Trabajo Fin de Máster

MODELOS PREDICTIVOS DEL *CHURN* – ABANDONO DE CLIENTES – PARA OPERADORES DE TELECOMUNICACIONES

David Lozano Núñez

Máster en Técnicas Estadísticas
Curso 2014-2015

Autorización de entrega

D. Javier Roca Pardiñas y D. Antonio Vidal Vidal

Certifican:

Que el proyecto titulado “Modelos predictivos del *churn* – abandono de clientes – para operadores de telecomunicaciones" ha sido realizado por D. David Lozano Núñez, con D.N.I. 53456796-N, bajo la dirección de D. Javier Roca Pardiñas y D. Antonio Vidal Vidal.

Esta memoria constituye la documentación que, con nuestra autorización, entrega dicho alumno como Trabajo Fin de Máster.

Firmado:

D. Javier Roca Pardiñas D. Antonio Vidal Vidal

Vigo, a 8 de Julio de 2015.

Agradecimientos

Tras la realización del presente Trabajo Fin de Máster, quería agradecer a mis tutores Antonio Vidal Vidal y Javier Roca Pardiñas toda la ayuda recibida.

También me gustaría hacer una mención especial a mi familia y amigos.

Muchas gracias a todos.

Resumen

El objetivo de las prácticas se enmarca en los modelos predictivos del *churn*, tasa de abandono, para los operadores de telecomunicaciones.

Reducir el *churn* influye enormemente en los resultados del operador. Con los datos demográficos, patrones de consumo, facturación, historial de incidencias, patrones de navegación web, etc., de los que dispone el operador, se pueden generar modelos para intentar predecir cuales son los clientes en riesgo de baja, de forma que se puedan focalizar acciones de retención y fidelización sobre ellos.

Para ello, será necesario trabajar con los datos proporcionados y prototipar de forma ágil las soluciones de minería de datos en diferentes herramientas, al mismo tiempo que se optimizan y documentan los resultados obtenidos.

Índice

Índice de figuras

Índice de tablas

1. Introducción

Entre las dos modalidades posibles para la realización del trabajo fin de máster, según el último reglamento publicado, he escogido la segunda. Esta modalidad consiste en la realización de un trabajo dentro de una empresa y tiene como objetivo analizar y estudiar problemas del área de la estadística o la investigación operativa en los que estén interesados.

El trabajo que presento a continuación se ha realizado para el área de Investigación y Desarrollo de la empresa Optare Solutions durante los meses de Junio y Julio de 2014, de lunes a viernes de 9:00 a 15:00, completando un total de 240 horas. El trabajo realizado ha sido dirigido por Antonio Vidal Vidal, responsable de Innovación en Optare Solutions, y avalado por el profesor Javier Roca Pardiñas del Máster en Técnicas Estadísticas.

Desde Septiembre he participado en el proyecto BDA4T, "*Big Data Analytics for Telecoms*", como personal colaborador en investigación, y desde Abril formo parte de la plantilla de Optare Solutions, como analista de datos, dentro del proyecto de innovación ALCHEMY. Parte del trabajo realizado durante este tiempo también se ve reflejado en el presente documento.

1.1. Descripción de la empresa

Optare Solutions S.L.[1] es una consultora tecnológica especializada en la implantación, mantenimiento, desarrollo e integración de sistemas BSS (*Business Support Systems*), en la realización de OSS (*Operations Support Systems*) y en la prestación de servicios profesionales de consultoría en el ámbito de los operadores de Telecomunicaciones, que está situada en el Parque Tecnológico y Logístico de Valladares, Vigo.

Nació en el año 2002 con el objetivo de proporcionar consultoría técnica para la provisión de servicios complejos en una operadora emergente del mercado español, ayudando a que sus sistemas soportasen un gran crecimiento de su volumen de negocio, más de 10 veces en menos de 3 años.

Durante estos años otras operadoras han requerido de sus servicios y han continuado ayudando a sus clientes a crecer, mejorar su eficiencia, reducir costes, y convertirse en empresas más ágiles.

Dentro del área de OSS, Optare Solutions está especializada en los procesos de provisión y aseguramiento de servicios y en los sistemas que interactúan con estos, principalmente los relacionados con la gestión de procesos de comunicación e intercambio de información entre operadoras.

Desde el año 2011, Optare Solutions cuenta además con una división de consultoría internacional proporcionando formación y servicios de los productos de *Oracle Communications* específicos para el sector *telecom*.

La experiencia, el conocimiento y los resultados obtenidos en los más de 12 años de existencia de Optare Solutions les han hecho merecedores del reconocimiento internacional como "*Cool Vendor in Telecom Operations Management System 2010*" por Gartner Inc, firma americana líder en análisis del mercado TIC (Tecnologías de la Información y la Comunicación).

[1] http://optaresolutions.com/es/

2. ¿Qué es el *churn*?

Actualmente, el anglicismo *churn* se utiliza para describir la tasa de abandono de los clientes, es decir, aquellos clientes que dejan la compañía o al proveedor de un servicio durante un período de tiempo determinado. Describe la infidelidad o falta de lealtad de los clientes, se cuantifica en tanto por ciento y se calcula como sigue:

$$\text{Churn rate} = \frac{\text{Número de clientes perdidos en un periodo determinado}}{\text{Número de clientes al comienzo de ese periodo}} \cdot 100$$

Los periodos más utilizados para el cálculo de la tasa de abandonos son: mensual, trimestral y anual. El número de clientes perdidos en un periodo determinado se obtiene sumando la cartera activa al comienzo de un periodo y los clientes dados de alta en ese periodo y posteriormente restando la cartera activa al finalizar dicho periodo.

Generalmente se hace una distinción entre *churn* voluntario y *churn* involuntario. El *churn* voluntario se produce debido a una decisión por parte del cliente de cambiarse a otra compañía y el *churn* involuntario se produce debido a la reubicación del cliente en otra zona geográfica, la caída en morosidad o impago y, en el extremo, la muerte.

En la mayoría de las aplicaciones de inteligencia de negocios el *churn* involuntario se excluye de los modelos predictivos, concentrándose principalmente en el *churn* voluntario, ya que éste, normalmente, se produce debido a factores de relación empresa-cliente y es en el que nos centraremos y analizaremos.

El siguiente esquema muestra la clasificación de las principales razones que los clientes tienen para abandonar la compañía. Este esquema proporciona una referencia útil para entender las razones de pérdida de clientes y cómo se relacionan entre sí [1].

- Voluntario
  - Deliberado
    - Tecnología
    - Rentabilidad
    - Calidad
    - Social / Psicológico
    - Conveniencia
  - Fortuito
    - Financiero
    - Ubicación
    - Gran cambio de vida
- Involuntario
  - Fraude
  - Impago
  - Poco uso

Dada la importancia que tiene la pérdida de clientes, el objetivo será detectar con antelación qué clientes están a punto de abandonar el servicio con el fin de retenerlos a tiempo, reduciendo así los costes y riesgos y ganar en eficiencia y competitividad, interviniendo de este modo en el ciclo de vida del cliente.

En la gestión de las relaciones con los clientes (*Customer Relationship Management*), el ciclo de vida del cliente es un término usado para describir la secuencia de pasos que un cliente atraviesa a la hora de considerar, comprar, utilizar y mantener la lealtad a un producto o servicio.

Los analistas de marketing Jim Sterne y Matt Cutler han desarrollado una matriz que divide el ciclo de vida del cliente en cinco pasos distintos: contacto, adquisición, conversión, retención y lealtad [2].

En términos sencillos, esto significa conseguir la atención de un cliente potencial, enseñándoles lo que tiene que ofrecer, convirtiéndolos en un cliente que paga y luego mantenerlos como clientes leales cuya satisfacción con el producto o servicio insta a otros clientes a unirse al ciclo.

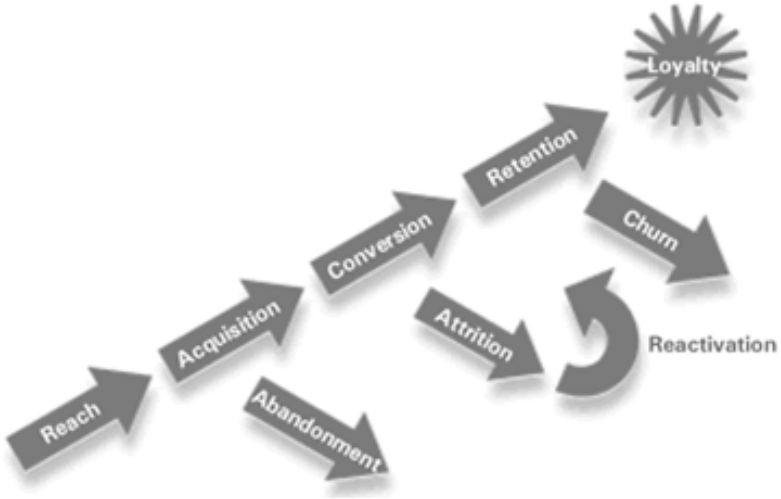


Figura 1: Ciclo de vida del cliente.

2.1. *Churn* en los operadores de telecomunicaciones

A partir del Informe Económico de las Telecomunicaciones y del Sector Audiovisual 2014, se observa que el *churn* medio en España en 2013 entre los operadores móviles es del 34’2% anual, ver Figura 2, lo que significa que cualquier operador móvil en España perdería todos sus clientes en menos de tres años si no fuera capaz de captar nuevos clientes [3].

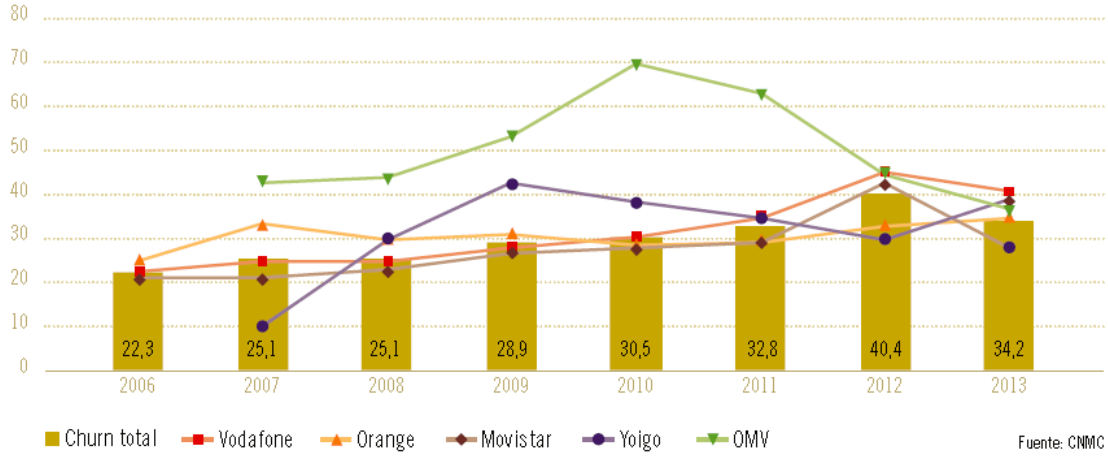


Figura 2: Índice de rotación de clientes.

Cada unidad, en tanto por ciento, que un operador consigue reducir en su tasa de *churn*, influye enormemente en sus resultados. Para ello, un operador puede establecer políticas de fidelización a todos los usuarios que considere en riesgo de baja, cuyo coste puede ser inasumible, o esperar hasta que un cliente solicite la portabilidad a otro operador y usar entonces políticas de retención, cuyos costes serían todavía más costosos al desarrollarse de manera individual y que, mientras el fenómeno de la portabilidad no estaba muy extendido, eran asumibles. En la actualidad se están batiendo récords de portabilidad en España: "Las portabilidades efectivas alcanzaron la cifra de 6,8 millones en 2013" [3].

En la situación de crisis actual, los operadores de telecomunicaciones sufren un ajuste en sus ingresos propiciado por el descenso del consumo de sus clientes, a pesar de que las comunicaciones son un bien esencial en la sociedad moderna. Tomando como referencia los datos anuales de la Comisión Nacional de los Mercados y la Competencia del año 2013, se observa que desde el comienzo de la crisis, en el año 2008, los ingresos totales de los operadores en España han descendido en un 28'8% [4].

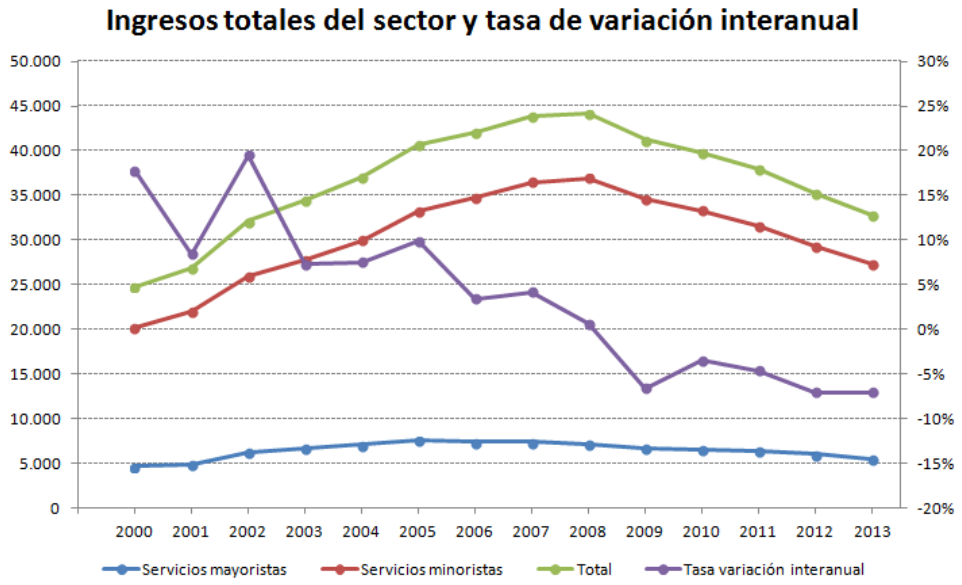


Figura 3: Ingresos totales del sector y tasa de variación interanual.

Una ayuda importante para los operadores sería el estudio de su base de clientes mediante técnicas de analítica avanzada, para proporcionarles información sobre sus clientes con mayor probabilidad de baja y mayor valor, con el objetivo de realizar acciones de fidelización de forma prioritaria sobre ellos.

3. Metodología

El presente trabajo fin de máster está englobado dentro del proyecto BDA4T (*Big Data Analytics for Telecoms*), que es un proyecto de investigación y desarrollo en Cooperación Nacional, financiado por el Centro para el Desarrollo Tecnológico Industrial (CDTI) y que cuenta con el apoyo de la Universidad de Vigo como organismo de investigación.

El objetivo de este proyecto es el desarrollo de un sistema de análisis predictivo de bajas en los operadores de telecomunicaciones, con datos proporcionados por sus sistemas de gestión de red, analizando la influencia de cada uno de los datos disponibles mediante modelos de análisis predictivo.

3.1. Modelo CRISP-DM

Como este proyecto se basa en la gestión de datos, su análisis y la conversión de los resultados en acciones automatizadas orientadas a la mejora de los resultados de los operadores, se usará como referencia el modelo CRISP-DM (Cross *Industry Standard Process for Data Mining*), definido en el año 1997, por una agrupación de empresas europeas del sector TIC, como una metodología y guía de buenas prácticas para la gestión de los datos de cara a su procesamiento y análisis.

CRISP-DM es un modelo de proceso de minería de datos, que describe los enfoques comunes que utilizan los expertos en esta materia tal y como demuestran las encuestas realizadas a lo largo del tiempo por diferente organismos. Por ejemplo, la encuesta *"What main methodology are you using for your analytics, data mining, or data science projects?"* realizada por KDnuggets, refleja la amplia aceptación que tiene el modelo en el mercado [5].

En el siguiente gráfico se muestran los resultados obtenidos tras la realización de 200 encuetas en 2014 frente a los resultados de 2007.

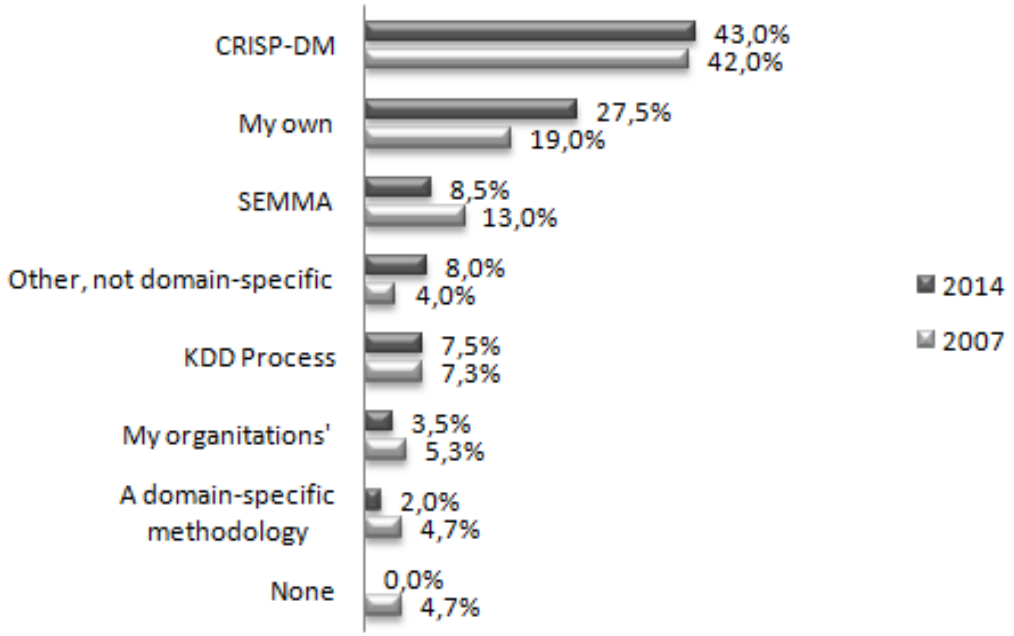


Figura 4: Metodología utilizada.

CRISP-DM divide el proceso de minería de datos en seis fases principales que se enlazan entre sí, como se muestran en la siguiente figura y que detallaremos a continuación.

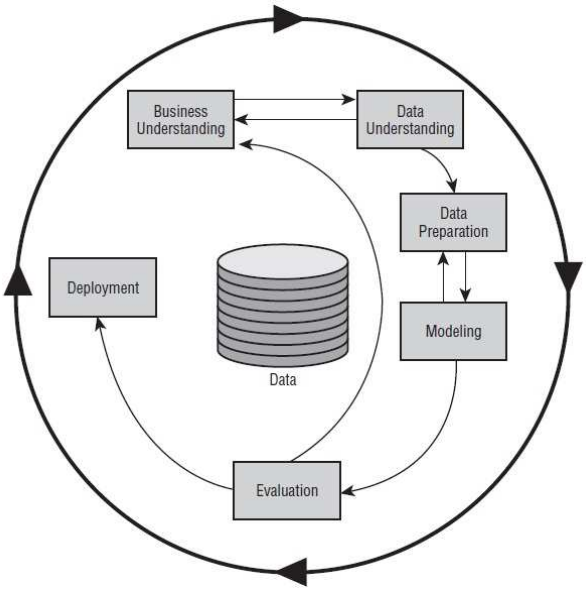


Figura 5: Diagrama de proceso que muestra la relación entre las diferentes fases.

Las flechas en el diagrama indican las dependencias más importantes y frecuentes entre fases. El círculo exterior en el diagrama simboliza la naturaleza cíclica de la minería de datos en sí, ya que un proceso de minería de datos continúa después del despliegue de una solución porque las lecciones aprendidas durante el proceso pueden provocar nuevas preguntas de negocio, a menudo más centradas. Posteriores procesos de minería de datos se beneficiarán de la experiencia de los anteriores [6, 7].

3.1.1. Comprensión del negocio (*Business Understanding*)

Esta fase inicial se enfoca en la comprensión de los objetivos del proyecto y exigencias desde una perspectiva de negocio, para convertir el conocimiento de los datos en la definición de un problema de minería de datos y en un plan preliminar diseñado para alcanzar los objetivos.

3.1.2. Comprensión de los datos (*Data Understanding*)

La fase de comprensión de datos comienza con la recolección de datos inicial y continúa con las actividades que permiten familiarizarse con los datos, identificar problemas de calidad de datos, obtener los primeros análisis y descubrir temas interesantes para formular hipótesis en cuanto a la información oculta.

3.1.3. Preparación de los datos (*Data Preparation*)

La fase de preparación de datos engloba todas las actividades necesarias para construir el conjunto de datos final, que será usado en la fase de modelado a partir de los datos iniciales. Estas tareas de preparación de datos van a ser ejecutadas repetidas veces y no pueden realizarse en cualquier orden. En general incluyen la selección y transformación de tablas, registros y atributos y limpieza de datos.

3.1.4. Modelado (*Modeling*)

En esta fase se seleccionan varias técnicas de modelado para aplicarlas a los datos disponibles y sus parámetros son calibrados para obtener resultados óptimos. Típicamente hay varias técnicas para el mismo tipo de problema de minería de datos que pueden ser aplicadas. Algunas técnicas tienen requerimientos específicos sobre la forma de los datos, por lo que es necesario volver a la fase de preparación, de ahí el carácter cíclico del proceso CRISP-DM.

3.1.5. Evaluación (*Evaluation*)

En esta etapa del proyecto, una vez construidos los modelos, se evalúan los resultados obtenidos mediante diferentes técnicas. Es importante evaluar el modelo a fondo y revisar los pasos ejecutados para crearlo antes de proceder al despliegue final del modelo, comparando los resultados con los objetivos de negocio.

Un objetivo clave es determinar si hay alguna cuestión importante de negocio que no ha sido suficientemente considerada para modificar los modelos creados en consecuencia. Al final de esta fase se obtendrá la decisión de uso de los resultados de minería de datos.

3.1.6. Despliegue (*Deployment*)

La creación del modelo no es el final del proyecto ya que el conocimiento obtenido tendrá que ser organizado y presentado de forma que el cliente pueda usarlo. Dependiendo de los requerimientos, la fase de desarrollo puede ser tan simple como la generación de un informe o tan compleja como la realización repetida de un proceso cruzado de minería de datos.

3.1.7. Tareas del modelo CRISP-DM

Además de las fases presentadas anteriormente, dentro de cada fase del modelo, es necesario realizar las tareas que se muestran a continuación, para conseguir los resultados buscados, y que serán acometidas en diferente grado dentro de este proyecto.

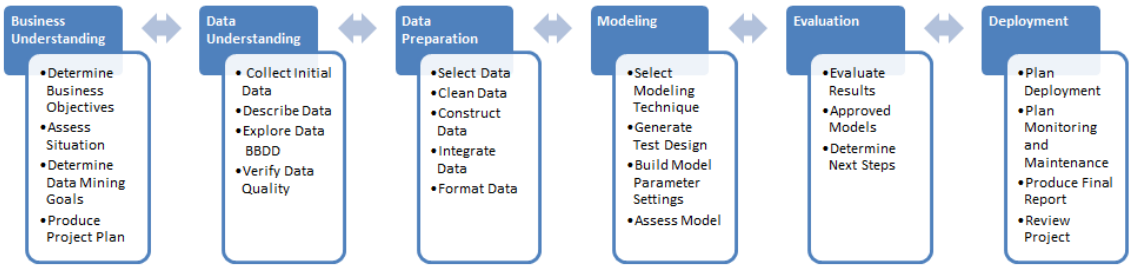


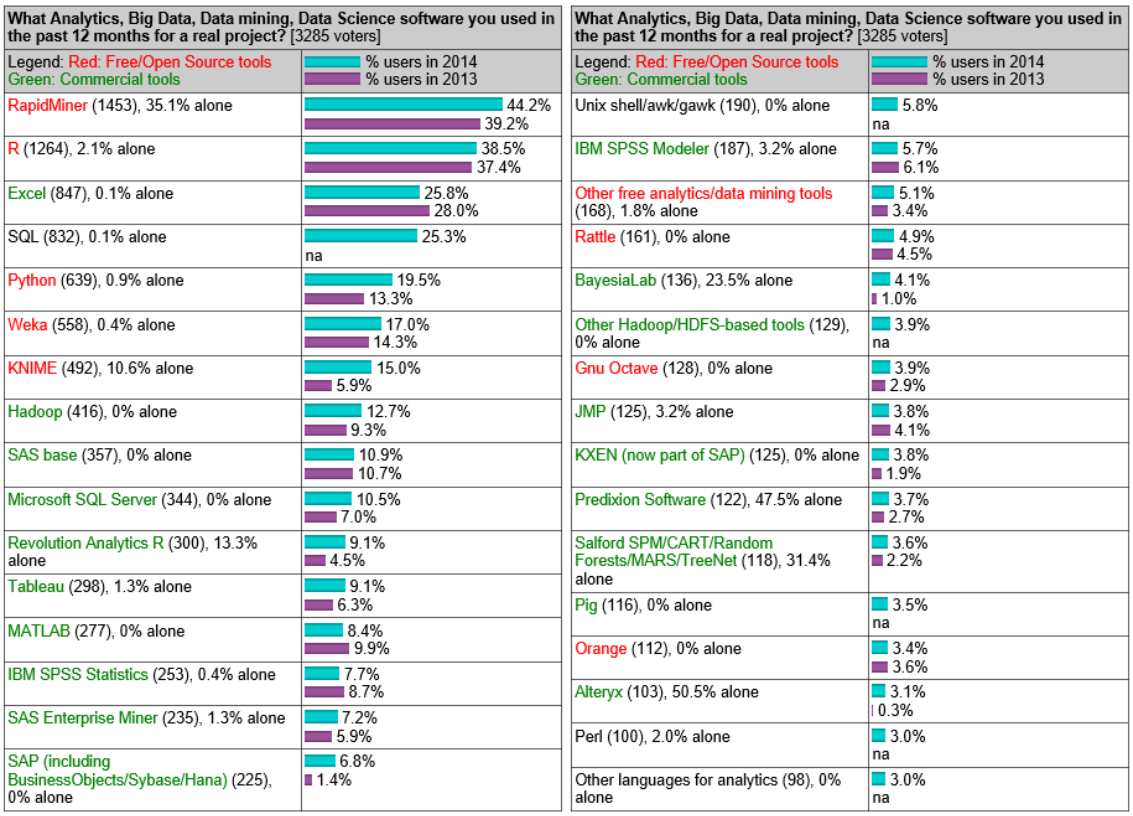
Figura 6: Tareas genéricas del modelo CRISP-DM.

4. Herramientas de analítica

Para el desarrollo del proyecto se han buscado herramientas de uso libre que permitiesen la generación de flujos de trabajo de forma ágil, reproducible y que eviten la necesidad de comentar continuamente las acciones realizadas.

A continuación se comentarán las herramientas que se van analizar en este proyecto para identificar si existe alguna adecuada a los requisitos, previamente presentados, valorando su adecuación al proyecto.

Las herramientas analizadas se han seleccionado tomando como referencia la encuesta realizada por KDnuggets, sitio líder en *Business Analytics*, *Data Mining* y *Data Science*, sobre el uso de herramientas en proyectos reales entre los meses de junio del año 2013 y 2014, *"What Analytics, Data Mining, Data Science software/tools you used in the past 12 months for a real project poll"*, en el que se identifican las principales herramientas usadas en el sector [8]. Los resultados de esta encuesta se presentan a continuación.



What Analytics, Big Data, Data mining, Data Science software you used in the past 12 months for a real project? [3285 voters]	
Legend: Red: Free/Open Source tools Green: Commercial tools	% users in 2014 % users in 2013
RapidMiner (1453), 35.1% alone	44.2% 39.2%
R (1264), 2.1% alone	38.5% 37.4%
Excel (847), 0.1% alone	25.8% 28.0%
SQL (832), 0.1% alone	25.3% na
Python (639), 0.9% alone	19.5% 13.3%
Weka (558), 0.4% alone	17.0% 14.3%
KNIME (492), 10.6% alone	15.0% 5.9%
Hadoop (416), 0% alone	12.7% 9.3%
SAS base (357), 0% alone	10.9% 10.7%
Microsoft SQL Server (344), 0% alone	10.5% 7.0%
Revolution Analytics R (300), 13.3% alone	9.1% 4.5%
Tableau (298), 1.3% alone	9.1% 6.3%
MATLAB (277), 0% alone	8.4% 9.9%
IBM SPSS Statistics (253), 0.4% alone	7.7% 8.7%
SAS Enterprise Miner (235), 1.3% alone	7.2% 5.9%
SAP (including BusinessObjects/Sybase/Hana) (225), 0% alone	6.8% 1.4%

What Analytics, Big Data, Data mining, Data Science software you used in the past 12 months for a real project? [3285 voters]	
Legend: Red: Free/Open Source tools Green: Commercial tools	% users in 2014 % users in 2013
Unix shell/awk/gawk (190), 0% alone	5.8% na
IBM SPSS Modeler (187), 3.2% alone	5.7% 6.1%
Other free analytics/data mining tools (168), 1.8% alone	5.1% 3.4%
Rattle (161), 0% alone	4.9% 4.5%
BayesiaLab (136), 23.5% alone	4.1% 1.0%
Other Hadoop/HDFS-based tools (129), 0% alone	3.9% na
Gnu Octave (128), 0% alone	3.9% 2.9%
JMP (125), 3.2% alone	3.8% 4.1%
KXEN (now part of SAP) (125), 0% alone	3.8% 1.9%
Predixion Software (122), 47.5% alone	3.7% 2.7%
Salford SPM/CART/Random Forests/MARS/TreeNet (118), 31.4% alone	3.6% 2.2%
Pig (116), 0% alone	3.5% na
Orange (112), 0% alone	3.4% 3.6%
Alteryx (103), 50.5% alone	3.1% 0.3%
Perl (100), 2.0% alone	3.0% na
Other languages for analytics (98), 0% alone	3.0% na

Figura 7: Herramienta utilizada en los últimos 12 meses.

En la ilustración anterior aparecen las herramientas más usadas en el último año y se muestra la comparativa del uso con respecto al año anterior. Las herramientas de analítica que son consideradas *Open-Source* están señaladas en rojo, las comerciales están en verde y las que están en negro son genéricas.

Por lo tanto, según el estudio anterior, las herramientas de analítica de uso libre más usadas y que se deberían analizar en este proyecto son: RapidMiner, R, Python, Weka, KNIME Other free analytics/data mining tools (no están especificadas), Rattle, GNU Octave y Orange.

Como se busca una herramienta que permita diseñar flujos completos de análisis de datos y poder completarlos y seguirlos en cualquier momento, el lenguaje de programación Python y la herramienta de análisis matemático GNU Octave no se van a analizar y Rattle se analizará conjuntamente con R.

El análisis final se hará con las herramientas RapidMiner, R / Rattle, WEKA, KNIME y Orange. Estas herramientas son igual de capaces que muchos productos con un alto precio asociado y la mayoría cuentan con una gran comunidad de desarrolladores.

4.1. Descripción y análisis de herramientas

A continuación se describen cada una de las herramientas seleccionadas y se analizan según los requisitos planteados, valorados de la siguiente forma: 0, no se puede realizar; 2’5, existen bastantes restricciones para su correcto desempeño; 5, se efectúa de forma aceptable; 7’5, el cumplimiento es satisfactorio; 10, no hay restricción alguna para llevarlo a cabo.

4.1.1. RapidMiner

RapidMiner[2] (anteriormente YALE, *Yet Another Learning Environment*) era la plataforma de minería de datos de código abierto más usada (con más de 3 millones de descargas), antes de convertirse en un producto comercial.

Ofrece un entorno integrado para el aprendizaje automático, minería de datos, minería de texto, análisis predictivos y análisis de negocio, incorpora extracción, transformación y carga de datos, y presentación de informes de predicción. La interfaz de usuario y las herramientas de visualización gráficas son excelentes, con una inteligencia considerable integrada en el proceso de construcción de flujos de trabajo. Esto proporciona el reconocimiento de errores y sugerencias de soluciones rápidas. Su capacidad de transformación de datos es única entre las herramientas de esta naturaleza y permite que los resultados sean inspeccionados al momento.

La versión inicial fue desarrollada por el departamento de inteligencia artificial de la Universidad de Dortmund en 2001, Nueva Zelanda. Se distribuye bajo licencia AGPL (*Affero General Public License*) y está hospedado en *SourceForge*, sitio web de colaboración para proyectos de software, desde el 2004. En la actualidad, RapidMiner se ha convertido en un producto comercial que sólo mantiene como *open-source* la versión anterior, lo que ha bloqueado su desarrollo.

Incorpora más de 500 operadores propios e incluye también la biblioteca de aprendizaje de WEKA por lo que muchas extensiones están disponibles para el análisis de series temporales y de texto y otros procesos especializados.

La mayoría de las fuentes de datos son compatibles, como Excel, Access, Oracle, IBM DB2, Microsoft SQL Server, Sybase, Ingres, My SQL, archivos de texto y otros.

Se utiliza para aplicaciones industriales y de negocio, así como para la investigación, la educación, la formación, la creación rápida de prototipos y el desarrollo de aplicaciones. Es compatible con todos los pasos del proceso de minería de datos, incluyendo la visualización, la validación y optimización de los resultados.

[2] https://rapidminer.com/

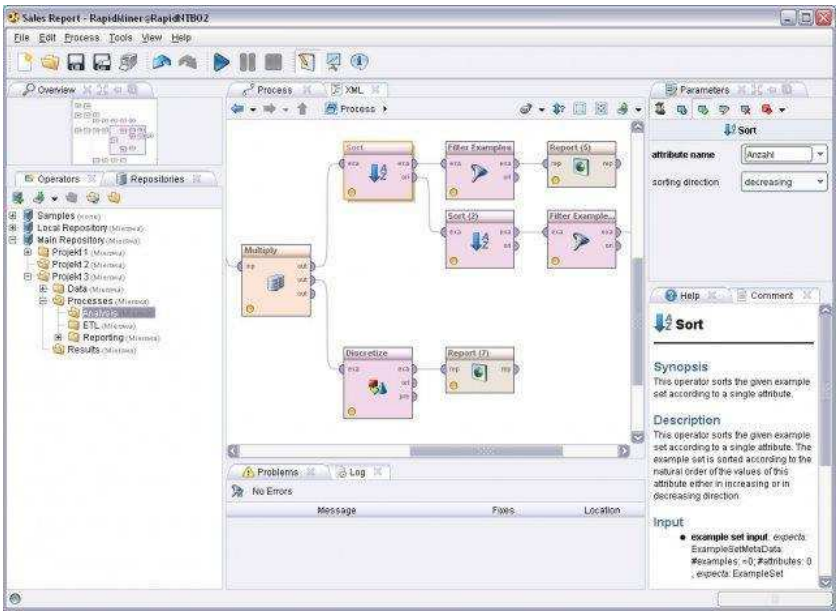


Figura 8: Herramienta RapidMiner.

4.1.1.1. Valoración

Requisito	Valoración	Comentarios
Aprendizaje	10	**Instalación:** ejecutable para todos los sistemas operativos **Escalado:** si **Implementación:** desarrollo de scripts. Cuenta con el apoyo de API completo (*Application Programming Interface*), por lo que proporciona una amplia gama de funcionalidad y soporte **Seguimiento:** flujos de datos **Soporte:** flujos de ejemplo para todos los nodos, internet
Integración	10	**Ficheros:** todos **BBDD:** todas **NoSQL:** algunas
Ejecución	2’5	**Multihilo**: si **RAM:** desde la versión 6 la herramienta permite un uso gratuito para ciertos proyectos, limitada a 1 Gb de memoria RAM, por lo que es inviable para proyectos como este **Puntos intermedios:** no, los flujos se ejecutan completos de principio a fin en cada ejecución
Algoritmos	10	Dispone de multitud de algoritmos propios e integra la librería WEKA de forma nativa. Soporta la integración de código en Java, R y Python entre otros
Extensión	5	Sólo es posible ampliar su funcionalidad mediante código incrustado en nodos
TOTAL	**37’5**	

Tabla 1: Valoración RapidMiner.

4.1.2. R

R[3] es, estrictamente hablando, un lenguaje de programación. También es un software de código abierto en el que se pueden incorporar miles de librerías, haciendo de R la herramienta más flexible y potente del mercado. R es un entorno estadístico que incluye herramientas de manipulación y análisis de datos, cálculo y generación de gráficas y es ampliamente utilizado en minería de datos.

Fue desarrollado inicialmente por Robert Gentleman y Ross Ihaka del Departamento de Estadística de la Universidad de Auckland en 1993.

El desarrollo de proyectos en R se basa en un intérprete de comandos que puede ejecutar scripts de texto, previamente desarrollados en cualquier editor. En la actualidad también existe algún IDE (*Integrated Development Environment*) que permite realizar el desarrollo en R de una forma más amigable, siendo el principal exponente RStudio[4].

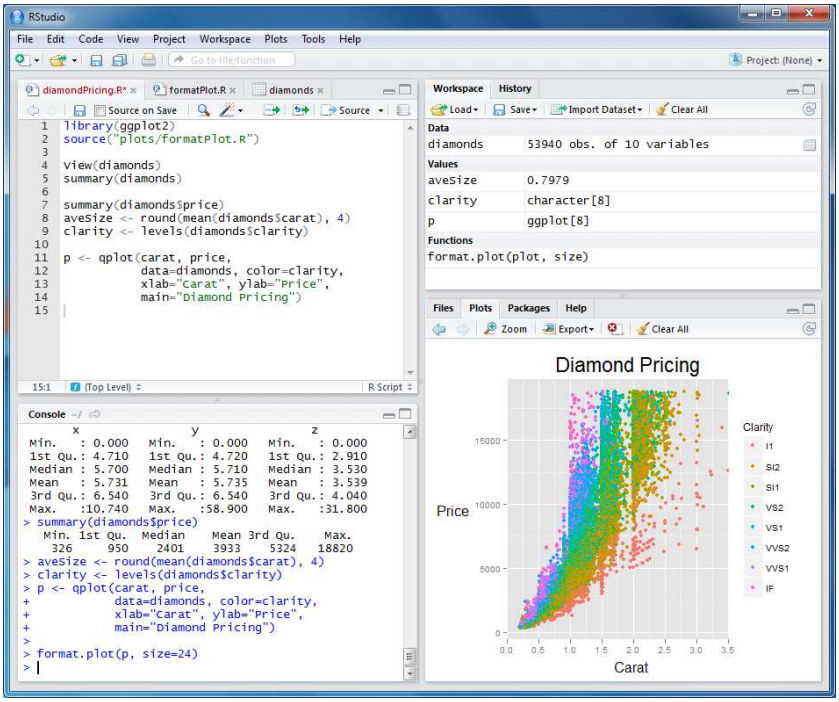


Figura 9: RStudio.

4.1.2.1. Rattle

Rattle[5] es un software libre y de código abierto que proporciona una interfaz gráfica de usuario para minería de datos, utilizando el lenguaje de programación R. Rattle se utiliza actualmente en todo el mundo en una variedad de situaciones.

[3] http://cran.r-project.org/
[4] http://www.rstudio.com/
[5] http://rattle.togaware.com/

Rattle proporciona la funcionalidad de minería de datos a través de una interfaz gráfica de usuario. Se utiliza como centro de enseñanza para aprender el software R, ya que hay un registro que replica el código para cualquier actividad que se realice, que se puede copiar y pegar. Rattle se puede utilizar para el análisis estadístico o la generación de modelos, permite que el conjunto de datos se divida en entrenamiento, validación y pruebas, y el conjunto de datos puede ser consultado y editado. También hay una opción para marcar un archivo de datos externo.

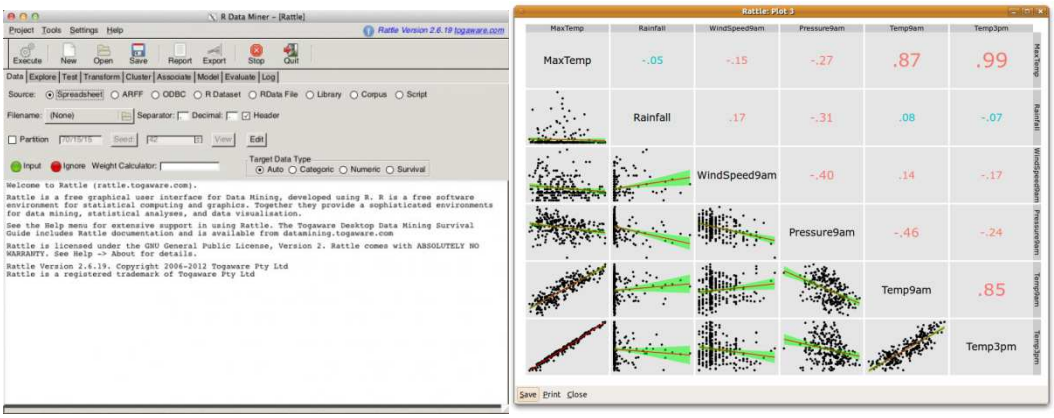


Figura 10: Rattle.

4.1.2.2. Valoración

La valoración de esta herramienta es la siguiente:

Requisito	Valoración	Comentarios
Aprendizaje	5	**Instalación:** ejecutable para todos los sistemas operativos **Escalado:** no **Implementación:** desarrollo de scripts. Rattle soporta un flujo no modificable de procesado de datos **Seguimiento:** scripts **Soporte:** internet
Integración	10	**Ficheros:** todos **BBDD:** todas **NoSQL:** algunas mediante librerías
Ejecución	7’5	**Multihilo:** algún paquete lo permite **RAM:** por defecto sólo trabaja con la RAM disponible **Puntos intermedios:** no
Algoritmos	10	Dispone de multitud de algoritmos desarrollados por la comunidad y distribuidos en paquetes que son *open-source*.
Extensión	5	Sólo es posible ampliar su funcionalidad programando en código R cualquier algoritmo
TOTAL	**37’5**	

Tabla 2: Valoración Rattle.

4.1.3. WEKA

WEKA (*Waikato Environment for Knowledge Analysis*)[6] es un conjunto de librerías de aprendizaje automático escrito en lenguaje de programación Java. Contiene una colección de algoritmos para tareas típicas de minería de datos: clustering, clasificación, preprocesamiento de datos, regresión, reglas de asociación, visualización y selección de características.

Este conjunto de librerías están incorporadas en muchos otros productos, como KNIME y RapidMiner, por ejemplo. El formato por defecto para los datos es el archivo plano. Los modelos pueden ser construidos mediante una interfaz gráfica de usuario o una entrada de línea de comandos.

La versión original se diseñó inicialmente como herramienta para analizar datos procedentes del dominio de la agricultura y la versión más reciente se utiliza en muchas y muy diferentes áreas, en particular con finalidades docentes y de investigación.

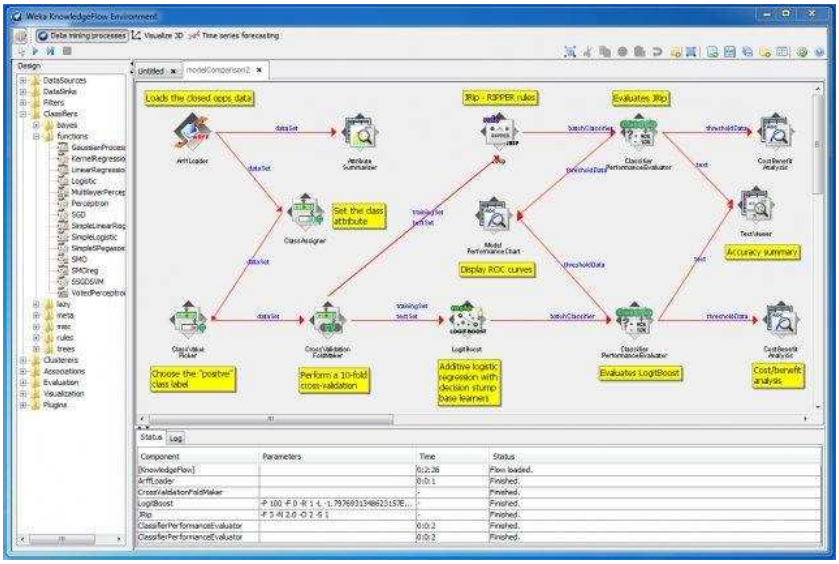


Figura 11: WEKA.

4.1.3.1. Valoración

Requisito	Valoración	Comentarios
Aprendizaje	7’5	**Instalación:** código Java ejecutable en todos los sistemas operativos **Escalado:** no **Implementación:** desarrollo de código Java. Soporta flujos con la interfaz de Pentaho aunque son básicos y no permiten el procesado de los datos **Seguimiento:** flujos de datos y código Java **Soporte:** internet

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

Integración	7’5	**Ficheros:** todos **BBDD:** todas **NoSQL:** no
Ejecución	10	**Multihilo:** si **RAM:** toda la RAM disponible **Puntos intermedios:** si, los flujos se ejecutan completos de principio a fin en cada ejecución
Algoritmos	7’5	Es la librería Java más amplia, aunque no todos los algoritmos están implementados
Extensión	5	Sólo es posible ampliar su funcionalidad programando en Java fuera de la plataforma
TOTAL	**37’5**	

Tabla 3: Valoración WEKA.

4.1.4. KNIME

KNIME (*Konstanz Information Miner*) [7] es una herramienta ampliamente aplicada a minería de datos que permite el desarrollo y ejecución de técnicas mediante modelos en un entorno visual, utilizada por más de 3000 organizadores.

KNIME fue desarrollado originalmente en el departamento de bioinformática y minería de datos de la Universidad de Constanza, Alemania, bajo la supervisión del profesor Michael Berthold. Está concebido como una herramienta gráfica y dispone de una serie de nodos (que ofrecen distintos tipos de algoritmos) y flechas (que representan flujos de datos) que se despliegan y combinan de manera gráfica e interactiva.

La interfaz gráfica de usuario permite el montaje fácil y rápido de nodos para el preprocesamiento de datos, extracción, transformación y carga (*ETL*), modelado, análisis de datos, aprendizaje automático, visualización y minería de datos a través de su concepto modular de canalización de datos (*data pipelining*).

KNIME incorpora cientos de diferentes nodos e integra diversos componentes para aprendizaje automático y minería de datos a través de su concepto de fraccionamiento de datos modular.

Desde 2006 KNIME no sólo se utiliza en la investigación farmacéutica, también se usa para análisis de datos de cliente de CRM (*Customer Relationship Management*), en inteligencia de negocio y para análisis de datos financieros.

KNIME se distribuye bajo la Licencia Pública General de GNU (GNU *General Public License*) que es la licencia más usada en el mundo del software y garantiza a los usuarios finales (personas, organizaciones, compañías) la libertad de usar, estudiar, compartir y modificar el software.

[7] https://www.knime.org/

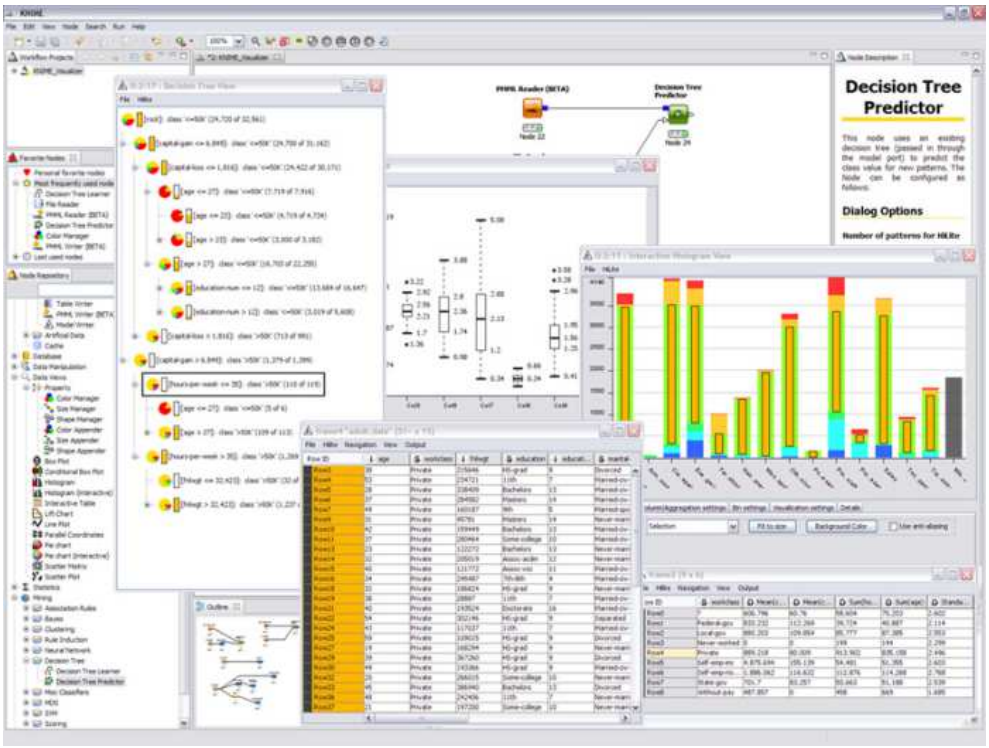


Figura 12: KNIME.

4.1.4.1. Valoración

Requisito	Valoración	Comentarios
Aprendizaje	10	**Instalación:** ejecutable para todos los sistemas operativos **Escalado:** posibilidades de contratar licencias y servidores web **Implementación:** flujos visuales que soportan la extracción, transformación y carga de los datos, la analítica y la exportación de los resultados **Seguimiento:** flujos visuales **Soporte:** internet, comunidad, foros propios
Integración	10	**Ficheros:** todos **BBDD:** todas **NoSQL:** Hadoop
Ejecución	10	**Multihilo:** automático a todos los núcleos de la máquina **RAM:** disponible **Puntos intermedios:** automáticos después de cada nodo
Algoritmos	10	Dispone de multitud de nodos con algoritmos implementados. Integra los algoritmos de la librería Weka
Extensión	10	Es posible desarrollar nodos propios e incluir código Java, Python o R
TOTAL	**50**	

Tabla 4: Valoración KNIME.

4.1.5. Orange

Orange[8] es una herramienta de visualización y análisis de código abierto fácil de usar. La mayoría de los análisis se pueden lograr a través de su interfaz de programación visual, arrastrando y soltando pequeñas aplicaciones o *widgets*, y la mayoría de las herramientas visuales están contempladas: incluyendo diagramas de dispersión, gráficos de barras, árboles, dendogramas y mapas de calor.

Son compatibles un gran número de *widgets*, más de 100, que permiten la transformación y preprocesado de datos, clasificación, regresión, asociación, visualización, modelado, evaluación de modelo, técnicas de exploración y aprendizaje no supervisado. Hay también complementos especializados en bioinformática, minería de textos y otros requisitos especiales. El entorno es extensible mediante secuencias de comandos en Python e incluye la creación de nuevos *widgets* si es necesario.

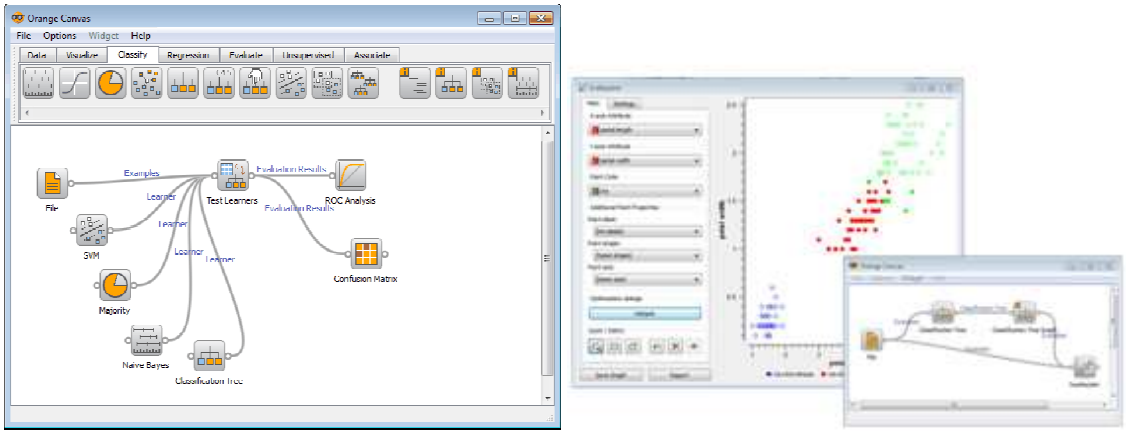


Figura 13: Herramienta Orange.

4.1.5.1. Valoración

La valoración de esta herramienta es la siguiente:

Requisito	Valoración	Comentarios
Aprendizaje	10	**Instalación:** ejecutable para todos los sistemas operativos **Escalado:** no **Implementación:** flujos visuales que soportan la extracción, transformación y carga de los datos, la analítica y la exportación de los resultados. Scripts de Python. **Seguimiento:** flujos visuales y/o scripts **Soporte:** internet
Integración	5	**Ficheros:** csv **BBDD:** PostgeSQL **NoSQL:** no

[8] http://orange.biolab.si/

Ejecución	5	**Multihilo:** automático a todos los núcleos de la máquina **RAM:** disponible **Puntos intermedios:** no
Algoritmos	5	Sólo los algoritmos disponibles en Python
Extensión	7’5	Es posible desarrollar nodos propios de la herramienta o incluir código Python de forma nativa
TOTAL	**32’5**	

Tabla 5: Valoración Orange.

4.2. Evaluación de herramientas

Además de las valoraciones anteriores en función de los requisitos, se van a incorporar, de forma adicional, la valoración de la popularidad y satisfacción con estas herramientas, para asegurarse un soporte futuro y una evolución continua.

4.2.1.1. Popularidad y satisfacción

Para evaluar la popularidad y la satisfacción con las herramientas analizadas se tomarán como referencia diferentes estudios que han sido realizados por diferentes organismos y que se describen a continuación.

KDNuggets

Tomando como referencia los resultados de la encuesta presentados al comienzo de este capítulo, se han valorado las herramientas analizadas de la siguiente forma: 10; herramienta más utilizada; 7’5, segunda más usada; 5, tercera herramienta analizada con mayor porcentaje de uso; 2’5, para la cuarta; 0, para la última.

Rexer Analytics

Otro estudio de referencia es la encuesta “*2013 Data Mining Survey*” realizada por Rexer Analytics en el año 2013 sobre el uso de las diferentes herramientas de analítica y la satisfacción con ellas [9].

De este informe se han obtenido las valoraciones de popularidad empleando el criterio anterior teniendo en cuenta el uso total (uso principal, secundario, frecuente y ocasional) y el uso principal, ver Figura 14.

La satisfacción se ha valorado como sigue: 10; herramienta con un índice de satisfacción más alto; 7’5, segunda con mayor índice; 5, tercera herramienta analizada con un índice alto de satisfacción; 2’5, para la cuarta; 0, para la herramienta Orange por no aparecer en dicho estudio, ver Figura 15.

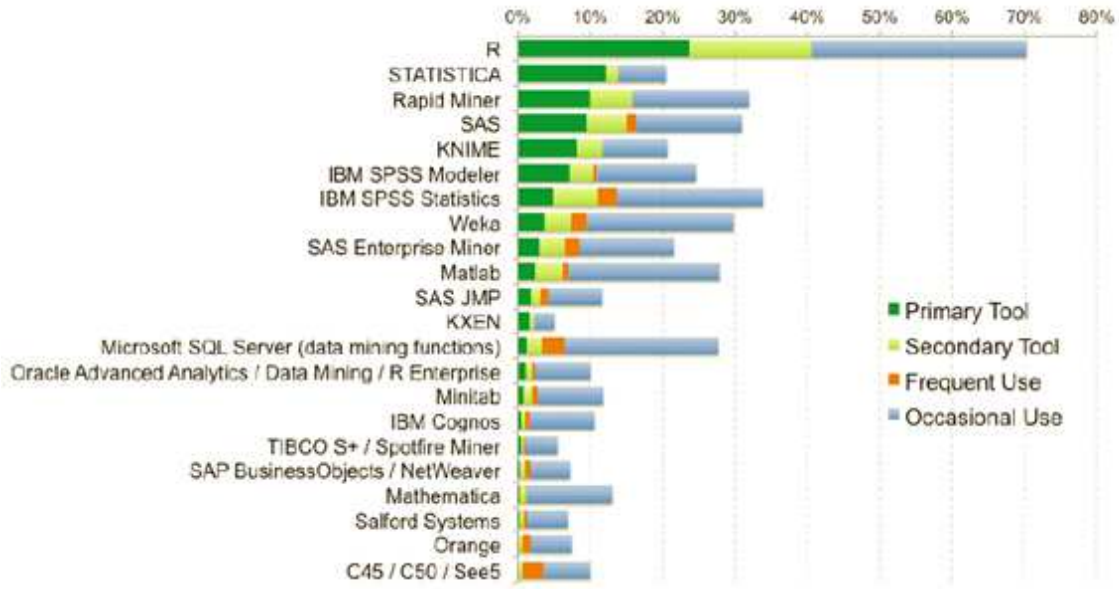


Figura 14: Herramientas ordenadas por uso principal.

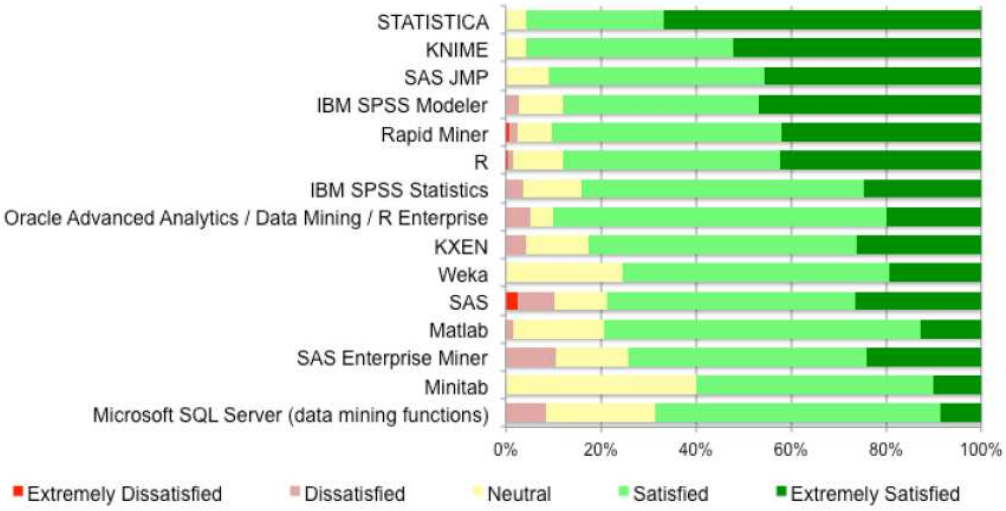


Figura 15: Satisfacción con la herramienta.

Por lo tanto, la valoración media de las herramientas utilizando esta encuesta es:

Herramienta	Uso total	Uso principal	Satisfacción	MEDIA
Rapidminer	7’5	7’5	7’5	**7’5**
R / Rattle	10	10	5	**8’33**
Weka	5	2´5	2´5	**3’33**
KNIME	2’5	5	10	**5’83**
Orange	0	0	0	**0**

Tabla 6: Valoración Rexer Analytics.

4.2.1.2. Características del sistema

Por último se ha utilizado la evaluación de las características de las herramientas de minería de datos de código abierto publicadas en el informe de 2010 *"Open Source Data Mining Software Evaluation"* [10].

Características	RAPIDMINER	RATTLE	WEKA	KNIME	ORANGE
Documentación	4	4	5	5	3
Fácil de aprender	3	4	4	4	5
Usabilidad	5	5	4	4	5
Apoyo	5	2	5	3	3
Extensibilidad	5	3	5	5	3
Fiabilidad	5	5	5	5	5
Instalación	5	4	5	5	5
Preprocesamiento	5	3	3	5	5
Visualización	4	3	3	5	5
MEDIA	**5**	**4**	**4**	**5**	**4**

Tabla 7: Evaluación características de las herramientas comparadas.

La evaluación se basa en una escala de 5 puntos. Las puntuaciones altas indican mejores resultados: alto, íntegro, fácil y simple; y las puntuaciones bajas son para los resultados negativos: bajo, incompleto, difícil y complejo.

4.3. Selección de la herramienta

A continuación se resumen las evaluaciones anteriores:

Valoración	RAPIDMINER	RATTLE	WEKA	KNIME	ORANGE
Herramienta	37’5	37’5	37’5	50	32’5
KDNuggets	10	7´5	5	2’5	0
Rexer Analytics	7´5	8´33	3’33	5’83	0
Características	10	8	8	10	8
TOTAL	**65**	**61’33**	**53’83**	**68´33**	**40’5**

Tabla 8: Valoración total.

Por lo tanto, una vez evaluadas todas las herramientas *open-source* que permiten el desarrollo del proyecto, una vez valorados el cumplimiento de los requisitos y necesidades y su influencia y desarrollo, la herramienta seleccionada para el estudio de minería de datos a realizar ha sido KNIME, por ser la más valorada.

Esta valoración se debe, en parte, a que el carácter abierto de la herramienta hace posible su extensión mediante la creación de nuevos nodos que implementen algoritmos a la medida del usuario, que existe la posibilidad de incorporar de manera sencilla código desarrollado en R y que guarda en memoria todas las salidas ejecutadas en cada nodo sin necesidad de ejecutar el flujo.

4.4. Descripción de la herramienta

A continuación detallaremos cada uno de los componentes en los que está organizada la herramienta KNIME [11, 12].

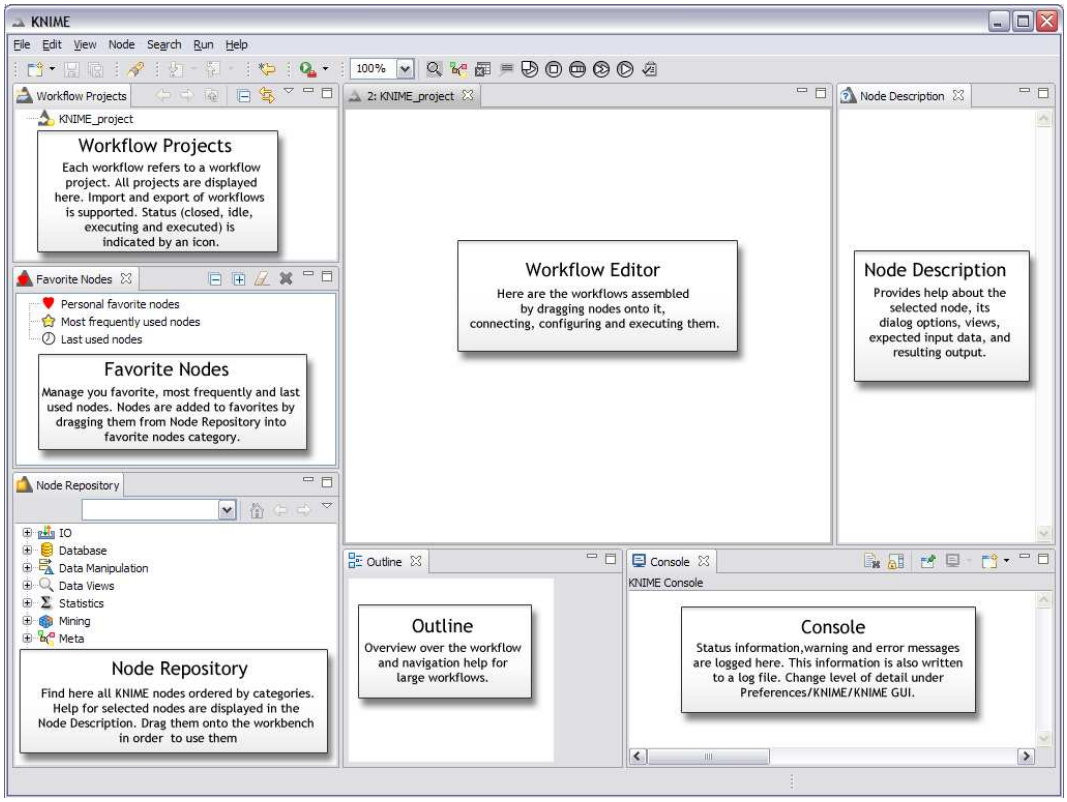


Figura 16: Organización de la herramienta KNIME.

- Proyectos (*Workflow Projects*):
  Cada flujo de trabajo se refiere a un proyecto. En esa ventana se muestran todos los proyectos disponibles, existiendo la posibilidad de importar y exportar otros flujos. Para cada uno de ellos se muestra el estado en el que están mediante un icono (cerrado, inactivo, ejecución y ejecutado).

- Nodos favoritos (*Favorite Nodes*):
  En este componente se pueden administrar los nodos favoritos, los nodos usados con mayor frecuencia o los últimos nodos utilizados, arrastrándolos desde el repositorio.

- Repositorio (*Node Repository*):
  Aquí se encuentran todos los nodos KNIME ordenados por categorías. La ayuda del nodo seleccionado se muestra en la ventana descripción del nodo. Para utilizar el nodo se arrastrará a la ventana de trabajo.

- Editor de flujo de trabajo (*Workflow Editor*):
  Los nodos arrastrados a esta sección podrán ser configurados, conectados y ejecutados, permitiendo examinar y explorar los datos. Todos estos nodos reunidos formarán el flujo de trabajo.

- Descripción del nodo (*Node Description*):
  Proporciona ayuda sobre el nodo seleccionado (o los nodos contenidos en una categoría o metanodo). En particular se describen las opciones de diálogo, las vistas disponibles, los datos de entrada esperados y los datos de salida resultantes.

- Esquema (*Outline*):
  Visión general de todo el flujo de trabajo, incluso si sólo es visible una pequeña parte en el editor (recuadrado en gris en el esquema). Esta ventana se puede utilizar para la navegación, desplazando el rectángulo gris con el ratón, útil para grandes flujos de trabajo.

- Consola (*Console*):
  Los mensajes de error y de advertencia se registran aquí con el fin de dar información acerca de lo que está sucediendo. Esta información también se escribe en un archivo que se guarda en el directorio de trabajo.

Los nodos implementan distintos tipos de acciones que pueden ejecutarse sobre una tabla de datos. Alguna de las acciones que pueden emplearse son:

- Manipulación de datos (*Data Manipulation*):
  Trasponer matrices, ordenar filas, filtrar filas o columnas, combinar o dividir columnas, convertir y remplazar valores de columnas, etc. Todo ello nos permitirá hacer muestreos, transformaciones, agrupaciones, etc.

- Visualización de datos (*Data Views*)*:*
  Histogramas, diagramas de caja, gráficos de dispersión, tablas interactivas, etc.

- Estadísticas (*Statistics*):
  Mínimo, máximo, media, desviación, varianza, etc. También permite la realización de test de hipótesis y de modelos de regresión lineal y polinómica.

- Minería (*Mining*):
  Permite la creación de modelos estadísticos y de minería de datos: árboles de decisión, análisis de componentes principales, clusterización; incluye nodos para validación de modelos, aplicación de dichos modelos sobre conjuntos nuevos de datos, matriz de confusión, etc.

Con la combinación de estos nodos pueden crearse informes a medida gracias a su integración con BIRT (*Business Intelligence and Reporting Tools*).

5. Predicción del *churn*

5.1. Introducción

Este trabajo está enmarcado en el tema de minería de datos [13, 14], que consiste en la exploración, extracción de información y análisis de grandes cantidades de datos con el fin de descubrir patrones y reglas significativas, y corresponde a un problema de clasificación supervisada, en el que se parte de un conjunto de elementos descrito por un conjunto de características donde se conoce la clase a la que pertenece cada uno.

Un concepto fundamental en el ámbito de los métodos de clasificación son los diversos criterios para la evaluación de los clasificadores, esto nos permite efectuar una medición sobre la capacidad de predicción del modelo generado. Se profundizará en esto en el apartado 5.6.

El objetivo del presente trabajo será predecir, mediante alguna técnica de clasificación, aquellos clientes que se darán de baja en los tres meses siguientes a partir de los últimos datos reales aportados por un operador de telecomunicaciones, que por motivos de confidencialidad no se citará. Tampoco se aportarán nombres ni contenidos específicos de las variables incluidas en los datos.

El estudio se va a realizar con las seis bases de datos recibidas a finales de junio de 2014, con un total de registros comprendidos entre 250.000 y 350.000 y conteniendo entre 6 y 12 variables, y para su desarrollo se seguirán cada una de las fases del modelo CRISP-DM que se mencionaron con anterioridad en el apartado 3.

5.2. Comprensión del negocio

En un proyecto general de análisis de datos esta fase es probablemente la más importante, ya que aglutina las tareas de comprensión de los objetivos y requisitos del proyecto desde la perspectiva de la empresa-cliente con el fin de convertirlos en objetivos técnicos y en un plan de proyecto. Sin lograr comprender dichos objetivos, ningún algoritmo, por muy sofisticado que sea, permitirá obtener resultados fiables.

Con la comprensión del problema a resolver se pueden recoger los datos correctos e interpretar correctamente los resultados. En este proyecto, la dedicación necesaria a esta fase es muy pequeña porque Optares Solutions cuenta con experiencia previa y conocimientos específicos en el sector *telecom* que le permiten identificar los objetivos adecuados al estudio a realizar en el proyecto.

5.3. Comprensión de los datos

Esta fase incluye la recogida inicial de los datos, identificar su calidad y establecer las primeras hipótesis. En este proyecto es necesario analizar las fuentes y los tipos de datos multiestructurados, buscando la forma más eficaz de almacenarlos, teniendo en cuenta los análisis que se van a llevar a cabo sobre ellos en fases posteriores.

Se contará con datos anonimizados distribuidos en seis bases de datos de texto plano: cuentas, direcciones, productos de fijo, productos de móvil, opciones de televisión y opciones de velocidad.

Para cada una de las bases de datos se comprobará que todas columnas tengan especificado un nombre, que no haya variables con todos sus campos vacíos, el número total de factores para variables categóricas, la existencia de atípicos para variables numéricas, la presencia de valores perdidos,… Por último, se relacionarán las bases de datos mediante alguna variable coincidente, observando la estructura jerárquica que se muestra a continuación.

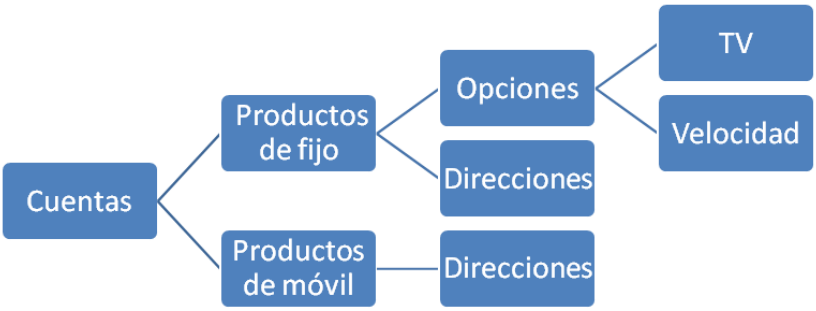


Figura 17: Estructura jerárquica de los datos.

5.4. Preparación de los datos

En esta fase se procede a la preparación de los datos para adaptarlos a las técnicas de análisis que se utilicen posteriormente. En esta fase se incluyen también las tareas generales de selección de datos a los que se va a aplicar una determinada técnica de modelado, limpieza de datos, generación de variables adicionales e integración de diferentes orígenes de datos.

Con el fin de generar una única base de datos que contenga toda la información necesaria para realizar predicciones del *churn* efectuaremos una serie de pasos:

- Selección de la ventana temporal
- Eliminar registros erróneos
- Creación y agrupación de variables
- Unión base de datos

Para cada uno de estos pasos crearemos flujos de trabajo en KNIME usando los nodos que aparecen en el repositorio de la herramienta.

5.4.1. Selección de la ventana temporal

En este trabajo el objetivo es la creación de un modelo para la identificación temprana de bajas voluntarias. Este modelo será la base para la realización de campañas de retención conociendo la deserción voluntaria predicha.

Entendiendo por identificación temprana de bajas voluntarias a aquellas efectuadas en el trimestre siguiente a la recepción de los datos, se construirá un modelo generando una imagen real de cómo eran los datos hace tres meses, que se utilizará como entrenamiento, y se usarán los datos recibidos como test [7].

Para seleccionar cada uno de los trimestres crearemos un flujo en el que se leerán todas las bases de datos y se eliminarán aquellos registros con fecha de baja anterior al primer día el trimestre de estudio o fecha de alta después del último día de ese trimestre. Por tanto, nos quedaremos con aquellos registros que se dan de alta o se dan de baja en el trimestre y aquellos que siguen activos a la finalización del mismo.

En la siguiente figura se muestra el caso particular de la selección la ventana temporal aplicado a cada uno de los trimestres de este estudio.

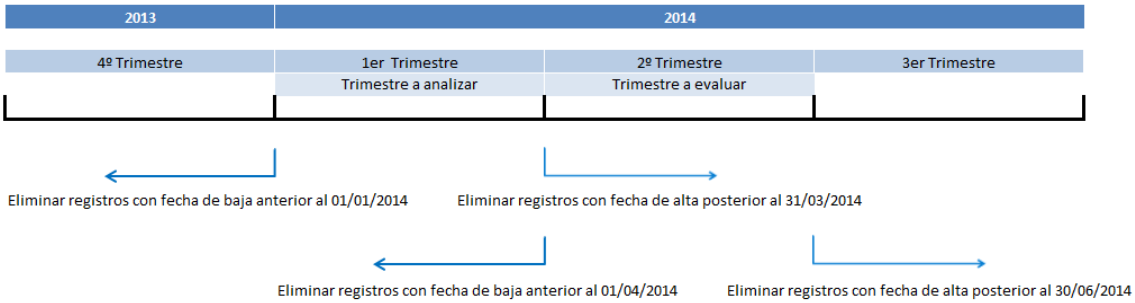


Figura 18: Selección de ventana temporal.

Como resultado de este proceso, tendremos divididas cada una de las bases de datos en dos, conteniendo la información de los registros activos y dados de baja en el primer y segundo trimestre del año, respectivamente, y que guardaremos para utilizarlas en el siguiente paso.

5.4.1.1. Flujo en KNIME

Para obtener de una manera rápida y automatizada las bases de datos en la que se han seleccionando aquellos registros activos o que se dan de baja en el trimestre en el que estamos haciendo la selección temporal hay que seguir los siguientes pasos:

1) Seleccionar el directorio en el que se encuentras las bases de datos originales para proceder a su lectura (botón derecho sobre el metanodo → *Configure* → *Browse*).

…\Proyecto final\Bases de datos originales\

Dentro de este metanodo se generan las rutas de cada una de las bases de datos proporcionadas, evitando así tener que modificar cada uno de los ficheros de lectura cada vez que se tengan datos nuevos. Sólo es necesario especificar la carpeta en la que se encuentran. Generamos a su vez una variable para cada una de las bases de datos para proceder a su lectura tal y como se muestra en la ilustración siguiente.

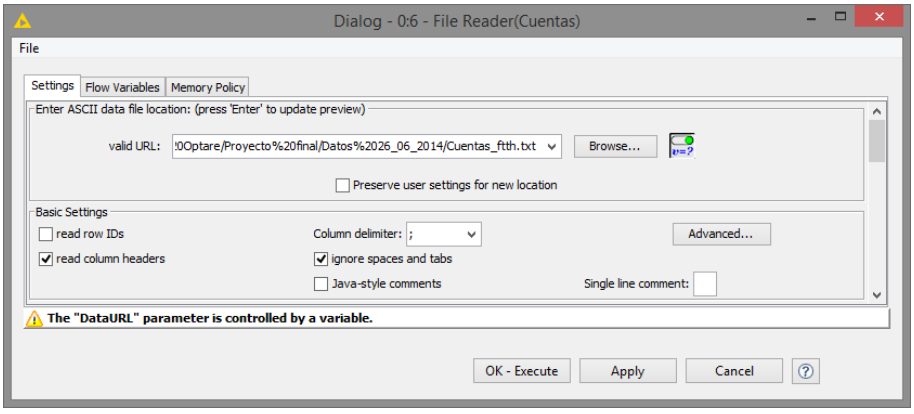


Figura 19: Lectura de base de datos.

En la figura anterior se muestra un ejemplo de la configuración del nodo de lectura: la ruta del fichero ha sido especificada por una variable, se han modificado ajustes básicos (encabezado, delimitador de columnas, ignorar espacios y tabulaciones) y ajustes avanzados (separador de decimales y seleccionado el formato de codificación UTF-8 para la correcta lectura de los caracteres).

2) Seleccionar la fecha de inicio y fin de la ventana temporal (botón derecho sobre el metanodo → *Configure* → Fecha de inicio (primer día del trimestre a estudiar); Fecha fin (último día del trimestre a estudiar). Formato: dd/mm/yyyy).

Del mismo modo que antes, y con el fin de no tener que hacer los mismos cambios en todos los flujos asociados a cada base de datos cada vez que se quieren seleccionar los registros de un determinado trimestre, se han generado como variables auxiliares la fecha de inicio y fin de la ventana temporal en el que deben estar recogías las altas y bajas de esos registros.

A continuación se muestra la configuración utilizada para seleccionar aquellos registros activos al final del segundo trimestre del año o se han dado de baja en ese periodo.

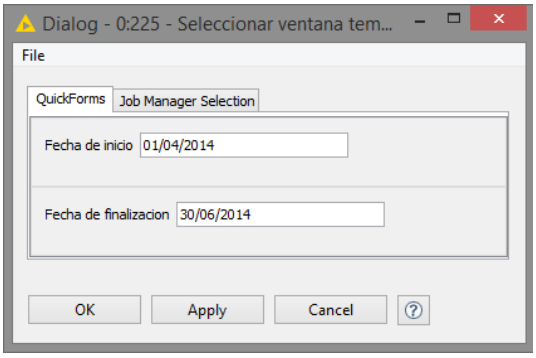


Figura 20: Configuración del metanodo.

Teniendo en cuenta estas variables se ha aplicado un filtro a cada una de las seis bases de datos eliminando los registros que tuviesen una fecha de baja anterior a la fecha de inicio o una fecha de alta posterior a la fecha de finalización.

3) Seleccionar la raíz en la que guardar la base de datos del trimestre de estudio (botón derecho sobre el metanodo → *Configure* → *Browse*).

…\Proyecto final\ Segundo trimestre\

Una vez obtenidas las bases de datos con el filtro anterior aplicado, procedemos a guardarlas en el directorio deseado utilizando de nuevo una variable para no tener que ir nodo a nodo modificándolo. A continuación se muestra la configuración del metanodo que es similar al utilizado en el paso 1.

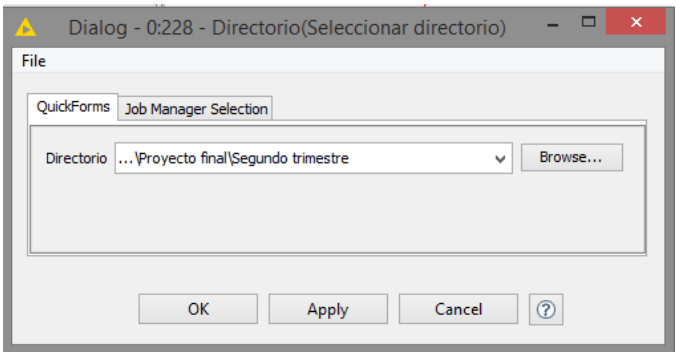


Figura 21: Seleccionar el directorio en el que guardar las bases de datos.

4) Ejecutar el flujo.

Por último debemos ejecutar el flujo.

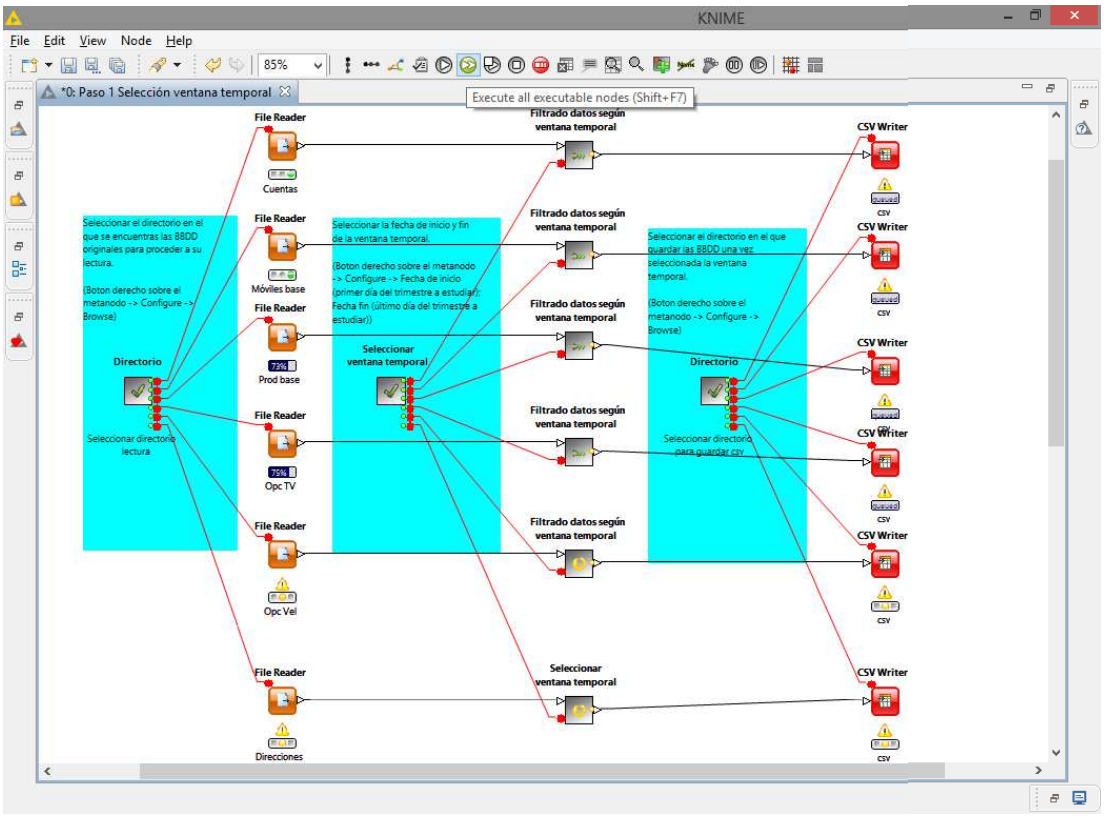


Figura 22: Flujo selección ventana temporal.

A medida que se van ejecutando los nodos aparece una barra de proceso. Los nodos ejecutados aparecen en verde, los nodos en amarillo indican que aún están por ejecutar y si hubiera un nodo en rojo significaría que ese nodo no está configurado. En los nodos que están a la espera de ser ejecutados pone *queued*. Pueden aparecer mensajes de aviso que se muestran en la consola o pasando el cursor por encima.

Este flujo tendremos que ejecutarlo dos veces, una por cada trimestre, modificando las fechas de inicio y fin del trimestre en el paso 2 y cambiando la carpeta en la que guardar las bases de datos.

5.4.2. Eliminar registros erróneos

Para cada uno de los instantes temporales seleccionados se eliminarán aquellos registros que presentan alguna disconformidad para alguna de sus variables. Todos los registros eliminados son por cuestiones de negocio y a la hora de enviarse los datos no se han depurado y se han detectado en la fase de comprensión de los datos al analizar la calidad de los datos.

Por ejemplo, una de las acciones realizadas ha sido la eliminación, en todas las bases de datos, de aquellos registros que tienen un mismo identificador de producto para códigos de cuenta distintos, para que pueda hacerse una correcta unión entre ellas. El número total de registros eliminados no ha superado el 1’5% de los datos en ninguna de las bases de datos.

5.4.2.1. Flujo en KNIME

Los pasos a realizar para efectuar el filtrado de los registros erróneos para las bases de datos son:

1) Seleccionar la raíz del trimestre de estudio (botón derecho sobre el metanodo → *Configure* → *Browse*).
…\Proyecto final\ Segundo trimestre\

Siguiendo el mismo formato que en el apartado anterior se genera una variable especificando la raíz en la que fueron guardadas las bases de datos.

2) Ejecutar el flujo.

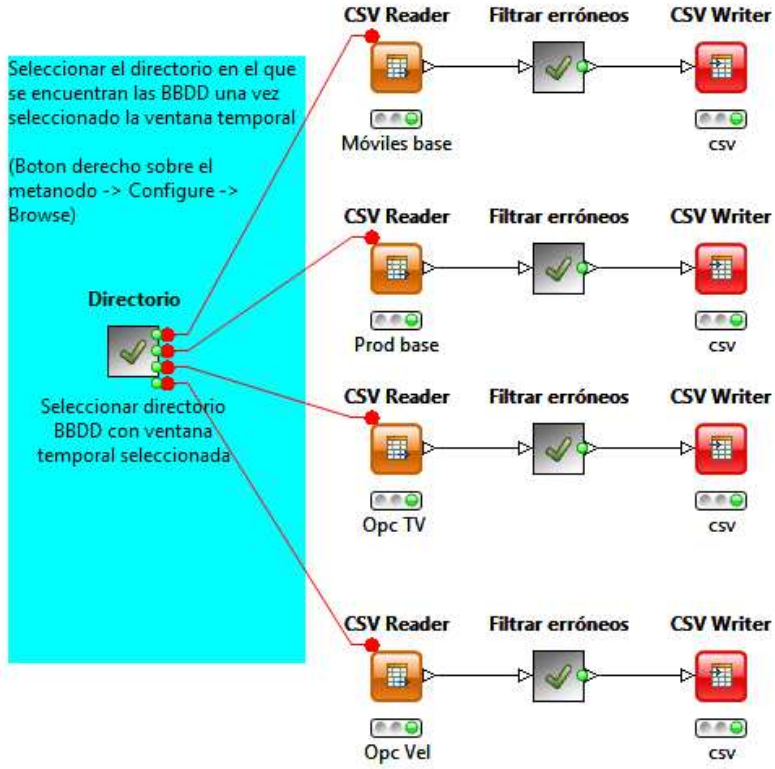


Figura 23: Flujo eliminar erróneos.

5.4.3. Creación y agrupación de variables

Es necesario crear nuevas variables para obtener la mayor información posible de los datos. En este paso se crearán variables de interés a partir de las variables existentes y se agruparán para obtener un único código de cuenta y así tener un único código identificativo.

En primer lugar se unirá la base de datos de direcciones a la de productos base y móviles base, ya que esta base de datos está ligada a las otras dos, como se mostró en la estructura jerárquica de la Figura 17, a partir de una variable que tienen en común.

En segundo lugar se crearán las variables de interés pasando de tener entre 6 y 12 a tener entre 27 y 89 variables y se agruparán por código de cuenta.

5.4.3.1. Flujo en KNIME

Para realizar este paso en KNIME se hará lo siguiente:

1) Seleccionar la fecha de finalización del trimestre (Formato: dd/mm/aaaa) y el directorio raíz en el que están guardadas las bases de datos (Botón derecho sobre el metanodo → *Configure* → *Browse*).

…\Proyecto final\ Segundo trimestre\

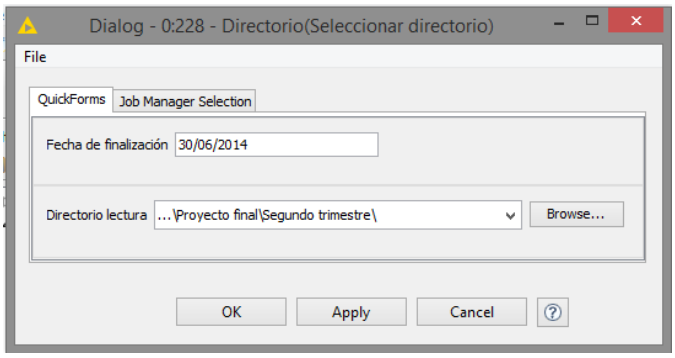


Figura 24: Seleccionar el directorio en el que guardar las bases de datos.

2) Ejecutar el flujo.

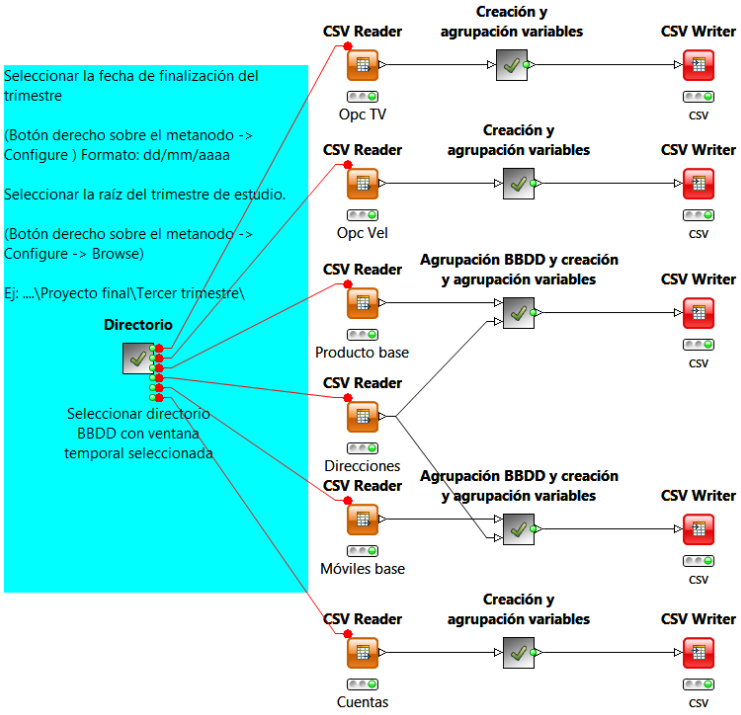


Figura 25: Flujo creación y agrupación de variables.

Cada una de las cajas que aparecen en gris son metanodos en los que se incluyen distintas operaciones realizadas con los nodos del repositorio, que facilitan una mejor visualización del flujo.

5.4.4. Unión de bases de datos

Una vez creadas y agrupadas las distintas variables a partir de las originales se unirán las distintas bases de datos en una.

5.4.4.1. Flujo en KNIME

1) Seleccionar la raíz del trimestre de estudio (botón derecho sobre el metanodo → *Configure* → *Browse*).

…\Proyecto final\ Segundo trimestre\

2) Ejecutar el flujo.

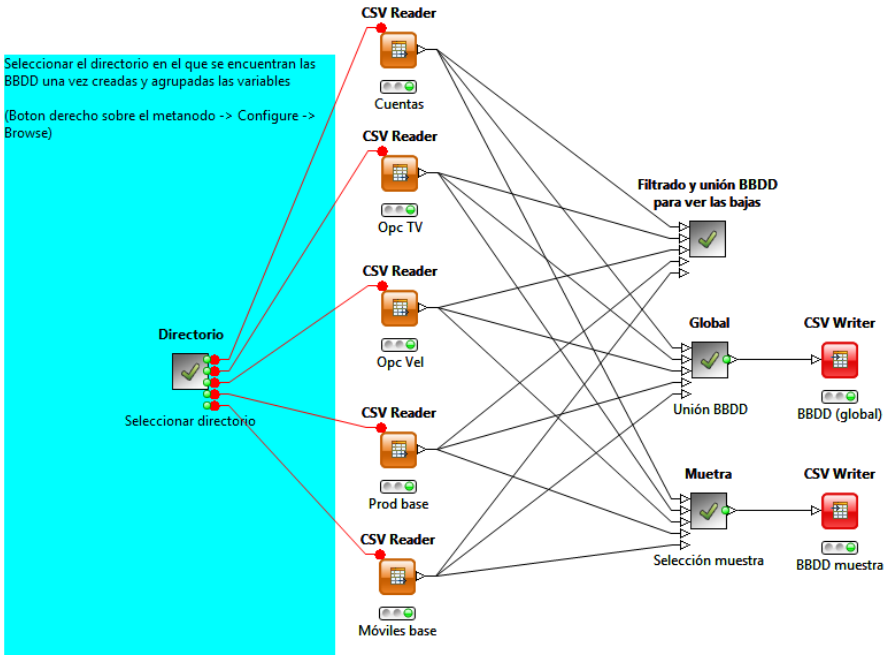


Figura 26: Flujo unión bases de datos.

Al final de la ejecución de este flujo se ha elaborado una base de datos global con los registros incluidos en todas las base de datos, obteniendo un total de 210000 registros y 205 variables.

Por otro lado, se ha construido una base de datos en la que se ha hecho un muestreo aleatorio estratificado sin reposición con un tamaño de 10000 registros para, si fuese el caso, usar en pasos posteriores y agilizar la obtención de resultados.

Estos tres últimos pasos, eliminar registros erróneos, creación y agrupación de variables y unión de bases de datos, se ejecutaran dos veces, una para cada ventana temporal, logrando tener así dos bases de datos finales, entrenamiento y test, que utilizaremos en la siguiente fase del modelo CRISP-DM.

5.5. Modelado

En esta fase se aplicarán varias técnicas de clasificación utilizando los nodos del repositorio de KNIME, con posibilidad de configuración de alguno de los parámetros. Entre las distintas técnicas se han utilizado:

- Naive Bayes; Decision Tree; KNN; Boosting; Tree ensemble.

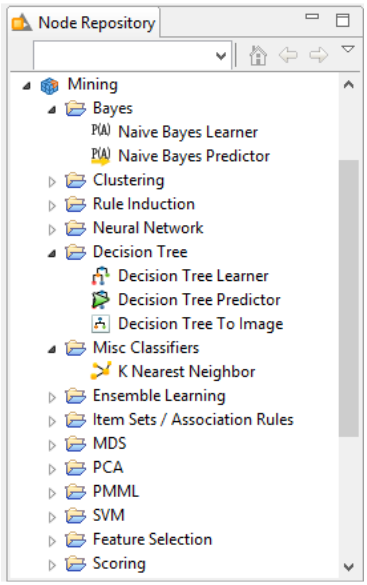


Figura 27: Repositorio de KNIME para minería de datos.

5.5.1. Naive Bayes

Naive Bayes es uno de los más eficientes y eficaces algoritmos de aprendizaje inductivo. Su rendimiento competitivo en la clasificación es sorprendente, porque la suposición de independencia condicional en la que se basa es rara vez cierta en aplicaciones del mundo real [13,15].

Naive Bayes es una técnica sencilla para la construcción de clasificadores: modelos que asignan etiquetas a los casos problemáticos. Los clasificadores de Naive Bayes son clasificadores probabilísticos simples basados en la aplicación del teorema de Bayes que asumen que el valor de una característica particular es independiente del valor de cualquier otra característica.

Por ejemplo, una fruta puede ser considerada como una manzana si es de color rojo, redonda y aproximadamente 7 cm de diámetro. Un clasificador de Naive Bayes considera cada una de estas características de manera independiente para contribuir a la probabilidad de que esta fruta es una manzana, independientemente de las posibles correlaciones entre las características de color, redondez y diámetro.

Para algunos tipos de modelos los clasificadores de Naive Bayes se pueden entrenar de manera muy eficiente en un entorno de aprendizaje supervisado. En muchas aplicaciones prácticas los modelos Naive Bayes utilizan el método de máxima verosimilitud para la estimación de parámetros. Una ventaja de Naive Bayes es que sólo se requiere de una pequeña cantidad de datos de entrenamiento para estimar los parámetros necesarios para la clasificación.

La clasificación bayesiana representa un método de aprendizaje supervisado, así como un método estadístico para la clasificación. Supone un modelo probabilístico subyacente que nos permite capturar la incertidumbre sobre el modelo de una manera basada en principios, determinando probabilidades de los resultados.

La clasificación bayesiana proporciona algoritmos de aprendizaje práctico y los conocimientos previos y los datos observados se pueden combinar. La clasificación bayesiana proporciona una perspectiva útil para la comprensión y evaluación de muchos algoritmos de aprendizaje, calcula probabilidades explícitas para hipótesis y es robusto al ruido en los datos de entrada.

KNIME ofrece dos nodos que pueden ser combinados entre sí para el desarrollo de este algoritmo, *Naive Bayes Learner* y *Naive Bayes Predictor*.

El primero de ellos crea un modelo bayesiano a partir de los datos de entrenamiento, se tomará la base de datos creada del primer trimestre, calculando el número de registros por categoría para la clase seleccionada si las variables son nominales, y la distribución gaussiana si las variables son numéricas.

En la configuración de este nodo: se selecciona la variable de clasificación, en nuestro caso ‘baja objetivo’; permite ignorar los valores perdidos en el modelo, si no es marcada la opción el modelo trata los valores perdidos como un posible valor y los considera durante el cálculo de la probabilidad de la clase; se puede especificar el número máximo de valores considerados para las variables nominales, aquellas con más valores que los especificados se omitirán en el aprendizaje. Ver Figura 28.

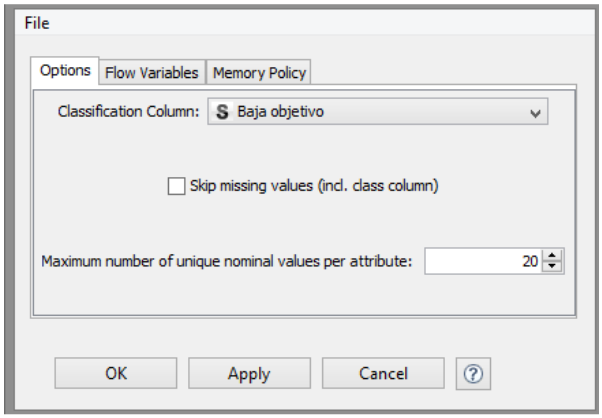


Figura 28: *Naive Bayes Learner*.

El segundo nodo predice la clase sobre la base de datos del segundo trimestre, a partir del modelo aprendido. La probabilidad de la clase es el producto de las probabilidades de las variables para esa clase. Para los valores nominales es el número de ocurrencias de la clase dada entre el número de ocurrencias totales y la probabilidad para valores numéricos se calcula suponiendo distribución normal.

En la configuración de este nodo se puede especificar que además de la predicción de la clase devuelva la probabilidad normalizada de que cada instancia pertenezca a una clase.

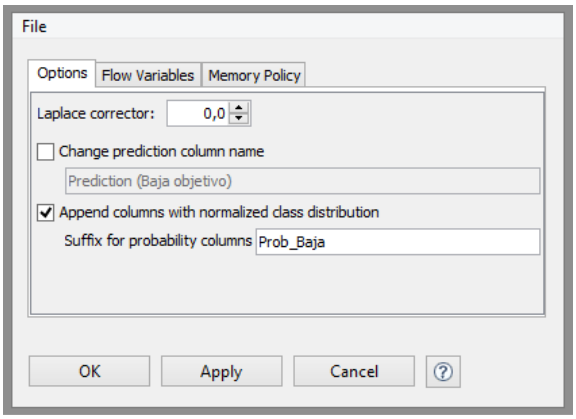


Figura 29: *Naive Bayes Predictor.*

Obtener la probabilidad de pertenecer o no a una clase nos servirá para dibujar la curva ROC (ver 5.6) y no quedarnos sólo con la predicción asignada por defecto, calculada a partir del valor 0'5.

5.5.2. Árboles de decisión

Un árbol de decisión es un modelo de predicción utilizado en el ámbito de la inteligencia artificial [13]. Dada una base de datos se construyen diagramas de construcciones lógicas muy similares a los sistemas de predicción basados en reglas, que sirven para representar y categorizar una serie de condiciones que ocurren de forma sucesiva para la resolución de un problema.

Un árbol de decisión tiene unas entradas, las cuales pueden ser un objeto o una situación descrita por medio de un conjunto de atributos, y a partir de ellas devuelve una respuesta. Los valores que pueden tomar las entradas y las salidas pueden ser valores discretos o continuos. Se utilizan más los valores discretos por simplicidad. Cuando se utilizan valores discretos se denomina clasificación y cuando se utilizan los continuos se denomina regresión [16].

El árbol de decisión suele contener nodos internos, nodos de probabilidad, nodos hojas y ramas. Un nodo interno contiene un test sobre algún valor de una de las propiedades, un nodo de probabilidad indica que debe ocurrir un evento aleatorio de acuerdo a la naturaleza del problema. Un nodo hoja representa el valor que devolverá el árbol de decisión y finalmente las ramas brindan los posibles caminos que se tienen de acuerdo a la decisión tomada. Un árbol de decisión lleva a cabo un test a medida que se recorre hacia las hojas para alcanzar así una decisión.

Un árbol de decisión indica las acciones a realizar en función del valor de una o varias variables. Es una representación en forma de árbol cuyas ramas se bifurcan en función de los valores tomados por las variables y que terminan en una acción concreta.

Los árboles de decisión son diagramas de decisiones. Éstos ayudan a las empresas a determinar cuáles son sus opciones al mostrarles las distintas decisiones y sus resultados. La opción que evita una pérdida o produce un beneficio extra tiene un valor. La habilidad de crear una opción tiene un valor que puede ser comprado o vendido.

Se usará el árbol de decisión como aprendizaje y utilizado como modelo predictivo. Es uno de los enfoques de modelado predictivo que se utilizan en las estadísticas, minería de datos y aprendizaje automático.

Un árbol de decisión puede ser usado para representar visualmente y de forma explícita la toma de decisiones.

Al igual que en el algoritmo anterior KNIME ofrece dos nodos que pueden ser combinados entre sí, *Decision Tree Learner y Decision Tree Predictor*.

El primero de ellos induce un árbol de decisión de clasificación en memoria. La variable objetivo debe ser nominal. Los otros atributos que se utilizan para la toma de decisiones pueden ser nominales o numéricos. Las divisiones numéricas son siempre binarias, divide el dominio en dos particiones. Las divisiones nominales pueden ser binarias o pueden tener tantos resultados como valores nominales tenga la variable. En el caso de una división binaria los valores nominales se dividen en dos subconjuntos.

El algoritmo proporciona dos medidas de calidad para el cálculo de división: el *Gain ratio* y el *Gini index* [15]. Además, existe un método de post poda para reducir el tamaño del árbol y aumentar la precisión de la predicción. Este método de poda se basa en el principio de longitud de descripción mínima (*MDL*).

El algoritmo permite un criterio de parada estableciendo el número mínimo de registros requeridos en un nodo y se puede ejecutar en múltiples hilos, y por lo tanto, aprovechar varios procesadores o núcleos. El valor predeterminado se establece en el número de procesadores disponibles para KNIME. Si el valor es 1, el algoritmo se ejecuta de manera secuencial.

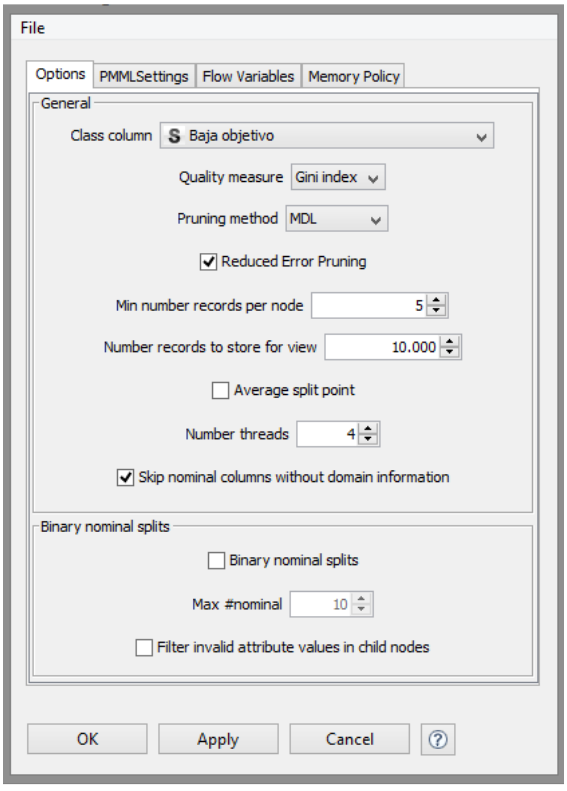


Figura 30: *Decision Tree Learner.*

La mayoría de las técnicas utilizadas en la aplicación del árbol de decisión se pueden encontrar en (Quinlan, 1993) y en (Shafer, Agrawal y Mehta, 1996).

Este nodo se unirá a la base de datos de entrenamiento que, como se ha mencionado con anterioridad, contiene la información de los registros activos al final del primer trimestre de 2014 o dados de baja durante esos meses, para cada una de las variables consideradas en cada una de las bases de datos proporcionadas.

El segundo nodo utiliza un árbol de decisión existente para predecir el valor de la clase. En este nodo se aplicará el modelo extraído del nodo anterior a la base de datos generada con la información de las variables para el segundo trimestre de 2014.

5.5.3. Algoritmo KNN

Este algoritmo clasifica un conjunto de datos de prueba basados en el algoritmo de k vecinos más próximos [13]. Es un método de clasificación supervisado, no paramétrico, que estima el valor de la función de densidad de probabilidad o directamente la probabilidad a posteriori de que un elemento x pertenezca a la clase C_j a partir de la información proporcionada.

Los ejemplos de entrenamiento son vectores en un espacio característico multidimensional. Cada ejemplo está descrito en términos de p atributos considerando q clases para la clasificación. Los valores de los atributos del i-esimo ejemplo (donde $1 \leq i \leq n$) se representan por el vector p-dimensional $x_i = (x_{1i}, x_{2i}, \dots, x_{pi}) \in X$.

El espacio es particionado en regiones según las localizaciones y etiquetas de los datos de entrenamiento. Un punto en el espacio es asignado a la clase C si es la clase más frecuente entre los k ejemplos de entrenamiento más cercano. Generalmente se usa la distancia euclídea, $d(x_i, x_j) = \sqrt{\sum_{r=1}^{p}(x_{ir} - x_{jr})^2}$.

La fase de entrenamiento del algoritmo consiste en almacenar los vectores característicos y las etiquetas de las clases de los datos de entrenamiento. En la fase de clasificación, se calcula la distancia entre los vectores almacenados y el nuevo vector, y se seleccionan los k ejemplos más cercanos. El nuevo dato es clasificado con la clase que más se repite en los vectores seleccionados.

Este método supone que los vecinos más cercanos nos dan la mejor clasificación y esto se hace utilizando todos los atributos. El problema de dicha suposición es que es posible que se tengan muchos atributos irrelevantes que dominen sobre la clasificación. Dos atributos relevantes perderían peso frente a veinte irrelevantes.

Para corregir el posible sesgo se puede asignar un peso a las distancias de cada atributo, dándole así mayor importancia a los atributos más relevantes. Otra posibilidad consiste en tratar de determinar o ajustar los pesos con ejemplos conocidos de entrenamiento. Finalmente, antes de asignar pesos es recomendable identificar y eliminar los atributos que se consideran irrelevantes.

La mejor elección de k depende fundamentalmente de los datos. Generalmente, valores grandes de k reducen el efecto de ruido en la clasificación, pero crean límites entre clases parecidas. Un buen k puede ser seleccionado mediante una optimización de uso. El caso especial en que la clase es predicha para ser la clase más cercana al ejemplo de entrenamiento es conocido como el algoritmo del vecino más cercano (*Nearest Neighbor Algorithm*).

La exactitud de este algoritmo puede ser severamente degradada por la presencia de ruido o características irrelevantes, o si las escalas de las características no son consistentes con lo que uno considera importante.

Para el desarrollo de este algoritmo en KNIME sólo es necesario un nodo *K Nearest Neighbor* que clasifica un conjunto de datos basados en el algoritmo de k vecinos más próximos usando los datos de entrenamiento. El algoritmo subyacente utiliza un árbol k-dimensional y por lo tanto debe presentar un rendimiento razonable. Sin embargo, este tipo de clasificador es apto para unos pocos miles de instancias. Este nodo sólo utiliza las columnas numéricas y la distancia euclídea.

En la configuración del nodo se selecciona la variable utilizada para la clasificación, el número de vecinos más cercanos para clasificar una nueva instancia, se puede asignar un peso en función de la distancia (vecinos más cercanos tienen una mayor influencia en la clase resultante que los más alejados) y como en los algoritmos anteriores se puede obtener la información de las probabilidades de cada clase.

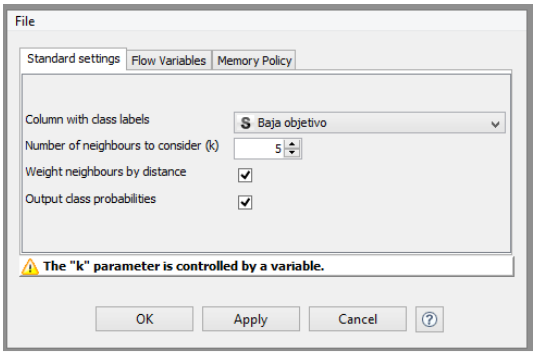


Figura 31: *K Nearest Neighbor*.

Para elegir el número de vecinos más cercanos para clasificar una nueva instancia se recomienda seleccionar un número impar para evitar empates. En nuestro caso hemos creado un flujo en el que hemos seleccionado aquel k que maximiza los criterios de éxito que veremos en la sección 5.6.1. Para ello hemos realizado un bucle obteniendo el valor del criterio de éxito elegido para valores de k comprendidos entre 1 y 31, con un salto de 3 unidades, y hemos seleccionado el k que maximiza el resultado y lo hemos incluido en el nodo para aplicar el algoritmo.

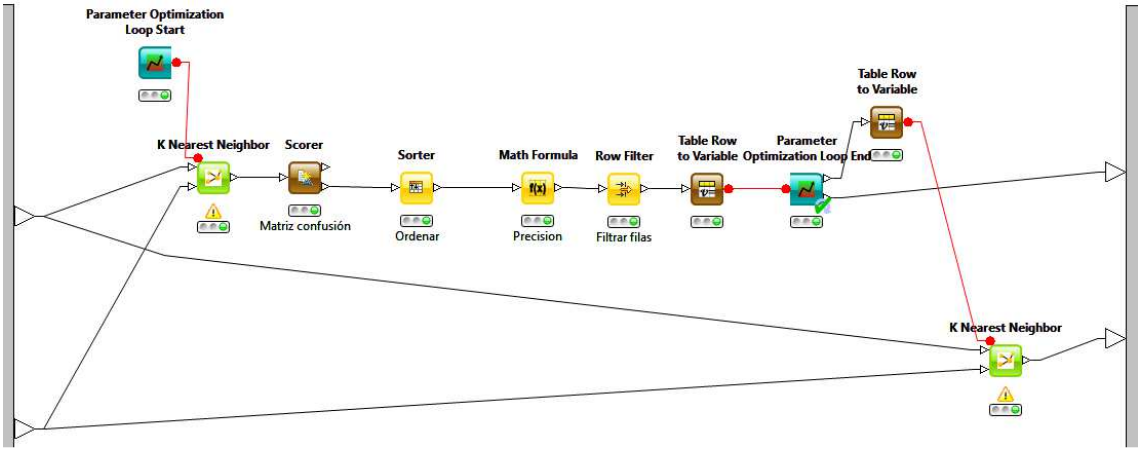


Figura 32: Flujo maximización del criterio según el valor de k.

5.5.4. Boosting Decision Tree

El rendimiento de los árboles de decisión individuales se puede mejorar mediante la técnica de Boosting [13, 19, 20]. La idea es formar un conjunto de clasificadores, pero, en lugar de elegir al azar los datos para cada instancia, como se haría en el caso de Bagging, elegimos el dato que es "más informativo" en cada iteración.

El árbol final se construye a continuación por votación entre los clasificadores individuales. El único inconveniente de Boosting es que oscurece la interpretación simple de un solo árbol de decisión, haciendo que el comportamiento de un árbol de decisión con Boosting sea más parecido al comportamiento aproximando por el vecino más cercano. Sin embargo, la ganancia en el rendimiento suele ser sustancial, de modo que Boosting casi siempre se utiliza con árboles de decisión.

A la hora de aplicar este algoritmo en KNIME son necesarios los nodos *Boosting Learner Loop Star, Boosting Learner Loop End, Boosting, Boosting Predictor Loop Start y Boosting Predictor Loop End*.

Los dos primeros se colocan antes y después del modelo elegido (*Decision Tree*) como se muestra en la ilustración siguiente. El bucle entrena repetidamente árboles de decisión que pondera de acuerdo a su error de clasificación. El algoritmo utilizado es AdaBoost. La salida contiene el conjunto de datos remuestreados y sobremuestreados. Las filas que han sido mal predichas se encuentran con mayor frecuencia que aquellas que se predijo correctamente.

El bucle se detiene una vez que se ha alcanzado el número máximo de iteraciones, indicadas en el nodo *Boosting Learner Loop End*, o si el peso de un modelo es ligeramente superior a 0, indicando que la predicción del error es demasiado grande.

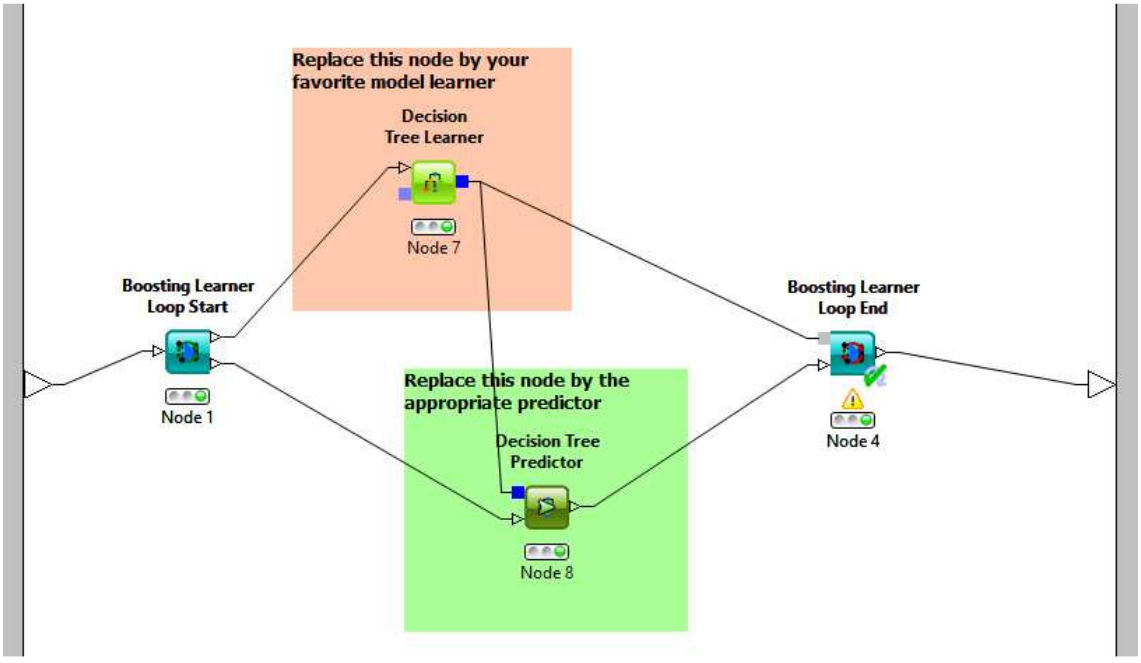


Figura 33: Metanodo *Boosting Learner*.

En la figura anterior se muestran los nodos para entrenar el algoritmo que se aplicará a la base de datos de entrenamiento, que como en casos anteriores, contienen la información del primer trimestre.

Una vez obtenido el mejor modelo se aplica esta información a la base de datos obtenida para el segundo trimestre mediante los otros dos nodos como se muestra en la siguiente ilustración.

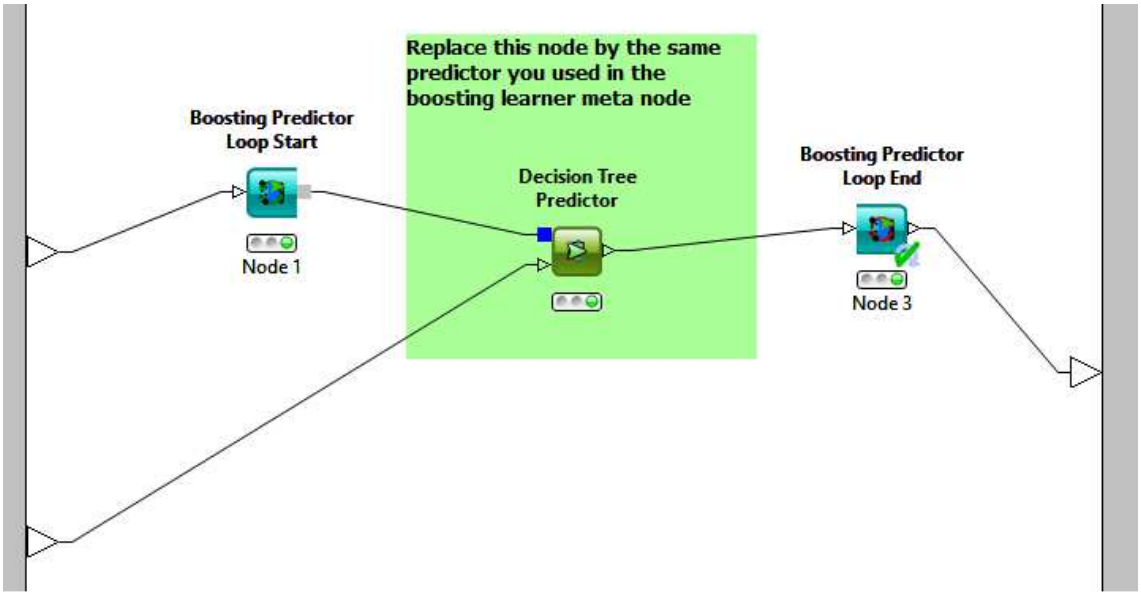


Figura 34: Metanodo *Boosting Predictor*.

Estos nodos utilizan los modelos ponderados obtenidos en la fase de entrenamiento y los aplica creando una predicción basada en la mayoría de votos de todos los modelos.

5.5.5. Tree ensemble

El resultado de los modelos individuales con alguna debilidad puede ser mejorado mediante un método de clasificación que reutiliza uno o más algoritmos de clasificación existentes, o combinando resultados de un mismo algoritmo con diferentes partes de los datos. El objetivo es generar resultados más robustos mediante la combinación de los resultados de los múltiples modelos, ya sea secuencialmente o independientemente [15, 21].

Por esta razón aplicamos *Tree Ensemble* como combinación de modelos *Decision Tree* con diferentes características para evitar un modelo individual y conseguir que, en promedio, se compensen los errores.

En KNIME son necesarios los nodos *Tree Ensemble Learner* y *Tree Ensemble Predictor*. El primero de ellos entrena un conjunto de árboles de decisión. Cada uno de los modelos utiliza un conjunto de diferentes filas y/o un conjunto diferente de columnas. El modelo de salida describe un conjunto de modelos de árboles de decisión y se aplica al nodo siguiente mediante votación de mayoría simple.

Igualmente, el primer nodo se aplica a la base de datos del primer trimestre y el segundo a la base de datos del segundo trimestre.

5.6. Evaluación de resultados

Al principio de este capítulo se introdujo que los criterios de evaluación son un concepto fundamental en el ámbito de los métodos de clasificación. En este trabajo emplearemos la matriz de confusión para este fin [14], comparando cada una de las técnicas de clasificación utilizadas en el apartado anterior.

Para evaluar cada una de las técnicas utilizadas anteriormente se ha seleccionado la base de datos generada con la ventana temporal del primer trimestre como entrenamiento y se han comparado las predicciones obtenidas de la variable objetivo, al aplicar cada uno de los modelos entrenados, con los valores reales de esta misma variable en la base de datos del segundo trimestre.

En nuestro caso la variable objetivo se refiere a la baja de un cliente. Ésta es una variable binaria cuyos resultados posibles usualmente son denotados como 'positivos' y 'negativos'. En esta situación la regla de clasificación basada en la prueba sería la siguiente: los individuos se clasifican como posibles bajas si el resultado de la prueba es positivo y no se consideran bajas si el resultado es negativo.

Cada una de las técnicas de clasificación comentadas anteriormente tiene la opción de devolver directamente la predicción que tiene cada cliente de darse de baja y la compararemos con los resultados reales mediante la matriz de confusión.

		Condicion positiva	Condición negativa	
Resultado de la prueba	Resultado positivo de la prueba	Verdaderos positivos VP	Falsos positivos FP	Precisión (P)
	Resultado negativo de la prueba	Falsos negativos FN	Verdaderos negativos VN	
		Sensibilidad (S)	Especificidad (E)	Exactitud (A)

Figura 35: Matriz de confusión.

En la figura anterior se muestra la matriz de confusión, donde cada fila representa el número de predicciones para cada clase y cada columna las instancias reales. Aporta las siguientes medidas:

- VP: instancias correctamente reconocidas por el clasificador.
- FP: instancias que son negativas pero el clasificador dice que no lo son.
- FN: instancias que son positivas y que el clasificador predice como negativos.
- VN: instancias que son negativas y correctamente reconocidas.
- Sensibilidad: Capacidad de la prueba para identificar una condición correctamente.

$$S = \frac{VP}{VP + FN}$$

- Especificidad: Capacidad de la prueba para excluir una condición correctamente.

$$E = \frac{VN}{VN + FP}$$

- Exactitud o *Accuracy*: Proporción del número total de predicciones que son correctas.

$$A = \frac{VP + VN}{VP + VN + FP + FN}$$

- Precisión: Proporción de casos predichos positivos que son correctos.

$$P = \frac{VP}{VP + FP}$$

Como se puede observar la regla de clasificación basada en un test binario es directa, aportando un único valor para cada medida. Esta regla de clasificación proviene de la probabilidad que cada individuo tiene de pertenecer a una clase u otra y por defecto, para cada una de las técnicas, se toman como valores positivos de la prueba los registros con probabilidad de darse de baja mayor o igual a 0'5 y como resultados negativos a los que tienen probabilidad inferior a 0'5.

Para cada una de las técnicas tenemos la opción de mostrar la probabilidad que tiene cada individuo de darse de baja, configurando el nodo como se mostró en la Figura 29, pasando a tener una respuesta continua. Una regla de clasificación sencilla sería tomar el valor umbral, en este caso la probabilidad (c), y clasificar como resultados positivos, 'Baja=SI', aquellos registros que tengan probabilidad de darse de baja mayor o igual al umbral especificado y como negativos al resto.

Para cada uno de los umbrales obtenemos una especificidad y sensibilidad que podemos representar gráficamente con la curva ROC, *Receiver Operating Characteristic* [14].

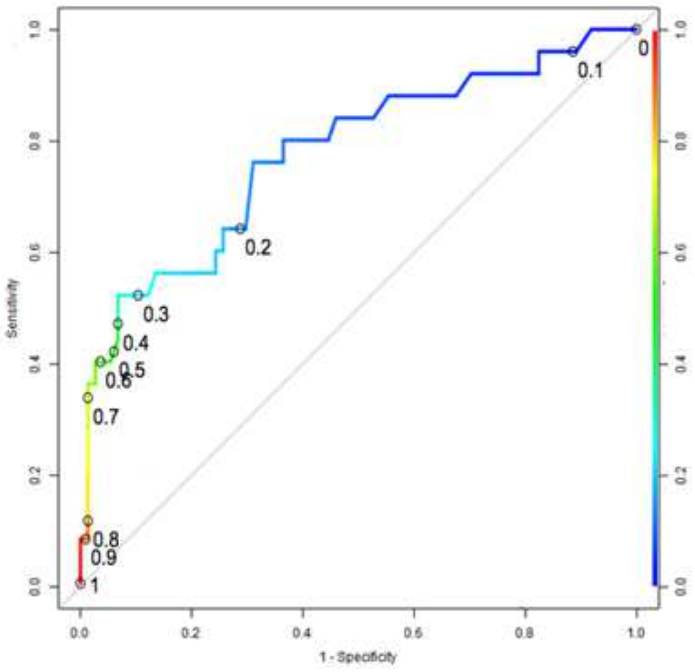


Figura 36: Curva ROC con umbrales.

En la ilustración anterior podemos ver la variación de la sensibilidad y la especificidad para cada valor umbral. Para umbrales pequeños la sensibilidad será grande ya que la mayoría de las predicciones serán positivas y la especificidad será pequeña debido a que habrá pocos valores negativos predichos. Para umbrales cercanos a uno la sensibilidad será pequeña debido a que no habrá muchos valores positivos predichos y la especificidad será grande ya que la mayoría de los registros serán predichos como negativos.

5.6.1. Criterios de éxito y comparativa de curvas ROC

Debido a que la evaluación de los resultados es realizada a partir de la clasificación obtenida según el umbral seleccionado, compararemos los umbrales con mejores resultados según los siguientes criterios de éxito seleccionados:

- *Closest topleft*: El umbral óptimo es el punto más cercano a la parte superior izquierda del gráfico con perfecta sensibilidad o especificidad.

$$ct = \min_{c}\{(1 - S(c))^2 + (1 - E(c))^2\}$$

- El umbral en el que se obtiene una mayor precisión.

- El umbral en el que se obtiene un mayor F1 score, siendo el F1 score una media armónica entre la sensibilidad y la precisión.

$$F1 = 2 \cdot VP/(2 \cdot VP + FN + FP)$$

Por otro lado, para decidir qué técnica de clasificación es mejor nos fijaremos en el área bajo la curva ROC de cada uno de ellas, que es un índice de precisión de la prueba de diagnóstico comprendido entre 0’5 (azar) y 1 (perfecta discriminación). Swets clasifica la exactitud de la prueba del siguiente modo: de 0’5 a 0’7 la exactitud es baja, de 0’7 a 0’9 la exactitud es regular-alta y si es superior a 0’9 la exactitud de la prueba es alta [22].

5.6.2. Selección de la técnica de clasificación

A continuación se muestra el flujo completo para comparar cada uno de las técnicas de clasificación y los resultados obtenidos siguiendo los siguientes pasos:

1) Seleccionar el directorio en el que se encuentran las bases de datos del primer y segundo trimestre (botón derecho sobre cada nodo → *Configure* → *Browse*).
…\Proyecto final\ Segundo trimestre\BBDD global.csv

2) Seleccionar la variable objetivo sobre la que aplicar los algoritmos. (Botón derecho sobre cada metanodo → *Configure* → Especificar 0, 1 ó 2).

   Haciendo este paso se aplica cada una de las técnicas de clasificación anteriormente citadas para cada uno de los casos de estudio que tenemos, que son: determinar aquellos registros que tienen mayor probabilidad de darse de baja a nivel de cuenta, a nivel de productos base y a nivel de móviles base. Especificaremos el valor 0, 1 ó 2 en la configuración de cada metanodo, respectivamente.

3) Ejecutar el flujo.

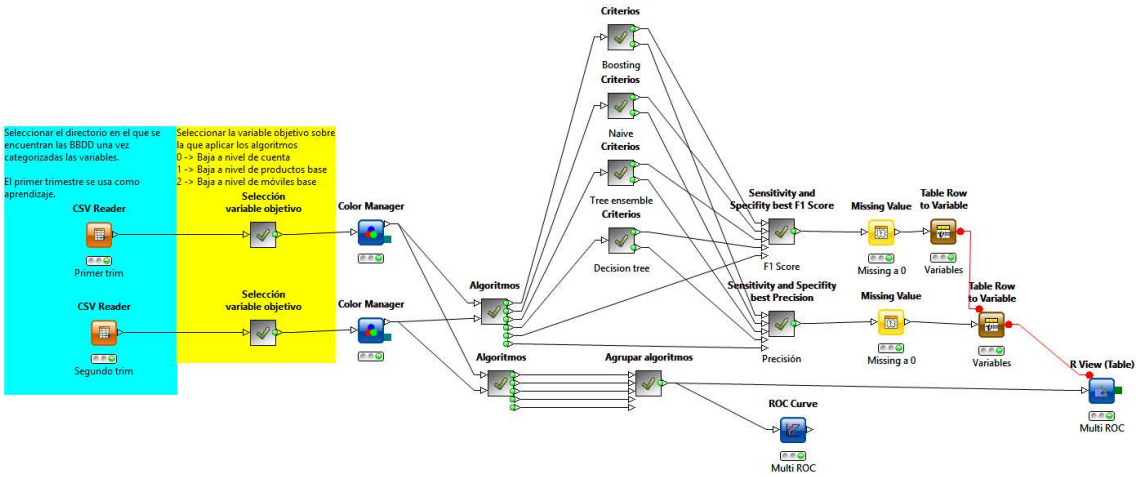


Figura 37: Flujo comparación técnicas de clasificación.

En la Figura 37 se observa el flujo necesario para visualizar las curvas ROC, con los criterios de éxito especificados, para cada una de las técnicas de clasificación comentadas. Los nodos necesarios para la aplicación de estas técnicas se han agrupado en el metanodo 'Algoritmos' para una mejor visualización del flujo completo.

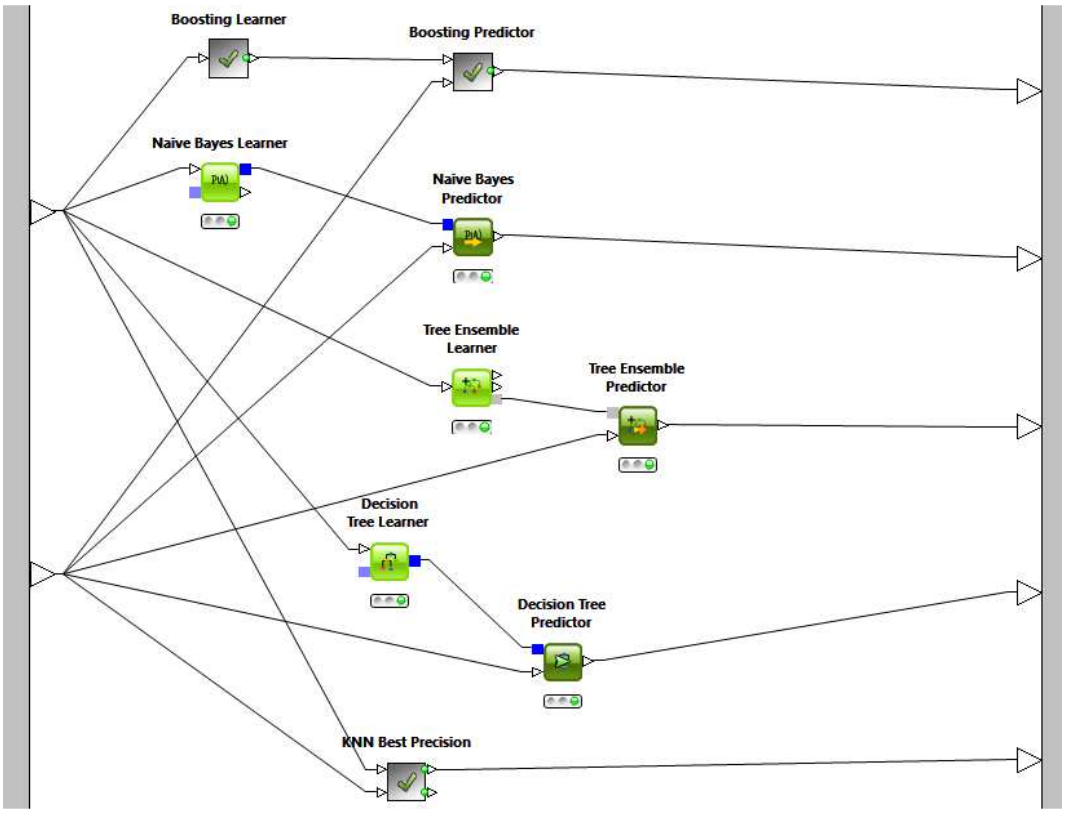


Figura 38: Técnicas utilizados.

Aunque KNIME tiene un nodo específico para dibujar las distintas curvas en un mismo gráfico y devuelve el área bajo la curva para cada una de las técnicas seleccionadas (ROC *Curve*), no permite la representación de la especificidad y sensibilidad para puntos en concretos, por lo que ha sido necesario la inclusión de un nodo que permite replicar código en R para la realización del mismo (*R View Table*).

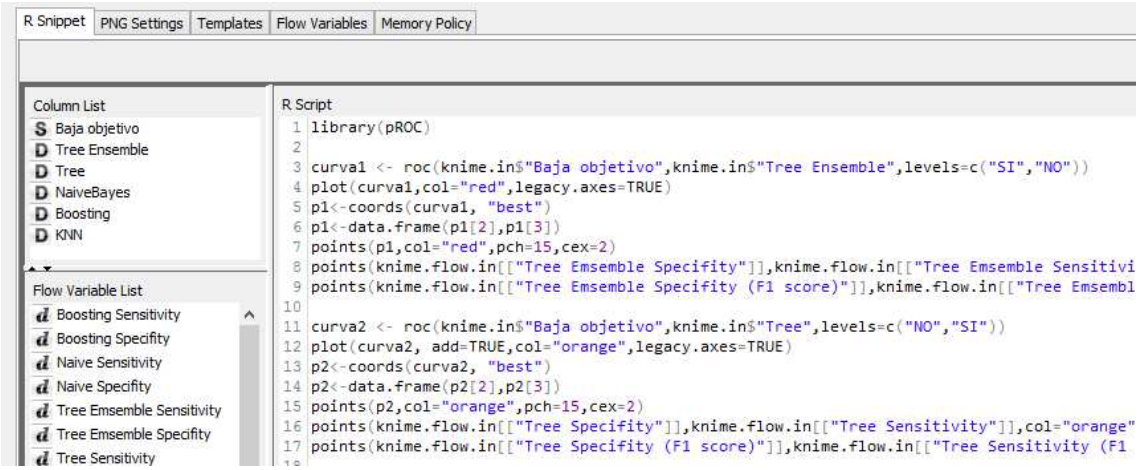
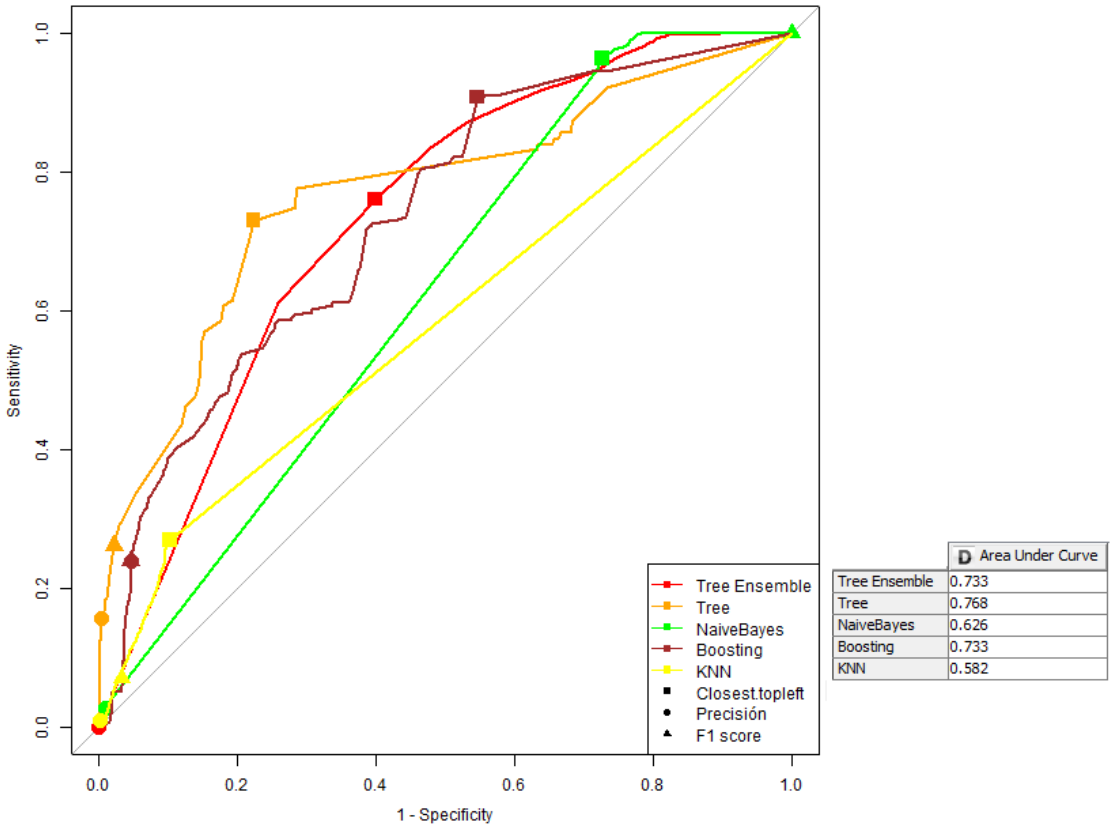


Figura 39: R View Table

En la Figura 39 se observa la configuración del nodo *R View Table*. En la sección *Column List* aparecen las variables asociadas a la predicción realizada con cada una de las técnicas de clasificación y la variable con la que comparar, en *Flow Variable List* aparece el listado de variables que contienen el valor de la sensibilidad y especificidad para cada una de las técnicas y en *R Script* se incluye el código necesario para la representación de las distintas curvas y criterios, ver Anexo.

A continuación se muestra el resultado obtenido. Donde se observan la especificidad y la sensibilidad obtenidas para cada uno de los umbrales, resaltando los criterios de éxito, y el área bajo la curva de cada una de las técnicas.



	Area Under Curve
Tree Ensemble	0.733
Tree	0.768
NaiveBayes	0.626
Boosting	0.733
KNN	0.582

Figura 40: Curvas ROC para cada uno de las técnicas y medidas.

Los resultados de la Figura 40 reflejan que *Decision Tree* es la técnica de clasificación con mayor área bajo la curva, 0'768, que tiene mayor sensibilidad y especificidad para el criterio de éxito Closest topleft y es, después de Boosting, la técnica con mayor sensibilidad y especificidad una vez seleccionado el umbral en el que mayor precisión se obtiene.

En un estudio de aplicación de técnicas de Data Mining a datos *telecom* para el cálculo de bajas se analizaron los artículos publicados en Elsevier, IEEE Xplore, SpringerLink, ScienceDirect, ACM Digital Library y en Microsoft Academic entre los años 2002 y 2013, encontrando un total de 61 artículos relevantes para este tema que se analizan bajo diferentes perspectivas entre las que destacó *Decisión Tree* como la técnica más usada (en un 25% de los artículos) [23].

Technique	Frequency
Decision Tree	26
Neural Network	16
Logistic Regression	15
Cluster Analysis	10
Genetic Algorithm	6
Markov Model	4
Naïve Bayes	4
k-nearest-neighbor	3
Bayesian Belief Network	3
Association Rule	2
Support Vector Machine	2
Bagging	2
CART	2
CHAID	2
K-Means	1
Fuzzy C means	1
influence diffusion model	1
Chr-PmRF	1
partial least squares (PLS)	1
C5.0	1
Structural Equation Model	1
Total	**104**

Figura 41: Técnicas de Data Mining usadas en los artículos analizados.

5.7. Despliegue

En el caso de este proyecto se plantea generar una plataforma que permita integrar los resultados de los estudios realizados dentro de los sistemas OSS y BSS del operador. Hasta que esté implementada la plataforma los resultados se entregarán en formato '.csv'.

5.7.1. Predicciones

Estos resultados se obtendrán aplicando la técnica *Decision Tree*, creando un flujo en KNIME en el que seleccionamos los datos del primer trimestre del 2014 como entrenamiento y los datos del segundo trimestre como test. Obtendremos de esta forma la probabilidad de darse de baja asociada a cada identificador de cliente, que ordenaremos de mayor a menor.

1) Seleccionar el directorio en el que se encuentra la base de datos a entrenar (botón derecho sobre el nodo → *Configure* → *Browse*).
   …\Proyecto final\Primer trimestre\BBDD global.csv

2) Seleccionar el directorio en el que se encuentra la base de datos a testear (botón derecho sobre el nodo → *Configure* → *Browse*).
…\Proyecto final\Segundo trimestre\BBDD global.csv

3) Seleccionar el directorio en el que guardar las predicciones (botón derecho sobre el nodo → *Configure* → *Browse*).
…\Proyecto final\Predicciones fijo.csv

4) Ejecutar el flujo.

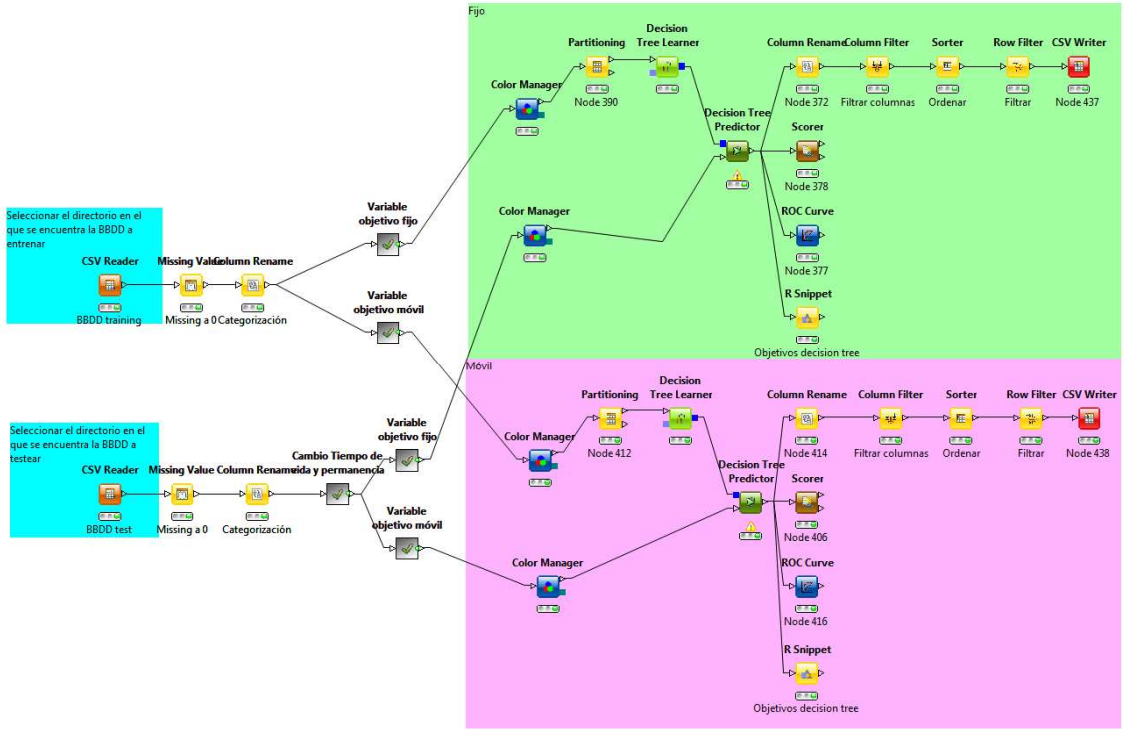


Figura 42: Flujo predicción.

6. Líneas futuras

6.1. Nuevas variables

Mensualmente se recibirán las bases datos actualizadas y además se dispondrá de nuevas bases de datos a partir de las cuales se podrán obtener nuevas variables que pueden ser de utilidad para la mejora de los resultados. Estas nuevas variables pueden aparecer en los árboles de decisión dando mayor número de reglas de asociación de probabilidad de baja.

6.2. Nuevo modelo

En el presente trabajo se han utilizado las variables de entrenamiento creadas en el primer trimestre para dar respuesta a la variable objetivo de ese mismo trimestre y se ha aplicado la información adquirida con la técnica de clasificación seleccionada a las variables del segundo trimestre.

Dado que el objetivo es predecir las bajas futuras, en el modelo de entrenamiento modificaremos la variable objetivo del primer trimestre por la del segundo. De esta forma, al aplicar el modelo a las variables del segundo trimestre obtendremos la predicción de la variable objetivo para los tres meses siguientes.

Sean $x_1, x_2, \dots, x_n$ las variables creadas, y la variable objetivo y t la imagen generada de las variables al seleccionar la ventana temporal con los últimos datos recibidos, podremos describir el modelo como sigue:

$$y_t = x_{1,t-3} + x_{2,t-3} + \cdots + x_{n,t-3}$$
$$\hat{y}_{t+3} = x_{1,t} + x_{2,t} + \cdots + x_{n,t}$$

Entendiendo por la suma de las variables a la información adicional adquirida al incluir, o no, cada una de las variables para la clasificación de la variable objetivo asociada al individuo.

6.3. Clases desbalanceadas

Otro aspecto importante para la mejora de resultados puede ser el tratamiento de clases desbalanceadas. El problema de clases desbalanceadas ocurre cuando el número de instancias pertenecientes a cada clase es muy diferente, lo que provoca que los clasificadores tengan gran exactitud para calcular modelos sobre la clase mayoritaria pero poca sobre los datos de la clase minoritaria.

En nuestro caso en particular estamos interesados en la clase minoritaria de la variable objetivo. Esta no supera el 3%, y al aplicar el árbol de decisión se obtienen pocas ramas, pocos nodos finales y por consiguientes pocas reglas a aplicar, ya que a medida que se divide la población menor es el tamaño sobre el que buscar una variable que aporte importancia, en términos de clasificación, al árbol.

Existen distintas propuestas para resolver la problemática de construcción de modelos de clasificación a partir de datos no balanceados. Algunas propuestas afectan a los algoritmos de clasificación y otras a los datos. Algunas de estas propuestas son: asignar un coste diferencial a las instancias de entrenamiento según las frecuencias de las clases; muestrear el conjunto de datos original, ya sea agregando casos sintéticos o repetidos de la clase minoritaria, o eliminando casos de la clase mayoritaria [24].

De las distintas propuestas se hará *oversampling*, que consiste en balancear la distribución de las clases añadiendo ejemplos de la clase minoritaria. En concreto, se cuadruplicarán cada uno de los registros que tengan el valor de la clase minoritaria. La selección de este número no ha sido arbitraria, sino que ha sido como consecuencia de una búsqueda del número de registros predichos aceptable para hacer campañas de fidelización y retención.

6.4. Presentación de resultados

Los resultados presentados hasta ahora están guardados en ficheros '.csv' que contienen la probabilidad de darse de baja asociada a cada cliente. Estos registros están ordenados de mayor a menor probabilidad para que se hagan campañas de fidelización y retención sobre aquellos que tienen más posibilidades de abandonar la compañía.

Con el fin de dar mayor información de cada uno de los registros, se añadirá la regla de clasificación que aplica el árbol de decisión para la asignación de una probabilidad a un cliente, aparte de otras variables especificadas por decisión del cliente.

Dar la regla de clasificación es de gran utilidad, de esta manera no sólo sabremos la probabilidad que tiene un cliente de darse de baja sino también los motivos por los que se le asigna. Esto es de gran valor para diseñar campañas, ya que se puede seleccionar a un tamaño considerable de clientes con alta probabilidad y misma regla y hacer una política de retención o fidelización individualizada y concreta.

De esta forma y de manera directa se reducen: los costes de formación, ya que ésta se centraría en la campaña en sí y no se haría de manera genérica, reduciéndose de esta forma el tiempo de formación; los costes de personal, reduciéndose el número de personas a las que llamar; los costes de gestión, planificación y desarrollo de campañas.

De manera indirecta se aumentan los beneficios, ya que hay una mayor aceptación de campañas cuando esta es personalizada al saber los motivos por los que un cliente está descontento y conociendo que se le puede ofrecer. De este modo se estará interviniendo en el ciclo de vida de un cliente, intentando que pase de ser posible baja a un cliente leal, Figura 1.

7. En la actualidad

Actualmente el trabajo presentando se lleva a cabo con 18 bases de datos, dando como resultado una base de datos global con 300000 registros y 60 variables. A medida que hemos adquirido mayor información y se han mejorado alguno de los aspectos mencionados en el apartado anterior se han obtenido mejores resultados.

Los aspectos más destacados que han propiciado el aumento del índice de acierto obtenido han sido: la inclusión de nuevas variables de interés obtenidas a partir de nuevas bases de datos proporcionadas, la modificación del modelo de entrenamiento para la obtención de las predicciones futuras y el tratamiento de clases desbalanceadas.

El descenso que se ha producido en el porcentaje de bajas en el primer trimestre de 2015 ha sido del 25%. Antes de que se realizasen acciones de fidelización y retención se tenía un 45% de precisión en productos de fijo y un 80% en productos de móvil, y se ha pasado a un 20% de precisión una vez realizadas.

Esa diferencia se traduce en beneficios, debido a que son clientes que permanecen en la compañía tras la realización de las campañas promovidas, en parte, por las reglas asociadas a los clientes incluidas en los resultados.

Para visualizar de forma dinámica e interactiva estas reglas en navegadores web se ha utilizado la librería de *JavaScript*, D3.js[9], para la construcción del *sunburst* mostrado a continuación.

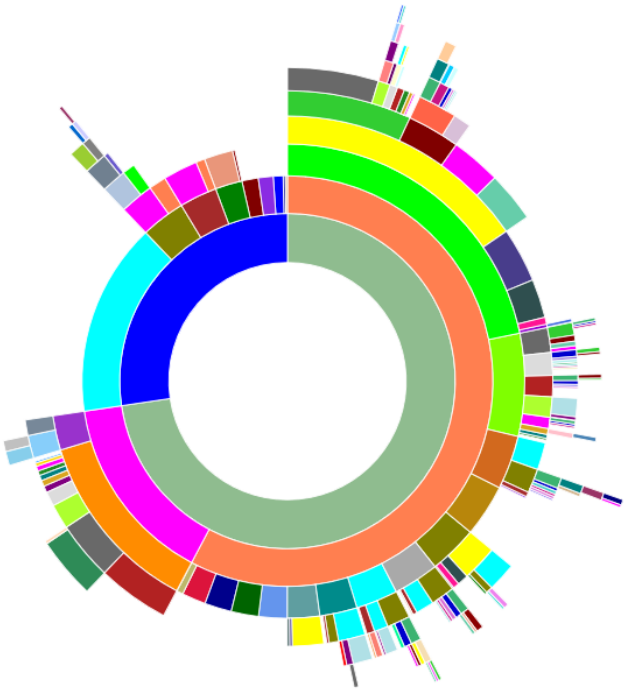


Figura 43: *Sunburst* general.

Cada trapecio circular representa cada uno de los nodos finales que proporciona el árbol de decisión. Cada vez que se selecciona un trapecio circular aparece la regla secuencial para acceder a ese nodo y la probabilidad de baja en ese nodo.

[9] http://d3js.org/

8. Anexo

Código R incluido en el nodo *R View Table*, comentado en el apartado 5.6.2.

```
library(pROC)

curva1 <- roc(knime.in$"Baja objetivo",knime.in$"Tree Ensemble",levels=c("SI","NO"))
plot(curva1,col="red",legacy.axes=TRUE)
p1<-coords(curva1, "best")
p1<-data.frame(p1[2],p1[3])
points(p1,col="red",pch=15,cex=2)
points(knime.flow.in[["Tree Emsemble Specifity"]],knime.flow.in[["Tree Emsemble Sensitivity"]],
col="red",pch=16,cex=2)
points(knime.flow.in[["Tree Emsemble Specifity (F1 score)"]],knime.flow.in[["Tree Emsemble
Sensitivity (F1 score)"]],col="red",pch=17,cex=2)

curva2 <- roc(knime.in$"Baja objetivo",knime.in$"Tree",levels=c("NO","SI"))
plot(curva2, add=TRUE,col="orange",legacy.axes=TRUE)
p2<-coords(curva2, "best")
p2<-data.frame(p2[2],p2[3])
points(p2,col="orange",pch=15,cex=2)
points(knime.flow.in[["Tree Specifity"]],knime.flow.in[["Tree Sensitivity"]],col="orange",pch=16,cex=2)
points(knime.flow.in[["Tree Specifity (F1 score)"]],knime.flow.in[["Tree Sensitivity (F1 score)"]],
col="orange",pch=17,cex=2)

curva3 <- roc(knime.in$"Baja objetivo",knime.in$"NaiveBayes",levels=c("NO","SI"))
plot(curva3, add=TRUE,col="green",legacy.axes=TRUE)
p3<-coords(curva3, "best")
p3<-data.frame(p3[2],p3[3])
points(p3,col="green",pch=15,cex=2)
points(knime.flow.in[["Naive Specifity"]],knime.flow.in[["Naive Sensitivity"]], col="green",
pch=16,cex=2)
points(knime.flow.in[["Naive Specifity (F1 score)"]],knime.flow.in[["Naive Sensitivity (F1
score)"]],col="green",pch=17,cex=2)

curva4 <- roc(knime.in$"Baja objetivo",knime.in$"Boosting",levels=c("NO","SI"))
plot(curva4, add=TRUE,col="brown",legacy.axes=TRUE)
p4<-coords(curva4, "best")
p4<-data.frame(p4[2],p4[3])
points(p4,col="brown",pch=15,cex=2)
points(knime.flow.in[["Boosting Specifity"]],knime.flow.in[["Boosting Sensitivity"]],
col="brown",pch=16,cex=2)
points(knime.flow.in[["Boosting Specifity (F1 score)"]],knime.flow.in[["Boosting Sensitivity (F1
score)"]],col="brown",pch=17,cex=2)

curva5 <- roc(knime.in$"Baja objetivo",knime.in$"KNN",levels=c("NO","SI"))
plot(curva5, add=TRUE,col="yellow",legacy.axes=TRUE)
p5<-coords(curva5, "best")
p5<-data.frame(p5[2],p5[3])
points(p6,col="yellow",pch=15,cex=2)
points(knime.flow.in[["KNNSpecifity"]],knime.flow.in[["KNNSensitivity"]],col="yellow",pch=16,
cex=2)
points(knime.flow.in[["KNN Specifity (F1 score)"]],knime.flow.in[["KNN Sensitivity (F1 score)"]],
col="yellow",pch=17,cex=2)

legend("bottomright",c("TreeEnsemble","Tree","NaiveBayes","Boosting","KNN","Closest.topleft","Preci
sión","F1
score"),col=c("red","orange","green","brown","yellow","black","black","black"),lty=c(1,1,1,1,1,0,0,0),pc
h=c(15,15,15,15,15,15,16,17))
```

9. Referencias

[1] Mattison, R. (2005). *The telco churn management handbook*. Oakwood Hills: XiT Press. pp. 33-61.

[2] Sterne, J y Cutler, M. (2000). *E-Metrics: Bussiness Metrics For The New Economy*. Cambrige, MA: NetGenesis Corp. pp. 26-33.

[3] CNMC. (2014). Informe económico de las telecomunicaciones y del sector audiovisual 2014. pp. 27-99.

[4] CNMC. CNMCData - Informe Anual 2014.
URL: http://data.cnmc.es/datagraph/jsp/inf_anual.jsp.

[5] KDnuggets. CRISP-DM, still the top methodology for analytics, data mining, or data science projects.
URL: http://www.kdnuggets.com/2014/10/crisp-dm-top-methodology-analytics-data-mining-data-science-projects.html.

[6] SPSS. (2000). CRISP -DM 1.0. pp. 10-12.

[7] Tsiptsis, K. and Chorianopoulos, A. (2009). *Data mining techniques in CRM*. Chichester, West Sussex, U.K.: Wiley. pp. 10-13, 33-34.

[8] KDnuggets. What analytics, data mining, data science software / tools you used in the past 12 months for a real project poll.
URL: http://www.kdnuggets.com/polls/2014/analytics-data-mining-data-science-software-used.html.

[9] Rexer, K. (2013). 2013 data miner survey. pp. 7, 9.

[10] Centers for disease control and prevention. (2010). 2010 Report: Open source data mining software evaluation. p. 2.

[11] KNIME. KNIME Quickstart Guide. pp. 11-15.

[12] Bakos, G. (2013). *KNIME essentials*. Birmingham, U.K.: Packt Publishing Ltd.

[13] Han, J., Kamber, M., y Pei, J. (2011). *Data mining: concepts and techniques*. Amsterdam: Elsevier. pp. 1-14, 327-383, 422-444.

[14] Olson, D. L., y Delen, D. (2008). *Advanced data mining techniques*. Lincoln, NE: Springer. pp. 3-19, 137-139, 144-147.

[15] Aggarwal, C. C. (2014). *Data classification: algorithms and applications*. Boca Ratón, FL: CRC Press. pp. 23, 67-71, 291-292, 296.

[16] Breiman, L., Friedman, J., Stone, C. J., y Olshen, R. A. (1984). *Classification and regression trees*. Boca Ratón, FL: CRC press.

[17] Quinlan, J. (1993*). C4.5. Programs for machine learning*. San Mateo, CA: Morgan Kaufmann Publishers.

[18] Shafer, J., Agrawal, R., y Mehta, M. (1996). SPRINT: A scalable parallel classifier for data mining. pp. 544-555.

[19] Lemmens, A., y Croux, C. (2006). Journal of Marketing Research. pp. 276-286.

[20] Friedman, J., Hastie, T., y Tibshirani, R. (2000). *Additive logistic regression: a statistical view of boosting*. pp. 337-407.

[21] Corzo, G. A., y Hong, L. Ensemble of model trees and artificial neural networks models for forecasting.

[22] Swets, J. A. (1988). Measuring the accuracy of diagnostic systems. Science, 240(4857), pp. 1285-1293.

[23] Hashmi, N., Butt, N. A., y Iqbal, M. (2013). Customer churn prediction in telecommunication.

[24] Laza, R., & Pavón, R. (2010). Clasificador bayesiano de documentos medline a partir de datos no balanceados.