UNIVERSIDAD
DE GRANADA

Práctica 1

Análisis Predictivo
mediante Clasificación

Mauricio Luque Jiménez

78004003D

maulujim@correo.ugr.es

Subgrupo de prácticas A1

12 de noviembre de 2025

Inteligencia de Negocio

Índice

Predicción de Churn de Usuarios de Spotify

Introducción

Este conjunto de datos representa la cancelación de usuarios de la plataforma Spotify. Cuenta con 8.000 instancias de manera que cada instancia representa a un usuario distinto con su respectiva información.

Contiene variables tanto numéricas (edad, tiempo de escucha, canciones reproducidas por día, etc.) como categóricas (tipo de suscripción, dispositivo utilizado para abrir la aplicación, etc.), así como dos variables booleanas (que indican si el usuario ha escuchado música sin conexión y si finalmente ha cancelado la suscripción). Es por esto último que se trata de un problema binario, ya que la variable clase a predecir es si el usuario cancela o no la suscripción.

Algunas variables están relacionadas entre sí; por ejemplo, el tipo de suscripción y el número de anuncios escuchados (sólo la suscripción gratuita contiene anuncios, mientras que el resto de suscripciones no), puede ser conveniente realizar una selección de características.

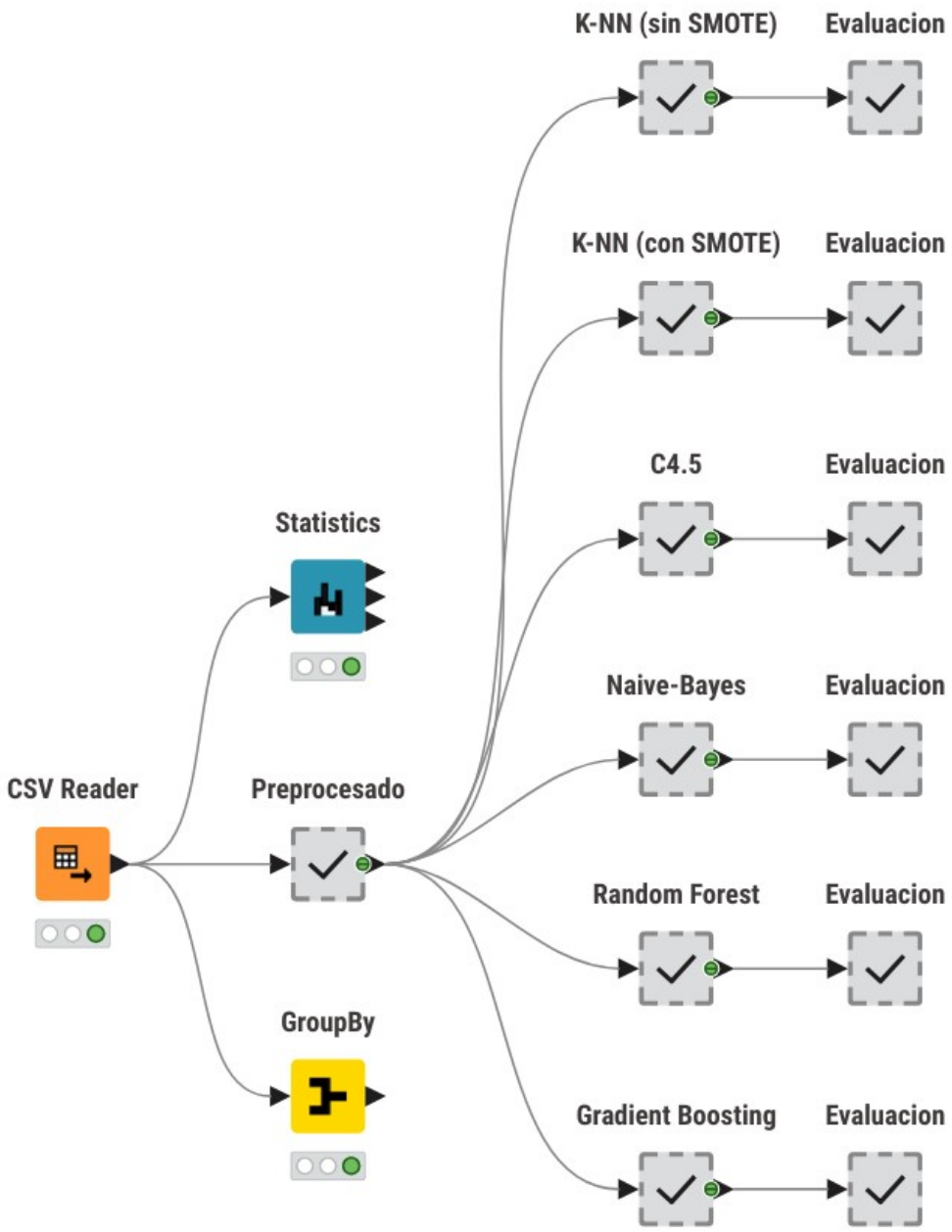
Procesado de datos

Para este problema, vamos a realizar un procesado básico dividido en ajustes comunes a todos los algoritmos y ajustes específicos para cada tipo de algoritmo. Empezando por los ajustes más generales, cabe destacar que no hay valores ausentes, como se puede comprobar en el nodo *Statistics,* por lo que no hay que preocuparse por imputar dichos valores. En segundo lugar, nos encontramos con el tipado correcto de los atributos, en el que principalmente va a haber que cambiar el tipo de la variable clase *is_churned*. Para ello, hacemos uso del nodo *Number to String* para esa variable con el objetivo de evitar ambigüedades en el entrenamiento y que Knime interprete la clase como variable categórica, puesto que no sigue un orden. En tercer lugar, podemos usar el nodo *Column Filter* para eliminar los identificadores de cada instancia. Por último, y de manera opcional, se podría optar por una selección de características para evitar datos redundantes. En este caso, habría que hacer dos selecciones separadas para variables numéricas y categóricas, ya que, como indica Knime, las variables deben ser del mismo tipo. Sin embargo, al realizar dicha selección, se ha dado el caso de que sólo se ha eliminado una característica, lo que no mejora en exceso el procesado de los datos.

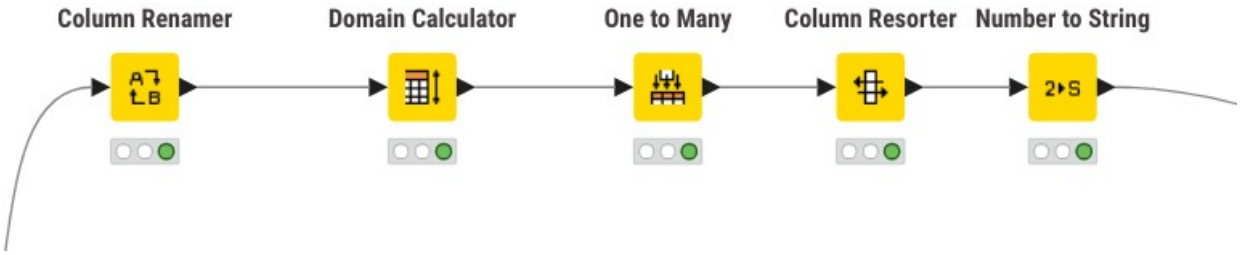
Por otra parte, tenemos aspectos del procesado que dependen del tipo de algoritmo que utilicemos. Por ejemplo, en este problema tiene sentido aplicar k-NN (o cualquier algoritmo que se maneje bien con datos numéricos) debido a la gran cantidad de variables numéricas que tiene. De esta manera, para las variables categóricas que resten, se puede aplicar una codificación en caliente con el nodo *One to Many*, que en este caso se puede usar para las variables que indican el tipo de suscripción, el dispositivo desde que usa la aplicación o el país. Además, en este caso no genera demasiadas columnas adicionales (teniendo en cuenta que las columnas originales se eliminan), por lo que parece razonable esta codificación. Otro aspecto importante es la normalización de los datos, ya que hay algunos que son [0,1] (como si se escucha música sin conexión) y otros que tienen un rango mucho mayor como el de la edad. Aunque nos estamos centrando en algoritmos numéricos, la normalización se realizará a nivel de cada algoritmo independiente, para no trastocar la información de otros algoritmos que no tengan problemas con los rangos de los datos. Otra cuestión es que, como se puede ver en el resultado del nodo *Group By*, la clase negativa es mucho más mayoritaria que la clase positiva. De esta manera, sería razonable solucionar ese desbalanceo de clases con el algoritmo SMOTE, indicándole que aumente el número de muestras de la clase minoritaria. Esto es algo que conviene aplicar a nivel más específico para no perjudicar a aquellos modelos robustos al desbalanceo. En cambio, si se pretende utilizar algoritmos más habituados a variables categóricas, como los árboles de decisión, el procesado es diferente. Ya no es necesaria la normalización y pasan a ser más importantes otras cuestiones como la complejidad del modelo, ajustando el número de nodos o la profundidad que se deben crear.

Resultados obtenidos

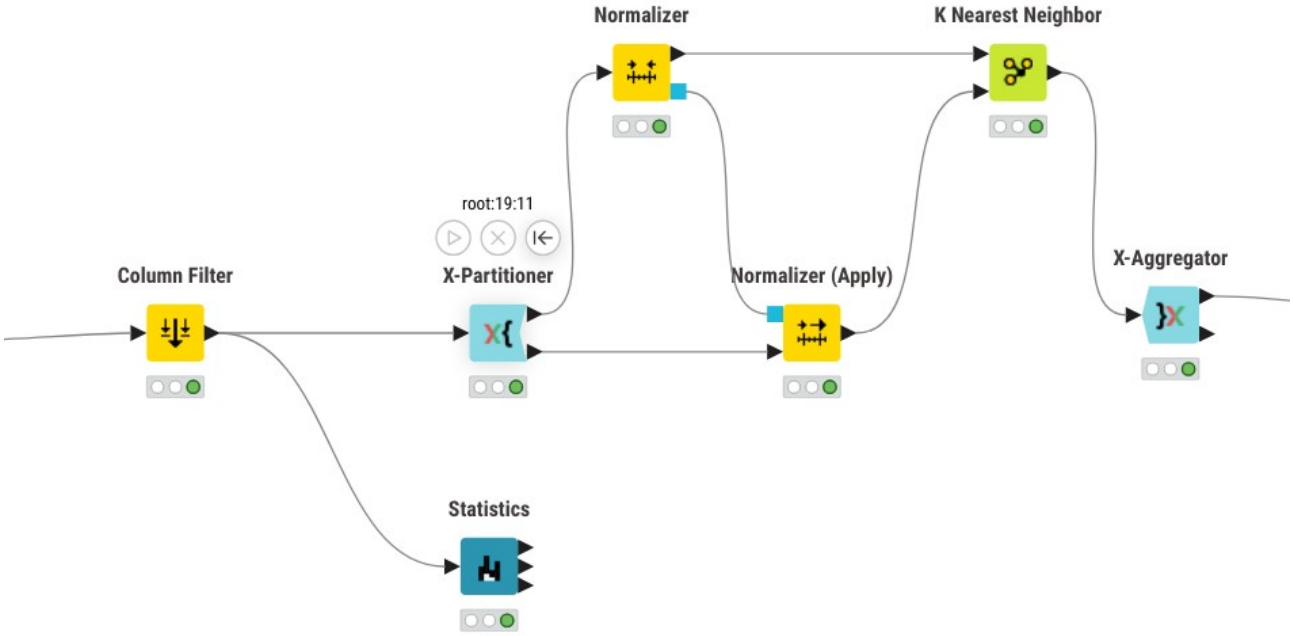
En primer lugar, conviene mostrar el flujo de trabajo del problema, el cual se muestra a continuación.



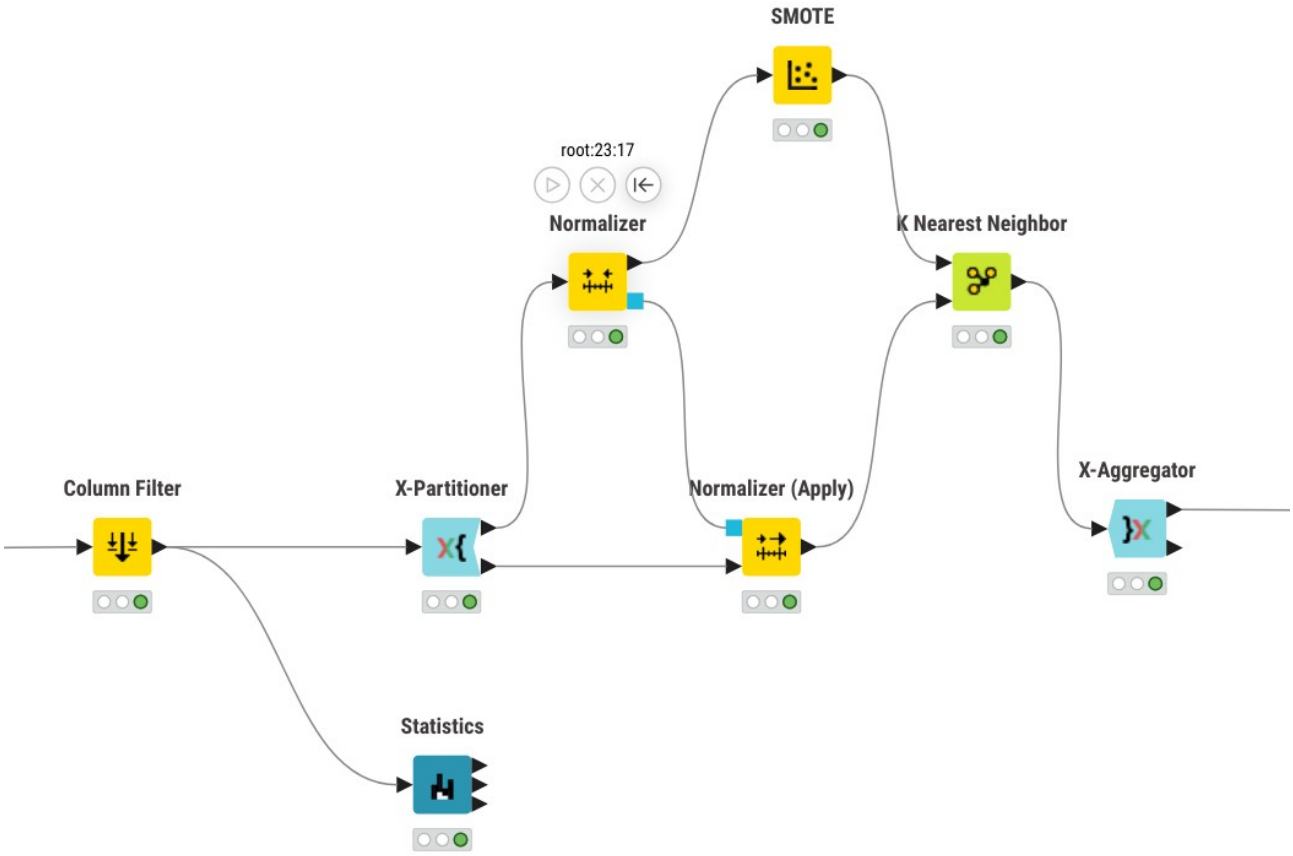
Dentro del metanodo del preprocesado se han realizado las siguientes tareas: renombrar algunas columnas, entre otras cosas para reconocer mejor a la variable clase, convertir variables categóricas en numéricas y cambiar el tipo de la variable clase para identificarla como cadena de caracteres.



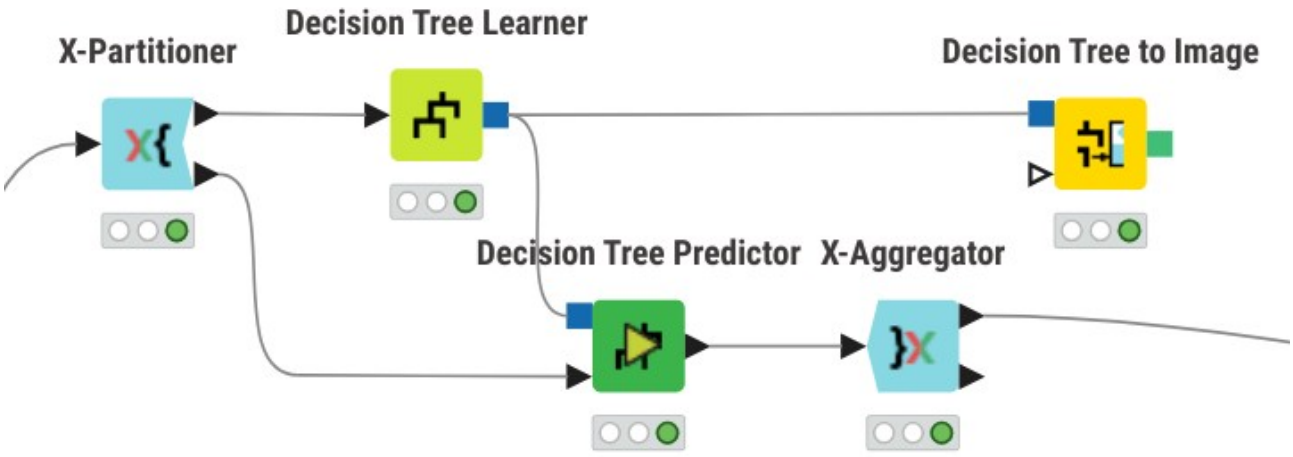
A continuación, los distintos flujos de trabajo de cada algoritmo, representados por metanodos en el flujo principal.



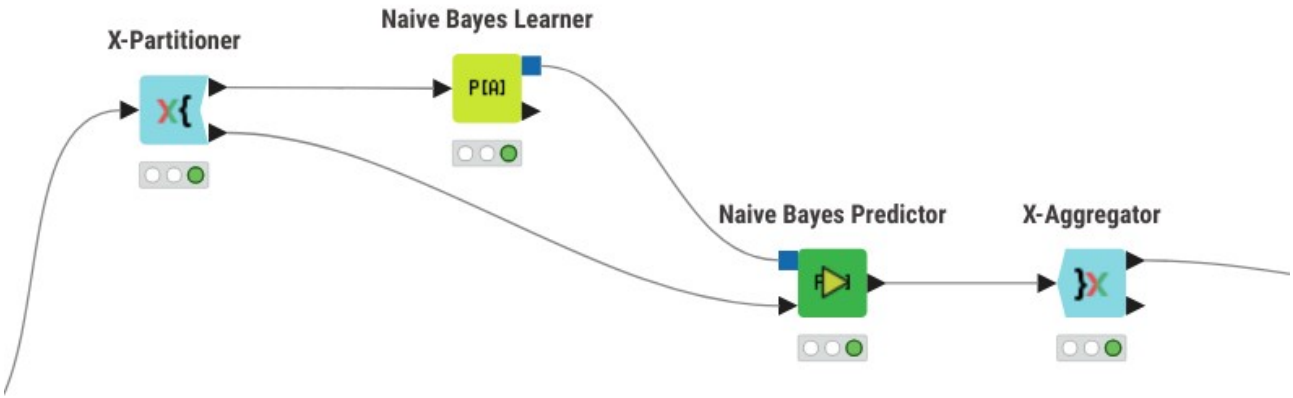
Flujo de trabajo del algoritmo K-NN sin aplicar SMOTE



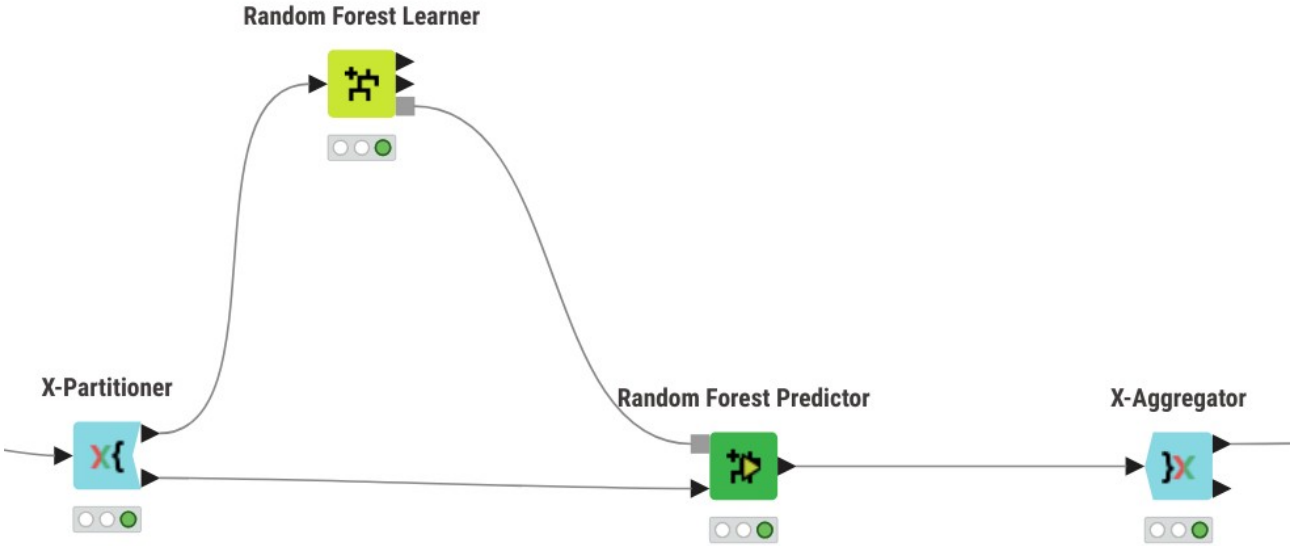
Flujo de trabajo del algoritmo K-NN aplicando SMOTE



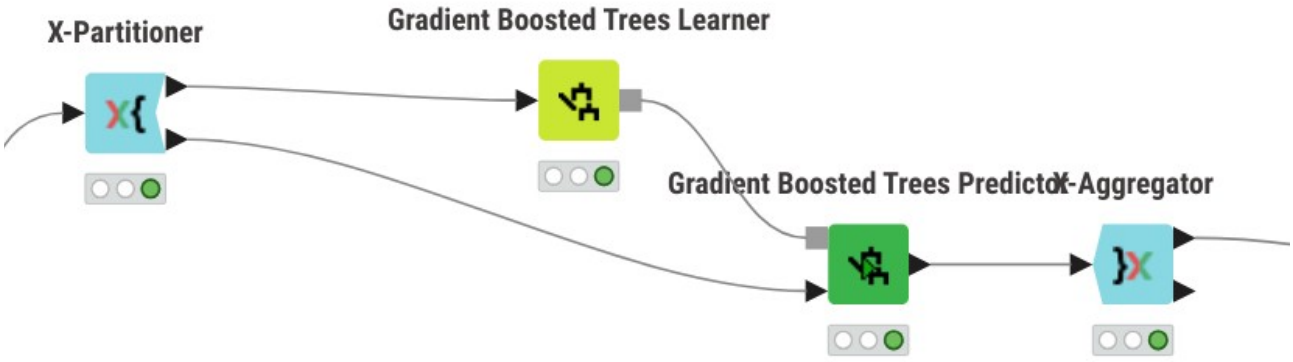
Flujo de trabajo del algoritmo C4.5



Flujo de trabajo del algoritmo Naïve-Bayes



Flujo de trabajo del algoritmo Naïve-Bayes



Flujo de trabajo del algoritmo Gradient Boosting

Finalmente, aquí se recogen todas las métricas obtenidas de los algoritmos anteriores.

	Precisión	TPR	TNR	G-Mean	F1	AUC
K-NN(sin SMOTE)	0’259	0’180	0’821	0’384	0’212	0’511
K-NN (con SMOTE)	0’269	0’422	0’600	0’503	0’328	0’502
C4.5	0’269	0’290	0’725	0’459	0’279	0’505
Naïve-Bayes	0	0	1	0	0	0’502
Random Forest	0’286	0’001	0’999	0’032	0’003	0’494
Gradient Boosting	0’180	0’004	0’993	0’063	0’008	0’500

Configuración de algoritmos

Análisis de resultados

Interpretación de los datos

Satisfacción de los Pasajeros de Aerolíneas

Introducción

Este conjunto de datos representa la valoración realizada por distintos pasajeros de su experiencia en vuelos comerciales. Cuenta con 25,976 instancias y 25 variables, siendo el conjunto de mayor tamaño de los tres que se evalúan en esta práctica.

Contiene tanto variables numéricas (edad, distancia recorrida, tiempos de retraso) como categóricas (tipo de cliente, clase de vuelo, nivel de satisfacción). Es importante destacar que hay variables que, aunque sean numéricas, son ordinales, como es el caso de todas las variables relacionadas con la valoración de cada pasajero. Por último, como la variable clase es una variable booleana (aunque, a diferencia del primer problema, aquí la variable es inicialmente categórica), se trata de un problema binario.

En este caso, es importante realizar un correcto preprocesado de datos antes de pasar al análisis de los datos, ya que hay varios aspectos que deben tenerse en cuenta. Aquí pueden entrar aspectos como la normalización de variables continuas, el correcto tratamiento de las variables según el tipo de algoritmo que se vaya a aplicar o la imputación de valores perdidos.

Procesado de datos

Si bien el preprocesado de este problema es más breve que el del primer problema, al menos a nivel general sin entrar en la especificidad de cada algoritmo, quizá sea más importante debido a una característica que no tenía el conjunto de datos anterior: los valores ausentes.

En este caso, nos encontramos con dos variables que presentan valores ausentes: la variable que mide la distancia de los vuelos y la variable que mide el tiempo de retraso en las llegadas. Como ambas son variables numéricas, podemos ejecutar el nodo *Missing Values* indicando únicamente que impute valores perdidos en base a la mediana de los valores presentes. El resto de tipos de datos no es necesario indicarlos.

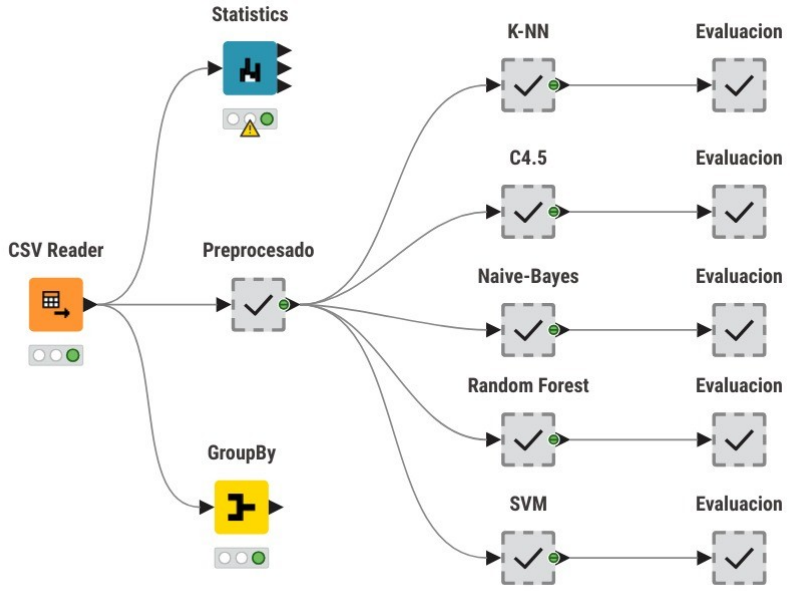
Al igual que en el primer problema, hay varias variables categóricas que se pueden transformar en numéricas, lo cual puede ser útil para algoritmos como k-NN. Sin embargo, otros algoritmos como los árboles de decisión no necesitan esta codificación, por lo que se va a realizar después de todo el preprocesado general, para que se aplique únicamente para ellos algoritmos que realmente la necesitan.

Por último, se han hecho algunos pequeños ajustes como eliminar la columna ID o renombrar algunas columnas (por ejemplo, la columna *satisfaction* se ha cambiado por *class*, mientras que la variable relativa al tipo de asiento llamada *Class* se ha cambiado por *Type of Seat*.

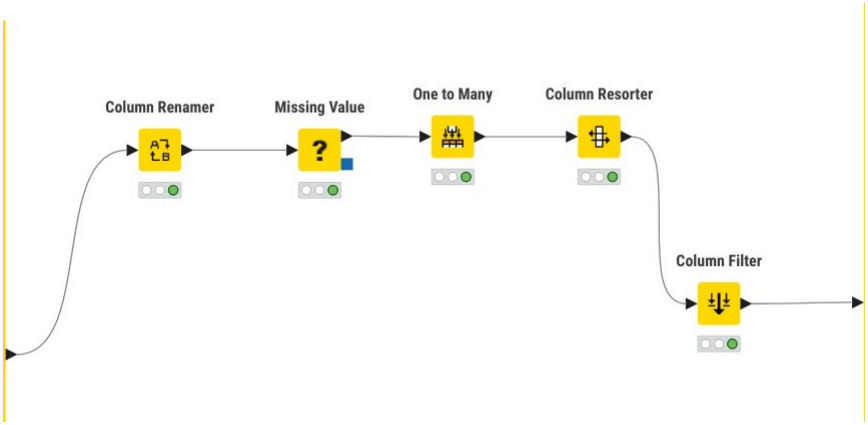
Finalmente, para los algoritmos que traten variables numéricas es necesario una normalización de los datos, pero como no es algo que afecte a todos los algoritmos no es algo que se vaya a aplicar hasta que no se llegue al entrenamiento de cada algoritmo.

Resultados obtenidos

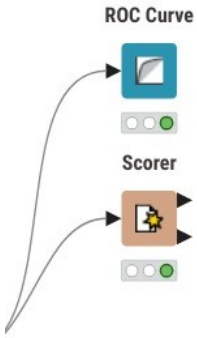
El flujo final del problema de los pasajeros es el que se puede observar a continuación. Después de leer el fichero con los datos, se ejecuta el nodo *Statistics,* para comprobar, entre otras cosas, si hay valores ausentes, además de usar el nodo *GroupBy* para conocer cuántas instancias pertenecen a cada clase.



Antes de pasar a cada algoritmo, se realiza un preprocesado básico en el que se realizan las tareas que se mencionaron en la introducción: cambiar el nombre de algunas columnas para identificar la clase, imputar los valores ausentes, pasar las variables categóricas a numéricas y, finalmente, eliminar la columna *ID*, que no va a ser relevante para la clasificación.



Por último, antes de pasar al flujo de cada algoritmo, hay que destacar el metanodo de evaluación, que en este caso es igual para todos los algoritmos: consta de un nodo *ROC Curve* y otro nodo *Scorer* para poder medir todas las medidas como el TPR, el TNR, etc.

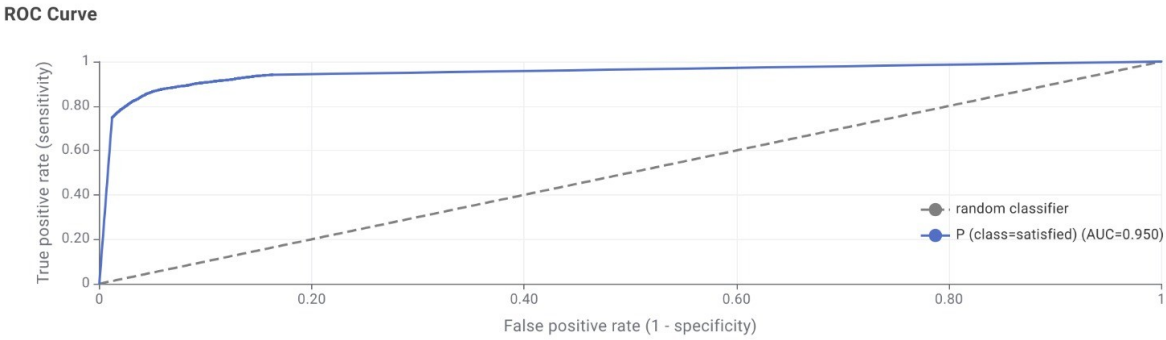


Finalmente, en esta tabla se recogen todas las medidas extraídas de cada algoritmo.

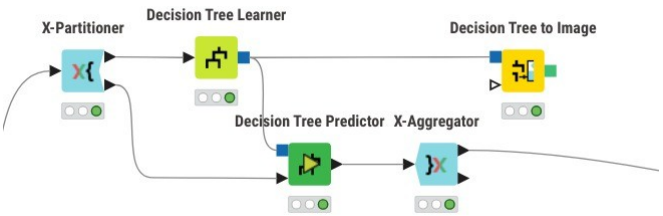
	Precisión	TPR	TNR	G-Mean	F1	AUC
K-NN	0’912	0’861	0’952	0’905	0’896	0’950
C4.5	0’937	0’927	0’945	0’936	0’928	0’944
Naïve-Bayes	0’852	0’834	0’867	0’850	0’831	0’944
Random Forest	0’958	0’962	0’956	0’957	0’952	0’919
SVM	0’594	0’075	0’999	0’274	0’139	

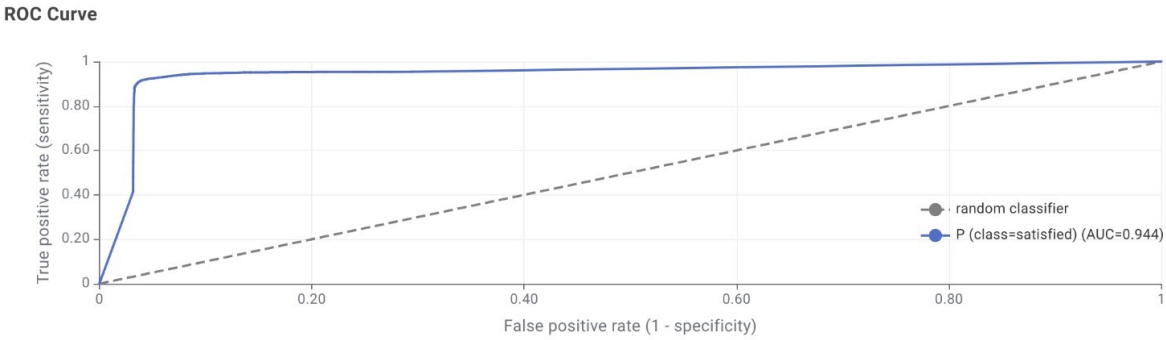
Y finalmente, el flujo específico que se ha utilizado para cada algoritmo, además de su respectiva curva ROC.

Algoritmo K-NN

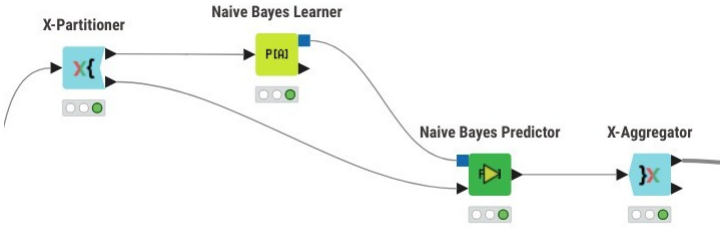


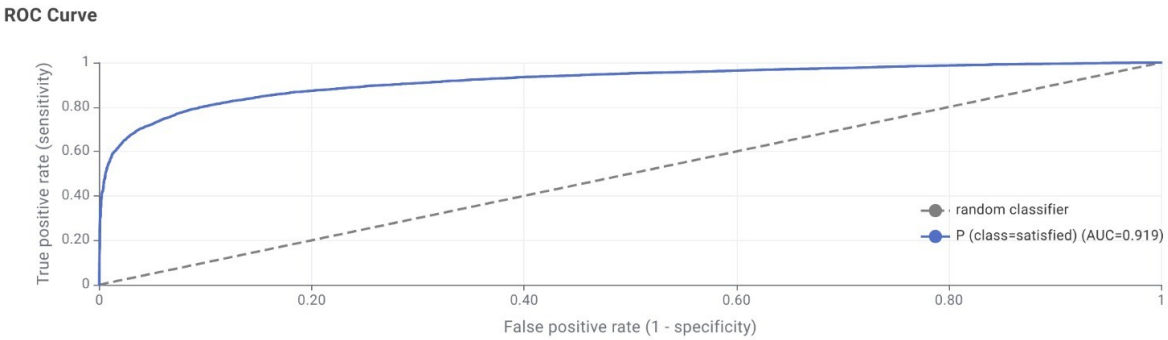
Algoritmo C4.5 (Árbol de clasificación)



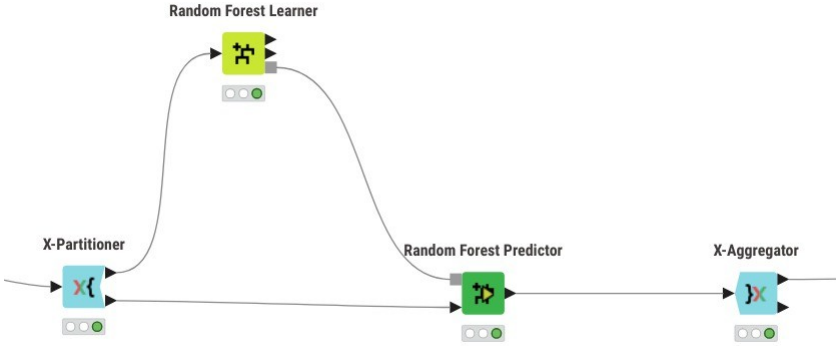


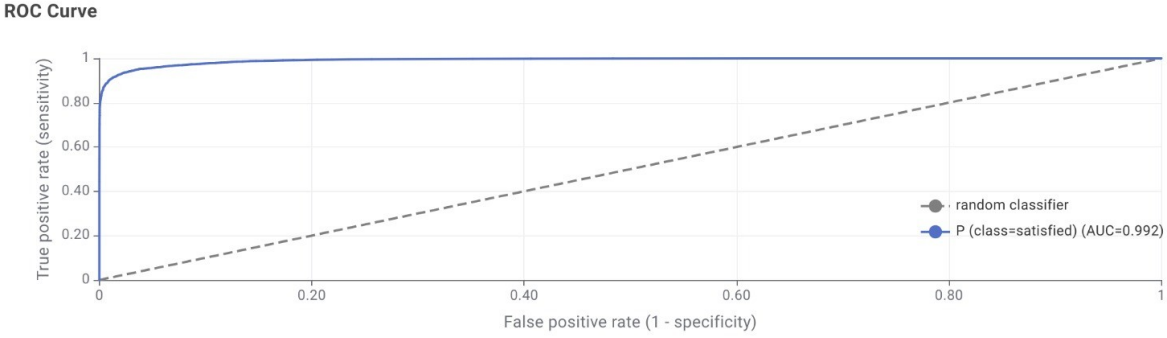
Algoritmo Naïve-Bayes



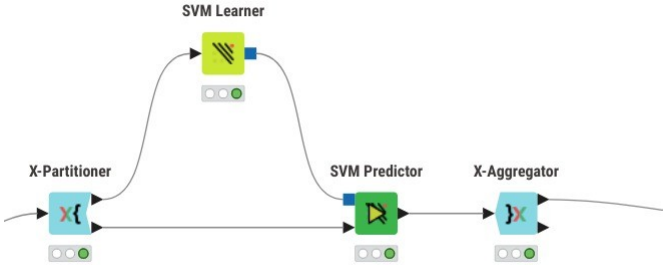


Algoritmo Random Forest





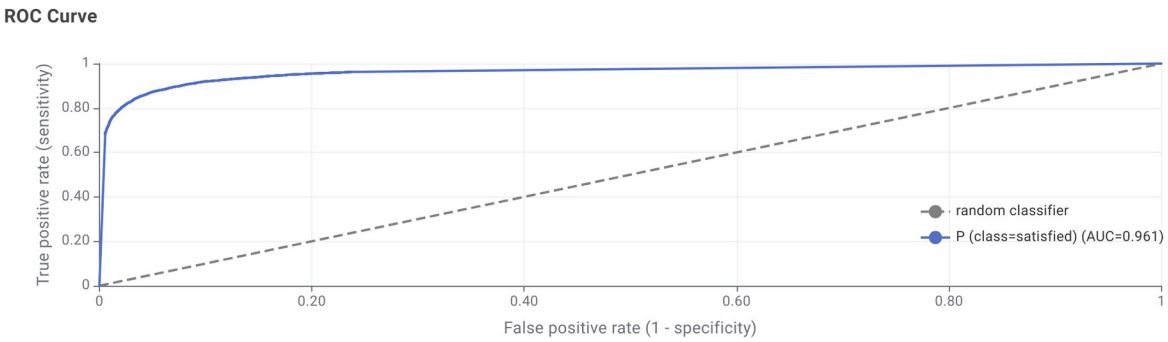
Algoritmo SVM



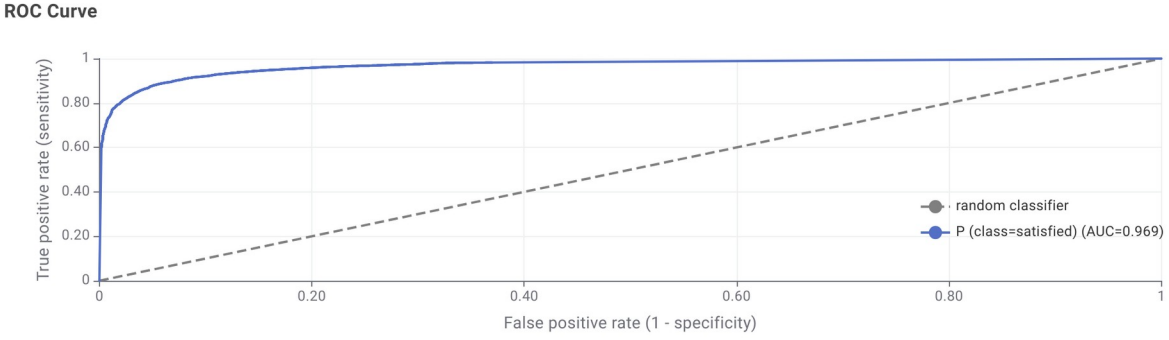
Configuración de algoritmos

Para la configuración del algoritmo K-NN, se han evaluado los 3 vecinos más cercanos, además de ponderar con pesos la distancia de cada vecino; es decir, dentro de los 3 vecinos más cercanos, se valora mejor al más cercano de los tres. Si, por ejemplo, aumentamos el número de vecinos a 5, podemos ver cómo, en este caso, los resultados obtenidos son generalmente mejores. Esto ocurre, en parte, porque no se trata de un problema excesivamente desbalanceado, lo que permite mantener una coherencia dentro de los vecinos más cercanos sin que aparezcan ejemplos de una clase incorrecta, como podría ser el caso de outliers. Aunque, eso sí, podemos apreciar que tiene una cierta tendencia conforme aumentan el número de vecinos a obtener mejores resultados en las métricas relacionadas con la clase mayoritaria a la par que aumenta la precisión. Lo mismo ocurre si aumentamos el valor de K a 10. Sin embargo, si aumentamos demasiado el valor de K, por ejemplo a 20, vemos cómo el sobreajuste puede llevar a modelos menos precisos .Por lo tanto, podemos decir que el valor ideal de K está entre 5 y 10.

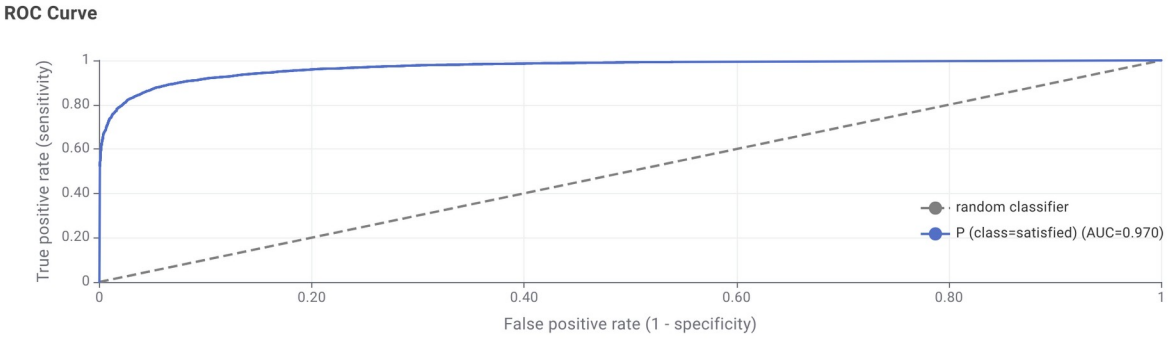
Resultados de K-NN con distintos valores de K						
	Precisión	TPR	TNR	G-Mean	F1	AUC
K = 3	0’912	0’861	0’952	0’905	0’896	0’950
K = 5	0’942	0’857	0’959	0’907	0’898	0’961
K = 10	0’947	0’854	0’963	0’907	0’898	0’969
K = 20	0’949	0’843	0’965	0’892	0’901	0’970



Curva ROC para K = 5



Curva ROC para K = 10



Curva ROC para K = 20

En el caso del árbol de decisión, se ha utilizado la razón de ganancia, de manera que el nodo *Decision Tree* de Knime actúa como C4.5.

En Naïve-Bayes se han dejado los valores por defecto, aunque podrían ser interesantes las opciones de ignorar valores ausentes (sobre todo teniendo en cuenta que, como vimos en la introducción, no es un problema en el que falten muchos datos, lo que seguiría permitiendo trabajar con una buena fuente de datos). Otro parámetro interesante podría ser el de la probabilidad por defecto, tratando de evitar que haya casos con probabilidad nula.

En Random Forest, al igual que en C4.5, se ha utilizado la razón de ganancia como criterio de división.

En SVM se ha utilizado el kernel RBF ya que utiliza la distancia euclídea para calcular la distancia entre dos puntos, algo que, normalizando los datos previamente, es relativamente sencillo sin perder eficiencia en el cálculo. Además, se ha optado por una penalización relativamente baja (la por defecto) para los errores en clasificación, con el objetivo de evitar el sobreajuste. Viendo los resultados, habría sido interesante probar con valores más altos, como 10, para provocar una mayor precisión en la clasificación.

Análisis de resultados

En primer lugar tenemos a K-NN, que a pesar de ser el más simple de todos los algoritmos utilizados y que la mayoría de sus métricas sean ligeramente inferiores a las de otros algoritmos, destaca por tener el mayor valor AUC y por que su diferencia entre el TPR y el TNR sea mayor que la del resto de algoritmos, ya que eso significa que tiende ligeramente a sobrevalorar la clase negativa (que en este caso es la mayoritaria) y asignarla más de lo habitual. De esta manera, es más probable que la tasa de falsos negativos aumente, decrementando el valor del TPR, mientras que es más difícil que aparezcan falsos positivos, de manera que el TNR aumenta más. Otros algoritmos más precisos en la clasificación no cometen este error de tender a la clase mayoritaria y tienen valores de TPR y TNR más parejos.

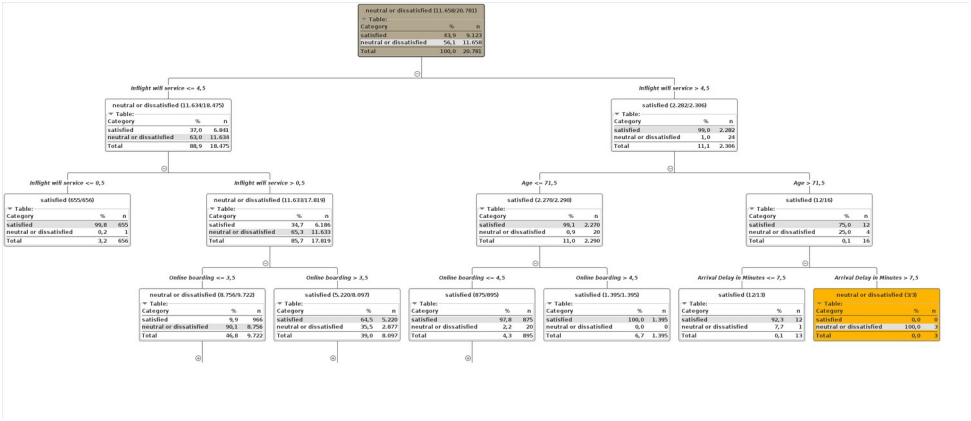
Uno de los algoritmos con mejores métricas es C4.5, lo cual se puede explicar desde el uso de la razón de ganancia para buscar aquellos atributos que generen mayor pureza en las ramas que se van creando.

Después aparece Naïve-Bayes, algo que puede tener sentido ya que, aunque su cálculo de probabilidades es robusto, el hecho de suponer que todas las variables son independientes le resta fiabilidad, ya que, por ejemplo, puede haber variables que se acaben relacionando indirectamente; por ejemplo, para un vuelo más largo puede tener más sentido comprar un billete con asientos más cómodos, o mismamente es más probable que al cliente le entren ganas de comprar comida y tenga mayor capacidad de juzgar la comida del avión que un cliente que en un vuelo corto no llega a comprar comida. Casos de este estilo, en el que las variables se pueden entrelazar, no son contemplados por Naïve-Bayes, lo que hace que su puntuación sea ligeramente menor, aunque siga siendo óptima.

Finalmente, de los algoritmos que se han escogido para este caso de estudio, se puede observar, por las medidas extraídas de la matriz de confusión, que el algoritmo que mejor funciona es, lógicamente, Random Forest, algo lógico debido a su naturaleza de combinar múltiples clasificadores, haciendo el análisis más robusto que el de otros algoritmos convencionales.

Interpretación de los datos

Para interpretar los datos, podemos analizar el modelo del árbol de decisión, que puede ser muy revelador de los atributos más relevantes para predecir si un cliente va a estar satisfecho con el servicio o no. Si analizamos el árbol generado con el algoritmo C4.5, vemos que el primer atributo que utiliza el modelo para clasificar es la calidad del WiFi durante el vuelo. Es decir, existe una gran correlación entre la calidad de la conexión a internet que ofrece el avión durante el vuelo y la satisfacción de los clientes. De hecho, no sólo lo usa para una primera división, sino que lo utiliza para realizar una segunda división con unos rangos diferentes (al ser una variable numérica puede utilizarla más de una vez reduciendo el rango de valores a comparar). Por otra parte, otra variable que encuentra relevante, especialmente si el WiFi es satisfactorio, es la edad de los clientes. Por último, para no ahondar en demasiados niveles, vemos que otra variable tremendamente importante, ya que la usa en los dos subárboles generados inicialmente, es la variable *Online boarding*.



Predicción de la Calidad del Aire y de la Contaminación

Introducción

Este conjunto de datos representa la calidad del aire en diferentes localizaciones geográficas. Contiene 5.000 instancias con 10 variables que muestran información como diferentes concentraciones de contaminantes, condiciones meteorológicas y determinados factores demográficos.

En este caso, la mayoría de los datos son numéricos, ya que la mayoría de las variables miden concentraciones de gases, además de distancia a áreas industriales. Sin embargo, la variable clase aparece como una variable categórica. En este caso, no nos enfrentamos a un problema binario, ya que los posibles valores a predecir son cuatro: bueno, moderado, pobre y peligroso. De esta manera, al ser el único problema multiclase, va a requerir adaptar la evaluación de los clasificadores, pudiendo ser necesarias algunas transformaciones.

Antes de proceder al análisis de los datos, es posible que haya que realizar un preprocesamiento para tratar valores perdidos, así como un posible desbalanceo por una mayoría de muestras clasificadas como “bueno” o “moderado”.

Procesado de datos

Este problema apenas requiere preprocesar los datos: como se puede comprobar con el nodo *Statistics* no hay valores ausentes y todos los atributos, a excepción de la variable clase, son numéricos (lo cual no deja lugar a la duda de alguna conversión de tipo, o de codificación en caliente para pasar de variables categóricas a variables numéricas. De esta manera, la única tarea relevante para el análisis es la normalización de los datos numéricos, pero por lo demás, más allá de renombrar la columna de la variable clase, no es necesario realizar nada más. En cualquier caso, se podría cuestionar si las distintas clases están balanceadas o no, pero dado que hay una proporción equitativa entre todas las clases, podría no ser necesario tener que generar ejemplos simbólicos mediante SMOTE, por ejemplo, aunque no es una opción descartable.

Resultados obtenidos

Configuración de algoritmos

Análisis de resultados

Interpretación de los datos