1. Introducción

Introducción

En este capítulo se introducen en primer lugar una serie de conceptos que aparecen a lo largo del proyecto. Seguidamente, se indica el hardware y software utilizado para su realización. A continuación, se exponen brevemente las fases de las que consta. Y, por último, se detalla la estructura de esta memoria.

1.1. Conceptos

La aplicación y algoritmos desarrollados han sido implementados usando el lenguaje de programación Ruby, y se hace uso del lenguaje de etiquetas XML para especificar la entrada de datos a los algoritmos de Data Mining en un proceso KDD. Por ello, en esta primera sección se introducen brevemente se introducen los tópicos mencionados.

1.1.1. Knowledge Discovery in Databases

El proceso KDD (*Knowledge Discovery in Databases*), de forma general, consiste en la recuperación de conocimiento no evidente a partir de datos en bruto. A partir de una o varias fuentes de datos, se desea obtener información. Estos datos hay que limpiarlos, e integrarlos en caso de que procedan de varias fuentes. Seguidamente los datos se seleccionan y se preparan para ser procesados por un algoritmo de Data Mining. El algoritmo de Data Mining realiza sobre los datos las operaciones necesarias para rescatar información no evidente, si la hubiera, obteniendo patrones de comportamiento. Finalmente, esos patrones se evalúan conforme a una medida de calidad, dependiente del algoritmo de Data Mining, y se presentan al usuario.

El concepto es interesante desde el momento en el que el volumen de datos a tratar no es abarcable por un humano y se requiere el uso de computación para examinar los datos de forma ágil y, generalmente, detectar patrones no evidentes, útiles y no redundantes.

En este proyecto se realiza desde la adaptación de los datos al algoritmo hasta la presentación al usuario.

1.1.2. Data Mining

Data Mining forma parte del proceso total de descubrimiento de conocimiento en bases de datos (KDD o *Knowledge Discovery in Databases*). A pesar de ello, en la literatura se tiende a usar como sinónimos Data Mining y KDD.

La minería de datos es la parte del proceso encargada de aplicar los diferentes procedimientos y algoritmos capaces de obtener el conocimiento no evidente a partir de datos preparados para su manipulación, obteniendo, generalmente, patrones. La calidad de estos patrones se puede medir, dependiendo de los algoritmos utilizados.

Los algoritmos de Data Mining pueden agruparse en procedimientos. Un procedimiento es un "problema tipo" que puede ser resuelto usando algoritmos de minería de datos. Como ejemplo de procedimientos se pueden encontrar el clustering, que agrupa registros similares; la clasificación, que asigna registros a clases predefinidas; o las reglas de asociación, que encuentra patrones internos a los registros.

En este proyecto se implementan algoritmos de Data Mining que realizan diferentes procedimientos.

1.1.3. CRM

Un CRM (*Customer Relationship Management*) es una aplicación gestiona la relación de la empresa con sus clientes, recopila los contactos realizados con cada cliente, mantiene una lista de productos ofrecidos y anota el resultado de cada transacción, ya sea una venta conseguida o una venta no realizada pero para la cual el cliente mostró algún grado de interés.

El uso de los CRM se hace fundamental si se quiere tratar a los clientes de forma individualizada. Se pueden tener anotados comportamientos y gustos de cada cliente, y ofrecerles ofertas concretas adaptadas a cada uno de ellos. Es importante saber si un cliente le gusta que se acuerden de él, o si por el contrario le molesta; o si es receptivo a la hora de ofrecerle un producto novedoso, o en cambio prefiere ceñirse a productos comunes.

El origen de la información a analizar en este proyecto será la base de datos de un CRM no comercial.

1.1.4. XML

XML es un lenguaje de etiquetas desarrollado por el W3C a partir de SGML. XML se propone como un estándar para el intercambio de información estructurada entre aplicaciones. Además, esa información habitualmente es autodescrita, con lo cual puede interpretarla fácilmente un humano que lea el documento XML.

La estructura de la información en un documento XML es en forma de árbol, quedando claramente delimitado de qué se compone cada parte entre etiquetas. Una característica notable de este lenguaje es que se puede conseguir saber rápidamente si un documento es XML comprobando si está bien formado. Esto consiste en comprobar que el documento tiene solo un elemento raíz y que cada etiqueta de cierre solo ocurre si todos los elementos interiores a cada parte también están cerrados.

Los datos de entrada al algoritmo implementado vendrán dados en un documento XML.

1.1.5. Ruby

Ruby es un lenguaje de programación interpretado y orientado a objetos, inspirado en Python y Perl. Es interpretado de una sola pasada y su implementación oficial se distribuye bajo una licencia de software libre.

Uno de los puntos fuertes de este lenguaje es que su curva de aprendizaje es muy abrupta para cualquier programador experimentado, consiguiendo programar de forma efectiva en un corto espacio de tiempo.

Ruby consigue gran fama cuando aparece la implementación del framework Rails.

Es un requisito de este proyecto implementar los algoritmos de minería de datos en este lenguaje de programación.

1.1.6. Ruby on Rails (RoR)

Ruby on Rails es un framework de aplicaciones web escrito en Ruby, siguiendo el paradigma de la arquitectura Modelo-Vista-Controlador. Este framework está muy extendido gracias a la rapidez con la que permite construir aplicaciones web, gracias a su autogeneración del código común y a que automatiza la gestión de la base de datos.

Es un requisito de este proyecto implementar la interfaz de usuario utilizando Ruby on Rails.

1.1.7. Arquitectura Modelo-Vista-Controlador

Esta arquitectura está muy extendida para el desarrollo de aplicaciones web amplias. Es un patrón abstracto que separa los datos, la interfaz de usuario y la lógica. Consiste en dividir la aplicación en tres partes:

- Modelo: Es la representación de la información con que opera el sistema. Esta parte se encarga del modelado de datos y gestión de la base de datos.
- Vista: Se encarga de presentar el modelo en una interfaz de usuario.
- Controlador: Es la parte de la arquitectura que gestiona los eventos, y los procesa según la lógica implementada, invocando las peticiones al modelo y, en su caso, a las vistas.

Ruby on Rails sigue la arquitectura Modelo-Vista-Controlador.

La aplicación web resultante de este proyecto sigue la arquitectura Modelo-Vista-Controlador, al estar escrita usando Ruby on Rails.

1.1.8. RubyGems

RubyGems es un gestor de paquetes para Ruby. RubyGems proporciona:

- Un formato estándar y autocontenido para distribuir librería (llamadas gemas).
- Una herramienta para gestionar la instalación de las gemas.
- Y un servidor para la distribución de las mismas.

Los algoritmos de minería de datos implementados durante el proyecto se empaquetarán en una "gema" utilizable por este gestor.

1.1.9. Subversion

Subversion es un sistema de control de versiones de código. Usando SVN es posible almacenar las diferentes versiones de un código en su desarrollo y así poder recuperar el estado del mismo en cualquier momento anterior. Además, es también una herramienta útil para compartir código y para el desarrollo simultáneo por más de un programador.

Durante el desarrollo de la aplicación objetivo del proyecto, el código generado se almacena en un servidor de SVN.

1.2. Hardware

En esta sección se describe el hardware empleado durante el desarrollo. La aplicación se ha desarrollado en un equipo portátil con las siguientes características:

- Procesador Intel Core i7
- 6 GB de memoria RAM

Además, se hace uso de un servicio Subversion (SVN) alojado en un servidor de características desconocidas, proporcionado por la empresa solicitante del proyecto.

1.3. Software

En esta sección se describe el software empleado durante el desarrollo del proyecto.

1.3.1. Sistemas operativos

Las librerías que implementan algoritmos de minería de datos se han desarrollado en un entorno Windows, concretamente en un sistema operativo Windows 7 Home Premium SP 1.

La aplicación web de minería de datos se desarrolla en un entorno Linux virtualizado, concretamente en un sistema operativo Ubuntu 10.04.4 LTS.

1.3.2. Virtualización

Para la virtualización del sistema operativo Ubuntu expuesto anteriormente, se ha utilizado la aplicación VMware Player 4.0.2, sobre el entorno Windows.

1.3.3. Entornos de desarrollo

En el entorno Windows, se ha utilizado el software de desarrollo RubyMine 4.0.2, que proporciona facilidades para el desarrollo en el lenguaje de programación Ruby. Haciendo uso de este entorno se ha realizado la implementación de los algoritmos de minería de datos requeridos en el proyecto.

En el entorno Linux, se ha utilizado el programada gedit. En este entorno se ha realizado la implementación de la aplicación, empleando la arquitectura modelo-vista-controlador mediante el uso del framework Ruby on Rails.

1.3.4. Frameworks

Para el desarrollo de la aplicación se hace uso del framework Ruby on Rails, que implementa la arquitectura modelo-vista-controlador. El uso de framework reduce considerablemente el tiempo de desarrollo de la aplicación.

1.3.5. Gestor de versiones

Se ha hecho uso de la aplicación RabbitVCS para acceder al servidor subversion en el que alojar el código de la aplicación.

1.4. Fases de diseño y desarrollo

En esta sección se enumeran las fases por las que ha pasado el diseño y el desarrollo a lo largo de la duración del proyecto, y se hace una descripción breve de las distintas tareas desarrolladas.

1.4.1. Primera fase: Prototipo

En este apartado se enumeran y describen brevemente las tareas desarrolladas en la primera fase del proyecto.

- *Estudio metódico de los algoritmos de Data Mining*

  Se realiza en primer lugar un estudio de los algoritmos de minería de datos aplicables al comercio electrónico. El resultado es la asimilación de los conceptos generales de la minería de datos y un estado del arte de los principales algoritmos existentes. A partir de esta tarea se puede realizar una elección efectiva de los algoritmos a utilizar.

- *Selección de algoritmos*

  Partiendo de la tarea anterior, se seleccionan razonadamente los algoritmos cuyo uso se adapta mejor a una aplicación de minería de datos para comercio electrónico. Los algoritmos seleccionados son kmedoids aplicado a procedimientos de clustering, y Apriori aplicado a procedimientos de descubrimiento de reglas de asociación.

- *Diseño de un prototipo*

  Se diseña un flujo de datos a partir del concepto de “procedimiento” de minería de datos y, a partir de él, se diseña un prototipo de aplicación indicando los algoritmos de minerías de datos usados.

- *Implementación del algoritmo kmedoids*

  Con el objetivo de mostrar que es factible la implementación de algoritmos de minería de datos en tiempos razonables, se implementa el algoritmo de clustering kmedoids.

- *Construcción de una base de datos de prueba*

  Además del algoritmo anterior, se construye una base de datos de prueba con datos controlados para mostrar el correcto funcionamiento del algoritmo.

1.4.2. Segunda fase: Relación con el CRM

En este apartado se enumeran y describen brevemente las tareas desarrolladas en la segunda fase del proyecto.

- *Adaptación del prototipo*

  Se mejora el diseño del prototipo de la primera fase para adaptarse al CRM de la empresa. En el nuevo diseño se tiene en cuenta la adición de clientes y ofertas nuevos al CRM a la vez que a los clústeres contemplados en el primer diseño.

- *Diseño de los campos de la base de datos*

  Se diseñan los campos que debe recoger la base de datos y se identifican los que realmente serán útiles a la hora de realizar el tratamiento de minería de datos.

- *Adaptación al algoritmo elegido para los casos de clientes y ofertas*

  En primer lugar se estudia cómo manipular los datos de clientes y ofertas de forma previa a al procesamiento en el algoritmo de minería de datos. A partir de lo anterior se programa un script que prepare los datos para que puedan servir como entrada del algoritmo.

- *Almacenamiento de la salida del algoritmo de clustering*

  Se diseña la estructura de la base de datos que recoge la salida proporcionada por el algoritmo de clustering.

- ***Empaquetamiento del algoritmo de minería de datos en "gem"***

  El código de la implementación del algoritmo kmedoids en Ruby se empaqueta en una gema, para facilitar su instalación y para separar su mantenimiento del que necesite el código fuente que haga uso del algoritmo.

- ***Implementación de la interfaz de acceso a clientes, ofertas y operaciones (CRM)***

  Como complemento al diseño de los campos de la base de datos y toma de contacto con el framework Ruby on Rails, se realiza la implementación de interfaces de tipo CRUD para la gestión de la base de datos a partir de una aplicación web. Por operaciones se entiende a la relación entre un cliente y una oferta.

1.4.3. Fase final: Implementación final

En este apartado se enumeran y describen brevemente las tareas desarrolladas en la tercera y última fase del proyecto.

- ***Implementación del algoritmo de reglas***

  Se implementa el algoritmo de reglas Apriori, que será finalmente el algoritmo utilizado en la aplicación de minería de datos.

- ***Diseño del modelo de datos final de la aplicación***

  Se diseña el modelo de datos final de la aplicación, que estará compuesto por tres secciones diferenciadas:
  - Sección de parametrización: Donde se incluye información de parametrización. Se hace uso de ella antes de la ejecución del algoritmo de reglas.
  - Sección de transacciones: Donde se almacenan las transacciones en el formato adecuado para la ejecución del algoritmo de reglas.
  - Sección de reglas: Donde se almacena la salida del algoritmo de reglas.

- ***Implementación del modelo final***

  Se implementa la aplicación final, que proporciona una interfaz amigable para ser utilizada por un operador. La interfaz permite cargar datos a la aplicación a partir de un archivo XML, configurar la información de parametrización, ejecutar el algoritmo con los parámetros de soporte y confianza deseados, y mostrar los resultados.

1.5. Estructura del documento

La presente memoria se estructura en cinco capítulos y un anexo, además de las secciones de objetivos e índices al inicio, y glosarios y referencias al final.

- *Objetivos e índices*
- *1. Introducción*: Expone los tópicos necesarios para la comprensión del proyecto y describe brevemente las fases de diseño y desarrollo del mismo.
- *2. Fundamentos de minería de datos*: Profundiza en los conceptos de minería de datos y expone un estado del arte de sus algoritmos.
- *3. Diseño y desarrollo*: Describe en profundidad las tareas desarrolladas en cada fase del proyecto.
- *4. Descripción de la interfaz realizada*: Describe los requisitos, forma de uso y pantallas de la interfaz realizada para manejar el algoritmo.
- *5. Conclusiones y líneas futuras*: Expone conclusiones y posibles líneas de trabajo futuras.
- *Glosario y referencias*
- *Anexo: Manual de usuario*

Resumen del capítulo

En este capítulo se han introducido conceptos que aparecen a lo largo del proyecto, se ha expuesto el hardware y software utilizado para su realización, se han descrito brevemente las fases de las que consta y se ha detallado la estructura de esta memoria. En el siguiente capítulo se introducen más en profundidad los conceptos de Data Mining y proceso KDD, se expone una clasificación de los distintos procedimientos de minería de datos, y se presenta el estado del arte de los algoritmos de Data Mining.