Inteligencia de Negocio

P1 -Análisis Predictivo Empresarial Mediante Clasificación

Grado Ing. Informática

Francisco Carrillo Pérez
Grupo 1 martes a las 09:30

Contents

List of Figures

List of Tables

1 Introducción

El problema que se aborda es conocido como *Adult Dataset* [1]. El objetivo de este *dataset* es predecir si los ingresos de una persona exceden los 50,000 dólares al año basándose en los datos del censo.

Este *dataset* se compone de las siguientes variables:

1. **age:** variable continua con la edad.
2. **workclass:** clase del trabajo. Este atributo puede tomar uno de hasta 8 valores distintos.
3. **fnlwgt:** variable continua. Pesos finales.
4. **education:** máxima educación de la persona. Este atributo puede tomar uno de hasta 16 valores distintos.
5. **education-num:** número asociada al atributo de educación. Variable continua.
6. **marital-status:** estado marital de la persona. Este atributo puede tomar uno de hasta 7 valores distintos.
7. **occupation:** ocupación de la persona. Este atributo puede tomar uno de hasta 14 valores distintos.
8. **relationship:** relación en la que se encuentra. Este atributo puede tomar uno de hasta 6 valores distintos.
9. **race:** raza de la persona. Este atributo puede tomar uno de hasta 5 valores distintos.
10. **sex:** sexo de la persona. Puede ser Male o Female.
11. **capital-gain:** cuanto capital ha ganado la persona. Es una variable continua.
12. **capital-loss:** cuanto capital ha perdido la persona. Es una variable continua.
13. **hour-per-week:** horas que trabaja a la semana. Es una variable continua.
14. **native-country:** país de nacimiento. Este atributo puede tomar uno de hasta 41 valores distintos.

2 Resultados Obtenidos

En este apartado se expondrán los resultados obtenidos con los distintos algoritmos utilizados.

Se expondrán los valores de las métricas a continuación ya que por espacio se ha decidido acortar su nombre en las tablas. Estos acortamientos serán usados a lo largo de todo el documento:

- TP: True Positive
- **FP**: False Positive
- **TN**: True Negative
- **FN**: False Negative
- **Rec.**: Recall
- **Pre.**: Precision
- **Acc.**: Accuracy
- **AUC**: Area Under the Curve

2.1 k-NN

Table 1: Primeros resultados con el algoritmo K-NN

Tipo	**TP**	**FP**	**TN**	**FN**	**Rec.**	**Pre.**	**F-measure**	**Acc.**	**AUC**	**G-mean**
Primer experimento	6571	4169	32986	5516	0.562	0.612	0.586	80.99%	0.8121	0.692

Figure 1: ROC Curve para el algoritmo K-NN en el primer experimento

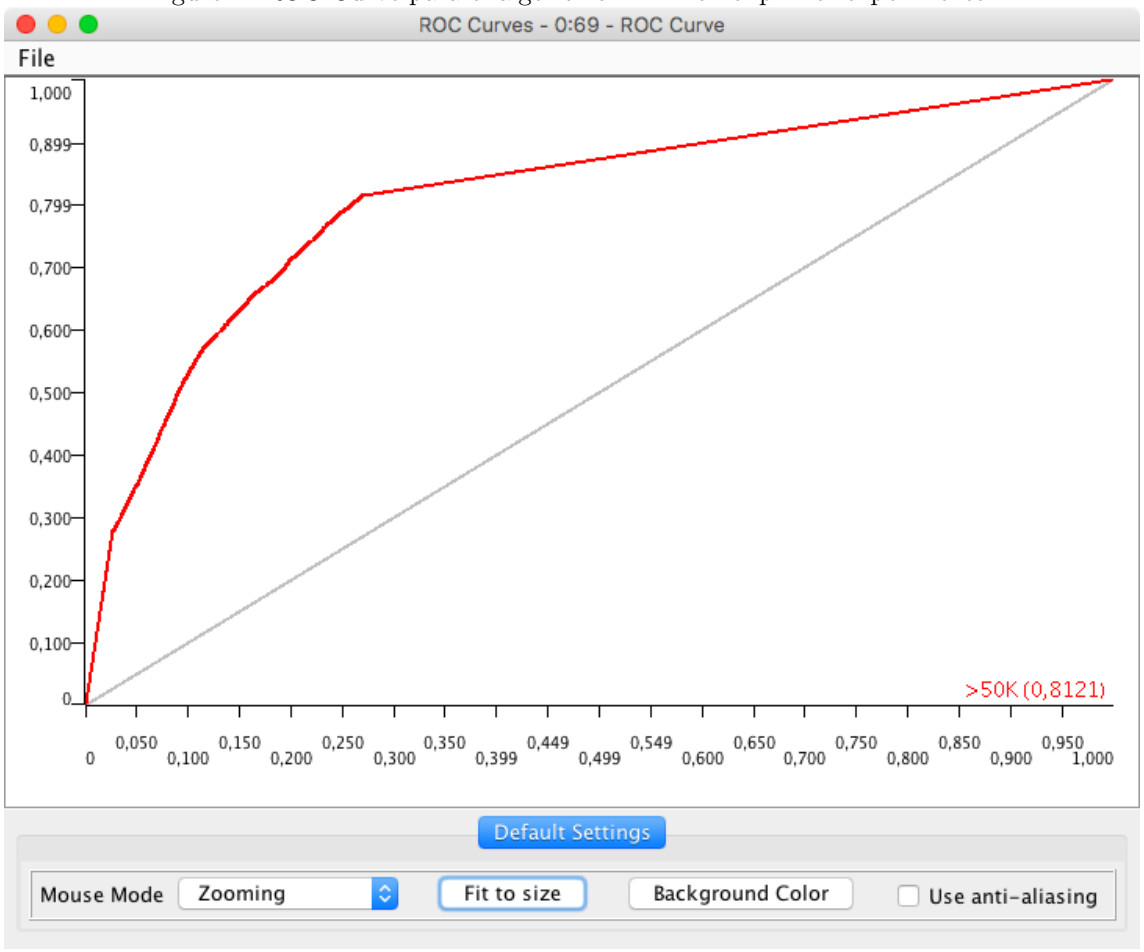
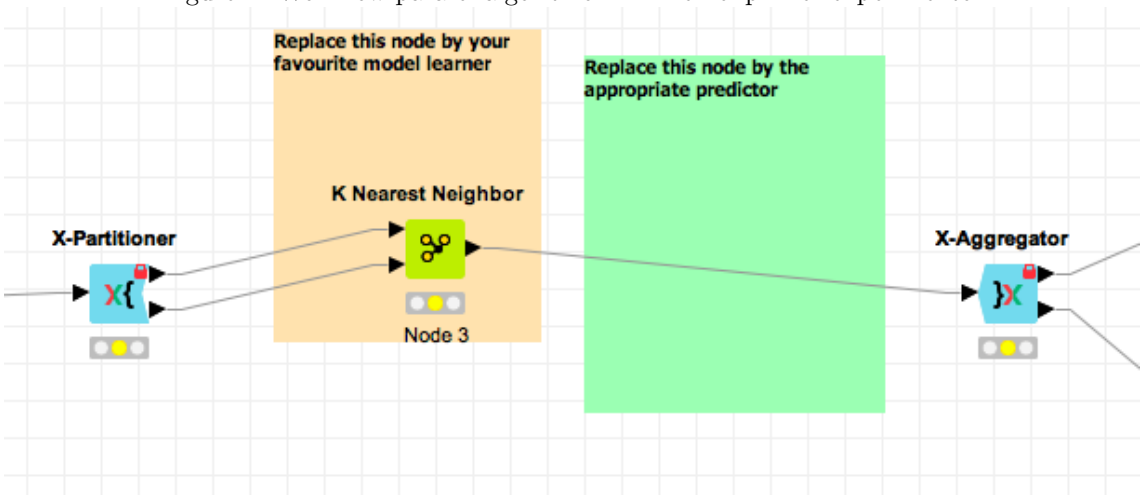


Figure 2: Workflow para el algoritmo K-NN en el primer experimento



2.2 Redes Neuronales: Multi Layer Perceptron

Table 2: Primeros resultados con el algoritmo MLP

Tipo	TP	FP	TN	FN	Rec.	Pre.	F-measure	Acc.	AUC	G-mean
Primer experimento	7115	2605	34550	4572	0.609	0.732	0.665	85.306%	0.9019	0.75

Figure 3: ROC Curve de Redes Neuronales en el primer experimento

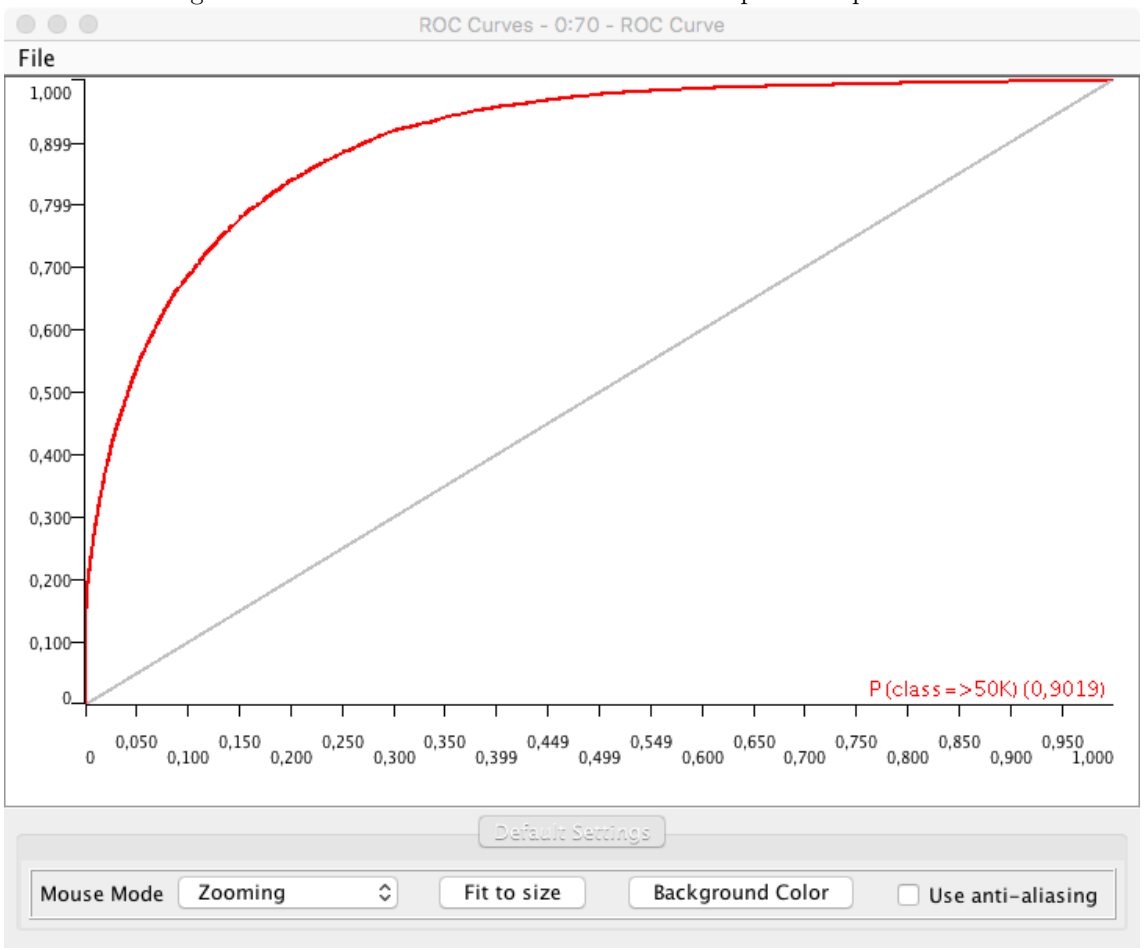
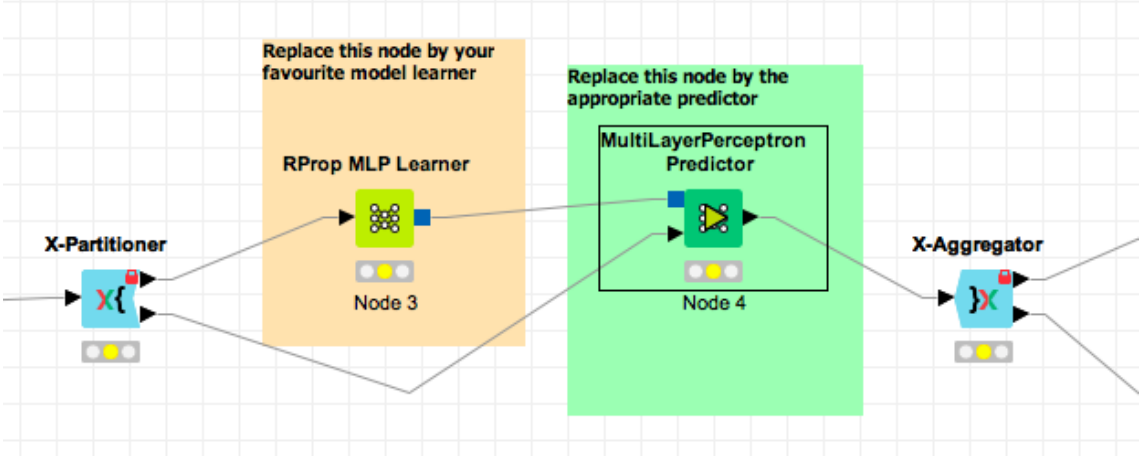


Figure 4: Workflow para el algoritmo MLP en el primer experimento



2.3 Árboles de decisión: Algoritmo C4.5

Table 3: Primeros resultados con el algoritmo C4.5

Tipo	TP	FP	TN	FN	Rec.	Pre.	F-measure	Acc.	AUC	G-mean
Primer experimento	7261	4030	33125	4426	0.621	0.643	0.632	82.687%	0.8019	0.744

Figure 5: ROC Curve para el algoritmo C4.5 en el primer experimento

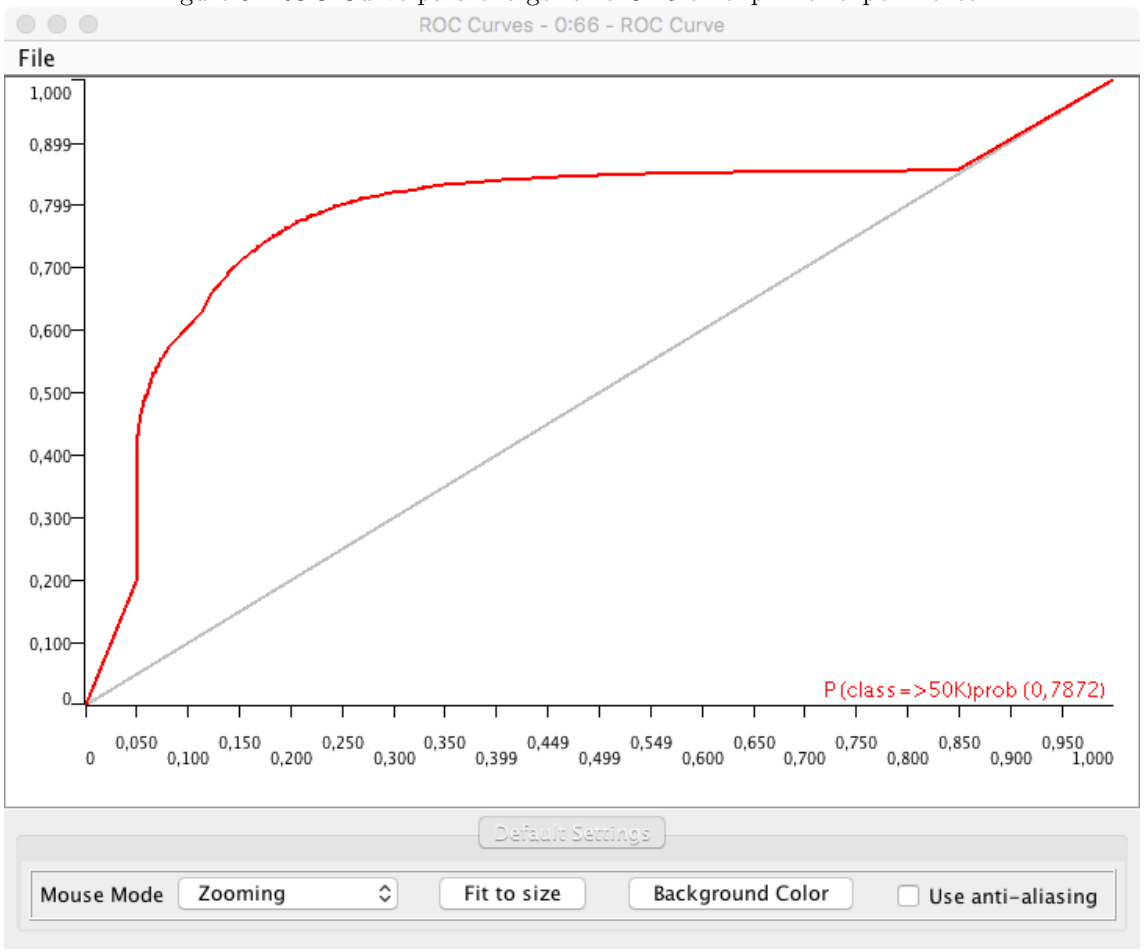
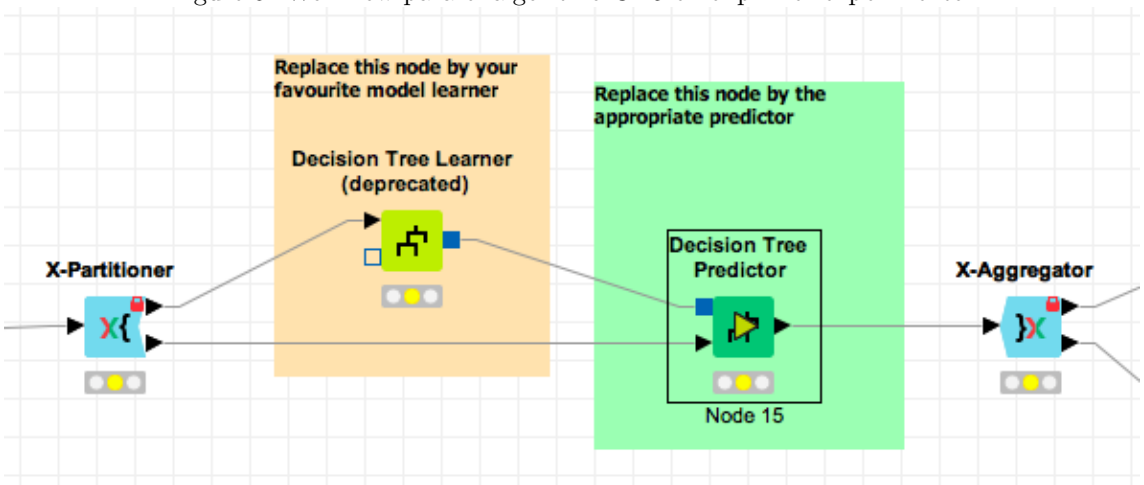


Figure 6: Workflow para el algoritmo C4.5 en el primer experimento



2.4 AdaBoost

Table 4: Primeros resultados con el algoritmo AdaBoost

Tipo	TP	FP	TN	FN	Rec.	Pre.	F-measure	Acc.	AUC	G-mean
Primer experimento	6050	1694	35461	5637	0.518	0.781	0.623	84.99%	0.8962	0.70

Figure 7: ROC Curve para el algoritmo AdaBoost en el primer experimento

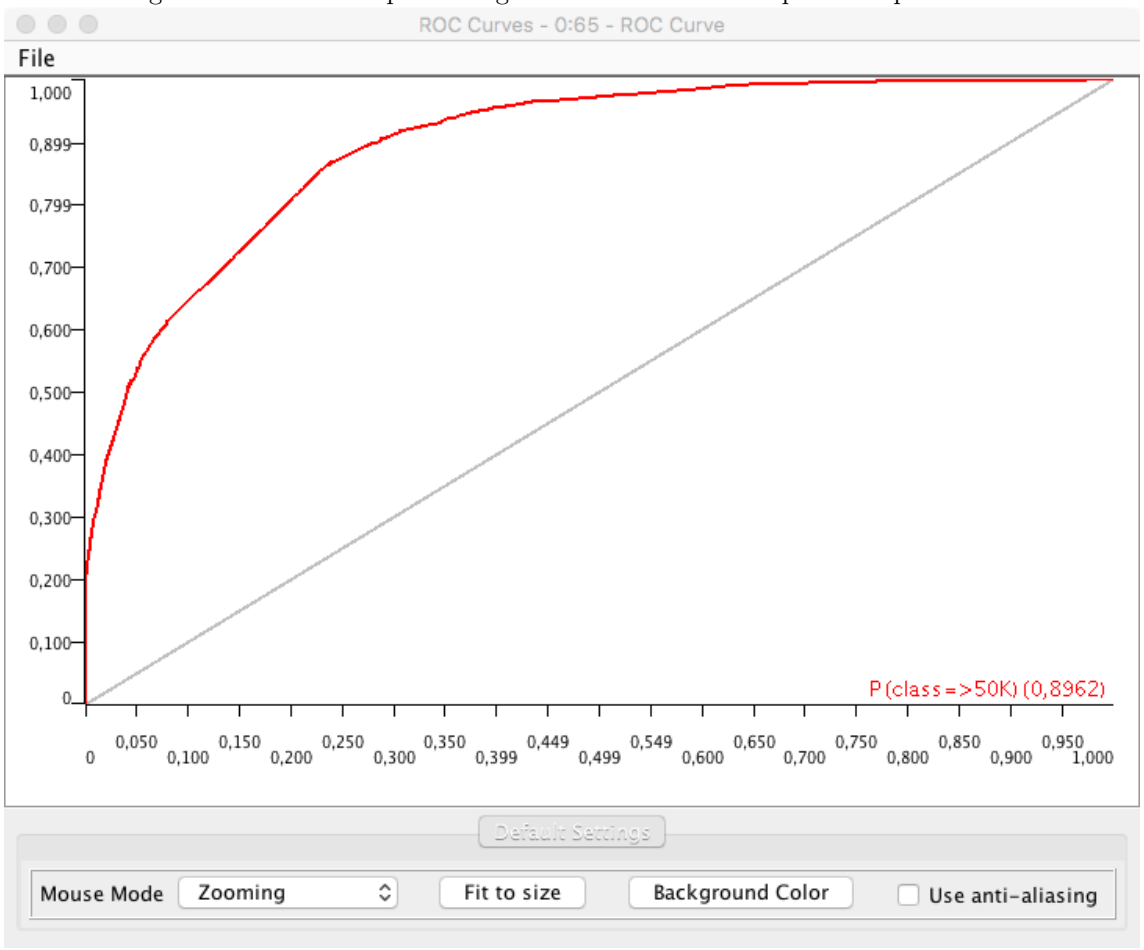
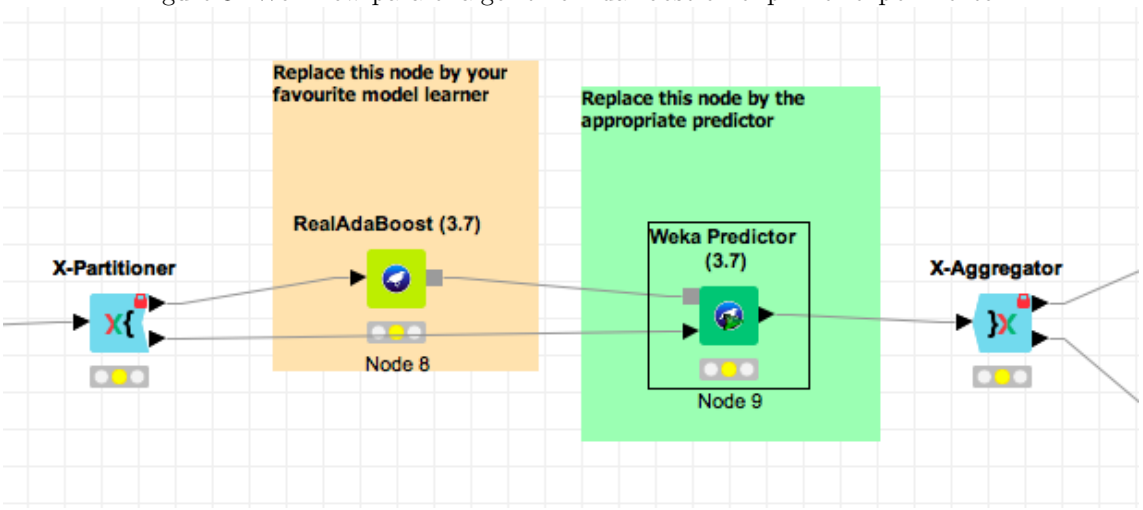


Figure 8: Workflow para el algoritmo AdaBoost en el primer experimento



2.5 Gradient Boosting

Table 5: Primeros resultados con el algoritmo Gradient Boosting

Tipo	TP	FP	TN	FN	Rec.	Pre.	F-measure	Acc.	AUC	G-mean
Primer experimento	7351	1948	35207	4336	0.629	0.791	0.701	87.134%	0.9256	0.77

Figure 9: ROC Curve para el algoritmo Gradient Boosting en el primer experimento

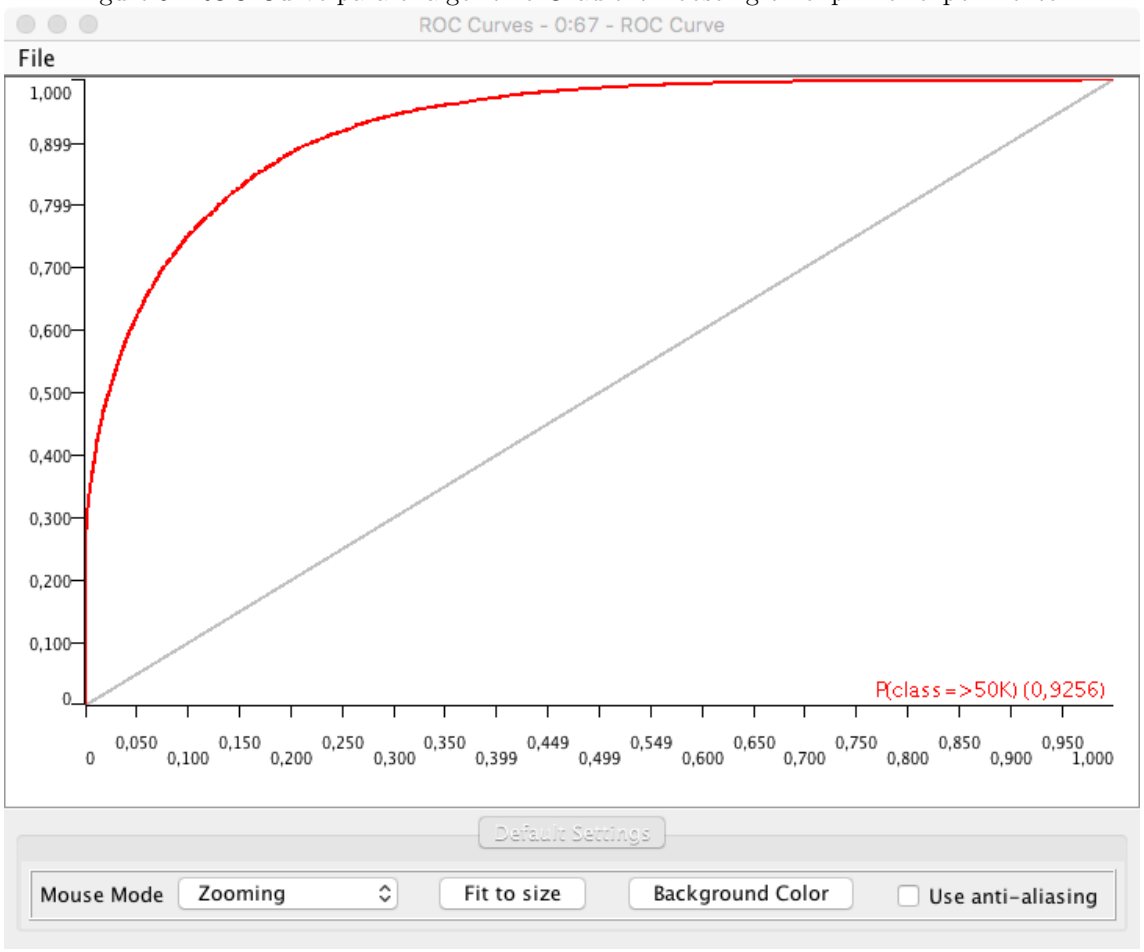
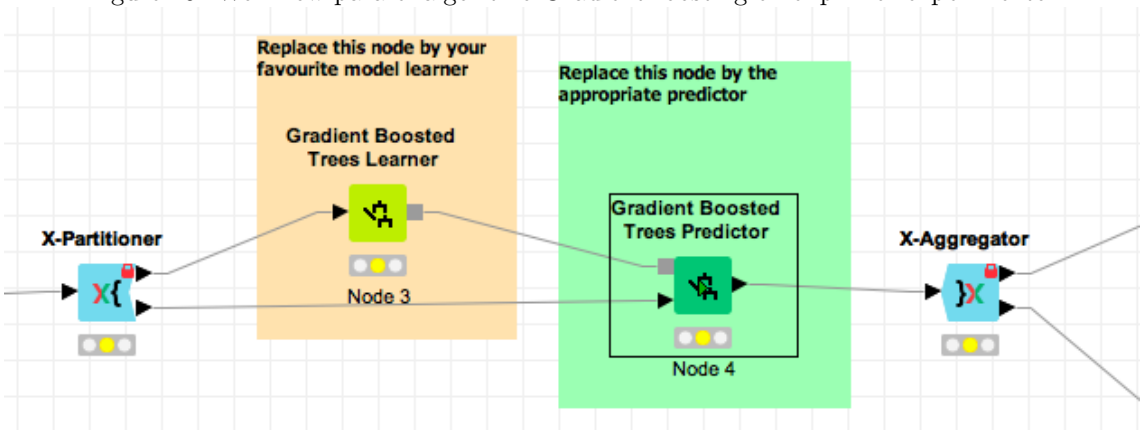


Figure 10: Workflow para el algoritmo GradientBoosting en el primer experimento



2.6 Random Forest

Tipo	TP	FP	TN	FN	Rec.	Pre.	F-measure	Acc.	AUC	G-mean
Primer experimento	7207	1923	35232	4480	0.617	0.789	0.692	86.69% s	0.8939	0.764

Figure 11: ROC Curve para el algoritmo Random Forest en el primer experimento

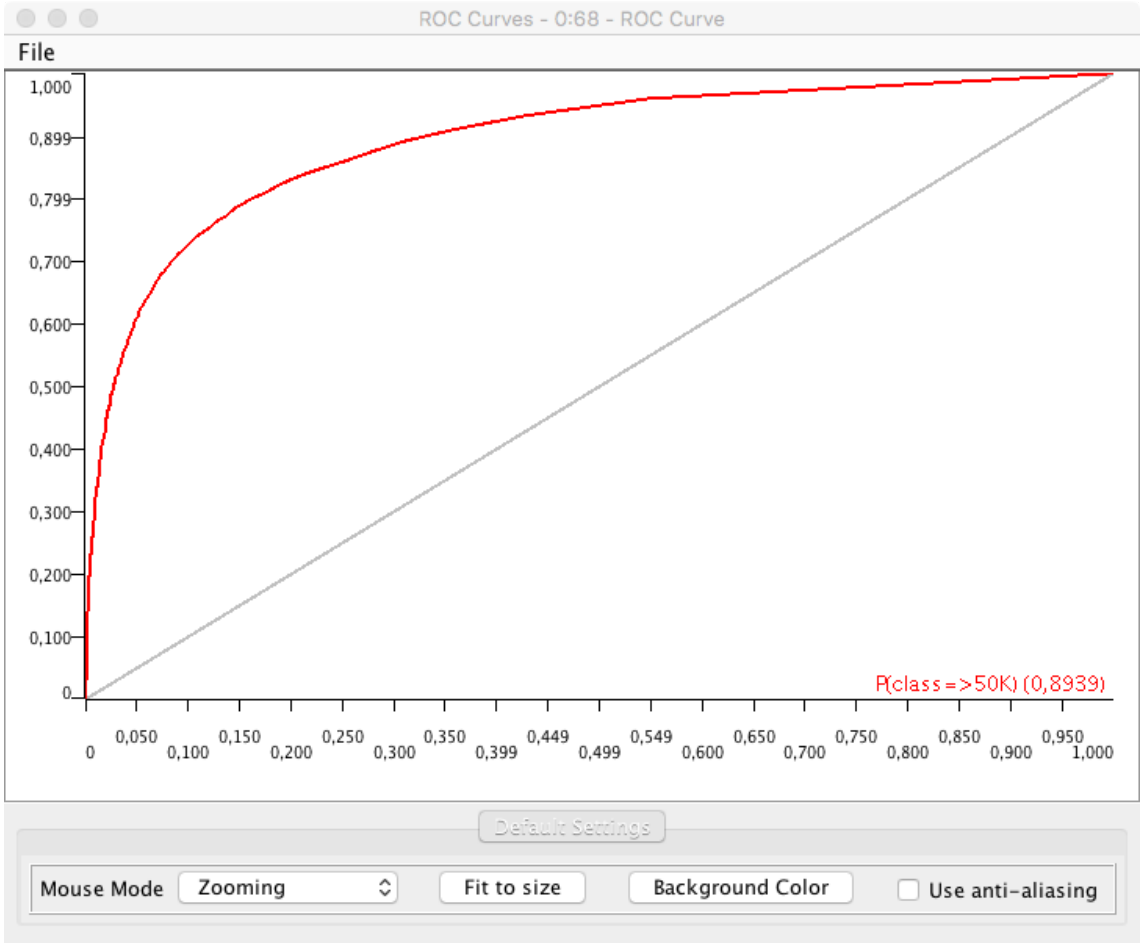
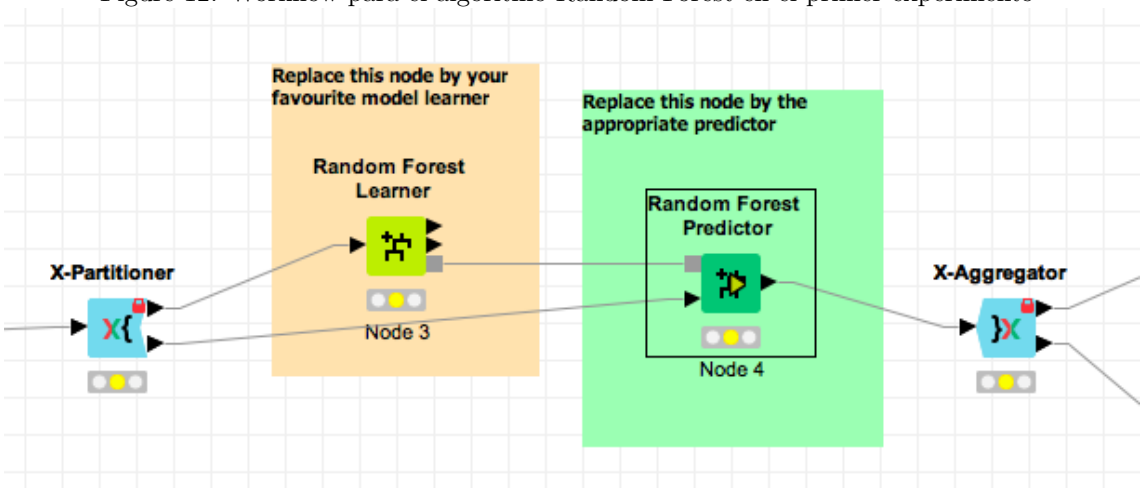


Figure 12: Workflow para el algoritmo Random Forest en el primer experimento



3 Análisis de resultados

En esta sección se va a realizar un análisis de los resultados que se han obtenido en un primer experimento con los distintos algoritmos empleados.

En primer lugar, en la tabla 6, se van a mostrar los resultados obtenidos por cada algoritmo. Una vez realizado esto, se expondrán las conclusiones a las que se puedan llegar con respecto a estos resultados.

Table 6: Compendio de los resultados en el primer experimento por cada algoritmo

Tipo	**TP**	**FP**	**TN**	**FN**	**Rec.**	**Pre.**	**F-measure**	**Acc.**	**AUC**	**G-mean**
KNN	6571	4169	32986	5516	0.562	0.612	0.586	80.99%	0.8121	0.692
Redes Neuronales	7115	2605	34550	4572	0.609	0.732	0.665	85.306%	0.9019	0.75
C45	7261	4030	33125	4426	0.621	0.643	0.632	82.687%	0.8019	0.744
AdaBoost	6050	1694	35461	5637	0.518	0.781	0.623	84.99%	0.8962	0.70
Gradient Boosting	7351	1948	35207	4336	0.629	0.791	0.701	87.134%	0.9256	0.77
Random Forest	7207	1923	35232	4480	0.617	0.789	0.692	86.69% s	0.8939	0.764

Como se puede observar, los algoritmos que mejores resultados obtienen son el algoritmo de **Gradient Boosting**[2] y el algoritmo de **Random Forest**[3]. Esto era esperable, ya que son de los algoritmos más potentes que se han utilizado junto con las redes neuronales.

El problema que tienen este tipo de algoritmos es que, debido a su alta complejidad, son cajas negras a la hora de saber porque están realizando cierta clasificación cosa que no pasa con algoritmos como el C4.5, o cualquier algoritmo basado en reglas.

Luego tenemos el algoritmo más simple, el K-NN, pero que nos sirve como referencia para comparar nuestros algoritmos contra un algoritmo real, no simplemente el algoritmo de votar la clase mayoritaria.

En la Figura 13 se pueden observar los distintos valores obtenidos en cuanto a Accuracy por los distintos algoritmos , en la Figura 14 se pueden observar los distintos valores obtenidos en el F1-Score y en la Figura 15 se pueden observar los distintos valores obtenidos en el G-mean.

En primer lugar quiero decir que los resultados que se obtienen en la Tabla 6 están sesgados y no los usaría como resultados finales a la hora de presentárselos a un cliente, ya que al existir un desbalanceo de las clases, como se explica en la sección 5, los algoritmos están aprendiendo a clasificar la clase negativa ya que es la que más instancias tiene. Es por esto que una mejor medida en este caso para comparar los resultados es utilizar el F1-Score, ya que hace uso del recall y del precision, ya que al existir tantos TN, esto hace que el accuracy pueda verse poco afectado y lo mismo ocurre con la métrica del G-mean. Recordemos que las ecuaciones de ambas expresiones son las siguientes:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$F1 - Score = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{2}$$

$$G - mean = \sqrt{TPR \times TNR} \tag{3}$$

Figure 13: Comparación del accuracy de los distintos algoritmos

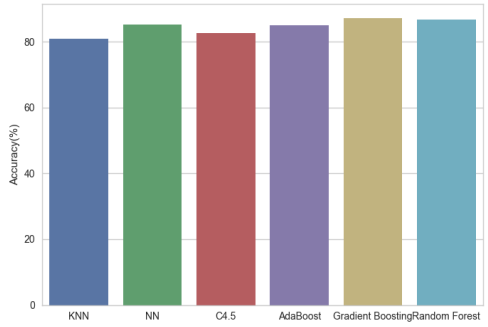


Figure 14: Comparación del F1-Score de los distintos algoritmos

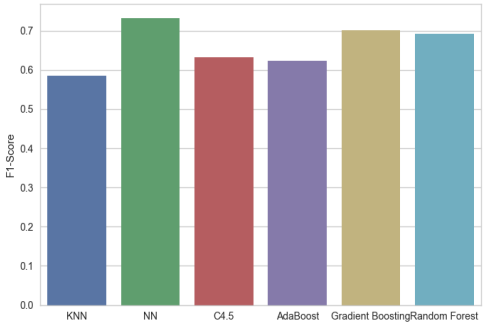
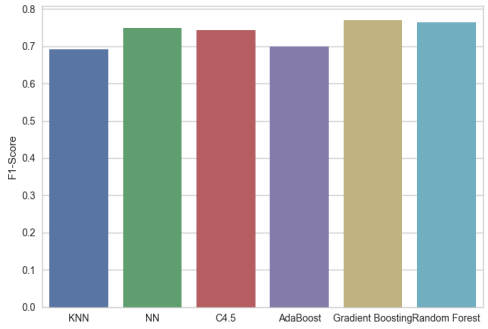


Figure 15: Comparación del G-mean de los distintos algoritmos



En cuanto a los resultados obtenidos en estas figuras se puede ver cómo los algoritmos que mejor desempeño tanto en el Accuracy como en el F1-Score son el **Gradient Boosting**, **Redes Neuronales** y **Random Forest**. Estos tres se encuentran en el podio en ambas métricas.

Encontramos que las redes neuronales tienen un mejor balance entre recall y precision, ya que son las que mejor puntuación obtienen dentro del F1-Score, seguidas muy de cerca por el algoritmo de Gradient Boosting. En cambio, si nos fijamos en la métrica de Accuracy, el algoritmo Gradient Boosting es el que mejor resultado obtiene relegando a las redes neuronales al tercer algoritmo que mejores resultados obtiene.

Esto lo que nos indica es que las redes neuronales tienen una ligera ventaja frente al algoritmo de Gradient Boosting a la hora de trabajar con conjuntos de datos que se encuentran desbalanceados, al menos con los parámetros por defecto que presentan ambos algoritmos en KNIME.

Sin embargo, Gradient Boosting es un algoritmo más potente a la hora de clasificar correctamente más número de instancias, es decir, aprende mejor las características de uno o ambos conjuntos a la hora de obtener un mejor resultado en la clasificación.

Si observamos la Figura 15 la cuál representa el valor del G-mean para cada algoritmo, podemos observar que el algoritmo que mejor relación tiene entre el TPR y el TNR es el Gradient Boosting, continuando la línea explicada en el anterior párrafo.

4 Configuración de algoritmos

En este apartado se realizará un estudio sobre como afecta la modificación de ciertos parámetros de los algoritmos a la hora de obtener un mejor desempeño en la clasificación.

4.1 k-NN

Table 7: Primeros resultados con el algoritmo K-NN

Tipo	TP	FP	TN	FN	Rec.	Pre.	F-measure	Acc.	AUC	G-mean
KNN Sin variables14 con K=3	9335	2851	8700	2353	0.799	0.766	0.782	77.705%	0.8133	0.769
KNN Sin variables14 con K=5	9406	2766	8884	2218	0.805	0.773	0.788	78.373%	0.8229	0.785
KNN Sin variables14 con K=7	9453	2753	8897	2234	0.809	0.774	0.791	78.631%	0.8266	0.7855
KNN Sin variables14 con K=9	9475	2770	8880	2212	0.811	0.774	0.792	78.652%	0.8287	0.7857

Cómo se puede observar, conforme aumentamos el valor de K clasificamos mejor los valores. A mayor valor de K, más se reduce el efecto de ruido en la clasificación, es decir, generalizamos más. Esto es debido a que si existe ruido, y es cercano al punto que estamos intentando clasificar, si utilizamos un valor de K bajo es bastante probable que se clasifique erróneamente esa muestra.

En cambio, si utilizamos un valor de K más grande, estamos aumentando el número de muestras que estamos usando para la clasificación, por lo que el ruido puede hacerse más indistinguible en el caso de tener poco, lo que lleva a que clasifiquemos esa muestra correctamente.

Es por ello que cuando utilizamos K=9 se obtienen mejores resultados que cuando utilizamos K=3.

Figure 16: ROC Curve para el algoritmo K-NN sin variables y con K=3

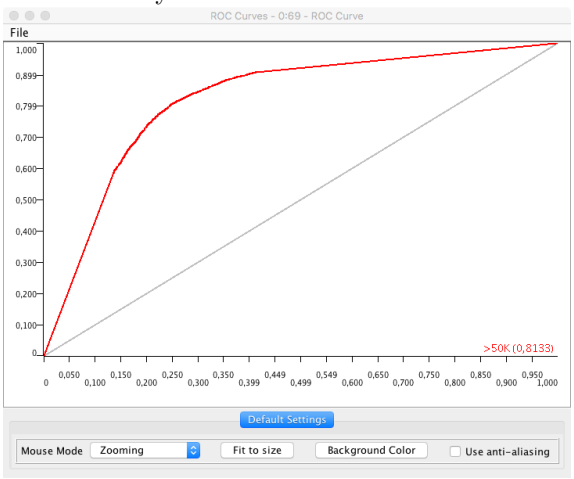


Figure 17: ROC Curve para el algoritmo K-NN sin variables y con K=5

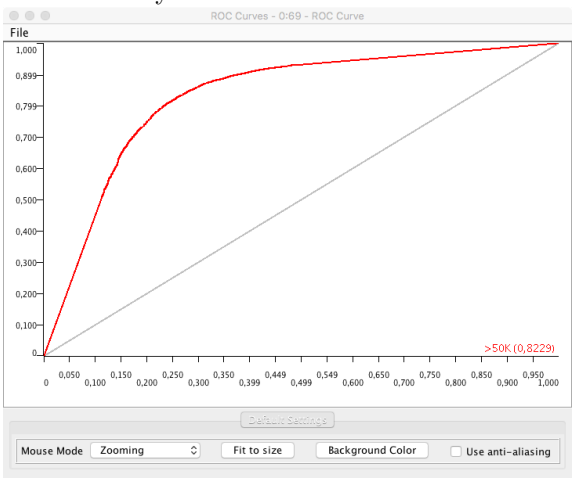


Figure 18: ROC Curve para el algoritmo K-NN sin variables y con K=5

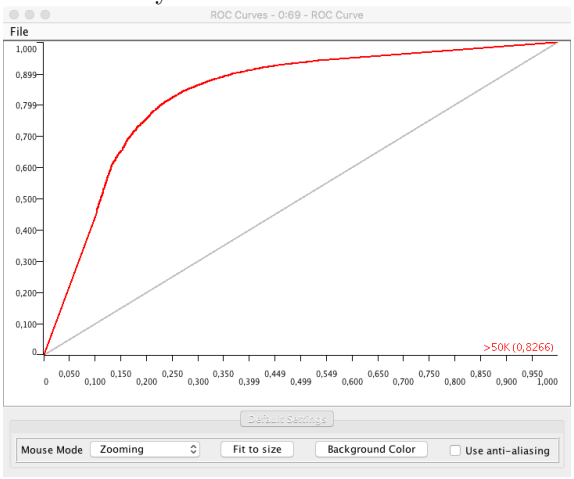
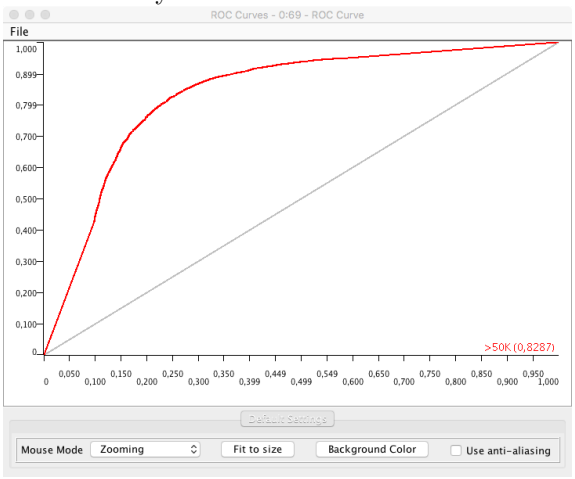


Figure 19: ROC Curve para el algoritmo K-NN sin variables y con K=9



4.2 Redes Neuronales: Multi Layer Perceptron

Table 8: Resultados de las modificaciones con el algoritmo MLP

Tipo	TP	FP	TN	FN	Rec.	Pre.	F-measure	Acc.	AUC	G-mean
MLP sin variables14	10136	2546	9104	1551	0.867	0.799	0.832	82,444%	0.907	0.823
MLP sin variables14, con 4 capas ocultas y 10 neu. por capa	10132	2560	9090	1555	0.867	0.798	0.831	82,367%	0.9063	0.822
MLP sin variables14, con 4 capas ocultas y 20 neu. por capa	10163	2595	9055	1524	0.87	0.797	0.831	82,35%	0.9069	0821
MLP sin variables14, con 5 capas ocultas y 10 neu. por capa	10083	2578	9072	1604	0.863	0.796	0.828	82,08%	0.9028	0.819
MLP sin variables14, con 5 capas ocultas y 20 neu. por capa	10087	2599	9051	1600	0.863	0.795	0.828	82,007%	0.9046	0.818

En este caso, podemos observar como conforme aumentamos la complejidad de la red, se reduce la calidad en el conjunto de test. Esto se pude deber a un sobreajuste en el conjunto de entrenamiento, ya que conforme se ha ido aumentando la complejidad del modelo aumentando capas y número de neuronas por capa, se han obtenido peores resultados.

Esto es debido a que las redes neuronales son un tipo de modelo que aprende muy bien las características del conjunto, y que dependiendo de las características generaliza más o mejor. Por eso se puede observar como en este caso, con una red menos compleja, se generaliza mejor que con una red más compleja en sus parámetros.

Figure 20: ROC Curve para MLP sin variables, con 4 capas ocultas y 10 neuronas por capa

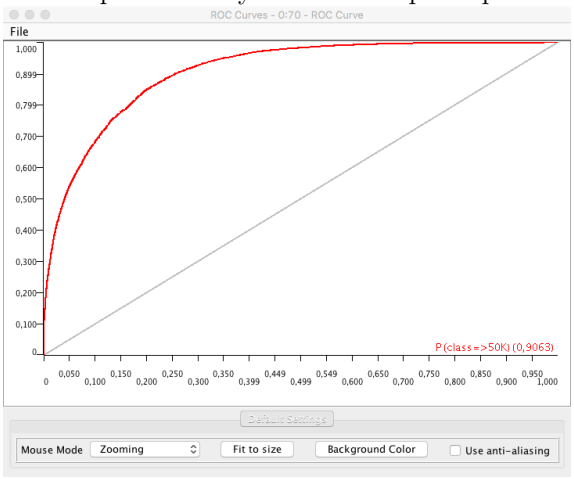


Figure 21: ROC Curve para MLP sin variables, con 4 capas ocultas y 20 neuronas por capa

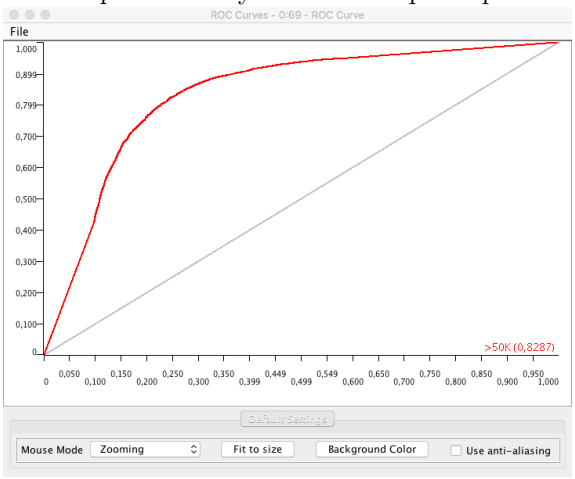


Figure 22: ROC Curve para MLP sin variables, con 5 capas ocultas y 10 neuronas por capa

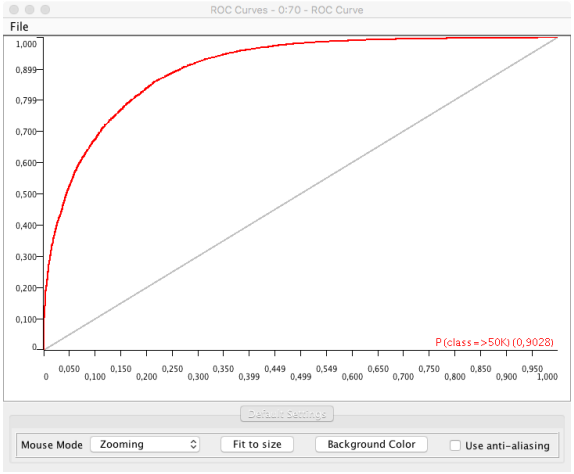
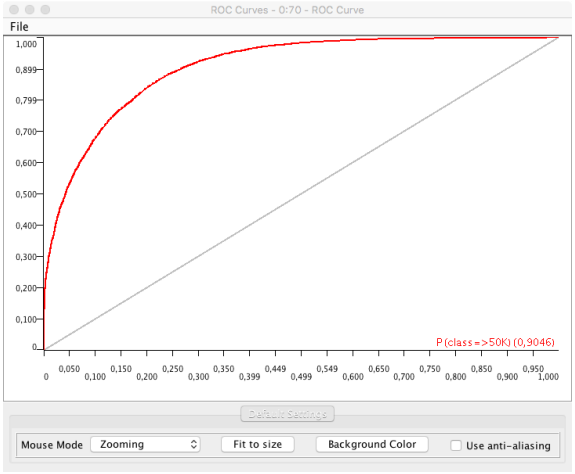


Figure 23: ROC Curve para MLP sin variables, con 5 capas ocultas y 20 neuronas por capa



4.3 Árboles de decisión: Algoritmo C4.5

Table 9: Resultados de las modificaciones con el algoritmo C4.5

Tipo	TP	FP	TN	FN	Rec.	Pre.	F-measure	Acc.	AUC	G-mean
C4.5 sin variables14	9419	2544	9106	2268	0.806	0.787	0.797	79.38%	0.8479	0.793
Sin variables 14 y con MDL prunning	9953	2396	9254	1734	0.852	0.806	0.828	82.303%	0.8939	0.882
Sin variables 14, con MDL prunning y 10 records por nodo	9980	2416	9234	1707	0.854	0.805	0.829	82.333%	0.8935	0.822
Sin variables 14, con MDL prunning y 15 records por nodo	10015	2470	9180	1672	0.857	0.802	0.829	82.251%	0.8905	0.821
Sin variables 14, con MDL prunning y 20 records por nodo	10010	2489	9161	1677	0.857	0.801	0.828	82.149%	0.8901	0.820

El primer cambio que se introdujo dentro del algoritmo C4.5 fue introducir un algoritmo de poda MDL [4]. Con la poda lo que conseguimos es evitar el sobreajuste haciendo el árbol que se obtiene generalice mejor, podando nodos hoja que no se ajusten a ciertas características.

Con el cambio de añadir más records mínimos por nodo, en el primer caso aumentando a 10 el número de records por nodo, se pudo observar una mejora en los TP con un descenso no muy acusado en los TN, pero conforme se fueron aumentando el número de *records* por nodo se aumentan los TP pero van descendiendo drásticamente los TN. Esto nos indica que si queremos más instancias en los nodos hoja, vamos a encontrar más instancias de la clase positiva que de la clase negativa.

Según lo que quisiese el cliente, podríamos jugar con estos parámetros para darle un modelo más ajustado a sus necesidades. Si quisiese una buena relación entre el número de TP y de TN les daría el modelo con MDL prunning a secas y el número de records por nodo por defecto en KNIME. En cambio, si lo que desea el cliente es obtener un número grande de TP, se le podria ofrecer el modelo con MDL prunning y 15 records por nodo como mínimo.

Figure 24: ROC Curve para el algoritmo C4.5 sin variables y con el algoritmo MDL de prunning

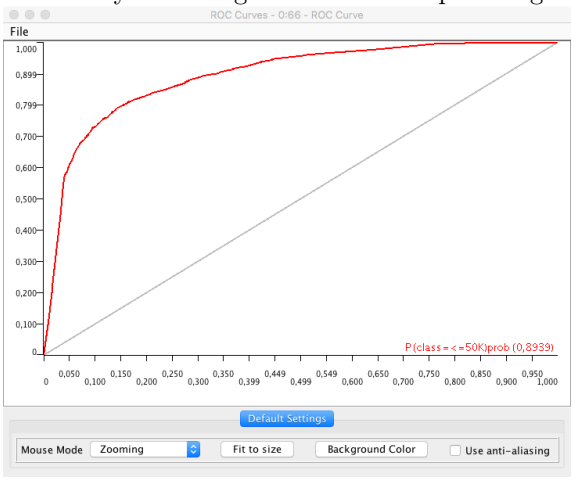


Figure 25: ROC Curve para el algoritmo C4.5 sin variables, con el algoritmo MDL de prunning y 10 records por nodo

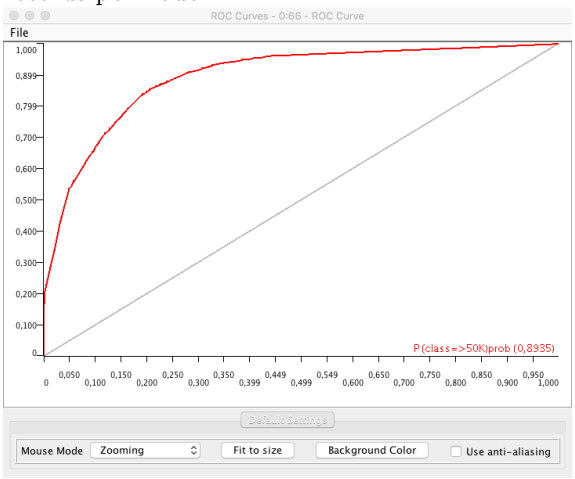


Figure 26: ROC Curve para el algoritmo C4.5 sin variables, con el algoritmo MDL de prunning y 15 records por nodo

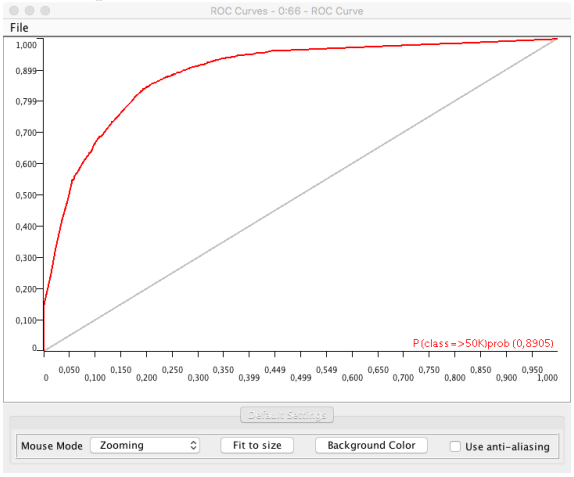
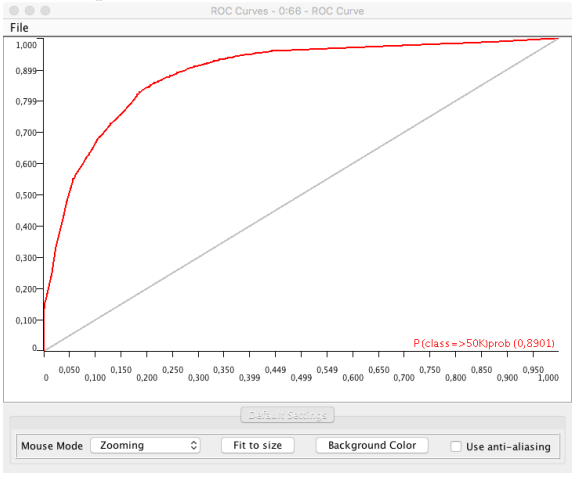


Figure 27: ROC Curve para el algoritmo C4.5 sin variables, con el algoritmo MDL de prunning y 20 records por nodo



4.4 AdaBoost

Table 10: Resultados de las modificaciones con el algoritmo Ada Boost

Tipo	TP	FP	TN	FN	Rec.	Pre.	F-measure	Acc.	AUC	G-mean
AdaBoost sin variables14	10026	2621	9029	1661	0.858	0.793	0.824	81,651%	0.8986	0.814
AdaBoost sin variables14 y con umbral de pesos a 200	10026	2621	9029	1661	0.858	0.793	0.824	81.651%	0.8986	0.814
AdaBoost sin variables14 y con umbral de pesos a 70	9984	3927	7723	1703	0.854	0.718	0.78	75.875%	0.7557	0.752
AdaBoost sin variables14, con umbral de pesos a 100 y 30 iteraciones	10032	2639	9011	1655	0.858	0.792	0.824	81.6%	0.8992	0.814
AdaBoost sin variables14, con umbral de pesos a 100 y 40 iteraciones	10032	2639	9011	1655	0.858	0.792	0.824	81.6%	0.8992	0.814

Con respecto a las modificaciones que se han realizado al algoritmo AdaBoost se puede observar como no han tenido mucho efecto sobre el propio algoritmo con los parámetros por defecto del paquete de Weka para KNIME.

Se puede observar como si se usa un umbral de pesos bajo, 70 en este caso, se reduce muchísimo el rendimiento del algoritmo. Cuando reducimos los pesos, estamos limitando cuánta importancia se le da a estos clasificadores más débiles.

También se ha podido observar como el número de iteraciones, a partir de 30 iteraciones hasta 40 al menos, no mejora el rendimiento del algoritmo en este dataset.

Figure 28: ROC Curve para el algoritmoAdaBoost sin variables y con umbral de pesos a 200

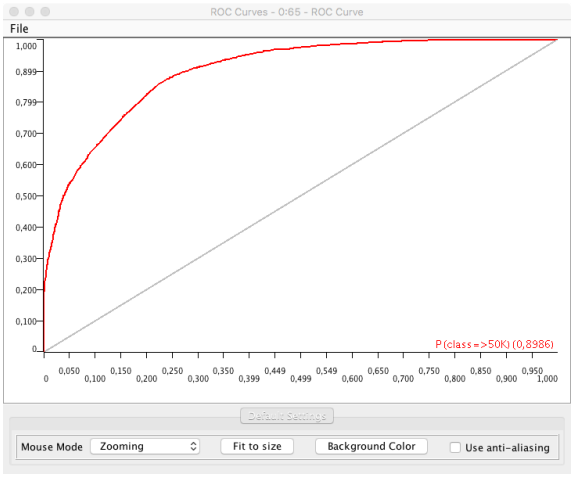


Figure 29: ROC Curve para el algoritmoAdaBoost sin variables y con umbral de pesos a 70

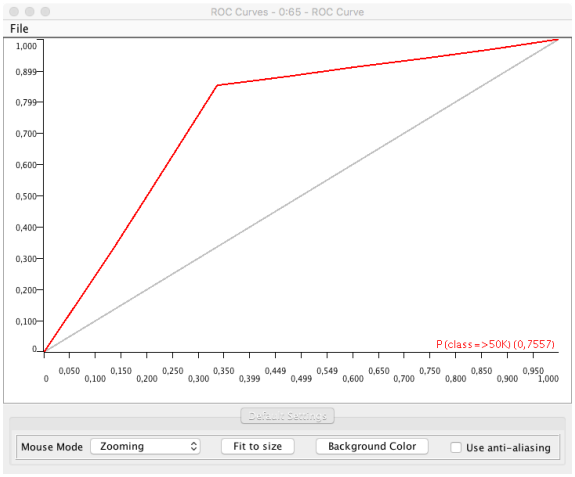


Figure 30: ROC Curve para el algoritmoAdaBoost sin variables, con umbral de pesos a 100 y 30 iteraciones

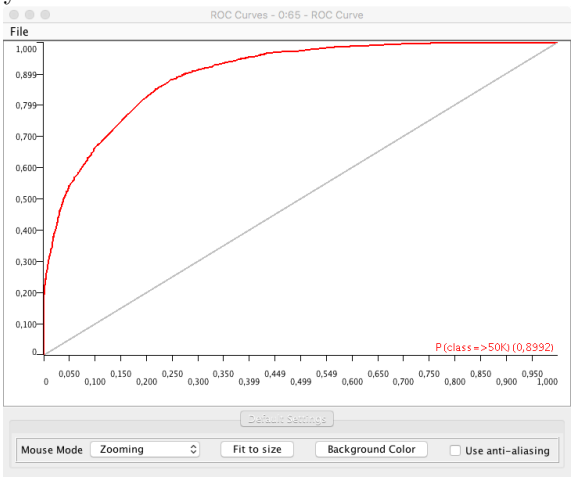
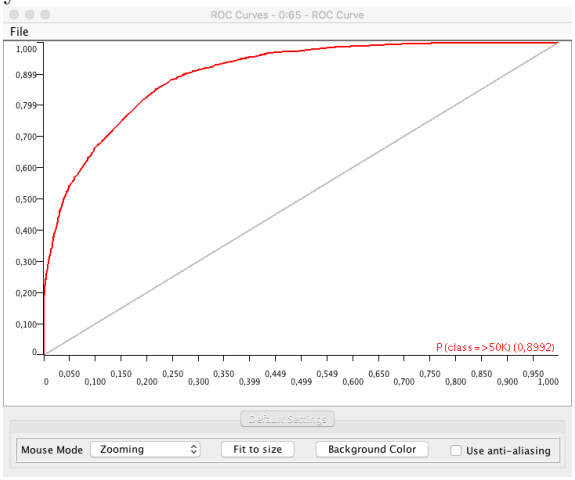


Figure 31: ROC Curve para el algoritmoAdaBoost sin variables, con umbral de pesos a 100 y 40 iteraciones



4.5 Gradient Boosting

Table 11: Resultados de las modificaciones con el algoritmo Gradient Boosting

Tipo	TP	FP	TN	FN	Rec.	Pre.	F-measure	Acc.	AUC	G-mean
Gradient Boosting sin variables14	10120	2144	9506	1567	0.866	0.825	0.845	84,098%	0.924	0.840
Gradient Boosting sin variables14 y con 200 modelos	10122	2121	9529	1565	0.866	0.827	0.846	84,205%	0.9258	0.841
Gradient Boosting sin variables14 y con 300 modelos	10115	2107	9543	1572	0.865	0.828	0.846	84,235%	0.9262	0.842
Gradient Boosting sin variables14 , con 300 modelos y learning rate 0.2	10121	2165	9485	1566	0.866	0.824	0.846	84,013%	0.9238	0.839
Gradient Boosting sin variables14 , con 300 modelos y learning rate 0.05	10136	2145	9505	1551	0.867	0.825	0.846	84,162%	0.9253	0.841

Como se puede observar en la Tabla 11, los resultados del Gradient Boosting mejora conforme más modelos utilizados. Esto es comprensible ya que a cuanto mayor número de modelos utilizamos, mejor podemos generalizar, ya que más modelos diferentes habrá que capten mejores características.

Dentro de las modificaciones, tanto por Accuracy y como por AUC, la mejor opción es usar Gradient Boost con 300 modelos y con learning rate 0.1.

El learning rate se ha comprobado empíricamente que mejora la generalización para valores ≤ 0.1 [5].

Figure 32: ROC Curve para el algoritmo Gradient Boosting sin variables y con 200 modelos

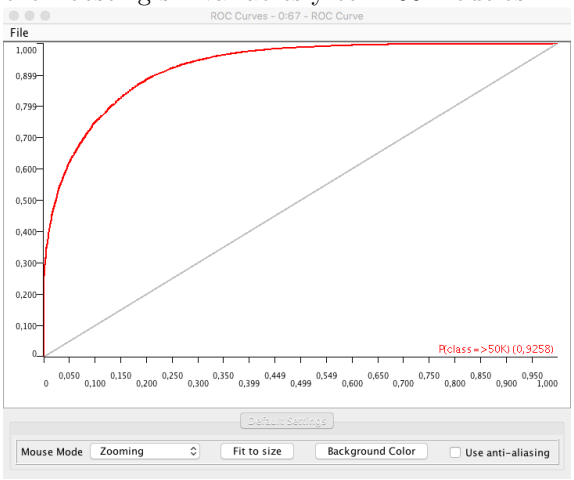


Figure 33: ROC Curve para el algoritmo Gradient Boosting sin variables y con 300 modelos

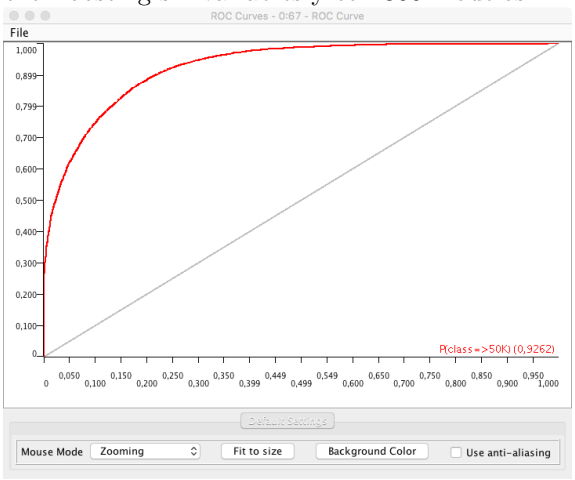


Figure 34: ROC Curve para el algoritmo Gradient Boosting sin variables, con 300 modelos y learning rate 0.2

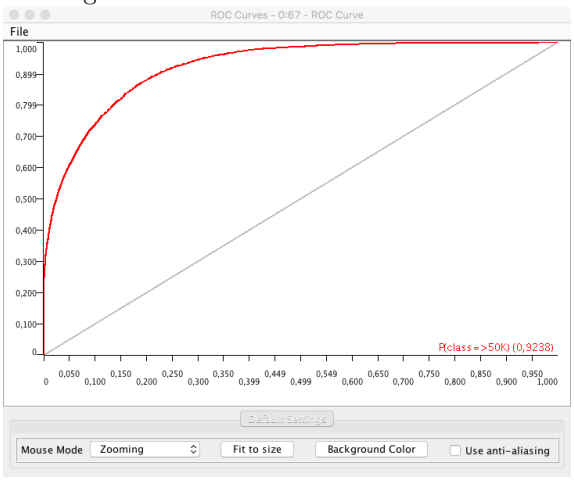
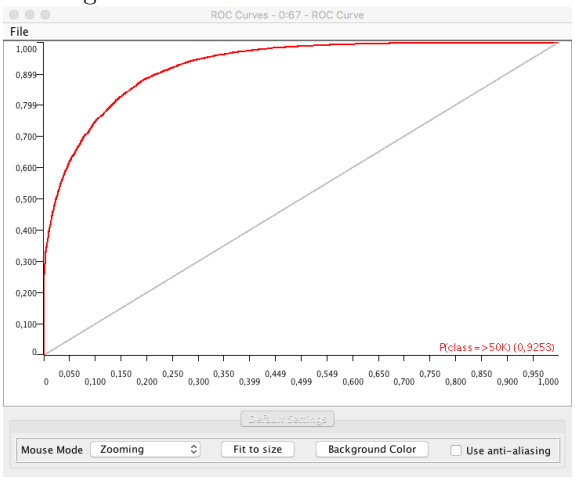


Figure 35: ROC Curve para el algoritmo Gradient Boosting sin variables, con 300 modelos y learning rate 0.05



4.6 Random Forest

Table 12: Resultados de las modificaciones con el algoritmo Random Forest

Tipo	TP	FP	TN	FN	Rec.	Pre.	F-measure	Acc.	AUC	G-mean
Random Forest sin variables14	10170	2350	9300	1517	0.87	0.812	0.84	83,437%	0.903	0.838
Random Forest sin variables14 y con 150 modelos	10176	2374	9276	1511	0.871	0.811	0.84	83,353%	0.9037	0.832
Random Forest sin variables14 y con 200 modelos	10192	2384	9266	1495	0.872	0.81	0.84	83,378%	0.9041	0.832
Random Forest sin variables14 con 200 modelos y un límite de 10 niveles	10336	3306	8344	1321	0.887	0.758	0.818	80,173%	0.8880	0.796
Random Forest sin variables14 con 200 modelos y un límite de 30 niveles	10202	2403	9247	1485	0.873	0.809	0.84	83,34%	0.9029	0.831

Se puede observar en la Tabla 12 que si se aumenta el número de modelos, se produce un aumento de los TP y una disminución de los TN. Esto nos lleva al caso comentado con el algoritmo C4.5 4.3, es decir, según lo que desease el cliente se le podría ofrecer un mejor equilibrio entre TP y TN o un modelo que clasifica mejor una de las clases.

Cuantos más modelos, estamos añadiendo más árboles de decisión, por lo que podemos estar introduciendo más árboles que no nos aporten conocimiento, o que introduzcan ruido. Es por esto que puede ser que a mayor número de modelos mejores se aprendiesen los atributos de la clase positiva pero perdiesemos valor de clasificación en la clase negativa.

Cuando se ha pretendido modificar el límite de niveles que se podían alcanzar con los árboles, es decir, se les obliga a generalizar más y no ceñirse tanto al conjunto de entrenamiento, se observa como se consiguen muy buenos resultados (literalmente los mejores de las modificaciones) en las clase positiva, pero sin embargo, re obtienen muy malos resultados en la clase negativa.

Esto lo que nos indica es que el problema es complejo, y necesita unos árboles de decisión más grandes para poder generalizar mejor.

Figure 36: ROC Curve para el algoritmo Random Forest sin variables y con 150 modelos

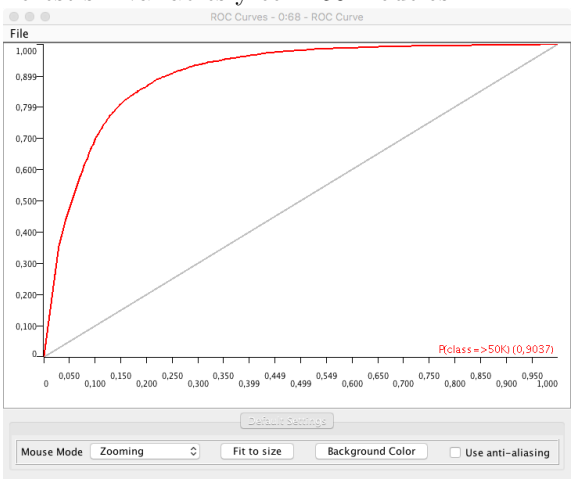


Figure 37: ROC Curve para el algoritmo Random Forest sin variables y con 200 modelos

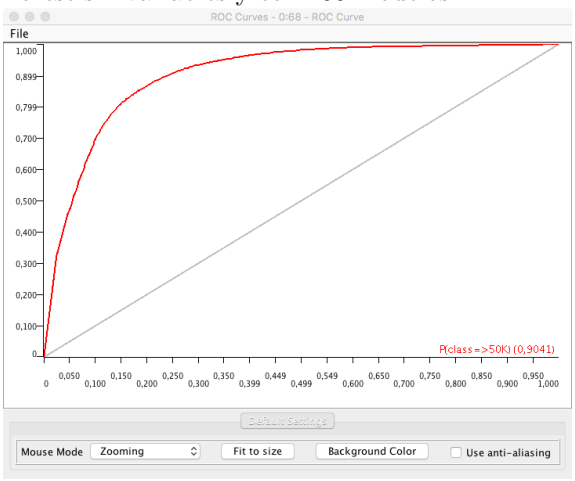


Figure 38: ROC Curve para el algoritmo Random Forest sin variables y con 200 modelos y un límite de 10 niveles

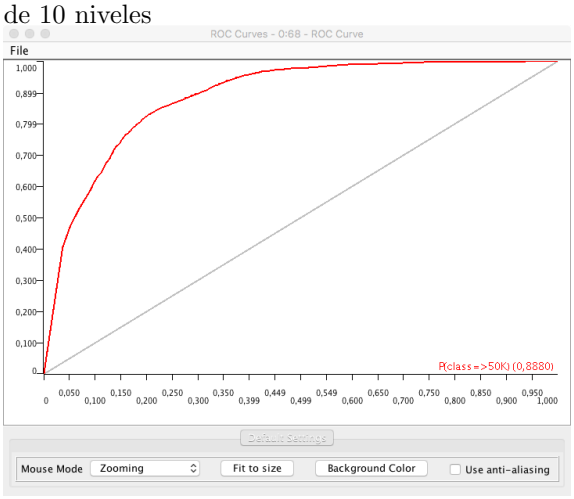
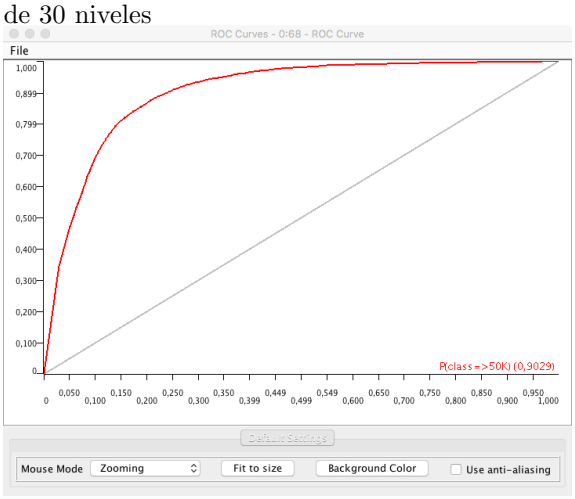


Figure 39: ROC Curve para el algoritmo Random Forest sin variables y con 200 modelos y un límite de 30 niveles



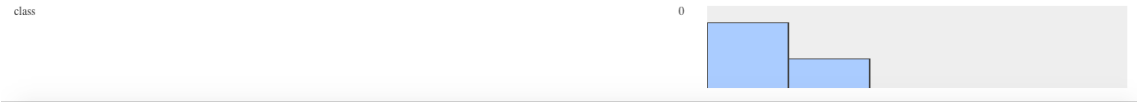
5 Procesado de datos

Para el procesado de los datos se realizaron distintas tareas que nos diesen una vista general de como se encontraban estos datos.

La primera que se realizó fue convertir las variables nominales en variables numéricas gracias al nodo **One-to-many**.

Una de ellas fue usar el nodo *statistics* para ver como era la distribución de los distintos atributos y sobre todo para ver si existía un desbalanceo entre las dos clases.

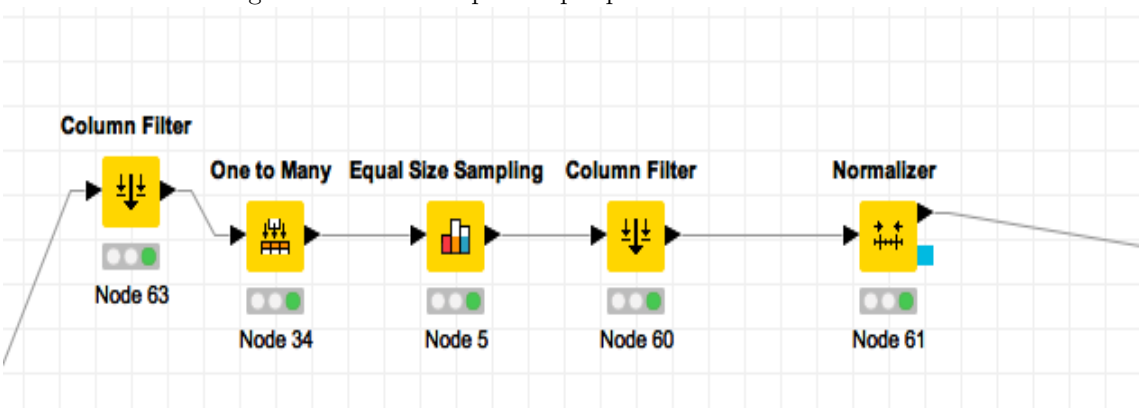
Figure 40: Diferencia entre las dos clases del conjunto de datos



Vista la diferencia entre las dos clases que se puede observar en la Figura 40, se decidió hacer uso del nodo **Equal Size Sampling** para hacer una igualación del número de instancias en las dos clases, de forma que los resultados obtenidos fuesen más robustos y realistas.

Este *WorkFlow* se puede observar en la siguiente imagen:

Figure 41: Workflow para el pre procesamiento de los datos



A continuación, se decidió hacer uso del nodo **Rank Correlation** para observar si existía una correlación grande o baja entre las distintas variables y la clase.

Como se puede observar en la Figura 54, existe una correlación mayor y menor entre las distintos atributos y la clase. Los atributos con mayor correlación con la clase son: **age, education-num, sex, capital_gain ,hours-per-week, capital_loss, marital y relationship**. Los p values de la correlación son los siguientes:

Table 13: Variables más correlacionadas con la clase y sus valores de correlación

Atributo	**Correlación Clase**
Age	0,2722
Education-num	0,3285
Marital Status	-0,2289
Relationship	-0,3356
Sex	0,2518
Capital Gain	0,2771
Capital Loss	0,1392
Hours Per Week	0,2658

Sin embargo, los atributos que tienen menos correlación con la clase son los siguientes: **native-country,fnlwgt,education y workclass**. Los p values de la correlación son los siguientes:

Table 14: Variables menos correlacionadas con la clase y sus valores de correlación

Atributo	**Correlación Clase**
Workclass	0,0264
Fnlwgt	-0,0064
Education	0,0326
Native-Country	0,0336

Cómo las variables nominales se querían pasar a variables numéricas usando el nodo **one-to-many** se decidió volver a hacer uso del nodo **Rank Correlation** para comprobar si aunque la variable tuviese un bajo valor de correlación con la clase como se puede observar en 5, quizás algun valor del atributo si tuviese una buena correlación. Es por esto que se comprobó la tabla resultado que se puede observar en la Figura 55, que ningún valor de los atributos nominales pasados a numéricos aumentaba su correlación con la clase.

Por esto, se decidió eliminar del *dataset* los atributos que obtenían una baja correlación 5 para los experimentos posteriores al primer experimento.

Los resultados que se obtuvieron realizando esta eliminación en los resultados de los algoritmos sin modificar sus parámetros fueron los siguientes:

Table 15: Valores de los experimentos entre los primeros experimentos y una vez eliminadas las variables menos correlacionadas

Tipo	**TP**	**FP**	**TN**	**FN**	**Rec.**	**Pre.**	**F-measure**	**Acc.**	**AUC**	**G-mean**
KNN PE EQ	9111	2763	8887	2576	0.78	0.767	0.7773	77.122%	0.8314	0.77
KNN SV 5	9335	2851	8700	2353	0.799	0.766	0.782	77.705%	0.8133	0.769
MLP PE EQ	9933	2502	9148	1754	0.85	0.799	0.824	81.763%	0.8995	0.816
MLP SV 5	10136	2546	9104	1551	0.867	0.799	0.832	82,444%	0.907	0.823
C4.5 PE EQ	9147	2575	9075	2540	0.783	0.78	0.781	78.082%	0.8019	0.780
C4.5 SV 5	9419	2544	9106	2268	0.806	0.787	0.797	79.38%	0.8479	0.793
AB PE EQ	10064	2674	8976	1623	0.861	0.79	0.824	81.587%	0.8968	0.814
AB SV 5	10026	2621	9029	1661	0.858	0.793	0.824	81,651%	0.8986	0.814
GB PE EQ	10138	2160	9490	1549	0.867	0.824	0.845	84,107%	0.924	0.840
GB SV 5	10120	2144	9506	1567	0.866	0.825	0.845	84,098%	0.924	0.840
RF PE EQ	10092	2272	9378	1595	0.864	0.816	0.805	83,43% s	0.907	0.833
RF SV 5	10170	2350	9300	1517	0.87	0.812	0.84	83,437%	0.903	0.838

PE: Primer experimento, usando Equal Sampling, SV: Sin variables, usando Equal Sampling. EQ: Equal Sampling

Figure 42: ROC Curve para el algoritmo KNN primer experimento

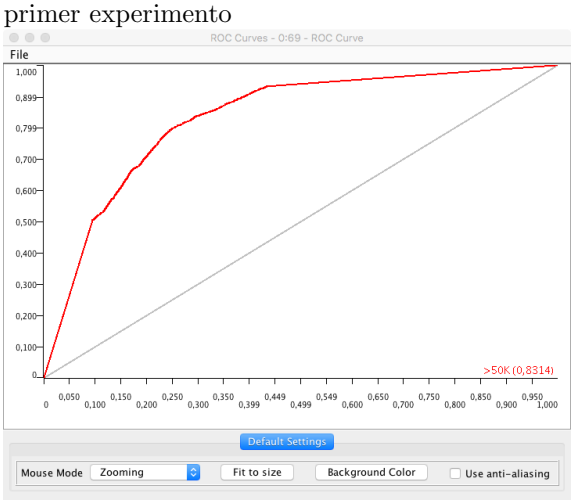


Figure 43: ROC Curve para el algoritmo KNN sin variables 5

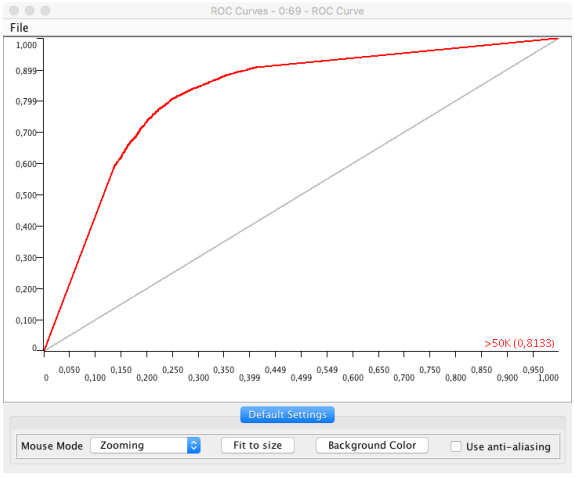


Figure 44: ROC Curve para el algoritmo Neural Networks primer experimento

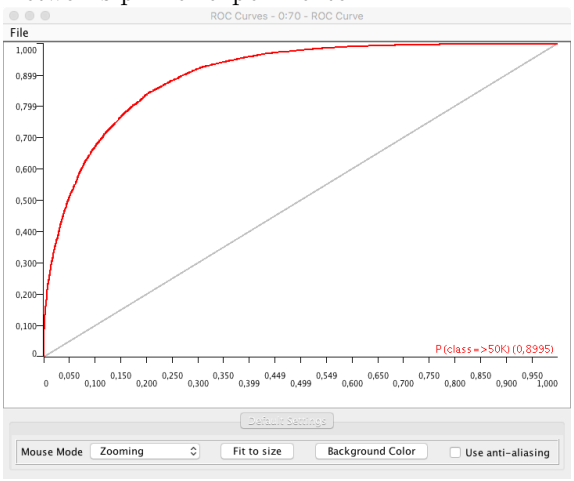


Figure 45: ROC Curve para el algoritmo Neural Networks sin variables 5

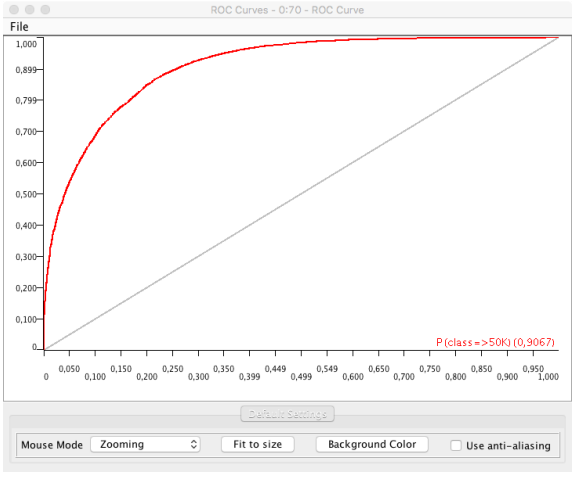


Figure 46: ROC Curve para el algoritmo C4.5 sin variables

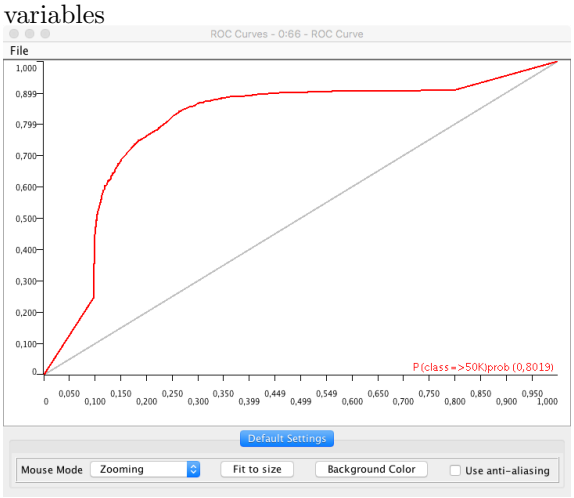


Figure 47: ROC Curve para el algoritmo C4.5 sin variables 5

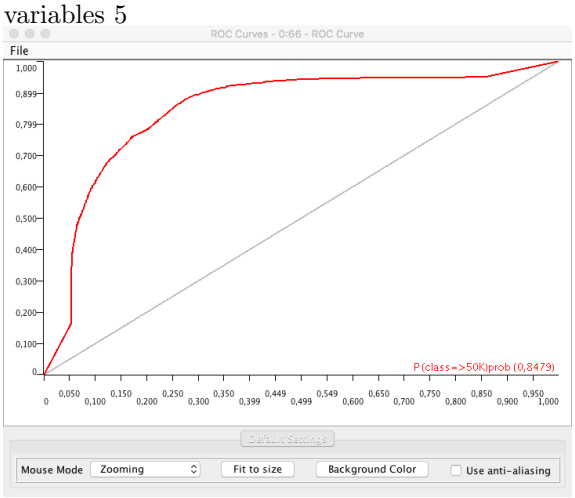


Figure 48: ROC Curve para el algoritmo AdaBoost primer experimento

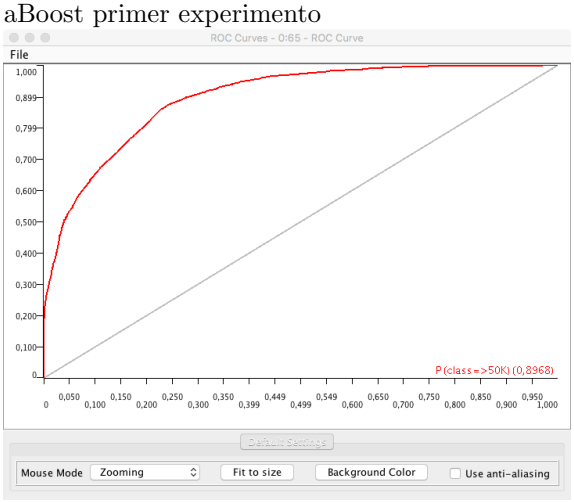


Figure 49: ROC Curve para el algoritmo AdaBoost sin variables 5

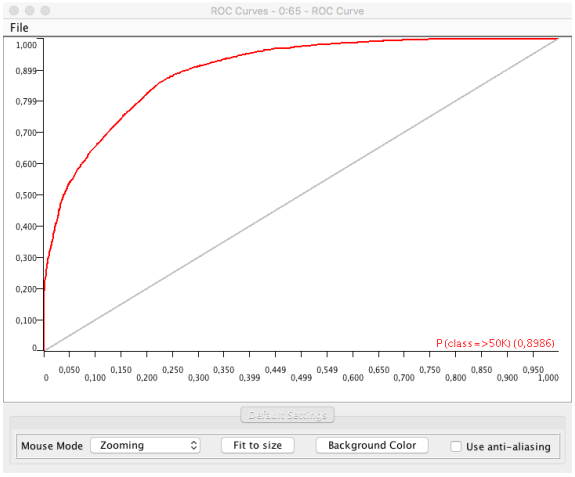


Figure 50: ROC Curve para el algoritmo Gradient Boosting primer experimento

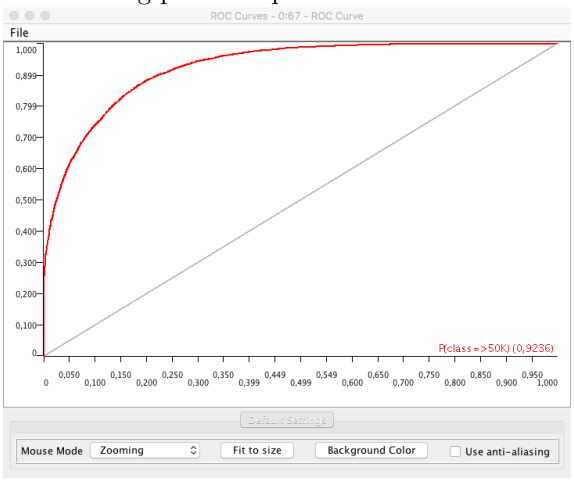


Figure 51: ROC Curve para el algoritmo Gradient Boosting sin variables 5

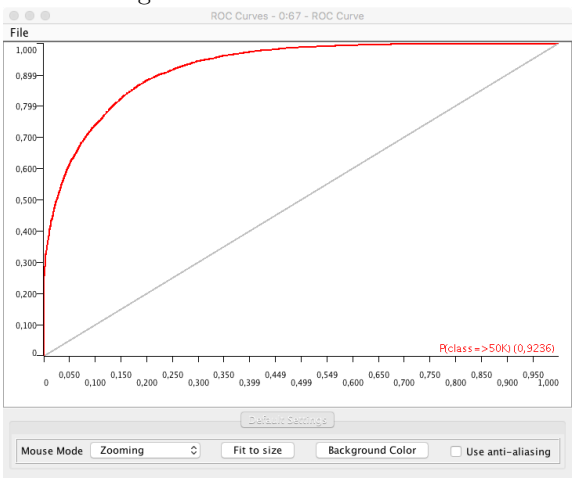


Figure 52: ROC Curve para el algoritmo Random Forest primer experimento

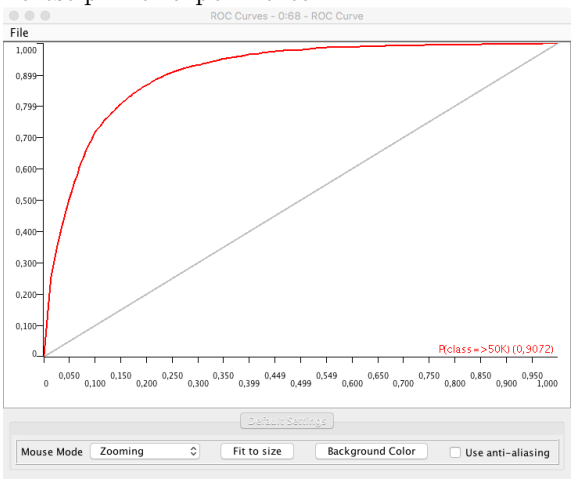


Figure 53: ROC Curve para el algoritmo Random Forest sin variables 5

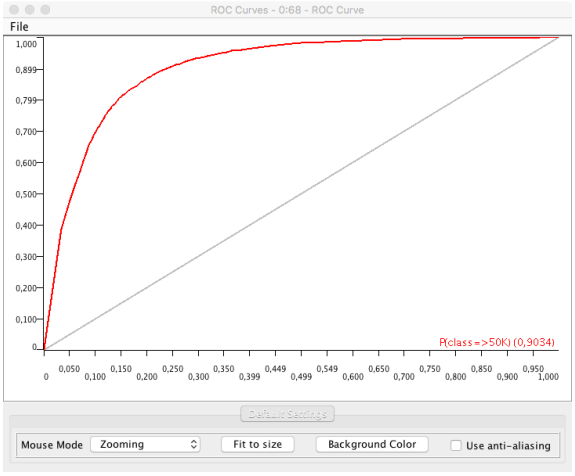


Figure 54: Rank correlation entre las distintas variables del conjunto de datos

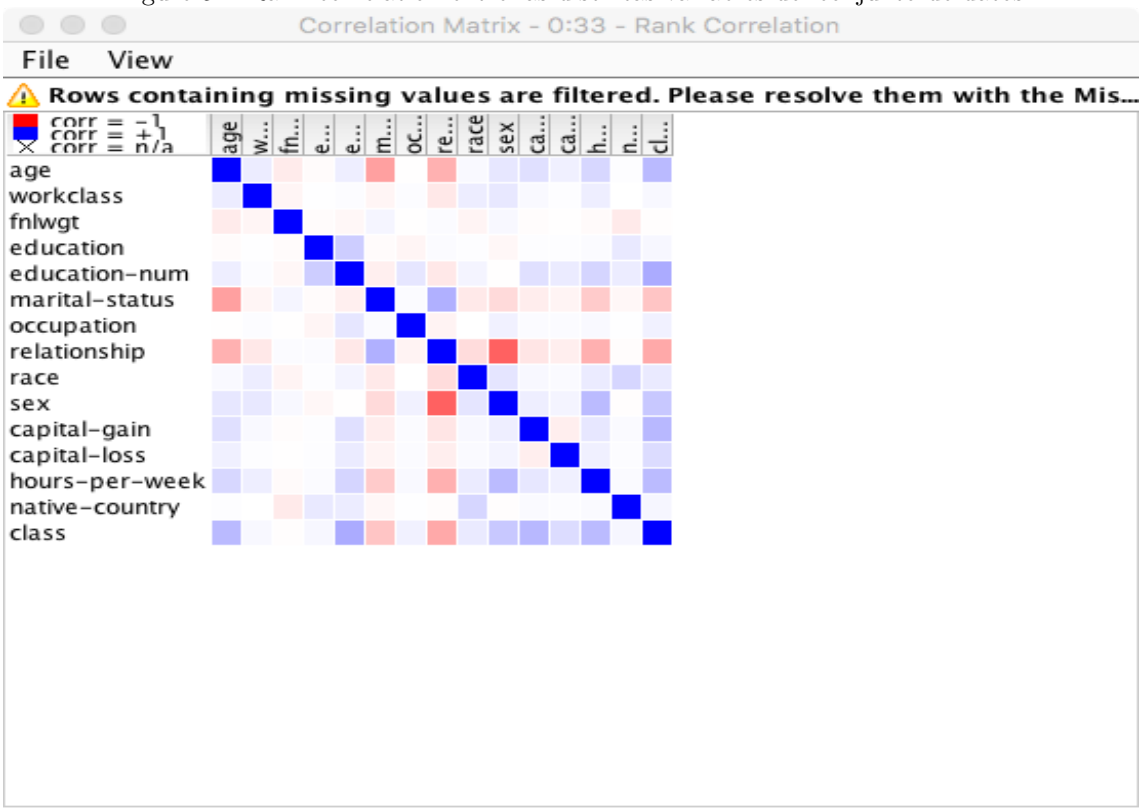
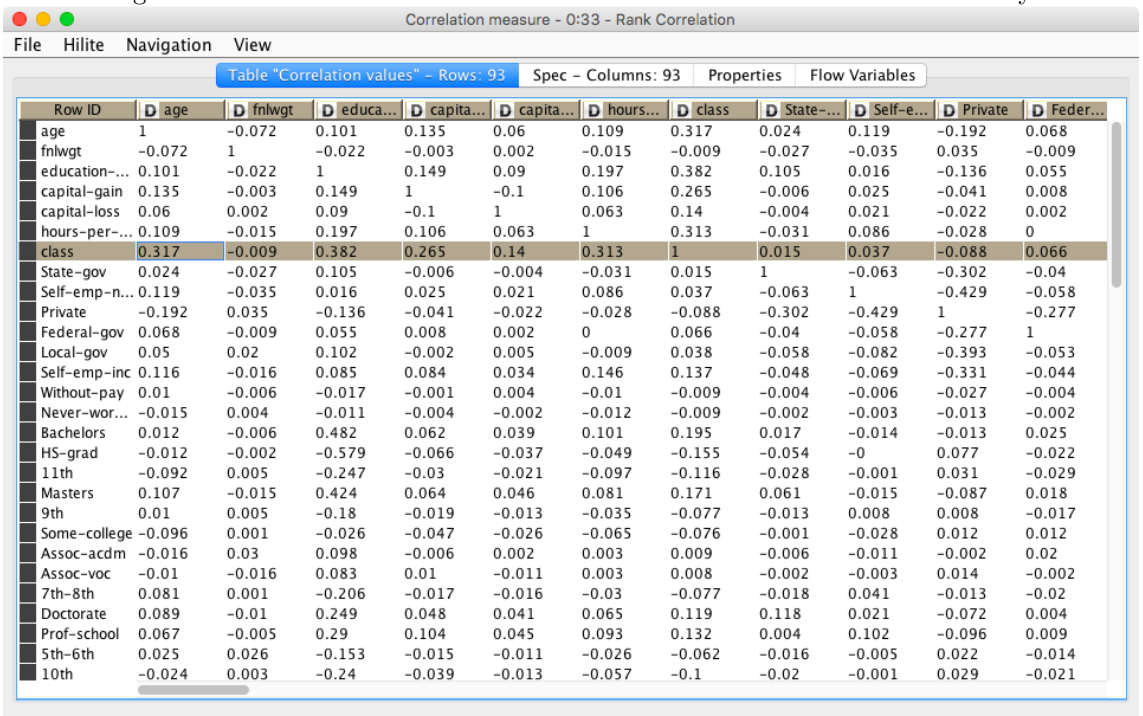
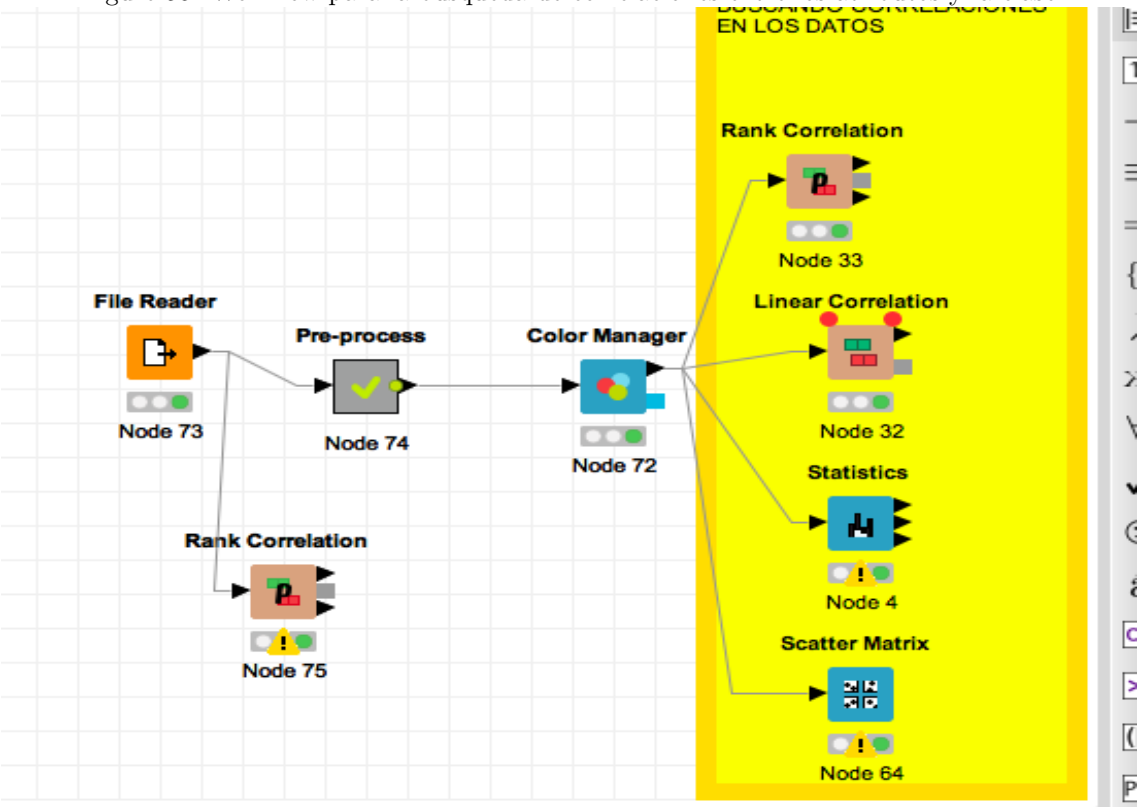


Figure 55: Valores del nodo Rank correlation una vez realizado el One-to-many



Row ID	age	fnlwgt	educa...	capita...	capita...	hours...	class	State-...	Self-e...	Private	Feder...
age	1	-0.072	0.101	0.135	0.06	0.109	0.317	0.024	0.119	-0.192	0.068
fnlwgt	-0.072	1	-0.022	-0.003	0.002	-0.015	-0.009	-0.027	-0.035	0.035	-0.009
education-...	0.101	-0.022	1	0.149	0.09	0.197	0.382	0.105	0.016	-0.136	0.055
capital-gain	0.135	-0.003	0.149	1	-0.1	0.106	0.265	-0.006	0.025	-0.041	0.008
capital-loss	0.06	0.002	0.09	-0.1	1	0.063	0.14	-0.004	0.021	-0.022	0.002
hours-per-...	0.109	-0.015	0.197	0.106	0.063	1	0.313	-0.031	0.086	-0.028	0
class	0.317	-0.009	0.382	0.265	0.14	0.313	1	0.015	0.037	-0.088	0.066
State-gov	0.024	-0.027	0.105	-0.006	-0.004	-0.031	0.015	1	-0.063	-0.302	-0.04
Self-emp-n...	0.119	-0.035	0.016	0.025	0.021	0.086	0.037	-0.063	1	-0.429	-0.058
Private	-0.192	0.035	-0.136	-0.041	-0.022	-0.028	-0.088	-0.302	-0.429	1	-0.277
Federal-gov	0.068	-0.009	0.055	0.008	0.002	0	0.066	-0.04	-0.058	-0.277	1
Local-gov	0.05	0.02	0.102	-0.002	0.005	-0.009	0.038	-0.058	-0.082	-0.393	-0.053
Self-emp-inc	0.116	-0.016	0.085	0.084	0.034	0.146	0.137	-0.048	-0.069	-0.331	-0.044
Without-pay	0.01	-0.006	-0.017	-0.001	0.004	-0.01	-0.009	-0.004	-0.006	-0.027	-0.004
Never-wor...	-0.015	0.004	-0.011	-0.004	-0.002	-0.012	-0.009	-0.002	-0.003	-0.013	-0.002
Bachelors	0.012	-0.006	0.482	0.062	0.039	0.101	0.195	0.017	-0.014	-0.013	0.025
HS-grad	-0.012	-0.002	-0.579	-0.066	-0.037	-0.049	-0.155	-0.054	-0	0.077	-0.022
11th	-0.092	0.005	-0.247	-0.03	-0.021	-0.097	-0.116	-0.028	-0.001	0.031	-0.029
Masters	0.107	-0.015	0.424	0.064	0.046	0.081	0.171	0.061	-0.015	-0.087	0.018
9th	0.01	0.005	-0.18	-0.019	-0.013	-0.035	-0.077	-0.013	0.008	0.008	-0.017
Some-college	-0.096	0.001	-0.026	-0.047	-0.026	-0.065	-0.076	-0.001	-0.028	0.012	0.012
Assoc-acdm	-0.016	0.03	0.098	-0.006	0.002	0.003	0.009	-0.006	-0.011	-0.002	0.02
Assoc-voc	-0.01	-0.016	0.083	0.01	-0.011	0.003	0.008	-0.002	-0.003	0.014	-0.002
7th-8th	0.081	0.001	-0.206	-0.017	-0.016	-0.03	-0.077	-0.018	0.041	-0.013	-0.02
Doctorate	0.089	-0.01	0.249	0.048	0.041	0.065	0.119	0.118	0.021	-0.072	0.004
Prof-school	0.067	-0.005	0.29	0.104	0.045	0.093	0.132	0.004	0.102	-0.096	0.009
5th-6th	0.025	0.026	-0.153	-0.015	-0.011	-0.026	-0.062	-0.016	-0.005	0.022	-0.014
10th	-0.024	0.003	-0.24	-0.039	-0.013	-0.057	-0.1	-0.02	-0.001	0.029	-0.021

Figure 56: Workflow para la búsqueda de correlaciones entre los atributos y la clase



6 Interpretación de resultados

Hemos podido observar en la sección 3 que los algoritmos más potentes son aquellos conocidos como algoritmos de caja negra, que son aquellos en los que no podemos saber porque toman la decisión que toman. Esto es lo más común en la literatura y en el mundo laboral.

Pero en el momento que queremos saber porque se toman las decisiones que se toman, tenemos que irnos al algoritmo como el C4.5 para saber las decisiones que toma en que se basan.

Es por ello que vamos a observar en KNIME el árbol de decisión obtenido para comprobar cuál es el camino que toma. Para ello primero tomaremos el árbol que se obtiene en los primeros experimentos 3 y a continuación, tomaremos el árbol con el que se obtiene un resultado más balanceado que se puede observar en la Tabla 9. En este caso, con el que se obtiene un resultado más balanceado es el en el que se eliminan las variables menos correlacionadas 14
y con se utiliza el algoritmo MDL de poda.

6.1 Primer modelo algoritmo C4.5

En este caso vamos a observar como se genera el árbol de decisión para el último fold del cross validation para ver si nos da muestras de como va haciendo los distintos splits.

Si observamos la vista simple del decision tree, que podemos observar en la Figura 57, se puede observar que el primer split que realiza se basa en la **relationship** de la persona. Se puede observar como las personas que se encuentran casadas, tienen más probabilidad de pertenecer a la clase positiva, mientras que las personas que no están casadas, son hijos únicos o se encuentran en otro tipo de relación casi en su mayoría pertenecen a la clase negativa.

Pasemos a observar con que atributo se realiza el split en el siguiente nivel. Si bajamos por el nodo en el que el **relationship** es igual a **husband**, podemos observar que el siguiente atributo que se utiliza para el split es el de **education**, como se puede observar en la Figura 58. Se puede observar en la figura como los estudios superiores (estudios universitarios), aumentan las posibilidades de pertenecer a la clase positiva, mientras que los estudios medios (la educación secundaria) si es el máximo nivel de educación que tienes, indica que con bastante probabilidad pertenezcas a la clase negativa.

Estos serían los dos principales diferenciadores entre las dos clases en este árbol de decisión, el estado civil y el grado educativo máximo que tiene la persona. A continuación, dependiendo ya de los distintos nodos, en los siguientes nodos se puede realizar el split por diversos atributos, desde el **capital-gain** al **occupation**.

Figure 57: Primer Split del Decision Tree en el primer experimento

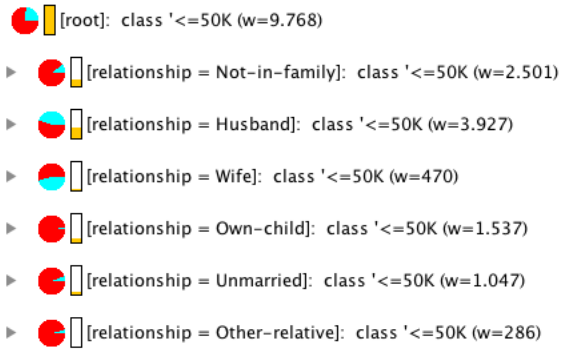
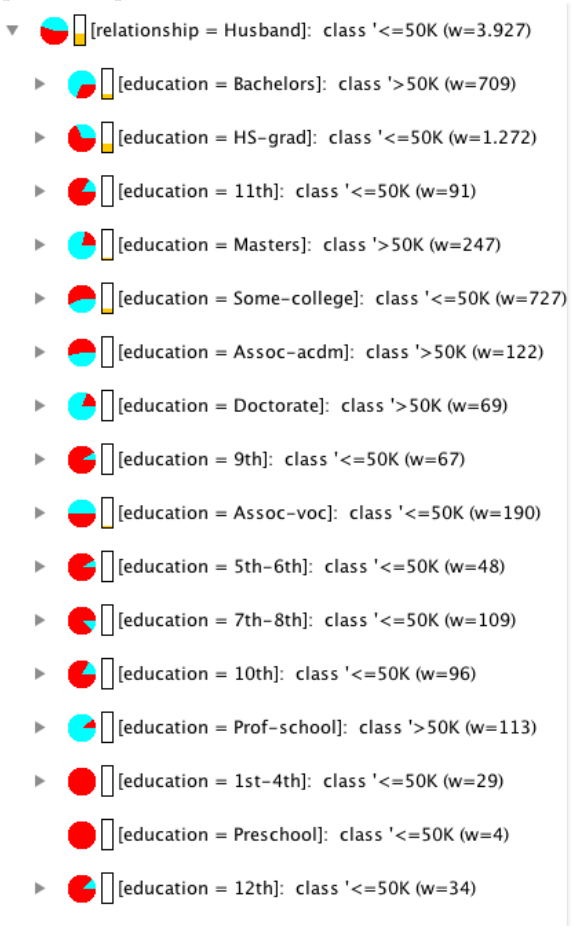


Figure 58: Segundo Split del Decision Tree en el primer experimento



6.2 Modelo más balanceado del algoritmo C4.5

Para este modelo se debe recordar que se le aplicó el pro-procesamiento explicado en 5, además de la eliminación de los atributos que se encuentran en la Tabla 14.

Partiendo de esto, el primer split se puede observar que sigue las directivas que se han podido observar en el apartado. El primer split, que se puede observar el la Figura 59, separa entre si el estado civil de la persona es esposo o no. En la Figura 57 habíamos observado como las personas casadas tenían más probabilidad de pertenecer a la clase positiva, por lo que este split que vemos en decision tree modificado continúa esta línea.

En el segundo split se puede observar también que las personas que tienen estudios superiores y están casadas casi en su totalidad pertenecen a la clase positiva, mientras que las que no tienen estudios superiores puede variar.

En el caso de que veamos el segundo split de las personas que no se encuentran casadas, podemos observar que el split que se realiza a continuación se basa en el **capital-gain** en lugar de en el **education-num**, como se puede observar en la Figura 61. También, dependiendo del capital-gain, si ganas más del valor de corte, se clasifica casi en su totalidad en la clase positiva correctamente.

Figure 59: Primer Split del Decision Tree en la modificación 9

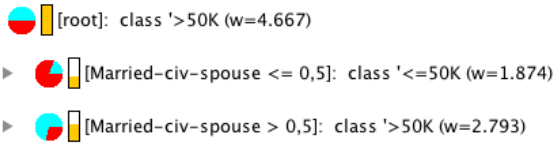


Figure 60: Segundo Split del Decision Tree en la modificación 9 si estas casado

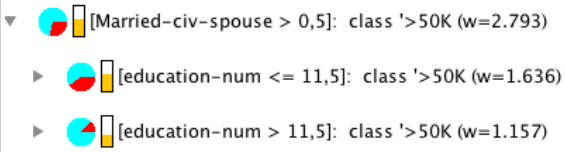
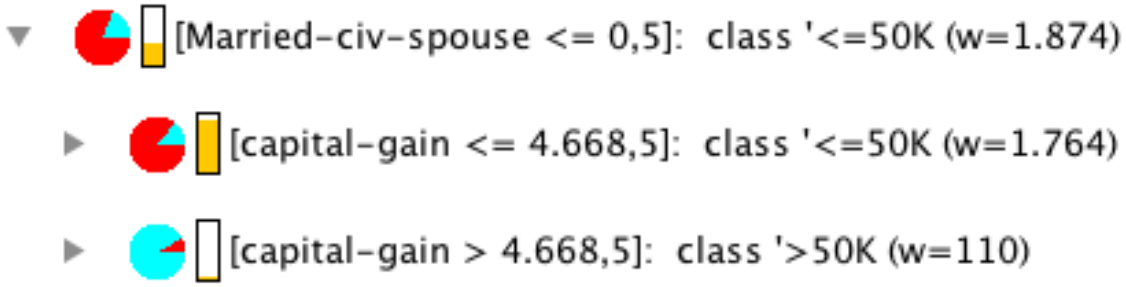


Figure 61: Segundo Split del Decision Tree en la modificación 9 si no estas casado



6.3 Uso de los historiogramas para ver las correlaciones con las dos clases

En este caso, vamos a hacer uso de los historiogramas para observar qué valores de las variables más correlacionadas con las clases hacen que sea más probable que pertenezcas a una clase o a otra. Vamos a explorar las atributos que se observó en la Tabla 13 que eran las que se encontraban más relacionadas con la clase. La clase positiva es la de color azul y la clase negativa la de color rojo.

En primer lugar comencemos con el atributo **Sex**. En la Figura 62 se puede observar como para un hombre es más probable pertenecer a la clase positiva que a la negativa mientras que en el caso de las mujeres es al contrario.

En segundo lugar, se evalúa el atributo **Age**. Si vemos la Figura 63, se puede observar como la edad en la que más probabilidad hay de que pertenezca a la clase positiva es entre los 33-49 años, mientras que en la tercera casi todas las muestras en ese rango de edad pertenecen a la clase negativa y lo mismo ocurre con las muestras cuya edad se encuentran entre 17-25 años.

En tercer lugar veamos el atributo **Relationship**. Se puede observar en la Figura 64 que las muestras que se encuentran dentro de un matrimonio tienen muchas más posibilidades de pertenecer a la clase positiva que las que se encuentran fuera de este.

En cuarto lugar, se evalúa el atributo **Race**. En la Figura 65 se puede observar como si la persona es de raza blanca tiene muchas más posibilidades de pertenecer a la clase positiva, aunque se reparte en más o menos un 50% entre ambas clases. En cambio, se puede observar como en los otros valores que puede tomar el atributo hay una mayor prevalencia de pertenecer a la clase negativa.

Por último, veamos el atributo **Education-Num**. Si vemos la Figura 66, se puede observar como cuanto mayor es el grado de estudio alcanzado más posibilidades hay de pertenecer a la clase positiva que si el nivel educativo alcanzado es más bajo.

Figure 62: Histograma del atributo Sex

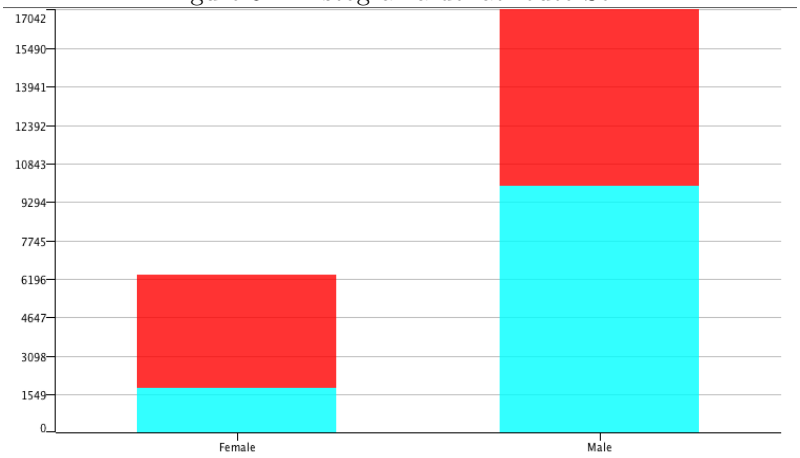


Figure 63: Histograma del atributo Age

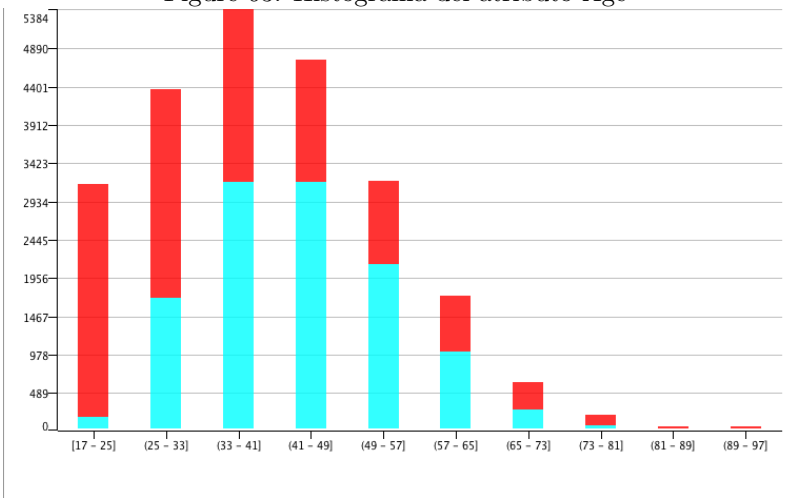


Figure 64: Histograma del atributo Relationship

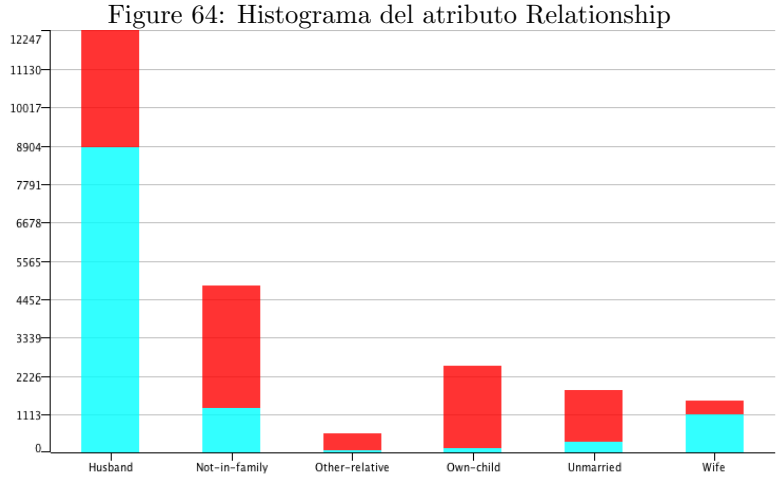


Figure 65: Histograma del atributo Race

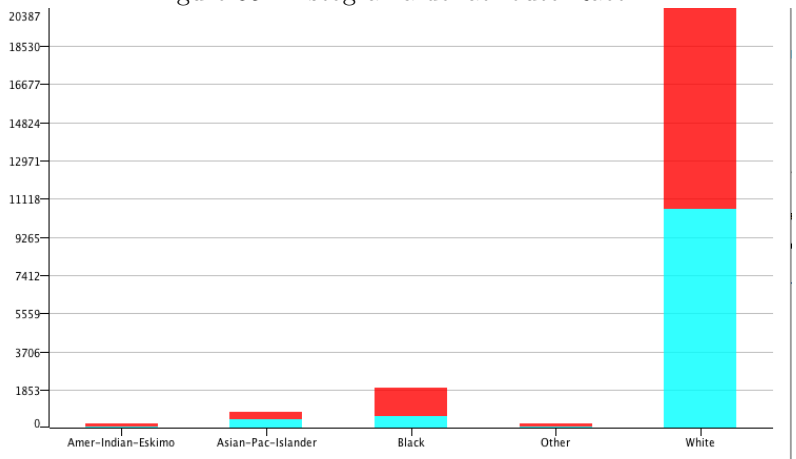
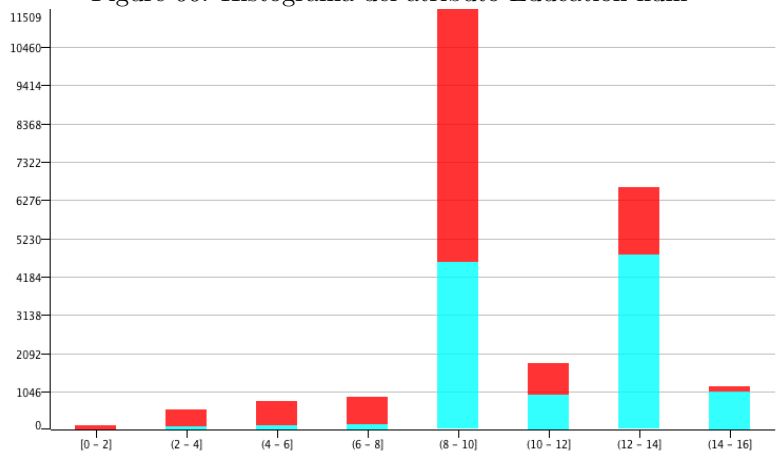


Figure 66: Histograma del atributo Education-num



7 Contenido Adicional

8 Bibliografía

References

[1] Center for Machine Learning and Intelligent Systems. Adult Data Set. `https://archive.ics.uci.edu/ml/datasets/Adult`. [Online; accessed 14-Octubre-2017].

[2] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *Ann. Statist.*, 29(5):1189–1232, 10 2001.

[3] Leo Breiman Statistics and Leo Breiman. Random forests. In *Machine Learning*, pages 5–32, 2001.

[4] Manish Mehta, Jorma Rissanen, and Rakesh Agrawal. Mdl-based decision tree pruning. pages 216–221. AAAI Press, 1995.

[5] T Hastie, T Tibshirani, and J Firedman. *10. Boosting and Additive Trees*. Springer, 2009.