Guía paso a paso para el Workflow de Clasificación en KNIME

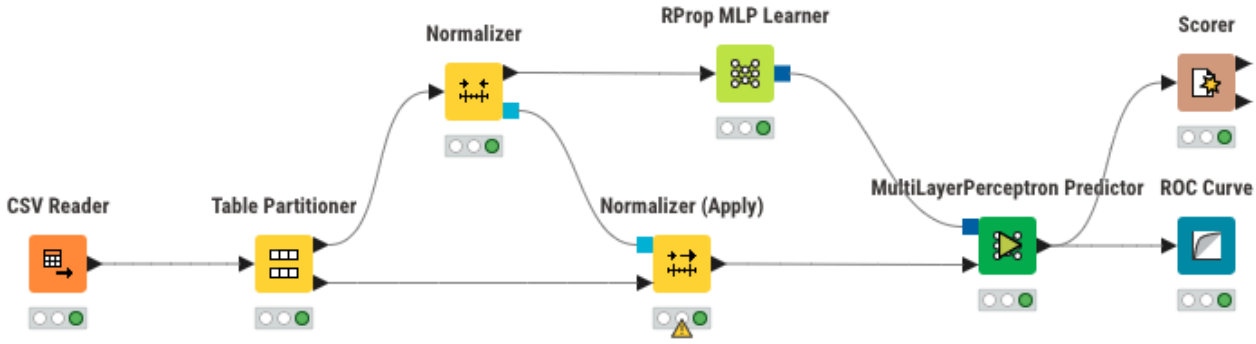
Introducción a la Clasificación

La **clasificación** es una tarea de aprendizaje supervisado cuyo objetivo es asignar una **etiqueta o clase** a cada observación de un conjunto de datos, en función de sus características (atributos). A diferencia de la regresión, donde la variable objetivo es numérica, en clasificación la salida es **categórica** (por ejemplo: “sí/no”, “positivo/negativo”, “A/B/C”).

En este caso trabajaremos con el **dataset de cáncer de mama** (Breast Cancer), que contiene características de células obtenidas a partir de imágenes de tejido mamario. La **variable objetivo** es **diagnosis (columna target)**, que toma dos valores:

- **Maligno (M)** → indica presencia de tumor cancerígeno.
- **Benigno (B)** → indica que el tumor no es cancerígeno.

Este problema es un caso típico de clasificación binaria en el ámbito de la salud.

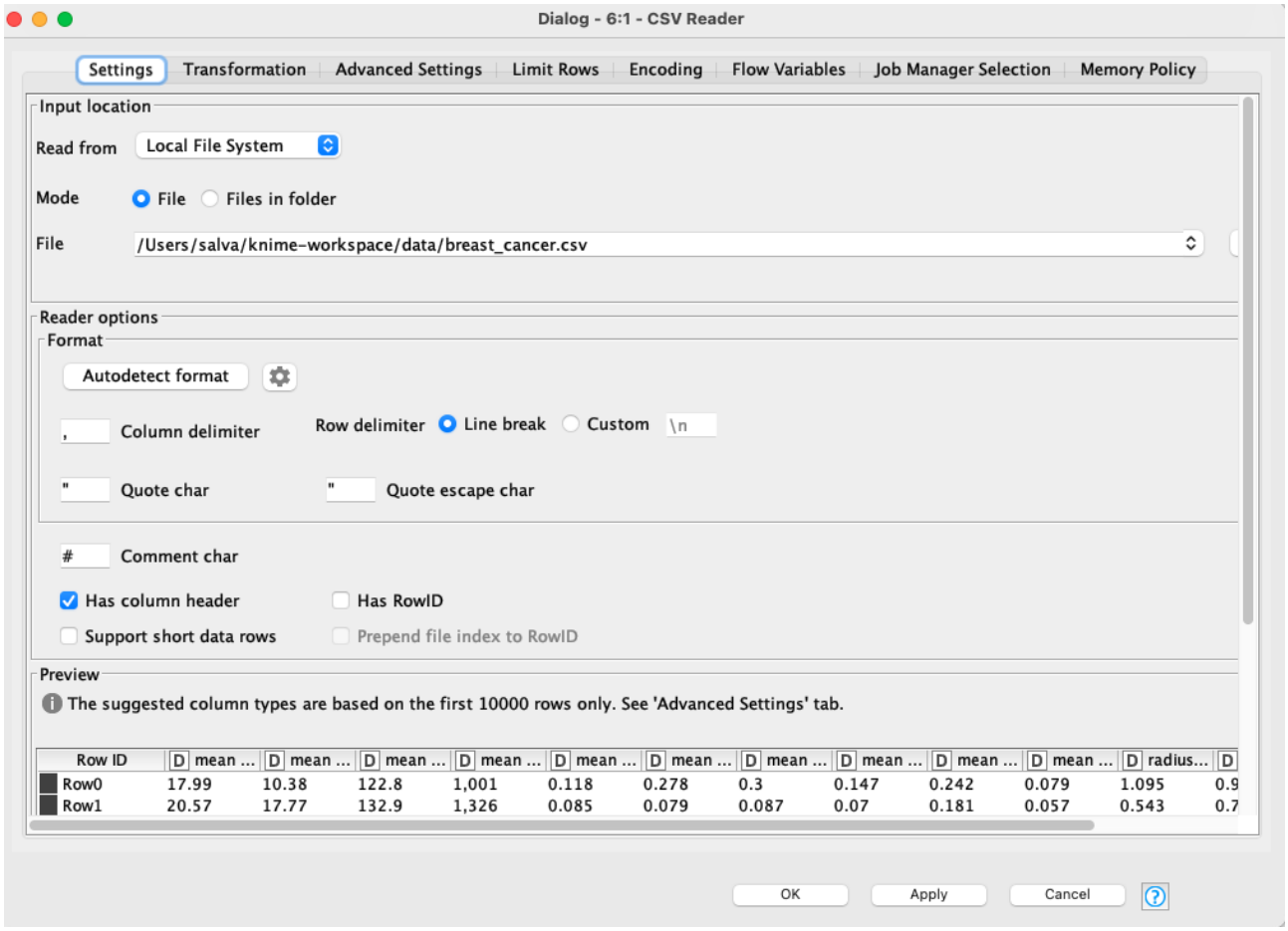


1. CSV Reader

Objetivo: Importar el dataset de cáncer de mama desde un archivo CSV.
Configuración:

- Seleccionar el archivo `breast_cancer.csv`.
- Indicar que contiene encabezados de columnas.
- Verificar que las variables predictoras (ej. radio, textura, perímetro, área, suavidad, etc.) estén en formato numérico.
- Revisar que la variable **target** (clase) sea categórica con valores:
    - **maligno**
    - **benigno**

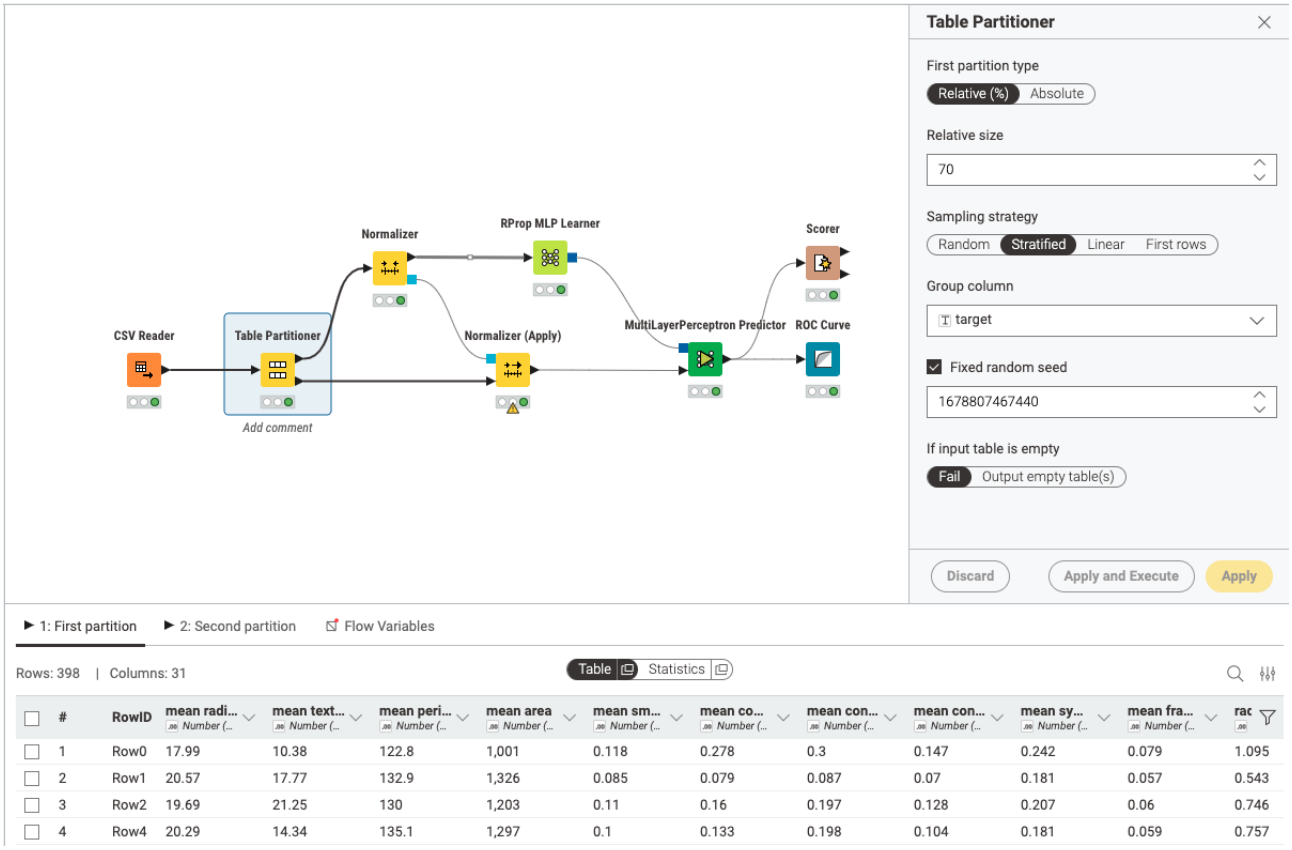
- Si la columna aparece como texto/string, KNIME la reconoce directamente como columna nominal, lista para usar en clasificación.



2. Table Partitioner

Objetivo: Dividir los datos en **entrenamiento** y **prueba**.
Configuración:

- Método: *Relative*.
- Training fraction: 0.7 (70%) y Test fraction: 0.3 (30%).
- Activar *Stratified sampling* para mantener la proporción de clases.
- Fijar *Random seed* (ej. 12345) para reproducibilidad.
- Salida superior: entrenamiento, salida inferior: prueba.

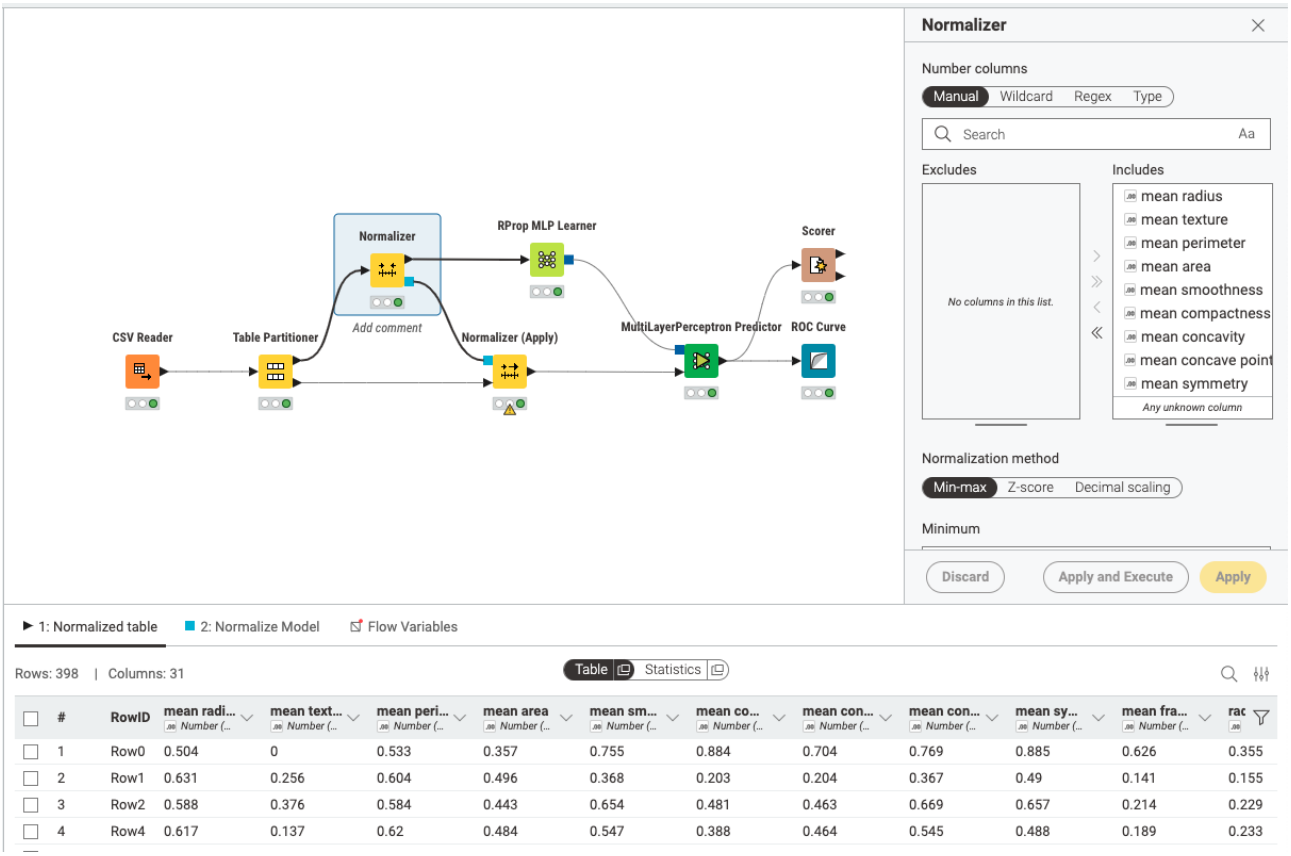


#	RowID	mean radi...	mean text...	mean peri...	mean area	mean sm...	mean co...	mean con...	mean con...	mean sy...	mean fra...	rac
1	Row0	17.99	10.38	122.8	1,001	0.118	0.278	0.3	0.147	0.242	0.079	1.095
2	Row1	20.57	17.77	132.9	1,326	0.085	0.079	0.087	0.07	0.181	0.057	0.543
3	Row2	19.69	21.25	130	1,203	0.11	0.16	0.197	0.128	0.207	0.06	0.746
4	Row4	20.29	14.34	135.1	1,297	0.1	0.133	0.198	0.104	0.181	0.059	0.757

3. Normalizer (Entrenamiento)

Objetivo: Escalar las variables numéricas del conjunto de entrenamiento.
Configuración:

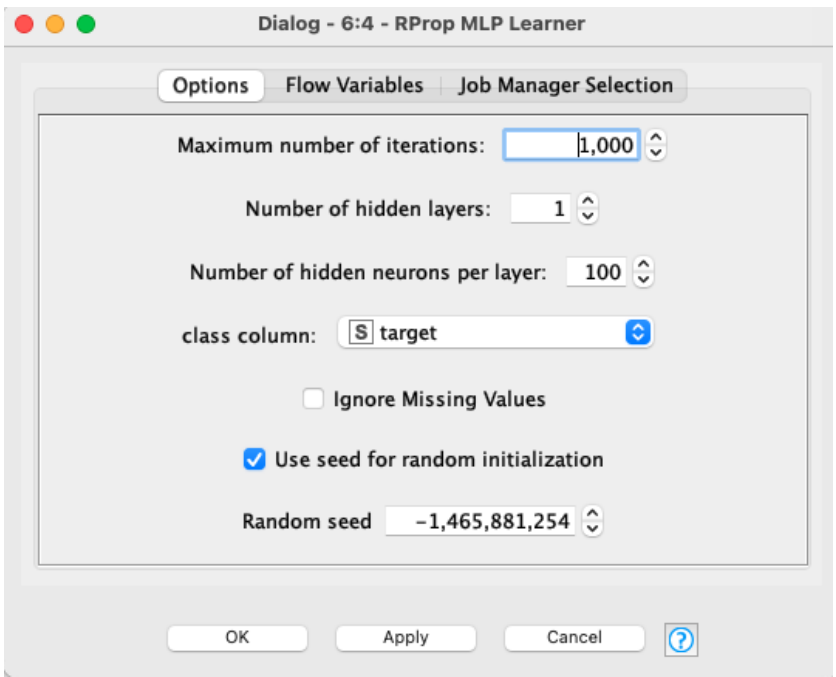
- Método: ***Min-Max [0,1]*** o *Z-Score*.
- Incluir solo variables numéricas predictoras.
- Excluir columna de clase e identificadores (al ser nominal, ha sido excluida automáticamente).
- Salida: datos normalizados y modelo de normalización (puerto azul).



4. RProp MLP Learner

Objetivo: Entrenar una red neuronal multicapa (MLP).
Configuración:

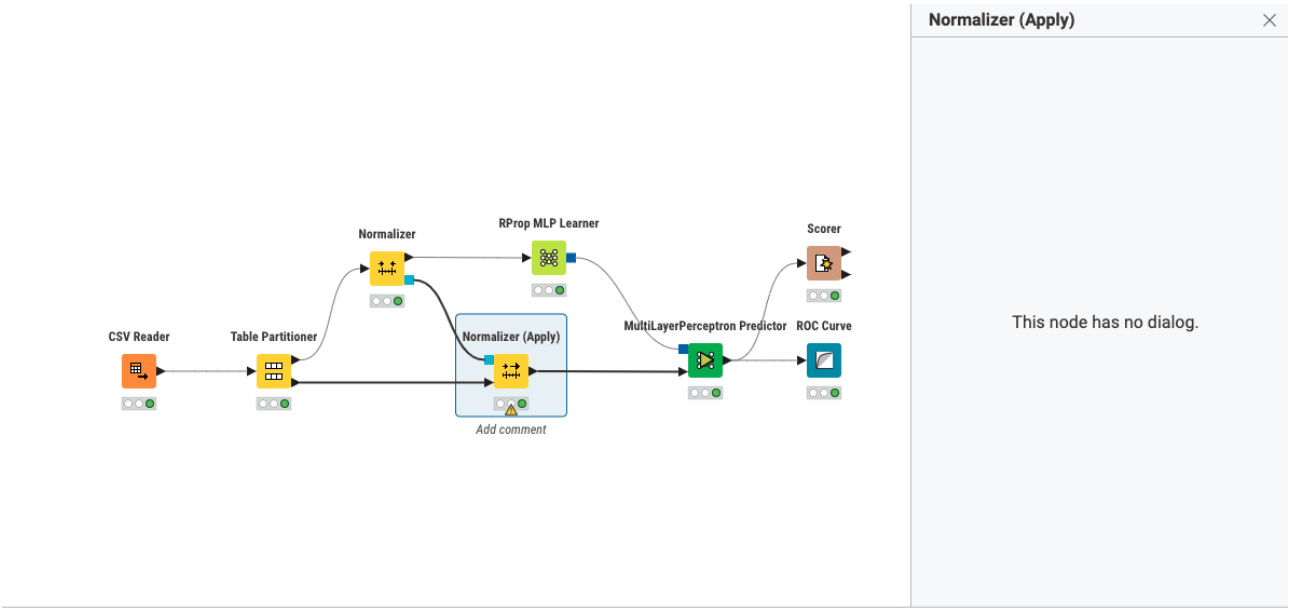
- **Target column:** columna de clase.
- **Hidden layers:** arquitectura (1 capa oculta con 100).
- **Iterations/Epochs:** definir número de ciclos de entrenamiento (1000).
- Entrada: conjunto de entrenamiento ya normalizado.
- Salida: modelo entreando (puerto azul).



5. Normalizer (Apply) sobre el conjunto de prueba

Objetivo: Aplicar la normalización aprendida al conjunto de prueba.
Configuración:

- Conectar el **puerto azul (PMML)** del Normalizer de entrenamiento a este nodo.
- Conectar la salida de prueba del Table Partitioner a este nodo.
- Resultado: test normalizado sin fuga de información (*no se ajusta con datos de test*).



1: Normalized output | Flow Variables

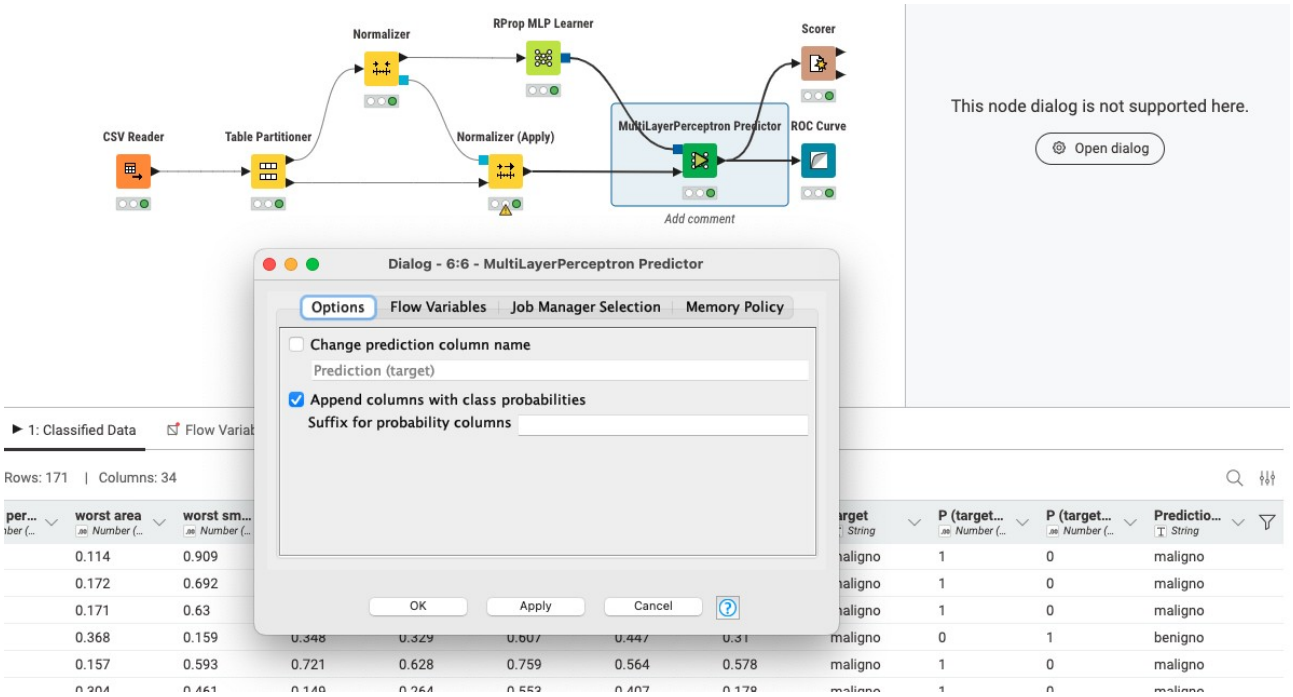
Rows: 171 | Columns: 31 — Table | Statistics

#	RowID	mean radi... Number (...	mean text... Number (...	mean peri... Number (...	mean area Number (...	mean sm... Number (...	mean co... Number (...	mean con... Number (...	mean con... Number (...	mean sy... Number (...	mean fra... Number (...	rac
1	Row3	0.183	0.346	0.211	0.093	1.031	0.906	0.566	0.55	1.001	1.037	0.138
2	Row5	0.233	0.184	0.246	0.132	0.862	0.516	0.37	0.423	0.669	0.569	0.079
3	Row8	0.26	0.396	0.282	0.15	0.857	0.595	0.436	0.489	0.84	0.52	0.069
4	Row12	0.562	0.499	0.601	0.409	0.514	0.775	0.484	0.584	0.871	0.61	0.305

6. MultiLayerPerceptron Predictor

Objetivo: Generar predicciones sobre el conjunto de prueba.
Configuración:

- Conectar el **modelo** (del Learner). Puertos azules.
- Conectar los **datos normalizados de prueba** (del Normalizer Apply).
- Activar **Append class probabilities** (necesarias para curva ROC).
- Output: columna con clase predicha y probabilidades.

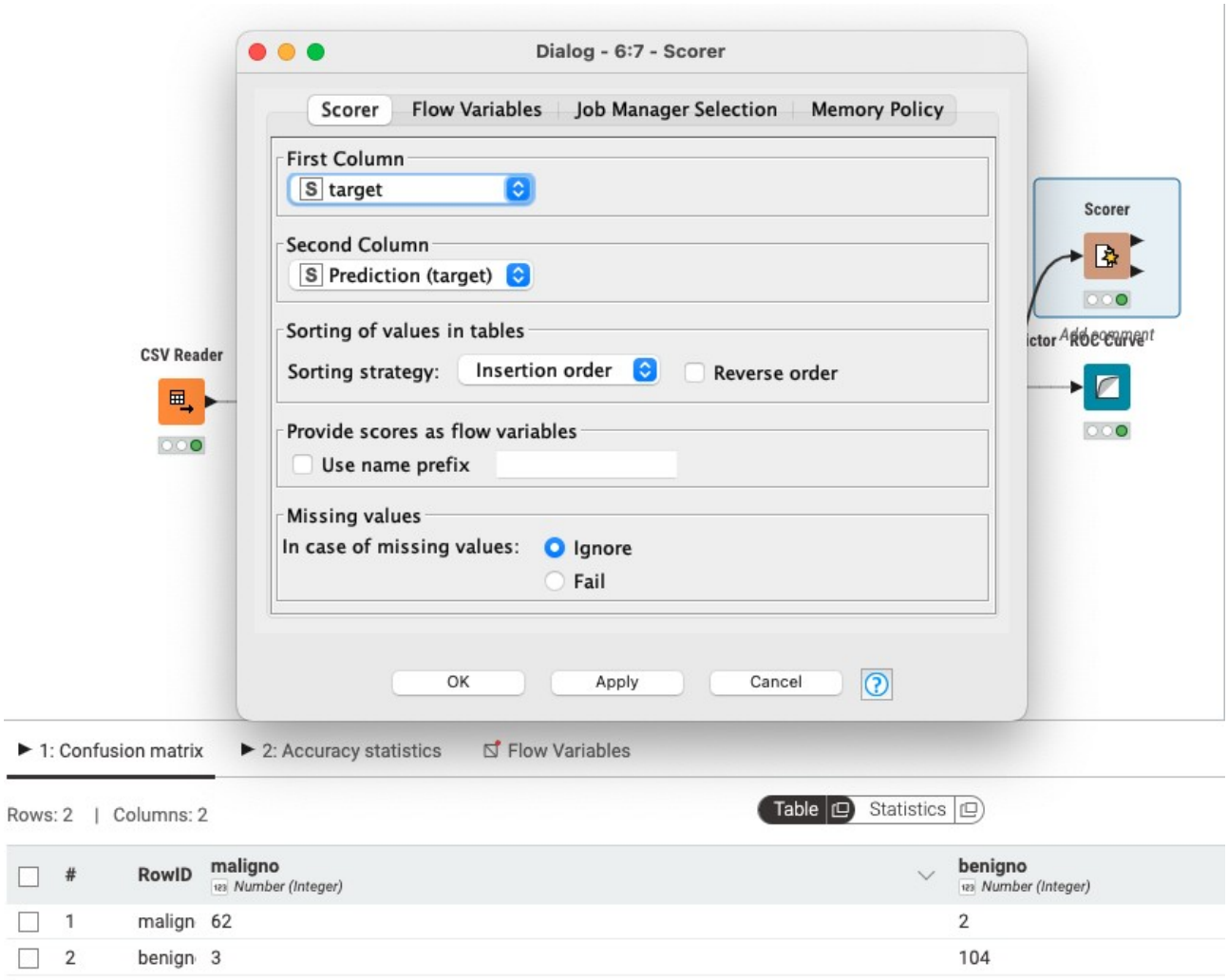


per...	worst area	worst sm...						P (target...	P (target...	Predictio...
	0.114	0.909					maligno	1	0	maligno
	0.172	0.692					maligno	1	0	maligno
	0.171	0.63					maligno	1	0	maligno
	0.368	0.159	0.348	0.329	0.607	0.447	0.31	maligno	0	1	benigno
	0.157	0.593	0.721	0.628	0.759	0.564	0.578	maligno	1	0	maligno
	0.304	0.461	0.149	0.264	0.553	0.407	0.178	maligno	1	0	maligno

7. Scorer

Objetivo: Evaluar el desempeño del modelo en clasificación.
Configuración:

- Columna real: la columna de clase original.
- Columna predicha: la generada por el Predictor.
- Definir cuál es la **clase positiva** (importante en binaria).
- Salida: matriz de confusión, accuracy, precisión, recall, F1, etc.

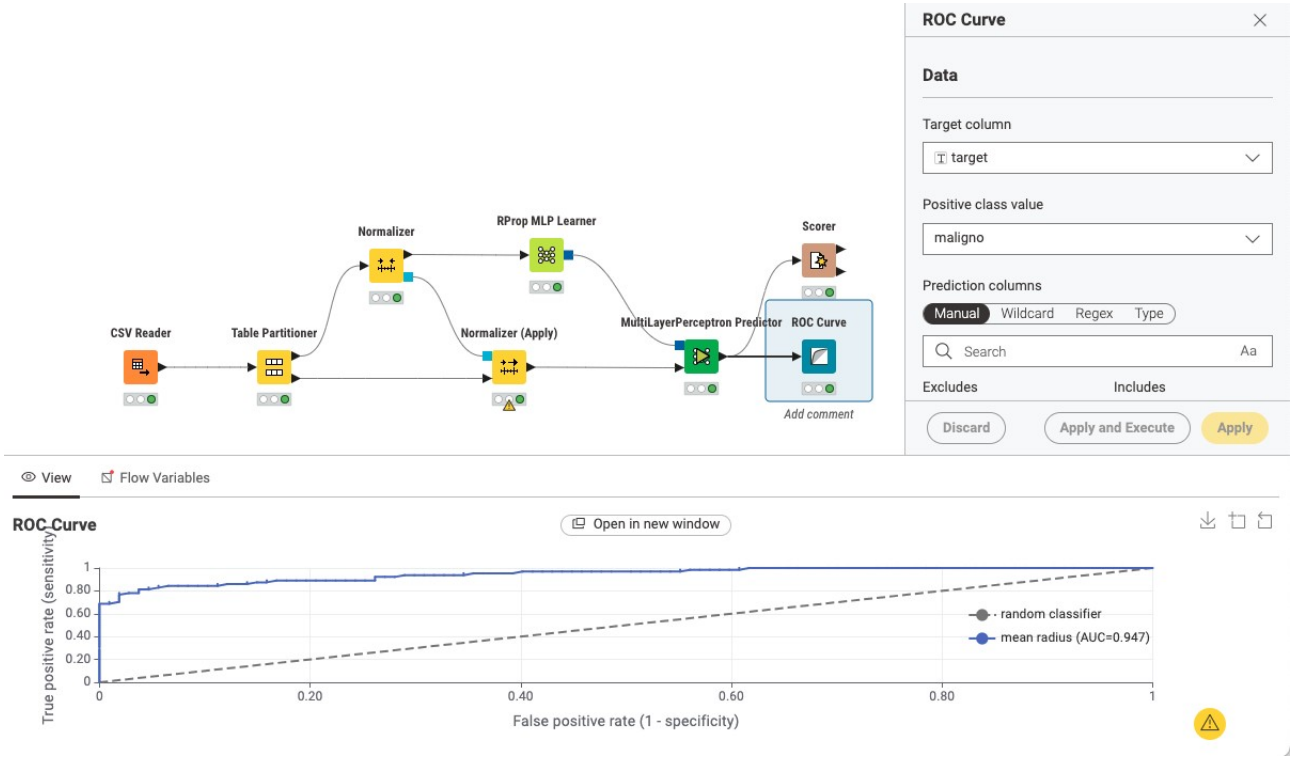


d

8. ROC Curve

Objetivo: Calcular y visualizar la curva ROC y el AUC.
Configuración:

- Clase real: columna objetivo.
- Probabilidad: seleccionar la probabilidad de la clase positiva.
- Salida: curva ROC y valor AUC para medir capacidad de discriminación del modelo.



✅ **Buenas prácticas:**

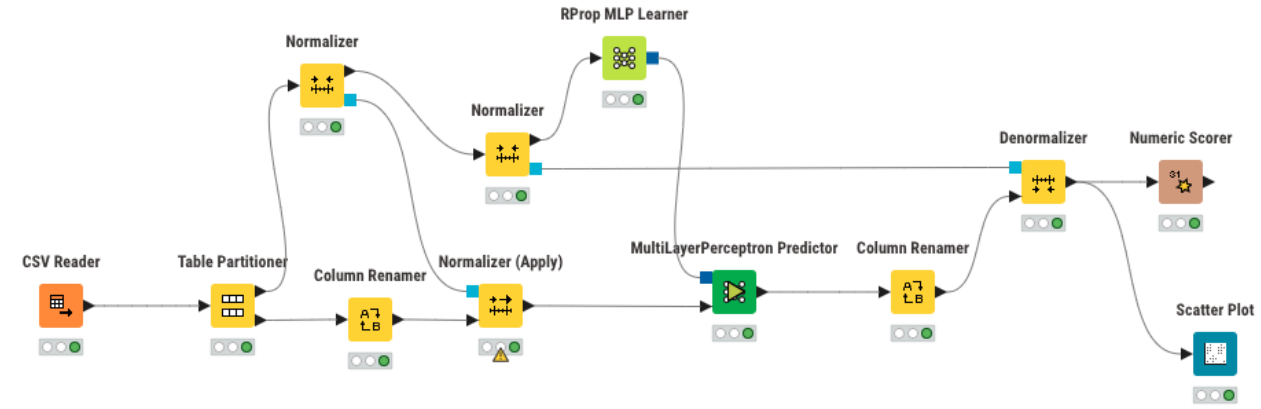
- Normalizar siempre con entrenamiento y aplicar a test (evitar leakage).
- Usar estratificación en la partición para problemas de clases desbalanceadas.
- Revisar probabilidades y métricas antes de ajustar hiperparámetros.

Guía paso a paso para el Workflow de Regresión en KNIME

Introducción a la regresión (Auto MPG)

La **regresión** es una tarea de aprendizaje supervisado en la que se predice una **variable numérica continua** a partir de un conjunto de características. En este caso, el objetivo es estimar el **consumo de combustible** de un automóvil, medido como **MPG (miles per gallon)**.

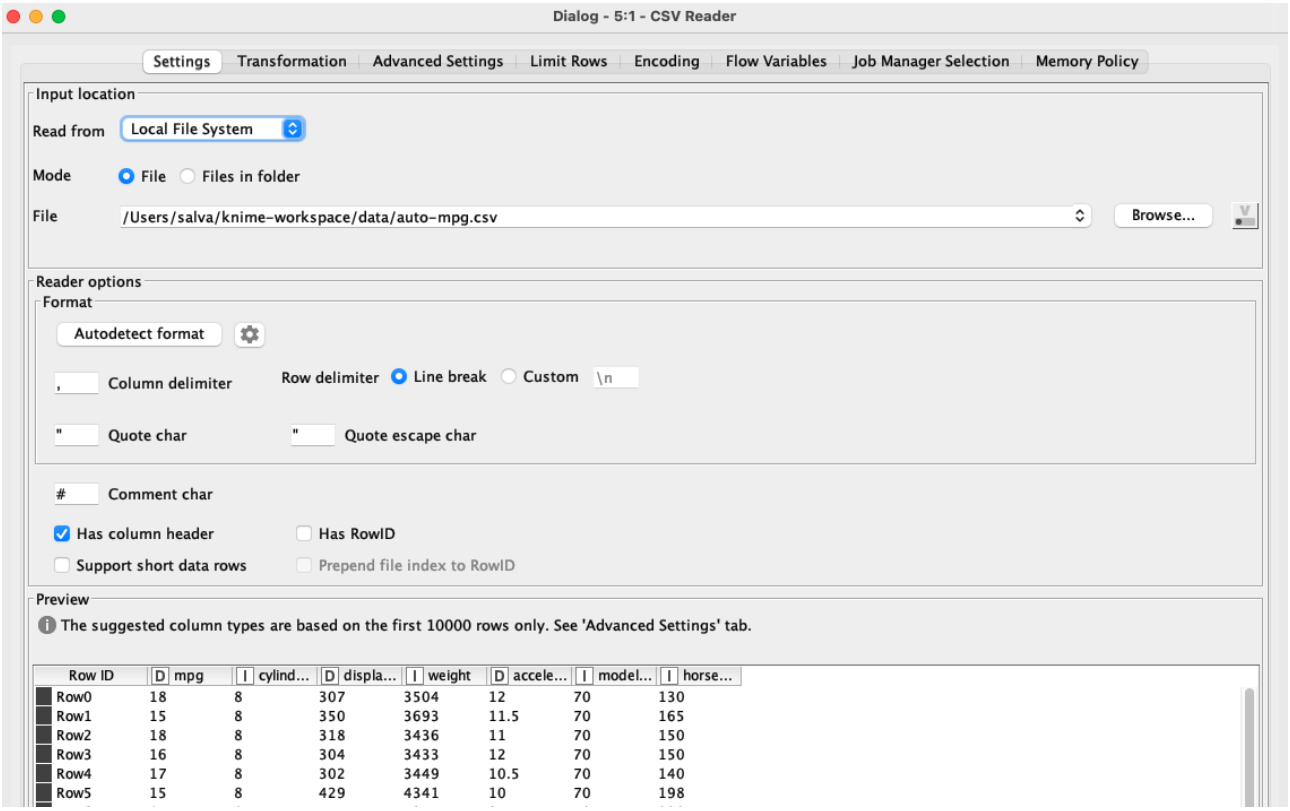
Dataset Auto MPG (resumen): contiene variables técnicas de vehículos (p. ej., `cylinders`, `displacement`, `horsepower`, `weight`, `acceleration`, `model year`, `origin`) y la **variable objetivo** `mpg` (numérica). Es común encontrar valores faltantes en `horsepower` en algunas versiones del dataset; si aparecen como `'?'`, trátalos como faltantes y rellénalos antes de entrenar.



1) CSV Reader

Objetivo: Cargar el archivo `auto-mpg.csv`.
Configuración recomendada:

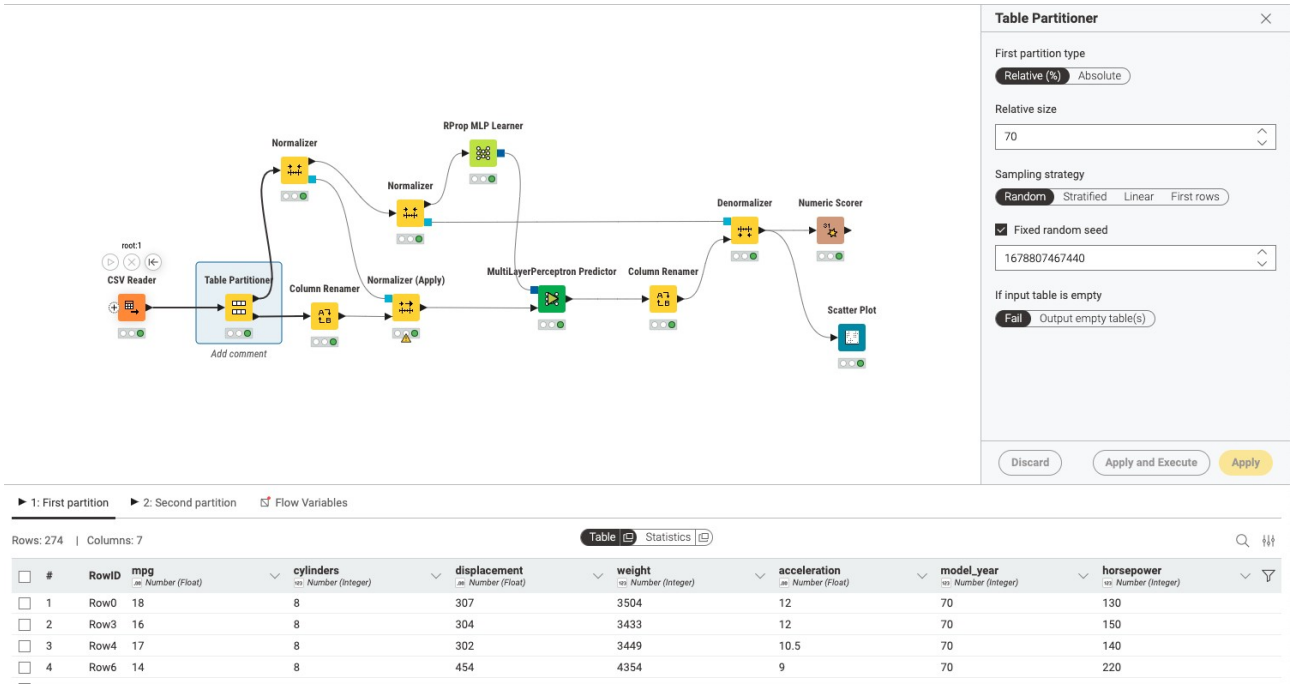
- **File:** seleccionar `auto-mpg.csv`.
- **Delimiter:** coma (`,`). Si tu archivo usa espacios/tabulaciones, ajustar el delimitador.
- **Has column header:** activado.
- **Tipos de columna:** verificar que **`mpg` sea numérica (Double)** y que las predictoras numéricas estén bien tipadas.



2) Table Partitioner

Objetivo: Dividir en **entrenamiento** y **prueba** sin fuga de información.
Configuración:

- **Partitioning:** *Relative*.
- **Training fraction:** 0.7 (70%); **Test:** 0.3 (20%).
- **Random seed:** fijar (ej. 12345) para reproducibilidad.
- **Stratified sampling:** no aplica a regresión (déjalo desactivado).
  Salidas:
- **Puerto superior:** entrenamiento.
- **Puerto inferior:** prueba.



▶ 1: First partition ▶ 2: Second partition Flow Variables

Rows: 274 | Columns: 7 Table Statistics

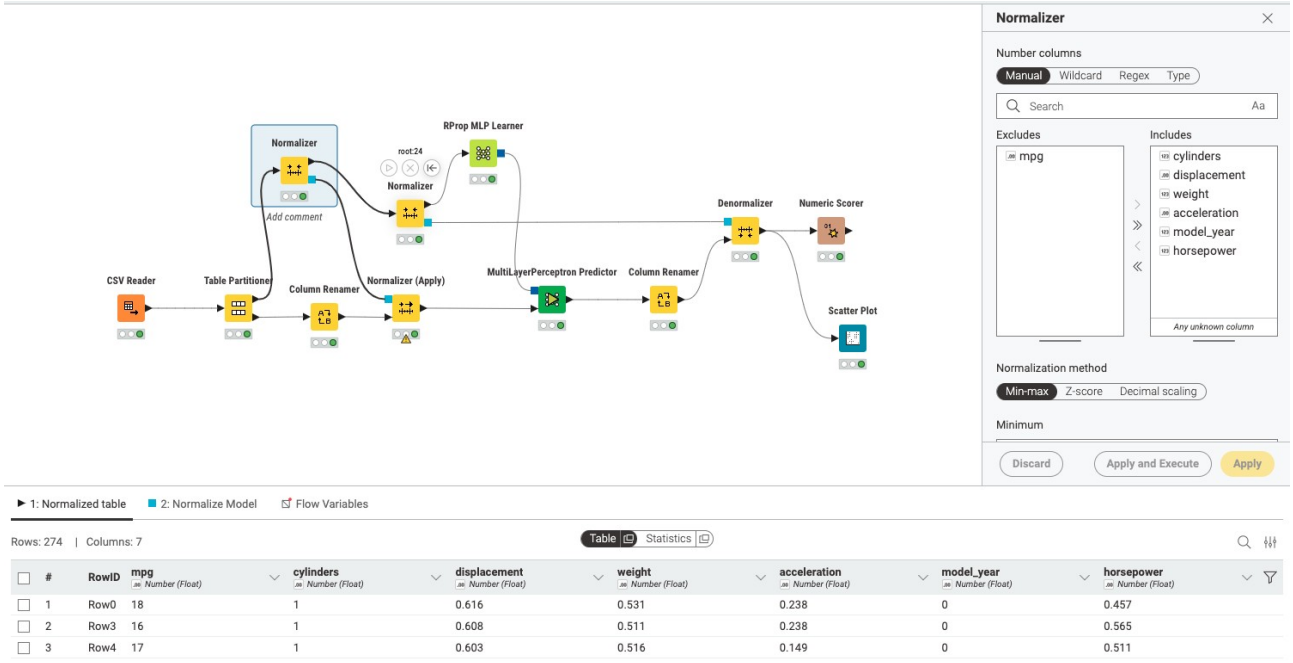
#	RowID	mpg Number (Float)	cylinders Number (Integer)	displacement Number (Float)	weight Number (Integer)	acceleration Number (Float)	model_year Number (Integer)	horsepower Number (Integer)
1	Row0	18	8	307	3504	12	70	130
2	Row3	16	8	304	3433	12	70	150
3	Row4	17	8	302	3449	10.5	70	140
4	Row6	14	8	454	4354	9	70	220

3) Normalizer (entrenamiento – X)

Objetivo: Escalar las columnas numéricas **excluyendo la variable objetivo `mpg` .**
Configuración:

- **Method:** *Min–Max [0,1]* o *Z-Score* (cualquiera funciona; *Min–Max* suele ser estable para MLP).
- **Include:** todas las **predictoras numéricas.**
- **Exclude:** mpg (variable objetivo).
  Salidas:
- **Tabla normalizada (X train)**.

- **Puerto azul (PMML):** modelo de normalización (lo usaremos para *Apply* y luego para **Denormalizer**).



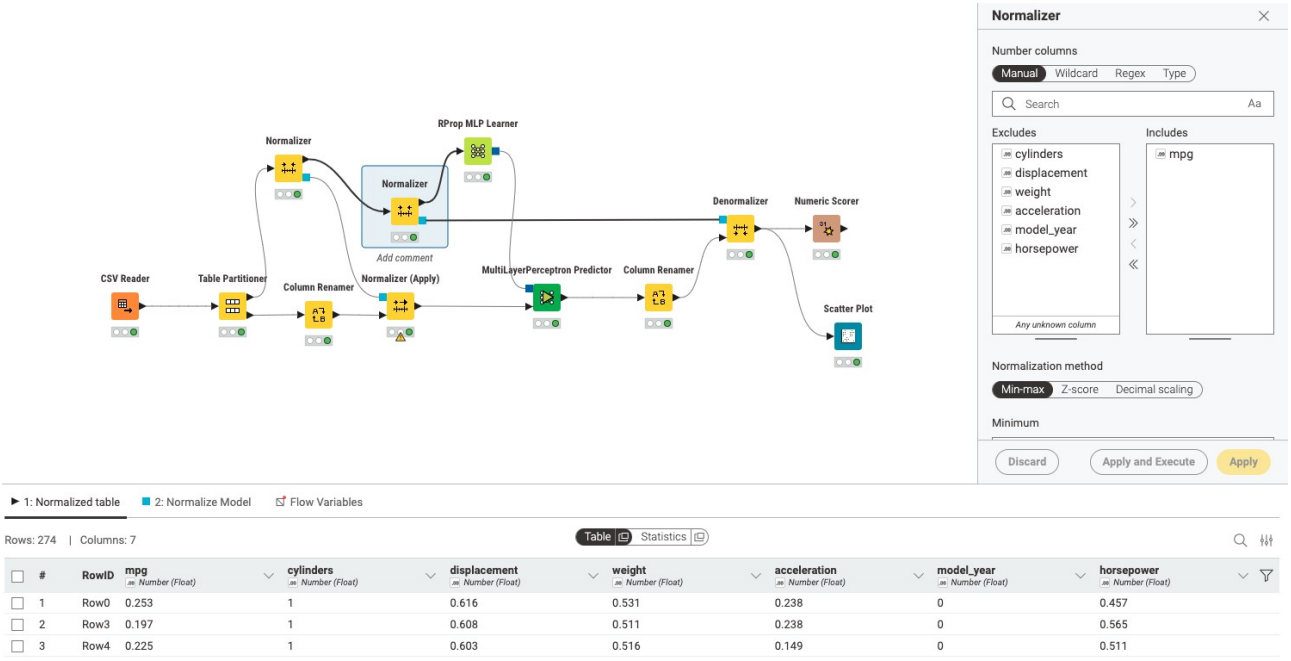
#	RowID	mpg Number (Float)	cylinders Number (Float)	displacement Number (Float)	weight Number (Float)	acceleration Number (Float)	model_year Number (Float)	horsepower Number (Float)
1	Row0	18	1	0.616	0.531	0.238	0	0.457
2	Row3	16	1	0.608	0.511	0.238	0	0.565
3	Row4	17	1	0.603	0.516	0.149	0	0.511

4) Normalizer (entrenamiento – y)

Objetivo: Escalar **la variable objetivo** `mpg` (el MLP de KNIME para regresión funciona mejor si la **y** está escalada).

Configuración:

- **Method:** *Min–Max [0,1]* o *Z-Score* (cualquiera funciona; *Min–Max* suele ser estable para MLP).
- **Include:** mpg (variable objetivo).t
- **Exclude:** odas las **predictoras numéricas.**
  Salidas:
- **Columna normalizada (mpg: y train)**.
- **Puerto azul (PMML):** modelo de normalización (lo usaremos para *Apply* y luego para **Denormalizer**).

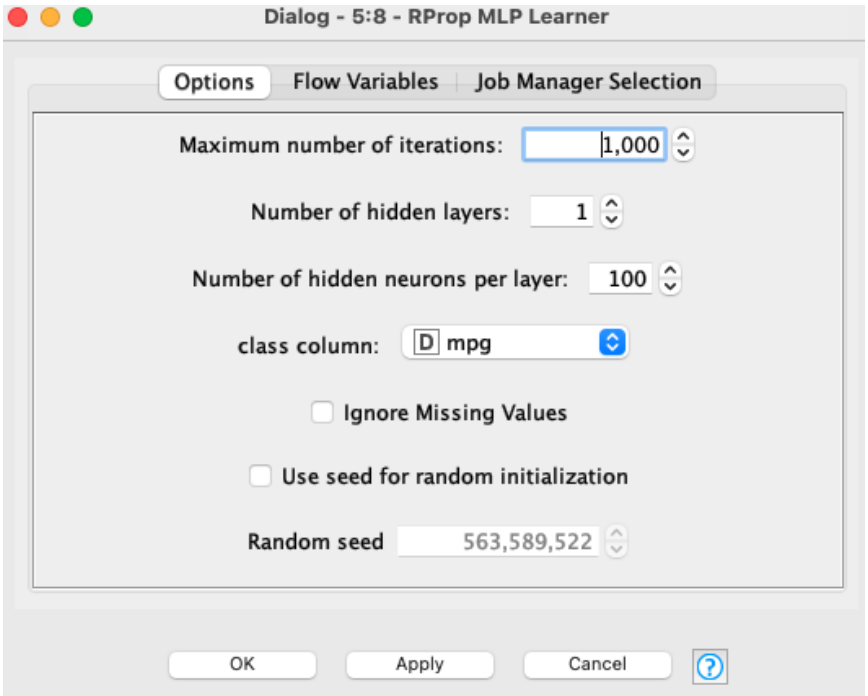


#	RowID	mpg	cylinders	displacement	weight	acceleration	model_year	horsepower
		Number (Float)	Number (Float)	Number (Float)	Number (Float)	Number (Float)	Number (Float)	Number (Float)
1	Row0	0.253	1	0.616	0.531	0.238	0	0.457
2	Row3	0.197	1	0.608	0.511	0.238	0	0.565
3	Row4	0.225	1	0.603	0.516	0.149	0	0.511

5) RProp MLP Learner

Objetivo: Entrenar una red **MLP para regresión**.
Entradas: datos **entrenamiento normalizados** (salida tabla del Normalizer).
Configuración clave:

- **Target column:** `mpg` (numérica).
- **Hidden layers:** arquitectura (`1`), **Neurons:** 100.
- **Iterations / Max epochs:** ajusta según convergencia (1000).
- **Early stopping / Tolerancia:** si está disponible, configúralo para evitar sobreajuste.
- La salida/activación de la capa final será **lineal** automáticamente al detectar objetivo numérico.

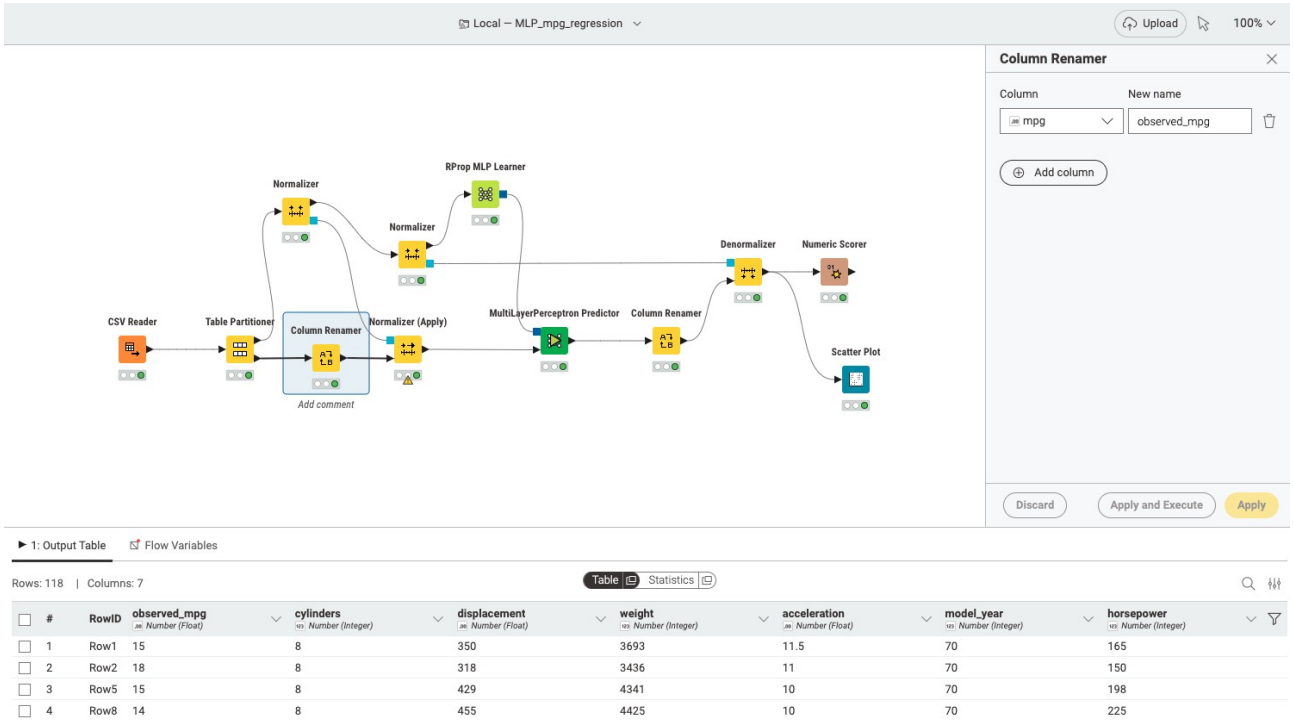


Configuración para datos de prueba

6) Column Renamer (opcional pero recomendable)

Objetivo: Asegurar nombres consistentes (especialmente la variable objetivo).
Sugerencias:

- Renombrar la columna objetivo **`mpg`** a observed_mpg para el dataset de prueba.



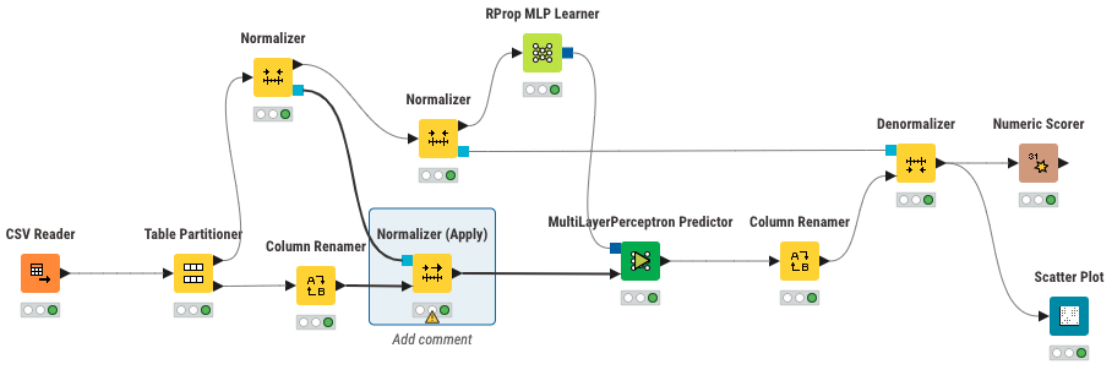
#	RowID	observed_mpg Number (Float)	cylinders Number (Integer)	displacement Number (Float)	weight Number (Integer)	acceleration Number (Float)	model_year Number (Integer)	horsepower Number (Integer)
1	Row1	15	8	350	3693	11.5	70	165
2	Row2	18	8	318	3436	11	70	150
3	Row5	15	8	429	4341	10	70	198
4	Row8	14	8	455	4425	10	70	225

7) Normalizer (Apply) — sobre el conjunto de prueba X

Objetivo: Aplicar **la misma** normalización aprendida en (4) a los datos de **prueba**.
Conexiones:

- **PMML (azul)** del Normalizer de (4) → **PMML (azul)** de este nodo.
- **Tabla de prueba** desde Table Partitioner → **datos** de este nodo.
  Resultado: prueba **normalizada de forma idéntica** a train (sin *leakage*).

  Si en (4) incluimos `mpg`, aquí también quedará `mpg` de prueba **escalada** (se revertirá después).



▶ 1: Normalized output | Flow Variables

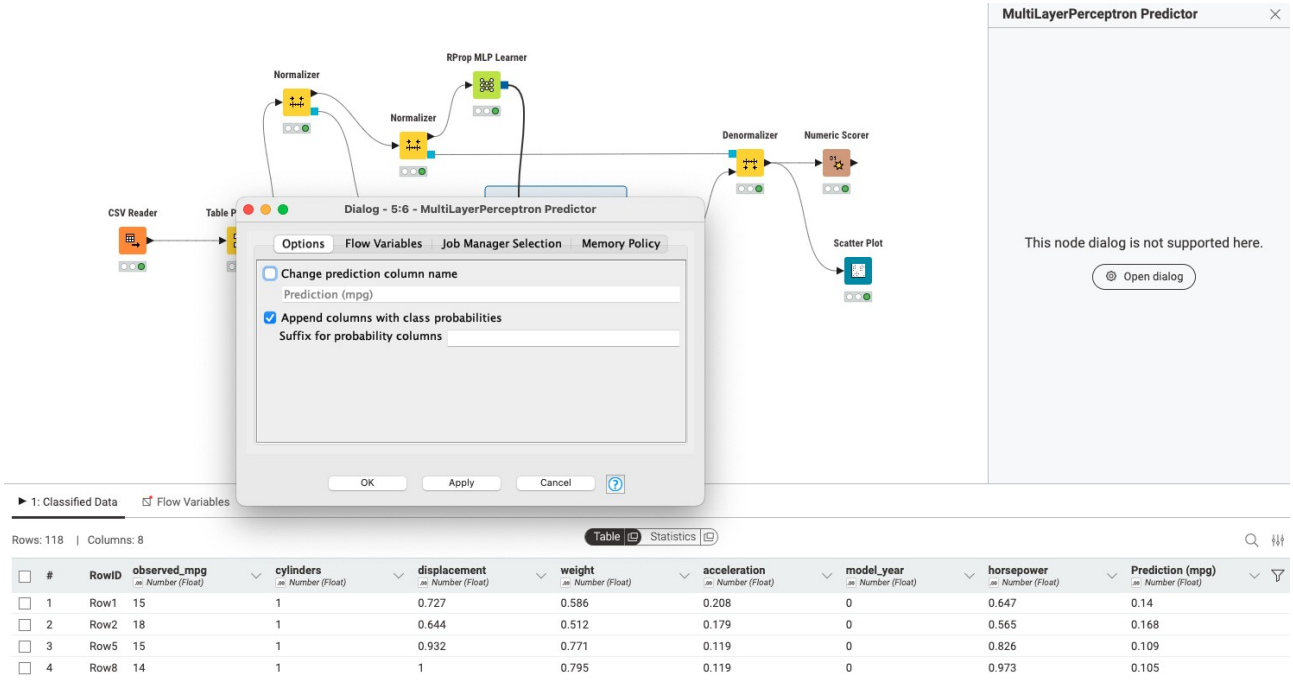
Rows: 118 | Columns: 7 — Table | Statistics

#	RowID	observed_mpg (Number (Float))	cylinders (Number (Float))	displacement (Number (Float))	weight (Number (Float))	acceleration (Number (Float))
1	Row1	15	1	0.727	0.586	0.208
2	Row2	18	1	0.644	0.512	0.179
3	Row5	15	1	0.932	0.771	0.119
4	Row8	14	1	1	0.795	0.119

8) MultiLayerPerceptron Predictor

Objetivo: Obtener predicciones del MLP sobre el **test normalizado**.
Conexiones:

- **Modelo** desde RProp MLP Learner.
- **Datos** desde Normalizer (Apply).
  Configuración:
- La salida será una columna con la **predicción de `mpg` en escala normalizada** (si `mpg` fue normalizada en 4).
- Nombra la columna de predicción con claridad (por defecto será `prediction(mpg)`).

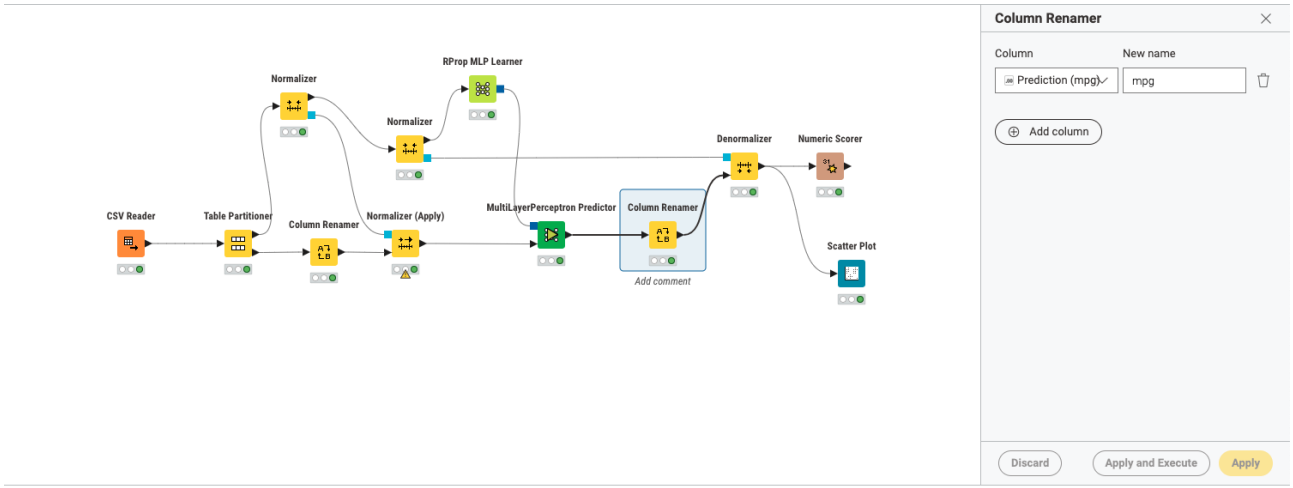


#	RowID	observed_mpg	cylinders	displacement	weight	acceleration	model_year	horsepower	Prediction (mpg)
		Number (Float)	Number (Float)	Number (Float)	Number (Float)	Number (Float)	Number (Float)	Number (Float)	Number (Float)
1	Row1	15	1	0.727	0.586	0.208	0	0.647	0.14
2	Row2	18	1	0.644	0.512	0.179	0	0.565	0.168
3	Row5	15	1	0.932	0.771	0.119	0	0.826	0.109
4	Row8	14	1	1	0.795	0.119	0	0.973	0.105

9) Column Renamer (post-predicción)

Objetivo: Dejar nombres claros para evaluar y graficar.
Sugerencias:

- Renombra `Prediction(mpg)` → **`mpg`** (note que aún está en escala normalizada).

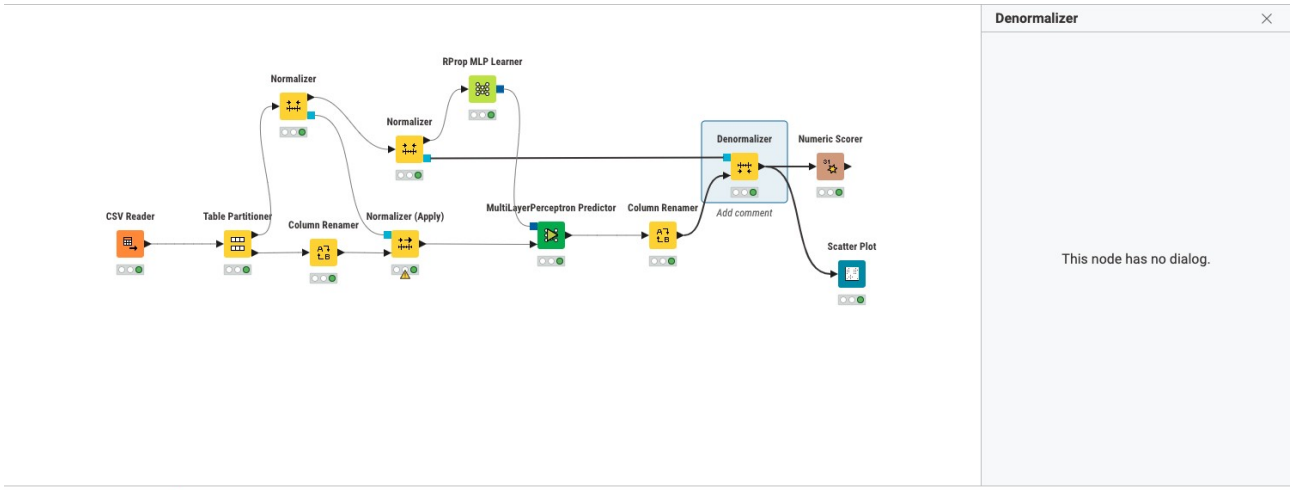


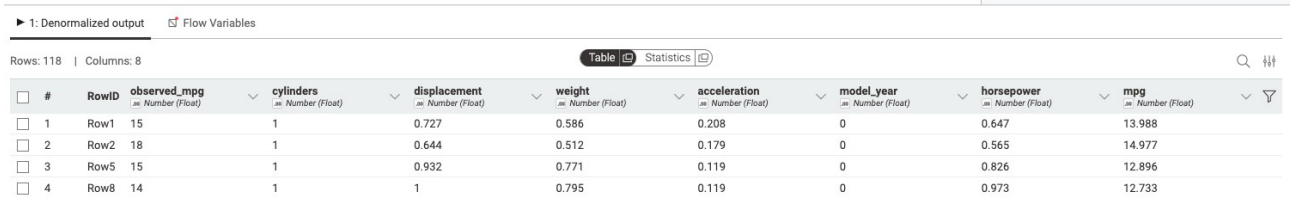
#	RowID	observed_mpg	cylinders	displacement	weight	acceleration	model_year	horsepower	mpg
		Number (Float)	Number (Float)	Number (Float)	Number (Float)	Number (Float)	Number (Float)	Number (Float)	Number (Float)
1	Row1	15	1	0.727	0.586	0.208	0	0.647	0.14
2	Row2	18	1	0.644	0.512	0.179	0	0.565	0.168
3	Row5	15	1	0.932	0.771	0.119	0	0.826	0.109

10) Denormalizer

Objetivo: Volver a la escala original (des-normalizar) **`mpg`** (es la variable **predicha)**.
Conexiones:

- **PMML (azul)** del Normalizer de (4) → **PMML (azul)** del Denormalizer.
- **Tabla** desde el Column Renamer (8).
  Configuración:
- Asegúrate de que el Denormalizer aplique la **inversa** a las columnas que fueron normalizadas, en este caso **`mpg`.**
  Resultado esperado:
- `mpg` (predicho) en **escala original**.
- `observed_mpg` (datos observados en el conjunto de test).





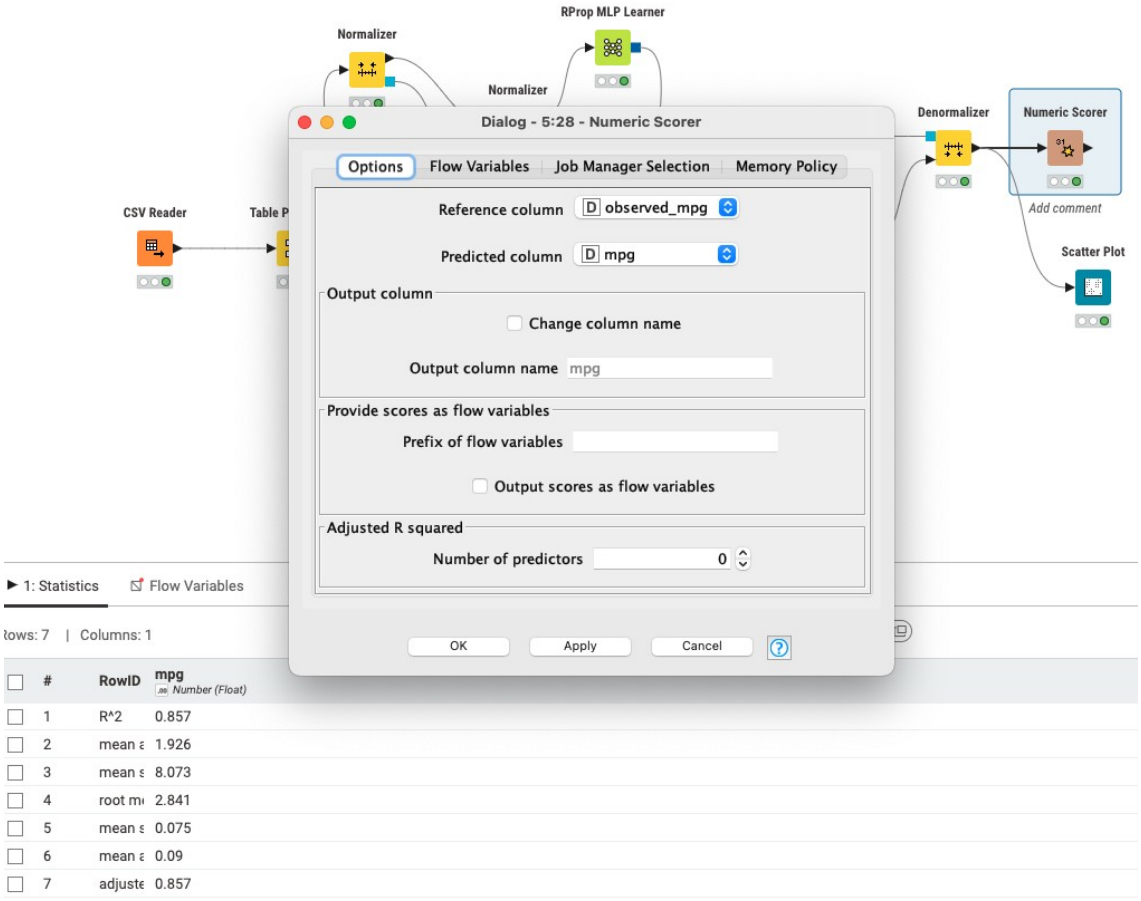
#	RowID	observed_mpg	cylinders	displacement	weight	acceleration	model_year	horsepower	mpg
		Number (Float)	Number (Float)	Number (Float)	Number (Float)	Number (Float)	Number (Float)	Number (Float)	Number (Float)
1	Row1	15	1	0.727	0.586	0.208	0	0.647	13.988
2	Row2	18	1	0.644	0.512	0.179	0	0.565	14.977
3	Row5	15	1	0.932	0.771	0.119	0	0.826	12.896
4	Row8	14	1	1	0.795	0.119	0	0.973	12.733

11) Numeric Scorer (Regresión)

Objetivo: Calcular métricas de desempeño.
Configuración:

- **Actual column (real):** `observed_mpg`.
- **Predicted column:** `mpg`.
  Métricas disponibles: MSE, RMSE, MAE, R^2, etc.

Trabajar en **escala original** hace que las métricas sean interpretables (ej. RMSE en MPG).

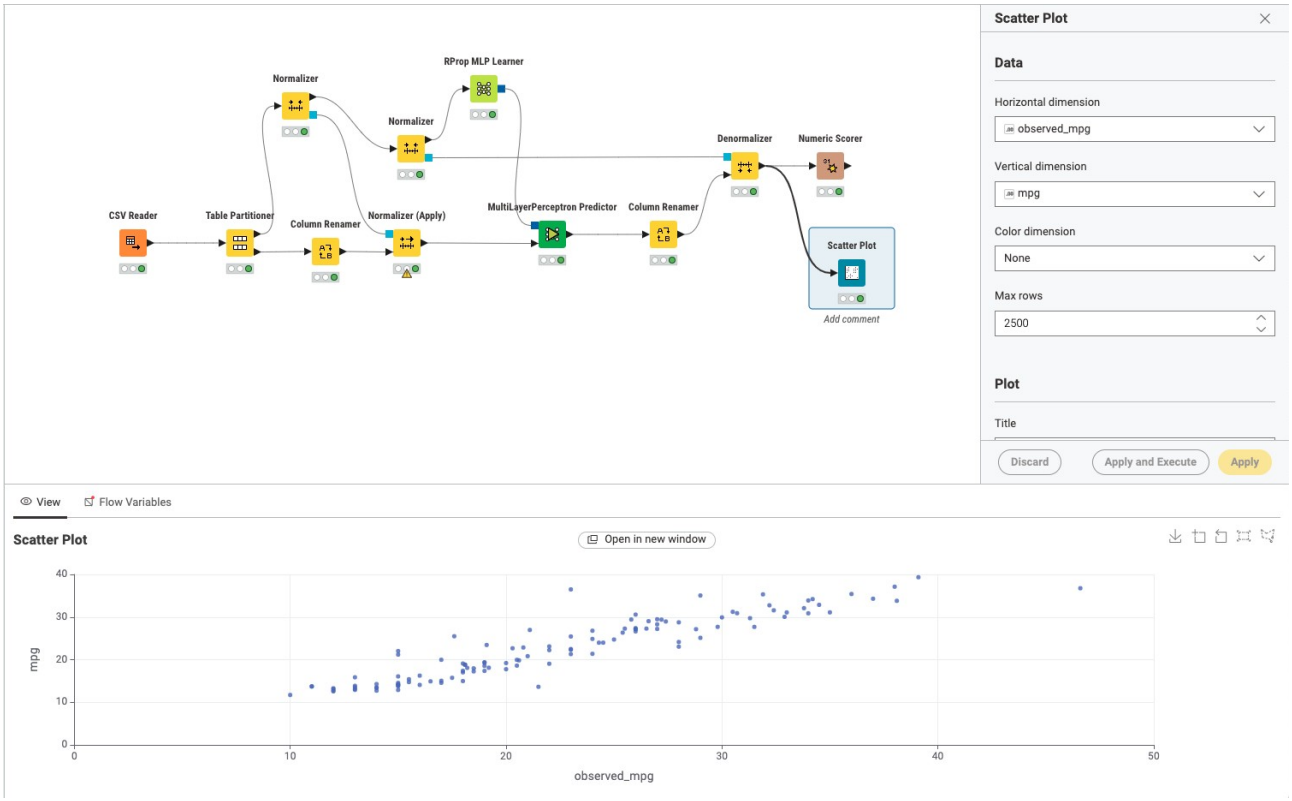


12) Scatter Plot

Objetivo: Visualizar **`mpg` vs `mpg_pred`** para evaluar ajuste.
Configuración:

- **X-axis:** `mpg` (real).
- **Y-axis:** `mpg_pred` (predicha).
- Espera un **patrón cercano a la diagonal**; grandes desviaciones indican errores sistemáticos.

(Opcional) Usa “Color by” para colorear por `cylinders` o `origin` y observar sesgos.


Normalizer
RProp MLP Learner
Normalizer
Denormalizer
Numeric Scorer
CSV Reader
Table Partitioner
Column Renamer
Normalizer (Apply)
MultiLayerPerceptron Predictor
Column Renamer
Scatter Plot
Add comment
Scatter Plot
Data
Horizontal dimension
observed_mpg
Vertical dimension
mpg
Color dimension
None
Max rows
2500
Plot
Title
Discard
Apply and Execute
Apply
View
Flow Variables
Scatter Plot
Open in new window
mpg
observed_mpg
0
10
20
30
40
50