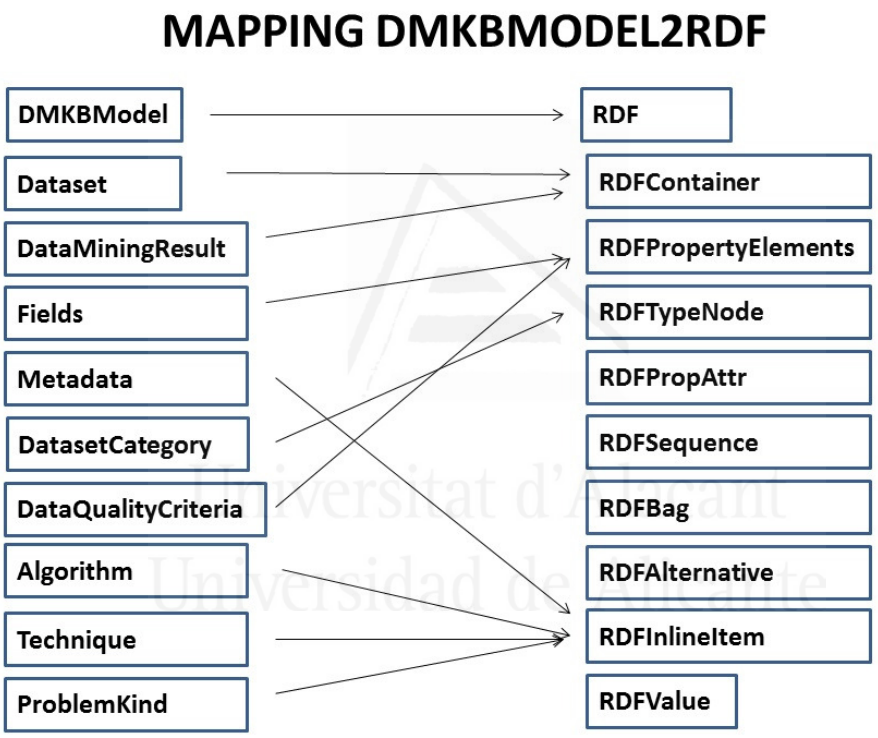


FIGURA 7.5: Transformación entre los elementos de ambos metamodelos.

La ejecución de esta transformación devuelve un modelo RDF que puede ser serializado en un documento RDF conteniendo la descripción de los objetos EMF. A continuación, se presenta un resumen del documento RDF obtenido 7.2.

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:foaf="http://xmlns.com/foaf/0.1/">
<rdf:RDFcontainer name="DMKB:Dataset" value="audiology"/>
 rdf:about="http://audiology.data.gov.uk/data/audiology/2012">
 <rdf:RDFinlineItem names="DMKB:Metadata_Instances" value="226.0"/>
 <rdf:RDFinlineItem names="DMKB:Metadata_Attributes" value="70.0"/>
 <rdf:RDFinlineItem names="DMKB:Metadata_Percentage of Numeric
  Attributes" value="0.0"/>
 <rdf:RDFinlineItem names="DMKB:Metadata_Percentage of Nominal
  Attributes" value="100.0"/>
 <rdf:RDFinlineItem names="DMKB:Metadata_Number of classes"
```

```
12     value="24.0"/>
13    ...
14  </rdf:RDFcontainer>
15  <rdf:RDFcontainer name="DMKB:DataMiningResult_Correctos"
16    value="192.0">
17    <rdf:RDFinlineItem names="DMKB:Algorithm_1" value="AdaBoostM1"/>
18    <rdf:RDFinlineItem names="DMKB:Technique_1" value="Classification"
19      parseResource="Classification"/>
20  </rdf:RDFcontainer>
21  <rdf:RDFcontainer name="DMKB:DataMiningResult_Correctos" value="172.0">
22    <rdf:RDFinlineItem names="DMKB:Algorithm_2" value="END"/>
23    <rdf:RDFinlineItem names="DMKB:Technique_2" value="Classification"
24      parseResource="Classification"/>
25  </rdf:RDFcontainer>
26  ...
27  <rdf:RDFpropertyElement aID="DMKB:DataSetDataQualityCriteria"
28    otherID="Null Values" anID="Percentage of null values" value="2.0"
29    oID="DMKB:DataSetDataQualityCriteria" ot="Null Values"/>
30  <rdf:RDFpropertyElement aID="DMKB:DataSetDataQualityCriteria"
31    otherID="UnbalanceColumns" anID="Percentage of unbalance columns"
32    value="71.43" oID="DMKB:DataSetDataQualityCriteria"
33    ot="UnbalanceColumns"/>
34  <rdf:RDFpropertyElement aID="DMKB:DataSetDataQualityCriteria"
35    otherID="Average Entropy" anID="Average Entropy" value="0.1428"
36    oID="DMKB:DataSetDataQualityCriteria" ot="Average Entropy"/>
37  ...
38  <rdf:RDF/>
```

Código 7.2: Ejemplo de un modelo RDF creado

El conocimiento obtenido es parte de la filosofía de datos abiertos, siendo publicado y listo para ser utilizado para otros usuarios. El conocimiento obtenido forma como una nueva capa en los mismos datos para futuro análisis. Cuando el proceso retorne al punto inicial, como una espiral, con toda la información generada, daría lugar a un nuevo giro sobre los datos. El ciclo es cerrado ya que esta información puede ser examinada y enriquecida en la medida en que los datos se analicen por nuestra propuesta.

El conocimiento será enriquecido cada vez que nuestra propuesta sea utilizada. Al ser éste un proceso iterativo, más conocimiento será generado cada vez, generando un efecto como cuando llega la "Primavera", permitiendo el florecimiento del conocimiento, por lo que nuestra propuesta es titulada: *Knowledge Spring Process*. En este caso, el término *Spring* tiene un doble sentido, permitiendo que el conocimiento pueda generarse de manera cíclica como un resorte, y el segundo, permitiendo su florecimiento como en la primavera. En nuestros días, es esencial que los usuarios inexpertos pueden aprovechar la gran cantidad de información disponible con el fin de extraer conocimiento y tomar decisiones bien sustentadas. El valor del conocimiento descubierto podría ser mayor si estuviese disponible para su posterior consumo. Este trabajo ha sido publicado en [152] y es la primera versión de KSP, una infraestructura que permite a los usuarios inexpertos aplicar técnicas de minería de datos de forma amigable sobre archivos de datos abiertos.

La principal contribución de esta propuesta es el concepto de la reutilización de los conocimientos obtenidos a partir de procesos de minería de datos después de que han sido anotados semánticamente en un archivo RDF (*Linked Open Knowledge*). Ha sido usado un enfoque basado en modelos con el fin de mantener una estructura estándar que tenga en cuenta la diversidad de los formatos de datos existentes. Como trabajo futuro, se tiene la intención de mejorar el proceso de obtención de LOK.

7.2. Taxonomía de requisitos para la minería de datos por parte de usuarios inexpertos

La extracción de conocimiento es un proceso complejo, compuesto por un conjunto contínuo de pasos a aplicar sobre las fuentes de datos. Debido a esto, este proceso solo está al alcance de personas expertas que tienen total dominio de los conceptos que lo definen. Otras de las razones causantes de que los usuarios inexpertos no puedan ser partícipes por si solos en la obtención de conocimiento es la dificultad que encuentran para expresar cuales son las metas a alcanzar al querer analizar los datos. Dicho de otra manera, los usuarios inexpertos no cuentan con los mecanismos que le permitan expresar sus requerimientos al aplicar técnicas de minería de datos. Por ejemplo, que tipo de conocimiento desean descubrir al analizar sus datos.

En este apartado se propone una taxonomía de requisitos para minería de datos que permita solucionar esta problemática. El objetivo es presentar un mecanismo que permita guiar al usuario a través de preguntas en la selección de que es lo que desea hacer, con la meta de poder resolver sus expectativas al finalizar el proceso. Teniendo en cuenta que estamos tratando con usuarios inexpertos en minería, nuestra taxonomía debe brindar un ambiente amigable que permita transformar las expectativas iniciales de los usuarios en requisitos de minería, logrando de esta manera una correcta ejecución del proceso KDD.

Los elementos que forman la taxonomía creada han sido identificados a partir de un detallado estudio teórico de los conceptos fundamentales relacionados con la temática y, provenientes de la experiencia acumulada en el área. Se vincularon en una única estructura los conceptos identificados que forman parte del proceso KDD, con sus posibles valores en cada caso. Esta taxonomía ha sido diseñada no

solo pensando en las funcionalidades de cada concepto representado, sino también en los objetivos que permite cumplir. Se trató de no utilizar términos específicos, sino un lenguaje sencillo, pensando siempre en que los usuarios principales para los que está pensada no son expertos en minería de datos.

La taxonomía de requisitos diseñada es mostrada en la Fig. 7.6. Como se puede apreciar presenta una estructura arbórea formada por nodos y sus respectivos arcos, donde se vinculan los conceptos del proceso de minería con las posibles respuestas en cada caso. Los nodos serían las preguntas y los arcos que nacen de un nodo corresponden a los posibles valores del atributo considerado en ese nodo, es decir, las posibles respuestas.

En la medida que se agreguen más requisitos crecerán los niveles del árbol, incrementándose de esta forma la información que puede ser obtenida de los usuarios. Cada arco conduce a otro nodo de decisión o a un nodo hoja. Los nodos hoja representan la predicción (o clase) del problema para todas aquellas instancias que logran esa hoja. Para clasificar una instancia desconocida, se recorre el árbol de arriba hacia abajo de acuerdo a los valores de los atributos probados en cada nodo y, cuando se llega a una hoja, la instancia se clasifica con la clase indicada por esa hoja.

La taxonomía está formada por los siguientes pasos:

1. El primer paso es seleccionar la fuente de datos que será analizada. Luego, la estructura de la fuente de datos podrá ser leída, y la composición de sus atributos será también conocida. El formato de la fuente de datos debe ser un archivo con extensión *.arff*.

2. Las técnicas de minería son agrupadas en dos tipos de modelos: predictivos y descriptivos. Los modelos predictivos pretenden estimar valores futuros o desconocidos de variables de interés. Por ejemplo, un modelo predictivo sería aquel que permita estimar la categoría de los clientes de acuerdo a los gastos frecuentes de estos en un supermercado. Los modelos descriptivos identifican patrones que explican o resumen los datos, es decir, sirven para explorar las propiedades de los datos examinados. Por ejemplo, un supermercado desea identificar grupos de personas con gustos similares, con el objeto de organizar diferentes ofertas para cada grupo y poder así remitirles esta información; para ello analiza las compras que han realizado sus clientes e infiere un modelo

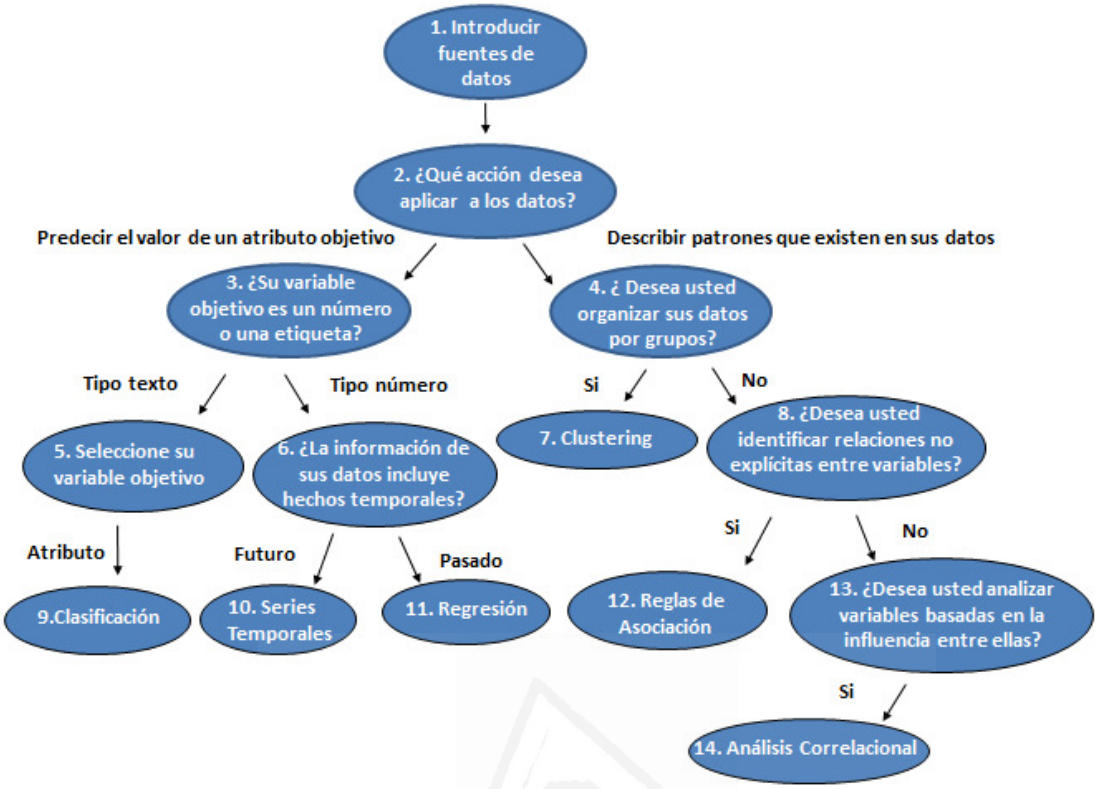


FIGURA 7.6: Taxonomía para ayudar a usuarios inexpertos a especificar sus requisitos de minería de datos.

descriptivo que caracteriza estos grupos. Para evitarnos estas explicaciones formales en la taxonomía, la pregunta que se le hace al usuario es: "¿Qué acción desea aplicarle a sus datos?". Las posibles respuestas a esta pregunta son:

a) Predecir el valor de un atributo que coincida con una respuesta real, en este caso, un modelo predictivo.

b) Describir patrones que existan en los datos que coincidan con una respuesta real, en este caso, un modelo descriptivo.

3. Si el usuario selecciona un modelo predictivo, la próxima pregunta está asociada al tipo de datos del atributo que desea predecir. Por lo que la pregunta es: "¿Su variable objetivo es un número o una etiqueta?. Cada posible respuesta es mostrada para que el usuario inexperto pueda identificar patrones similares respecto a los valores a predecir en sus datos. Las posibles respuestas son:

a) Tipo Texto. Un ejemplo de atributo a predecir puede ser "Calidadde-Servicio", y sus valores son representados por cadenas como: "baja", "media" o "alta".

b) Tipo Número. Patrones similares con valores númericos son mostrados. Si el atributo a predecir es "Temperatura" en grados celsius, los valores pueden ser números como "25", "14" o "10".

En el caso que el atributo seleccionado sea nominal el próximo paso es mostrarle al usuario la lista de atributos de tipo nominal a partir de la fuente de datos introducida por él en el sistema. En este punto, con las características previamente seleccionadas solamente pueden ser aplicadas técnicas de clasificación. Estas se utilizan cuando se desea representar, categorizar, dar, asignar, calificar un atributo en dependencia del valor de otros.

Por otra parte, si el atributo seleccionado fue númerico, la manera más simple de hacer una pregunta con vistas a obtener lo que el usuario quiere hacer es: "¿La información de sus datos incluye hechos temporales?". Las posibles respuestas son: "Futuro" o "Pasado".

a) En el caso que la opción seleccionada sea predecir el futuro, la técnica a aplicar es "Series Temporales". Esta técnica permite el estudio de la evolución de una variable a través del tiempo para poder realizar predicciones, a partir de ese conocimiento.

b) La respuesta en el caso que el usuario desee analizar el pasado, es la técnica de "Regresión". Esta técnica relaciona una o más variables con un conjunto de variables predictoras.

4. En el caso que el usuario seleccionó aplicar un modelo descriptivo la primera pregunta es: "¿Desea usted organizar sus datos por grupos?".

a) Si la respuesta seleccionada es "Sí", el usuario desea aplicar la técnica "Clustering". Esta técnica permite la clasificación de una población de individuos caracterizados por múltiples atributos en un número determinado de grupos, con base en las semejanzas o diferencias de los individuos.

b) Si la respuesta seleccionada es "No", la próxima pregunta es: "Desea usted identificar relaciones no explícitas entre variables?"

1) Si la respuesta seleccionada es "Yes", el usuario desea aplicar la técnica "Reglas de Asociación". Esta técnica permite identificar relaciones no explícitas entre atributos categóricos, o combinaciones de valores de los atributos que suceden más frecuentemente.
2) Si la respuesta seleccionada es "No", la próxima pregunta es: "Desea usted analizar variables basadas en la influencia entre ellas?". La única opción de respuesta en este caso será Sí, siendo la respuesta a esta pregunta que el usuario desea aplicar "Análisis Correlacional". Esta técnica es utilizada para examinar el grado de similitud de los valores de dos variables númericas.

Después de usar la taxonomía para determinar la técnica de minería que utilizará, el usuario inexperto necesita seleccionar un algoritmo de minería específico. Para ello, se ejecutará el recomendador diseñado 4.3. El recomendador devolverá el modelo de minería obtenido luego de ejecutar el algoritmo sugerido por el recomendador.

7.3. Otros trabajos futuros

Teniendo en cuenta el trabajo de investigación realizado, consideramos que existen varios trabajos que deben ser estudiados a partir de la propuesta desarrollada en esta tesis. A continuación establecemos una lista de trabajos futuros que pueden derivar de esta tesis:

- Aunque de manera general se obtuvieron buenos resultados, se debe tener en cuenta continuar con el diseño de experimentos para mejorar la precisión del recomendador bajo otras circunstancias, como la presencia de más cantidad de clasificadores, y la realización de experimentos más complejos. Así como la inclusión de nuevas meta-características para comprobar su incidencia en el proceso de minería.
- Interpretación de los resultados de minería por usuarios inexpertos. Cuando se ejecuta un algoritmo de minería se genera mucha información, por lo que a veces se hace muy difícil interpretar los resultados. Esta línea de investigación está enfocada a reducir en la medida de lo posible este incoveniente.

- Extensión de la propuesta a otras técnicas de minería.
- Estudiar y analizar la percepción del usuario al utilizar nuestra propuesta a gran escala.

Capítulo 8

Conclusiones

En este capítulo se presentan las conclusiones finales de esta tesis. Comprobaremos el cumplimiento de la hipótesis de trabajo y de los objetivos planteados en la primera parte de esta tesis. Estableceremos las discusiones surgidas a lo largo de este trabajo y finalmente resumiremos las principales contribuciones que han derivado de la presente tesis. Finalmente, se muestran los resultados alcanzados durante la investigación.

8.1. Conclusiones

El trabajo de investigación desarrollado en la presente tesis ha tenido como objetivo desarrollar una propuesta que permita a usuarios inexpertos facilitar la obtención de conocimiento al aplicar técnicas de minería de datos, específicamente técnicas de clasificación. Dicha propuesta ha sido implementada utilizando flujos de trabajo científicos para lograr mayor alcance en la comunidad científica. Siendo el capítulo final de esta tesis, presentamos las principales aportaciones:

- Identificación de criterios de calidad que inciden en los resultados de minería al aplicar técnicas de clasificación: A partir de estudiar trabajos publicados y analizar los criterios de calidad de datos existentes en el estándar ISO/IEC 25012, se propusieron varios criterios de calidad para técnicas de clasificación.
- Demostración de la influencia de criterios de calidad al aplicar técnicas de clasificación: A partir de experimentos realizados en un caso de estudio se

comprobó la validación práctica de los criterios definidos. Definiendo la necesidad de tenerlos en cuenta en etapas iniciales del proceso de descubrimiento de conocimiento.

- Diseño del metamodelo *DMKB*, donde se reprentan los elementos necesarios para almacenar los resultados de experimentos de minería. A partir del metamodelo diseñado se creó la base de conocimientos conformada por un conjunto de modelos a partir de las fuentes de datos analizadas. El objetivo de la base de conocimiento es habilitar la aplicación de técnicas de minería de datos amigable, permitiendo representar de una manera estructurada y homogénea los elementos necesarios de minería de datos que se presentan en el proceso KDD.

- Propuesta de desarrollo de software dirigido por modelos para permitir la aplicación de técnicas de minería de datos a usuarios inexpertos utilizando flujos de trabajos científicos.

- Se crearon los flujos de trabajos necesarios para que los usuarios expertos pudieran alimentar la base de conocimientos, así como configurar los parámetros necesarios y la construcción del recomendador.

- Construcción del flujo de trabajo para que el usuario inexperto pueda utilizar analizar sus datos y obtener el mejor modelo de minería recomendado por el recomendador. La aplicación del algoritmo de minería devuelto por el sistema recomendador permite que el usuario inexperto obtenga directamente el modelo de minería aconsejado, teniendo en cuenta la calidad de sus datos y la información histórica almacenada en la base de conocimiento.

- Aplicación de la propuesta a tres casos de estudio reales: Para cada uno de los casos de estudio el objetivo fue comprobar el desempeño de nuestra propuesta, teniendo en cuenta el resultado obtenido por el recomendador y los obtenidos por los expertos en cada caso. En cada caso se comprobó la validación de nuestra propuesta, teniendo en cuenta que los resultados obtenidos se asemejan bastante a los óptimos que obtuvieron los usuarios expertos. De esta forma el usuario inexperto obtiene un modelo de minería considerablemente bueno y en un tiempo mucho más corto que el comparado con el que tardaría un usuario experto.

8.2. Resultados de investigación

En este apartado se muestran las publicaciones obtenidas durante el período de desarrollo de esta tesis doctoral y las colaboraciones realizadas en proyectos de investigación.

8.2.1. Producción científica

En la Tabla 8.1 aparece la cronología de las publicaciones realizadas en el marco de la presente tesis. Podemos mencionar un total de 11 trabajos publicados como producción científica de esta tesis, de los cuales 1 ha sido enviado a *Lecture Notes in Business Information Processing (LNBIP)*, 6 en congresos internacionales y 4 en congresos o talleres nacionales.

Año	Contribución	Ámbito
2010	ADIS'10	Congreso Nacional
2011	CICCI'11	Congreso Nacional
	ICCSA'11	Congreso Internacional
	JISBD'11	Congreso Nacional
2012	EDBT/ICDT'12	Workshop Internacional
	JISBD'12	Congreso Nacional
	ICWE'12	Workshop
	MTSR'12	Congreso Internacional
2013	SIMPDA'13	Congreso Internacional
2014	DATA'14	Congreso Internacional
	LNBIP'14 (Enviado)	Revista

Tabla 8.1: Cronología de las contribuciones

- Roberto Espinosa, José Jacobo Zubcoff, Marta Elena Zorilla, Jose Noberto Mazón, Hacia la consideración de aspectos de calidad de datos en procesos de minería: el caso de las técnicas de clasificación. ADIS 2010. Actas de los Talleres de las Jornadas de Ingeniería del Software y Bases de Datos. Vol.4 No1. 2010. ISSN 1988-3455.

- Roberto Espinosa, José Jacobo Zubcoff, Marta Elena Zorilla, Jose Noberto Mazón, Aspectos de calidad de datos en procesos de minería: el caso de las técnicas de clasificación. (2011). XIV Convención y Feria Internacional Informática. Conferencia Internacional de Ciencias Computacionales e Informáticas (CICCI' 2011).

- Roberto Espinosa, José Jacobo Zubcoff, Jose-Norberto Mazón: A Set of Experiments to Consider Data Quality Criteria in Classification Techniques for Data Mining. ICCSA (2) 2011: 680-694.

- Roberto Espinosa, José Jacobo Zubcoff, Jose Noberto Mazón, Towards a reverse engineering approach for guiding user in applying data mining. JISBD2011 XVI Jornadas de Ingeniería de Software y Bases de Datos. Coruña, España. ISBN: 978-84-9749-486-1.

- Jose-Norberto Mazón, José Jacobo Zubcoff, Irene Garrigós, Roberto Espinosa, Rolando Rodríguez: Open business intelligence: on the importance of data quality awareness in user-friendly data mining. EDBT/ICDT Workshops 2012: 144-147.

- Jose Norberto Mazon, José Zubcoff, Irene Garrigos, Roberto Espinosa y Rolando Rodríguez, Open Business Intelligence: calidad de datos para un uso amigable de técnicas de minería de datos. JISBD 2012 (XVII Jornadas de Ingeniería del Software y de Bases de Datos) Almería, España. ISBN: 978-84-15487-28-9.

- Rolando Rodríguez, Roberto Espinosa, Devis Bianchini, Irene Garrigós, Jose-Norberto Mazón, José Jacobo Zubcoff: Extracting Models from Web API Documentation. ICWE Workshops 2012: 134-145.

- Roberto Espinosa, Diego García-Saiz, José Jacobo Zubcoff, Jose-Norberto Mazón, Marta E. Zorrilla: Towards the Development of a Knowledge Base for Realizing User-Friendly Data Mining. Metadata and Semantics Research Conference. MTSR 2012: 121-126.

- Roberto Espinosa, Diego García-Saiz, Marta E. Zorrilla, José Jacobo Zubcoff, Jose-Norberto Mazón: Development of a Knowledge Base for Enabling Non-expert Users to Apply Data Mining Algorithms. International Symposium on Data-Driven Process Discovery and Analysis. SIMPDA 2013: 46-61.

- Roberto Espinosa, Larisa Garriga, José Jacobo Zubcoff, Jose-Norberto Mazón: Knowledge Spring Process - Towards Discovering and Reusing Knowledge within Linked Open Data Foundations. International Conference on Data Management Technologies and Applications. DATA 2014: 291-296.

- Roberto Espinosa Oliva, Diego García-Saiz, Marta Zorrilla, José Jacobo Zubcoff and Jose Norberto Mazón. Enabling non-expert users to apply data mining for bridging the big data divide. Lecture Notes in Business Information Processing. Series Editors: van der Aalst, W., Mylopoulos, J., Rosemann, M.,Shaw, M.J., Szyperski, C. ISSN: 1865-1348.

8.2.2. Proyectos relacionados con la tesis doctoral

Durante el desarrollo de esta tesis doctoral se ha contribuido en los siguientes proyectos de investigación:

- IN.MIND: GRE11-28 "Ingeniería inversa de datos para guiar al usuario en la aplicación de técnicas de Minería de Datos (IN.MIND)".
- OPEN.MIND: "Minería de datos abiertos (Open.Mind)" (GV/2014/098)

Apéndice A

Fichero resultado del recomendador caso estudio e-learning.

Este fichero es el creado luego de haber evaluado el recomendador sobre el fichero de prueba. En el atributo *bestAcc* se tiene el algoritmo dado por el experto, y en el atributo *predictedbestAcc* se obtiene el algoritmo sugerido por el recomendador para cada fuente de datos.


```
@relation 'metalearning9classifiers'
@attribute numAtt numeric
@attribute numIns numeric
@attribute numClasses numeric
@attribute predictedbestAcc{J48,NaiveBayes,BayesNet,RandomForest,OneR,
JRip,Ridor,DecisionTable,NNge}
@attribute bestAcc{J48,NaiveBayes,BayesNet,RandomForest,OneR,JRip,Ridor,
DecisionTable,NNge}
@attribute hasMissingValues {yes,no}
@attribute numMissing numeric
@attribute percMissing numeric
@attribute isbalanced {balanced,quite_unbalanced,unbalanced}
@data
12,126,2,J48,J48,no,0,0,quite_unbalanced
22,1488,2,NNge,NNge,no,0,0,quite_unbalanced
22,1488,3,NNge,NNge,no,0,0,unbalanced
20,83,4,DecisionTable,BayesNet,no,0,0,unbalanced
21,115,5,J48,J48,no,0,0,unbalanced
```

```
22,217,3,Ridor,Ridor,no,0,0,unbalanced
18,231,5,NNge,NNge,yes,1048,0.252044,unbalanced
19,836,2,DecisionTable,J48,yes,4708,0.296399,balanced
```

Apéndice B

Transformación del modelo *DMKB* a modelo *RDF*.

En este caso se pretende obtener el fichero *RDF* a partir del modelo de minería *DMKB* para reutilizar el conocimiento adquirido.

```
module dmkb2rdf;
create OUT : rdf from IN : dmkb;
rule datasetdataqualitycriteria2rdfpropropertyElement{
from dmkb:dmkb!DataQualityCriteria(dmkb.dsdqc2rdfpropropertyCondition())
to   rdf:rdf!RDFpropertyElt(  aID <- 'DMKB:DataSetDataQualityCriteria',
otherID <- dmkb.name,
anID <- dmkb.description,
value <- dmkb.value,
oID <- 'DMKB:DataSetDataQualityCriteria',
ot <- dmkb.name)}
rule metadata2rdfinlineitem{
from dmkb:dmkb!Metadata
to  rdf:rdf!RDFinlineItem( names <-
'DMKB:Metadata_' + dmkb.name.toString(),
     value <- dmkb.value.toString())}
rule dataSet2container{
from dmkb:dmkb!DataSet
to   rdf:rdf!RDFcontainer( name <- 'DMKB:Dataset',
 value <- dmkb.name,
 inline <- dmkb.GetInlineItemsMetadata())}
```

```
rule algorithm2rdfinlineitem{
from dmkb:dmkb!Algorithm
to   rdf:rdf!RDFinlineItem(names <-
'DMKB:Algorithm_' + dmkb.GetParentDMR().Id.toString(),
   value <- dmkb.name)}
rule technique2rdfinlineitem{
from dmkb:dmkb!Technique
to   rdf:rdf!RDFinlineItem( names <-
'DMKB:Technique_'+dmkb.GetParentAlgT().GetParentDMR().Id.toString(),
 value <- dmkb.problemKind.name,
 parseResource <- dmkb.name)}
rule parameter2rdfinlineitem{
from dmkb:dmkb!Parameter
to   rdf:rdf!RDFinlineItem(names <-'DMKB:Parameter_'+
dmkb.GetParentAlgP().GetParentDMR().Id.toString(),
 value <- dmkb.value,
 parseResource <- dmkb.name)}
rule dataminingresult2rdfcontainer{
from dmkb:dmkb!DataMiningResults
to   rdf:rdf!RDFcontainer( name <- 'DMKB:DataMiningResult_' + dmkb.name,
 value <- dmkb.value.toString(),
 inline <- dmkb.GetInlineItemsDMR())}
rule dmkb2rdf{
from dmkb:dmkb!DMKBModel
to   rdf:rdf!RDF(container <- rdf!RDFcontainer.allInstances(),
             propertyElement <-rdf!RDFpropertyElt.allInstances())
}
helper context dmkb!DataQualityCriteria def: dsdqc2rdfpropropertyCondition():
Boolean=
if(self.oclIsTypeOf(dmkb!DatasetDataQualityCriteria))
  then true else false endif;

helper context dmkb!DataQualityCriteria def: fdqc2rdfpropropertyCondition():
Boolean=
if(self.oclIsTypeOf(dmkb!FieldDataQualityCriteria))
  then true else false endif;

helper context dmkb!DataSet def:GetInlineItemsMetadata():
Sequence(rdf!RDFinlineItem)=rdf!RDFinlineItem.allInstances() ->
```

```
select(i | i.parseResource='DMKB:Metadata');

helper context dmkb!DataMiningResults def:GetInlineItemsDMR():Sequence
(rdf!RDFinlineItem)=rdf!RDFinlineItem.allInstances() ->
select(i|i.names='DMKB:Algorithm_'+self.Id.toString())
.union(rdf!RDFinlineItem.allInstances() ->
select(i|i.names='DMKB:Technique_'+self.Id.toString()))
.union(rdf!RDFinlineItem.allInstances() ->
select(i|i.names='DMKB:Parameter_'
+self.Id.toString()));

helper context dmkb!Algorithm def:GetParentDMR():
dmkb!DataMiningResults =
dmkb!DataMiningResults.allInstances()->
select(i | i.algorithms.name=self.name).first();

helper context dmkb!Technique def:GetParentAlgT():dmkb!Algorithm =
   dmkb!Algorithm.allInstances()->
   select(i | i.technique=self).first();

helper context dmkb!Parameter def:GetParentAlgP():dmkb!Algorithm =
  dmkb!Algorithm.allInstances()->
  select(i | i.parameters.includes(self)).first();
```

Bibliografía

[1] John Gantz and David Reinsel. The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east. *IDC iView: IDC Analyze the Future*, 2012.

[2] Kate Crawford et al. Six provocations for big data. 2011.

[3] Daniel Abadi, Rakesh Agrawal, Anastasia Ailamaki, Magdalena Balazinska, Philip A. Bernstein, Michael J. Carey, Surajit Chaudhuri, Jeffrey Dean, An-Hai Doan, Michael J. Franklin, Johannes Gehrke, Laura M. Haas, Alon Y. Halevy, Joseph M. Hellerstein, Yannis E. Ioannidis, H.V. Jagadish, Donald Kossmann, Samuel Madden, Sharad Mehrotra, Tova Milo, Jeffrey F. Naughton, Raghu Ramakrishnan, Volker Markl, Christopher Olston, Beng Chin Ooi, Christopher Ré, Dan Suciu, Michael Stonebraker, Todd Walter, and Jennifer Widom. The beckman report on database research. *URL: http://beckman.cs.wisc.edu/beckman-report2013.pdf*, 2013.

[4] Lev Manovich. Trending: The promises and the challenges of big social data. *Debates in the digital humanities*, pages 460–75, 2011.

[5] Danah Boyd and Kate Crawford. Critical questions for big data: Provocations for a cultural, technological, and scholarly phenomenon. *Information, Communication & Society*, 15(5):662–679, 2012.

[6] Gene Bellinger, Durval Castro, and Anthony Mills. Data, information, knowledge, and wisdom. *URL: http://www. systems-thinking. org/dikw/dikw. htm*, page 47, 2004.

[7] Mike Gualtieri and Noel Yuhanna. The forrester wave^{TM}: Big data hadoop solutions, q1 2014 https://www.forrester.com/The+Forrester+Wave+Big+Data+Hadoop+Solutions+Q1+2014/fulltext/-/E-RES112461.

[8] Robert Nisbet, John Elder, and Gary Miner. *Handbook of Statistical Analysis and Data Mining Applications*. Academic Press, 2009. ISBN 0123747651, 9780123747655.

[9] Joaquin Vanschoren and Hendrik Blockeel. Stand on the Shoulders of Giants: Towards a Portal for Collaborative Experimentation in Data Mining. *International Workshop on Third Generation Data Mining at ECML PKDD*, 1: 88–89, September 2009.

[10] Hans-Peter Kriegel, Karsten M. Borgwardt, Peer Kröger, Alexey Pryakhin, Matthias Schubert, and Arthur Zimek. Future trends in data mining. *Data Min. Knowl. Discov.*, 15(1):87–97, 2007.

[11] Usama M. Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. Knowledge discovery and data mining: Towards a unifying framework. pages 82–88, 1996.

[12] Dorian Pyle. *Data preparation for data mining*, volume 1. Morgan Kaufmann, 1999.

[13] Kevin Strange. Etl was the key to this data warehouse's success. Technical report, Technical Report CS-15-3143, Gartner, 2002.

[14] Usama M. Fayyad, Gregory Piatetsky-Shapiro, and Padhraic Smyth. The kdd process for extracting useful knowledge from volumes of data. *Commun. ACM*, 39(11):27–34, 1996.

[15] Rakesh Agrawal, Tomasz Imieliński, and Arun Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD Record*, volume 22, pages 207–216. ACM, 1993.

[16] George EP Box, Gwilym M Jenkins, and Gregory C Reinsel. *Time series analysis: forecasting and control*. John Wiley & Sons, 2013.

[17] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.

[18] Jiawei Han and Micheline Kamber. *Data Mining: Concepts and Techniques*. Morgan Kaufmann, 2000. ISBN 1-55860-489-8.

[19] M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society. Series B (Methodological)*, 36(2):

111–147, 1974. ISSN 00359246. doi: 10.2307/2984809. URL http://dx.doi.org/10.2307/2984809.

[20] Jiawei Han, Micheline Kamber, and Jian Pei. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2012. ISBN 9780123814791.

[21] David L. Olson and Dursun Delen. *Advanced Data Mining Techniques*. Springer Publishing Company, Incorporated, 1st edition, 2008. ISBN 3540769161, 9783540769163.

[22] John Ross Quinlan. *C4. 5: programs for machine learning*, volume 1. Morgan kaufmann, 1993.

[23] Delgado M. Calvo-Flores, Gibaja E. Galindo, Pegalajar M. C. Jiménez, and Pérez O. Piñeiro. Predicting students' marks from Moodle logs using neural network models. *Current Developments in Technology-Assisted Education*, pages 586–590, 2006. URL http://www.formatex.org/micte2006/pdf/586-590.pdf.

[24] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: an update. *SIGKDD Explorations*, 11(1):10–18, 2009.

[25] Spss modeler, @BOOKLET, 2014. URL http://www-01.ibm.com/software/analytics/spss/products/modeler/.

[26] ZhaoHui Tang and Jamine Maclennan. *Data mining with SQL Server 2005*. John Wiley & Sons, 2005.

[27] Oracle data mining, @BOOKLET, 2014. URL http://www.oracle.com/technetwork/database/options/advanced-analytics/odm/index.html.

[28] Sas enterprise minner, @BOOKLET, 2014. URL http://www.sas.com/en_us/software/analytics/enterprise-miner.html.

[29] Graham Williams. *Data mining with Rattle and R: the art of excavating data for knowledge discovery*. Springer, 2011.

[30] P. González-Aranda, Ernestina Menasalvas Ruiz, Socorro Millán, Carlos Ruiz, and Javier Segovia. Towards a methodology for data mining project development: The importance of abstraction. In Tsau Young Lin, Ying

Xie, Anita Wasilewska, and Churn-Jung Liau, editors, *Data Mining: Foundations and Practice*, volume 118 of *Studies in Computational Intelligence*, pages 165–178. Springer, 2008. ISBN 978-3-540-78487-6.

[31] Pavel Brazdil, Christophe Giraud-Carrier, Carlos Soares, and Ricardo Vilalta. *Metalearning: Applications to Data Mining*. Springer Publishing Company, Incorporated, 1 edition, 2008. ISBN 3540732624, 9783540732624.

[32] Pavel Brazdil, João Gama, and Bob Henery. Characterizing the applicability of classification algorithms using meta-level learning. In *Machine Learning: ECML-94*, pages 83–102. Springer, 1994.

[33] Kalousis Alexandros and Hilario Melanie. Model selection via meta-learning: a comparative study. *International Journal on Artificial Intelligence Tools*, 10(04):525–554, 2001.

[34] Yonghong Peng, Peter A Flach, Pavel Brazdil, and Carlos Soares. Decision tree-based data characterization for meta-learning. 2002.

[35] Alexandros Kalousis, João Gama, and Melanie Hilario. On data and algorithms: Understanding inductive performance. *Machine Learning*, 54(3): 275–312, 2004.

[36] Suhas Gore and Nitin Pise. Dynamic algorithm selection for data mining classification.

[37] Ricardo BC Prudêncio, Marcilio CP De Souto, and Teresa B Ludermir. Selecting machine learning algorithms using the ranking meta-learning approach. In *Meta-Learning in Computational Intelligence*, pages 225–243. Springer, 2011.

[38] M. M. Molina, J. M. Luna, C. Romero, and S. Ventura. Meta-learning approach for automatic parameter tuning: A case study with educational datasets. In *Proceedings of the 5th International Conference on Educational Data Mining, EDM 2012*, pages 180–183, 2012.

[39] Bruno Feres De Souza, André C. P. L. F. De Carvalho, and Carlos Soares. Empirical evaluation of ranking prediction methods for gene expression data classification. In *Proceedings of the 12th Ibero-American conference on Advances in artificial intelligence*, IBERAMIA'10, pages 194–203, Berlin,

Heidelberg, 2010. Springer-Verlag. ISBN 3-642-16951-1, 978-3-642-16951-9. URL http://dl.acm.org/citation.cfm?id=1948131.1948159.

[40] Marcílio Carlos Pereira de Souto, Ricardo Bastos Cavalcante Prudêncio, Rodrigo G. F. Soares, Daniel S. A. de Araujo, Ivan G. Costa, Teresa Bernarda Ludermir, and Alexander Schliep. Ranking and selecting clustering algorithms using a meta-learning approach. In *IJCNN*, pages 3729–3735. IEEE, 2008.

[41] Robin Burke. Knowledge-based recommender systems. *Encyclopedia of library and information systems*, 69(Supplement 32):175–186, 2000.

[42] Enrique García Salcines, Cristóbal Romero, Sebastián Ventura, and Carlos de Castro-Lozano. Sistema recomendador colaborativo usando minería de datos distribuida para la mejora continua de cursos e-learning. *IEEE-RITA*, 3(1):19–30, 2008.

[43] Roy T Fielding and Richard N Taylor. Principled design of the modern web architecture. *ACM Transactions on Internet Technology (TOIT)*, 2(2): 115–150, 2002.

[44] S. Beydeda, M. Book, and V. Gruhn. *Model-Driven Software Development*. Springer, 2005. ISBN 9783540285540. URL http://books.google.es/books?id=7MqHFDvuER0C.

[45] Anneke G. Kleppe, Jos Warmer, and Wim Bast. *MDA Explained: The Model Driven Architecture: Practice and Promise*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2003. ISBN 032119442X.

[46] Donald E. Knuth. Backus normal form vs. backus naur form. *Commun. ACM*, 7(12):735–736, December 1964. ISSN 0001-0782. doi: 10.1145/355588.365140. URL http://doi.acm.org/10.1145/355588.365140.

[47] Aleš Wojnar, Irena Mlỳnková, and Jiří Dokulil. Structural and semantic aspects of similarity of document type definitions and xml schemas. *Information Sciences*, 180(10):1817–1836, 2010.

[48] Colin Atkinson and Thomas Kuhne. Model-driven development: a metamodeling foundation. *Software, IEEE*, 20(5):36–41, 2003.

[49] Dave Thomas. Mda: Revenge of the modelers or uml utopia? *Software, IEEE*, 21(3):15–17, 2004.

[50] Object Management Group. Unified Modeling Language Specification 1.5. http://www.omg.org/cgi-bin/doc?formal/03-03-01, .

[51] Object Management Group. Meta Object Facility Specification http://doc.omg.org/formal/2006-01-01.pdf, .

[52] Anneke Kleppe, Jos Warmer, and Wim Bast. *MDA Explained. The Practice and Promise of The Model Driven Architecture*. Addison Wesley, 2003.

[53] Jean Bézivin. On the unification power of models. *Software and System Modeling*, 4(2):171–188, 2005.

[54] Carlo Batini, Cinzia Cappiello, Chiara Francalanci, and Andrea Maurino. Methodologies for data quality assessment and improvement. *ACM Comput. Surv.*, 41(3), 2009.

[55] Ahmed K Elmagarmid, Panagiotis G Ipeirotis, and Vassilios S Verykios. Duplicate record detection: A survey. *Knowledge and Data Engineering, IEEE Transactions on*, 19(1):1–16, 2007.

[56] Rohit Ananthakrishna, Surajit Chaudhuri, and Venkatesh Ganti. Eliminating fuzzy duplicates in data warehouses. In *Proceedings of the 28th international conference on Very Large Data Bases*, pages 586–597. VLDB Endowment, 2002.

[57] Paolo Missier and Carlo Batini. A multidimensional model for information quality in cooperative information systems. In *IQ*, pages 25–40, 2003.

[58] Diane M. Strong, Yang W. Lee, and Richard Y. Wang. 10 potholes in the road to information quality. *IEEE Computer*, 30(8):38–46, 1997.

[59] Ee-Peng Lim, Jaideep Srivastava, Satya Prabhakar, and James Richardson. Entity identification in database integration. In *Data Engineering, 1993. Proceedings. Ninth International Conference on*, pages 294–301. IEEE, 1993.

[60] Diane M. Strong, Yang W. Lee, and Richard Y. Wang. Data quality in context. *Commun. ACM*, 40(5):103–110, 1997.

[61] Roald K Pearson. Data mining in face of contaminated and incomplete records. In *Proc. of SIAM Intl. Conf. Data Mining*, 2002.

[62] Erhard Rahm and Hong Hai Do. Data cleaning: Problems and current approaches. *IEEE Data Eng. Bull.*, 23(4):3–13, 2000.

[63] Olga G. Troyanskaya, Michael Cantor, Gavin Sherlock, Patrick O. Brown, Trevor Hastie, Robert Tibshirani, David Botstein, and Russ B. Altman. Missing value estimation methods for dna microarrays. *Bioinformatics*, 17 (6):520–525, 2001.

[64] Edwin M Knox and Raymond T Ng. Algorithms for mining distancebased outliers in large datasets. In *Proceedings of the International Conference on Very Large Data Bases*, pages 392–403. Citeseer, 1998.

[65] Dimitri Theodoratos and Mokrane Bouzeghoub. Data currency quality satisfaction in the design of a data warehouse. *International Journal of Cooperative Information Systems*, 10(03):299–326, 2001.

[66] Colin Atkinson and Thomas Kühne. Model-driven development: A metamodeling foundation. *IEEE Software*, 20(5):36–41, 2003.

[67] Sunita Sarawagi. Special issue on data cleaning. *IEEE Data Engineering Bulletin*, 23(4), 2000.

[68] Roger H. L. Chiang, Terence M. Barron, and Veda C. Storey. Reverse engineering of relational databases: Extraction of an eer model from a relational database. *Data Knowl. Eng.*, 12(2):107–142, 1994.

[69] Matthias Jarke and Yannis Vassiliou. Data warehouse quality: A review of the dwq project. In Diane M. Strong and Beverly K. Kahn, editors, *IQ*, pages 299–313. MIT, 1997.

[70] Xingquan Zhu, Taghi M. Khoshgoftaar, Ian Davidson, and Shichao Zhang. Editorial: Special issue on mining low-quality data. *Knowl. Inf. Syst.*, 11: 131–136, February 2007. ISSN 0219-1377. doi: 10.1007/s10115-006-0058-y. URL http://portal.acm.org/citation.cfm?id=1229087.1229093.

[71] Roberto Espinosa, José Jacobo Zubcoff, and Jose-Norberto Mazón. A set of experiments to consider data quality criteria in classification techniques for data mining. In *ICCSA (2)*, pages 680–694, 2011.

[72] Laure Berti-Equille. Measuring and modelling data quality for quality-awareness in data mining. In Fabrice Guillet and Howard J. Hamilton, editors, *Quality Measures in Data Mining*, volume 43 of *Studies in Computational Intelligence*, pages 101–126. Springer, 2007. ISBN 978-3-540-44911-9.

[73] Yan Zhao. On interactive data mining. In John Wang, editor, *Encyclopedia of Data Warehousing and Mining*, pages 1085–1090. IGI Global, 2009. ISBN 9781605660103.

[74] Chunqiu Zeng, Yexi Jiang, Li Zheng, Jingxuan Li, Lei Li, Hongtai Li, Chao Shen, Wubai Zhou, Tao Li, Bing Duan, et al. Fiu-miner: a fast, integrated, and user-friendly system for data mining in distributed environment. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1506–1509. ACM, 2013.

[75] Harry Dimitropoulos, Herald Kllapi, Omiros Metaxas, Nikolas Oikonomidis, Eva Sitaridi, Manolis M Tsangaris, and Yannis Ioannidis. Aition: a scalable platform for interactive data mining. In *Scientific and Statistical Database Management*, pages 646–651. Springer, 2012.

[76] Salvatore Camiolo and Andrea Porceddu. gff2sequence, a new user friendly tool for the generation of genomic sequences. *BioData mining*, 6(1):15, 2013.

[77] Stefanie De Bodt, Diana Carvajal, Jens Hollunder, Joost Van den Cruyce, Sara Movahedi, and Dirk Inzé. Cornet: a user-friendly tool for data mining and integration. *Plant physiology*, 152(3):1167–1179, 2010.

[78] Marta E. Zorrilla and Diego García-Saiz. A service oriented architecture to provide data mining services for non-expert data miners. *Decision Support Systems*, 55(1):399–411, 2013.

[79] Diego García-Saiz, Camilo Palazuelos, and Marta E. Zorrilla. Data mining and social network analysis in the educational field: An application for non-expert users. In *Educational Data Mining*, pages 411–439. Springer, 2014.

[80] Pance Panov, Larisa N. Soldatova, and Saso Dzeroski. Towards an ontology of data mining investigations. In *Discovery Science*, pages 257–271, 2009.

[81] Larisa Soldatova and Ross D. King. An ontology of scientific experiments. *J R Soc Interface*, 3(11):795–803, 2006.

[82] Melanie Hilario, Alexandros Kalousis, Phong Nguyen, and Adam Woznica. A data mining ontology for algorithm selection and meta-mining. In *ECML/PKDD09 Workshop on Third Generation Data Mining: Towards Service-Oriented Knowledge Discovery*, SoKD-09, pages 76–87, 2009.

[83] Monika Záková, Petr Kremen, Filip Zelezný, and Nada Lavrac. Automating knowledge discovery workflow composition through ontology-based planning. *IEEE T. Automation Science and Engineering*, 8(2):253–264, 2011.

[84] Jörg-Uwe Kietz, Floarea Serban, Abraham Bernstein, and Simon Fischer. Designing kdd-workflows via htn-planning. In Luc De Raedt, Christian Bessière, Didier Dubois, Patrick Doherty, Paolo Frasconi, Fredrik Heintz, and Peter J. F. Lucas, editors, *ECAI*, volume 242 of *Frontiers in Artificial Intelligence and Applications*, pages 1011–1012. IOS Press, 2012. ISBN 978-1-61499-097-0.

[85] Claudia Diamantini, Domenico Potena, and Emanuele Storti. Ontology-driven kdd process composition. In *IDA*, pages 285–296, 2009.

[86] Melanie Hilario, Phong Nguyen, Huyen Do, Adam Woznica, and Alexandros Kalousis. Ontology-based meta-mining of knowledge discovery workflows. In *Meta-Learning in Computational Intelligence*, pages 273–315. 2011.

[87] Joaquin Vanschoren and Larisa Soldatova. Exposé: An ontology for data mining experiments. In *International Workshop on Third Generation Data Mining: Towards Service-oriented Knowledge Discovery (SoKD-2010),*, pages 31–46, September 2010.

[88] Joaquin Vanschoren, Hendrik Blockeel, Bernhard Pfahringer, and Geoffrey Holmes. Experiment databases - a new way to share, organize and learn from experiments. *Machine Learning*, 87(2):127–158, 2012.

[89] Hendrik Blockeel and Joaquin Vanschoren. Experiment databases: Towards an improved experimental methodology in machine learning. In Joost Kok, Jacek Koronacki, Ramon Lopez de Mantaras, Stan Matwin, Dunja Mladenic, and Andrzej Skowron, editors, *Knowledge Discovery in Databases: PKDD 2007*, volume 4702 of *Lecture Notes in Computer Science*, pages 6–17. Springer Berlin / Heidelberg, 2007. ISBN 978-3-540-74975-2. URL http://dx.doi.org/10.1007/978-3-540-74976-9_5.

[90] Melanie Hilario. e-lico annual report 2010. Technical report, Université de Geneve, 2010.

[91] Fernando Silva Parreiras, Steffen Staab, and Andreas Winter. On marrying ontological and metamodeling technical spaces. In *Proceedings of the the*

6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering, ESEC-FSE '07, pages 439–448, New York, NY, USA, 2007. ACM. ISBN 978-1-59593-811-4. doi: 10.1145/1287624.1287687. URL http://doi.acm.org/10.1145/1287624.1287687.

[92] Norbert Jankowski, Wlodzislaw Duch, and Krzysztof Grabczewski, editors. *Meta-Learning in Computational Intelligence*, volume 358 of *Studies in Computational Intelligence*. Springer, 2011. ISBN 978-3-642-20979-6.

[93] Cristóbal Romero, Juan Luis Olmo, and Sebastián Ventura. A meta-learning approach for recommending a subset of white-box classification algorithms for moodle datasets. In Sidney K. D'Mello, Rafael A. Calvo, and Andrew Olney, editors, *Proceedings of the 6th International Conference on Educational Data Mining, Memphis, Tennessee, USA, July 6-9, 2013*, pages 268–271. International Educational Data Mining Society, 2013. ISBN 978-0-9839525-2-7. URL http://www.educationaldatamining.org/EDM2013/papers/rn_paper_44.pdf.

[94] Marta E. Zorrilla and Diego García-Saiz. Meta-learning: Can it be suitable to automatise the kdd process for the educational domain? In *Rough Sets and Intelligent Systems Paradigms*, pages 285–292. Springer, 2014.

[95] Ricardo Vilalta and Youssef Drissi. A perspective view and survey of meta-learning. *Artificial Intelligence Review*, 18(2):77–95, 2002.

[96] Saddys Segrera, Joel Pinho, and María N. Moreno. Information-theoretic measures for meta-learning. In *Proceedings of the 3rd international workshop on Hybrid Artificial Intelligence Systems*, HAIS '08, pages 458–465, Berlin, Heidelberg, 2008. Springer-Verlag. ISBN 978-3-540-87655-7. doi: 10.1007/978-3-540-87656-4_57.

[97] Sarah Daniel Abdelmessih, Faisal Shafait, Matthias Reif, and Markus Goldstein. Landmarking for meta-learning using rapidminer. *German Research Center for Artificial Intelligence, Germany*, 2010.

[98] Melanie Hilario and Alexandros Kalousis. Building algorithm profiles for prior model selection in knowledge discovery systems. *Engineering Intelligent Systems*, 8:956–961, 2002.

[99] Yonghong Peng, Peter Flach, Carlos Soares, and Pavel Brazdil. Improved dataset characterisation for meta-learning. In *Discovery Science*, volume 2534 of *Lecture Notes in Computer Science*, pages 193–208. 2002. ISBN 978-3-540-00188-1.

[100] Pavel Brazdil, Christophe Giraud Carrier, Carlos Soares, and Ricardo Vilalta. *Metalearning: applications to data mining*. Springer, 2008.

[101] Shawkat Ali and Kate A Smith. On learning algorithm selection for classification. *Applied Soft Computing*, 6(2):119–138, 2006.

[102] Joao Gama and Pavel Brazdil. Characterization of classification algorithms. In *Progress in Artificial Intelligence*, pages 189–200. Springer, 1995.

[103] Kate A Smith, Frederick Woo, Vic Ciesielski, and Remzi Ibrahim. Matching data mining algorithm suitability to data characteristics using a self-organizing map. In *Hybrid information systems*, pages 169–179. Springer, 2002.

[104] Pavel B Brazdil, Carlos Soares, and Joaquim Pinto Da Costa. Ranking learning algorithms: Using ibl and meta-learning on accuracy and time results. *Machine Learning*, 50(3):251–277, 2003.

[105] Qinbao Song, Guangtao Wang, and Chao Wang. Automatic recommendation of classification algorithms based on data set characteristics. *Pattern recognition*, 45(7):2672–2689, 2012.

[106] Juan Trujillo and Sergio Luján-Mora. A UML based approach for modeling ETL processes in data warehouses. In Il-Yeol Song, Stephen W. Liddle, Tok Wang Ling, and Peter Scheuermann, editors, *Conceptual Modeling - ER 2003, 22nd International Conference on Conceptual Modeling, Chicago, IL, USA, October 13-16, 2003, Proceedings*, volume 2813 of *Lecture Notes in Computer Science*, pages 307–320. Springer, 2003. ISBN 3-540-20299-4. doi: 10.1007/978-3-540-39648-2_25. URL http://dx.doi.org/10.1007/978-3-540-39648-2_25.

[107] Lilia Muñoz, Jose-Norberto Mazón, Jesús Pardillo, and Juan Trujillo. Modelling ETL processes of data warehouses with UML activity diagrams. In Robert Meersman, Zahir Tari, and Pilar Herrero, editors, *On the Move to Meaningful Internet Systems: OTM 2008 Workshops, OTM Confederated*

International Workshops and Posters, ADI, AWeSoMe, COMBEK, EI2N, IWSSA, MONET, OnToContent + QSI, ORM, PerSys, RDDS, SEMELS, and SWWS 2008, Monterrey, Mexico, November 9-14, 2008. Proceedings, volume 5333 of *Lecture Notes in Computer Science*, pages 44–53. Springer, 2008. ISBN 978-3-540-88874-1. doi: 10.1007/978-3-540-88875-8_21. URL http://dx.doi.org/10.1007/978-3-540-88875-8_21.

[108] Vasiliki Tziovara, Panos Vassiliadis, and Alkis Simitsis. Deciding the physical implementation of ETL workflows. In Il-Yeol Song and Torben Bach Pedersen, editors, *DOLAP 2007, ACM 10th International Workshop on Data Warehousing and OLAP, Lisbon, Portugal, November 9, 2007, Proceedings*, pages 49–56. ACM, 2007. ISBN 978-1-59593-827-5. doi: 10.1145/1317331.1317341. URL http://doi.acm.org/10.1145/1317331.1317341.

[109] Sergio Luján-Mora, Juan Trujillo, and Il-Yeol Song. A UML profile for multidimensional modeling in data warehouses. *Data Knowl. Eng.*, 59(3): 725–769, 2006. doi: 10.1016/j.datak.2005.11.004. URL http://dx.doi.org/10.1016/j.datak.2005.11.004.

[110] Jose-Norberto Mazón, Jesús Pardillo, and Juan Trujillo. A model-driven goal-oriented requirement engineering approach for data warehouses. In Jean-Luc Hainaut, Elke A. Rundensteiner, Markus Kirchberg, Michela Bertolotto, Mathias Brochhausen, Yi-Ping Phoebe Chen, Samira Si-Said Cherfi, Martin Doerr, Hyoil Han, Sven Hartmann, Jeffrey Parsons, Geert Poels, Colette Rolland, Juan Trujillo, Eric S. K. Yu, and Esteban Zimányi, editors, *Advances in Conceptual Modeling - Foundations and Applications, ER 2007 Workshops CMLSA, FP-UML, ONISW, QoIS, RIGiM,SeCoGIS, Auckland, New Zealand, November 5-9, 2007, Proceedings*, volume 4802 of *Lecture Notes in Computer Science*, pages 255–264. Springer, 2007. ISBN 978-3-540-76291-1. doi: 10.1007/978-3-540-76292-8_31. URL http://dx.doi.org/10.1007/978-3-540-76292-8_31.

[111] Jose-Norberto Mazón and Juan Trujillo. A model driven modernization approach for automatically deriving multidimensional models in data warehouses. In Christine Parent, Klaus-Dieter Schewe, Veda C. Storey, and Bernhard Thalheim, editors, *Conceptual Modeling - ER 2007, 26th International Conference on Conceptual Modeling, Auckland, New Zealand, November 5-9, 2007, Proceedings*, volume 4801 of *Lecture Notes in*

Computer Science, pages 56–71. Springer, 2007. ISBN 978-3-540-75562-3. doi: 10.1007/978-3-540-75563-0_6. URL http://dx.doi.org/10.1007/978-3-540-75563-0_6.

[112] Sergio Luján-Mora, Panos Vassiliadis, and Juan Trujillo. Data mapping diagrams for data warehouse design with UML. In Paolo Atzeni, Wesley W. Chu, Hongjun Lu, Shuigeng Zhou, and Tok Wang Ling, editors, *Conceptual Modeling - ER 2004, 23rd International Conference on Conceptual Modeling, Shanghai, China, November 2004, Proceedings*, volume 3288 of *Lecture Notes in Computer Science*, pages 191–204. Springer, 2004. ISBN 3-540-23723-2. doi: 10.1007/978-3-540-30464-7_16. URL http://dx.doi.org/10.1007/978-3-540-30464-7_16.

[113] Sergio Luján-Mora, Juan Trujillo, and Il-Yeol Song. Extending the UML for multidimensional modeling. In Jean-Marc Jézéquel, Heinrich Hußmann, and Stephen Cook, editors, *UML 2002 - The Unified Modeling Language, 5th International Conference, Dresden, Germany, September 30 - October 4, 2002, Proceedings*, volume 2460 of *Lecture Notes in Computer Science*, pages 290–304. Springer, 2002. ISBN 3-540-44254-5. doi: 10.1007/3-540-45800-X_23. URL http://dx.doi.org/10.1007/3-540-45800-X_23.

[114] Sergio Luján-Mora, Juan Trujillo, and Il-Yeol Song. Multidimensional modeling with UML package diagrams. In Stefano Spaccapietra, Salvatore T. March, and Yahiko Kambayashi, editors, *Conceptual Modeling - ER 2002, 21st International Conference on Conceptual Modeling, Tampere, Finland, October 7-11, 2002, Proceedings*, volume 2503 of *Lecture Notes in Computer Science*, pages 199–213. Springer, 2002. ISBN 3-540-44277-4. doi: 10.1007/3-540-45816-6_24. URL http://dx.doi.org/10.1007/3-540-45816-6_24.

[115] José Jacobo Zubcoff and Juan Trujillo. Extending the UML for designing association rule mining models for data warehouses. In A Min Tjoa and Juan Trujillo, editors, *Data Warehousing and Knowledge Discovery, 7th International Conference, DaWaK 2005, Copenhagen, Denmark, August 22-26, 2005, Proceedings*, volume 3589 of *Lecture Notes in Computer Science*, pages 11–21. Springer, 2005. ISBN 3-540-28558-X. doi: 10.1007/11546849_2. URL http://dx.doi.org/10.1007/11546849_2.

[116] José Jacobo Zubcoff and Juan Trujillo. Conceptual modeling for classification mining in data warehouses. In A Min Tjoa and Juan Trujillo, editors, *Data Warehousing and Knowledge Discovery, 8th International Conference, DaWaK 2006, Krakow, Poland, September 4-8, 2006, Proceedings.*, volume 4081 of *Lecture Notes in Computer Science*, pages 566–575. Springer, 2006. ISBN 3-540-37736-0. doi: 10.1007/11823728_54. URL http://dx.doi.org/10.1007/11823728_54.

[117] José Jacobo Zubcoff, Jesús Pardillo, and Juan Trujillo. Integrating clustering data mining into the multidimensional modeling of data warehouses with UML profiles. In Il Yeal Song, Johann Eder, and Tho Manh Nguyen, editors, *Data Warehousing and Knowledge Discovery, 9th International Conference, DaWaK 2007, Regensburg, Germany, September 3-7, 2007, Proceedings*, volume 4654 of *Lecture Notes in Computer Science*, pages 199–208. Springer, 2007. ISBN 978-3-540-74552-5. doi: 10.1007/978-3-540-74553-2_18. URL http://dx.doi.org/10.1007/978-3-540-74553-2_18.

[118] José Jacobo Zubcoff and Juan Trujillo. A UML 2.0 profile to design association rule mining models in the multidimensional conceptual modeling of data warehouses. *Data Knowl. Eng.*, 63(1):44–62, 2007. doi: 10.1016/j.datak.2006.10.007. URL http://dx.doi.org/10.1016/j.datak.2006.10.007.

[119] Jesús Pardillo, José Jacobo Zubcoff, and Juan Trujillo. Un perfil UML para el análisis de series temporales con modelos conceptuales sobre almacenes de datos. In Maria Lencastre, João Falcão e Cunha, and Antonio Valecillo, editors, *Memorias de la XI Conferencia Iberoamericana de Software Engineering (CIbSE 2008), Recife, Pernambuco, Brasil, February 13-17, 2008*, pages 369–374, 2008.

[120] Pete Chapman, Julian Clinton, Randy Kerber, Thomas Khabaza, Thomas Reinartz, Colin Shearer, and Rudiger Wirth. Crisp-dm 1.0 step-by-step data mining guide. 2000.

[121] Óscar Marbán Gallego. *Modelo matemático paramétrico de estimación para proyectos de data mining.* PhD thesis, Informatica, 2003.

[122] Alex Guazzelli, Michael Zeller, Wen-Ching Lin, and Graham Williams. Pmml: An open standard for sharing models. *The R Journal*, 1(1):60–65, 2009.

[123] Stefano Rizzi, Elisa Bertino, Barbara Catania, Matteo Golfarelli, Maria Halkidi, Manolis Terrovitis, Panos Vassiliadis, Michalis Vazirgiannis, and Euripides Vrachnos. Towards a logical model for patterns. In *Conceptual Modeling-ER 2003*, pages 77–90. Springer, 2003.

[124] Dennis Wegener and Stefan Rüping. On integrating data mining into business processes. In *Business Information Systems*, pages 183–194. Springer, 2010.

[125] Oscar Marbán, Ernestina Menasalvas, and Covadonga Fernández-Baizán. A cost model to estimate the effort of data mining projects (dmcomo). *Information Systems*, 33(1):133–150, 2008.

[126] Oscar Marbán, Javier Segovia, Ernestina Menasalvas, and Covadonga Fernández-Baizán. Toward data mining engineering: A software engineering approach. *Information systems*, 34(1):87–107, 2009.

[127] William J. Frawley, Gregory Piatetsky-shapiro, and Christopher J. Matheus. Knowledge discovery in databases: an overview, 1992.

[128] Michael L. Brodie. Data quality in information systems. *Information & Management*, 3(6):245–258, 1980.

[129] Iso/iec 25012 - software engineering – software product quality requirements and evaluation (square) – data quality model. *English*, 2008, 2008.

[130] Mouzhi Ge and Markus Helfert. A review of information quality research - develop a research agenda. In Mary Ann Robbert, Robert O'Hare, M. Lynne Markus, and Barbara D. Klein, editors, *ICIQ*, pages 76–91. MIT, 2007.

[131] César Guerra-García, Ismael Caballero, and Mario Piattini. Capturing data quality requirements for web applications by means of dq_webre. In *Proceedings of the 2nd International Workshop on Business intelligencE and the WEB*, BEWEB '11, pages 28–35, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0610-2. doi: http://doi.acm.org/10.1145/1966883.1966892. URL http://doi.acm.org/10.1145/1966883.1966892.

[132] Donald Michie, David J Spiegelhalter, and Charles C Taylor. Machine learning, neural and statistical classification. 1994.

[133] Ricardo Vilalta and Youssef Drissi. A perspective view and survey of meta-learning. *Artificial Intelligence Review*, 18:77–95, 2002. ISSN 0269-2821. URL http://dx.doi.org/10.1023/A:1019956318069. 10.1023/A:1019956318069.

[134] Angélica Caro, Alejandra Fuentes, and M Antonieta Soto. Desarrollando sistemas de información centrados en la calidad de datos. *Ingeniare. Revista chilena de ingeniería*, 21(1):54–69, 2013.

[135] Wolfgang Lehner, Jens Albrecht, and Hartmut Wedekind. Normal forms for multidimensional databases. In *Scientific and Statistical Database Management, 1998. Proceedings. Tenth International Conference on*, pages 63–72. IEEE, 1998.

[136] Joseph L. Horner and Il-Yeol Song. A taxonomy of inaccurate summaries and their management in olap systems. *Conceptual Modeling–ER 2005*, pages 433–448, 2005.

[137] Jose-Norberto Mazón, Jens Lechtenbörger, and Juan Trujillo. A survey on summarizability issues in multidimensional modeling. *Data Knowl. Eng.*, 68 (12):1452–1469, 2009.

[138] Roberto Espinosa, José Jacobo Zubcoff Vallejo, Marta E. Zorrilla Pantaleón, José Norberto Mazón López, et al. Hacia la consideración de aspectos de calidad de datos en procesos de minería: el caso de las técnicas de clasificación. 2010.

[139] Ilkay Altintas, Chad Berkley, Efrat Jaeger, Matthew Jones, Bertram Ludascher, and Steve Mock. Kepler: an extensible system for design and execution of scientific workflows. In *Scientific and Statistical Database Management, 2004. Proceedings. 16th International Conference on*, pages 423–424. IEEE, 2004.

[140] A. Kalousis and M. Hilario. Model selection via meta-learning: a comparative study. In *Tools with Artificial Intelligence, 2000. ICTAI 2000. Proceedings. 12th IEEE International Conference on*, pages 406–413, 2000. doi: 10.1109/TAI.2000.889901.

[141] Ricardo Vilalta, Christophe G. Giraud-Carrier, Pavel Brazdil, and Carlos Soares. Using meta-learning to support data mining. *IJCSA*, 1(1):31–45, 2004.

[142] Xindong Wu, Vipin Kumar, J. Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J. McLachlan, Angus Ng, Bing Liu, Philip S. Yu, Zhi-Hua Zhou, Michael Steinbach, David J. Hand, and Dan Steinberg. Top 10 algorithms in data mining. *Knowl. Inf. Syst.*, 14(1):1–37, December 2007. ISSN 0219-1377. doi: 10.1007/s10115-007-0114-2. URL http://dx.doi.org/10.1007/s10115-007-0114-2.

[143] Cristóbal Romero and Sebastián Ventura. Educational Data Mining: A Review of the State-of-the-Art. *IEEE Tansactions on Systems, Man and Cybernetics, part C: Applications and Reviews*, 40(6):601–618, 2010.

[144] Marta E. Zorrilla and Diego García-Saiz. *Business Intelligence Applications and the Web: Models, Systems and Technologies*, chapter Mining Service to Assist Instructors involved in Virtual Education. Information Science Reference (IGI Global Publishers), September 2011. ISBN 978-1-61350-038-5.

[145] Gema María Ramírez Pacheco and Federico García Erviti. Comportamiento segmentado del mercado inmobiliario y definición de patrones. comportamiento segmentado del mercado inmobiliario y definición de patrones territoriales. aplicación a la valoración de áreas periurbanas. *Catastro*, (77): 23–41, 2013.

[146] Graham Upton and Ian Cook. *Understanding statistics*. Oxford University Press, 1996.

[147] Daniel Zwillinger and Stephen Kokoska. *CRC standard probability and statistics tables and formulae*. CRC Press, 2010.

[148] Arthur Asuncion and David Newman. Uci machine learning repository [http://www. ics. uci. edu/˜ mlearn/mlrepository. html]. irvine, ca: University of california. *School of Information and Computer Science*, 2007.

[149] Roberto Espinosa, Diego García-Saiz, Marta E Zorrilla, Jose Jacobo Zubcoff, and Jose-Norberto Mazón. Development of a knowledge base for enabling non-expert users to apply data mining algorithms. In *SIMPDA*, pages 46–61. Citeseer, 2013.

[150] Frédéric Jouault, Freddy Allilaire, Jean Bézivin, and Ivan Kurtev. Atl: A model transformation tool. *Science of Computer Programming*, 72(1–2):

31 – 39, 2008. ISSN 0167-6423. doi: http://dx.doi.org/10.1016/j.scico.2007.08.002. URL http://www.sciencedirect.com/science/article/pii/S0167642308000439. Special Issue on Second issue of experimental software and toolkits (EST).

[151] Guillaume Hillairet, Frédéric Bertrand, and Jean Y. Lafaye. Bridging EMF applications and RDF data sources. In *4th International Workshop on Semantic Web Enabled Software Engineering*, 2008. URL http://www.abdn.ac.uk/~{}r01srt7/swese2008/pdf/swese2008_submission_14.pdf.

[152] Roberto Espinosa, Larisa Garriga, José Jacobo Zubcoff Vallejo, and José Norberto Mazón López. Knowledge spring process - towards discovering and reusing knowledge within linked open data foundations. In *Proceedings of the 3rd International Conference on Data Management Technologies and Applications*, pages 291–296. INSTICC, 2014.