# DEFINICIÓN DEL PROBLEMA

Se presenta un problema donde en un futuro lejano, una nave espacial Titanic, con casi 13.000 pasajeros a bordo, emprende su viaje inaugural hacia tres exoplanetas habitables. Mientras rodeaba Alpha Centauri en ruta hacia su primer destino, la nave espacial Titanic chocó con una anomalía del espacio-tiempo escondida dentro de una nube de polvo. La nave permaneció intacta, sin embargo, casi la mitad de los pasajeros fueron transportados a una dimensión alternativa.

El objetivo del algoritmo será predecir dado un individuo (si hubiera estado en esa situación) si sería transportado o no.

# VISUALIZACION y

# Selección de variables

En la visualización/exploración de nuestro conjunto de datos podemos extraer la siguiente información:

* El conjunto de datos es información sobre los pasajeros a bordo de la nave espacial Titanic.
* Se obtuvo la información personal de aproximadamente 8700 pasajeros.
* La variable de estudio es representada por si el pasajero fue transportado a otra dimensión o no.
* Los datos recogen 8693 individuos de los cuales 4315 no fueron transportados a otra dimensión y 4378 que sí que lo fueron.
* Que cada individuo tiene 14 atributos/variables, 7 nominales, 6 numéricos y la variable respuesta:
* **@attribute 'PassengerId':** Una identificación única para cada pasajero. Cada Id toma la forma gggg\_pp donde gggg indica un grupo con el que viaja el pasajero y pp es su número dentro del grupo. Las personas en un grupo a menudo son miembros de la familia, pero no siempre.
* **@attribute 'HomePlanet':** Planeta del que parte el pasajero.
* **@attribute 'CryoSleep':** Indica si el pasajero eligió ser puesto en animación suspendida durante la duración del viaje. Los pasajeros en crio sueño están confinados en sus cabinas.
* **@attribute 'Cabin':** El número de cabina donde se hospeda el pasajero. Toma la forma cubierta/número/lado, donde lado puede ser P para babor o S para estribor.
* **@attribute 'Destination':** El planeta en el que desembarcará el pasajero.
* **@attribute 'Age':** Edad del pasajero.
* **@attribute 'VIP'** : Si el pasajero es VIP o no.
* **@attribute 'RoomService', 'FoodCourt', 'ShoppingMall', 'Spa', 'VRDeck'**: Dinero facturado en cada uno de los servicios de lujo
* **@attribute 'Name':** Los nombres y apellidos del pasajero/a.
* **@attribute 'Transported':** Si el pasajero fue transportado a otra dimensión o no.

Version1

* + Las variables que borraremos serán Destination, RoomService, FoodCourt, ShoppingMall, Spa y VRDeck y Name (dinero gastado, el nombre y el destino de un pasajero) debido a que son variables que estamos seguros de que no van a influenciar en nada a que un pasajero se transporte o no.
  + El resto de variables hemos decidido tratarlas para su posterior análisis. Como desconocemos totalmente los motivos por lo que un individuo se transporta o no, no queremos descartar las diferentes teorías por muy extrañas que sean. Por lo que trabajaremos con ellas, y finalmente, en la poda podremos concretar y descartar estas opciones. Principalmente barajamos dos teorías que podrían influir en el destino de un pasajero:
* Más probable: La localización en el momento del impacto
* Menos probable: Las condiciones físicas y la tendencia genética a la transportación.

Sospechamos que el motivo más probable de que un individuo se transporte o no será su localizacion en la nave en el momento del impacto. Por eso hemos decidido dejar las variables que pueden tener relación con esta; PassengerId, CryoSleep, Cabin y VIP.

Sin embargo, también pensamos que el destino de un individuo también puede depender de alguna forma de su genética/rasgos físicos, por lo que hemos creído oportuno dejar las variables: PassengerId, HomePlanet y Age. Características físicas como por ejemplo son la altura o el peso, capacidades físicas como la velocidad para escapar, la fuerza para agarrase a las cosas… O simplemente la tendencia genética a ser transportado a otra dimensión. (que no tenemos ni idea si podría influir).

Es destacable el motivo por el que creemos que la variable PassengerId es muy importante, ya que extrayendo unicamente la informacion de la familia a la que permanece un individuo, esta nos puede arrojar información sobre su localización (entendemos que una persona tiene más probabilidades de moverse por la nave acompañado de sus allegados, que en solitarioo con gente sin ninguna relación), y también sobre sus carcterísticas genéticas.

Parecido (con respecto a la ubicación )con los que pensamos de la variable VIP, que creemos que un individuo de una determinada clase social tiene más probabilidades de estar con sus iguales en zonas reservadas para estos.

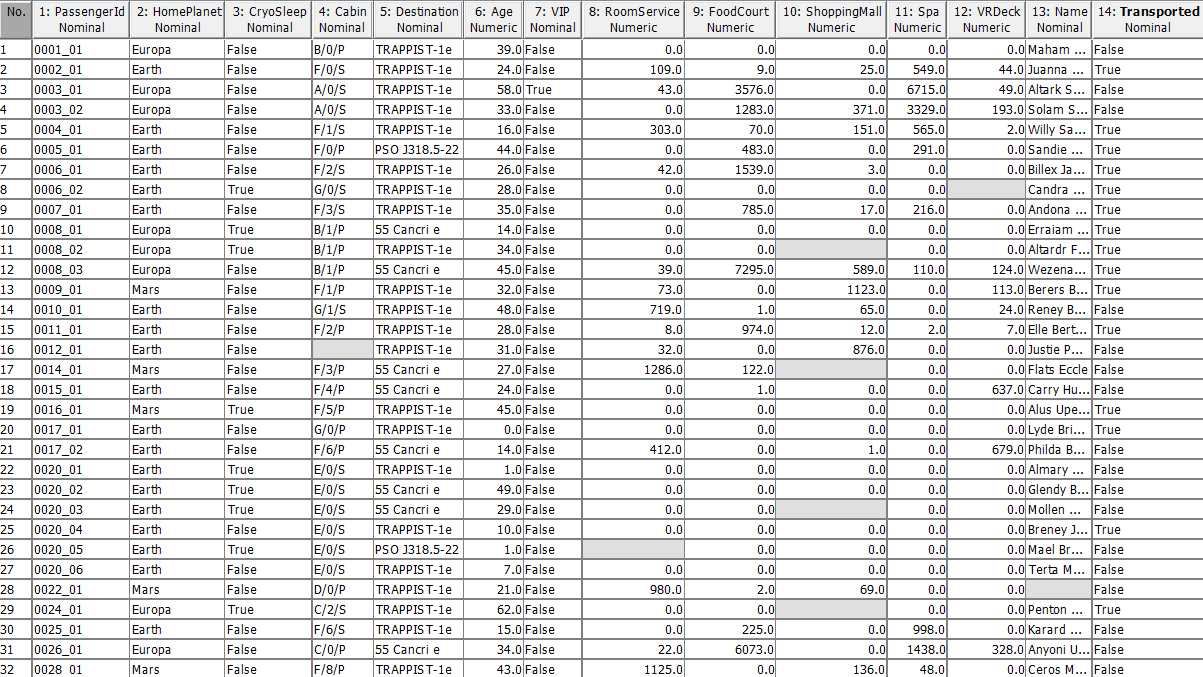


Imagen 1. Conjunto de datos sin las variables que hemos considerado innecesarias borradas.

Interfaz de usuario gráfica, Tabla

Descripción generada automáticamente

Imagen 2. Variables a eliminar y eliminación.



* Para eliminar las variables que hemos considerado innecesarias hemos seguido el siguiente proceso:



**Pantalla Preprocess → Seleccionar variables → Remove.**

Version2 (definitiva)

La versión 2 ( con la que trabajaremos finalmente), mantiene los cambios realizados en la versión 1, pero además elimina las variables Cabin y IdPassenger.

Hemos decidido esto debido a que Weka no dispone de opciones prediseñadas para generar nuevos atributos a partir de atributos ya existentes (ingeniería de características), sino que habría que programar sobre weka o exportar los datos como csv y realizarlo externamente. Por eso, aunque seguimos con el concepto inicial, trabajaremos solo en weka con estas variables y ya en el proyecto final de la asignatura profundizaremos más sobre esto.

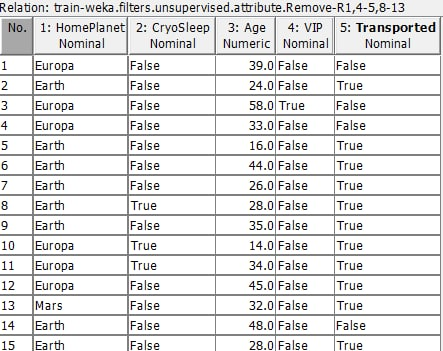


Imagen X: Variables con las que trabajaremos finalmente.

# Preprocesado

Imputación de valores perdidos

La imputación de variables se va a realizar sobre los valores perdidos. A estos se les va a aplicar un filtro donde los valores perdidos numéricos se van a sustituir por la media y los valores perdidos nominales por la moda.

Explicacion: Cuando en los datos que manejamos hay algunos valores de atributos faltantes, tenemos dos formas de operar: una es borrando el individuo entero y la otra es imputando los valores faltantes.

Si borras el individuo entero te ahorras el tiempo de su procesado pero sin embargo pierdes información, mientras que si imputas los valores faltantes, tienes que dedicarle tiempo a realizar esto pero a cambio evitas perder el resto de la información.

Al imputar los valores, lo realizamos introduciendo la media o la moda, ya que lo que buscamos es poner una especie de parche que salve al individuo entero. Y este parche será mejor cuanto mas desaprecibido pase (ya que es un valor ficticio) por eso lo hacemos con la idea de los valores más comunes (la media y la moda).

**Filter → Unsupervised → Attribute → ReplaceMissingValues**.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente



Imagen 3. Valores perdidos del atributo HomePlanet antes de aplicar la imputación.

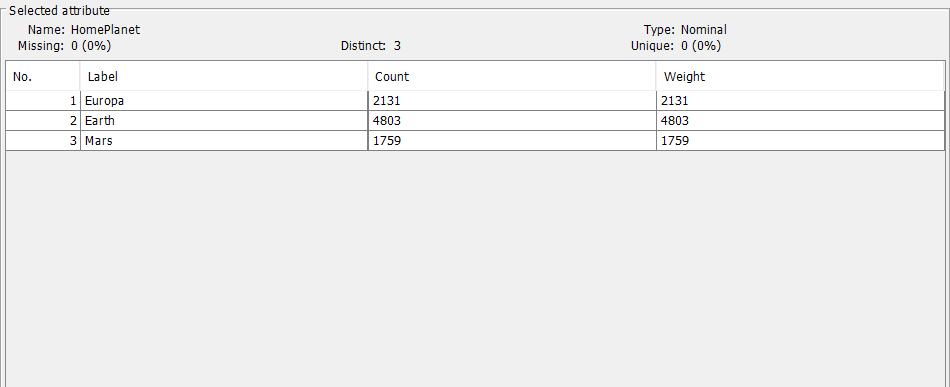




Imagen 3. Variables perdidos de la variable HomePlanet despues de aplicar la imputación.

Tratar variables categóricas/nominales🡪BINARIZACION

Las variables categóricas las hemos tratado de la siguiente manera:

* En primer lugar, hemos imputado los valores perdidos como hemos explicado anteriormente.
* Vamos a tratar las variables HomePlanet, CryoSleep y VIP como variables numéricas binarias (vamos a binarizarlas) aplicando el método de One Hot Encoding. Ya que se tratan de 3 variables numerales sin orden (no ordinales), lo que significa que la diferencia entre individuos con respecto al atributo será unicamente que el valor sea distinto.

Explicación: Cuando binarizamos una variable categórica creamos una columna (atributo) por cada “categoría” de la variable, los cuales tendrán valor 0 o 1 en función del valor de su variable categórica inicial. Con esto conseguimos entregarle solo variables numéricas al algoritmo, y además ya normalizadas (por lo general entre 0 y 1).

*En el cado que sean únicamente dos categorías se crea una columna solo, 1 si es un valor y 0 si es el otro valor*

Imagen que contiene Patrón de fondo

Descripción generada automáticamente

Imagen 4. Aplicación de nominal a binario de las variables nominales.

* El proceso de One Hot Encoding se realiza de la siguiente manera:

**Filter → Unsupervised → Attribute → NominaltoBinary**

Discretización

Explicación: Pasar valores numéricos continuos a discretos (dividirlos en intervalos/grupos). Principalmente hay dos formas de hacerlo:

A rangos de igual tamaño, por ejemplo, dividir la edad en 5 intervalos de 20 años de tamaño.

A rangos igualmente poblado, donde los intervalos serán de distinto tamaño, pero tendrás el mismo número de individuos en cada intervalo.

En nuestro conjunto de datos solo sería interesante hacerlo para la variable edad, sin embargo, la discretización creemos que es algo más voluntario y que realmente no nos aportará mas información, por lo que hemos decidido no aplicarla.

Normalización

Es necesario normalizar para la aplicación de algoritmos como KNN, donde la distancia entre individuos es fundamental. Normalizar hace posible que la semejanza entre individuos sea justa y equitativa entre los atributos.

Hemos normalizado la edad ya que es la única variable numérica como tal. Los otros atributos son nominales y se les ha aplicado One Hot Encoding, por lo que van a estar dentro de ese rango de normalización.

* El proceso de normalización se realiza de la siguiente manera:

**Filter → Unsupervised → Attribute →Normalize**.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente



Figura x. Función para normalizar las variables.

**Imagen que contiene edificio

Descripción generada automáticamente**Una vez preprocesados los datos, llegamos a que los datos definitivos y con los que vamos a trabajar son los siguientes:

# Entrenamiento de modelos// Optimización de hiperparámetros

Vamos a realizar una serie de experimentos, con el objetivo de obtener el mejor modelo predictivo posible. Para ello vamos a aplicar varios algoritmos a nuestro conjunto de datos.

Y por cada clasificador aplicado iremos añadiendo imágenes sobre las diferentes versiones de estos, y como varían los rendimientos.

Al escoger un conjunto por percentage split se va a escoger un conjunto de prueba y un conjunto de entrenamiento aleatorios por lo que los resultados van a ser menos fiables.

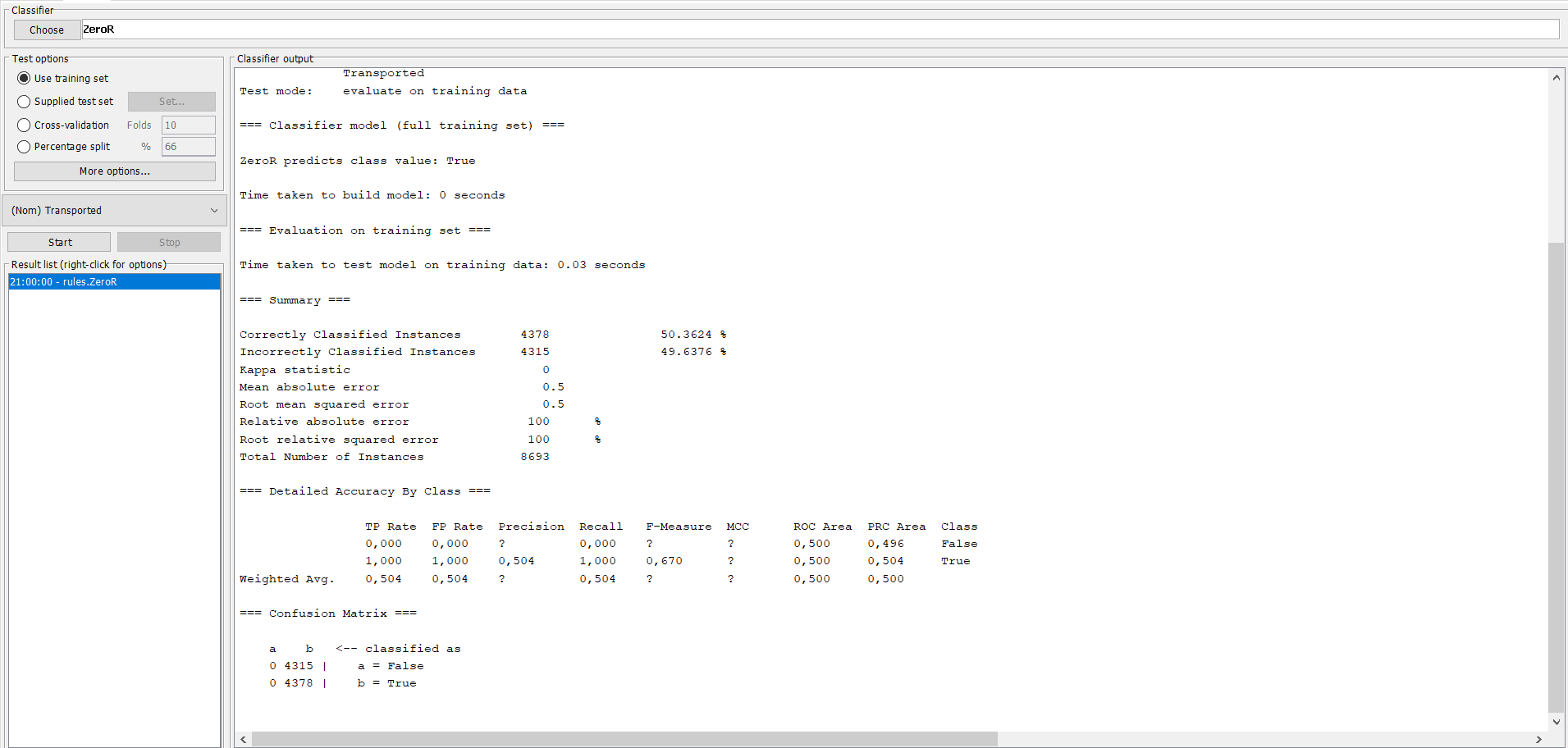
Sin embargo, al obtener el redimiento por validación cruzada va a coger una serie de conjuntos de entrenamiento y una serie de conjuntos de prueba (10 en nuestro caso) y se le va a hacer la media, por lo que va a ser mucho más fiable.

ZeroR

ZeroR es un clasificador que clasifica datos por el valor de la variable respuesta más común en el conjunto de entrenamiento. Si por ejemplo en nuestro conjunto de entrenamiento lo más común es que NO se haya transportado, predice cualquier individuo de prueba como que NO se transportará.

ZeroR lo vamos a emplear como clasificador baseline antes de probar el resto.

*(con baseline nos referimos a valor inicial a partir del cual pueden compararse valores posteriores de lo que se está midiend.)*



Interpretación de la matriz de confusión:

En la matriz de confusión se puede apreciar fácilmanete la predicción del algoritmo:

De 4315 individuos que No fueron transportados, clasifica todos a que SI lo fueron 🡪 4315 FALLOS

De 4378 ind que SI fueron transportados, clasifica a todos como que SI lo fueron 🡪 4378 ACIERTOS

Las evalúa todas a SI.

Evidentemente en este clasificador existe un gran problema de desbalanceo de de clases. (Aunque en el rendimiento 53% no se aprecia notablemente debido a que el conjunto de prueba esta bastante equitativo). Por lo que no se ve con claridad como de torpe es este clasificador, para ver un rendimiento mucho más “justo” existe la precisión balanceada.

Que en este caso saldría un 50%, ya que tendría un 100% de efectividad para los SI y un 0% de efectividad para los NO.

J48

En el caso del J48 es un clasificador de tipo árbol. El objetivo fundamental de este es predecir la variable respuesta de individuos pruebas a través de reglas que ha sacado anteriormente del conjunto de entrenamiento. Estas reglas se basan en los patrones que existen dentro de los datos de entrenamiento.

En nuestro clasificador no se aprecia sobre ajuste. Debido a que el rendimiento con el conjunto de prueba es muy similar al rendimiento con el conjunto de prueba.

La poda es un proceso que se le aplica a los clasificadores de tipo árbol que consiste en iterativamente ir cortando nodos y observando su rendimiento de acierto con esto. El objetivo de este proceso es simplificar el árbol, y al igual que pasa con la acotación del número minimo de individuos en las hojas, evitar que el clasificador sobreaprenda y caiga en individualidades, perdiendo así patrones del gran volumen de los datos.

J48/Unpruned=True/NminObj=2/ Conjunto de entrenamiento J48

Diagrama

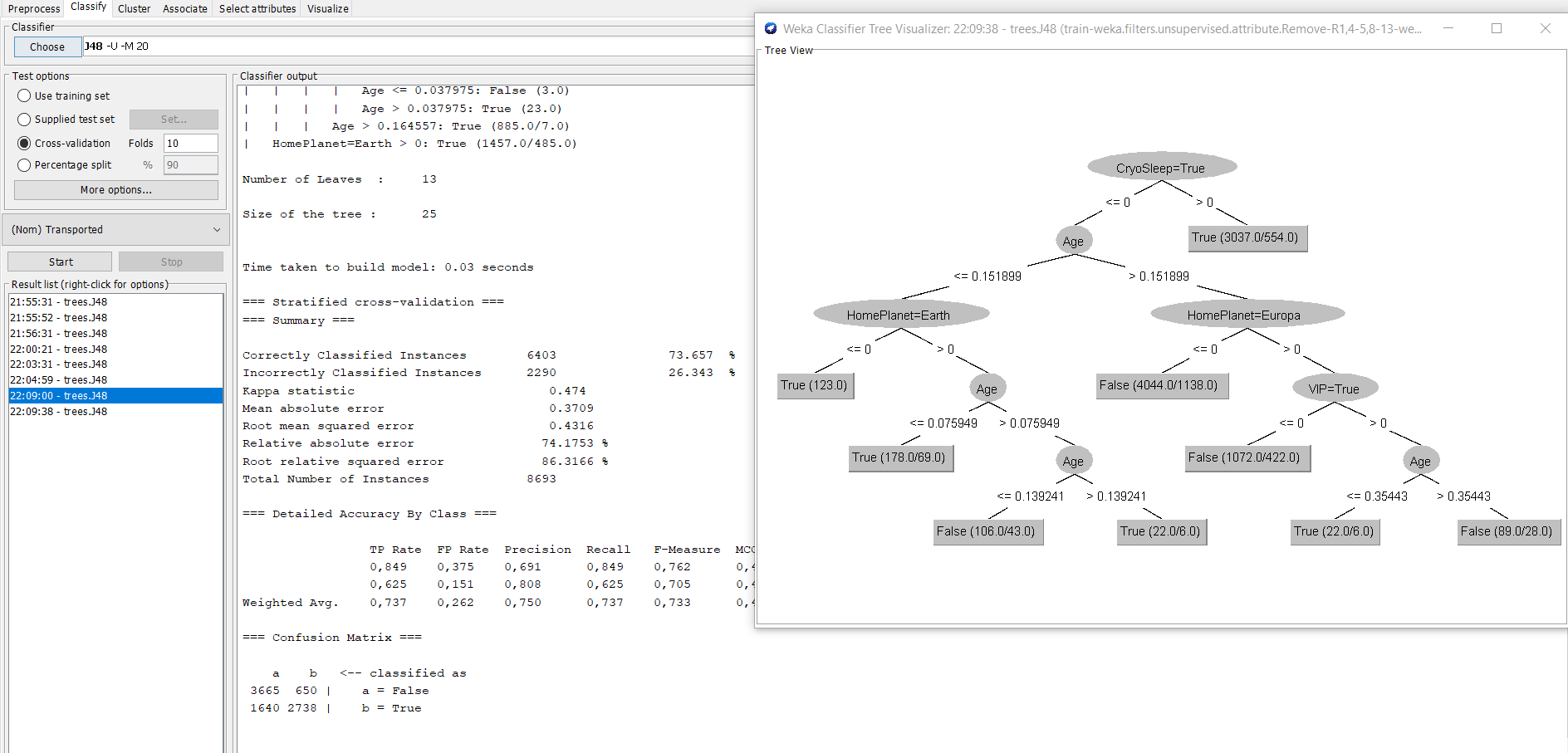
Descripción generada automáticamente

J48/Unpruned=True/NminObj=2/CrossValidation

Diagrama

Descripción generada automáticamente

J48/Unpruned=True/NminObj=20/CrossValidation



J48/Unpruned=False/NminObj=2/CrossValidation (Podado)

Diagrama

Descripción generada automáticamente

Hiperparametros J48/confidenceFactor de 0.1 a 0.5 con 5 pasos/NumMinObj de 1 a 21 con 5 pasos

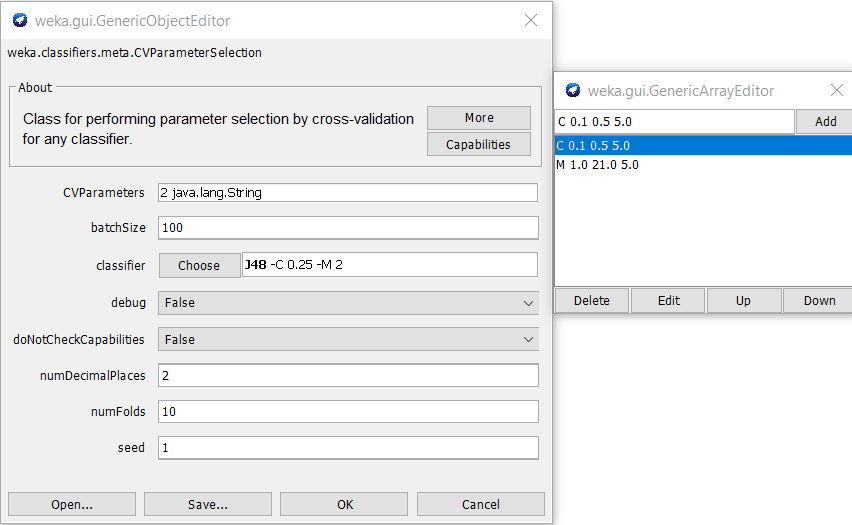


Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

KNN

KNN es un tipo de clasificador que para predecir la variable objetivo de un individuo prueba se basa en las semejanzas (a través de distancias) con K individuos de entrenamientos y sus variables respuesta.

Para aplicar este algoritmo es muy importante que los datos estén normalizados 🡪 esto consigue que en las distancias entre los individuos no se ponderen algunas variables por encima de otras, sino que todas tengan un peso equitativo en la decisión.

Hay que tener en cuenta que el valor de k es crucial para el buen funcionamiento del algoritmo. Un valor de K alto te proporcionará una mayor seguridad en tu predicción ya que será más dificil que caigas en individualidades, mientras que si este K es demasiado alto, pecarás de que las distancias de los individuos seleccionados para la decisión serán mayores => menos certeza (datos más diferentes)

También podemos destacar que lo ideal es que k sea un valor impar, y así evitar un posible empate.

Por ejemplo si fuera k=5, y las variables respuesta de los 5 más semejantes fueran SI,NO,NO,SI y SI. KNN predeciría el individuo prueba como SI.

IBk(knn)/k=3/Distancia Euclidea/Percentage Split

Imagen que contiene Tabla

Descripción generada automáticamente

IBk(knn)/k=3/Distancia Euclidea/CrossValidation

Tabla

Descripción generada automáticamente

IBk(knn)/k=5/Distancia Euclidea/CrossValidation

Tabla

Descripción generada automáticamente

IBk(knn)/k=7/Distancia Euclidea/CrossValidation

Tabla

Descripción generada automáticamente

Hiperparametros IBk/k(vecinos mas cercanos) de 1 a 9 con 5 pasos

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

Descripción generada automáticamente

Tabla

Descripción generada automáticamente

(El mejor rendimiento se ha obtenido con 9 vecinos)

Naive Bayes

Naive Bayes es un algoritmo de clasificación de Aprendizaje Automático.

En este algoritmo se asume que las variables predictoras son independientes entre sí. En otras palabras, que la presencia de una cierta característica en un conjunto de datos no está en absoluto relacionada con la presencia de cualquier otra característica.

Proporcionan una manera fácil de construir modelos con un comportamiento muy bueno debido a su simplicidad.

Lo consiguen proporcionando una forma de calcular la probabilidad ‘posterior’ de que ocurra un cierto evento A, dadas algunas probabilidades de eventos ‘anteriores’.

NaiveBayesMultinomial/Percentage Split

Gráfico

Descripción generada automáticamente

NaiveBayesMultinomial/CrossValidation

Tabla

Descripción generada automáticamente

Clustering

El clastering es una técnica de clasificación de datos de aprendizaje no supervisado. Sirve fundamentalmente para agrupar los datos según similitud.

Pestaña Clusters->SimpleKMeans->use training set->Start (valores por defecto -> grupos=2)

Interfaz de usuario gráfica

Descripción generada automáticamente

Como se aprecia en la imagen, el algoritmo te clasifica los datos en dos grupos con una proporción del 65%/35% de la población.

La interpretación que puedo dar sobre este resultado la realizo comparando con las gráficas de la variable respuesta y sus proporciones de población:

Es que el valor de la variable respuesta no es debido a todo el resto de atributos por igual, sino que el valor de algunos atributos es bastante más influyente que otros en que un individuo sea transportado o no. Si fuera así, veríamos que la proporción entre la población de los clusters y la proporción de la variable respuesta sería similar, ya que los clusters se calculan a través de las distancias entre individuos.

Interfaz de usuario gráfica

Descripción generada automáticamente

# Comparación de clasificadores