Alejandro Inglés, Israel Brea

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.

No.	1: PassengerId Nominal	2: HomePlanet Nominal	3: CryoSleep Nominal	4: Cabin Nominal	5: Destination Nominal	6: Age Numeric	7: VIP Nominal	8: RoomService Numeric	9: FoodCourt Numeric	10: ShoppingMall Numeric	11: Spa Numeric	12: VRDeck Numeric	13: Name Nominal	14: **Transported** Nominal
1	0001_01	Europa	False	B/0/P	TRAPPIST-1e	39.0	False	0.0	0.0	0.0	0.0	0.0	Maham ...	False
2	0002_01	Earth	False	F/0/S	TRAPPIST-1e	24.0	False	109.0	9.0	25.0	549.0	44.0	Juanna ...	True
3	0003_01	Europa	False	A/0/S	TRAPPIST-1e	58.0	True	43.0	3576.0	0.0	6715.0	49.0	Altark S...	False
4	0003_02	Europa	False	A/0/S	TRAPPIST-1e	33.0	False	0.0	1283.0	371.0	3329.0	193.0	Solam S...	False
5	0004_01	Earth	False	F/1/S	TRAPPIST-1e	16.0	False	303.0	70.0	151.0	565.0	2.0	Willy Sa...	True
6	0005_01	Earth	False	F/0/P	PSO J318.5-22	44.0	False	0.0	483.0	0.0	291.0	0.0	Sandie ...	True
7	0006_01	Earth	False	F/2/S	TRAPPIST-1e	26.0	False	42.0	1539.0	3.0	0.0	0.0	Billex Ja...	True
8	0006_02	Earth	True	G/0/S	TRAPPIST-1e	28.0	False	0.0	0.0	0.0	0.0		Candra ...	True
9	0007_01	Earth	False	F/3/S	TRAPPIST-1e	35.0	False	0.0	785.0	17.0	216.0	0.0	Andona ...	True
10	0008_01	Europa	True	B/1/P	55 Cancri e	14.0	False	0.0	0.0	0.0	0.0	0.0	Erraiam ...	True
11	0008_02	Europa	True	B/1/P	TRAPPIST-1e	34.0	False	0.0	0.0		0.0	0.0	Altardr F...	True
12	0008_03	Europa	False	B/1/P	55 Cancri e	45.0	False	39.0	7295.0	589.0	110.0	124.0	Wezena...	True
13	0009_01	Mars	False	F/1/P	TRAPPIST-1e	32.0	False	73.0	0.0	1123.0	0.0	113.0	Berers B...	True
14	0010_01	Earth	False	G/1/S	TRAPPIST-1e	48.0	False	719.0	1.0	65.0	0.0	24.0	Reney B...	False
15	0011_01	Earth	False	F/2/P	TRAPPIST-1e	28.0	False	8.0	974.0	12.0	2.0	7.0	Elle Bert...	True
16	0012_01	Earth	False		TRAPPIST-1e	31.0	False	32.0	0.0	876.0	0.0	0.0	Justie P...	False
17	0014_01	Mars	False	F/3/P	55 Cancri e	27.0	False	1286.0	122.0		0.0	0.0	Flats Eccle	False
18	0015_01	Earth	False	F/4/P	55 Cancri e	24.0	False	0.0	1.0	0.0	0.0	637.0	Carry Hu...	False
19	0016_01	Mars	True	F/5/P	TRAPPIST-1e	45.0	False	0.0	0.0	0.0	0.0	0.0	Alus Upe...	True
20	0017_01	Earth	False	G/0/P	TRAPPIST-1e	0.0	False	0.0	0.0	0.0	0.0	0.0	Lyde Bri...	True
21	0017_02	Earth	False	F/6/P	55 Cancri e	14.0	False	412.0	0.0	1.0	0.0	679.0	Philda B...	False
22	0020_01	Earth	True	E/0/S	TRAPPIST-1e	1.0	False	0.0	0.0	0.0	0.0	0.0	Almary ...	False
23	0020_02	Earth	True	E/0/S	55 Cancri e	49.0	False	0.0	0.0	0.0	0.0	0.0	Glendy B...	False
24	0020_03	Earth	True	E/0/S	55 Cancri e	29.0	False	0.0	0.0		0.0	0.0	Mollen ...	False
25	0020_04	Earth	False	E/0/S	TRAPPIST-1e	10.0	False	0.0	0.0	0.0	0.0	0.0	Breney J...	True
26	0020_05	Earth	True	E/0/S	PSO J318.5-22	1.0	False		0.0	0.0	0.0	0.0	Mael Br...	False
27	0020_06	Earth	False	E/0/S	TRAPPIST-1e	7.0	False	0.0	0.0	0.0	0.0	0.0	Terta M...	False
28	0022_01	Mars	False	D/0/P	TRAPPIST-1e	21.0	False	980.0	2.0	69.0	0.0	0.0		False
29	0024_01	Europa	True	C/2/S	TRAPPIST-1e	62.0	False	0.0	0.0		0.0	0.0	Penton ...	True
30	0025_01	Earth	False	F/6/S	TRAPPIST-1e	15.0	False	0.0	225.0	0.0	998.0	0.0	Karard ...	False
31	0026_01	Europa	False	C/0/P	55 Cancri e	34.0	False	22.0	6073.0	0.0	1438.0	328.0	Anyoni U...	False
32	0028_01	Mars	False	F/8/P	TRAPPIST-1e	43.0	False	1125.0	0.0	136.0	48.0	0.0	Ceros M...	False

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

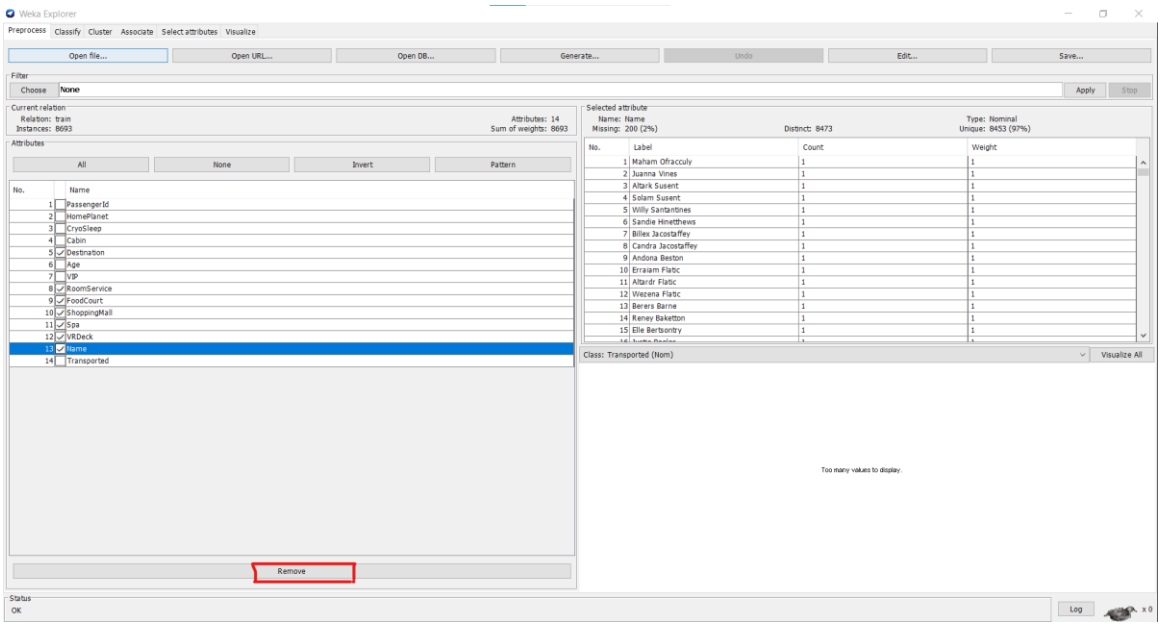


Imagen 2. Variables a eliminar y eliminación.

- 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.

Relation: train-weka.filters.unsupervised.attribute.Remove-R1,4-5,8-13

No.	1: HomePlanet Nominal	2: CryoSleep Nominal	3: Age Numeric	4: VIP Nominal	5: **Transported** Nominal
1	Europa	False	39.0	False	False
2	Earth	False	24.0	False	True
3	Europa	False	58.0	True	False
4	Europa	False	33.0	False	False
5	Earth	False	16.0	False	True
6	Earth	False	44.0	False	True
7	Earth	False	26.0	False	True
8	Earth	True	28.0	False	True
9	Earth	False	35.0	False	True
10	Europa	True	14.0	False	True
11	Europa	True	34.0	False	True
12	Europa	False	45.0	False	True
13	Mars	False	32.0	False	True
14	Earth	False	48.0	False	False
15	Earth	False	28.0	False	True

Imagen X: Variables con las que trabajaremos finalmente.

Preprocesado

Imputación de valores perdidos

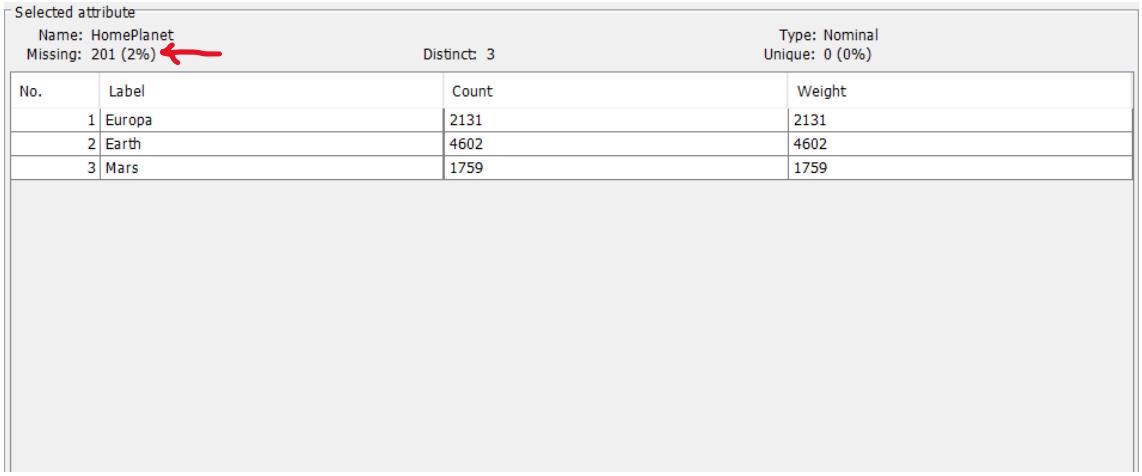
La imputación de variables se va a realizar sobre los valores perdidos. A estos se les va a aplicar un filtro donde los valores perdidos numéricos se van a sustituir por la media y los valores perdidos nominales por la moda.

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).

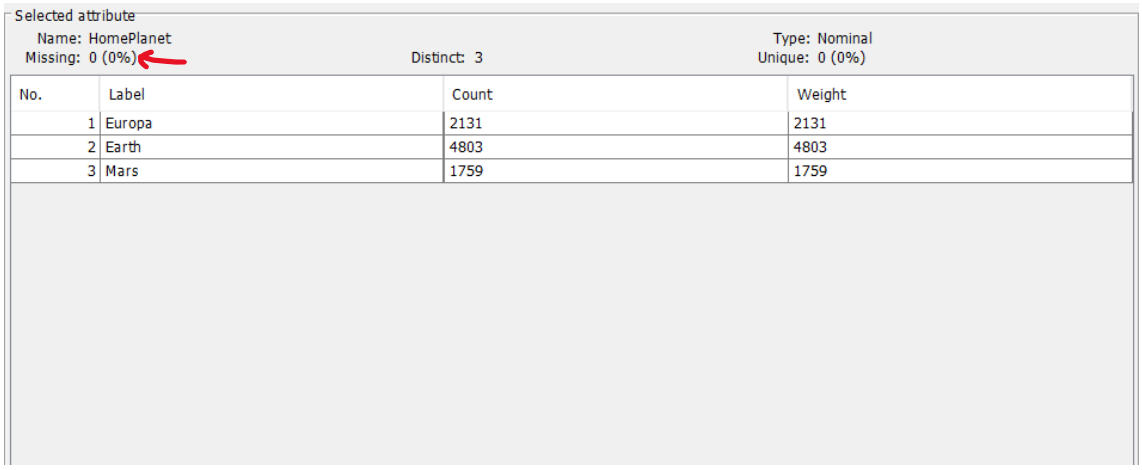
Filter → Unsupervised → Attribute → ReplaceMissingValues.


Selected attribute

Name: HomePlanet
Missing: 201 (2%)
Distinct: 3
Type: Nominal
Unique: 0 (0%)

No.	Label	Count	Weight
1	Europa	2131	2131
2	Earth	4602	4602
3	Mars	1759	1759

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


Selected attribute

Name: HomePlanet
Missing: 0 (0%)
Distinct: 3
Type: Nominal
Unique: 0 (0%)

No.	Label	Count	Weight
1	Europa	2131	2131
2	Earth	4803	4803
3	Mars	1759	1759

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

No.	Name
1	HomePlanet=Europa
2	HomePlanet=Earth
3	HomePlanet=Mars
4	CryoSleep=True
5	Age
6	VIP=True
7	Transported

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.

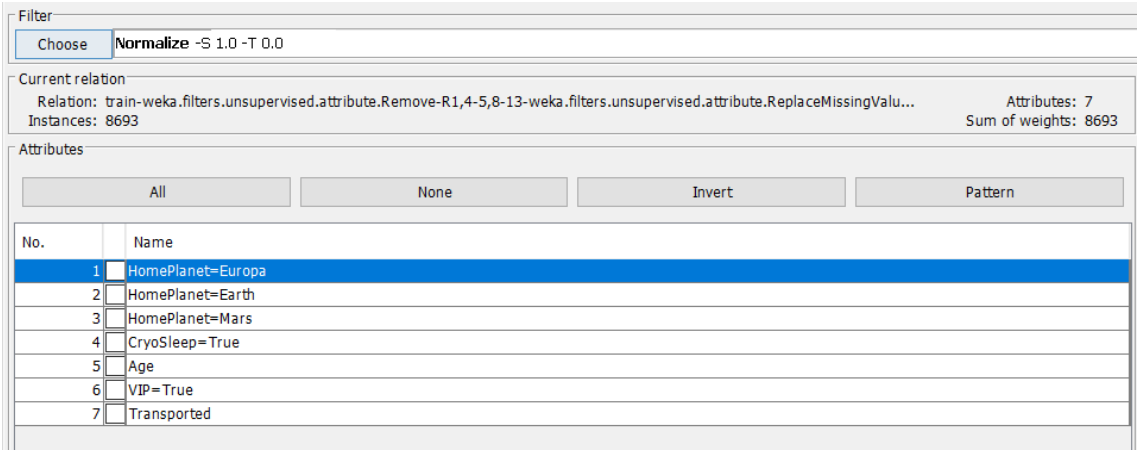


Figura x. Función para normalizar las variables.

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

No.	1: HomePlanet=Europa Numeric	2: HomePlanet=Earth Numeric	3: HomePlanet=Mars Numeric	4: CryoSleep=True Numeric	5: Age Numeric	6: VIP=True Numeric	7: **Transported** Nominal
1	1.0	0.0	0.0	0.0	0.4936...	0.0	False
2	0.0	1.0	0.0	0.0	0.3037...	0.0	True
3	1.0	0.0	0.0	0.0	0.7341...	1.0	False
4	1.0	0.0	0.0	0.0	0.4177...	0.0	False
5	0.0	1.0	0.0	0.0	0.2025...	0.0	True
6	0.0	1.0	0.0	0.0	0.5569...	0.0	True
7	0.0	1.0	0.0	0.0	0.3291...	0.0	True
8	0.0	1.0	0.0	1.0	0.3544...	0.0	True
9	0.0	1.0	0.0	0.0	0.4430...	0.0	True
10	1.0	0.0	0.0	1.0	0.1772...	0.0	True
11	1.0	0.0	0.0	1.0	0.4303...	0.0	True
12	1.0	0.0	0.0	0.0	0.5696...	0.0	True
13	0.0	0.0	1.0	0.0	0.4050...	0.0	True
14	0.0	1.0	0.0	0.0	0.6075...	0.0	False
15	0.0	1.0	0.0	0.0	0.3544...	0.0	True
16	0.0	1.0	0.0	0.0	0.3924...	0.0	False
17	0.0	0.0	1.0	0.0	0.3417...	0.0	False
18	0.0	1.0	0.0	0.0	0.3037...	0.0	False
19	0.0	0.0	1.0	1.0	0.5696...	0.0	True
20	0.0	1.0	0.0	0.0	0.0	0.0	True
21	0.0	1.0	0.0	0.0	0.1772...	0.0	False
22	0.0	1.0	0.0	1.0	0.0126...	0.0	False
23	0.0	1.0	0.0	1.0	0.6202...	0.0	False
24	0.0	1.0	0.0	1.0	0.3670...	0.0	False
25	0.0	1.0	0.0	0.0	0.1265...	0.0	True
26	0.0	1.0	0.0	1.0	0.0126...	0.0	False
27	0.0	1.0	0.0	0.0	0.0886...	0.0	False
28	0.0	0.0	1.0	0.0	0.2658...	0.0	False
29	1.0	0.0	0.0	1.0	0.7848...	0.0	True
30	0.0	1.0	0.0	0.0	0.1898...	0.0	False

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.

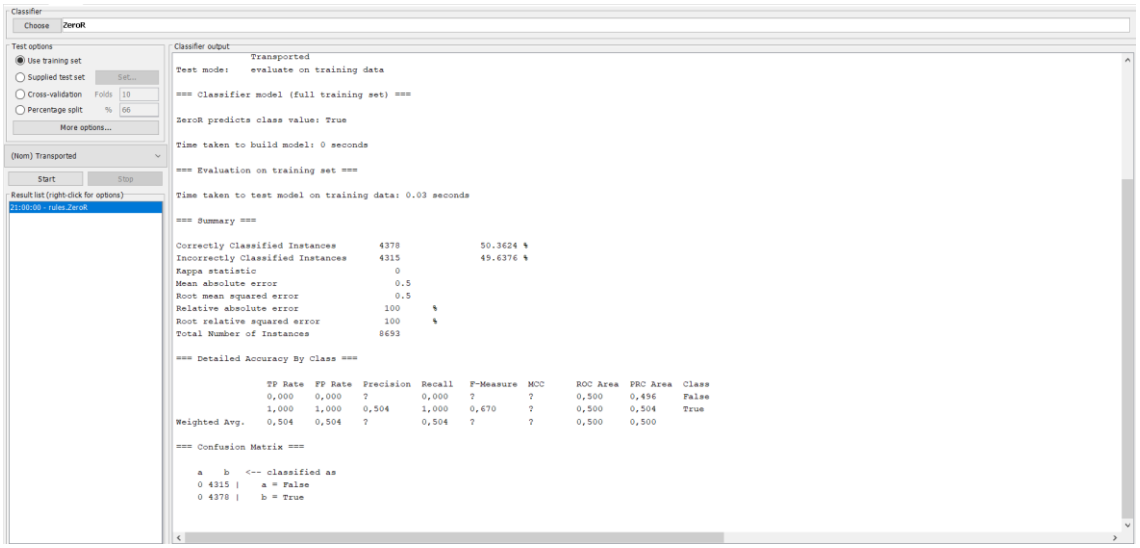
ZeroR

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 midiendo.)



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.

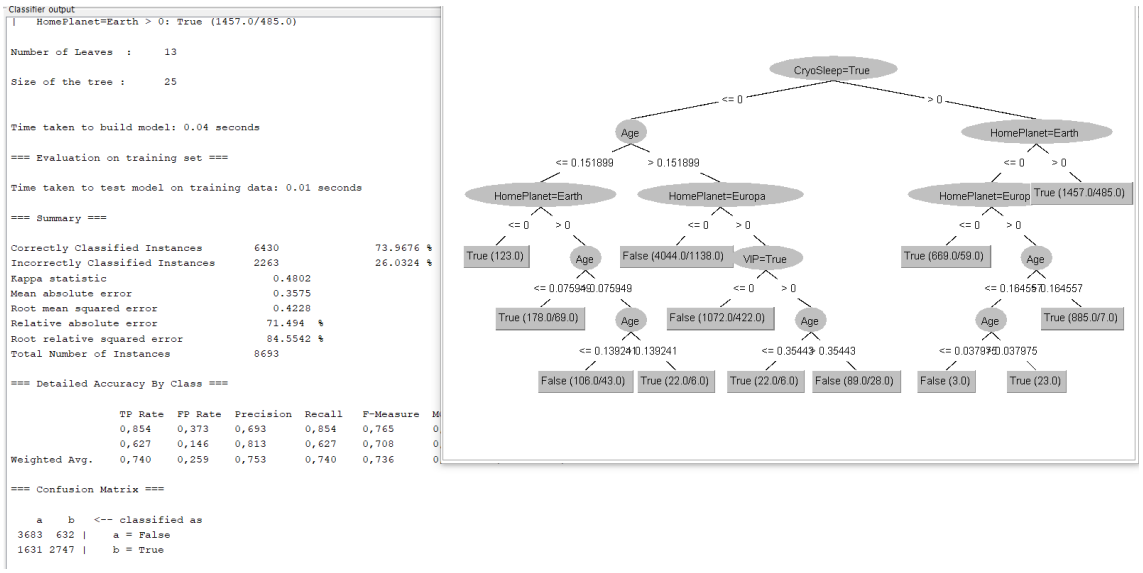
J48

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.

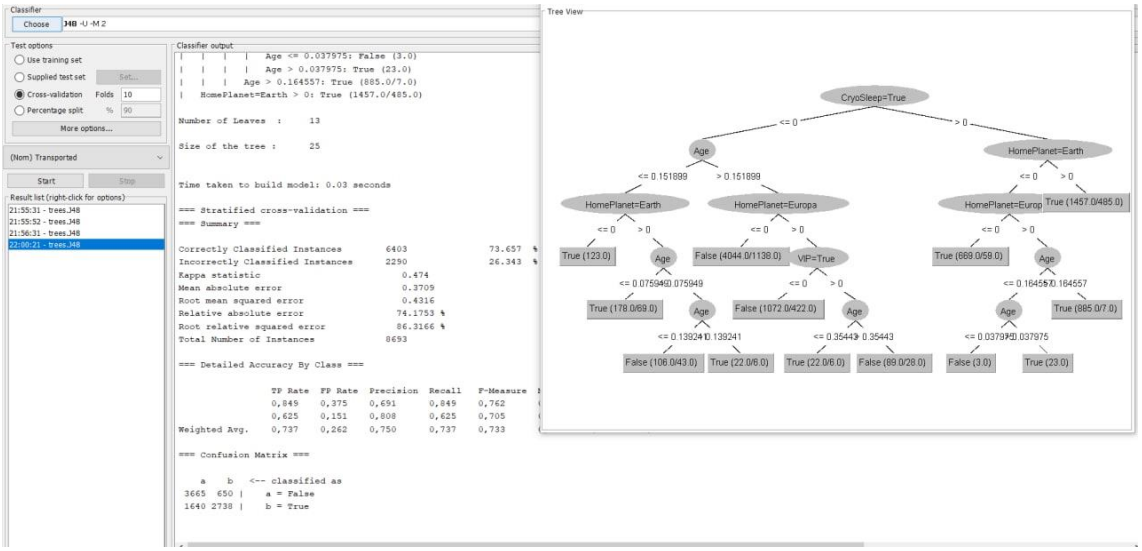
La poda es un proceso que se le aplica a los clasificadores de tipo árbol que consiste en iterativamente ir cortando nodos y observando su rendimiento de acierto con esto. El objetivo de este proceso es simplificar el árbol, y al igual que pasa con la acotación del número minimo de individuos en las 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



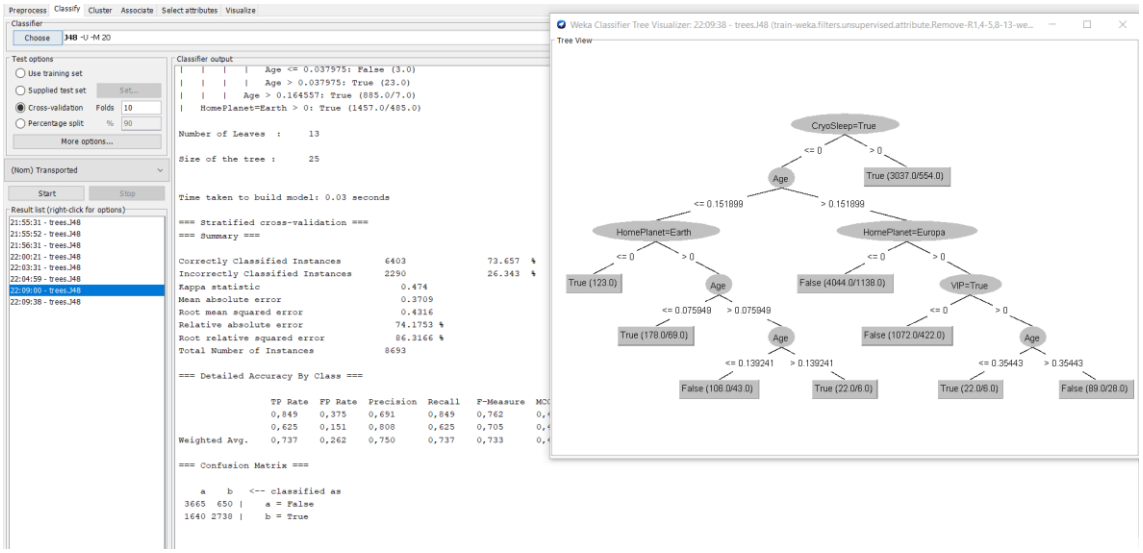
```
Classifier output
|   HomePlanet=Earth > 0: True (1457.0/485.0)

Number of Leaves  :     13

Size of the tree :      25


Time taken to build model: 0.04 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0.01 seconds

=== Summary ===

Correctly Classified Instances        6430               73.9676 %
Incorrectly Classified Instances      2263               26.0324 %
Kappa statistic                          0.4802
Mean absolute error                      0.3575
Root mean squared error                  0.4228
Relative absolute error                 71.494  %
Root relative squared error             84.5542 %
Total Number of Instances             8693

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  M
                 0,854    0,373    0,693      0,854    0,765      0
                 0,627    0,146    0,813      0,627    0,708      0
Weighted Avg.    0,740    0,259    0,753      0,740    0,736      0

=== Confusion Matrix ===

    a    b   <-- classified as
 3683  632 |    a = False
 1631 2747 |    b = True
```


J48/Unpruned=True/NminObj=2/CrossValidation

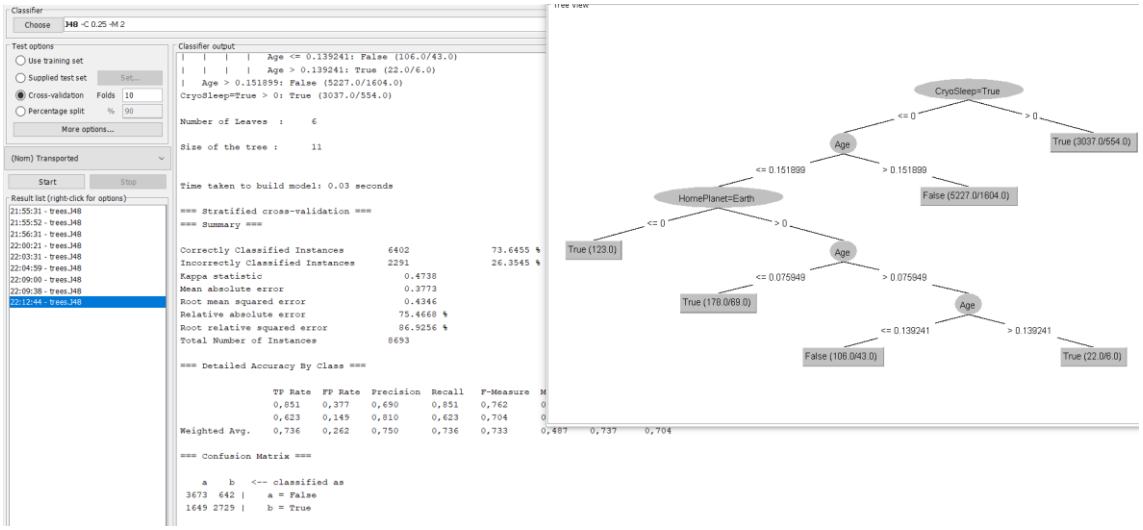


```
Classifier
Choose  J48 -U -M 2

Test options
Use training set
Supplied test set   Set...
Cross-validation   Folds  10
Percentage split   %  90
More options...

(Nom) Transported

Start   Stop
Result list (right-click for options)
21:55:31 - trees.J48
21:55:52 - trees.J48
21:56:31 - trees.J48
22:00:21 - trees.J48

Classifier output
|   |   |   |   Age <= 0.037975: False (3.0)
|   |   |   |   Age > 0.037975: True (23.0)
|   |   |   Age > 0.164557: True (885.0/7.0)
|   HomePlanet=Earth > 0: True (1457.0/485.0)

Number of Leaves  :     13

Size of the tree :      25

Time taken to build model: 0.03 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        6403               73.657  %
Incorrectly Classified Instances      2290               26.343  %
Kappa statistic                          0.474
Mean absolute error                      0.3709
Root mean squared error                  0.4316
Relative absolute error                 74.1753 %
Root relative squared error             86.3166 %
Total Number of Instances             8693

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure
                 0,849    0,375    0,691      0,849    0,762
                 0,625    0,151    0,808      0,625    0,705
Weighted Avg.    0,737    0,262    0,750      0,737    0,733

=== Confusion Matrix ===

    a    b   <-- classified as
 3665  650 |    a = False
 1640 2738 |    b = True
```

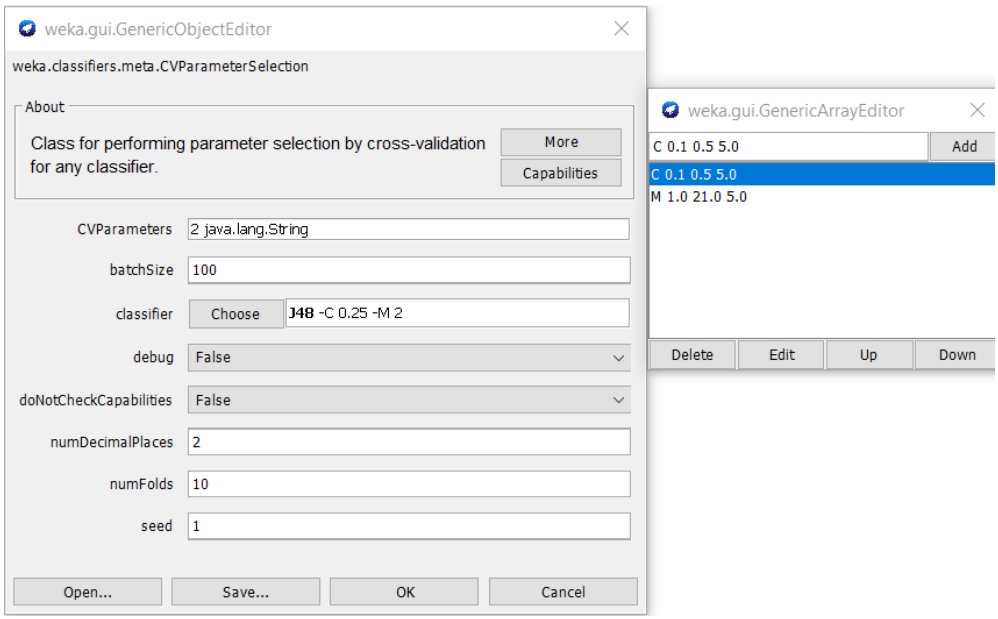
J48/Unpruned=True/NminObj=20/CrossValidation



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



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




```
=== Summary ===

Correctly Classified Instances        6427               73.933  %
Incorrectly Classified Instances      2266               26.067  %
Kappa statistic                          0.4795
Mean absolute error                      0.3732
Root mean squared error                  0.432
Relative absolute error                 74.6528 %
Root relative squared error             86.4019 %
Total Number of Instances             8693

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,853    0,373    0,693      0,853    0,765      0,493    0,760     0,686     False
                 0,627    0,147    0,812      0,627    0,708      0,493    0,760     0,733     True
Weighted Avg.    0,739    0,259    0,753      0,739    0,736      0,493    0,760     0,710

=== Confusion Matrix ===

    a    b   <-- classified as
 3680  635 |    a = False
 1631 2747 |    b = True
```

```
=== Run information ===

Scheme:       weka.classifiers.meta.CVParameterSelection -P "C 0.1 0.5 5.0" -P "M 1.0 21.0 5.0" -X 10 -S 1 -W weka.classifiers.trees.J48 -- -C 0.25 -M 2
Relation:     train-weka.filters.unsupervised.attribute.Remove-R1,4-5,8-13-weka.filters.unsupervised.attribute.ReplaceMissingValues-weka.filters.unsupervised.attribute.NominalTo
Instances:    8693
Attributes:   7
              HomePlanet=Europa
              HomePlanet=Earth
              HomePlanet=Mars
              CryoSleep=True
              Age
              VIP=True
              Transported
Test mode:    evaluate on training data

=== Classifier model (full training set) ===

Cross-validated Parameter selection.
Classifier: weka.classifiers.trees.J48
Cross-validation Parameter: '-C' ranged from 0.1 to 0.5 with 5.0 steps
Cross-validation Parameter: '-M' ranged from 1.0 to 21.0 with 5.0 steps
Classifier Options: -C 0.4 -M 16
```

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

```
=== Classifier model (full training set) ===

IB1 instance-based classifier
using 3 nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0.62 seconds

=== Summary ===

Correctly Classified Instances         644               74.1082 %
Incorrectly Classified Instances       225               25.8918 %
Kappa statistic                          0.4831
Mean absolute error                      0.3482
Root mean squared error                  0.4254
Relative absolute error                 69.646  %
Root relative squared error             85.0796 %
Total Number of Instances              869

=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
                 0,856     0,372     0,694       0,856   0,766      0,497    0,781     0,724     False
                 0,628     0,144     0,816       0,628   0,710      0,497    0,781     0,825     True
Weighted Avg.    0,741     0,257     0,755       0,741   0,738      0,497    0,781     0,775

=== Confusion Matrix ===

   a   b   <-- classified as
 369  62 |   a = False
 163 275 |   b = True
```

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

```
              VIP=True
              Transported
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

IB1 instance-based classifier
using 3 nearest neighbour(s) for classification


Time taken to build model: 0.01 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        6304               72.5181 %
Incorrectly Classified Instances      2389               27.4819 %
Kappa statistic                          0.4513
Mean absolute error                      0.3569
Root mean squared error                  0.4326
Relative absolute error                 71.3766 %
Root relative squared error             86.531  %
Total Number of Instances             8693

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,841    0,389    0,681      0,841    0,752      0,464    0,769     0,705     False
                 0,611    0,159    0,796      0,611    0,691      0,464    0,769     0,812     True
Weighted Avg.    0,725    0,273    0,739      0,725    0,722      0,464    0,769     0,759

=== Confusion Matrix ===

    a    b   <-- classified as
 3630  685 |    a = False
 1704 2674 |    b = True
```

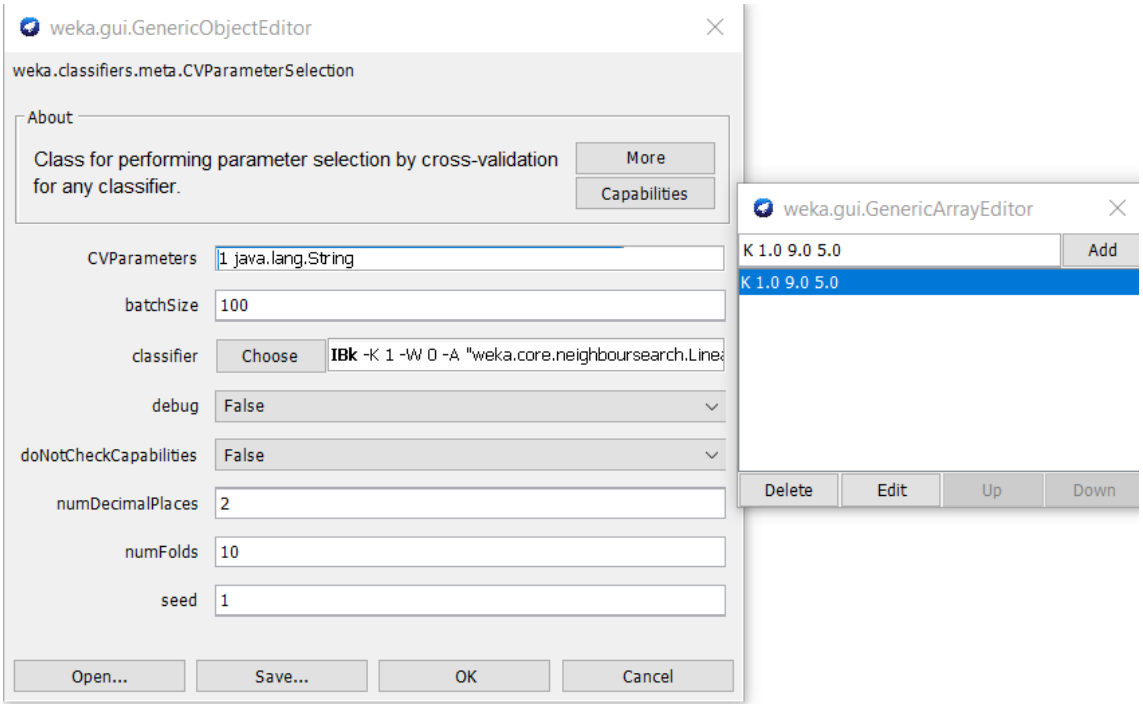
IBk(knn)/k=5/Distancia Euclidea/CrossValidation

```
              VIP=True
              Transported
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

IB1 instance-based classifier
using 5 nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        6308               72.5641 %
Incorrectly Classified Instances      2385               27.4359 %
Kappa statistic                          0.4522
Mean absolute error                      0.357
Root mean squared error                  0.4315
Relative absolute error                 71.4083 %
Root relative squared error             86.3092 %
Total Number of Instances             8693

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,841    0,388    0,681      0,841    0,753      0,465    0,771     0,708     False
                 0,612    0,159    0,796      0,612    0,692      0,465    0,771     0,814     True
Weighted Avg.    0,726    0,273    0,739      0,726    0,722      0,465    0,771     0,761

=== Confusion Matrix ===

    a    b   <-- classified as
 3629  686 |    a = False
 1699 2679 |    b = True
```

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

```
              VIP=True
              Transported
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

IB1 instance-based classifier
using 7 nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        6313               72.6216 %
Incorrectly Classified Instances      2380               27.3784 %
Kappa statistic                          0.4533
Mean absolute error                      0.357
Root mean squared error                  0.4309
Relative absolute error                 71.4053 %
Root relative squared error             86.188  %
Total Number of Instances             8693

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,841    0,387    0,682      0,841    0,753      0,466    0,772     0,705     False
                 0,613    0,159    0,796      0,613    0,693      0,466    0,772     0,815     True
Weighted Avg.    0,726    0,272    0,739      0,726    0,723      0,466    0,772     0,761

=== Confusion Matrix ===

    a    b   <-- classified as
 3628  687 |    a = False
 1693 2685 |    b = True
```

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



```
Classifier Options: -K 9 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""

IB1 instance-based classifier
using 9 nearest neighbour(s) for classification


Time taken to build model: 12.58 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 2.75 seconds

=== Summary ===

Correctly Classified Instances        6459               74.3012 %
Incorrectly Classified Instances      2234               25.6988 %
Kappa statistic                          0.4869
Mean absolute error                      0.3446
Root mean squared error                  0.4151
Relative absolute error                 68.9304 %
Root relative squared error             83.0134 %
Total Number of Instances             8693

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0,858    0,370    0,695      0,858    0,768      0,500    0,809     0,766     False
                 0,630    0,142    0,818      0,630    0,712      0,500    0,809     0,844     True
Weighted Avg.    0,743    0,255    0,757      0,743    0,740      0,500    0,809     0,805

=== Confusion Matrix ===

    a    b   <-- classified as
 3702  613 |    a = False
 1621 2757 |    b = True
```

(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

```
HomePlanet=Earth0.41    0.24
HomePlanet=Mars 0.13    0.11
CryoSleep=True  0.08    0.29
Age     0.25    0.18
VIP=True        0.02    0.01


Time taken to build model: 0.01 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances         633               72.8423 %
Incorrectly Classified Instances       236               27.1577 %
Kappa statistic                          0.4573
Mean absolute error                      0.4119
Root mean squared error                  0.4373
Relative absolute error                 82.3857 %
Root relative squared error             87.4717 %
Total Number of Instances              869

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0,780    0,322    0,704      0,780    0,740      0,460    0,784     0,729     False
               0,678    0,220    0,758      0,678    0,716      0,460    0,784     0,824     True
Weighted Avg.  0,728    0,271    0,731      0,728    0,728      0,460    0,784     0,777

=== Confusion Matrix ===

   a   b   <-- classified as
 336  95 |   a = False
 141 297 |   b = True
```

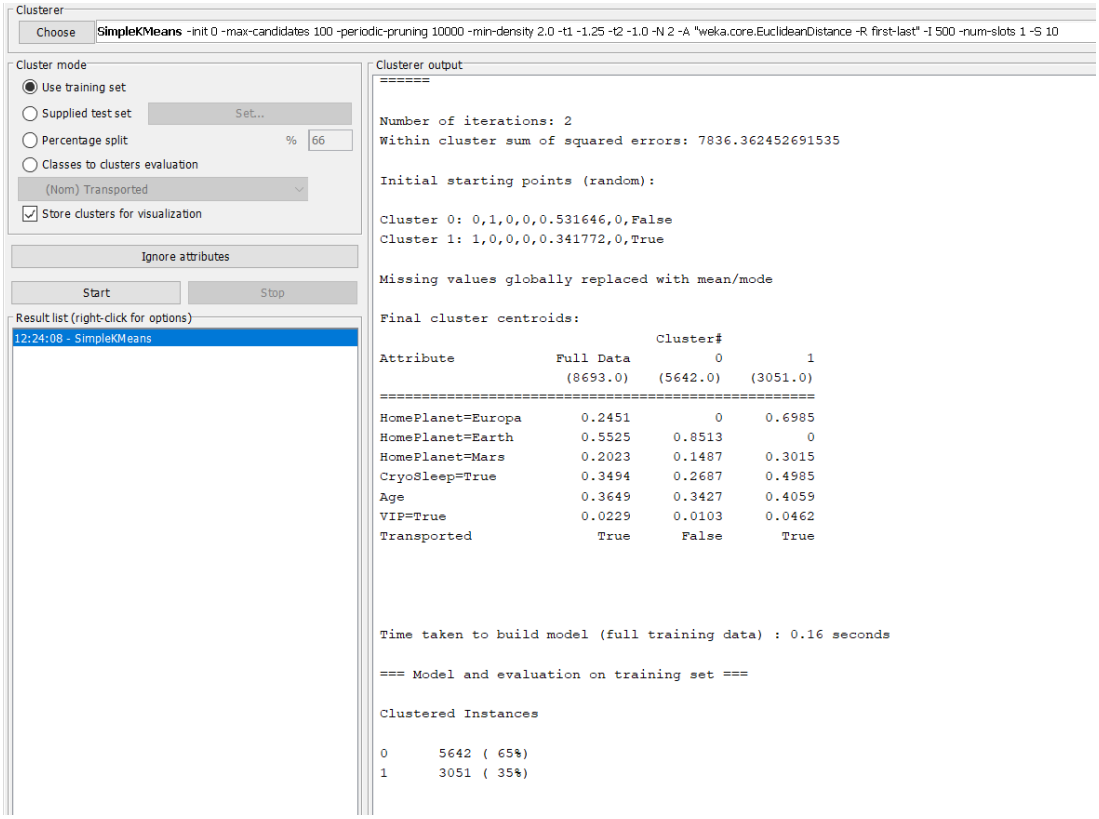
NaiveBayesMultinomial/CrossValidation

```
----------------------------------------
        False   True
HomePlanet=Europa       0.11    0.17
HomePlanet=Earth0.41    0.24
HomePlanet=Mars 0.13    0.11
CryoSleep=True  0.08    0.29
Age     0.25    0.18
VIP=True        0.02    0.01


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        6052               69.6192 %
Incorrectly Classified Instances      2641               30.3808 %
Kappa statistic                          0.3926
Mean absolute error                      0.4192
Root mean squared error                  0.446
Relative absolute error                 83.8361 %
Root relative squared error             89.2024 %
Total Number of Instances             8693

=== Detailed Accuracy By Class ===

               TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
               0,721    0,328    0,684      0,721    0,702      0,393    0,759     0,690     False
               0,672    0,279    0,710      0,672    0,690      0,393    0,759     0,804     True
Weighted Avg.  0,696    0,303    0,697      0,696    0,696      0,393    0,759     0,747

=== Confusion Matrix ===

    a    b   <-- classified as
 3111 1204 |    a = False
 1437 2941 |    b = True
```

Clustering

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

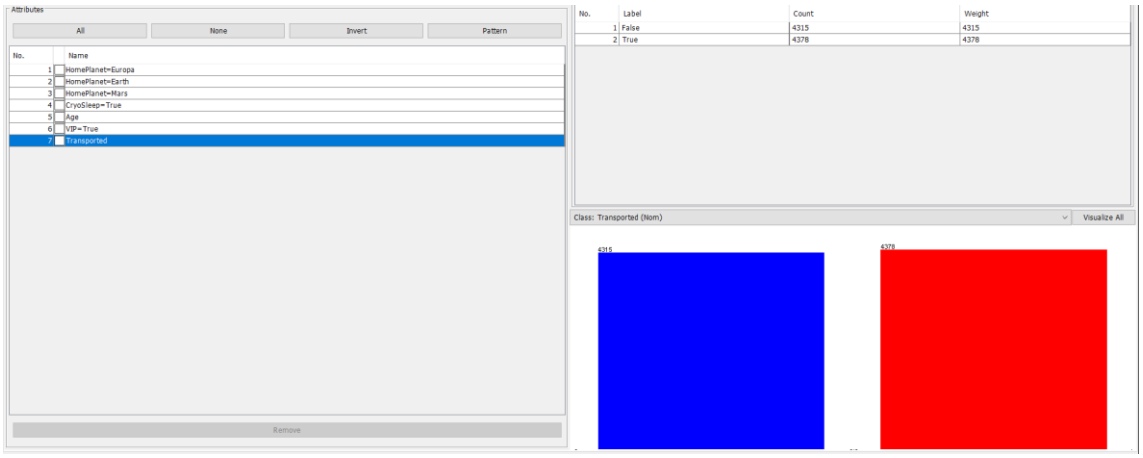
Pestaña Clusters->SimpleKMeans->use training set->Start (valores por defecto -> grupos=2)



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.



Comparación de clasificadores

	Resultado	Configuración
ZeroR	50,3624%	ZeroR. Por defecto
J48	73,9676%	J48/Unpruned=True/NminObj=2/ Conjunto de entrenamiento J48
J48	73,657%	J48/Unpruned=True/NminObj=2/CrossValidation
J48	73,657%	J48/Unpruned=True/NminObj=20/CrossValidation
J48	73,6455%	J48/Unpruned=False/NminObj=2/CrossValidation (Podado)
J48	73,933%	Hiperparametros J48/confidenceFactor de 0.1 a 0.5 con 5 pasos/NumMinObj de 1 a 21 con 5 pasos. M=16, C=0,4
KNN	74.1082%	IBk(knn)/k=3/Distancia Euclidea/Percentage Split
KNN	72,5181%	IBk(knn)/k=3/Distancia Euclidea/CrossValidation
KNN	72,5641%	IBk(knn)/k=5/Distancia Euclidea/CrossValidation
KNN	72,6216%	IBk(knn)/k=7/Distancia Euclidea/CrossValidation
KNN	74,3012%	Hiperparametros IBk/k(vecinos mas cercanos) de 1 a 9 con 5 pasos. K = 9
Naive Bayes	72,8423%	NaiveBayesMultinomial/Percentage Split
Naive Bayes	69,6192%	NaiveBayesMultinomial/CrossValidation
SimpleKmeans	65%/35%	Pestaña Clusters->SimpleKMeans->use training set->Start (valores por defecto -> grupos=2)