Sesión 3 - Integración de KNIME y WEKA

SISTEMAS DE AYUDA A LA TOMA DE DECISIONES

CURSO 2023/2024

Óscar Brizuela García (820773@unizar.es)

David Arruga Escuer (816058@unizar.es)

Entrega: 7/11/2023

Índice

Ejercicio 1

Para la realización de este primer ejercicio, se procedió inicialmente a leer el fichero “yellow-small.data” mediante el nodo *File Reader (Complex Format)*. Además, este nodo también permitió renombrar cada una de las columnas según lo especificado en el guión de la práctica. Así, las columnas ahora están etiquetadas como “Color”, “Size”, “Act”, “Age” y “Inflated(True/False)”.

Posteriormente, se añadió una nueva columna a los datos anteriores utilizando el nodo *Rule Engine*. Este nodo permite aplicar reglas definidas por el usuario a cada fila de la tabla de entrada. Si una regla coincide, el resultado se agrega en una nueva fila de la columna creada por el nodo. De esta manera se han establecido reglas basadas en los valores de las columnas “Color” y “Size” para crear la nueva columna “class”. Estas reglas son las siguientes:

```
$Color$ = "YELLOW" AND $Size$ = "SMALL" => "inflated"

TRUE => "not inflated"
```

Continuando con el ejercicio, se ha procedido a crear una nueva columna denominada “full sentence” que completa la columna anterior. Se ha utilizado el mismo nodo “Rule Engine”, ajustando las reglas para comparar las columnas “class” e “Inflated(True/False)”. En este caso, siguiendo las instrucciones del guión, las reglas utilizadas son las siguientes:

```
$class$ = "inflated" AND $Inflated(True/False)$ = "T" => "inflated is T"

$class$ = "not inflated" AND $Inflated(True/False)$ = "F" => "not inflated is F"
```

El workflow realizado para este ejercicio se muestra en la figura 1 y en la figura 2 se muestra el archivo excel que genera este workflow con las dos columnas anteriores creadas. Como se puede observar en la figura 2, la columna denominada "full sentence" no incluye comillas, ya que se interpreta que su contenido es una cadena de texto.

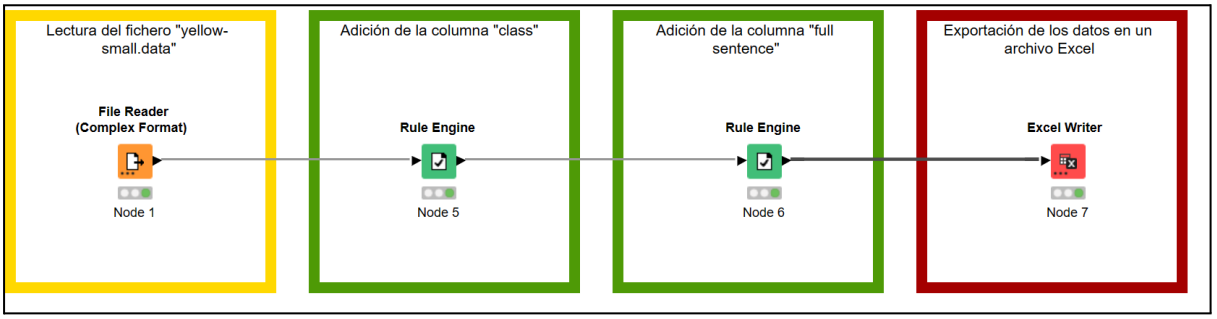

Figura 1: Workflow del ejercicio 1

	A	B	C	D	E	F	G
1	Color	Size	Act	Age	Inflated(True/False)	class	full sentence
2	YELLOW	SMALL	STRETCH	ADULT	T	inflated	inflated is T
3	YELLOW	SMALL	STRETCH	CHILD	T	inflated	inflated is T
4	YELLOW	SMALL	DIP	ADULT	T	inflated	inflated is T
5	YELLOW	SMALL	DIP	CHILD	T	inflated	inflated is T
6	YELLOW	SMALL	STRETCH	ADULT	T	inflated	inflated is T
7	YELLOW	SMALL	STRETCH	CHILD	T	inflated	inflated is T
8	YELLOW	SMALL	DIP	ADULT	T	inflated	inflated is T
9	YELLOW	SMALL	DIP	CHILD	T	inflated	inflated is T
10	YELLOW	LARGE	STRETCH	ADULT	F	not inflated	not inflated is F
11	YELLOW	LARGE	STRETCH	CHILD	F	not inflated	not inflated is F
12	YELLOW	LARGE	DIP	ADULT	F	not inflated	not inflated is F
13	YELLOW	LARGE	DIP	CHILD	F	not inflated	not inflated is F
14	PURPLE	SMALL	STRETCH	ADULT	F	not inflated	not inflated is F
15	PURPLE	SMALL	STRETCH	CHILD	F	not inflated	not inflated is F
16	PURPLE	SMALL	DIP	ADULT	F	not inflated	not inflated is F
17	PURPLE	SMALL	DIP	CHILD	F	not inflated	not inflated is F
18	PURPLE	LARGE	STRETCH	ADULT	F	not inflated	not inflated is F
19	PURPLE	LARGE	STRETCH	CHILD	F	not inflated	not inflated is F
20	PURPLE	LARGE	DIP	ADULT	F	not inflated	not inflated is F
21	PURPLE	LARGE	DIP	CHILD	F	not inflated	not inflated is F

Figura 2: Resultados del ejercicio 1

Ejercicio 2

En este segundo ejercicio se van a realizar diferentes tareas sobre un conjunto de datos que describe el número de visitantes a un sitio web.

Para la realización de las siguientes tareas, primero se ha estudiado el coeficiente de Kurtosis. Esta variable estadística/aleatoria es una característica de forma de su distribución de frecuencias/probabilidad. Este coeficiente se utiliza para evaluar qué tan “picuda” o “aplanada” es la distribución de los datos en comparación con una distribución normal o gaussiana. Una Kurtosis grande implica una mayor concentración de valores de la variable muy lejos del centro de la misma

Un coeficiente de Kurtosis es el cuarto momento con respecto a la media estandarizado que se define como la siguiente fórmula:

$$\beta_2 = \frac{\mu_4}{\sigma^4}$$

En la anterior fórmula, μ_4 es el cuarto momento centrado o con respecto a la media y σ es la desviación estándar. Si el coeficiente de Kurtosis es mayor que 3, la distribución es leptocúrtica, es decir, la distribución tiene colas más pesadas y picos más pronunciados que una distribución normal. En cambio, si este coeficiente es menor que 3, la distribución es platocúrtica, es decir, la distribución tiene colas más ligeras y es más aplanada que una

distribución normal. Por último, si el coeficiente de Kurtosis es igual a 3, la distribución tiene la misma forma que una distribución normal.

Una de las tareas pedidas sobre el conjunto de datos comentado es el entrenamiento de una red Bayesiana Naïve para descubrir cuándo los datos recogidos para una fila se corresponden a un día de diario o a fin de semana. Para ello, analizando los datos proporcionados mediante el nodo *File Reader (Complex Format)* se ha decidido prescindir de la columna "post" debido a que no aporta ninguna información adicional, utilizando el nodo *Column Filter*.

Para poder realizar el entrenamiento de la red correctamente, y conocer si un día es fin de semana o no de manera exacta, se ha utilizado el nodo *Rule Engine* reemplazando la columna "weekday" con las siguientes reglas:

```
$weekday$ = "Sat" OR $weekday$ = "Sun" => "weekend"

TRUE => "working day"
```

Posteriormente, se ha utilizado el nodo *Partitioning* para entrenar la red con el 80% de los datos,A continuación, se utilizaron los nodos de aprendizaje de Naive Bayes, *Naive Bayes Learner,* y de predicción de Naive Bayes *Naive Bayes Predictor*, para llevar a cabo el entrenamiento de la red Bayesiana Naïve requerida. Es relevante señalar que se ha mantenido la configuración por defecto en estos dos nodos ya que su influencia se limita a la velocidad de obtención de resultados y no afecta los resultados finales. Por tanto, la única configuración que ha sido modificada ha sido el incremento de los valores nominales diferentes posibles a 87. Por último, se ha utilizado el nodo *Scorer* para la visualización de los resultados. El workflow para la realización de estas tareas se puede observar en la figura 3.

Tras realizar diferentes ejecuciones, se concluyó que el clasificador tiene una buena precisión y sensibilidad, pero la especificidad era un poco baja, lo que significa que hay una tendencia a clasificar algunos días de la semana como fin de semana. Un ejemplo de estas pruebas realizadas se pueden observar en la figura 4.

Analizando las diferentes variables que existen en el fichero proporcionado, se llegó a la conclusión de que la columna "Fecha" carecía de relevancia para el proceso de entrenamiento, ya que no aportaba información valiosa. Después de eliminar esta columna utilizando el nodo *Column Filter* comentado anteriormente, se observaron mejoras en los resultados, particularmente en cuanto al *recall* y la puntuación *F-measure*. No obstante, la especificidad continuó siendo baja, como se ilustra en la figura 5. En resumen, la mejora en el *recall* y la *F-measure* sugiere que la eliminación de la columna "Fecha" puede haber ayudado al modelo a ser más preciso en la clasificación de los datos. Sin embargo, la baja especificidad persiste, lo que indica que el modelo todavía tiene dificultades para clasificar correctamente ciertos casos.

Figura 3: Workflow del ejercicio 2

Row ID	TruePo...	FalsePo...	TrueNe...	FalseN...	Recall	Precision	Sensitivity	Specificity	F-meas...	Accuracy	Cohen'...
working day	7	1	2	3	0.7	0.875	0.7	0.667	0.778	?	?
weekend	2	3	7	1	0.667	0.4	0.667	0.7	0.5	?	?
Overall	?	?	?	?	?	?	?	?	?	0.692	0.297

Figura 4: Ejemplo de una prueba realizada para el ejercicio 2

Row ID	TruePo...	FalsePo...	TrueNe...	FalseN...	Recall	Precision	Sensitivity	Specificity	F-meas...	Accuracy	Cohen'...
working day	8	3	2	0	1	0.727	1	0.4	0.842	?	?
weekend	2	0	8	3	0.4	1	0.4	1	0.571	?	?
Overall	?	?	?	?	?	?	?	?	?	0.769	0.451

Figura 5: Ejemplo de una prueba realizada para el ejercicio 2 eliminado la columna "Fecha"

Además, como se aprecia en la figura 3, se ha utilizado el nodo *ROC Curve (legacy)* para representar la curva ROC y evaluar el rendimiento del clasificador Bayesiano Naïve. La figura 5 muestra la curva ROC para la variable "working day". Es relevante señalar que la curva correspondiente a la variable "weekend" sería la simétrica.

La curva ROC, Receiver Operating Characteristic, es una representación gráfica que se utiliza en la evaluación de modelos de clasificación binaria. Esta curva muestra la relación entre la sensibilidad y la especificidad del modelo en diferentes umbrales de decisión.
En esta curva, el Área Bajo la Curva, AUC, es una métrica que cuantifica la capacidad de un modelo de clasificación para distinguir entre las clases positivas y negativas. En este caso, tiene un valor de 0,9091, esto significa que este modelo tiene un rendimiento bastante bueno en términos de su capacidad para distinguir entre día de la semana y fin de semana.
.

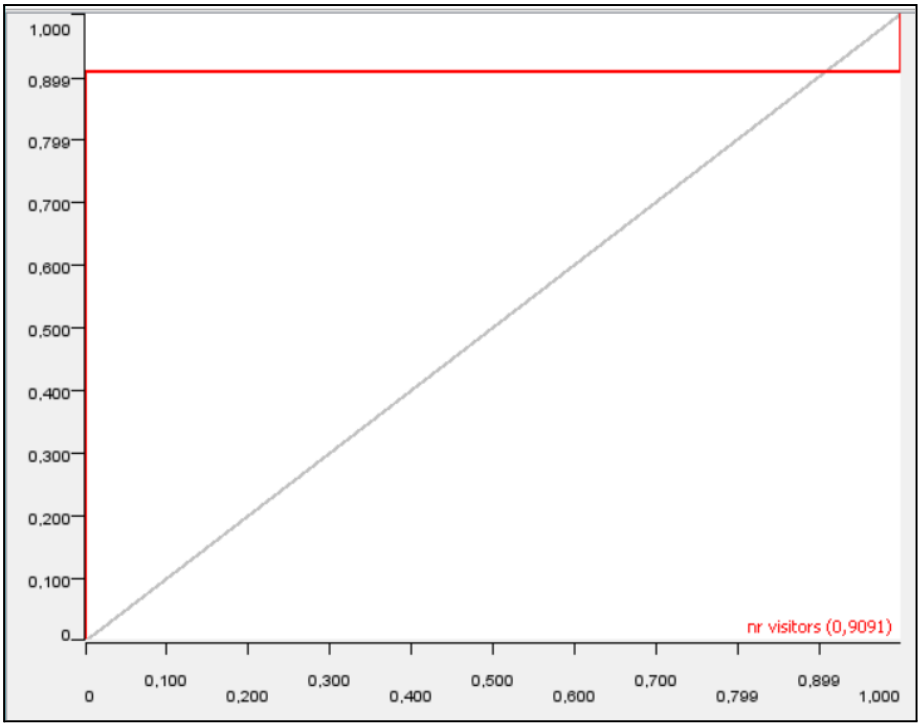

Figura 6: Curva ROC del ejercicio 2

Ejercicio 3

En este ejercicio se va a llevar a cabo una comparativa a gran escala de los modelos usados en ejercicios anteriores con los nodos de KNIME y Weka.

La arquitectura de esta batería de experimentos ha seguido un diseño modular basado en el modelo de aprendizaje automático utilizado para las predicciones de los datos, los conjuntos de datos y los tipos de nodos utilizados (KNIME ó WEKA).

Para este ejercicio se han utilizado 2 workflows: “ejercicio_3_knime”, para realizar el conjunto de experimentos con los nodos de Knime, y “ejercicio_3_weka”, donde el conjunto de experimentos se ha llevado a cabo con los nodos de Weka. Como ambos flujos tienen la misma estructura, se ha ilustrado la estructura de uno de ellos en la figura 7.

En ambos workflows, cada uno de los 3 metanodos contiene internamente la estructura de un modelo de aprendizaje automático para implementar un clasificador: una red Naive-Bayes, una red neuronal profunda o MLP (Multi-Layer Perceptron) y un árbol de decisión. Cabe destacar que la realización de estos entrenamientos se ha llevado a cabo utilizando los mismos nodos que en ejercicios anteriores.

Debido a que cada conjunto de datos es diferente entre sí y requiere de un preprocesamiento particular, dependiendo también del modelo de aprendizaje utilizado, se ha optado por realizar este preprocesado de los datos dentro del propio metanodo. No obstante, para ambos workflows se han reordenado las columnas del dataset “wine.data” inmediatamente después de leer los datos, con el objetivo de poner la clase a predecir como última columna, puesto que esta se encontraba como primera columna a diferencia de los otros 2 datasets.

Workflow *ejercicio_3_knime*

En cuanto al metanodo “Naive-Bayes” de KNIME, los valores de todas las *features* de los datasets “wine.data” e “iris.data” son de tipo *double* (salvo “Proline” y “Magnesium”, que son de tipo *integer*). Por tanto, no ha sido necesario cambiar el tipo de dato de ninguna columna para estos datasets. Sin embargo, en el caso del dataset “adult.data” se han eliminado las columnas “fnlwgt”, puesto que se trata de algún tipo de identificador (probablemente, indica el número de la persona del correspondiente registro) y “Education”, puesto que ya se contaba con otra columna, “Education-num”, que categoriza el nivel de educación de un individuo de manera numérica, por lo que la columna “Education” es redundante. Además, como se verá posteriormente, los valores numéricos suelen funcionar mejor que los nominales para la mayoría de modelos de aprendizaje automático.

A continuación, debido a que el dataset cuenta con dos columnas que, aparentemente, representan el capital ganado o perdido probablemente fuera del salario convencional (como puede ser, por ejemplo, en inversiones personales o apuestas), se ha decidido combinar ambas columnas en una sola. Esta columna contendrá el valor de la columna “Capital-gain” ó el valor, multiplicado por -1, de la columna “Capital-loss”. De esta forma, la nueva columna “Capital-balance” contendrá un balance real, ya sea positivo en caso de que el individuo haya generado beneficios, o negativo en caso de que haya generado pérdidas, más allá de su salario. Tras la creación de esta nueva columna “Capital-balance” mediante el uso del nodo “Math Formula”, se han eliminado las columnas originales “Capital-gain” y “Capital-loss”, ya que de no hacerlo ahora habría columnas redundantes.

Además, para ser consistentes, se ha reordenado las columnas con el nodo “Column Resorter” para poner la clase a predecir como última columna.

En el metanodo Decision Tree de KNIME no se realizó ningún preprocesado para los datos de “wine.data” e “iris.data”, pero se llevó a cabo el mismo preprocesado que en el nodo de Naive-Bayes para los datos provenientes del conjunto de datos “adults.data”.

En cuanto al metanodo MLP de KNIME, se implementa el aprendizaje automático mediante un Perceptrón Multicapa, o MLP por sus siglas en inglés. Este modelo es, al fin y al cabo, una red neuronal profunda que ha de admitir múltiples pesos correspondientes a los valores de cada una de las *features* de entrada. Por tanto, estos valores han de ser normalizados (distribuidos en un conjunto de valores entre 0 y 1) para poder ser utilizados como entrada en la red neuronal. Esta normalización se realiza, por tanto, para los datos de los 3 conjuntos. En el caso del dataset de “wine.data”, el preprocesado ha sido algo más complejo que en los metanodos de Naive-Bayes y Decision Tree. Además de realizar la eliminación de las columnas “fnlwgt”, “Education”, “Capital-gain” y “Capital-loss” por las razones explicadas anteriormente y la adición de la columna “Capital-balance”, han sido

necesarias otras transformaciones de los datos para poder utilizar el MLP con los datos disponibles correctamente. En primer lugar, ha sido necesario codificar todos los datos del “adults.data” como *doubles* puesto que todos, salvo la clase a predecir (columna “Class”), eran de tipo *string,* ya que eran valores nominales. Para ello, se ha utilizado el nodo “Category to Number”. Obviamente, es necesario hacer esto antes de normalizar los valores entre 0 y 1. Asimismo, se ha decidido eliminar aquellas filas (ejemplos utilizados para entrenar el MLP) que tuvieran algún valor nulo en alguna de sus columnas mediante el nodo “Missing Value”. Al principio se optó por simplemente marcar el *checkbox* “Ignore Missing Values” en la configuración del nodo “RProp MLP Learner” en vez de utilizar el nodo “Missing Value”. Sin embargo, el nodo “MultiLayer Perceptron Predictor”, que se encarga de predecir la clase, no admite valores nulos. Por tanto, debido a que los datos de validación, utilizados en este nodo, son una porción del total de los datos (por ejemplo, el 20%), finalmente se optó por hacer uso del nodo “Missing Values”. Por último, se realizó otro cambio de tipo de dato, esta vez de *double* a *string*, mediante el nodo “Number to String”, para poder obtener en los resultados calculados por el nodo “Scorer” las clases “<=50k” ó “>50k” en vez de las clases 0.0 y 1.0 provenientes de la codificación de los valores nominales realizada previamente. En la figura 8 se muestra el contenido de este metanodo. Para una mejor visualización, en esta figura solo se han puesto anotaciones en el preprocesado, el entrenamiento y la predicción correspondientes al dataset “wine.data” pues, al fin y al cabo, es el más complejo.

Workflow *ejercicio_3_knime*

En el metanodo Naive Bayes de WEKA, para el dataset “wine.data”, se ha utilizado de nuevo el nodo “Number to String” ya que el nodo “NaiveBayes (3.7)” de WEKA no admite clases numéricas y, en este dataset, las clases pueden ser 1, 2 ó 3. Por otro lado, el preprocesamiento de los datos de“adults.data” es exactamente el mismo que su metanodo homónimo de KNIME.

En el metanodo Decision Tree de WEKA, el nodo que implementa el aprendizaje es el “J48 (3.7)” de WEKA, que sigue el algoritmo de clasificación *J48* basado en árboles de decisión. Es relevante señalar que el preprocesamiento de los datos ha sido el mismo que en el metanodo Naive-Bayes de WEKA, para cada uno de los conjuntos de datos.

Finalmente, en el metanodo MLP de WEKA, el preprocesamiento es casi idéntico a su homónimo en KNIME, exceptuando que, como en el metanodo Naive-Baye de WEKA, la clase tiene que ser convertida de *double* a *string* en el dataset “wine.data”. La única diferencia es que se ha hecho uso del nodo “Rule Engine” para renombrar las clases de vuelta a “<=50k” ó “>50k” en vez de las clases 0.0 y 1.0 provenientes de la codificación de los valores nominales realizada previamente en el dataset “adults.data”. De esta manera, la columna perteneciente a la clase en la tabla obtenida por el nodo “Scorer” tiene el formato deseado.

Realización de los experimentos

Una vez realizados los dos workflows comentados anteriormente, se han realizado cinco ejecuciones de cada metanodo con tres proporciones distintas para el conjunto de entrenamiento y prueba: 30%, 50% y 80%.

Posteriormente, las ejecuciones sobre el mismo conjunto de datos se han exportado en el mismo archivo Excel, utilizando el nodo “Excel writer” y la opción “Append” para poder realizar un promedio de las mismas. Estos promedios se han juntado en otro fichero Excel para poder representar de manera gráfica los resultados obtenidos y poder comparar qué modelos proporcionan los mejores resultados en cada caso.

Observando los gráficos mostrados en las figuras 10,11,12,13,14,15,16,17 y 18, se puede concluir que, en líneas generales, Naïve Bayes proporciona los mejores resultados en comparación con Multi-Layer Perceptron y Decision Tree (J48).

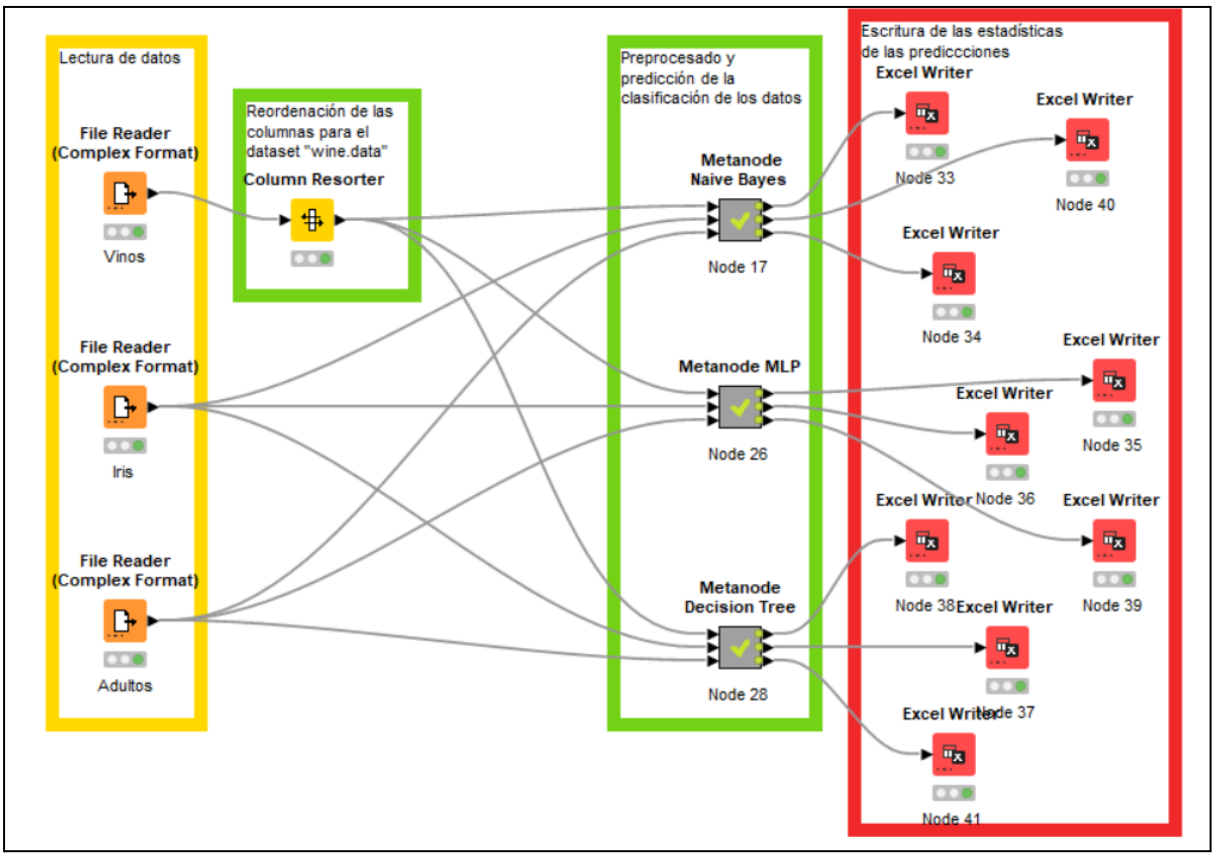

Figura 7: Workflow para los nodo de Knime del ejercicio 3

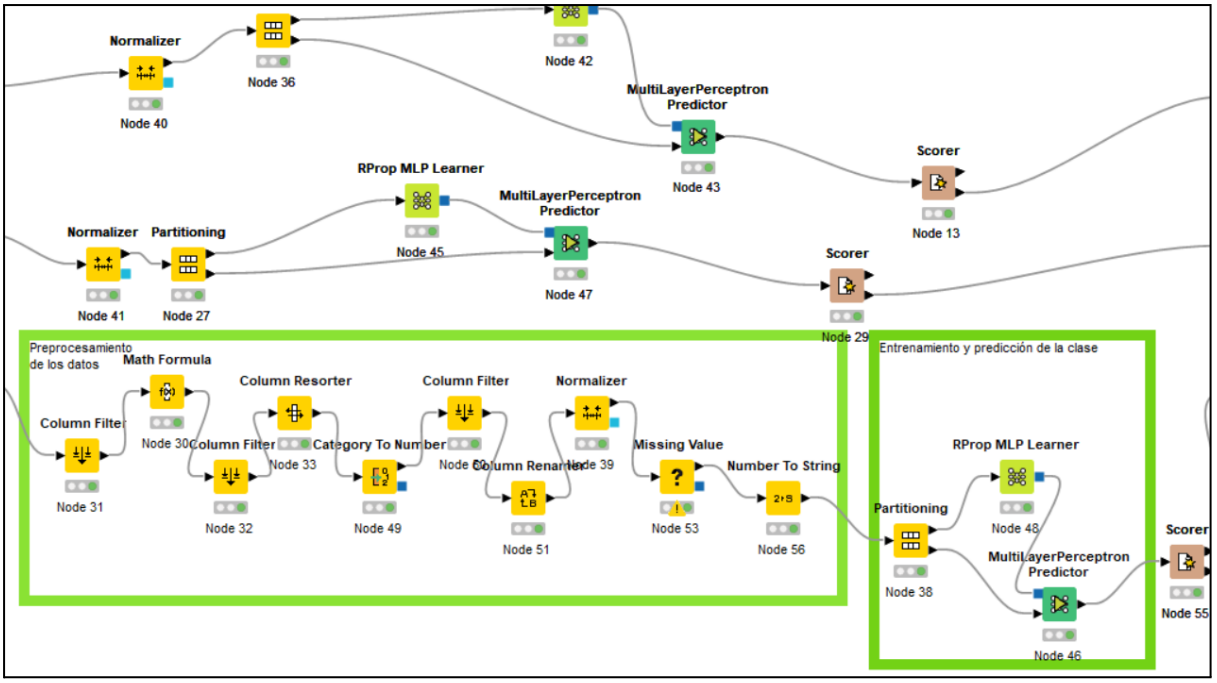

Figura 8: Contenido del metanodo MLP de Knime del ejercicio 3

Gráficos con los resultados de los experimentos

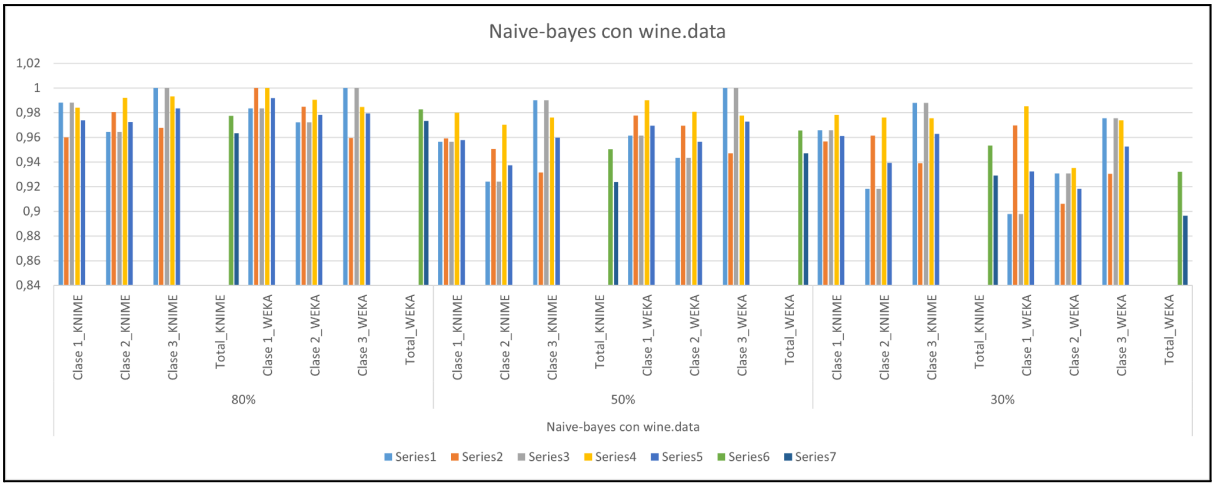

Figura 10: Gráfico de Naïve bayes con el fichero "wine.data"

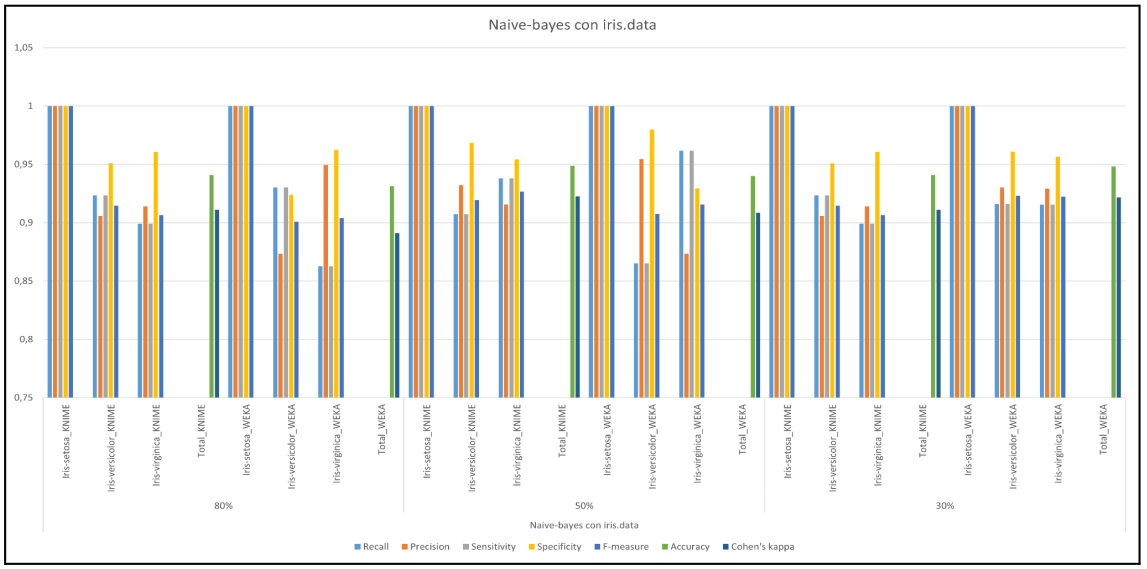

Figura 11: Gráfico de Naïve bayes con el fichero "iris.data"

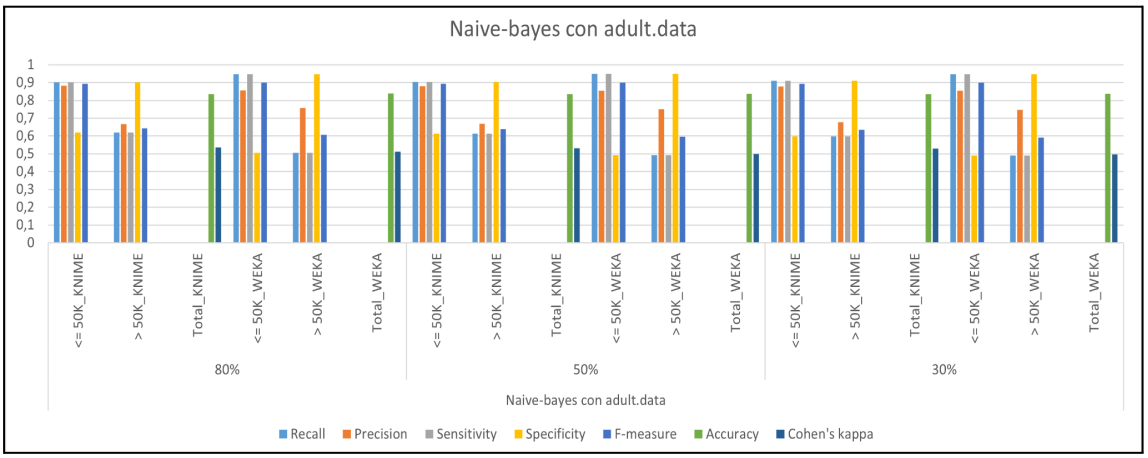

Figura 12: Gráfico de Naïve bayes con el fichero "adult.data"

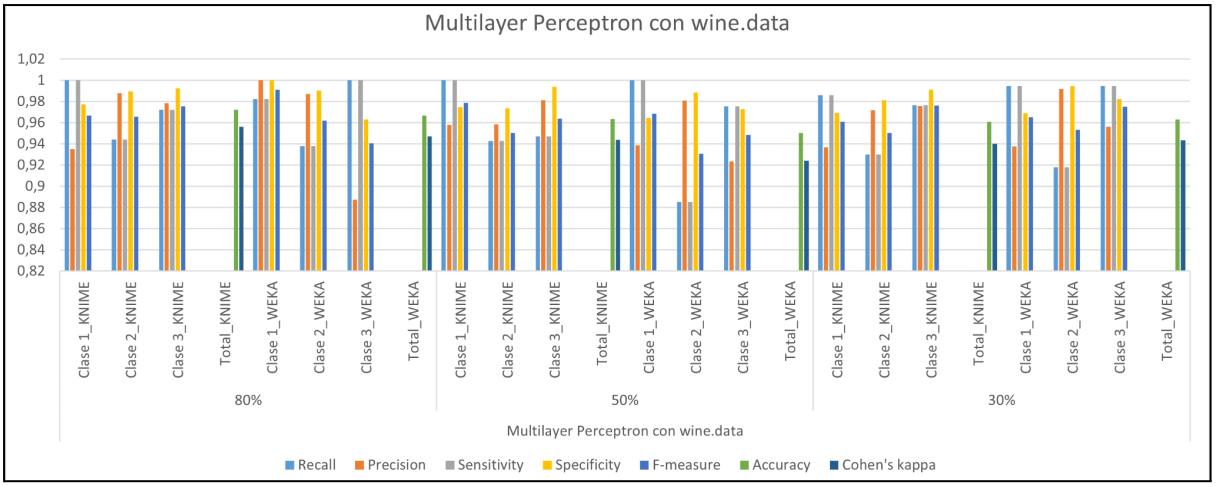

Figura 13: Gráfico de MLP con el fichero "wine.data"

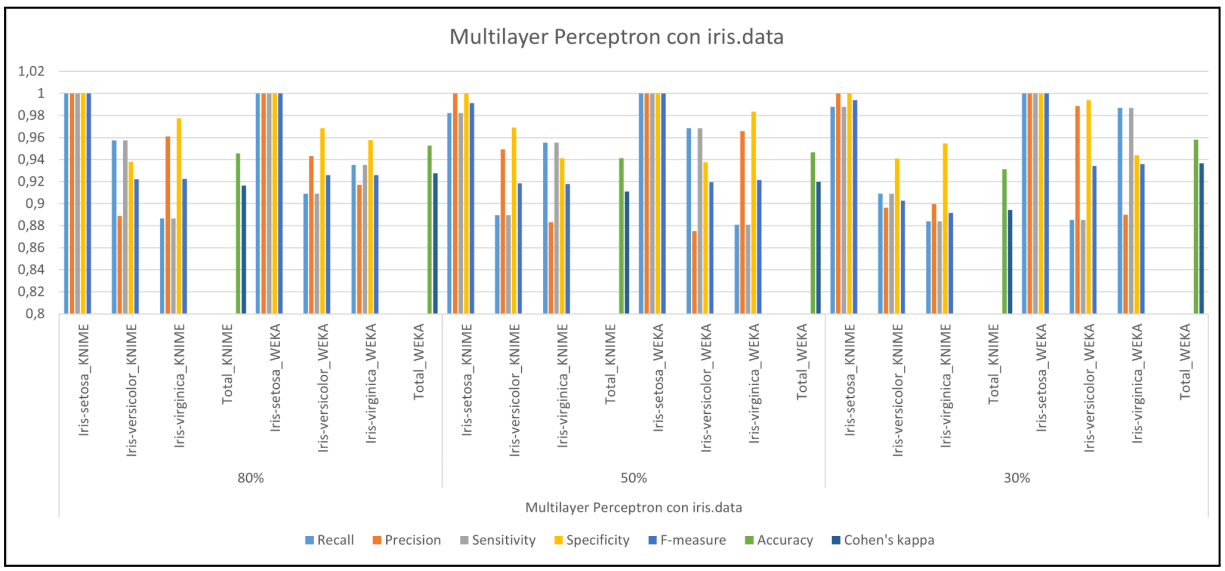

Figura 14: Gráfico de MLP con el fichero "iris.data"

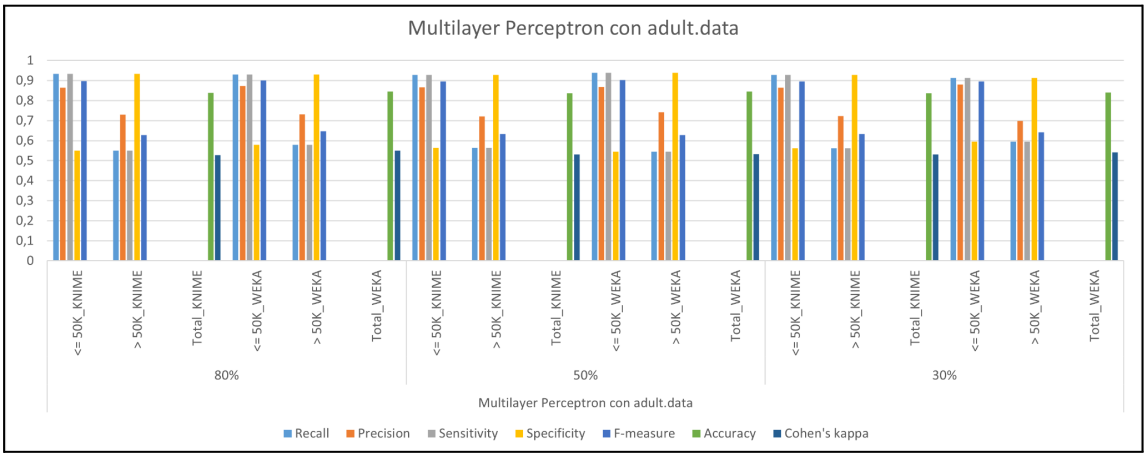

Figura 15: Gráfico de MLP con el fichero "adult.data"

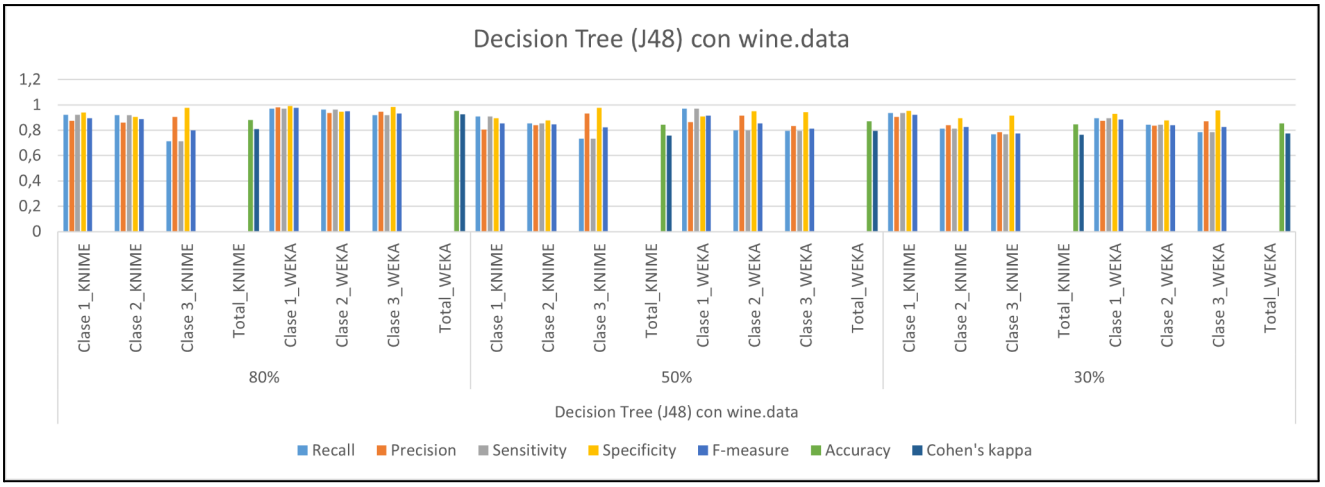

Figura 16: Gráfico de Decision Tree (J48) con el fichero "wine.data"

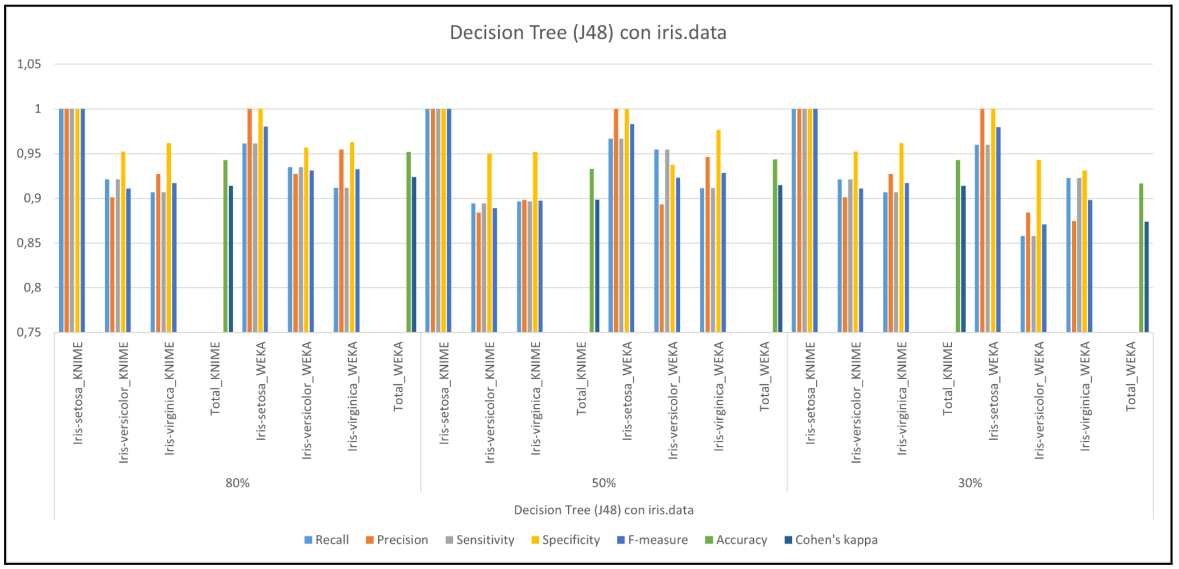

Figura 17: Gráfico de Decision Tree (J48) con el fichero "iris.data"

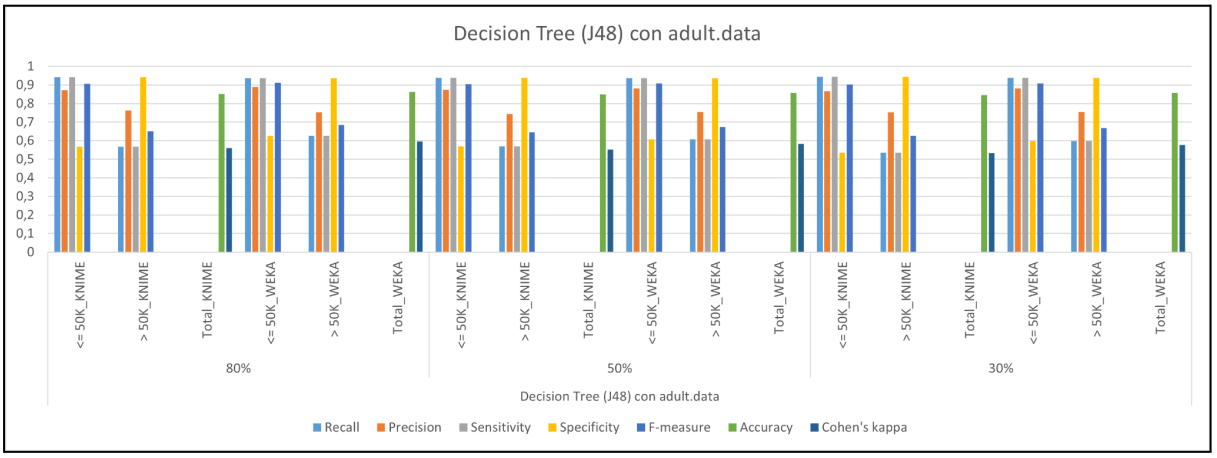

Figura 18: Gráfico de MLP con el fichero "adult.data"