Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

Trabajo de Fin de Grado

Minería de datos:

Predicción de las devoluciones de un distribuidor de moda

Data Mining:

Predicting return rates for a fashion distributor

Eduardo José Díaz Hernández

07/2021

D. **Jesús Manuel Jorge Santiso**, con N.I.F. 42.097.398-S profesor Titular de Universidad adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutor

C E R T I F I C A

Que la presente memoria titulada:

“Predicción de las devoluciones de un distribuidor de moda”

ha sido realizada bajo su dirección por D. **Eduardo José Díaz Hernández**,

con N.I.F. 43.388.478-J.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 6 de Septiembre de 2021

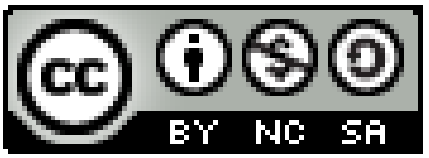
Agradecimientos

En primer lugar quiero agradecer a mis padres y familiares más cercanos quienes han sido las personas que me han apoyado y dado libertad para elegir los estudios que más me han motivado y además apoyarme en el transcurso de los mismos.

También quiero agradecer a todos los compañeros y amigos que han formado parte de este camino, ya que también han sido un apoyo para superar las situaciones más difíciles y hacer más soportable todo lo que ha ido sucediendo.

Y por último pero no menos importante dar las gracias a todo el profesorado que ha aportado un granito de arena con sus conocimientos para desarrollar las habilidades necesarias para poder seguir adelante y completar este camino.

Licencia



Resumen

En este proyecto se pretende realizar predicciones de la tasa de artículos devueltos en una empresa textil, para ello se han creado diferentes modelos con el uso de diferentes técnicas de minería de datos y aprendizaje automático usando la herramienta KNIME.

Una vez creados los modelos se pretende seleccionar aquel que realice el mejor trabajo de predicción.

Palabras clave: Minería de datos, Aprendizaje automático, KNIME, Tasa de retorno de artículos.

Abstract

This project aims to make predictions of the rate of returned items in a textile company, for this, different models have been created with the use of different data mining and machine learning techniques using the KNIME tool.

Once the models have been created, it is intended to select the one that performs the best prediction work.

Keywords: Data Mining, Machine Learning, KNIME, Article Return Rate.

Índice general

Índice de figuras

1.- Introducción

Con el avance de la tecnología han surgido diferentes maneras de trabajar con la gran inmensidad de datos que se almacenan actualmente, un ejemplo claro puede ser el cual vamos a tratar a lo largo de este Trabajo de Fin de Grado para resolver un problema el cual puede aplicarse a casos reales.

En este trabajo vamos a resolver un problema extraído de la competición llamada "Data Mining Cup" acerca de las tasas de devolución en una empresa textil, haciendo uso de diferentes atributos que forman parte del conjunto de datos siguiente teniendo en cuenta el siguiente escenario:

"Un distribuidor de moda vende artículos de tamaños y colores particulares a sus clientes. En algunos casos, los artículos se devuelven al distribuidor por diversas razones y los datos del pedido y de la devolución relacionados se registran durante un período de dos años. El objetivo es utilizar estos datos y métodos de minería de datos para construir un modelo que permita una buena predicción de las tasas de retorno."

1.1.- Antecedentes y estado del arte

Este problema ha sido tratado previamente en la literatura debido a la importancia del mismo. A continuación, hemos seleccionado una serie de artículos que tratan la temática:

1.1.1.- Aprovechar el poder de las imágenes para predecir las tasas de devolución de productos [1]

Basándonos en los datos de ventas y devoluciones de una gran marca de ropa, demostramos que la demanda de productos por parte de los consumidores difiere por compras online y offline, de modo que algunos productos alcanzan cuotas de mercado más altas cuando se compran en línea o al contrario. Además, controlando la demanda en línea, los productos con menor demanda offline tienen tasas de devolución más altas. Esta relación no se debe simplemente a la falta de información sobre los atributos de contacto del producto en línea. Se demuestra que un algoritmo de aprendizaje automático puede predecir con precisión la tasa de retorno utilizando la imagen del producto, que los consumidores ven en línea. Se extraen varias características visuales utilizando técnicas de procesamiento de imágenes y se predicen las tasas de retorno utilizando un modelo de árbol de regresión potenciado por gradientes. La incorporación de datos de imágenes mejora la precisión predictiva de los modelos hasta en un 37%, por lo que las empresas pueden utilizar este enfoque para pronosticar mejor la rentabilidad de un producto individual y hacer que la comercialización y la venta al por menor sean efectivas.

1.1.2.- Gestión de devoluciones de productos: el papel de la previsión [2]

En este artículo se analiza las formas de influir activamente en las devoluciones de productos y se revisa los métodos basados en datos para pronosticar los flujos de rentabilidad que explotan el hecho de que las devoluciones futuras son una función de las ventas pasadas. En particular, se evalúa el valor de la previsión de rendimiento a un nivel operativo, específicamente el control de inventarios para concluir con implicaciones para la gestión de la cadena de suministro.

1.1.3.- Predecir el volumen de devolución de productos mediante métodos de aprendizaje automático [3]

En 2015, los consumidores estadounidenses devolvieron bienes por valor de 261.000 millones de dólares y las tasas de devolución de las ventas en línea a veces superaron el 30%. Los fabricantes y minoristas tienen interés en predecir el volumen de devoluciones para abordar los desafíos operativos en la gestión de devoluciones de productos. En este documento, se desarrollan y prueban modelos basados en datos para predecir el volumen de devolución, el tipo de producto y los niveles de período utilizando un rico conjunto de datos compuesto por operaciones detalladas de cada producto e información del minorista.

1.2.- Objetivos

1.2.1.- Objetivos generales

Los objetivos generales que se persiguen a la hora de realizar este Trabajo de Fin de Grado son:

- Ponerse en contacto con problemas de predicción del mundo real.
- Aprender y entender el funcionamiento de las herramientas de minería de datos a
- Aprender a utilizar diferentes herramientas que son específicas para tratar problemas de minería de datos.
- Crear modelos para la resolución de problemas numericos y evaluar los mismos.
- Aprender y entender el funcionamiento de las principales técnicas de minería de datos.

1.2.2.- Objetivos específicos

Los objetivos que se persiguen a la hora de elaborar este Trabajo de Fin de Grado son los siguientes:

- Crear un conjunto de modelos con los que se hará una comparación para comprobar cuál cumple mejor con los objetivos.
- Elaborar un conjunto de gráficas para hacer un análisis preliminar de los datos con los que se está trabajando.
- Evaluar y analizar los resultados finales y extraer la información final de los modelos.

1.3.- Estructura de la memoria

En esta memoria del Trabajo de Fin de Grado se va a desarrollar lo que abarca la minería de datos y las técnicas más importantes dentro de este campo, además de cómo se realiza un proyecto de minería de datos.

Se ha estructurado de la siguiente manera:

Un primer capítulo de introducción donde se explica brevemente el problema a resolver, el estado del arte y los objetivos que se plantean para el desarrollo del Trabajo de Fin de Grado.

Un segundo capítulo donde se presentan los conceptos básicos de la minería de datos y las técnicas más habituales relativas a la predicción numérica.

Otro capítulo donde se analizan diferentes herramientas que pueden ser utilizadas a la hora de desarrollar un proyecto de minería de datos y donde se estudia más en profundidad la herramienta con la que se ha trabajado a la hora de realizar el proyecto.

Un cuarto capítulo donde comentamos en profundidad el desarrollo del proyecto realizado, detallando cada una de las fases que conforman el mismo. Obteniendo diferentes modelos y analizando los mismos para comprobar cual de ellos tiene el mejor comportamiento para nuestro caso particular.

Por último un capítulo de conclusiones donde se explican los resultados y cómo puede trabajarse este problema en un futuro.

2.- Minería de datos

En este capítulo se detalla un poco acerca del campo en el que estaremos trabajando a los largo del Trabajo de Fin de Grado el cual es la minería de datos, además de explicar un poco más en profundidad algunas de las técnicas que existen y algunos ejemplos de los mismos.

La minería de datos es un campo de la estadística y las ciencias de la computación referido al proceso que intenta descubrir patrones en grandes volúmenes de conjuntos de datos haciendo uso de técnicas o métodos relacionados con la inteligencia artificial o aprendizaje automático.

Como se habla en el libro **"Introducción a la minería de datos" [4]** el concepto de minería de datos surge por la aparición de nuevas necesidades y gracias al potencial que tienen los datos en la sociedad actual, estos pasan de ser un producto a ser una materia prima que se debe explotar.

Y es que ha aumentado el volumen de información almacenada de manera digital siendo gran parte de ella información histórica que puede ser de utilidad para explicar acontecimientos del pasado, entender cómo afecta al presente y predecir lo que puede pasar en el futuro. Pero para poder sacar conclusiones hay que analizar dichos datos y esa acción se ha realizado de forma manual durante muchos años y debido al crecimiento exponencial que se ha experimentado esto en muchas ocasiones resulta imposible.

Es debido a que las técnicas se han ido quedando cada vez más obsoletas que han surgido conceptos nuevos como el almacén de datos o mas conocido como ***data warehouse***, esto no es más que un repositorio de fuentes heterogéneas de datos que se organizan mediante un esquema unificado que permite facilitar su análisis.

Los dos retos a los cuales se enfrenta la minería de datos son en primer lugar trabajar con grandes volúmenes de datos y en segundo lugar utilizar las técnicas adecuadas para analizar los mismos y poder extraer conocimiento novedoso y útil.

2.1.- Tipos de datos

Y para resolver los retos que se han nombrado necesitamos trabajar con diferentes tipos de datos que se pueden diferenciar en datos estructurados en bases de datos relacionales, otros tipos de datos estructurados o datos no estructurados provenientes de la web.

2.1.1.- Bases de datos relacionales

Estas bases de datos se basan en una colección de relaciones o tablas, cada una de ellas consta de un conjunto de atributos y diferentes tuplas o filas que representan los objetos que pertenecen a la tabla. Una de las principales características de las bases de datos relacionales es la existencia de un esquema asociado para que los datos tengan una estructura interna.

Es común para trabajar con estas tablas utilizar lenguajes de consulta de datos como puede ser SQL, pero hay muchas tecnicas de mineria de datos que no nos permiten trabajar con varias tablas simultáneamente y es por ello que cuando se requiere trabajar con atributos de diferentes tablas se combinan en una única para simplificar dicha tarea.

Como en una estructura de datos es bastante común encontrar diferentes tipos de datos, es común distinguir solo entre dos tipos a la hora de realizar técnicas de minería de datos los cuales son atributos numéricos o categóricos.

- Los **atributos numéricos** contienen valores enteros o reales.
- Los **atributos categóricos o nominales** contienen valores que reflejan un conjunto finito de categorías.

2.1.2.- Otros tipos de bases de datos

- **Bases de datos espaciales:** es un sistema administrador de bases de datos que maneja datos existentes en un espacio o datos espaciales. En este tipo de bases de datos es imprescindible establecer un cuadro de referencia para definir la localización y relación entre objetos.

- **Bases de datos temporales:** es un sistema de gestión de base de datos en el cual se implementa y trata con especial énfasis aspectos temporales

- **Bases de datos documentales:** está constituida por un conjunto de programas que almacenan, recuperan y gestionan datos de documentos o datos de algún modo estructurados.

  El concepto central de una base de datos orientada a documentos es el concepto mismo de Documento. Mientras cada implementación de base de datos orientada a documentos difiere en los detalles, en general todas ellas comparten el principio de que los documentos encapsulan y codifican datos o información siguiendo algún formato estándar.

- **Bases de datos multimedia:** almacenan imágenes, audio o video, es por ello que soportan objetos de gran tamaño.

2.1.3.- World Wide Web

La web es el repositorio más grande y diverso que puede existir, es por ello que hay una inmensidad de conocimiento que se puede extraer. Pero esto no es una tarea sencilla ya que muchos de los datos son no estructurados o en su defecto semi-estructurados. Es por todo esto que la minería web se organiza en tres categorías:

- **Minería del contenido:** se encarga de encontrar patrones de los datos de páginas web.
- **Minería de la estructura:** se encarga de entender los hipervínculos y URLs.
- **Minería del uso de usuario:** se encarga de entender el uso que hace un usuario sobre las páginas webs.

2.2.- Tipos de modelos

En función de la tarea realizada, los modelos de minería de datos pueden clasificarse en las siguiente categorías:

- **Modelos predictivos:** se encargan de estimar el valor de determinadas variables de la base de datos en función de otras.

- **Modelos descriptivos:** se encargan de identificar patrones que explican los datos y sirven para explorar las propiedades de los datos examinados.

Además, los modelos de minería de datos también se pueden clasificar como metodos eager o lazy en función de si construyen un modelo o no:

- **Método eager (anticipativo):** en este método de aprendizaje el sistema intenta construir un modelo o una función objetivo general a partir del conjunto de entrenamiento.

  La principal ventaja de este método es que la función objetivo se aproximará globalmente durante el entrenamiento por lo cual hace que requiera menos espacio. Mientras que la principal desventaja es que generalmente no puede proporcionar buenas aproximaciones en la función objetivo.

- **Método lazy (perezoso):** no se construye modelo. Las tareas de minería de datos encomendadas se realizan sobre la marcha para cada instancia teniendo en cuenta el conjunto de datos de entrenamiento.

  Las principales desventajas son que las tareas de minería de datos son más lentas al no contar con un modelo, los datos ruidosos pueden tener un efecto considerable en los resultados obtenidos y suelen ser más lentos de evaluar implicando generalmente un aumento de los costes computacionales.

2.3.- Proceso del descubrimiento de conocimiento

Uno de los términos más relacionados con la minería de datos es la extracción o descubrimiento de conocimiento en bases de datos, también conocido como *Knowledge Discovery in Databases* o **KDD**. Muchas veces se usan ambos términos para referirse a un proceso que consta de diferentes fases siendo la minería de datos únicamente una fase de este proceso.

Las propiedades deseables del conocimiento extraído en este proceso son:

- **Válido:** los patrones deben ser precisos para datos nuevos.
- **Novedoso:** se debe aportar algo desconocido.
- **Potencialmente útil:** la información debe aportar algún tipo de beneficio.
- **Comprensible:** en caso contrario puede dificultar o en algunos casos imposibilitar la interpretación, revisión, validación y toma de decisiones.

Es por esto el KDD es un proceso complejo que no solo incluye la obtención de un modelo, sino la evaluación e interpretación de los mismos como se puede ver en la imagen 2.1

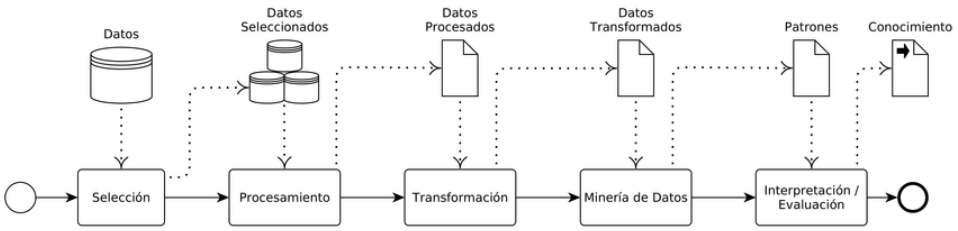


Imagen 2.1: Proceso KDD

2.3.1.- Fases del proceso de extracción del conocimiento

2.3.1.1.- Fase de integración y recopilación

En esta fase se reconocen y se reúnen los datos con los que se trabajará en el futuro. Estos datos pueden ser recogidos de dos tipos de fuentes diferenciadas:

- **Fuentes internas:** son las más fáciles de obtener y representan las bases de datos transaccionales propias de la organización utilizadas tradicionalmente para el procesamiento transaccional en línea. Aunque pueden no ser suficientes para otras funciones más complejas.

- **Fuentes externas:** representan la información que no pertenece al ámbito de las operaciones de la empresa pero que son importantes para el análisis.

Es por ello que es importante señalar la utilización de *data warehouses* o almacenes de datos debido a que nos permite coleccionar datos de las bases de datos transaccionales y otras fuentes accesibles para el análisis y toma de decisiones.

Un almacén de datos es un conjunto de datos históricos, internos o externos, y descriptivos de un contexto o área de estudio, que están integrados y organizados de tal forma que permiten aplicar eficientemente herramientas para resumir, describir y analizar los datos con el fin de ayudar en la toma de decisiones. Su propósito es la recuperación de información, realizar análisis e informes.

Y aunque estos almacenes de datos son muy aconsejables de usar no son imprescindibles.

El objetivo de esta fase es crear un almacén de datos que contenga la información que será utilizada. En ocasiones durante la construcción de este almacén de datos también se realizan tareas propias de limpieza e integración propias de la siguiente fase que aunque no sean dos fases muy diferenciadas es conveniente visualizarlas individualmente.

2.3.1.2.- Fase de selección, limpieza y transformación

Esta fase tiene como objetivo la mejora en la calidad del almacén de datos y trata de mostrar el conjunto de datos de la manera más apropiada para la siguiente fase: la minería de datos.

Extraer la información de diversas fuentes da lugar a diversos tipos de imperfecciones que será necesario resolver debido a que mientras mayor sea el

número de imperfecciones en el almacén la calidad del mismo será menor, a esto es lo que se le denomina limpieza.

Un tipo de datos sobre los que hay que efectuar tareas de limpieza son los datos irrelevantes o innecesarios. Estos aparecen cuando al hacer la recopilación de datos útiles, estos están acompañados de otros que no se incluyen en el dominio del problema.

También existen valores que no se adecuan al comportamiento general de los datos, los denominados valores outliers. Estos valores no tienen porqué representar siempre un error, sino que puede tratarse simplemente de un dato fuera de lo común. La mayoría de las veces estos valores van a hacer que el modelo se distorsione pero en otras ocasiones estos valores son justamente los que serán interesantes en algún aspecto es por ello que no es recomendable eliminar los valores outliers.

Otro tipo de datos que pueden llegar a presentar algún tipo de problema son los valores faltantes o *missing values*. Estos son datos que por algún motivo no existen, ya sea porque se han perdido o porque nunca se han llegado a recopilar, además también se reconocen como datos faltantes los valores que han sido introducidos para informar de que tal valor no existe.

Con el fin de que la minería de datos sea más acertada, además de tratar con estos valores que representan un problema se requiere realizar una transformación de algunos valores debido a que puede ocurrir que distintas fuentes de donde se han extraído los valores han podido almacenarlos utilizando diferentes formatos.

2.3.1.3.- Fase de minería de datos

Esta fase es la más característica del KDD y es por ello que muchas veces se le suele utilizar este nombre para nombrar todo el proceso. Su objetivo es producir nuevo conocimiento y para ello es necesaria la creación de un modelo basado en los datos recopilados con anterioridad.

Esta fase es muy amplia al contar con diferentes técnicas y por ello se ampliará más adelante.

2.3.1.4.- Fase de evaluación e interpretación

Una vez obtenido el conocimiento y el patrón resultante de la fase de minería, el siguiente paso es evaluar cómo ese conocimiento se aproxima a la realidad en cuanto a las metas definidas en la primera fase.

La evaluación se realiza sobre dos conjuntos de datos, uno de ellos se denomina *training-set* o conjunto de entrenamiento y el otro se denomina *test-set* o conjunto de prueba. El conjunto de entrenamiento será el utilizado para construir el modelo y el conjunto de prueba se utilizará para aplicar el modelo obtenido y comparar los resultados. Esta separación en dos conjuntos es necesaria ya que si se creara el modelo con el mismo conjunto con el que se realiza la prueba, el modelo resultante utilizará la misma información que ha sido utilizada para su construcción en la prueba, obteniendo estimaciones muy optimistas.

Existen diferentes tipos o formas de evaluación:

- **Validación simple:** la forma más básica, consiste en repartir aleatoriamente las observaciones disponibles en dos grupos, uno de ellos se emplea para entrenar al modelo y el otro para evaluarlo.

- **Validación cruzada con n pliegues:** consiste en dividir los datos de forma aleatoria en n grupos de aproximadamente el mismo tamaño, n-1 grupos se emplean para entrenar el modelo y uno de los grupos se emplea como validación. Este proceso se repite n veces utilizando un grupo distinto como validación en cada iteración.

  El proceso genera n estimaciones del error cuyo promedio se emplea como estimación final.

- ***Bootstrapping*:** consiste en utilizar como conjunto de entrenamiento una selección de los datos originales obtenidos con muestreo por reemplazamiento y dejar el resto que no ha sido utilizado como conjunto de prueba.

Para la interpretación de los resultados obtenidos también debe tenerse en cuenta el contexto sobre el cual el modelo está trabajando, teniendo que contrastarse el conocimiento que nos proporciona con el previo.

2.3.1.5.- Fase de difusión, uso y monitorización

Una vez construido y validado el modelo se puede usar con dos finalidades: que un analista recomiende acciones basándose en el modelo o que se puede aplicar dicho modelo con diferentes conjuntos de datos.

Para ello es necesario que se haga una difusión del mismo y se distribuya todo lo necesario. Algo que también es importante una vez se haya distribuido es ver como evoluciona con el tiempo y ser monitorizado para realizar las modificaciones necesarias.

2.4.- Técnicas de minería de datos

Como se habló anteriormente en KDD hay una fase denominada minería de datos, en ella se aplican diferentes técnicas que se explicarán a continuación.

2.4.1.- Modelo de regresión

Se habla de un modelo de regresión cuando la variable de respuesta y las variables explicativas son todas ellas cuantitativas. Además si se cuenta solamente con una única variable explicativa se considera regresión simple, mientras que si se cuenta con varias sería regresión múltiple.

La función de regresión más simple es la regresión lineal, un ejemplo de la misma se puede ver en la imagen 2.2, en ella cada variable explicativa participa de forma aditiva y constante para todo el dominio.

El modelo de regresión lineal se escribe:

$$Y = \beta_0 + \beta_1 X_1 + \ldots + \beta_m X_m + \varepsilon$$

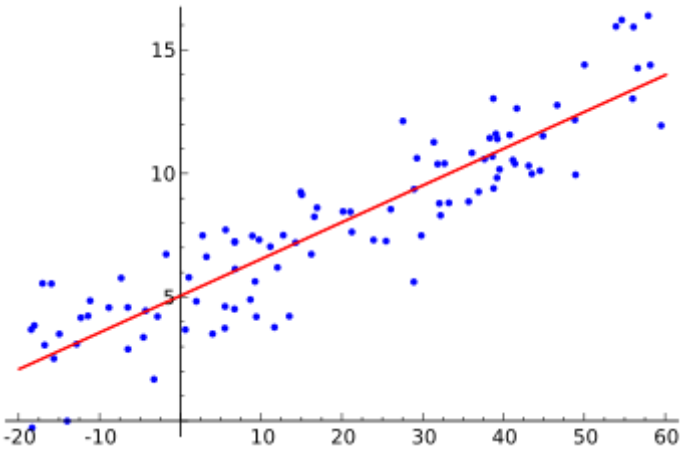


Imagen 2.2: Ejemplo regresión lineal

La suposición de linealidad puede parecer muy restrictiva pero si el número de individuos con el que se cuenta es pequeño los modelos lineales muestran una capacidad de generalización mayor a otros modelos más sofisticados. Además las variables explicativas pueden ser transformaciones de las originales dando mayor versatilidad al modelo.

También existen modelos no lineales que mediante simples transformaciones se pueden convertir en modelos lineales.

2.4.2.- Árboles de regresión

Este método o técnica puede ser considerado como el más fácil de entender y utilizar, un árbol de decisión es un conjunto de condiciones organizadas de tal manera que la decisión final se puede tomar siguiendo una serie de condiciones que se cumplen desde la raíz. De una forma más gráfica puede verse representado de una manera similar a la imagen 2.3.

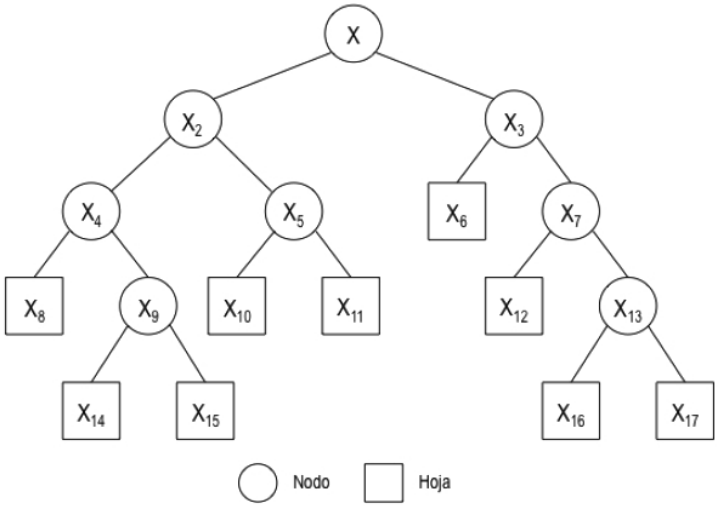


Imagen 2.3: Ejemplo de árbol de decisión

Una de las principales ventajas de este método es que las opciones posibles a partir de una determinada condición son excluyentes, esto permite analizar situaciones determinadas para realizar la toma de decisiones.

Nos centraremos en los árboles de decisión para regresión debido a que son los más relacionados con el problema a tratar en el Trabajo de Fin de Grado.

Lo que difiere a un árbol de regresión de un árbol de clasificación es que la función aprendida tiene un dominio real en vez de uno discreto y que las hojas del árbol tienen valores reales a los cuales se le puede maximizar o minimizar su valor.

Existen diferentes tipos de árboles de regresión entre ellos podríamos destacar los siguientes:

2.4.2.1.- Extreme Gradient Boosting (XGBoost)

El potenciador del gradiente o gradient boosting es una técnica de aprendizaje automático utilizado para el análisis de la regresión y para problemas de clasificación estadística. Se produce un modelo predictivo en forma de un conjunto de modelos de predicción débiles, típicamente árboles de decisión.

La idea básica utilizada para XGBoost es realizar una ordenación de las características según sus valores para luego atravesar los puntos de división con el objetivo de encontrar el mejor punto de segmentación. Una vez hecho esto los datos se dividen en nodos secundarios.

La ventaja de esto es que se pueden encontrar los puntos de segmentación con un gran nivel de precisión, pero consta de desventajas como la gran cantidad de espacio que consume o una sobrecarga notable en el tiempo.

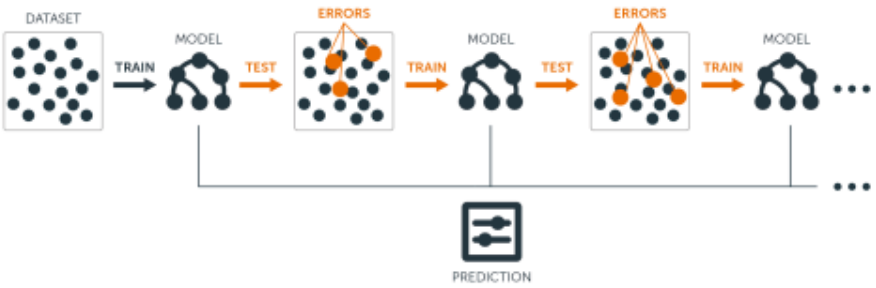


Imagen 2.4: Ejemplo gradient boosting

El XGBoost se convirtió en el método de referencia y un componente clave para obtener soluciones en una gran variedad de problemas dentro del aprendizaje automático.

2.4.2.2.- Light Gradient Boosting Machine (Light GBM)

Este es un sistema que se basa en algoritmos de árboles de decisión y se diferencia del anteriormente nombrado XGBoost en el uso de algoritmos basados en histogramas para que el entrenamiento sea más ágil y se reduzca el uso de memoria.

Mientras que en otras técnicas el árbol crece horizontalmente, el algoritmo LightGBM crece verticalmente, lo que significa que crece en forma de hojas donde se elige la hoja con grandes pérdidas para crecer.

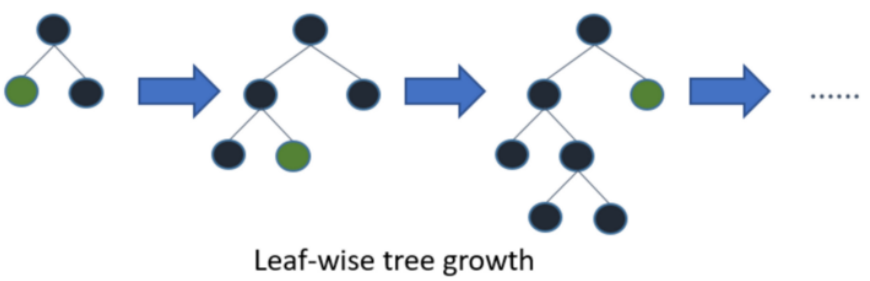


Imagen 2.5: Ejemplo LightGBM

2.4.3.- Redes Neuronales

Las redes neuronales son un método de aprendizaje cuya finalidad era emular los procesadores biológicos de información, estas parten de la idea de que la capacidad humana de procesar información se debe a la naturaleza biológica de de nuestro cerebro y por ello para imitar esta característica se debe estudiar al propio cerebro.

Una neurona artificial es el modelado de una neurona biológica como se ve en la imagen 2.6. Cada una de ellas tienen diferentes entradas a las que se le asocia un peso que se terminaran sumando para dar un valor resultante y este valor será el cual se transmita a la salida, estas entradas se representan mediante un vector x y los pesos con un vector w dando como resultado el valor de salida según:

$$y = f(\sum_i w_i x_i)$$

Donde f es la función de activación.

Cuando se tiene una red de neuronas las salidas de unas se conectan con las entradas de otras.

Normalmente, cada entrada se pesa por separado y la suma se transmite a través de una función no lineal conocida como función de activación o función de transferencia.

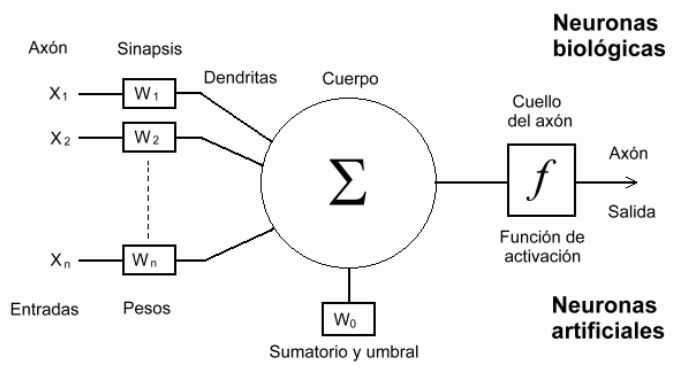


Imagen 2.6: Neurona artificial

Las unidades de procesamiento se organizan en capas. Hay tres partes normalmente en una red neuronal:

- **Capa de entrada:** que contiene unidades que representan los campos de entrada.

- **Capas ocultas:** pueden ser una o varias.

- **Capa de salida:** con una unidad o unidades que representan el campo o los campos de destino.

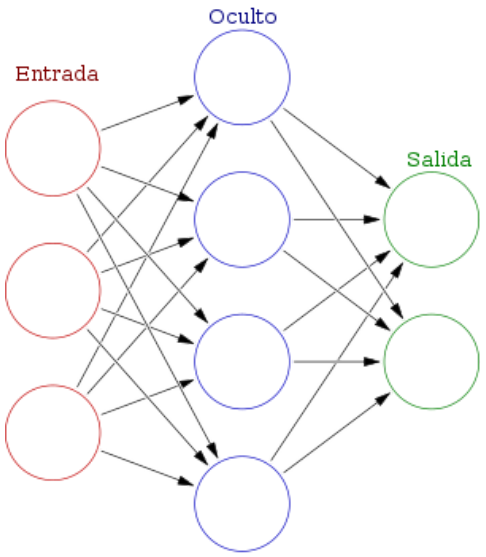


Imagen 2.7: Ejemplo red neuronal

Las unidades se conectan con fuerzas de conexión variables, los datos de entrada se presentan en la primera capa y los valores se propagan desde cada neurona hasta cada neurona de la siguiente capa y al final se envía un resultado desde la capa de salida.

La red aprende examinando los registros individuales, generando una predicción para cada registro y realizando ajustes a las ponderaciones cuando realiza una predicción incorrecta. Este proceso se repite muchas veces y la red sigue mejorando sus predicciones hasta haber alcanzado uno o varios criterios de parada.

Al principio, todas las ponderaciones son aleatorias y las respuestas que resultan no son del todo fiables debido a que la red aprende a través del entrenamiento. Continuamente se presentan a la red ejemplos para los que se conoce el resultado, y las respuestas que proporciona se comparan con los resultados conocidos. La información procedente de esta comparación se pasa hacia atrás a través de la red, cambiando las ponderaciones gradualmente. A medida que progresa el entrenamiento, la red se va haciendo cada vez más precisa en la replicación de resultados conocidos. Una vez entrenada, la red se puede aplicar a casos futuros en los que se desconoce el resultado.

Este continuo entrenamiento o aprendizaje se puede dividir en dos tipos:

- **Aprendizaje supervisado:** a la red se le proporciona un conjunto de datos de entrada y la respuesta correcta. El conjunto de datos es propagado hasta llegar a la capa de salida y se comparan los resultados obtenidos. Una vez hecho esto se ajustan los pesos para asegurar que la respuesta sea lo más correcta posible.

- **Aprendizaje no supervisado:** a la red solo se le proporciona un conjunto de datos de entrada, esta red se debe auto-organizar según el tipo de estructura existente en el conjunto de datos de entrada.

Dentro del aprendizaje supervisado podemos destacar dos redes neuronales, el perceptrón simple y el perceptrón multicapa.

2.4.3.1.- Perceptrón simple

La estructura de un perceptrón simple se basa en varios nodos de entrada y uno o más de salida y no consta de capas ocultas como se puede ver en la imagen 2.8.

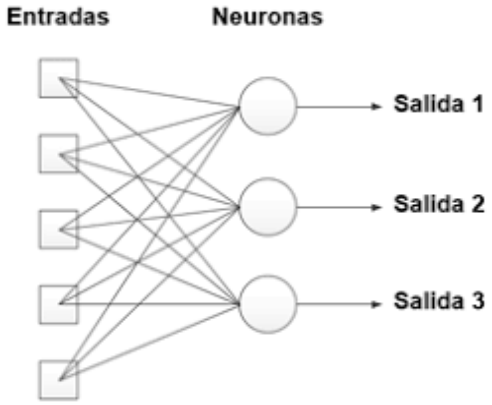


Imagen 2.8: Ejemplo perceptrón simple

Su funcionamiento se divide en varios pasos:

1. La red comienza en un estado aleatorio
2. Se selecciona un conjunto de entrada a partir del conjunto de entrenamiento introducido.
3. Se propaga la actividad para calcular la salida
4. Si la salida es correcta se vuelve al paso dos.
5. En caso de no ser la salida correcta se recalculan los pesos y se vuelve al paso dos.

2.4.3.2.- Perceptrón multicapa

En este caso la estructura de esta red neuronal es en cascada y tiene una o varias capas ocultas como se puede ver en la imagen 2.9. La activación se propaga en la red desde la capa de entrada y en las capas intermedias se aplica alguna función para luego dirigirse a la capa de salida.

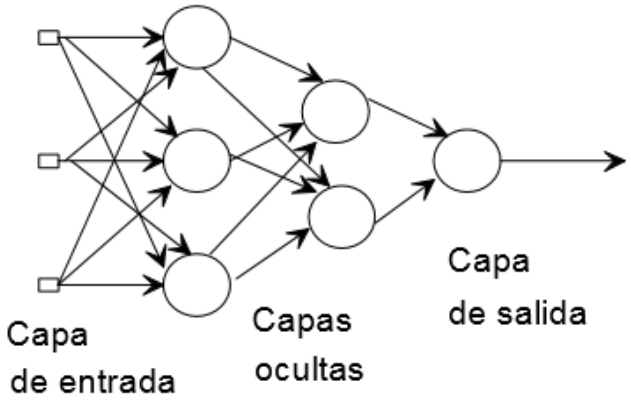


Imagen 2.9: Ejemplo perceptrón multicapa

Si pensamos en el aprendizaje hay que actualizar dos conjuntos de pesos, los que hay entre la capa de entrada y la intermedia, y los que hay entre la capa intermedia y la de salida.

2.4.4.- Métodos basados en vecindad

Una idea que también surge es pensar que ante una situación nueva se pueda actuar según como se ha hecho con anterioridad en situaciones similares si hubo éxito en las mismas.

Es por ello que aparecen dos conceptos importantes, el primero de ellos es el entendimiento de “similitud” o lo que es inversamente equivalente, matemáticamente hablando, a la “distancia”. Y la segunda es determinar cuándo se puede explotar dicha similitud.

Hay diferentes tipos de medidas de distancia, las más tradicionales son aquellas que se aplican sobre dos instancias que tengan todos sus atributos numéricos.

- **Distancia euclídea:** es la longitud de la recta que une dos puntos en el espacio euclídeo.

$$d(x,\ y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

- **Distancia de Manhattan o por cuadras:** se recorre el camino de manera zigzagueante.

$$d(x,\ y) = \sum_{i=1}^{n} |x_i - y_i|$$

- **Distancia de Chebychev:** se calcula la discrepancia más grande en alguna de las dimensiones.

$$d(x,\ y) = max_{i=1..n}|x_i - y_i|$$

- **Distancia del coseno:** considerando cada ejemplo como un vector, la distancia sería el coseno del ángulo que forman

$$d(x,\ y) = arccos\left(\frac{x^T y}{||x||\cdot||y||}\right)$$

2.4.4.1.- Método KNN (k-nearest neighbors)

El método de los k vecinos más cercanos es un método de clasificación supervisada, además es un método de clasificación no paramétrico, que estima el valor de la función de densidad de probabilidad o directamente la probabilidad a posteriori de que un elemento x pertenece a la clase C_j a partir de la información proporcionada por el conjunto de prototipos.

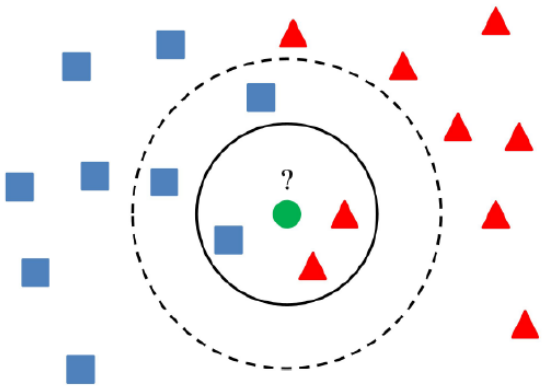


Imagen 2.10: Ejemplo KNN

En este método se determina para cada región la probabilidad de que un elemento situado en dicha región pertenezca a una de las clases existentes. Cuando un caso nuevo aparece se genera un círculo con centro en dicho punto y con un radio prefijado. Luego se calcula el número de ejemplos que caen dentro del círculo y se le etiqueta según la clase más numerosa.

Como se ve en la imagen 2.10 el radio de la circunferencia es determinante para elegir en qué clase va a ser etiquetado el nuevo caso, es por ello que la distancia puede ser un parámetro fundamental al utilizar este método.

A la hora de utilizar este método para regresión la salida en vez de ser la clase a la cual vamos a asignar, el valor es el promedio de los valores de los k vecinos más cercanos.

2.4.5.- Máquinas de soporte vectorial

Las máquinas de soporte vectorial son un conjunto de algoritmos de aprendizaje supervisado.

Estos métodos están propiamente relacionados con problemas de clasificación y regresión en los que dado un conjunto de ejemplos de entrenamiento podemos etiquetar las clases y entrenar una **SVM** (Support Vector Machine) para construir un modelo que prediga la clase de una nueva muestra.

Intuitivamente, una **SVM** es un modelo que representa a los puntos de muestra en el espacio, separando las clases a dos espacios lo más amplios posibles mediante un hiperplano de separación definido como el vector entre los dos puntos, de las dos clases, más cercanos al que se llama vector soporte. Cuando las nuevas muestras se ponen en correspondencia con dicho modelo, en función de los espacios a los que pertenezcan, pueden ser clasificadas a una o la otra clase.

Más formalmente, una **SVM** construye un hiperplano o conjunto de hiperplanos en un espacio de dimensionalidad muy alta que puede ser utilizado en problemas de clasificación o regresión. Si se realiza una buena separación entre las clases se podrá hacer una clasificación correcta.

Cuando se adapta esta técnica a modelos de regresión se denomina **SVR** o **Support Vector Regression** el cual utiliza el mismo principio pero con algunos cambios menores.

En principio dado que la salida es un número real, se vuelve difícil predecir la información a mano dado que existen posibilidades infinitas. La idea básica de una SVR consiste en realizar un mapeo de los datos de entrenamiento a un espacio de

mayor dimensión a través de un mapeo no lineal donde, posteriormente, se podrá realizar una regresión lineal

2.4.5.1.- Funcionamiento

Dado un conjunto de puntos en el que cada uno de ellos pertenece a una de dos posibles categorías, un algoritmo basado en SVM construye un modelo capaz de predecir si un punto nuevo pertenece a una categoría o a la otra.

La SVM busca un hiperplano que separe de forma óptima a los puntos de una clase de la de otra, que han podido ser previamente proyectados a un espacio de dimensionalidad superior.

Es en la “separación óptima” donde aparece la característica principal de una SVM, en ella se busca el hiperplano que tenga la máxima distancia con los puntos más cercanos del mismo. Así los puntos del vector que son etiquetados con una categoría estarán de una lado del hiperplano y los casos que se encuentren en otra categoría aparecerán en el otro lado del hiperplano.

Se le llama atributo a la variable predictora y característica a un atributo transformado que es usado para definir el hiperplano, la elección de la representación más adecuada se realiza mediante un proceso denominado selección de características y al vector formado por los puntos más cercanos al hiperplano se le llama vector de soporte.

Es importante también tener en cuenta que los modelos basados en SVM están estrechamente relacionados con las redes neuronales y pueden ser un método de entrenamiento alternativo para clasificadores polinomiales, funciones de base radial y perceptrón multicapa.

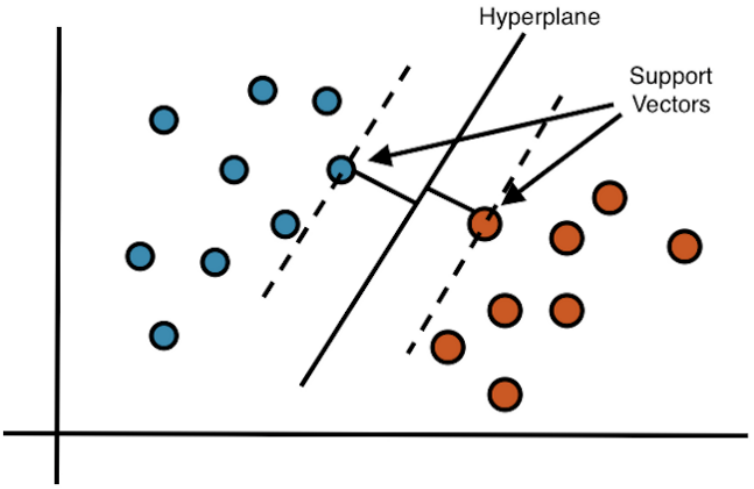


Imagen 2.11: Ejemplo SVM

2.4.5.2.- SVM lineal con margen máximo

Este es el modelo más sencillo e intuitivo, aunque a su vez, el que tiene mas condiciones de aplicabilidad más restringidas debido a que parte de la hipótesis de que el conjunto de datos es linealmente separable.

Hay que recordar que la idea que hay detrás de una SVM es seleccionar el hiperplano separador que está a la mínima distancia de un vector x y el hiperplano h de la forma:

$$dist(h,\ x)\ =\ |h(x)|\ /\ ||w||$$

Donde $||w||$ es la norma asociada al producto escalar y el hiperplano equidistante a dos clases es el que maximiza el valor mínimo de la distancia.

2.4.5.3.- SVM con margen blando

Debido a que no siempre se puede encontrar una transformación de los datos que permita separarlos linealmente surgen los SVM con margen blando que son más robustos a la hora de tratar con este tipo de problemas.

La idea principal es introducir variables de holgura que permitan que las restricciones no se cumplan de manera estricta, este vector de variables tendrá la misma dimensión que el conjunto de datos.

Cada componente indica la máxima diferencia posible entre un margen teórico y un margen real, el mínimo valor que podrá tener es cero. Se añade un término de regularización en la función a optimizar la cual incluye una constante que determina la holgura.

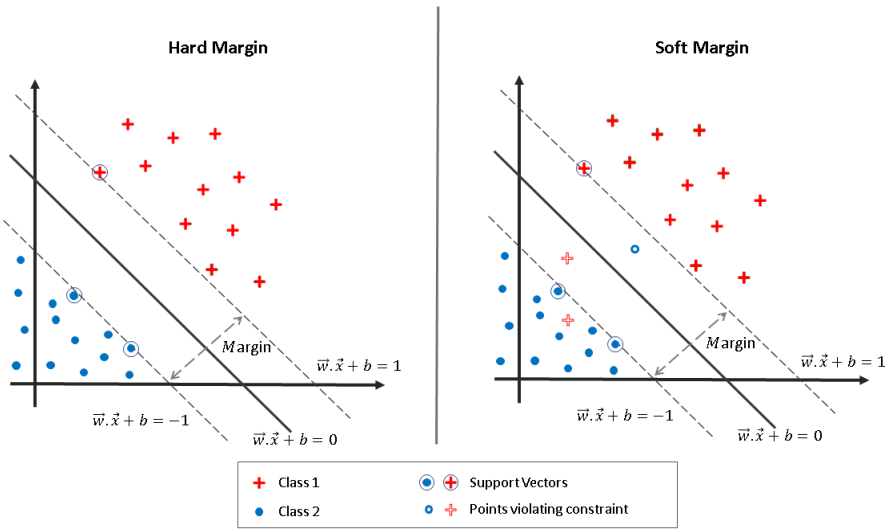


Imagen 2.12: Diferencia entre Hard Margin y Soft Margin

3.- Herramientas

Existen diferentes herramientas con las cuales se pueden aplicar las técnicas de minería de datos que se han nombrado en el capítulo 2, en este capítulo se nombran algunas de ellas y se ampliará la información de la herramienta que haya sido seleccionada para realizar el proyecto.

3.1 Herramientas disponibles

3.1.1.- KNIME [5]

Imagen 3.1: Logo KNIME

KNIME es una herramienta de minería de datos desarrollada sobre la plataforma Eclipse y programada en java. Es una herramienta gráfica que dispone de una serie de nodos los cuales encapsulan distintos tipos de algoritmos y flechas que funcionan como flujo de datos.

Los nodos implementan diferentes tipos de acciones que pueden ejecutarse sobre diferentes tablas de datos, entre estas acciones se encuentran:

- Manipulación de filas o columnas.
- Visualización de gráficas
- Creación de modelos estadísticos y de minería de datos.

- Validación de modelos
- Aplicación de dichos modelos sobre conjuntos de datos
- Creación de informes

Además al ser una herramienta abierta da la posibilidad de que se puedan crear nuevos nodos con algoritmos desarrollados en Python o R.

3.1.2.- WEKA [6]

Imagen 3.2: Logo WEKA

WEKA nos aporta una colección de herramientas de visualización y algoritmos para el análisis de datos y modelado predictivo, además de ofrecer una interfaz gráfica de usuario para acceder fácilmente a sus funcionalidades.

Las características de Weka son:

- Está disponible libremente bajo la licencia pública general de GNU.
- Al estar completamente implementado en Java es muy portable al poder correr en casi cualquier plataforma.
- Contiene una gran cantidad de técnicas para el procesamiento y modelado de datos.
- Su interfaz gráfica hace que sea fácil de utilizar.

Weka es capaz de realizar varias tareas estándar de minería de datos como puede ser el preprocesamiento de datos, clustering, clasificación, regresión o visualización.

3.1.3.- RapidMiner [7]

Imagen 3.3: Logo RapidMiner

Es un programa informático orientado al análisis y a la minería de datos que permite el desarrollo de procesos de análisis de datos mediante el encadenamiento de operadores a través de un entorno gráfico. RapidMiner nos ofrece más de 500 operadores orientados al análisis de datos, incluyendo los necesarios para realizar operaciones de entrada y salida, preprocesamiento de datos y visualización. Además nos permite utilizar los algoritmos incluidos en Weka.

Las características más importantes de RapidMiner son:

- Es multiplataforma y está desarrollado en Java.
- Permite el desarrollo de programas a través de scripts.
- Dispone de módulos de integración con lenguajes como R o Python.
- Incluye gráficos y herramientas de visualización de datos.

3.1.4.- Python [8]

Imagen 3.4: Logo Python

Python es un lenguaje de programación interpretado, dinámico y multiplataforma que mediante el uso de diferentes librerías estadística o numéricas (Pandas, Numpy, Matplotlib, etc.) o librerías de aprendizaje profundo como Tensorflow se pueden llevar a cabo grandes proyectos de minería de datos.

Las tareas que nos permite hacer estas librerías junto al lenguaje de programación son:

- Recopilación y limpieza de datos
- Exploración de datos
- Modelado de datos
- Visualización e interpretación de datos

Es por ello que es el lenguaje de programación más utilizado por profesionales del sector para llevar a cabo tareas que requieren trabajar con grandes cantidades de datos.

3.1.5.- R [9]

Imagen 3.5: Logo R

R es un entorno y lenguaje de programación con un enfoque al análisis estadístico que nos proporciona un amplio abanico de herramientas estadísticas (modelos lineales y no lineales, tests estadísticos, análisis de series temporales, algoritmos de clasificación y agrupamiento, etc.) y gráficas.

Además también se puede integrar con distintas bases de datos o mediante librerías se puede hacer uso de lenguajes de programación interpretados como Python o Perl.

3.2.- Entorno de KNIME

Finalmente la herramienta con la cual se lleva a cabo el desarrollo del proyecto es KNIME debido a que es una herramienta suficientemente potente para realizar el trabajo, cuenta con una interfaz intuitiva y además es una herramienta nueva para aprender a trabajar.

Esta decisión se tomó en base a la facilidad que ofrece la herramienta y al tener un diseño visual que se agradece a la hora de realizar las diferentes tareas de minería de datos.

3.2.1.- Interfaz de KNIME

La interfaz gráfica que se nos presenta es la que se muestra en la imagen 3.6, en ella podemos observar diferentes ventanas las cuales se van a explicar a continuación.

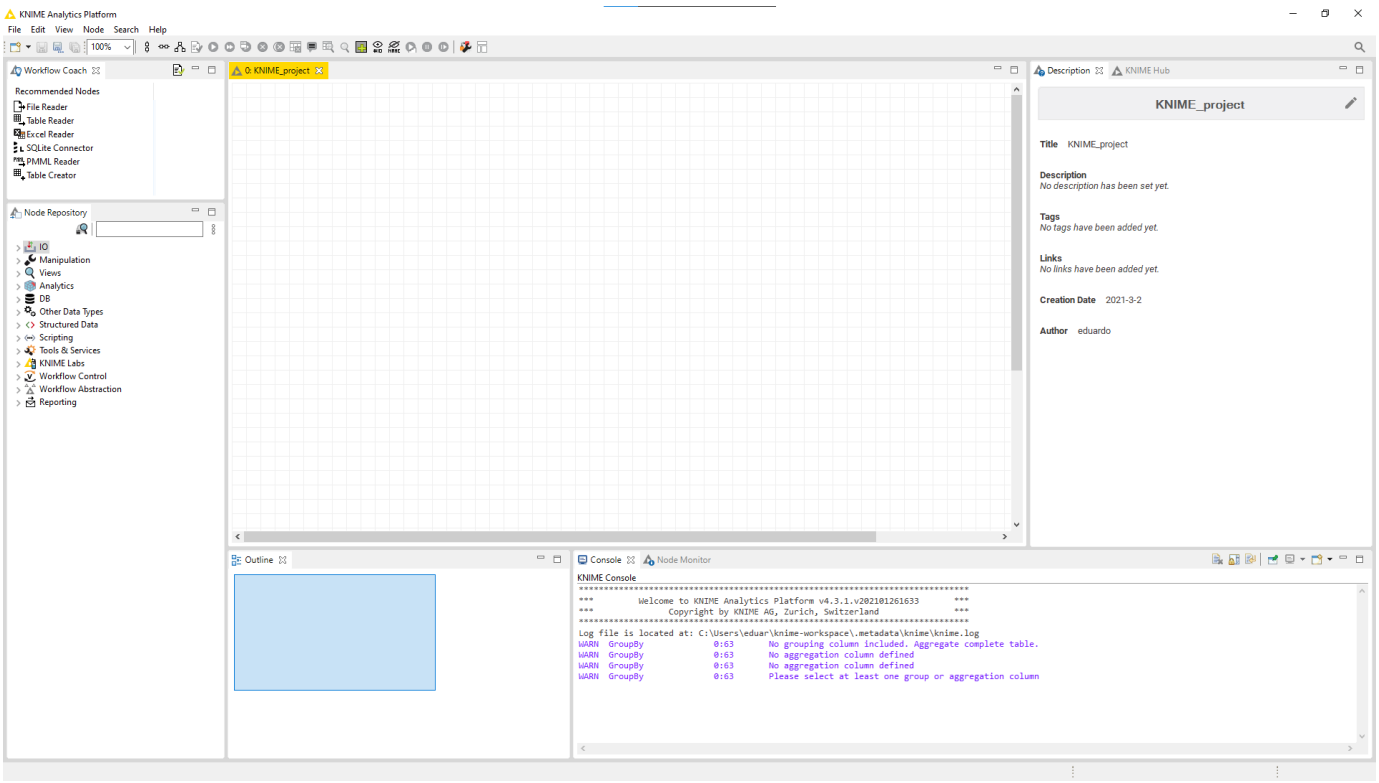


Imagen 3.6: Interfaz gráfica de KNIME

- **Workflow Coach:** en esta ventana se muestran los nodos recomendados para realizar diferentes acciones dependiendo del nodo que estamos seleccionando o en el caso de empezar un proyecto y no exista ningún nodo las opciones que nos recomienda son lectura de fichero o conexión a alguna base de datos existente como se ve reflejado en la imagen 3.7.

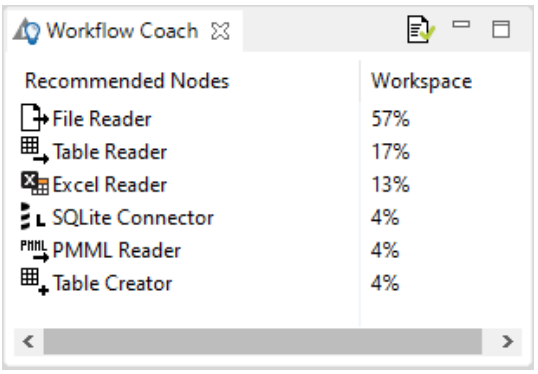


Imagen 3.7: Ventana Workflow Coach

- **Node Repository:** en esta ventana aparecen todos los nodos existentes por defecto en la herramienta, se agrupan en categorías para así estar mejor organizados y que al usuario le sea fácil encontrar los nodos que quiera

utilizar. Además cuenta con un buscador para ser más preciso en la búsqueda de los nodos deseados.

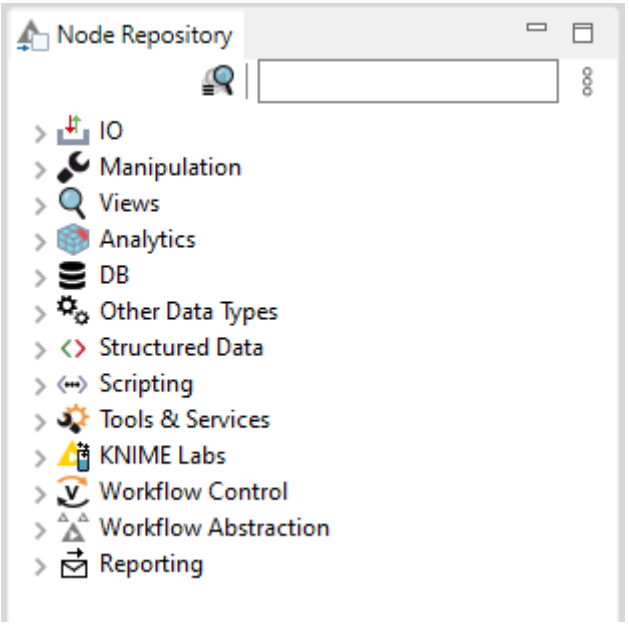


Imagen 3.8: Ventana Node Repository

- **Proyecto:** en esta ventana es donde se va a realizar el flujo de trabajo y donde se harán las conexiones entre nodos, se pueden tener varios flujos de trabajo distintos además de poder crear un grupo donde agrupar los flujos de trabajo que estén relacionados.

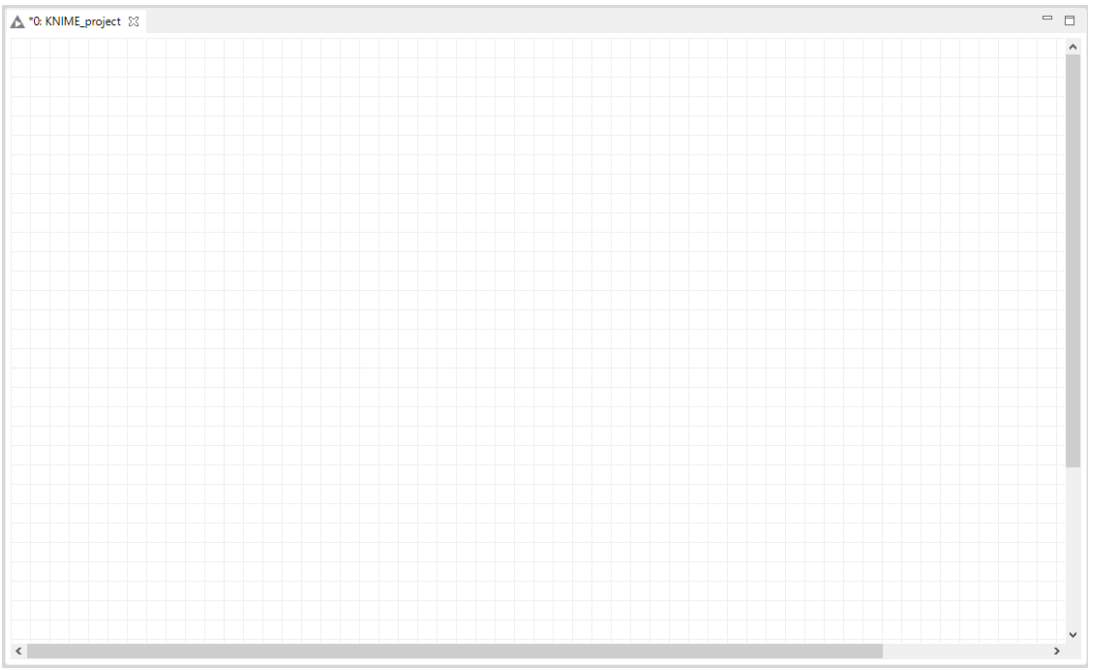


Imagen 3.9: Ventana del proyecto

- **Description y KNIME Hub:** estas ventanas nos aportarán información. En el caso de la ventana *"Description"* nos mostrará la información a tener en cuenta de cada nodo que estemos seleccionando o de la categoría seleccionada en *"Node Repository"*. En cambio *"KNIME Hub"* nos permite buscar la información que queramos dentro de la propia página web de KNIME.

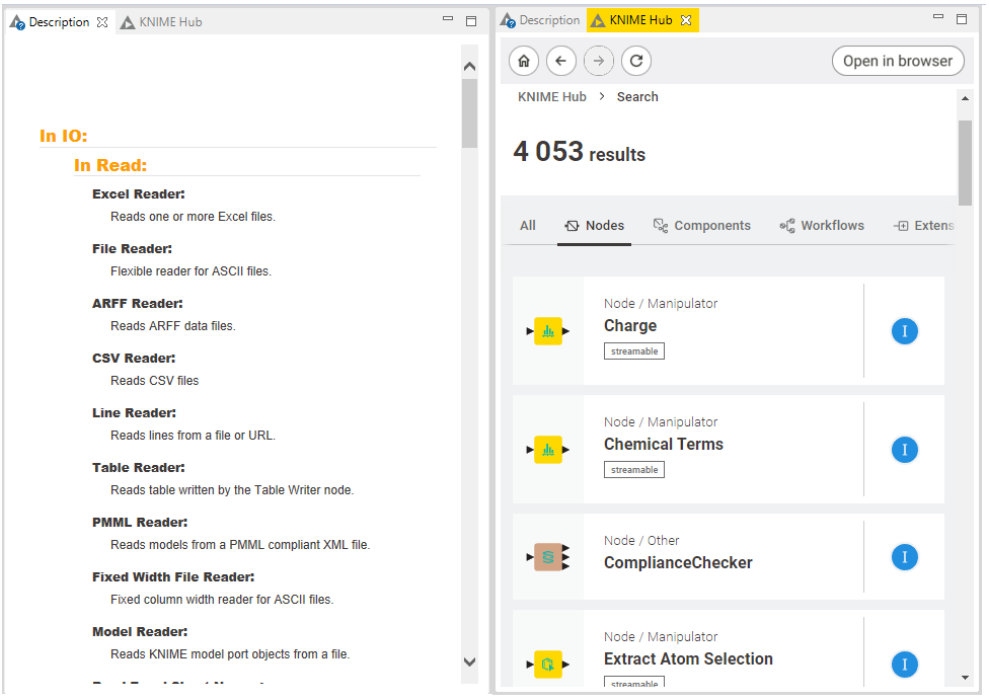


Imagen 3.10: Ventanas Description y KNIME Hub

- **Outline:** visualización general del flujo de trabajo en el que se está trabajando activamente.

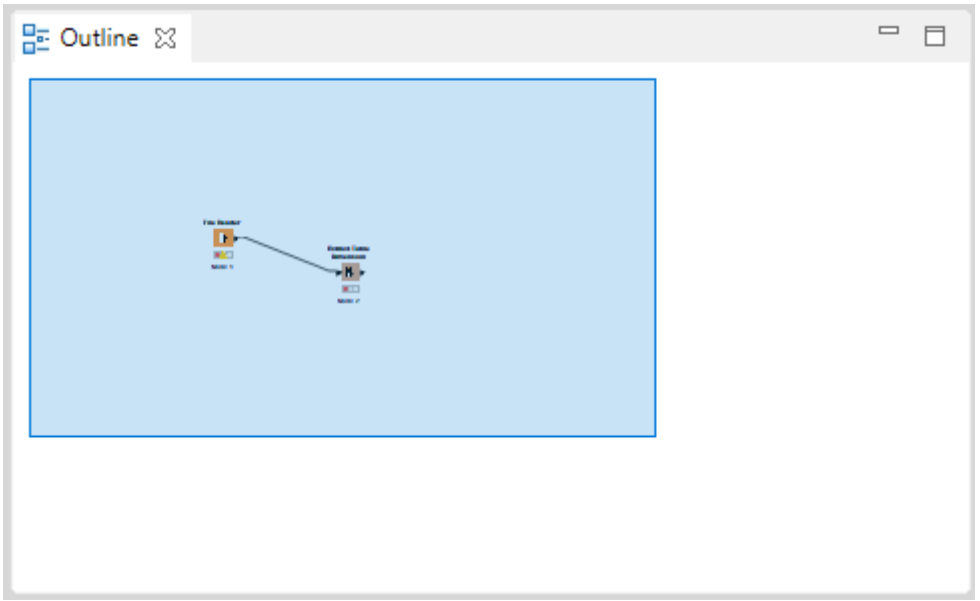


Imagen 3.11: Ventana Outline

- **Console:** se muestran los mensajes de ejecución e indica lo que pasa durante las ejecuciones.

Imagen 3.12: Ventana Console

3.2.2.- Nodos

Como se ha mencionado anteriormente en KNIME se trabaja con un flujo de nodos en el que cada nodo lleva a cabo una función diferente. Cada nodo se muestra como un cuadro de color con puertos de entrada y salida, asi como un estado como se puede ver en la imagen 3.13

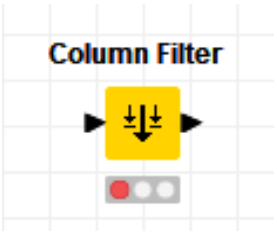


Imagen 3.13: Nodo en KNIME

En la imagen anterior se ve como el nodo tiene un puerto de entrada, que representa la información que el nodo va a procesar, un puerto de salida, que representa el resultado dado por el nodo post-ejecución y un recuadro donde se ilumina una luz roja. Este recuadro es el estado actual del nodo y se pueden diferenciar cuatro estados:

- **Luz roja** si el nodo no se ha configurado
- **Luz amarilla** si el nodo está configurado y no se ha ejecutado
- **Luz verde** si el nodo se ha ejecutado y no ha habido problema alguno
- **Cruz roja** si el nodo se ha ejecutado y ha habido algún fallo

Como se puede ver para ejecutar un nodo tiene que estar configurado, para ello se selecciona el nodo que queramos configurar con el click derecho del ratón y nos aparecerá un menú de configuración como el que se muestra en la imagen siguiente:

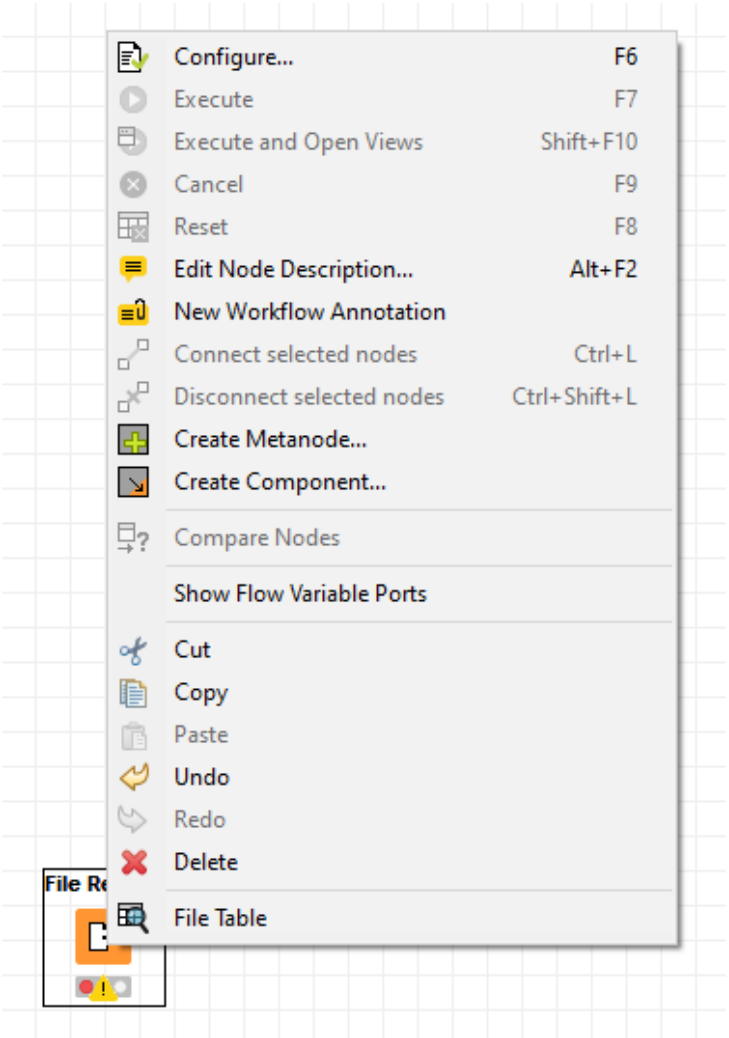


Imagen 3.14: Menú configurar nodo

A su vez, los nodos pueden recibir o dar como salida diferentes tipos de información como se puede ver en la imagen 3.15, cada uno de ellos se emplea para tareas totalmente diferentes. Para crear un flujo de información simplemente tienes que unir la salida de un nodo con la entrada del otro, siempre y cuando tanto la salida como la entrada sean del mismo tipo. Para ello se realiza un click izquierdo del ratón sobre el puerto de salida del primer nodo y manteniendo presionado el click izquierdo se une con el puerto de entrada del segundo nodo.

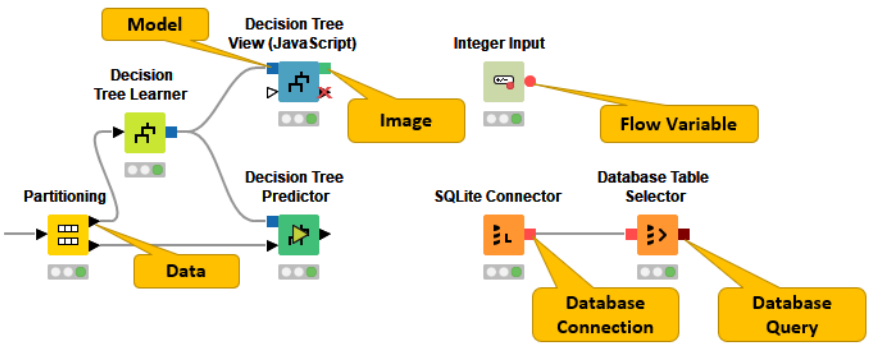


Imagen 3.15: Tipos de puertos

3.2.3.- Tipos de datos

Los tipos de datos básicos que se aceptan en KNIME son Integer, Double y String, junto con otros tipos de datos admitidos como Long, Boolean value, JSON, URI, Document, Date & Time, Vector de bits, Image y Blob. KNIME también admite tipos de datos personalizados.

Los nodos lectores en KNIME se encargan de asignar un tipo de datos a cada columna en función de su interpretación del contenido. Si el nodo lector no reconoce el tipo de datos correcto de una columna, el tipo de datos puede corregirse posteriormente. Además hay nodos disponibles para convertir tipos de datos dentro del flujo en el que estemos trabajando.

4.- Desarrollo del proyecto

En este capítulo se explicará el desarrollo del proyecto mostrando cada uno de los pasos que se han llevado a cabo de inicio a final y mostrando gráficamente la información más relevante dentro del proyecto.

4.1.- Problema a tratar

Para la realización del Trabajo de Fin de Grado se ha propuesto la resolución de un problema tratado en una competición de minería de datos llamada **"Data Mining Cup"** la cual es realizada con el objetivo de fomentar a los estudiantes universitarios que se adentren en el mundo de la minería de datos con el objetivo de resolver problemas en un entorno competitivo, pero que a su vez se pueda mostrar un escenario real cercano al mundo laboral que les espera.

El problema a tratar es el que se realizó en la **Data Mining Cup** de 2016, el escenario que se muestra es el siguiente:

"Un distribuidor de moda vende artículos de tamaños y colores particulares a sus clientes. En algunos casos, los artículos se devuelven al distribuidor por diversas razones y los datos del pedido y de la devolución relacionados se registran durante un período de dos años. El objetivo es utilizar estos datos y métodos de minería de datos para construir un modelo que permita una buena predicción de las tasas de retorno."

Además del escenario descrito, se nos ofrecen dos conjuntos de datos para realizar la resolución del problema. Uno de ellos es utilizado para entrenar el modelo y el otro está enfocado en realizar la predicción, en nuestro caso el segundo conjunto no será utilizado debido a que dicho conjunto no contiene el atributo a predecir y no podríamos comparar las predicciones de los modelos.

Para realizar la evaluación de los diferentes modelos seleccionados vamos a utilizar una validación cruzada con 10 pliegues sobre el conjunto de entrenamiento original.

El conjunto de entrenamiento disponible cuenta con un total de 2.325.165 filas. La información que se nos proporciona en forma de columnas dentro del conjunto de datos y que nos servirá de ayuda para la creación del modelo es la siguiente:

- **OrderID:** es el identificador de los pedidos que se han realizado, es un número de siete dígitos el cual va precedido por una letra **"a"** para diferenciarse del resto de identificadores.

- **OrderDate:** es la fecha en la cual se ha realizado el pedido, el formato en el cual está mostrada esta información es **"año-mes-día"** y el rango que abarca es desde "2014-01-01" hasta *"2015-09-30"*.
- **ArticleID:** es el identificador del artículo que hay dentro de un pedido, es un número de siete dígitos el cual va precedido por una letra **"i"** para diferenciarse del resto de identificadores.
- **ColorCode:** es el código de color que tiene cada artículo, está representado por un número de cuatro dígitos.
- **SizeCode:** es el código de tamaño que tiene cada artículo, se representa con varias unidades diferentes.
- **ProductGroup:** es el grupo al que se le ha categorizado un artículo, se representa mediante un número entero.
- **Quantity:** es la cantidad de un mismo artículo que hay en un pedido, se representa con un número entero.
- **Price:** es el precio que ha tenido el conjunto de artículos que pertenece a un pedido, se representa mediante un número el cual puede tener decimales.
- **RRP:** es el precio recomendado de venta sobre el conjunto de artículos que pertenece a un pedido, se representa mediante un número el cual puede tener decimales.
- **VoucherID:** es el identificador del cupón que se aplica a un conjunto de artículos dentro de un pedido, es un número de siete dígitos el cual va precedido por una letra **"v"** para diferenciarse del resto de identificadores o en caso de no haber cupón aplicable a dicho artículo un **"0"**.
- **VoucherAmount:** es el precio a descontar según el cupón que tiene un artículo, se representa mediante un número el cual puede tener decimales.
- **CustomerID:** es el identificador del cliente que ha realizado el pedido, es un número de siete dígitos el cual va precedido por una letra **"c"** para diferenciarse del resto de identificadores.
- **DeviceID:** es el identificador del dispositivo con el cual se ha realizado el pedido, el rango es **{1, 2, 3, 4, 5}**.
- **PaymentMethod:** es la forma o método de realización del pago del pedido.
- **ReturnQuantity:** es el total de artículos que han sido devueltos, se representa con un número entero. Esta información como se ha mencionado anteriormente solo aparece en el conjunto de entrenamiento debido a que es la columna a predecir por el modelo que se ha de crear.

En la siguiente imagen se puede comprobar los datos almacenados en el conjunto que vamos a utilizar:

Row ID	orderID	orderDate	articleID	colorCode	sizeCode	product...	quantity	price	rrp	vouche...	vouche...	custom...	deviceID	payme...	return...
Row18	a1000008	2014-01-01	i1001642	1001	44	7	1	110	149.99	0	0	c1089298	2	BPRG	1
Row19	a1000009	2014-01-01	i1001767	1999	44	8	1	39.99	39.99	0	0	c1054307	2	BPRG	0
Row20	a1000009	2014-01-01	i1001768	1995	44	8	1	39.99	39.99	0	0	c1054307	2	BPRG	0
Row21	a1000009	2014-01-01	i1002263	1001	44	13	1	69.99	69.99	0	0	c1054307	2	BPRG	0
Row22	a1000009	2014-01-01	i1002266	1001	44	13	1	59.99	59.99	0	0	c1054307	2	BPRG	0
Row23	a1000009	2014-01-01	i1002268	1001	44	13	1	69.99	69.99	0	0	c1054307	2	BPRG	1
Row24	a1000009	2014-01-01	i1002269	3001	44	13	1	69.99	69.99	0	0	c1054307	2	BPRG	0
Row25	a1000010	2014-01-01	i1001485	1978	34	6	1	50	69.99	0	0	c1089299	2	BPRG	1
Row26	a1000010	2014-01-01	i1001486	1953	36	6	1	55	79.99	0	0	c1089299	2	BPRG	1
Row27	a1000010	2014-01-01	i1001486	1978	36	6	1	55	79.99	0	0	c1089299	2	BPRG	1
Row28	a1000010	2014-01-01	i1002067	1970	32	9	1	10	49.99	0	0	c1089299	2	BPRG	1
Row29	a1000010	2014-01-01	i1002243	3978	34	13	1	20	49.99	0	0	c1089299	2	BPRG	1
Row30	a1000010	2014-01-01	i1003000	3975	I	17	1	15	25.99	0	0	c1089299	2	BPRG	1
Row31	a1000011	2014-01-01	i1000577	3953	44	3	1	25	29.99	0	0	c1065839	2	BPRG	0
Row32	a1000011	2014-01-01	i1000605	1927	42	3	1	49.99	49.99	0	0	c1065839	2	BPRG	1
Row33	a1000011	2014-01-01	i1001987	3001	44	8	1	25	39.99	0	0	c1065839	2	BPRG	0
Row34	a1000011	2014-01-01	i1001998	3992	44	8	1	39.99	39.99	0	0	c1065839	2	BPRG	0
Row35	a1000012	2014-01-01	i1000548	3854	36	3	1	20	27.99	0	0	c1035011	2	CBA	0
Row36	a1000012	2014-01-01	i1001962	1972	38	8	1	10	39.99	0	0	c1035011	2	CBA	0
Row37	a1000013	2014-01-01	i1000339	1950	40	3	1	5	19.99	0	0	c1002813	3	BPRG	1
Row38	a1000013	2014-01-01	i1000339	1950	42	3	1	5	19.99	0	0	c1002813	3	BPRG	1
Row39	a1000013	2014-01-01	i1000339	1966	40	3	1	5	19.99	0	0	c1002813	3	BPRG	1
Row40	a1000013	2014-01-01	i1001944	3972	42	8	1	15	39.99	0	0	c1002813	3	BPRG	1
Row41	a1000013	2014-01-01	i1002244	2934	40	13	1	35	69.99	0	0	c1002813	3	BPRG	1
Row42	a1000013	2014-01-01	i1002244	2934	42	13	1	35	69.99	0	0	c1002813	3	BPRG	1
Row43	a1000014	2014-01-01	i1000614	1953	44	3	1	35.99	35.99	0	0	c1004678	2	KKE	0
Row44	a1000014	2014-01-01	i1000614	1995	44	3	1	35.99	35.99	0	0	c1004678	2	KKE	0
Row45	a1000014	2014-01-01	i1003018	3976	I	17	1	10	17.99	0	0	c1004678	2	KKE	1
Row46	a1000015	2014-01-01	i1000502	1999	34	3	1	29.99	29.99	0	0	c1086877	2	BPRG	0
Row47	a1000016	2014-01-01	i1000576	3978	44	3	1	25	29.99	v1000040	10	c1089300	2	BPRG	1
Row48	a1000016	2014-01-01	i1000620	2001	44	3	1	39.99	39.99	v1000040	10	c1089300	2	BPRG	0

Imagen 4.1: Ejemplo entradas del fichero

4.2.- Preprocesado

Para empezar con el desarrollo del Trabajo de Fin de Grado uno de los pasos principales y más importantes en un proyecto de minería de datos es el preprocesado de los datos, debido a que hay que manipular la información que tenemos para lograr encontrar la manera más óptima para trabajar con dicha información.

En primer lugar se procedería a realizar una limpieza de datos de aquella información que puede haber sido mal almacenada o pueden haber errores en los mismos, un caso puede ser aquellas compras que cuentan con cero artículos pedidos o compras que cuentan con un número de artículos pedidos menor a la cantidad devuelta. Estas dos opciones son imposibles que pasen por lo cual se puede ver como un error a la hora de recopilar la información y, como se ha comentado anteriormente, al contar con una gran cantidad de datos se ha decidido por eliminar los datos que entran dentro de los dos casos nombrados. En total se eliminarían 31.644 datos, lo que nos dejaría con 2.293.521 datos.

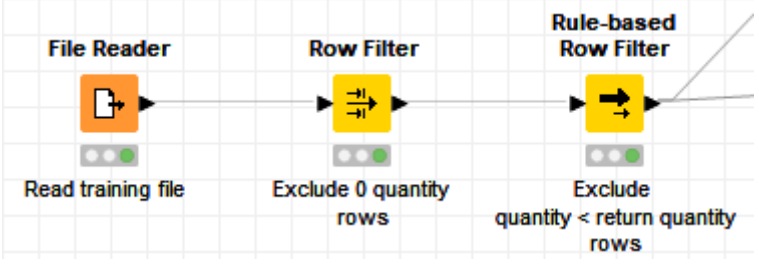


Imagen 4.2: Flujo primera fase preprocesamiento

Lo que se muestra en la imagen 4.2 es el flujo que se ha realizado dentro de KNIME para la realización del primer paso del preprocesamiento nombrado anteriormente, como se describe en primer lugar hacemos una lectura del fichero de entrenamiento. Una vez hecho esto se procede a filtrar los datos que queremos excluir y así limpiar el conjunto de datos.

El siguiente paso que se ha crear una estandarización del código de tamaños/tallas que tienen los artículos, esto se debe a que la información está recopilada según las tallas de diferentes países o continentes y según el tipo de prenda por lo cual la forma en la que se mide la talla es diferentes según de donde proviene dicha información.

Los atributos que intervienen en la realización de esta agrupación son sizeCode, atributo al cual estamos haciendo esa agrupación y productGroup, atributo que nos dice el tipo de producto del artículo. Como conclusión de este estudio se extrajo qué artículos marcados con el grupo 17 están relacionados con prendas que abarcan tallas desde 80 hasta 100, y los productos marcados con el grupo 2 se relacionan con tallas más pequeñas que abarcan desde 24 hasta 30.

Para ello se busco información de diferentes tipos de prendas y su conversión en una talla global para estandarizar esta medida, la forma en la que se realizó se muestra en la siguiente tabla:

Size Code	New Size Code
25, 26, 75, 80, 30, 32	XS
27, 28, 29, 85, 34, 36	S
90, 38, 40	M
95, 100, 42	L
44	XL

Además de las medidas descritas en la tabla aparecen otras medidas que no encajan con ninguna información existente por lo que se ha decidido tratar a estas medidas como si fueran la media, es decir, la tratamos como la talla M.

Esta estandarización nos facilita tratar con los datos debido a que minimiza tanta disparidad de valores, esto se puede ver comparando las imágenes 4.3 y 4.4.

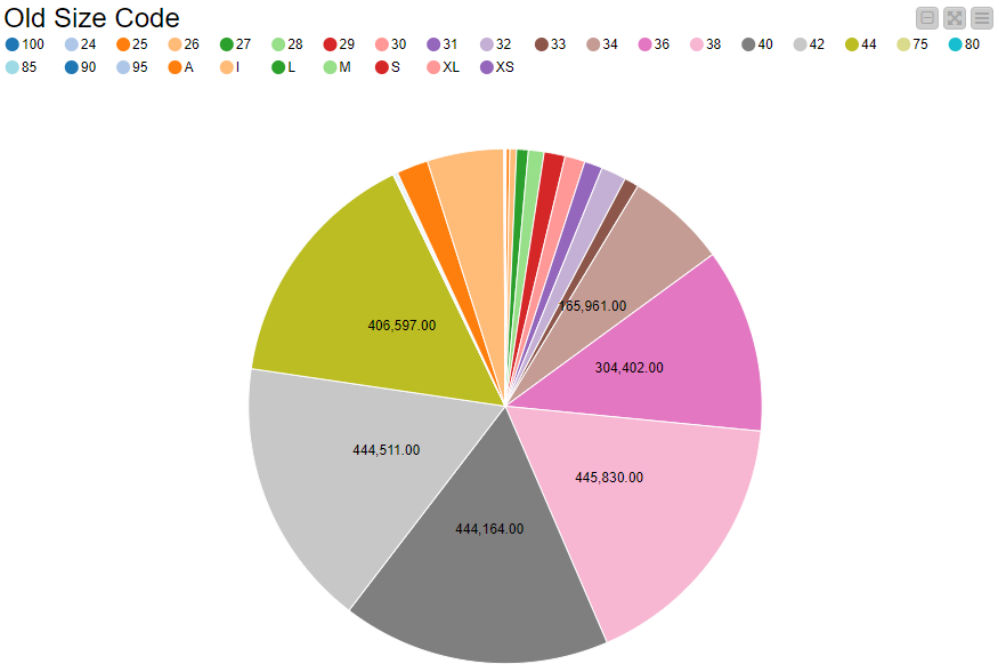


Imagen 4.3: Gráfica código de talla antiguo

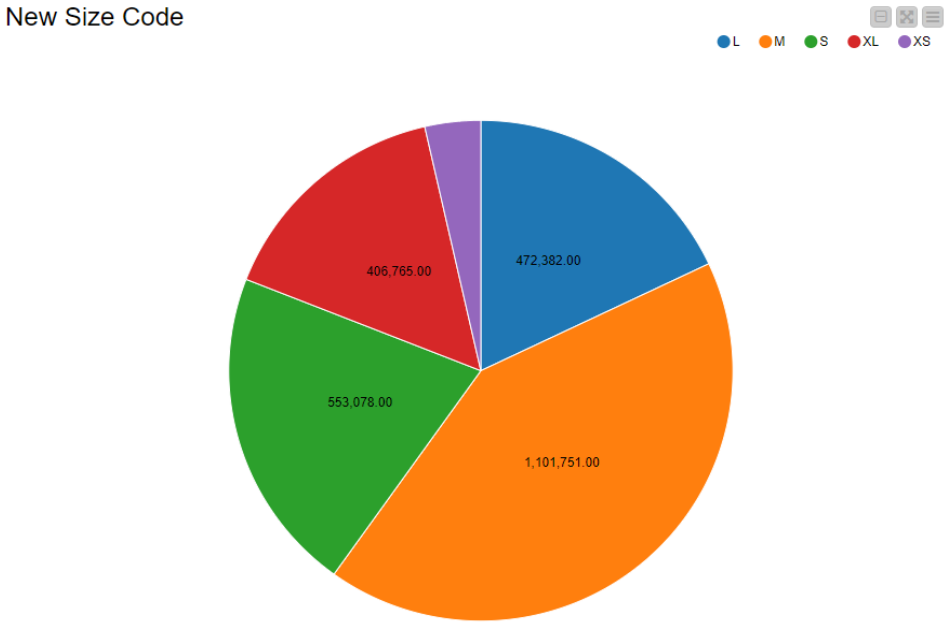


Imagen 4.4: Gráfica código de talla nuevo

Para realizar esto dentro de KNIME se hace uso de un nodo que se encarga de crear un bucle necesario para ir cambiando las variables que nos interese, en este caso es el código de talla, y repetir el proceso hasta que se cambien todos los datos, después de realizar esto también se ha realizado un redondeo de los valores numéricos con valores decimales para así evitar que se mezcle la información por decimales que se diferencien por números muy pequeños. Este flujo se muestra en la imagen siguiente:

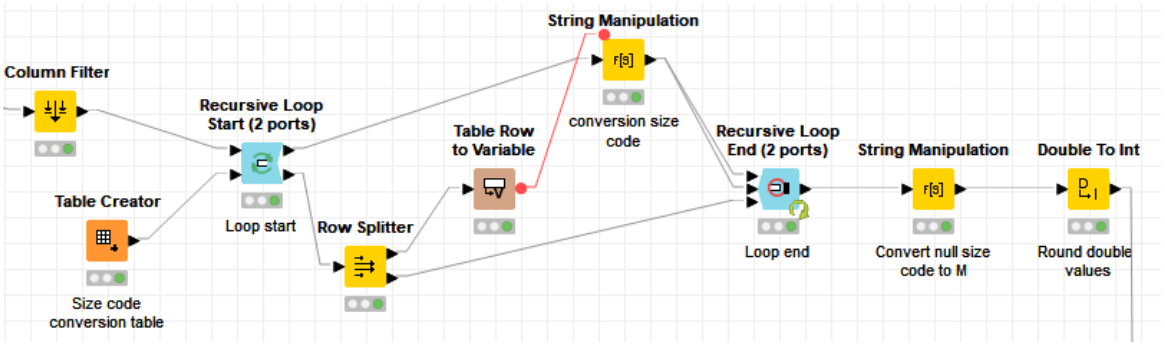


Imagen 4.5: Flujo segunda fase preprocesamiento

El siguiente paso se encarga de calcular el precio real después de descontar el precio de los vales de descuento que corresponden a cada artículo, el flujo dentro de la aplicación es sencillo usando dos nodos para calcular el precio y comprobar si hay algún error de cálculo.

Y por último se eliminan aquellos datos que no van a ser usados después de recalcular el precio.

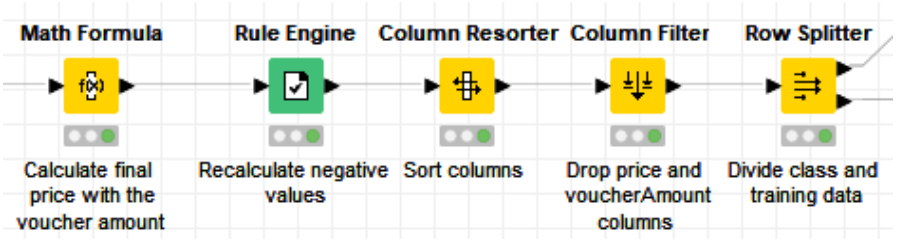


Imagen 4.6: Flujo cuarta fase preprocesamiento

Con el objetivo de enriquecer el conjunto actual de datos se amplió la información creando nuevos atributos como el porcentaje de artículos devueltos por cada cliente sobre el número de pedidos realizados o el ratio de veces que un artículo ha sido devuelto sobre el número de veces que ha sido pedido. Para esto se realizaron dos flujos diferentes, uno agrupando la información por clientes y el otro agrupando por artículos, respectivamente.

Los flujos implementados hacen uso el nodo ***Group By***, el cual nos permite agrupar en función de una variable y después gracias a nodos como ***Math Formula*** podemos especificar una fórmula con la cual extraer la información que necesitamos. Ambos flujos pueden verse en la imagen 4.7.

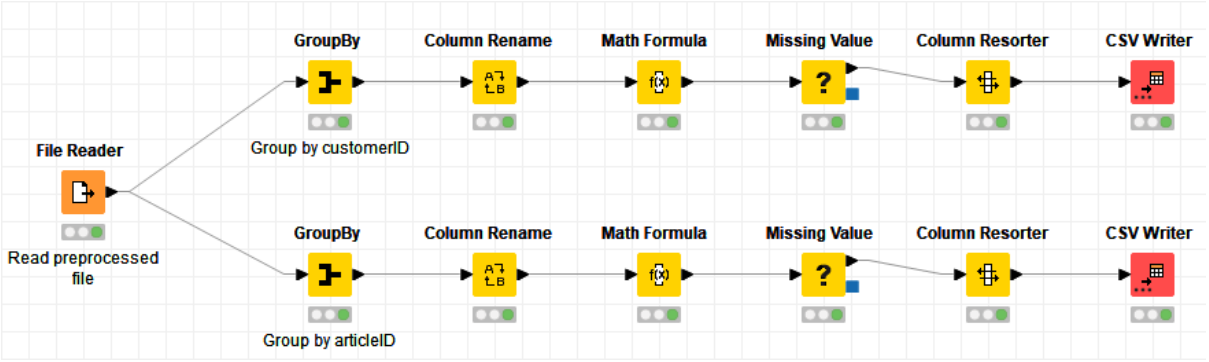


Imagen 4.7: Flujo agrupamiento de datos

Una vez obtenido los nuevos atributos generados se han de juntar con el conjunto de datos para obtener la vista minable, para ello en el flujo en KNIME usaremos el nodo ***Joiner*** que nos permite juntar la información de dos tablas según un atributo en común , en nuestro caso será en primer lugar los clientes y en segundo lugar los artículos:

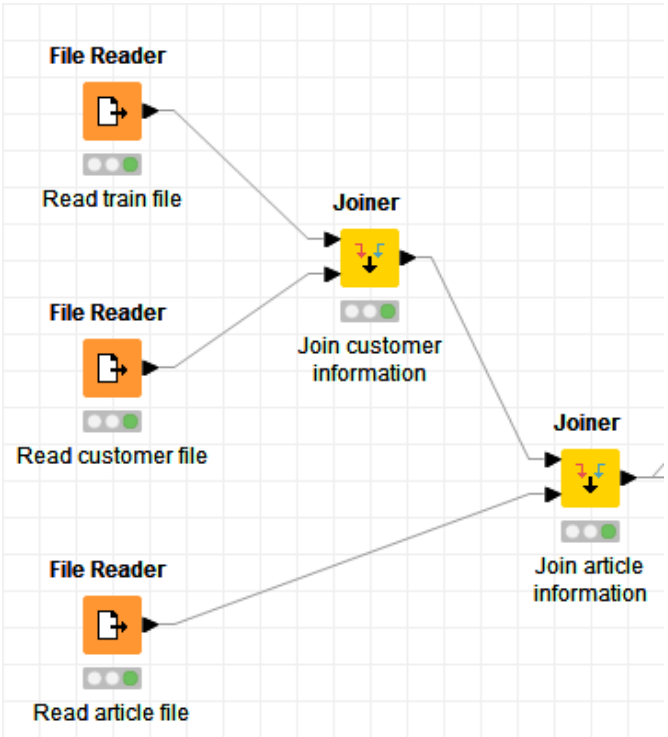


Imagen 4.8: Flujo unión de información

4.3.- Minería de datos

Una vez realizada la fase de preprocesado se pasa a la fase de minería de datos donde se usarán diferentes modelos para llevar a cabo la labor de predecir las devoluciones que tendrán los pedidos.

Para comprobar gráficamente aquellas variables más significativas dentro del conjunto de entrenamiento mostramos la matriz de correlación:

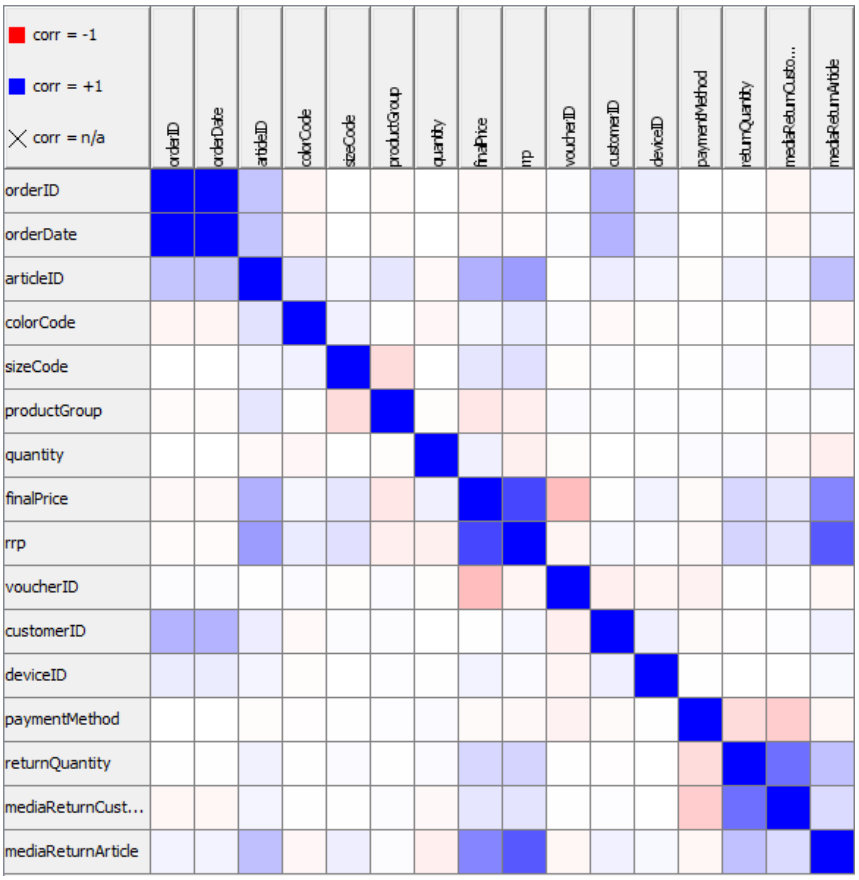


Imagen 4.9: Matriz de correlación

Las variables orderID y orderDate no serán utilizadas en la construcción de nuestros modelos por razones obvias. Tampoco se va a utilizar paymentMethod porque después de analizar los datos y debido a la variabilidad de la misma no aportaba suficiente información.

Los modelos que vamos a generar son los correspondientes a los métodos siguientes:

- Regresión lineal.
- Árbol de regresión utilizando XGBoost, con una profundidad máxima de cuatro niveles.
- Método por vecindad de los k vecinos más cercanos (k = 5).
- Red neuronal mediante un perceptrón multicapa, con una capa oculta y diez neuronas por cada capa.

No se ha podido implementar un árbol de regresión LightGBM debido a que en KNIME no está disponible dicho algoritmo. Tampoco se ha podido utilizar el SVR

porque el tiempo de generación del modelo excedió el límite máximo que se había fijado.

Para cada método generado se calcularon las medidas de error siguientes:

- **Error absoluto medio:** es una medida de la diferencia entre dos variables continuas. Considerando dos series de datos (unos calculados y otros observados) relativos a un mismo fenómeno, el error absoluto medio sirve para cuantificar la precisión de una técnica de predicción comparando por ejemplo los valores predichos frente a los observados.

- **Error cuadrático medio:** en el análisis de regresión, el término de error cuadrático medio se utiliza a veces para referirse a la estimación insesgada de la varianza del error: la suma residual de cuadrados, dividida por el número de grados de libertad. El denominador es el tamaño reducido de la muestra por el número de parámetros del modelo estimado a partir de los mismos datos.

- **Raíz del error cuadrático medio:** es una medida de uso frecuente de las diferencias entre los valores predichos por un modelo o un estimador y los valores observados. Representa la raíz cuadrada del segundo momento de la muestra de las diferencias entre los valores previstos y los valores observados o la media cuadrática de estas diferencias. Estas desviaciones se denominan residuos cuando los cálculos se realizan sobre la muestra de datos que se utilizó para la estimación y se denominan errores (o errores de predicción) cuando se calculan fuera de la muestra.

4.4.- Evaluación y resultados

En nuestro experimento haremos una validación cruzada con 10 pliegues. Para ello el conjunto de datos original se dividirá en 10 partes iguales de las cuales una se utilizará como conjunto de prueba y las nueve restantes como conjunto de entrenamiento, repitiendo el proceso diez veces de forma que al final se hayan usado todos los datos para evaluar el modelo.

Un forma de hacer esta forma de validación cruzada en KNIME es con el siguiente flujo:

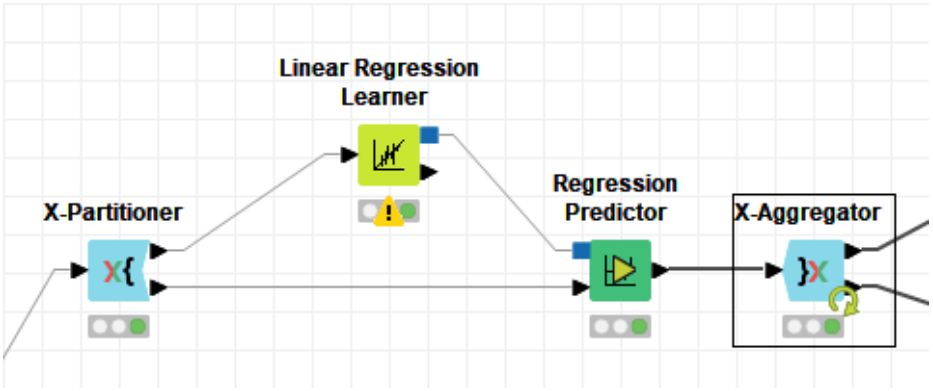


Imagen 4.10: Ejemplo validación cruzada

El nodo ***X-partitioner*** nos permite elegir el número de pliegues que se van a utilizar y el atributo que queremos predecir, luego aplicamos el modelo (en este caso es el de regresión linear) y por último en el nodo ***X-aggregator*** se almacenan los errores de cada uno de los pliegues.

Luego simplemente tendríamos que usar un nodo ***Numeric Scorer*** para obtener en forma de tabla el valor medio de cada error y comparar entre los demás modelos.

Los resultados de cada uno de los modelos creados se ven reflejados en las siguientes imágenes:

Mean absolute error:	0,335
Mean squared error:	0,163
Root mean squared error:	0,403

Imagen 4.11: Resultados regresión lineal

Mean absolute error:	0,349
Mean squared error:	0,177
Root mean squared error:	0,421

Imagen 4.12: Resultados XGBoost

Mean absolute error:	0,287
Mean squared error:	0,289
Root mean squared error:	0,538

Imagen 4.13: Resultados KNN

Mean absolute error:	0,327
Mean squared error:	0,161
Root mean squared error:	0,402

Imagen 4.14: Resultados perceptrón multicapa

Analizando estos resultados podemos comprobar que la regresión lineal tiene un comportamiento mejor que el XGBoost. Esto puede ser debido a que en nuestro experimento el número de niveles de profundidad ha sido limitado a cuatro debido a la gran cantidad de datos que deben ser procesados y para no exceder los tiempos máximos de generación del modelo.

Atendiendo al error cuadrático medio y a la raíz cuadrada del error cuadrático medio el modelo que mejor se comporta es la red neuronal perceptron multicapa.

Si nos fijamos en el error absoluto medio el modelo que mejor comportamiento tiene es KNN (k = 5).

5.- Conclusiones y líneas futuras

En este trabajo hemos abordado el problema de predicción de tasa de devolución de productos de una empresa textil utilizando diferentes técnicas de minería de datos.

Hemos creado modelos para las técnicas de regresión lineal múltiple, XGBoost, KNN y perceptrón multicapa. Tales modelos han sido evaluados utilizando una validación cruzada con 10 pliegues.

Los resultados obtenidos indican que el modelo que mejor comportamiento tiene es el perceptrón multicapa usando el error cuadrático medio. Si se tiene en cuenta el error absoluto medio el modelo que mejor comportamiento tiene parece ser KNN.

Como líneas futuras de trabajo y con el fin de estudiar con más detalle este problema se podría aumentar la potencia de cómputo para generar por ejemplo árboles de regresión con mayor profundidad, realizar un preprocesado de datos más sofisticado, y utilizar otras técnicas de minería de datos más avanzadas.

6.- Summary and conclusions

In this work we have approached the problem of predicting the return rate of products of a textile company using different data mining techniques.

We have created models for multiple linear regression, XGBoost, KNN, and multilayer perceptron techniques. Such models have been evaluated using a 10-fold cross-validation.

The results obtained indicate that the model with the best performance is the multilayer perceptron using the mean square error. If the mean absolute error is taken into account, the model with the best behavior seems to be KNN.

As future lines of work and in order to study this problem in more detail, the computing power could be increased to generate, for example, regression trees with greater depth, perform more sophisticated data preprocessing, and use more advanced data mining techniques.

7.- Presupuesto

Al ser un proyecto en el cual las tecnologías utilizadas han sido gratuitas o de código abierto el coste se calculará utilizando las horas de trabajo del autor haciendo una estimación del salario por hora de un ingeniero informático junior.

Para el desarrollo del Trabajo de Fin de Grado se han empleado alrededor de **300 horas**, por lo cual si usamos un salario aproximado de **20€ la hora** el coste final sería de **6.000€.**

Si a esto le añadimos el coste del hardware utilizado, los cuales serían un **portatil Asus TUF Gaming F15** con un coste de **800€** y un **monitor BenQ ZOWIE XL2411K** con un coste de **230€,** en total sumaría un cantidad total de **7.030€.**

8.- Bibliografía

[1] Daria Dzyabura, Siham El Kihal, Marat Ibragimov, Leveraging the Power of Images in Predicting Product Return Rates.

[2] Toktay, Beril & van der Laan, Erwin & de Brito, Marisa. (2003). Managing Product Returns: The Role of Forecasting. Working Paper. 10.1007/978-3-540-24803-3_3.

[3] Hailong Cui, Sampath Rajagopalan, Amy R. Ward, Predicting product return volume using machine learning methods, European Journal of Operational Research.

[4] Hernández Orallo, J., Ramírez Quintana, M. and Ferri Ramírez, C., 2010. *Introducción a la minería de datos*. Madrid: Pearson.

[5] “Open for Innovation,” KNIME. https://www.knime.com/.

[6] “Weka 3 - Data Mining with Open Source Machine Learning Software in Java.” https://www.cs.waikato.ac.nz/ml/weka/.

[7] “RapidMiner,” RapidMiner. https://rapidminer.com/.

[8] “Welcome to Python.Org.” https://www.python.org/.

[9] “R: The R Project for Statistical Computing.” https://www.r-project.org/.