

### Objetivo:

El objetivo es predecir si un pasajero sobrevivió o no en función de una serie de atributos o características, como la edad, el género, la clase del billete, el puerto de embarque, etc.

### Tipo de problema:

En cuanto al tipo de problema y aprendizaje, este se trata de un problema de aprendizaje supervisado. Algunas razones para considerarlo de esa manera son:

- Tenemos un conjunto de datos etiquetado: El conjunto de datos del Titanic proporciona información sobre si cada pasajero sobrevivió o no, lo que hace que sea un problema supervisado. El modelo se entrena utilizando estos datos etiquetados para aprender a hacer predicciones.
- Objetivo de clasificación: El objetivo principal es predecir si un pasajero sobrevivió (clase positiva) o no (clase negativa). Esto se trata de una tarea de clasificación binaria.

### Entorno de desarrollo:

En este caso optaremos por el uso de la herramienta RapidMiner. Esta nos permitirá llevar a cabo el análisis de datos y la implementación de diversos modelos. Esto con el fin de obtener una solución lo mas acertada posible a nuestro objetivo.

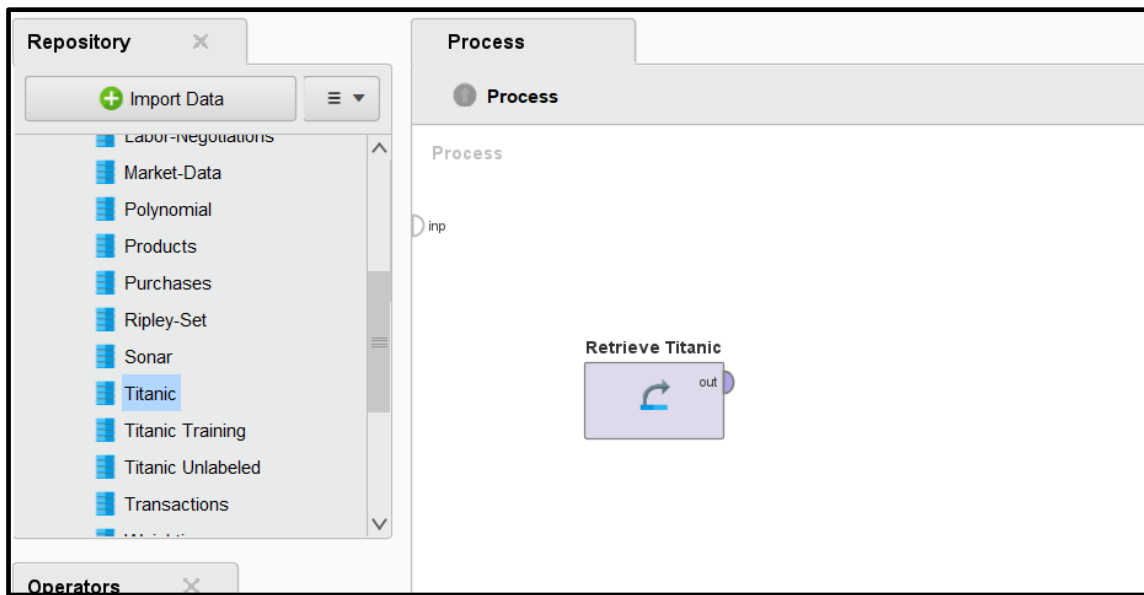
#### Sobre RapidMiner:

RapidMiner es utilizado en una variedad de industrias, desde la salud y las finanzas hasta la fabricación y la investigación académica, para abordar una amplia gama de problemas de análisis de datos y aprendizaje automático. Es una plataforma de código abierto y una suite de software para la ciencia de datos, el aprendizaje automático y el análisis avanzado de datos. Fue desarrollada para facilitar y acelerar el proceso de análisis de datos y la construcción de modelos predictivos, lo que la convierte en una herramienta valiosa para científicos de datos, analistas y profesionales en el campo de la inteligencia empresarial. Cabe destacar, que además de su edición de código abierto, también existe una versión comercial con características adicionales y soporte profesional.

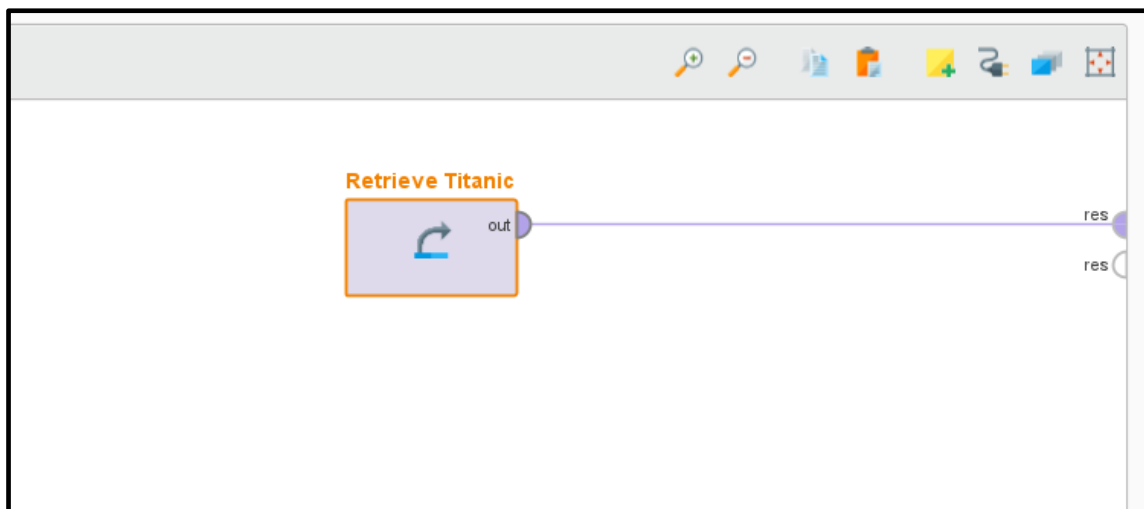
Para más información puede visitar el sitio oficial: <https://rapidminer.com/>

### Análisis del dataset, preparación de datos y selección de atributos:

En esta instancia se trabajará con el dataset Titanic (este viene precargado por defecto en la herramienta RapidMiner).



Lo que se hará es conectar el “out” del dataset a la salida “res” del programa y darle al botón “Run”. De esta manera tendremos una primera visualización de los datos que vamos a utilizar.

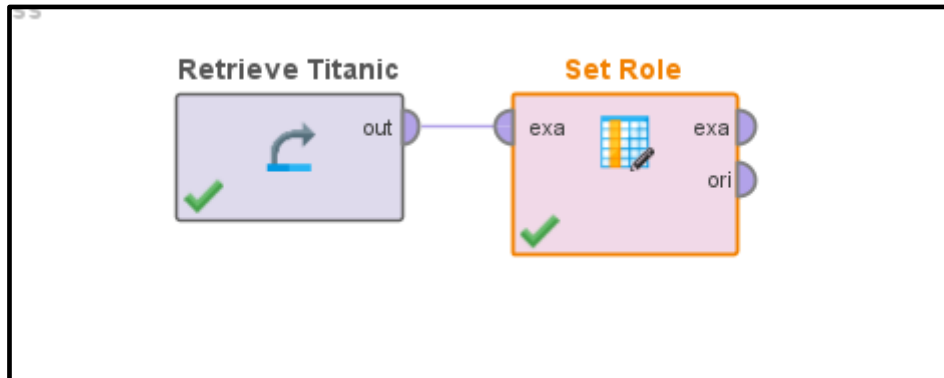


Logramos visualizar así los siguientes atributos:

- Passenger Class
- Name
- Sex
- Age
- No of Siblings or Spouses on Board
- No of Parents or Children on Board
- Ticket Number
- Passenger Fare
- Cabin

- Port of Embarkation
- Lifeboat
- Survived

En base a estos identificaremos al atributo/columna “Survived” como nuestra variable objetivo (label), en la cual nos basaremos para llevar a cabo nuestra investigación. Para eso usaremos el operador Set Role el cual nos permitirá a través de sus parámetros de operador seleccionar la variable “Survived” como “label”.



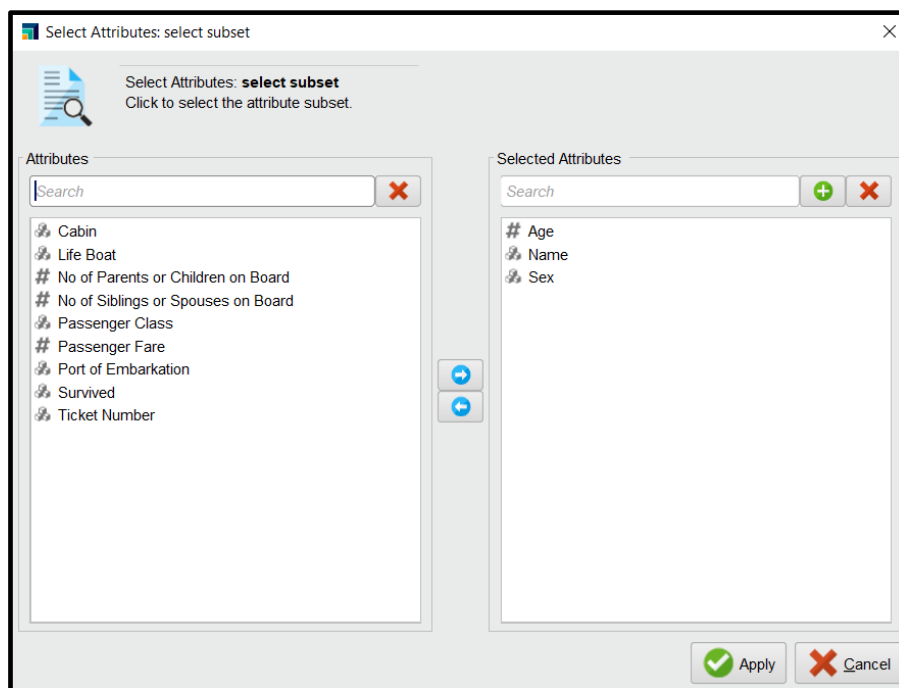
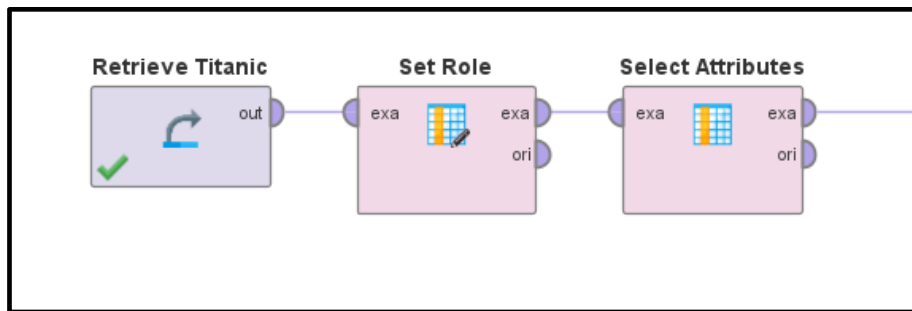
Edit Parameter List: set roles

Edit Parameter List: **set roles**  
This parameter defines new attribute roles.

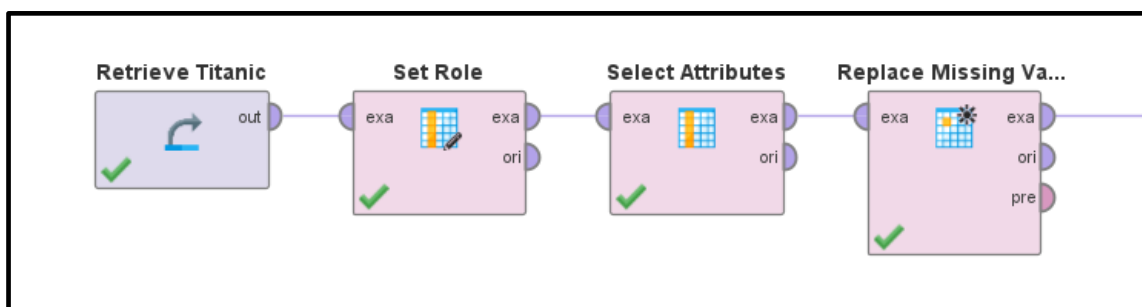
attribute name	target role
Survived	label

Buttons: Add Entry, Remove Entry, Apply, Cancel

Además, excluirémos los parámetros “Name”, “Sex” y “Age”, esto pues dichos datos no son relevantes con relación al objetivo. Para dicha tarea usaremos el operador “Select Attributes”.



El siguiente paso es el tratar con los Missing Values. Estos los podemos eliminar o sustituir (dependiendo que sea más conveniente para nuestro objetivo). En este caso debido al alto volumen de estos optaremos por reemplazarlos, puesto que eliminarlos podría significar un desbalance notorio de datos al momento de entrenar y aplicar un modelo. Para esto se usará el operador “Replace Missing Values” y en su parámetro indicaremos que reemplazaremos con el average.



Parameters

Replace Missing Values

attribute filter type

all

☐

invert selection

☐

include special attributes

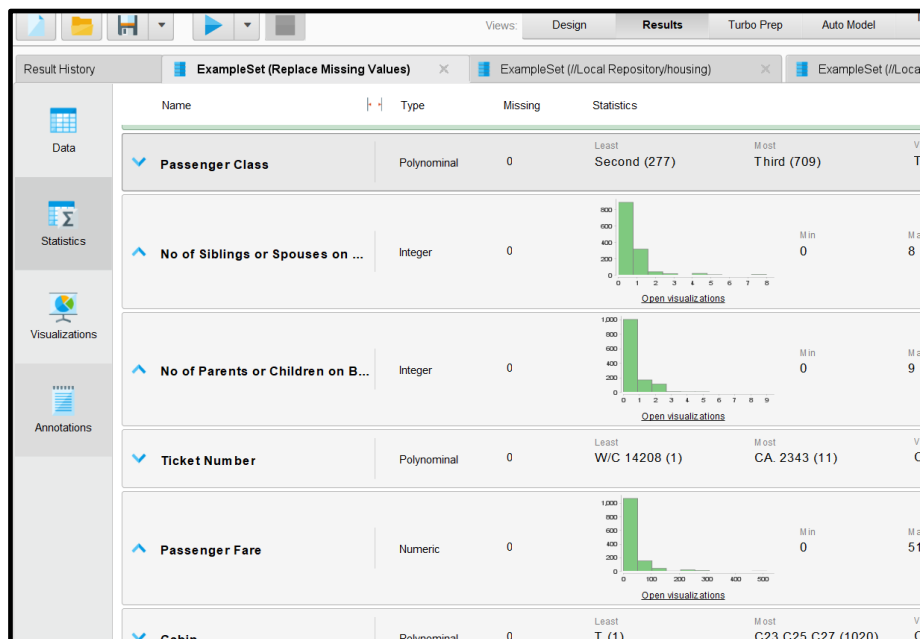
default

average

columns

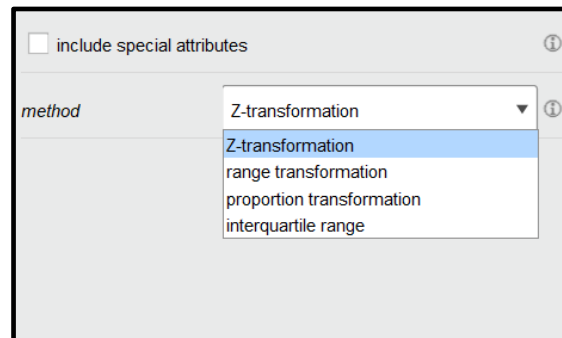
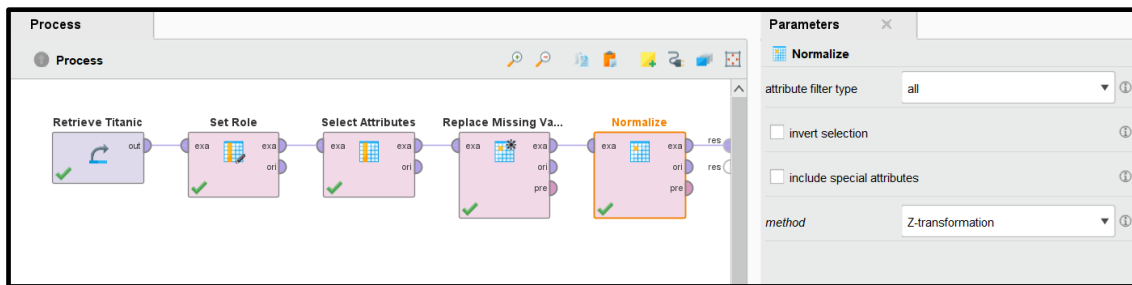
Edit List (0)...

El próximo paso es tener en cuenta la distribución que presentan los datos de carácter cuantitativo (es decir que pueden afectar resultados matemáticos). Lo ideal es que esta se lo mas parecida a una distribución Gaussiana (esto podemos visualizarlo en la ventana “Results” y dando click en la sección “Statistics”).



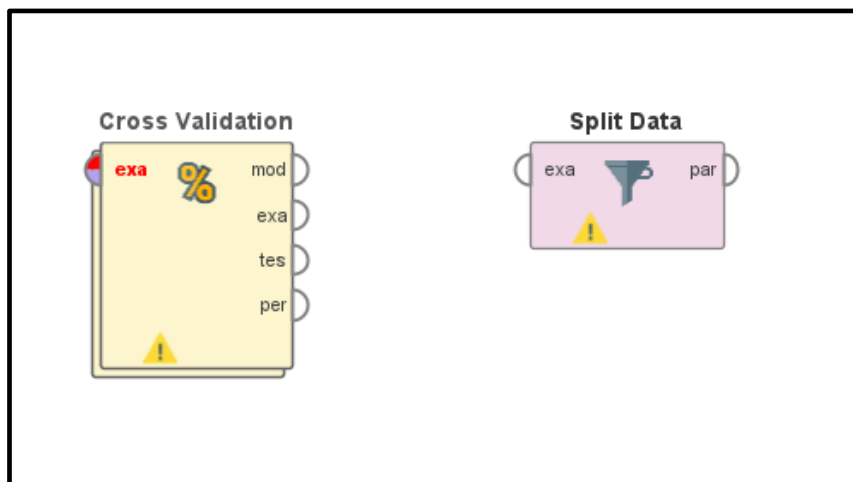
Para solucionar esto usaremos el operador “Normalize” y en sus parámetros elegiremos el método “Z-transformation” (cabe destacar que esta es la configuración estándar, pero podríamos probar con otros métodos en caso de que los resultados no nos parezcan satisfactorios).

Obs: el operador Normalize también nos ayuda a lidiar con posibles outliers. Igualmente si quisiéramos tratar estos mas a fondo podemos usar el operador “Detect Outlier”.



### Descripción del Proceso:

Iniciamos seleccionando el operador que nos ayudará a crear nuestro modelo de predicción. En este caso estudiaremos 2 opciones: “Split Data” y “Cross Validation”.



Cross Validation: Divide el conjunto de datos en múltiples particiones llamadas "pliegues" y, a continuación, entrena y evalúa el modelo múltiples veces, utilizando diferentes combinaciones de pliegues como datos de entrenamiento y prueba (Generalmente se eligen 10 pasos/pliegues iterativos).

Parameters

✕

🔗

Cross Validation

☐

split on batch attribute

ⓘ

☐

leave one out

ⓘ

number of folds

10

ⓘ

sampling type

automatic

▼ ⓘ

☐

use local random seed

ⓘ


☒

enable parallel execution





ⓘ

Split Data: Divide el conjunto de datos en dos partes, una para entrenamiento y otra para pruebas (o validación). El porcentaje en el cual divide los datos lo selecciona el usuario (generalmente se elige una proporción 70-30).

Edit Parameter List: partitions

 Edit Parameter List: **partitions**  
The partitions that should be created.

ratio
0.7
0.3

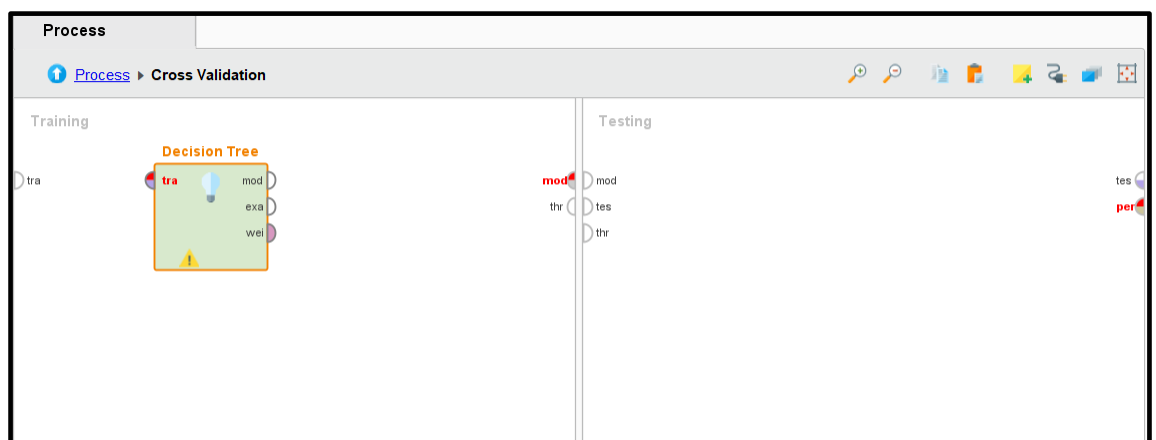
 Add Entry  Remove Entry  OK  Cancel

## Modelo elegido:

Como es sabido en la comunidad del ML, es muy difícil afirmar que determinado algoritmo es indiscutiblemente el mejor para evaluar un determinado problema. Debido a esto en este caso usaremos 3 (esto podría expandirse tanto como uno quiera), Árboles de Decisión, K-NN y otro. La idea es comparar los resultados obtenidos y así decidir cual es el enfoque (dentro de los seleccionados) que mejor funciona en relación con nuestro objetivo.

## Árboles de Decisión:

- Cross Validation (Configuraremos las dos etapas de un modelo predictivo, entrenamiento y prueba):



Configuramos los parámetros del operador:

Parameters	
Decision Tree	
criterion	gain_ratio
maximal depth	20
<input checked="" type="checkbox"/> apply pruning	
confidence	0.25
<input checked="" type="checkbox"/> apply prepruning	
minimal gain	0.1
minimal leaf size	2
minimal size for split	4
number of prepruning alternatives	3



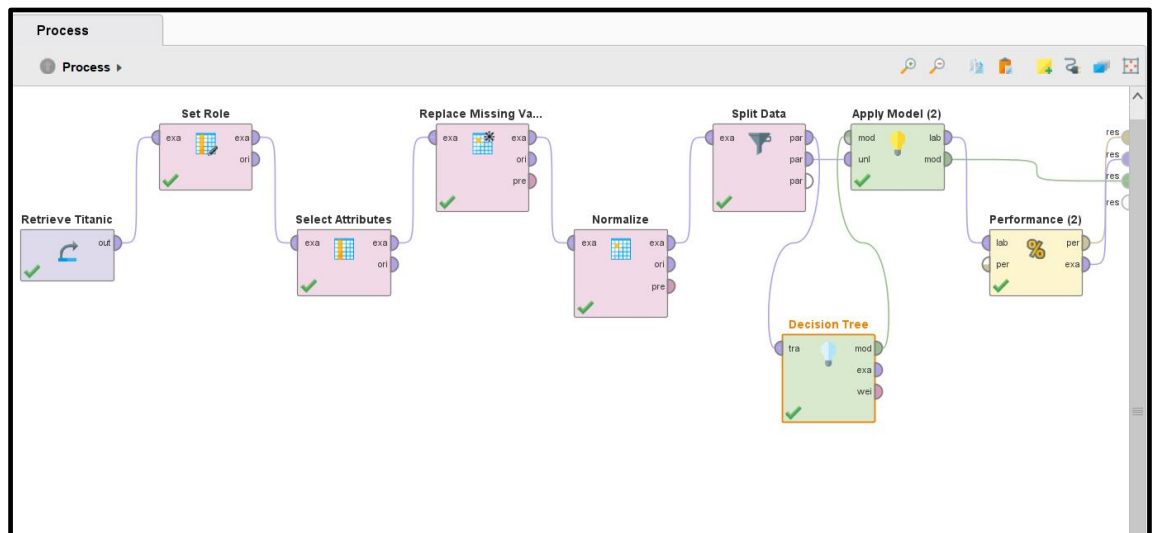
The screenshot displays the Orange3 data mining environment. At the top, there is a 'Process' menu and a toolbar with icons for file operations, search, and workflow management. The main workspace shows a workflow titled 'Process' with the following steps:

- Retrieve Titanic**: A process that takes an input (inp) and outputs data (out).
- Set Role**: A process that sets the role of variables (exa, ori) in the dataset.
- Select Attributes**: A process that selects specific attributes (exa, ori) for analysis.
- Replace Missing Values**: A process that replaces missing values in the dataset.
- Normalize**: A process that normalizes the data.
- Cross Validation**: A process that performs cross-validation on the model, resulting in multiple output metrics (res).

The workflow is visualized as a sequence of connected boxes, with data flowing from left to right. The 'Cross Validation' process has several output ports labeled 'res', indicating different performance metrics.

accuracy: 62.11% +/- 0.37% (micro average: 62.11%)			
	true Yes	true No	class precision
pred. Yes	4	0	100.00%
pred. No	496	809	61.99%
class recall	0.80%	100.00%	

- Split Data:



accuracy: 71.50%

	true Yes	true No	class precision
pred. Yes	60	22	73.17%
pred. No	90	221	71.06%
class recall	40.00%	90.95%	

**K-NN (repetimos el proceso aplicado para Arboles de decisi3n):**

En ambos casos seteamos el n3mero de k vecinos en 10:

Parameters

k-NN

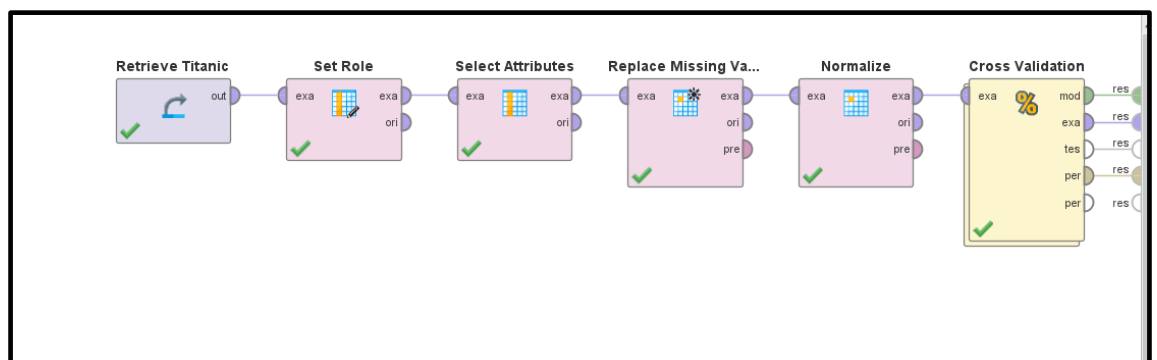
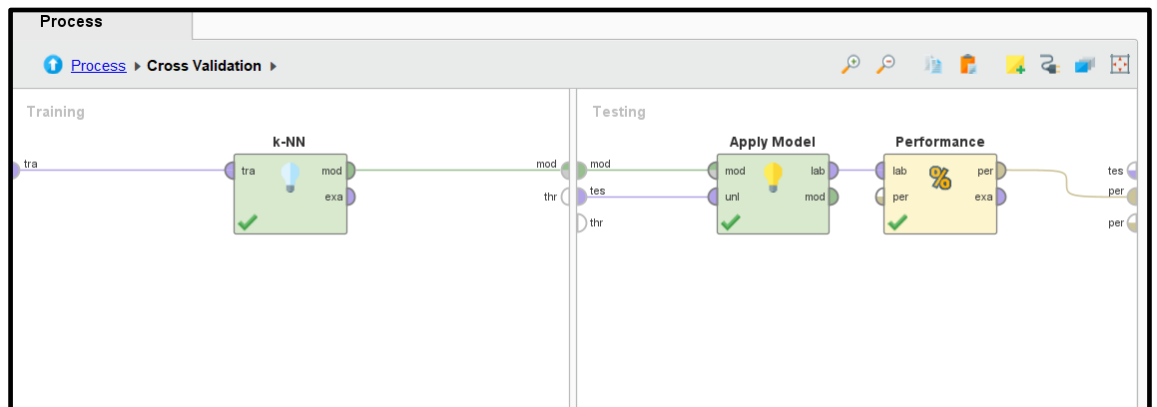
k
10

☒ weighted vote

measure types
MixedMeasures

mixed measure
MixedEuclideanDistance

- Cross Validation:

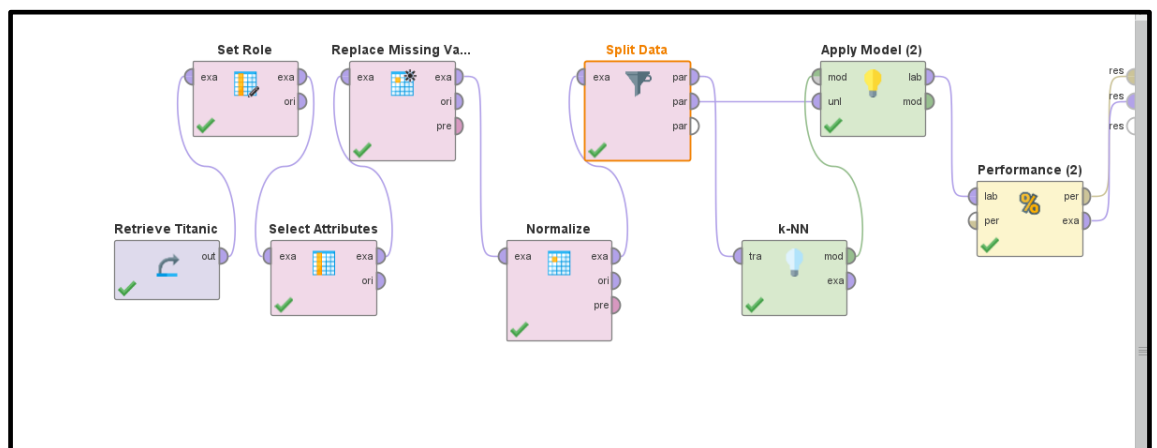


Resultados:

accuracy: 87.32% +/- 2.59% (micro average: 87.32%)

	true Yes	true No	class precision
pred. Yes	358	24	93.72%
pred. No	142	785	84.68%
class recall	71.60%	97.03%	

- Split Data:



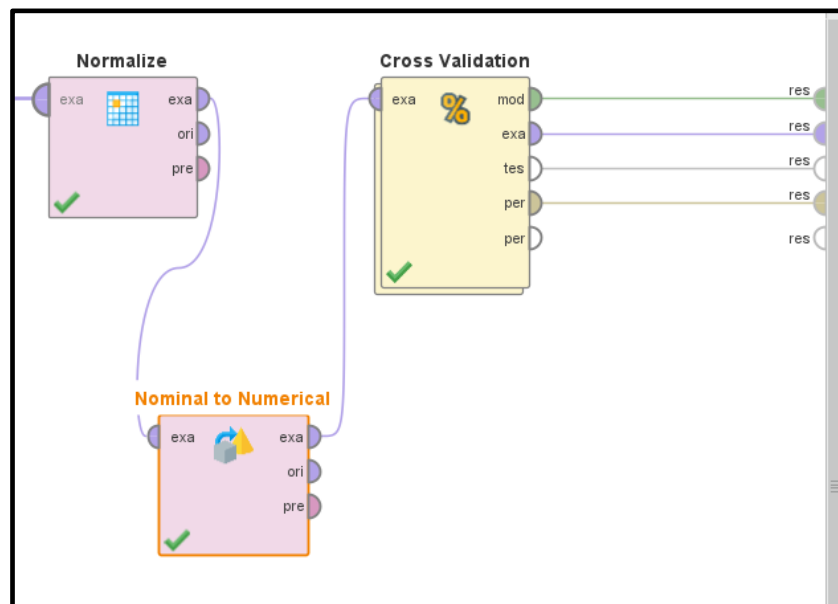
Resultados:

accuracy: 86.77%			
	true Yes	true No	class precision
pred. Yes	107	9	92.24%
pred. No	43	234	84.48%
class recall	71.33%	96.30%	

## Máquinas de Soporte Vectorial (SVM):

- Cross Validation:

Primero usaremos el operador “Nominal to Numerical”, esto pues el algoritmo SVM no soporta valores nominales.



**Parameters**

**Nominal to Numerical**

attribute filter type all

☐ invert selection

☐ include special attributes

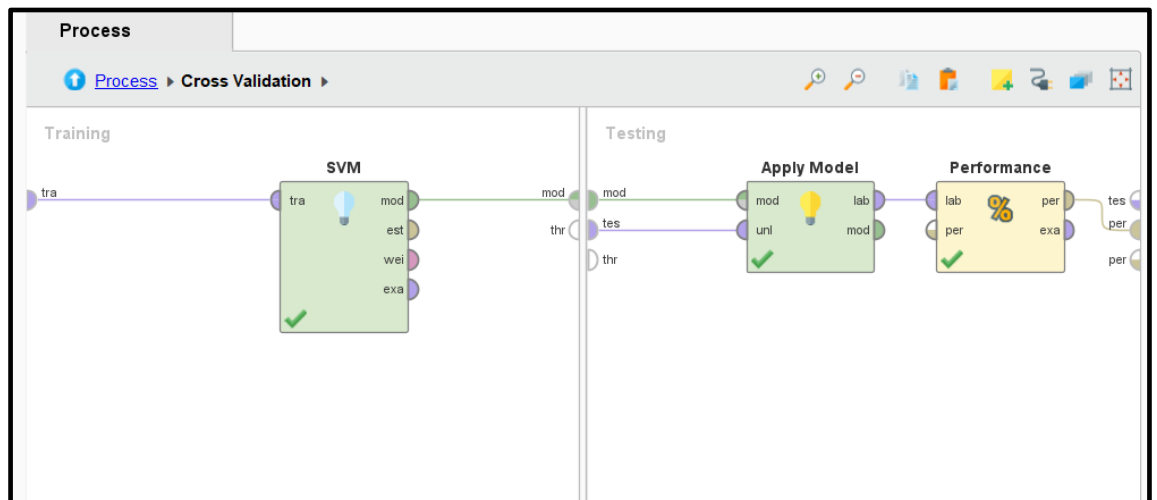
coding type dummy coding

☐ use comparison groups

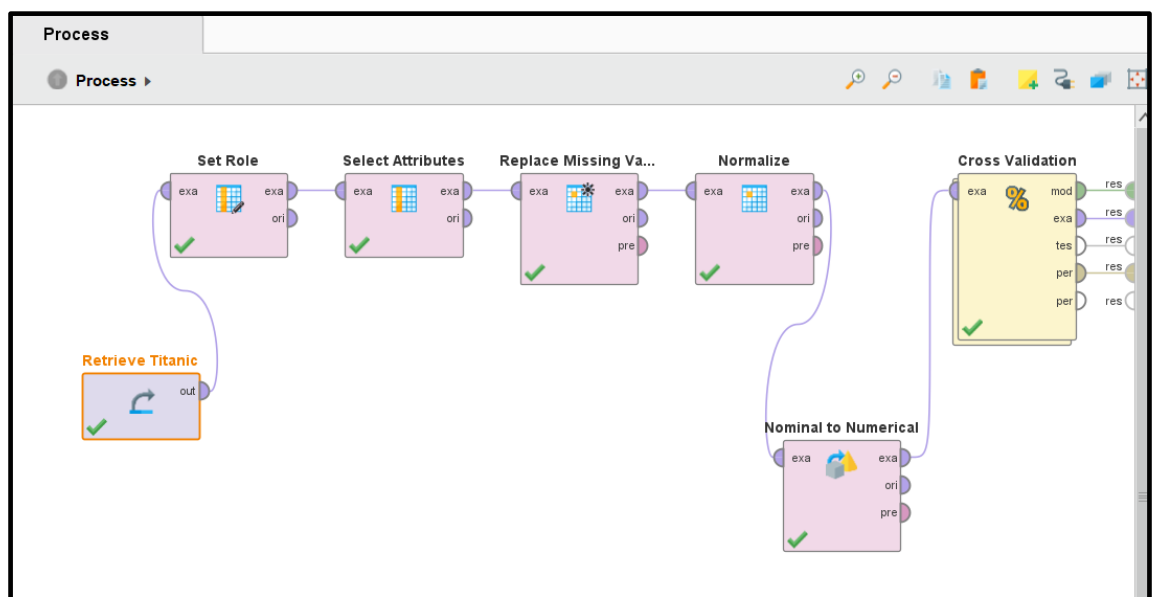
unexpected value handling all 0 and warning

☐ use underscore in name

Configuramos el Cross Validation:



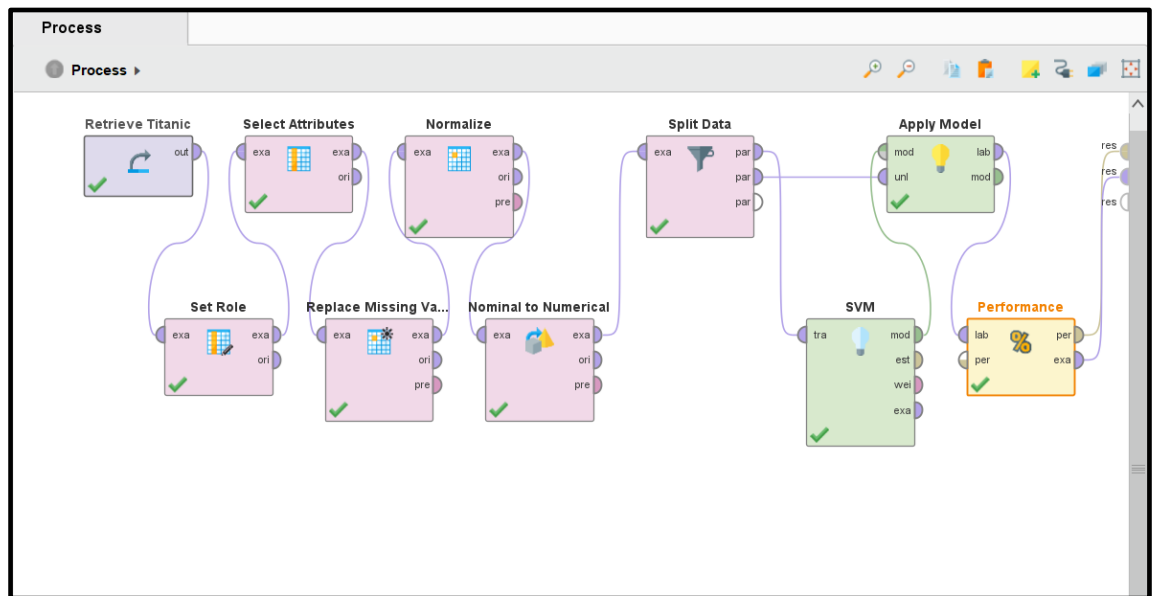
Conectamos las salidas y damos "Run":



Resultados:

accuracy: 89.23% +/- 2.95% (micro average: 89.23%)			
	true Yes	true No	class precision
pred. Yes	384	25	93.89%
pred. No	116	784	87.11%
class recall	76.80%	96.91%	

- Split Data:



Resultados:

accuracy: 87.53%			
	true Yes	true No	class precision
pred. Yes	111	10	91.74%
pred. No	39	233	85.66%
class recall	74.00%	95.88%	

### Análisis de Resultados:

En base a los resultados obtenidos, podemos concluir en una primera instancia que el uso del modelo SVM junto con el enfoque de Cross Validation es el modelo más efectivo en relación con nuestro objetivo.

## PerformanceVector

PerformanceVector:

accuracy: 89.23% +/- 2.95% (micro average: 89.23%)

ConfusionMatrix:

True:	Yes	No
Yes:	384	25
No:	116	784

#### Posibles mejoras por evaluar:

- Probar con más modelos.
- Probar con diversos valores de K.
- Probar diferentes parámetros en el operador "Decision Tree".
- Probar otro método de normalización.
- Probar con diferentes proporciones en el operador "Split Data".
- Probar diversos valores de folds en el operador "Cross Validation".