Inteligencia de Negocio

Pablo Jesús Jiménez Ortiz
Grupo IN2 (Jueves)

Noviembre de 2019

Contents

1 Introducción

En esta primera práctica de introducción a la Inteligencia de Negocio, hemos de usar algoritmos de aprendizaje supervisado con el fin de realizar análisis predictivos en una empresa.

El conjunto de datos a analizar pertenece a bombas de agua en Tanzania, de las cuales conocemos variables tan heterogéneas como el año de instalación, en qué región del país fueron instaladas o la calidad del agua. El objetivo en este caso es la detección precoz de fallos en dichas bombas de agua.

En este primer intento de introducción a la Inteligencia de Negocio, se han considerado algoritmos clásicos así como algunos más novedosos para aportar diversidad a los resultados y de esta forma poder interpretar y comparar el comportamiento de dichos algoritmos de forma más adecuada, además de destacar con esto la necesidad de seguir avanzando en el desarrollo de este tipo de algoritmos.

Para el experimento se ha realizado una validación cruzada de 5 particiones con una semilla aleatoria fijada. Esta configuración es seguida por cada uno de los algoritmos usados.

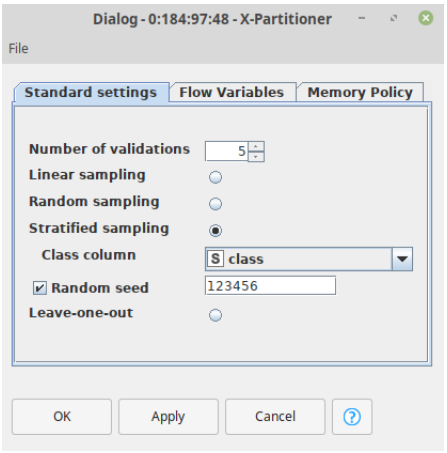

Figure 1: Particionado de la validación cruzada

2 Resultados obtenidos

A continuación y para cada uno de los algoritmos seleccionados, se muestran los resultados obtenidos a través de los cuales conoceremos algunas de las estadísticas más usuales a la hora de medir el rendimiento de este tipo de algoritmos.

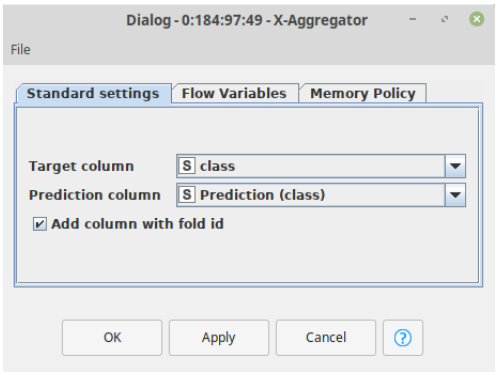

Figure 2: Final del bucle de validación cruzada

2.1 KNN

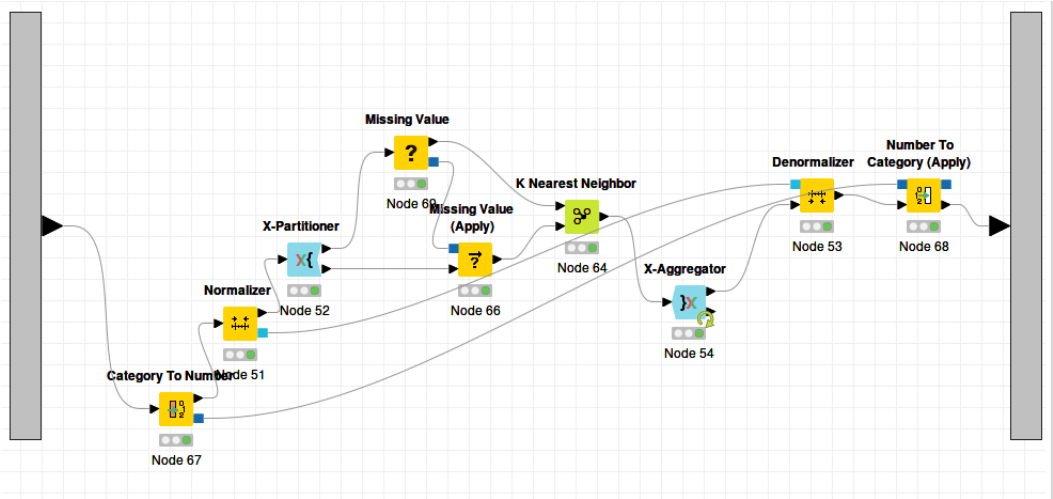

Figure 3: Esquema general del nodo KNN

A la vista de esta configuración, la matriz de confusión muestra los siguientes resultados:

row ID	functional	non functional	functional needs repair
functional	2597	739	981
non functional	836	2848	633
functional needs repair	819	515	2983

Table 1: Matriz de confusión

Correct classified: 46.183 Wrong classified: 13.217
Accuracy: 77,749 % Error: 22,251 %
Cohen's kappa (κ) 0,587

Figure 4: Estadísticas de la matriz confusión

Para este algoritmo se han debido categorizar y normalizar las variables para el correcto funcionamiento del KNN. Para sustentar el análisis comparativo, es necesario mostrar la configuración usada a la hora de normalizar los datos, dejando la explicación de la categorización de variables para el apartado 5.

Por último, la configuración del nodo K Nearest Neighbor.

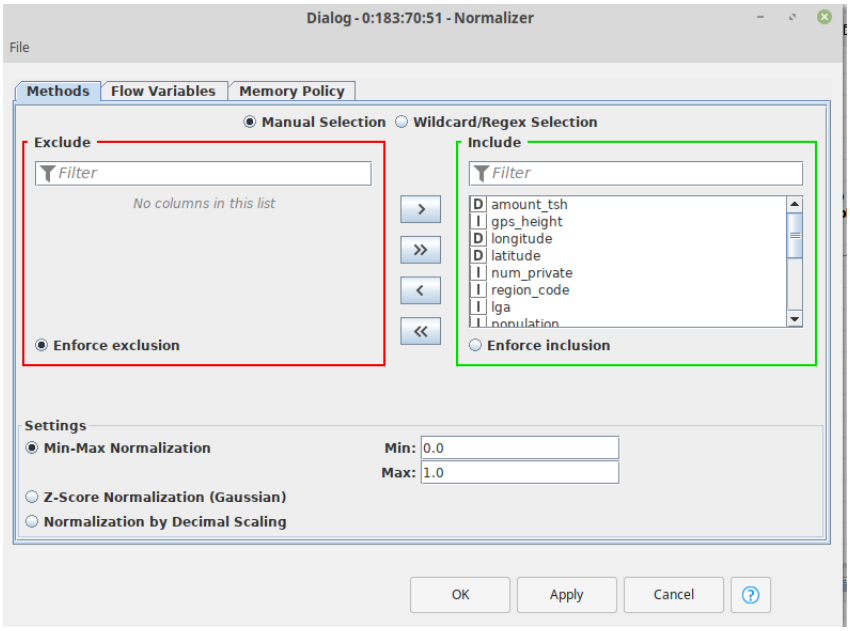

Figure 5: Normalización

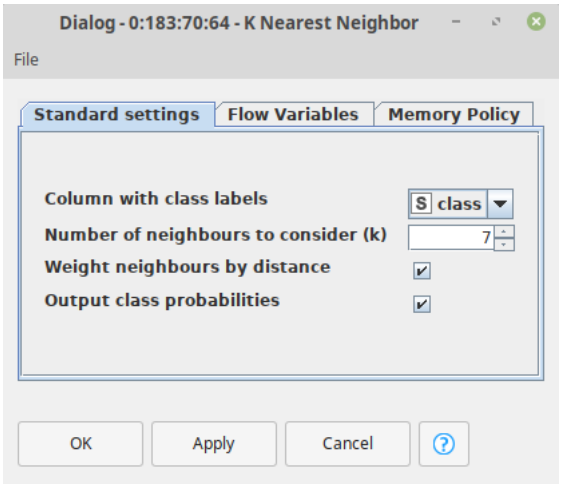

Figure 6: Configuración del clasificador KNN

2.2 Naive Bayes

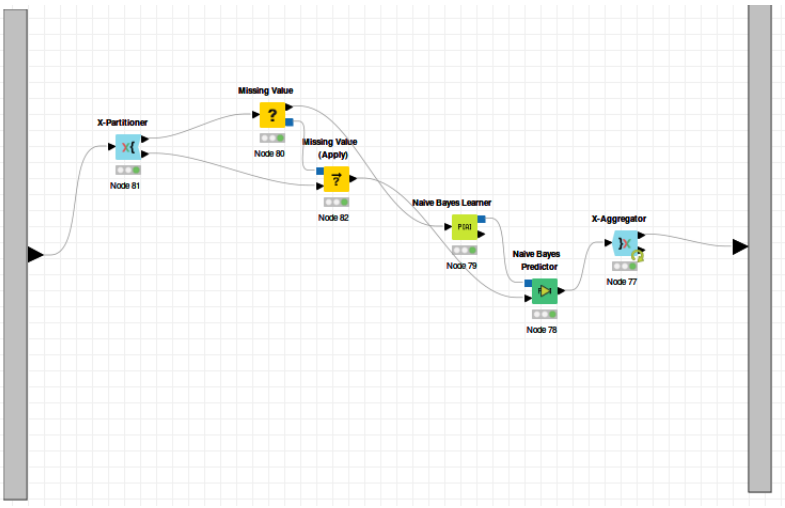

Figure 7: Visión general del nodo Naive Bayes

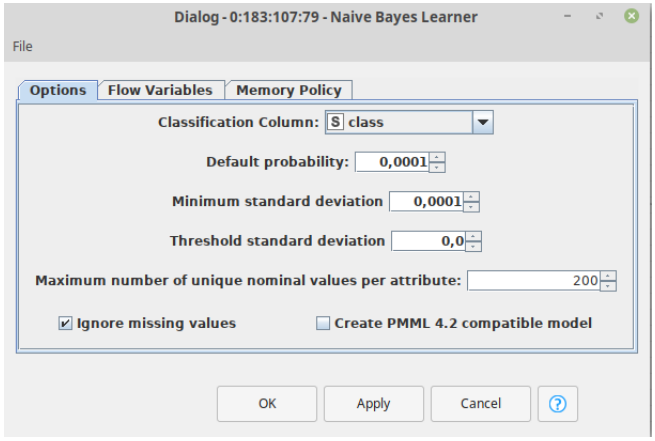

Figure 8: Configuración del clasificador Naive Bayes

row ID	TruePositives	FalsePositives	TrueNegatives	FalseNegatives	Recall	Precision	Sensitivity	Specifity	F-measure	Accuracy	Cohen's kappa
functional	18133	5476	21665	14126	0.5621066989057317	0.7680545554661358	0.5621066989057317	0.7982388268671015	0.6491372520942222		
non functional	16381	10836	25740	6443	0.7177094286715738	0.6018664805085057	0.7177094286715738	0.7037401574803149	0.6547031434223936		
functional needs repair	1972	6602	48481	2345	0.4567987028028724	0.2299976673645674	0.4567987028028724	0.8801445091952145	0.3059498875184237		
Overall										0.6142424242424243	0.3544869984345255

Table 2: Resultados obtenidos

row ID	functional	non functional	functional needs repair
functional	18133	9578	4548
non functional	4389	16381	2054
functional needs repair	1087	1258	1972

Table 3: Matriz de confusión

Figure 9: Estadísticas de la matriz de confusión

2.3 ZeroR

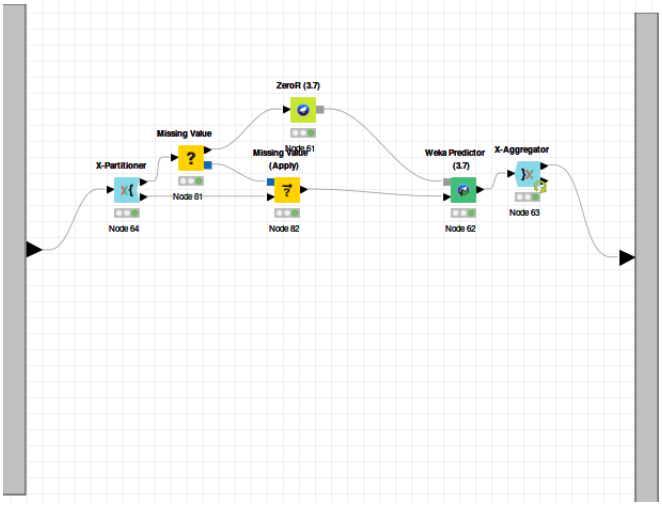

Figure 10: Visión general del nodo ZeroR

row ID	TruePositives	FalsePositives	TrueNegatives	FalseNegatives	Recall	Precision	Sensitivity	Specifity	F-measure	Accuracy	Cohen's kappa
functional	32259	27141	0	0	1	0.543080808080808	1	0	0.7038915982063955		
non functional	0	0	36576	22824	0		0	1			
functional needs repair	0	0	55083	4317	0		0	1			
Overall										0.543080808080808	0

Table 4: Estadísticas obtenidas

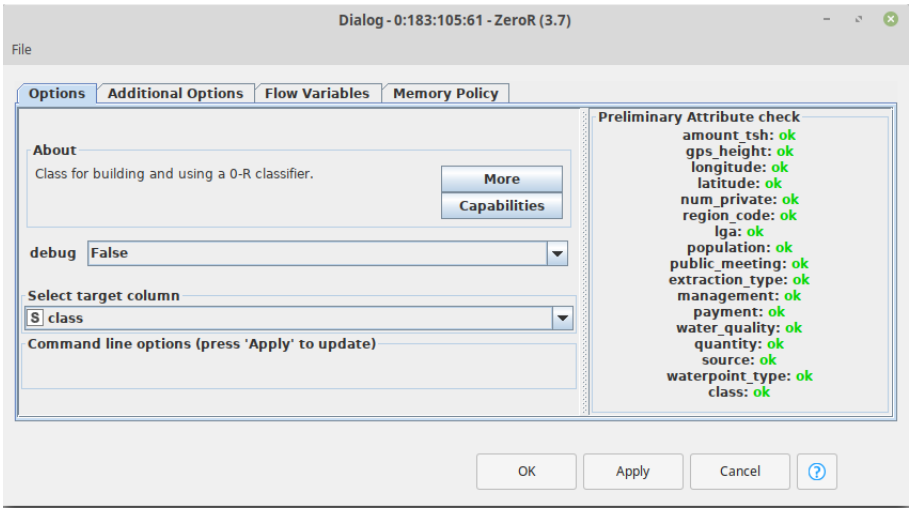

Figure 11: Configuración del clasificador Zero R

row ID	functional	non functional	functional needs repair
functional	32259	0	0
non functional	22824	0	0
functional needs repair	4317	0	0

Table 5: Matriz de confusión

Figure 12: Estadísticas de la matriz de confusión

2.4 XGBoosting

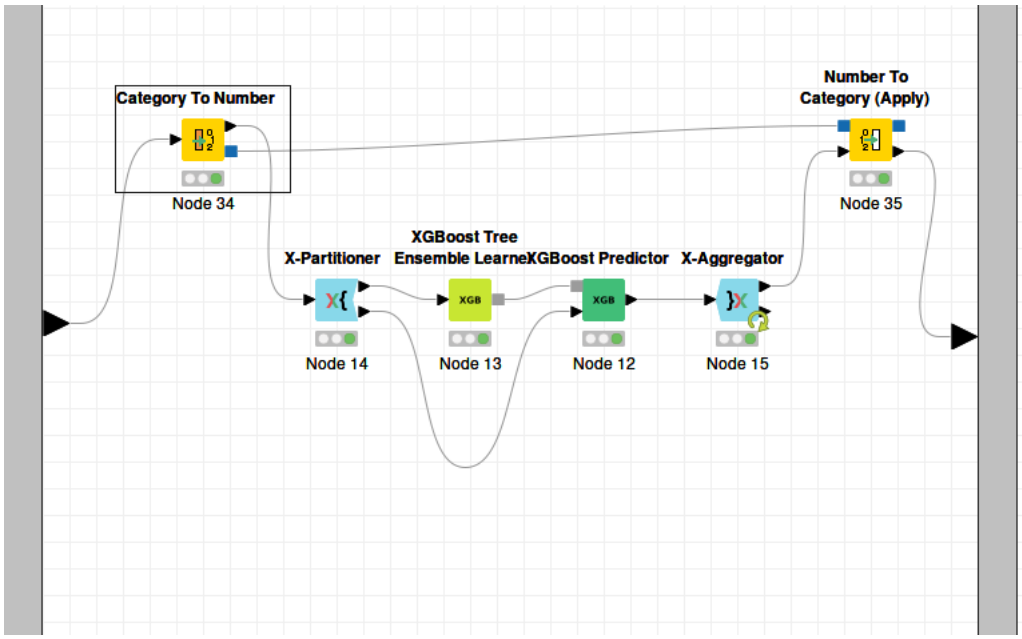

Figure 13: Visión general del flujo perteneciente al nodo XGBoost

row ID	TruePositives	FalsePositives	TrueNegatives	FalseNegatives	Recall	Precision	Sensitivity	Specifity	F-measure	Accuracy	Cohen's kappa
functional	29240	8243	18898	3019	0.906413714002294	0.7800869727609849	0.906413714002294	0.6962897461405254	0.8385191133033318		
non functional	16913	3234	33342	5911	0.7410182264283211	0.8394798232987541	0.7410182264283211	0.9115813648293963	0.7871820530124967		
functional needs repair	1120	650	54433	3197	0.2594394255269863 4	0.6327683615819211	0.2594394255269863 4	0.9881996260189169	0.3679973711473468		
Overall										0.7958417508417508	0.6109865840736038

Table 6: Estadísticas obtenidas

row ID	functional	non functional	functional needs repair
functional	29240	2588	431
non functional	5692	16913	219
functional needs repair	2551	646	1120

Table 7: Matriz de confusión

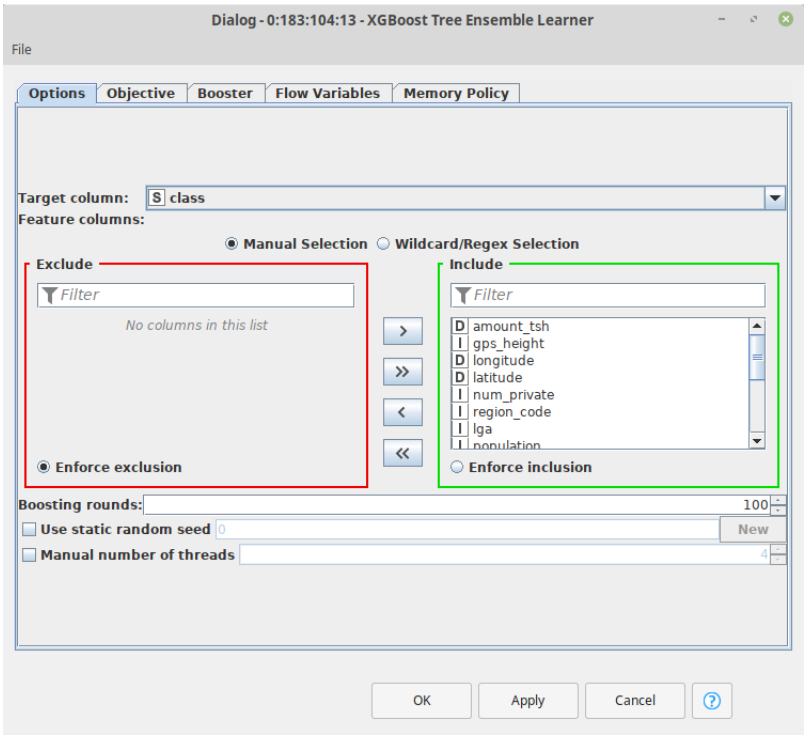

Figure 14: Configuración del clasificador XGBoost Tree Ensemble Learner

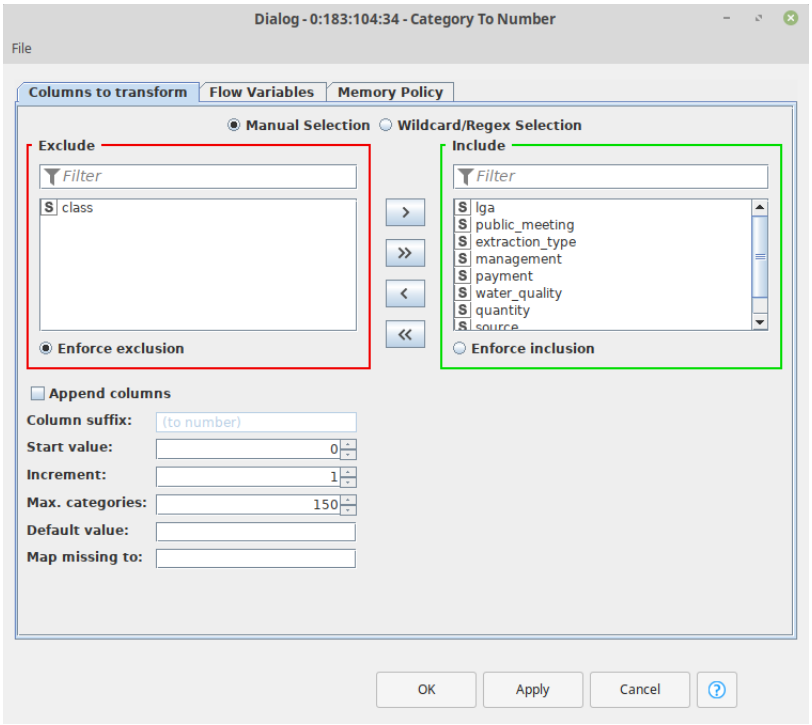

Figure 15: Transformación de variables categóricas a numéricas

Correct classified: 47.273	Wrong classified: 12.127
Accuracy: 79,584 %	Error: 20,416 %
Cohen's kappa (κ) 0,611	

Figure 16: Estadísticas de la matriz de confusión

2.5 Random Forest

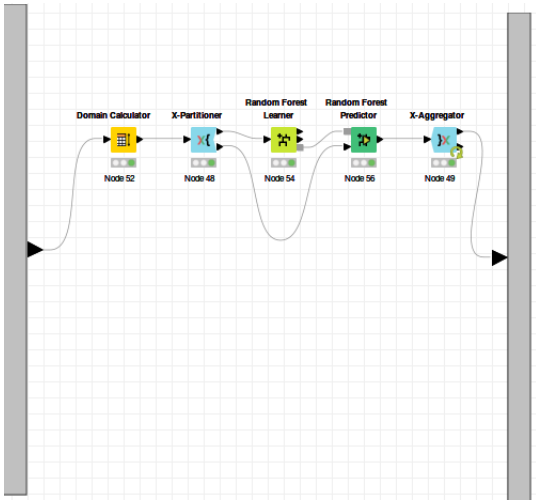

Figure 17: Visión general del flujo perteneciente al nodo Random Forest

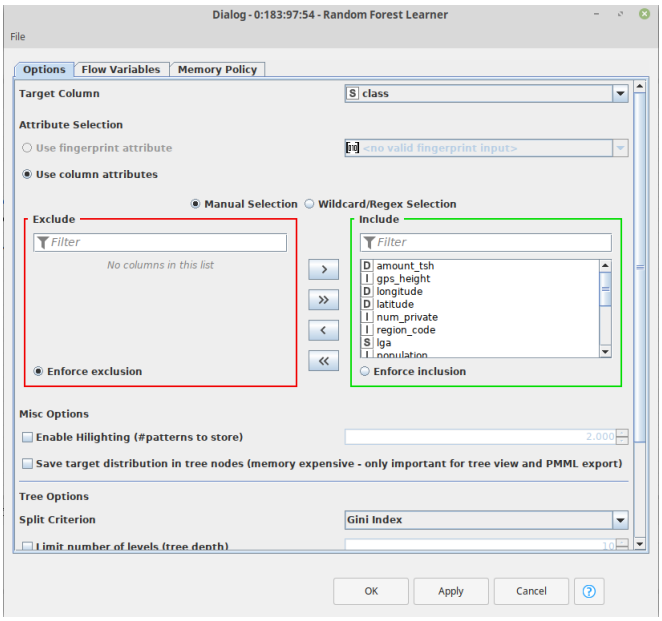

Figure 18: Configuración del clasificador Random Forest

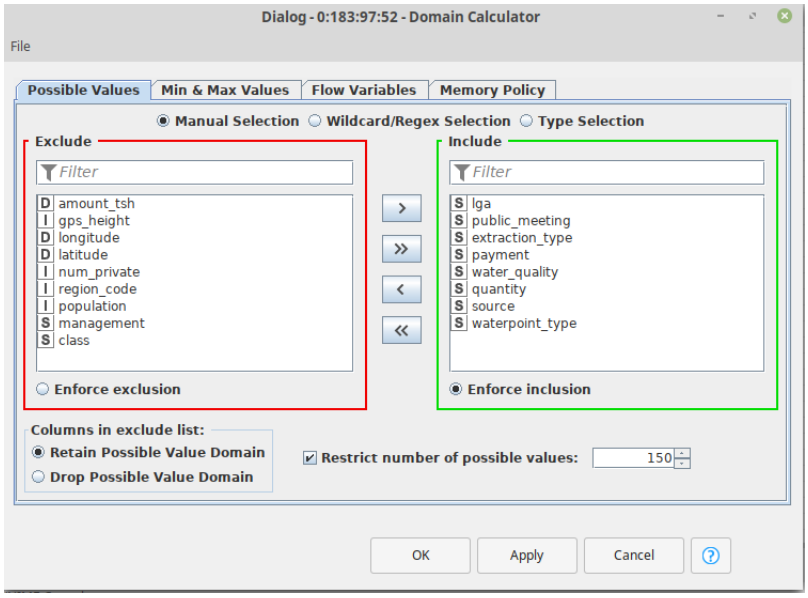

Figure 19: Calculador de dominio para determinadas variables

row ID	TruePositives	FalsePositives	TrueNegatives	FalseNegatives	Recall	Precision	Sensitivity	Specifity	F-measure	Accuracy	Cohen's kappa
functional	29417	7628	19513	2842	0.9119005548839084	0.7940882710217303	0.9119005548839084	0.7189491912604546	0.8489264688906846		
non functional	17394	2891	33685	5430	0.7620925341745531	0.8574808972146907	0.7620925341745531	0.9209590988626422	0.8069776612772275		
functional needs repair	1255	815	54268	3062	0.290711141996757	0.6062801932367 15	0.290711141996757	0.9852041464698728	0.3929857523093785		
Overall										0.8091919191919192	0.6383162999168186

Table 8: Estadísticas obtenidas

row ID	functional	non functional	functional needs repair
functional	29417	2296	546
non functional	5161	17394	269
functional needs repair	2467	595	1255

Figure 20: Estadísticas de la matriz de confusión

2.6 Multi Layer Perceptron

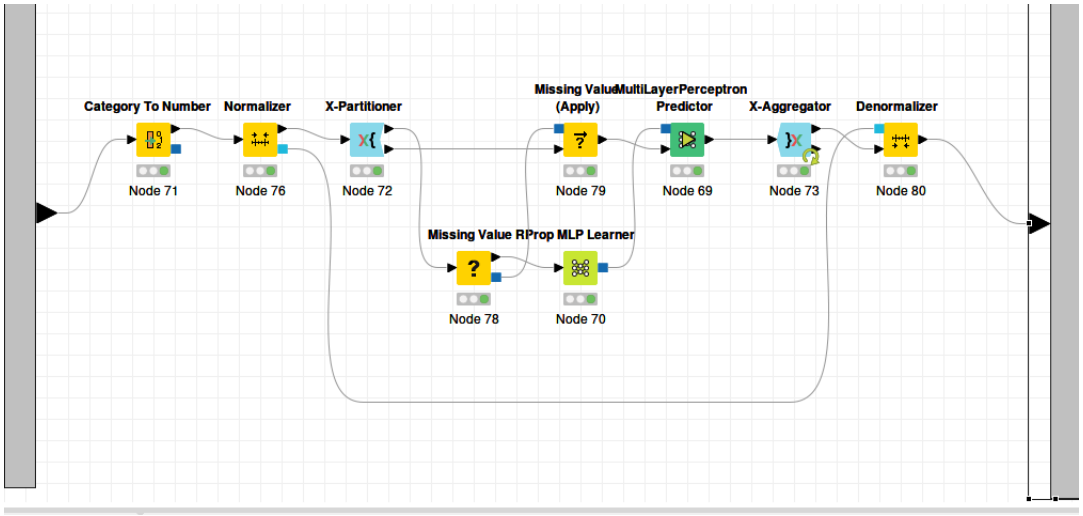

Figure 21: Visión general del flujo perteneciente al nodo Multi Layer Perceptron

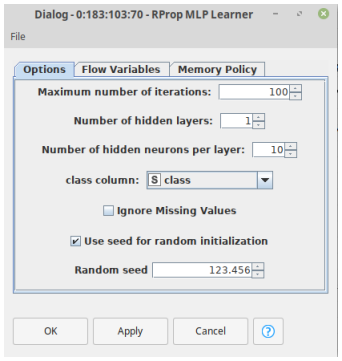

Figure 22: Configuración del clasificador MultiLayer Perceptron

row ID	TruePositives	FalsePositives	TrueNegatives	FalseNegatives	Recall	Precision	Sensitivity	Specifity	F-measure	Accuracy	Cohen's kappa
functional	29417	7628	19513	2842	0.9119005548839084	0.7940882710217303	0.9119005548839084	0.7189491912604546	0.8489264688906846		
non functional	17394	2891	33685	5430	0.7620925341745531	0.8574808972146907	0.7620925341745531	0.9209590988626422	0.8069776612772275		
functional needs repair	1255	815	54268	3062	0.290711141996757	0.606280193236715	0.290711141996757	0.9852041464698728	0.3929857523093785		
Overall										0.8091919191919192	0.6383162999168186

Table 9: Estadísticas obtenidas

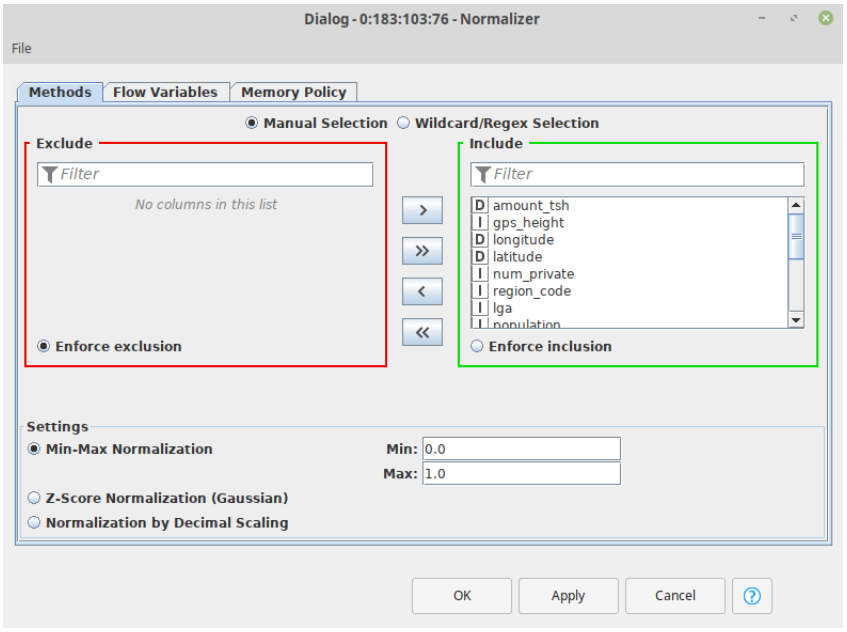

Figure 23: Normalización de variables numéricas

row ID	functional	non functional	functional needs repair
functional	29417	2296	546
non functional	5161	17394	269
functional needs repair	2467	595	1255

Table 10: Matriz de confusión

Figure 24: Estadísticas de la matriz de confusión

3 Análisis de resultados

row ID	TP	FP	TN	FN	PPV	TPR	TNR	F1-score	Accuracy	G-mean
KNN	27489	7197	19944	4770	0.7925099463760595	0.852134288105645	0.734829225157508	0.8212413174994398	0.7985353535353535	0.7913110505097336
Naive Bayes	18133	5476	21665	14126	0.7680545554661358	0.5621066989057317	0.7982388268671015	0.649137252094222	0.67	0.6698472899912713
ZeroR	32259	27141	0	0	0.543080808080808	1	0	0.7038915982063955	0.543080808080808	0
XGBoost	29240	8243	18898	3019	0.7800869727609849	0.906413714002294	0.6962897461405254	0.838519113303318	0.81040404040404403	0.7944347517706838
Random Forest	29417	7628	19513	2842	0.7940882710217303	0.9119005548839084	0.7189491912604546	0.8489264688906846	0.8237373737373738	0.8096975771507198
Red Neuronal	26719	13106	14035	5540	0.6709102322661644	0.8282649803155708	0.5171143288751335	0.7413295599578269	0.6860942760942761	0.6544522056091354

Table 11: Tabla resumen. Medidas estadísticas obtenidas durante las ejecuciones

No sorprende ver que algoritmos tales como el clasificador ZeroR o Naive Bayes consiguen una puntuación menor que el resto debido a su mejor complejidad de cálculo.

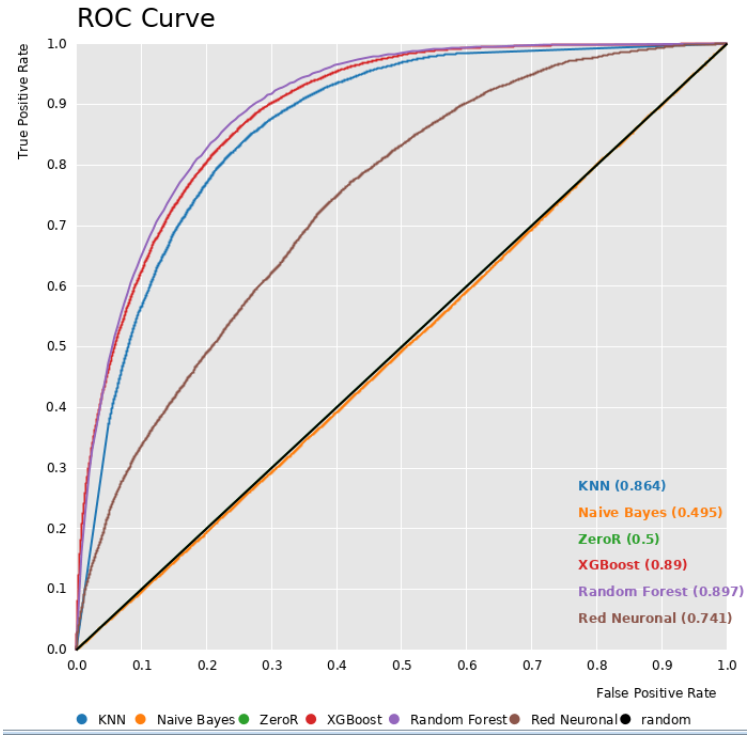

Figure 25: Curva ROC correspondiente a todos los algoritmos

Como era de esperar, aquellos algoritmos que consiguieron un valor de accuracy mayor también consiguen un valor de **Area Under Curve** mayor.

row ID	AUC
KNN	0.8659779108659242
Naive Bayes	0.49475904751468575
ZeroR	0.499972862509105
XGBoost	0.8952119562476301
Random Forest	0.9003469571612629
Red Neuronal	0.7543145758002797

Table 12: Area Under Curve

4 Configuración de algoritmos

Para este apartado, se ha considerado el estudio de los algoritmos Random Forest y MultiLayer Perceptron con el objetivo de saber si somos capaces de mitigar la aparición del sobreaprendizaje, el cual es el efecto de sobreentrenar un algoritmo de aprendizaje con unos ciertos datos para los que se conoce el resultado deseado. [2]

Para estudiar el sobreaprendizaje nos valemos de la métrica **G-Mean**. Esta medida es importante para evitar el sobreaprendizaje (overfitting) de la clase negativa y el underfitting de la clase positiva.

La media geométrica (G-Mean) es una métrica que mide el equilibrio entre los resultados de la clasificación tanto en las clases mayoritarias como en las minoritarias. Un nivel bajo de G-Mean es indicativo de un rendimiento pobre en la clasificación de los casos positivos, incluso si los casos negativos se clasifican correctamente como tales. De forma similar a F1-score, penaliza más los errores al clasificar ejemplos positivos que los errores al clasificar ejemplos negativos. [6]

	G-Mean(tra)	G-Mean (tst)	Overfitting
RF-InformationGainRatio-Limit=5	0.5724370478692741	0.570001822675882	1.0042723112392167
RF-GiniIndex-SaveTargetDistributions-Softvoting	0.966487368544734	0.7499016179895633	1.2888188868505481
RF-InformationGainRatioWithoutLimit	0.9556019037163072	0.7559102670934424	1.2641737324070235
RF-GiniIndex	0.9580673170645135	0.753905534332331	1.2708055232848119

Table 13: Random Forest. Sobreaprendizaje en diferentes casos

Tras los resultados, vemos como el criterio de partición *Gini Index* provoca un sobreaprendizaje mayor que si aplicamos el criterio *Information Gain Ration*. Si aplicamos Gini Index, además, es conveniente usarlo con su configuración por defecto.

En el caso de aplicar Information Gain Ratio, el sobreaprendizaje se hace mayor a medida que aumentamos el límite de número de niveles (profundidad del árbol).

	G-Mean(tra)	G-Mean (tst)	Overfitting
RN-50Iter-HiddenLayer=1-HiddenNeurons/layer=10	0.6404445906053976	0.6364762171111961	1.0062349124562942
RN-100Iter-HiddenLayer=1-HiddenNeurons/layer=10	0.6612659376143024	0.6559481748580404	1.0081069861310505
RN-200Iter-HiddenLayer=1-HiddenNeurons/layer=10	0.6726676009403153	0.6649522163626951	1.0116029158002113
RN-200Iter-HiddenLayer=5-HiddenNeurons/layer=10	0.6703426124683692	0.6480264858523048	1.0344370594462873

Table 14: Red Neuronal. Sobreaprendizaje en diferentes casos

Para evaluar el sobreaprendizaje que aparece en la Red Neuronal, el criterio seguido ha sido el de aumentar la robustez y fiabilidad del algoritmo con cada una de las pruebas. En este sentido, se han ido aumentando el número de iteraciones así como el número de capas ocultas y el número de neuronas por capa y tras los resultados, podemos concluir que a mayor número de iteraciones y en general, mayor grado de complejidad, el algoritmo sufre un mayor sobreaprendizaje.

5 Procesado de datos

En el dataset usado, existen variables con una cantidad de valores posibles que se escapan de los límites admisibles por algunos algoritmos. Éstos algoritmos, al ser sobrepasados, ignoran estos valores. Es por tanto conveniente la eliminación de estas columnas teniendo en cuenta que su manejo no es nada fácil.

Tanto el procesado general aplicado a todos los algoritmos como el procesamiento de datos específico para cada algoritmo se ve reflejado en los siguientes apartados.

Para cumplir las restricciones de cada algoritmo, se ha realizado un procesado personalizado de manera individual para cada uno de ellos (imputación de valores perdidos, numerización de variables categóricas, etc...), como se describe a continuación.

5.1 Procesado general

Cada algoritmo tiene unas necesidades diferentes, y por este motivo, el procesado general que se aplica a todos los algoritmos se reduce a un estudio de la correlación lineal entre variables con sus correspondientes relaciones y supresiones tras la aplicacion de un umbral de correlación igual a 0,8 y a un posterior borrado tras el análisis de sus características a través del nodo *Data Explorer*, mediante el cual se han suprimido columnas tales como *recordedby* al tener solo un único valor nominal, o columnas como *funder* ó *installer* que cuentan con más de 999 valores nominales diferentes.

5.2 KNN

Bajo el enclave del algoritmo de los *Vecinos más cercanos*, al tratarse de un problema de distancias se hace necesaria una normalización del valor de las

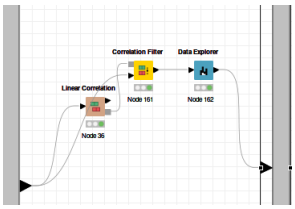

Figure 26: Procesado de datos general

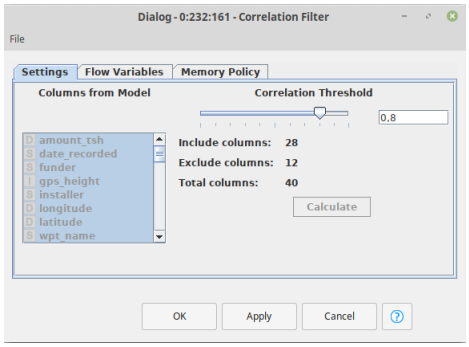

Figure 27: Filtro de correlación

variables así como que todas las variables sean de tipo numérico. Esto se consigue a través del uso del nodo *Category to Number*, con una configuración como la que sigue:

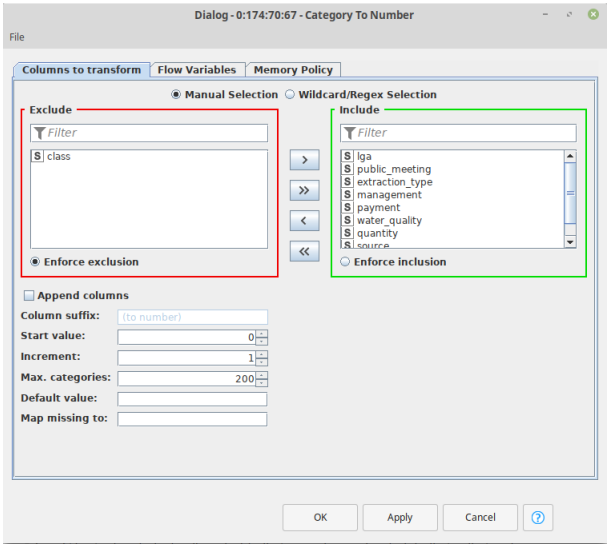

Figure 28: Conversión de variables numéricas a variables categóricas

5.3 Naive Bayes

Naive Bayes es un algoritmo en el que es adecuado que no haya valores perdidos pues en el caso de haber algún valor perdido en la fila que intenta predecir, se utilizan aquellas filas que sí que contienen dicho valor perdido para su determinada clase, pero en nuestro problema las clases están desequilibradas y por tanto la predicción para una fila que pertenezca a la clase minoritaria no sería de buena calidad. En la sección 5.8 puede verse como el accuracy mejora tras la adición del balanceo de clases.

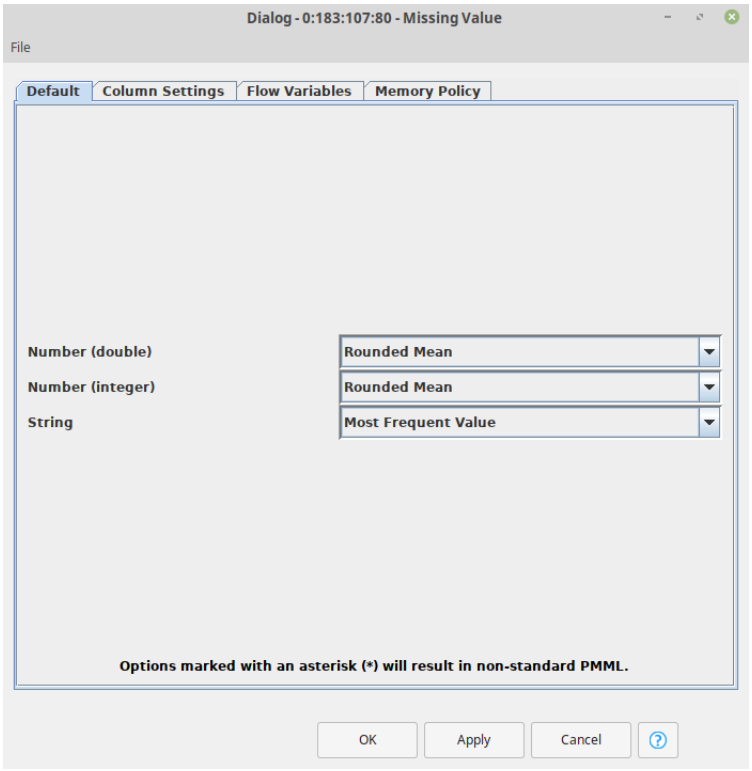

Figure 29: Imputación de valores perdidos

5.4 Zero R

A pesar de no ser necesario para este algoritmo, se ha añadido el uso de la imputación de valores perdidos para intentar mejorar los resultados. La configuración del nodo *Missing Values* ha sido la siguiente:

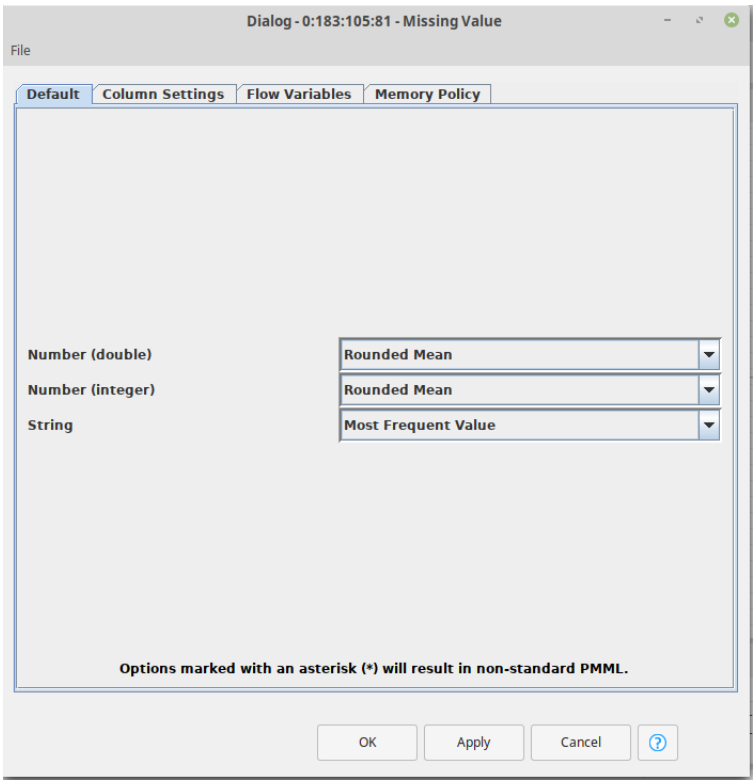

Figure 30: Imputación de valores perdidos

5.5 XGBoost

Los modelos XGBoost representan todos los problemas como un problema de modelado predictivo de regresión que sólo toma valores numéricos como entrada [1] y por tanto se hace necesario la transformación de aquellas variables que sean categóricas a numéricas.

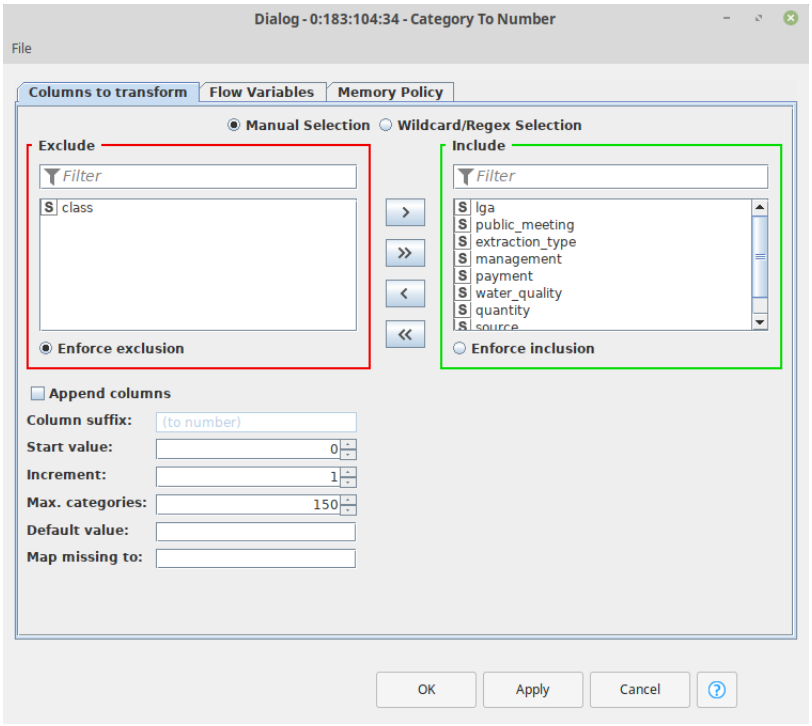

Figure 31: Transformación de variables categóricas a numéricas

5.6 Random Forest

Se realiza un cálculo del dominio de determinadas variblaes, bajo el cual se escanean los datos y se actualiza la lista de valores posibles y/o los valores mínimos y máximos de las columnas seleccionadas.

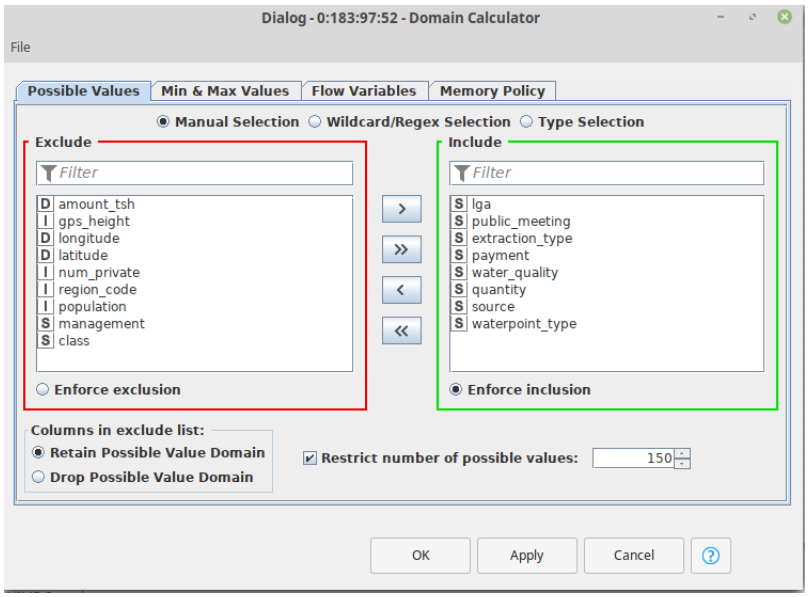

Figure 32: Calculador de dominio

5.7 Red Neuronal

Para el correcto funcionamiento de una red neuronal, es necesario un procesado previo que en este caso consiste en numerizar aquellas variables que son de tipo categórico así como una imputación de valores perdidos.

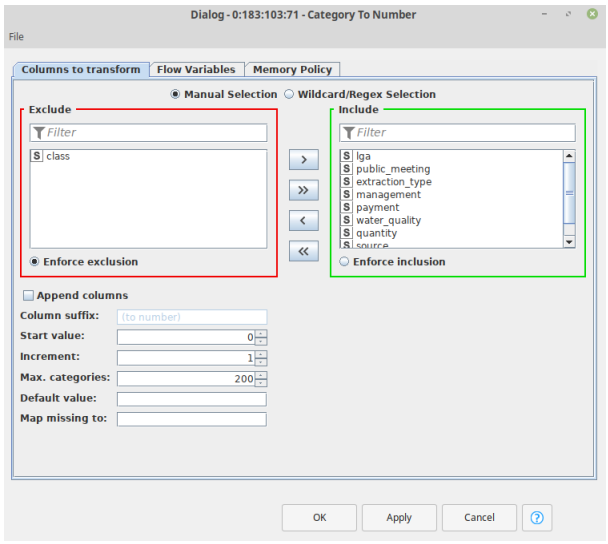

Figure 33: Transformación de variables categóricas a numéricas

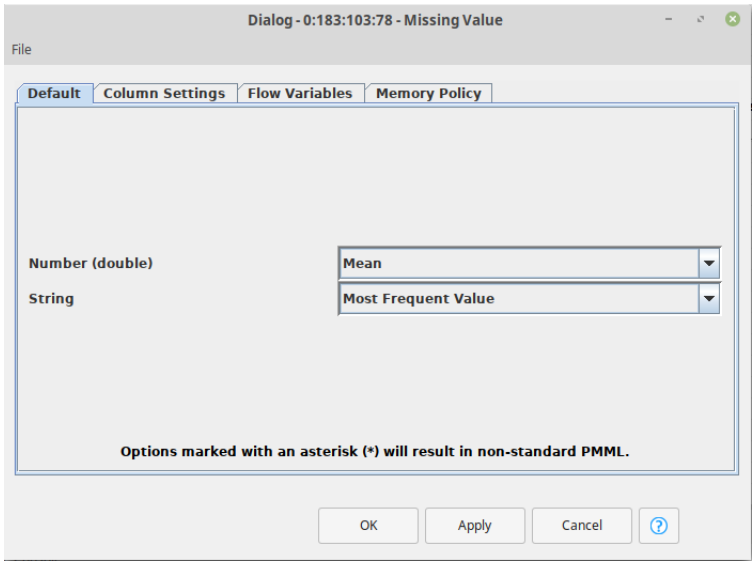

Figure 34: Imputación de valores perdidos

5.8 Balanceo de clases

Como se ha nombrado con anterioridad, sabemos que en el conjunto de datos usado existe un desequilibrio notable entre clases y una consecuencia de esto es que algunos algoritmos tienden a favorecer la clase con la mayor proporción de observaciones (conocida como clase mayoritaria), lo cual puede derivar en métricas de exactitud sesgadas. Esto puede ser particularmente problemático cuando estamos interesados en la clasificación correcta de una clase "rara" (también conocida como clase minoritaria) pero encontramos altos valores de exactitud que son producto realmente de la clasificación correcta de la clase mayoritaria y por tanto son un reflejo de la distribución de clases subyacente [3].

Para hacer más notable la necesidad de este balanceo de clases, a continuación se muestran los datos con este balanceo y sin él.

row ID	TP	FP	TN	FN	PPV	TPR	TNR	F1-score	Accuracy	G-mean
KNN	27489	7197	19944	4770	0.7925099463760595	0.852134288105645	0.7348292251575108	0.8212413174994398	0.7985353535353535	0.7913110505097336
Naive Bayes	18133	5476	21665	14126	0.7680545554661358	0.5621066989057317	0.7982388268671015	0.649137252094222	0.67	0.6698472899912713
ZeroR	32259	27141	0	0	0.543080808080808	1	0	0.7038915982063955	0.543080808080808	0
XGBoost	29240	8243	18898	3019	0.7800869727609849	0.906413714002294	0.6962897461405254	0.838519113303318	0.8104040404040403	0.7944347517706838
Random Forest	29417	7628	19513	2842	0.7940882710217303	0.9119005548839084	0.7189491912604546	0.8489264688906846	0.8237373737373738	0.8096975771507198
Red Neuronal	26719	13106	14035	5540	0.6709102322661644	0.828264980315708	0.5171143288751335	0.7413295599578269	0.6860942760942761	0.6544522056091354

Table 15: Medidas estadísticas sin balanceo de clases

row ID	TP	FP	TN	FN	PPV	TPR	TNR	F1-score	Accuracy	G-mean
KNN	2597	1655	6979	1720	0.6107714016933208	0.6015751679406995	0.8083159601575168	0.6061384058816665	0.7394023627519111	0.6973254688313815
Naive Bayes	953	382	8252	3364	0.7138576779026217	0.2207551540421589	0.95575631225388	0.3372257607926397	0.7107559261833063	0.4593344445372794
ZeroR	2589	5181	3453	1728	0.3332046332046332	0.5997220291869354	0.39993050729673385	0.42839414246711344	0.46652768126013433	0.4897419068956196
XGBoost	2874	1445	7189	1443	0.6654318129196574	0.6657400972897846	0.8326384063006718	0.665585919407133	0.777005636630376	0.7445272148268459
Random Forest	2944	1547	7087	1373	0.6555332887998219	0.6819550613852212	0.8208246467454251	0.6684831970935512	0.7745347849586904	0.7481747939871931
Red Neuronal	2115	1858	6776	2202	0.5323433173923987	0.4899235580264072 5	0.7848042622191337	0.5102533172496984	0.6865106941548915	0.6200758796314266

Table 16: Medidas estadísticas con balanceo de clases

Al estar ante clases desbalanceadas, es primordial prestar atención a medidas como el **f1-score** o el **G-mean** pues aportarán más claridad que medidas como por ejemplo accuracy.

La estadística F1-score transmite el equilibrio entre las medidas de precisión y de recall. En este caso al ser clases desbalanceadas buscamos un equilibrio entre ambas estadísticas pues la primera de ellas nos muestra que proporción de predicciones positivas fue correcta y la segunda nos indica qué proporción de predicciones positivas reales se identificó correctamente, por tanto nos interesa aquel algoritmo que nos proporcione un valor de f1-score mayor, que como vemos en este caso se consigue con Random Forest [5].

Si nos basamos en todo lo dicho en el apartado 4 acerca de la medida G-Mean y tras el análisis realizado, podemos en consecuencia asegurar que el mejor algoritmo sería de nuevo Random Forest sin aplicar el balanceo de clases.

6 Interpretación de resultados

7 Contenido adicional

References

[1] Data Preparation for Gradient Boosting with XGBoost
https://machinelearningmastery.com/data-preparation-gradient-boosting-xgboost-python/

[2] Sobreajuste
https://es.wikipedia.org/wiki/Sobreajuste

[3] Porqué es importante trabajar con datos balanceados
http://amsantac.co/blog/es/2016/09/20/balanced-image-classification-r-es.html

[4] Missing Values (Naive Bayes algorithms)
https://www.ibm.com/support/knowledgecenter/SSLVMB$_2$2.0.0/*com.ibm.spss.statistics.algorithms/alg*$_r$

[5] Classification Accuracy is Not Enough
https://machinelearningmastery.com/classification-accuracy-is-not-enough-more-performance-

[6] Predictive Accuracy: A Misleading Performance Measure for Highly Imbalanced Data
https://support.sas.com/resources/papers/proceedings17/0942-2017.pdf