

UNIVERSIDAD DE BUENOS AIRES

FACULTAD DE INGENIERÍA

Introducción a los sistemas inteligentes (75.50)

Profesor Adjunto: Ochoa María

Ayudante: Vazquez Diego



Tutorial Weka

Introducción a Weka	pág. 2
Cargando datos	pág. 7
Formato arff	pág. 11
Comprender los datos	pág. 15
Eliminación de atributos	pág. 18
Aplicar filtros	pág. 19
Clasificadores	pág. 21
J48	pág. 23
Multilayer Perceptron (MLP)	pág. 29
Clustering	pág. 37
Redes Neuronales con Clustering	pág. 44

Introducción a Weka

Weka es un software integral que le permite preprocesar los grandes datos, aplicar diferentes algoritmos de aprendizaje automático en los grandes datos y comparar varios resultados. Este software facilita el trabajo con big data y entrena una máquina usando algoritmos de aprendizaje automático.

La base de cualquier aplicación de Machine Learning son los datos, no solo unos pocos datos, sino una gran cantidad de datos que se denomina Big Data en la terminología actual.

Para entrenar y para analizar big data, debe tener varias consideraciones sobre los datos:

- Los datos deben estar limpios.
- No debe contener valores nulos.

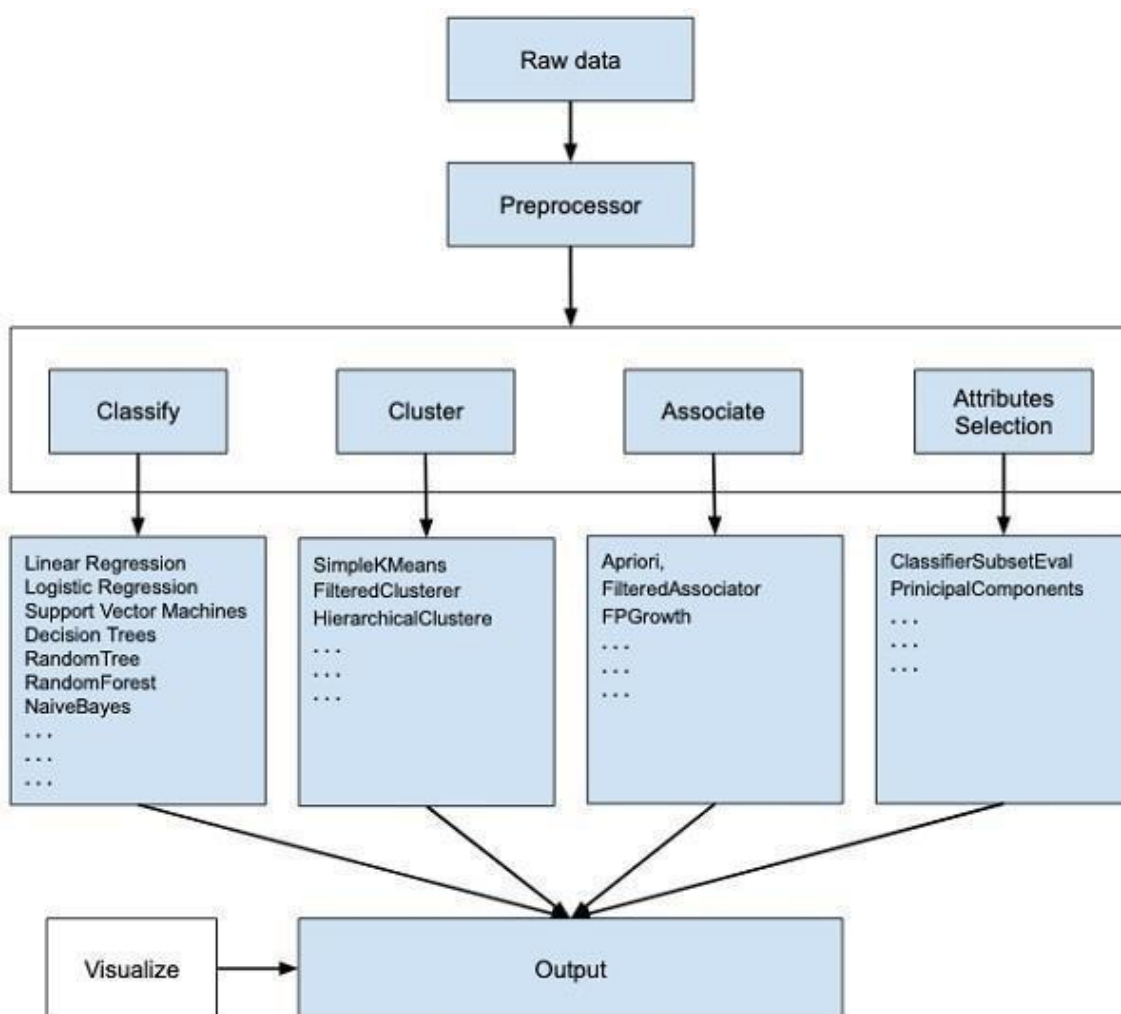
Además, no todas las columnas de la tabla de datos serían útiles para el tipo de análisis que intenta lograr. Las columnas de datos irrelevantes o "características", tal como se denominan en la terminología de aprendizaje automático, deben eliminarse antes de que los datos se introduzcan en un algoritmo de aprendizaje automático.

En resumen, su big data necesita mucho preprocesamiento antes de que pueda usarse para Machine Learning. Una vez que los datos estén listos, aplicaría varios algoritmos de aprendizaje automático, como clasificación, regresión, agrupación, etc., para resolver el problema.

El tipo de algoritmos que aplica se basa en gran medida en su conocimiento del dominio. Incluso dentro del mismo tipo, por ejemplo clasificación, hay varios algoritmos disponibles. Es posible que desee probar los diferentes algoritmos en la misma clase para crear un modelo de aprendizaje automático eficiente. Mientras lo hace, preferiría la visualización de los datos procesados y, por lo tanto, también necesita herramientas de visualización.

Weka como es código abierto proporciona herramientas para el preprocesamiento de datos, la implementación de varios algoritmos de aprendizaje automático y herramientas de visualización para que pueda desarrollar técnicas de aprendizaje automático y aplicarlas a problemas de minería de datos del mundo real.

Lo que ofrece WEKA se resume en el siguiente diagrama:



Primero, comenzará con los datos sin procesar recopilados en el campo. Estos datos pueden contener varios valores nulos y campos irrelevantes. Utiliza las herramientas de preprocesamiento de datos proporcionadas en WEKA para limpiar los datos.

Luego, guardaría los datos preprocesados en su almacenamiento local para aplicar algoritmos ML.

A continuación, según el tipo de modelo de ML que intente desarrollar, seleccionaría una de las opciones, como Clasificar, Agrupar o Asociar. La Selección de atributos permite la selección automática de características para crear un conjunto de datos reducido.

Tener en cuenta que, en cada categoría, WEKA proporciona la implementación de varios algoritmos. Seleccionaría un algoritmo de su elección, establecería los parámetros deseados y lo ejecutaría en el conjunto de datos.

Entonces, WEKA le daría la salida estadística del procesamiento del modelo. Le proporciona una herramienta de visualización para inspeccionar los datos.

Los diversos modelos se pueden aplicar en el mismo conjunto de datos y comparar los resultados de diferentes modelos y seleccionar el que mejor se adapte a su propósito.

Para la instalación: sitio web oficial de WEKA (https://waikato.github.io/weka-wiki/downloading_weka/) y descargue el archivo de instalación. WEKA admite la instalación en Windows, Mac OS X y Linux. Solo necesita seguir las instrucciones en esta página para instalar WEKA para su sistema operativo.

Se iniciará la aplicación WEKA GUI Chooser y verá la siguiente pantalla:



La aplicación le permite ejecutar cinco tipos diferentes de aplicaciones que se enumeran aquí:

Explorer

Experimenter

KnowledgeFlow

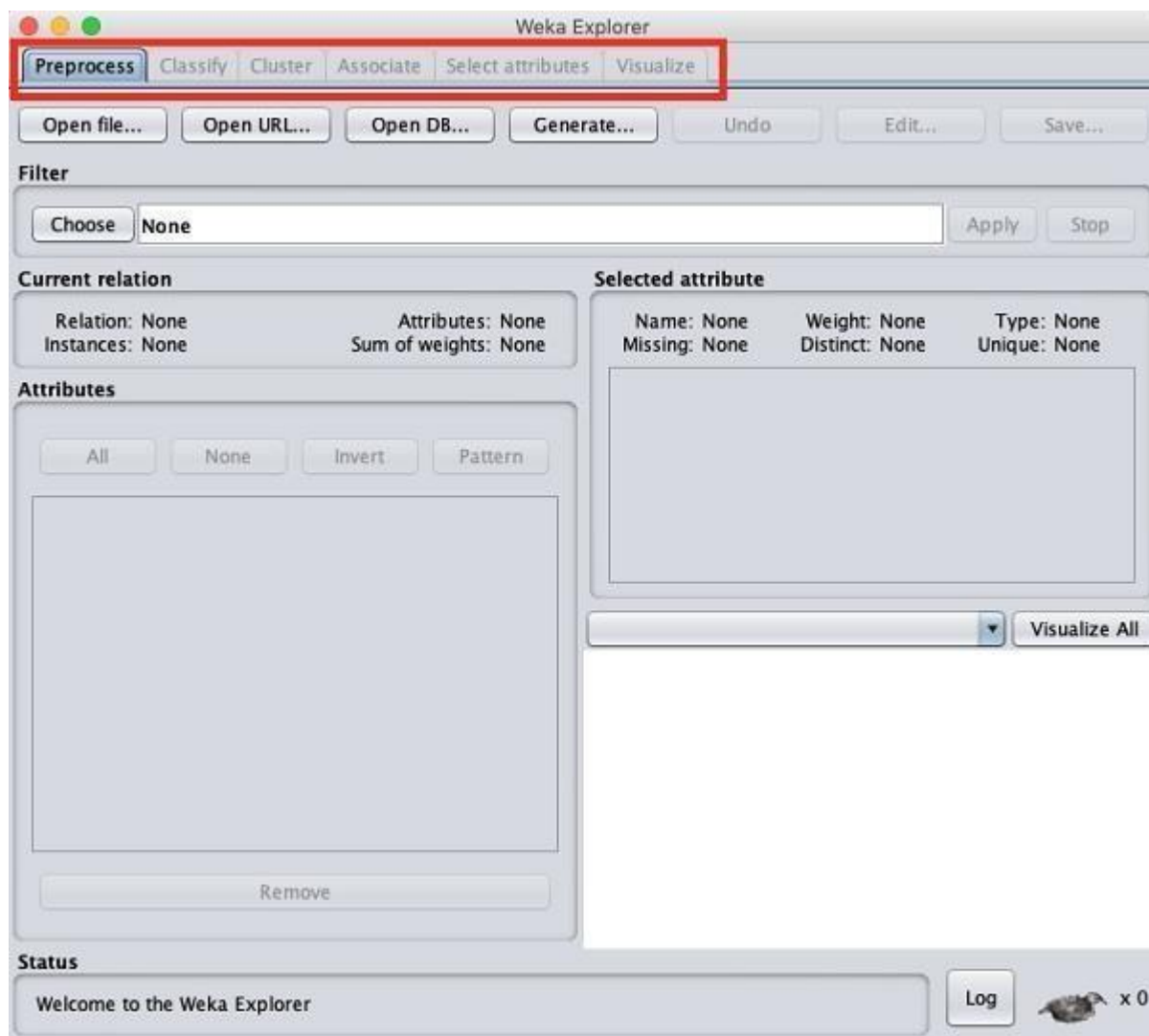
Workbench

Simple CLI

Usaremos Explorer

Hagamos un vistazo a varias funcionalidades que proporciona el explorador para trabajar con big data.

Cuando hace clic en el botón Explorador en el selector de aplicaciones, se abre la siguiente pantalla:



En la parte superior, verá varias pestañas que se enumeran aquí:

- Preprocess
- Classify
- Cluster
- Associate
- Select Attributes
- Visualize

Debajo de estas pestañas, hay varios algoritmos de aprendizaje automático preimplementados. Veamos ahora cada uno de ellos en detalle.

Pestaña Preprocess

Inicialmente, cuando abre el explorador, solo está habilitada la pestaña Preprocesar. El primer paso en el aprendizaje automático es preprocesar los datos. Por lo tanto, en la opción Preprocesar, seleccionará el archivo de datos, lo procesará y lo adaptará para aplicar los diversos algoritmos de aprendizaje automático.

Pestaña Classify

La pestaña Clasificar le proporciona varios algoritmos de aprendizaje automático para la clasificación de sus datos. Para enumerar algunos, puede aplicar algoritmos como Regresión lineal, Regresión logística, Máquinas de vectores de soporte, Árboles de decisión, RandomTree, RandomForest, NaiveBayes, etc. La lista es muy exhaustiva y proporciona algoritmos de aprendizaje automático supervisados y no supervisados.

Pestaña Cluster

En la pestaña Clúster, se proporcionan varios algoritmos de agrupamiento, como SimpleKMeans, FilteredClusterer, HierarchicalClusterer, SelfOrganizationMap, etc.

Pestaña Associate

En la pestaña Asociar, encontrará Apriori, FilteredAssociator y FPGrowth.

Pestaña Select Attributes

Seleccionar atributos le permite selecciones de características basadas en varios algoritmos como ClassifierSubsetEval, PrincipalComponents, etc.

Pestaña Visualize

Por último, la opción Visualizar le permite visualizar sus datos procesados para su análisis.

Como notó, WEKA proporciona varios algoritmos listos para usar para probar y construir sus aplicaciones de aprendizaje automático. Para usar WEKA de manera efectiva, debe tener un conocimiento sólido de estos algoritmos, cómo funcionan, cuál elegir bajo qué circunstancias, qué buscar en su salida procesada, etc. En resumen, debe tener una base sólida en aprendizaje automático para usar WEKA de manera efectiva en la creación de sus aplicaciones.

Cargando datos

Comenzamos con la primera pestaña que usa para preprocesar los datos. Esto es común a todos los algoritmos que aplicaría a sus datos para construir el modelo y es un paso común para todas las operaciones posteriores en WEKA.

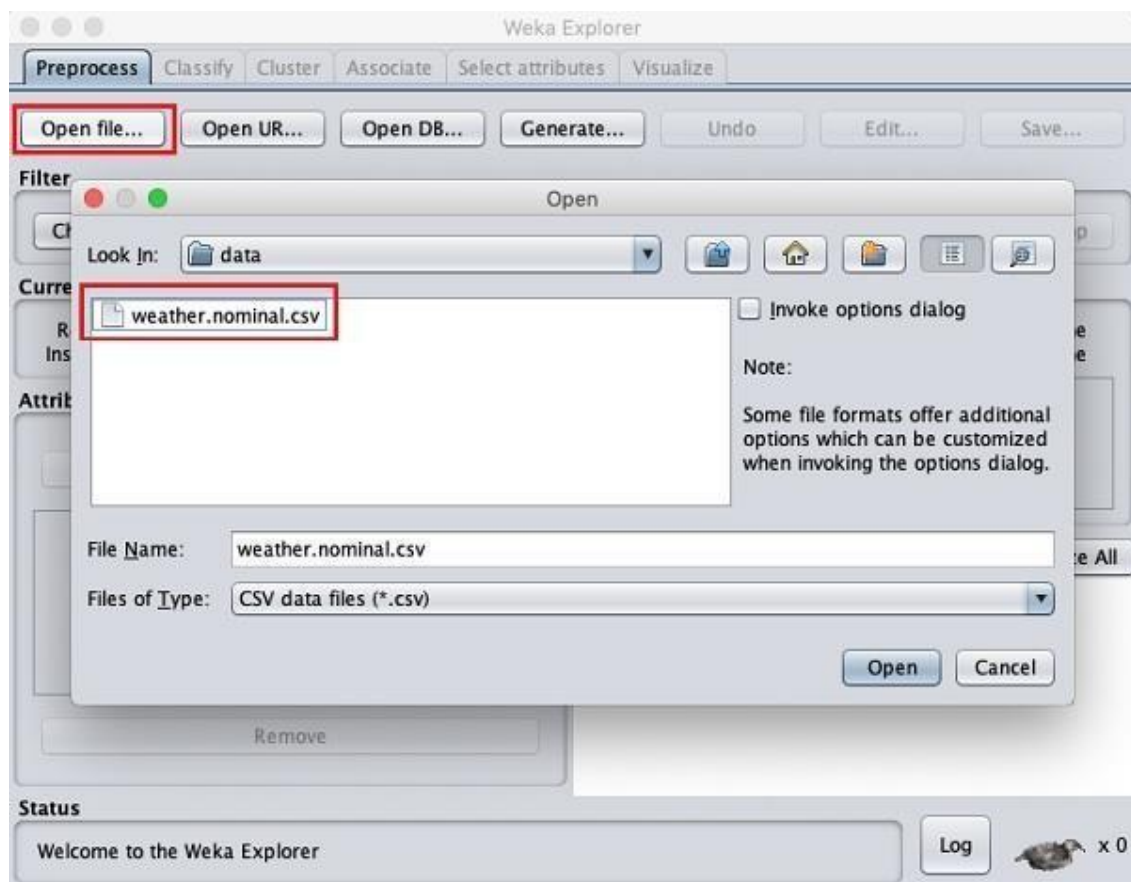
Para que un algoritmo de aprendizaje automático brinde una precisión aceptable, es importante que primero limpie sus datos. Esto se debe a que los datos sin procesar recopilados del campo pueden contener valores nulos, columnas irrelevantes, etc.

Primero cargamos el archivo de datos en el explorador WEKA. Los datos se pueden cargar desde las siguientes fuentes:

- Sistema de archivos locales
- Web
- Base de datos

Carga de datos desde el sistema de archivos local

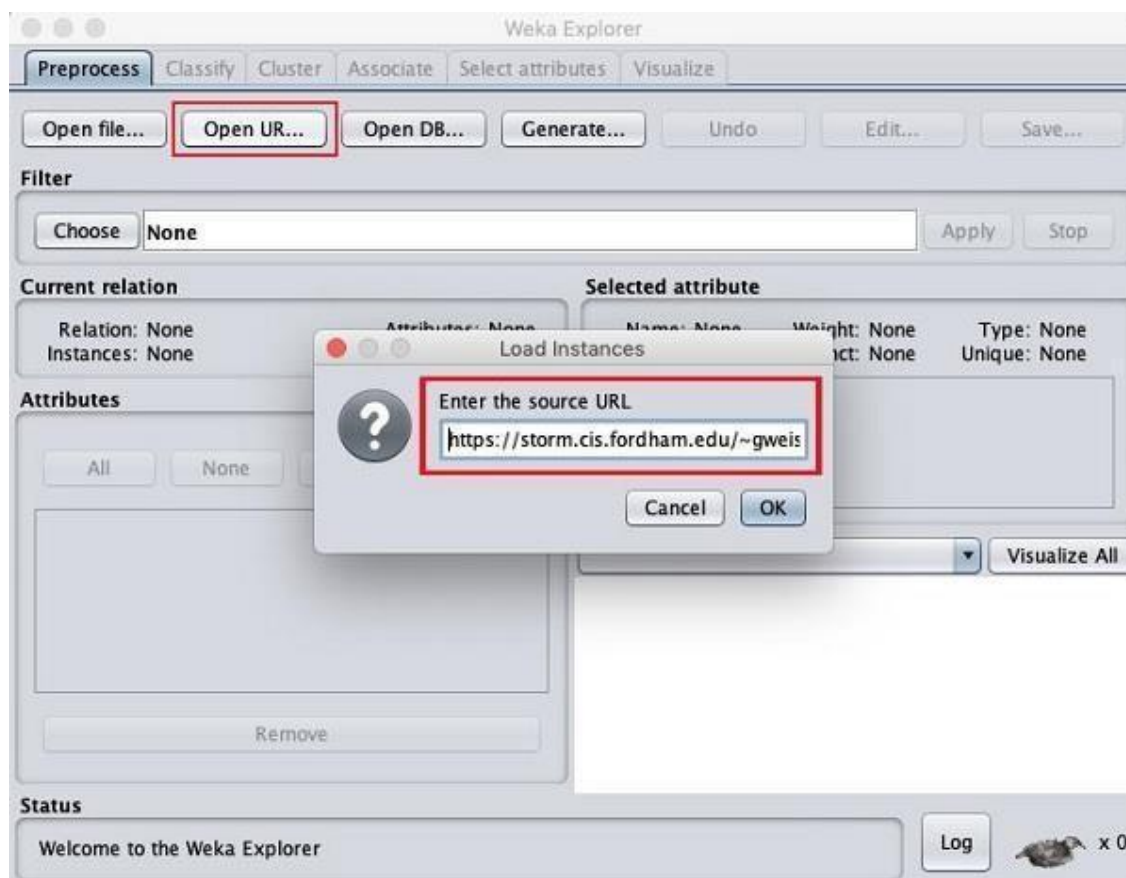
Justo debajo de las pestañas de Machine Learning, encontrará los siguientes tres botones:



Ahora, navegue a la carpeta donde se almacenan sus archivos de datos. La instalación de WEKA presenta muchas bases de datos de muestra para que experimente. Estos están disponibles en la carpeta de datos de la instalación de WEKA.

Cargando datos desde la web

Una vez que haga clic en el botón Abrir URL..., podrá ver una ventana de la siguiente manera:



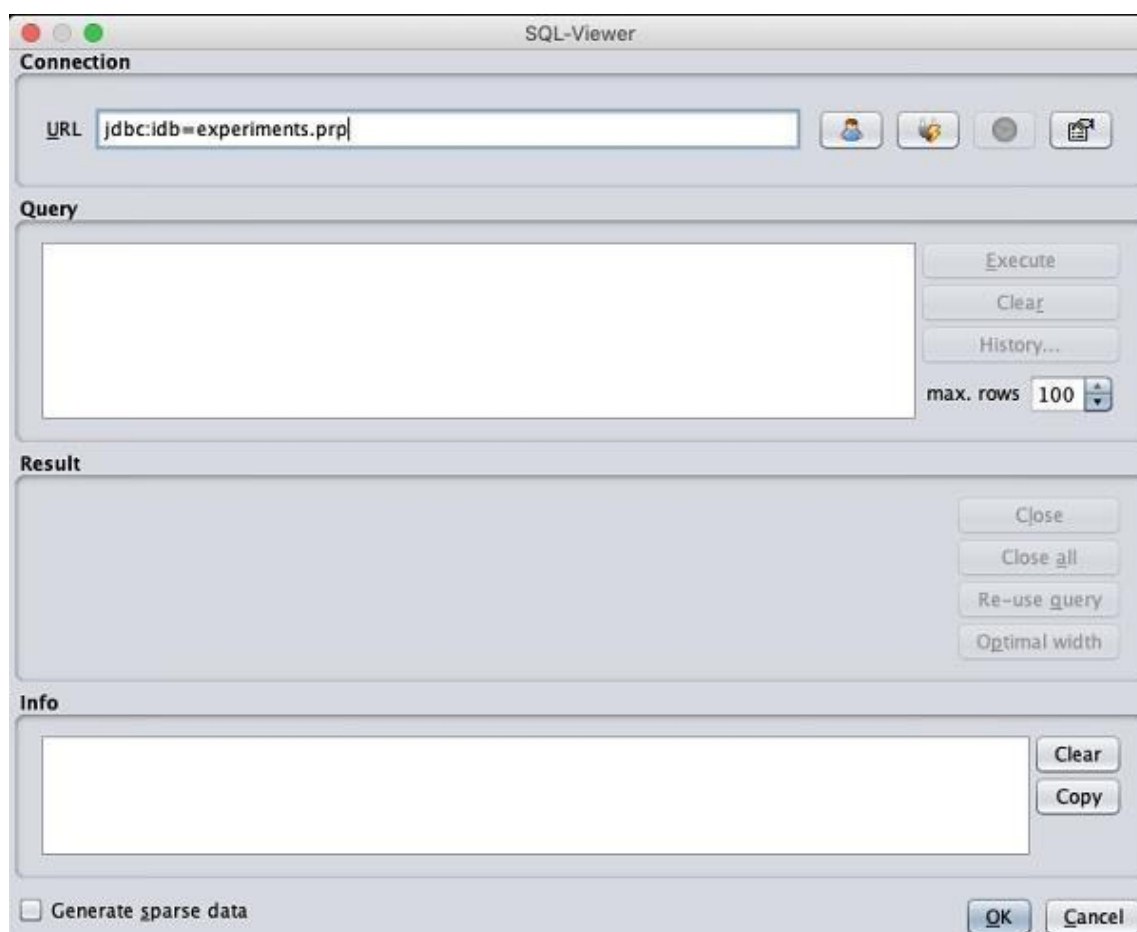
Abriremos el archivo desde una URL pública. Escriba la siguiente URL en el cuadro emergente:

<https://storm.cis.fordham.edu/~gweiss/data-mining/weka-data/weather.nominal.arff>

Puede especificar cualquier otra URL donde se almacenen sus datos. El Explorador cargará los datos del sitio remoto en su entorno.

Cargando datos desde la base de datos

Una vez que haga clic en el botón Open DB ..., puede ver una ventana de la siguiente manera:

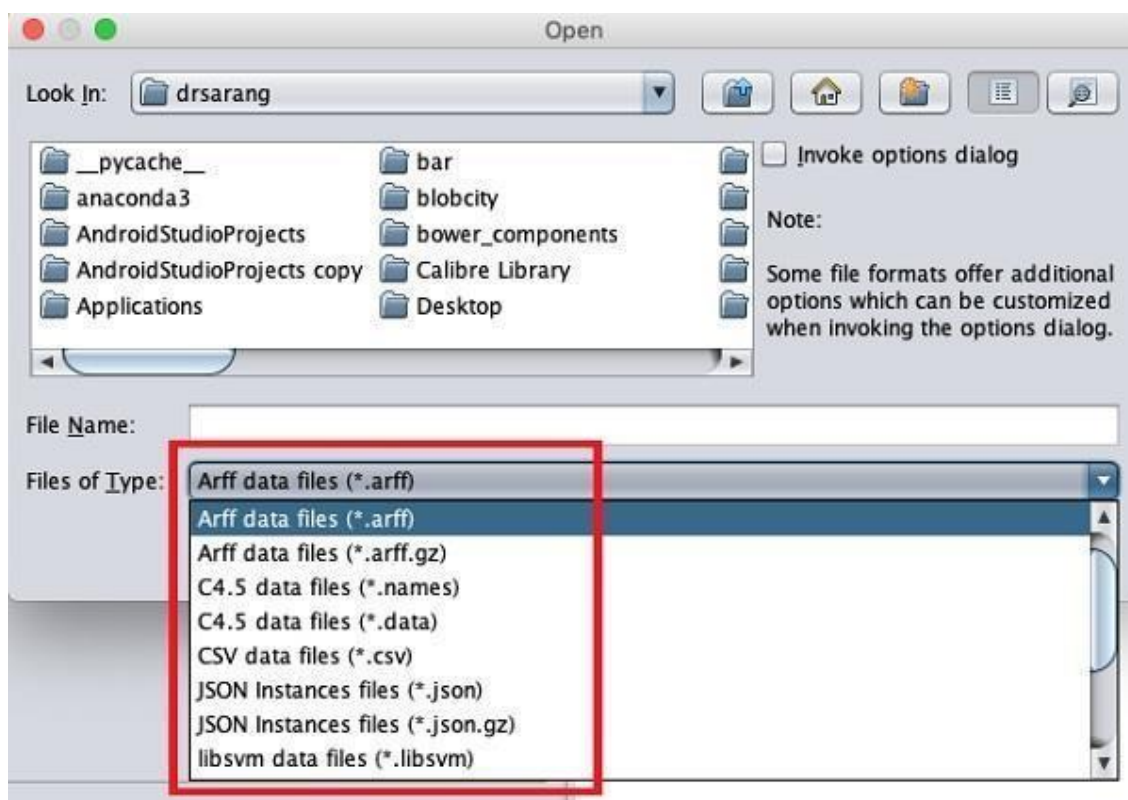


Configure la cadena de conexión a su base de datos, configure la consulta para la selección de datos, procese la consulta y cargue los registros seleccionados en WEKA.

Formato de archivos

WEKA admite una gran cantidad de formatos de archivo para los datos. Aquí está la lista completa

- arff
- arff.gz
- bsi
- csv
- dat
- data
- json
- json.gz
- libsvm
- names
- xrf
- xrf.gz



Como notará, admite varios formatos, incluidos CSV y JSON. El tipo de archivo predeterminado es arff.

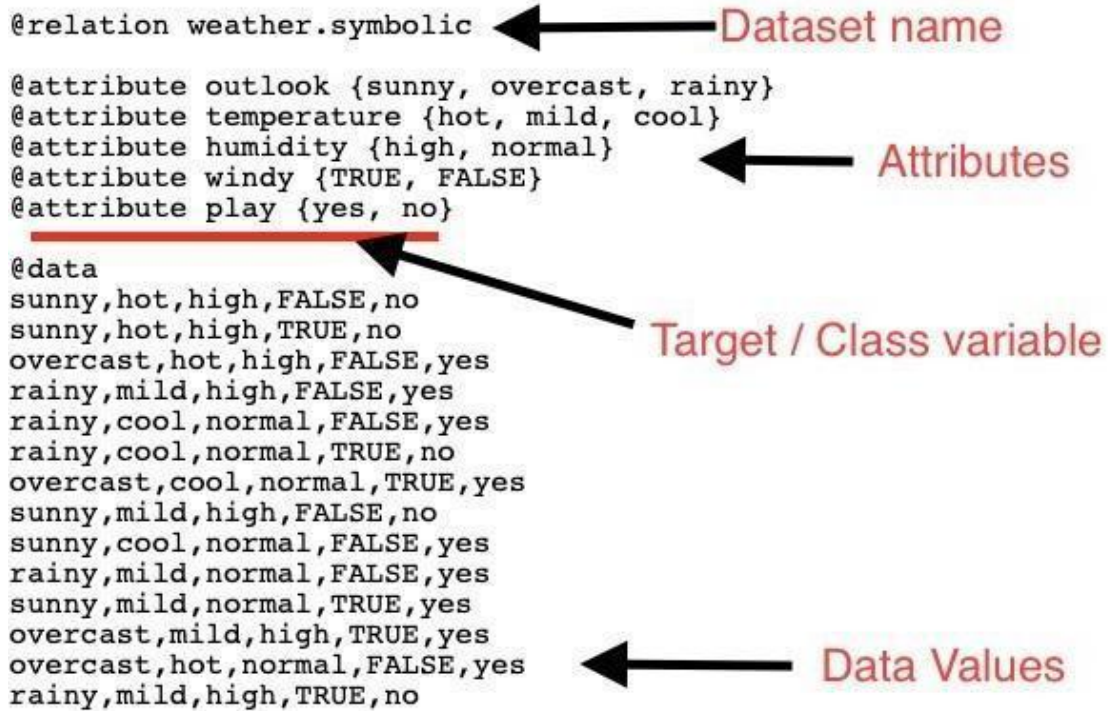
Formato arff

Un archivo arff contiene dos secciones: encabezado y datos.

El encabezado describe los tipos de atributos.

La sección de datos contiene una lista de datos separados por comas.

Como ejemplo del formato arff:



The diagram shows an ARFF file format example with annotations. The text is as follows:

```
@relation weather.symbolic
@attribute outlook {sunny, overcast, rainy}
@attribute temperature {hot, mild, cool}
@attribute humidity {high, normal}
@attribute windy {TRUE, FALSE}
@attribute play {yes, no}

@data
sunny,hot,high,FALSE,no
sunny,hot,high,TRUE,no
overcast,hot,high,FALSE,yes
rainy,mild,high,FALSE,yes
rainy,cool,normal,FALSE,yes
rainy,cool,normal,TRUE,no
overcast,cool,normal,TRUE,yes
sunny,mild,high,FALSE,no
sunny,cool,normal,FALSE,yes
rainy,mild,normal,FALSE,yes
sunny,mild,normal,TRUE,yes
overcast,mild,high,TRUE,yes
overcast,hot,normal,FALSE,yes
rainy,mild,high,TRUE,no
```

Annotations with arrows pointing to the corresponding parts of the file:

- Dataset name** points to `@relation weather.symbolic`.
- Attributes** points to the list of attribute definitions: `@attribute outlook {sunny, overcast, rainy}`, `@attribute temperature {hot, mild, cool}`, `@attribute humidity {high, normal}`, `@attribute windy {TRUE, FALSE}`, and `@attribute play {yes, no}`.
- Target / Class variable** points to the `play` attribute in the data rows, which is the last value in each row.
- Data Values** points to the list of data rows starting with `@data`.

De la captura de pantalla, puede inferir los siguientes puntos:

La etiqueta `@relation` define el nombre de la base de datos.

La etiqueta `@attribute` define los atributos.

La etiqueta `@data` inicia la lista de filas de datos, cada una de las cuales contiene campos separados por comas.

- Los atributos pueden tomar valores nominales como en el caso de la perspectiva que se muestra aquí:

```
@attribute outlook (sunny, overcast, rainy)
```

- Los atributos pueden tomar valores reales como en este caso:

```
@attribute temperature real
```

- También puede establecer una variable Target o Class llamada play como se muestra aquí

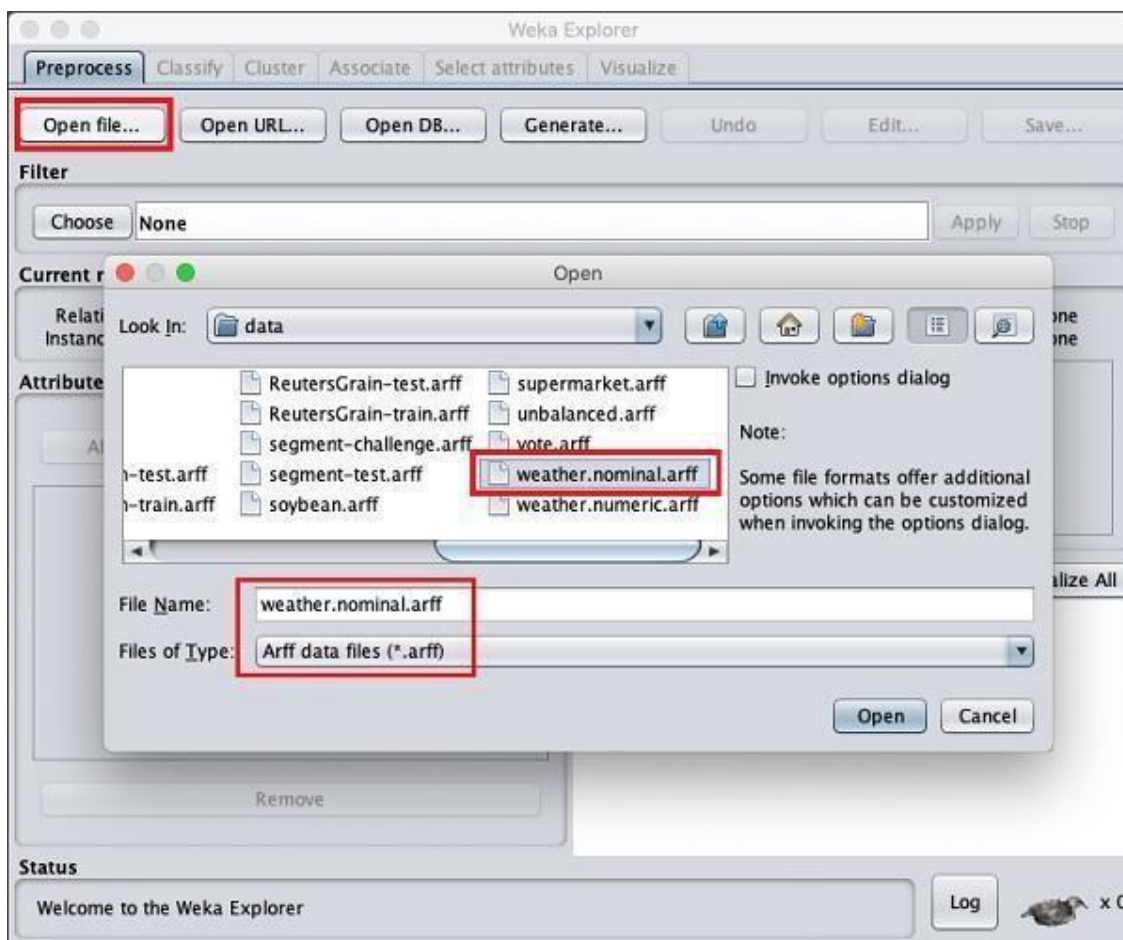
```
@attribute play (yes, no)
```

El Target asume dos valores nominales sí o no.

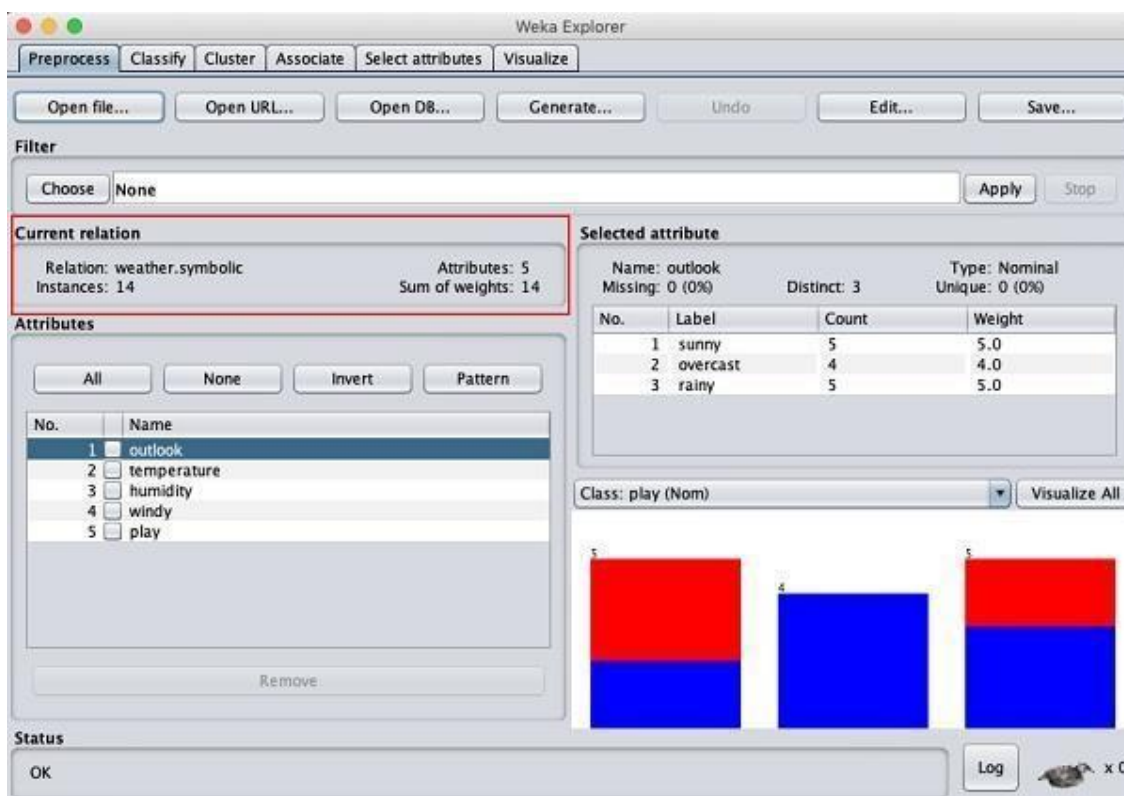
Los datos que se recopilan del campo contienen muchas cosas no deseadas que conducen a un análisis erróneo. Por ejemplo, los datos pueden contener campos nulos, pueden contener columnas que son irrelevantes para el análisis actual, etc. Por lo tanto, los datos deben ser preprocesados para cumplir con los requisitos del tipo de análisis que busca. Esto es lo que se hace en el módulo de preprocesamiento.

Para demostrar las funciones disponibles en el preprocesamiento, utilizaremos la base de datos meteorológica que se proporciona.

Con la opción Abrir archivo... en la etiqueta Preprocesar, seleccione el archivo weather-nominal.arff.



Cuando abre el archivo, su pantalla se ve como se muestra aquí:



Esta pantalla nos dice varias cosas sobre los datos cargados, que se analizan más adelante.

Comprender los datos

Veamos primero la subventana de relación actual resaltada. Muestra el nombre de la base de datos que está cargada actualmente. Puede inferir dos puntos de esta subventana:

Hay 14 instancias: el número de filas en la tabla.

La tabla contiene 5 atributos: los campos, que se analizan en las próximas secciones.

En el lado izquierdo, observe la subventana Atributos que muestra los diversos campos en la base de datos.

The screenshot shows the Weka Explorer window with the 'weather.symbolic' dataset loaded. The 'Attributes' list on the left is highlighted with a red box, showing 5 attributes: outlook, temperature, humidity, windy, and play. The 'Selected attribute' panel on the right shows details for the 'outlook' attribute, including its type (Nominal), missing values (0), distinct values (3), and a table of counts and weights for each value.

No.	Label	Count	Weight
1	sunny	5	5.0
2	overcast	4	4.0
3	rainy	5	5.0

The 'Class' is set to 'play (Nom)' and the 'Visualize All' button is visible. The 'Status' bar at the bottom shows 'OK' and a 'Log' button.

La base de datos meteorológica contiene cinco campos: outlook, temperature, humidity, windy and play. Cuando selecciona un atributo de esta lista haciendo clic en él, se muestran más detalles sobre el atributo en el lado derecho.

Seleccionemos primero el atributo de temperature. Al hacer clic en él, verá la siguiente pantalla:

En la subventana Atributo seleccionado, puede observar lo siguiente:

Se muestran el nombre y el tipo del atributo.

El tipo para el atributo de temperature es Nominal.

El número de valores perdidos es cero.

Hay tres valores distintos sin valor único.

La tabla debajo de esta información muestra los valores nominales para este campo como hot, mild y cool.

The screenshot shows the Weka Explorer interface. The 'Selected attribute' window is highlighted, showing the following information:

Name: temperature
Missing: 0 (0%)
Distinct: 3
Type: Nominal
Unique: 0 (0%)

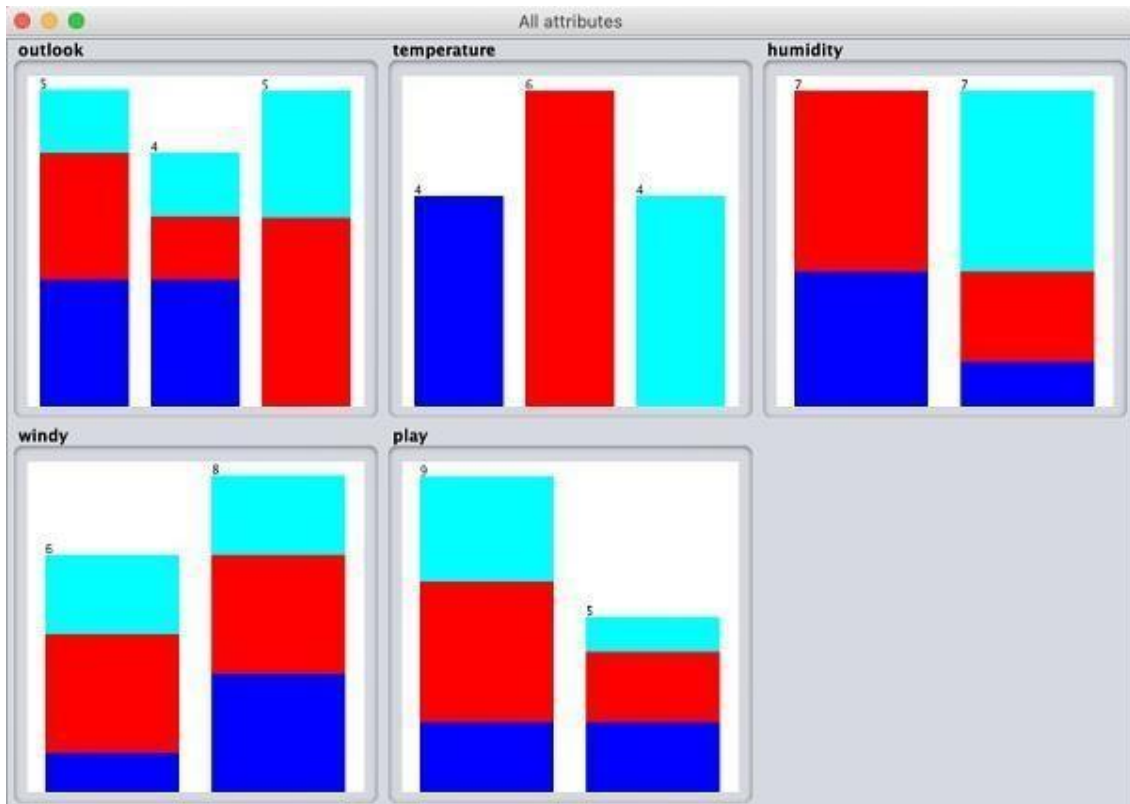
No.	Label	Count	Weight
1	hot	4	4.0
2	mild	6	6.0
3	cool	4	4.0

Below the table, there is a bar chart showing the distribution of the 'play' class for each temperature value. The chart has three bars, each with a red top section and a blue bottom section. The counts for the red sections are 4, 6, and 4, corresponding to the 'hot', 'mild', and 'cool' values respectively.

También muestra el conteo y el peso en términos de un porcentaje para cada valor nominal.

En la parte inferior de la ventana, verá la representación visual de los valores de clase (play en este caso).

Si hace clic en el botón Visualizar todo, podrá ver todas las funciones en una sola ventana como se muestra aquí:



Eliminación de atributos

Muchas veces, los datos que desea utilizar para la creación de modelos vienen con muchos campos irrelevantes. Por ejemplo, la base de datos del cliente puede contener su número de móvil que es irrelevante para analizar su calificación crediticia.

Para eliminar los atributos, selecciónelos y haga clic en el botón Eliminar en la parte inferior.



Los atributos seleccionados se eliminarían de la base de datos. Después de preprocesar por completo los datos, puede **guardarlos para la creación de modelos**.

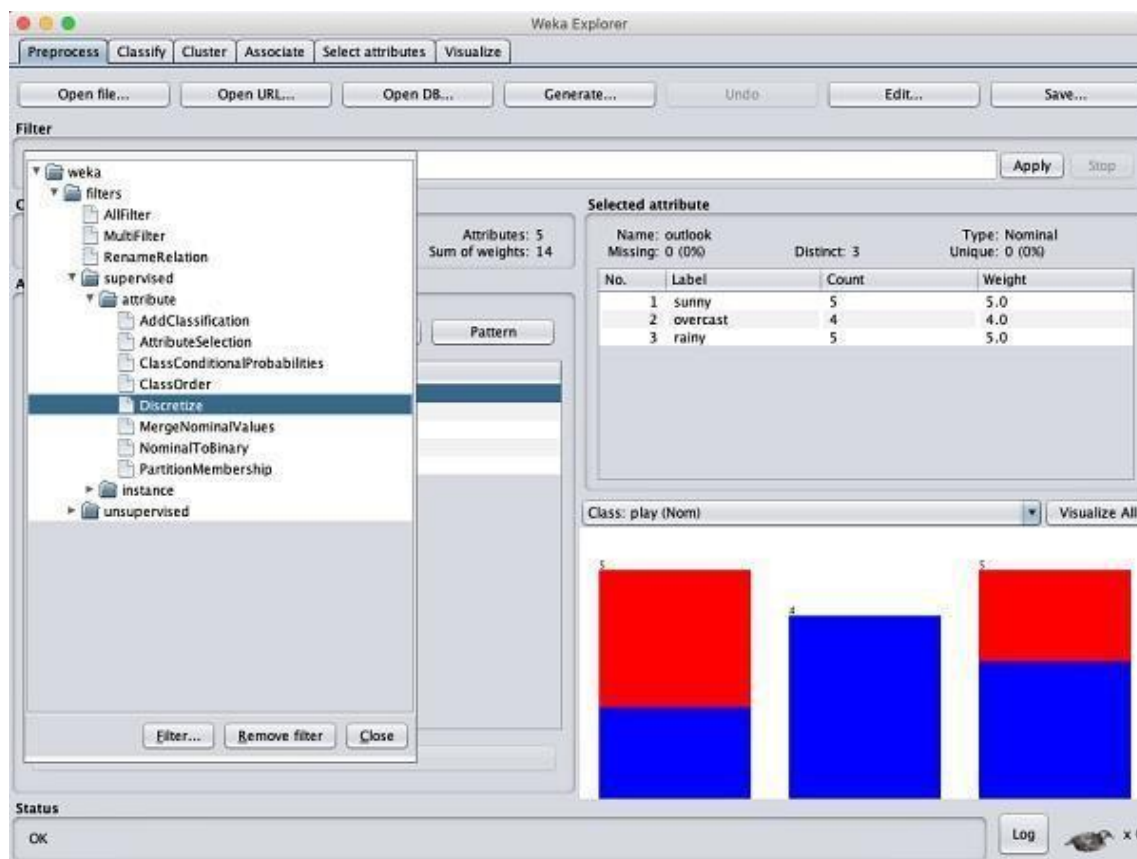
A continuación, aprenderemos a preprocesar los datos aplicando filtros a estos datos.

Aplicar filtros

Algunas de las técnicas de aprendizaje automático, como la minería de reglas de asociación, requieren datos categóricos. Para ilustrar el uso de filtros, usaremos la base de datos weather-numeric.arff que contiene dos atributos numéricos: temperatura y humedad.

Los convertiremos en nominales aplicando un filtro en nuestros datos sin procesar. Haga clic en el botón Elegir en la subventana Filtro y seleccione el siguiente filtro:

weka→filters→supervised→attribute→Discretize



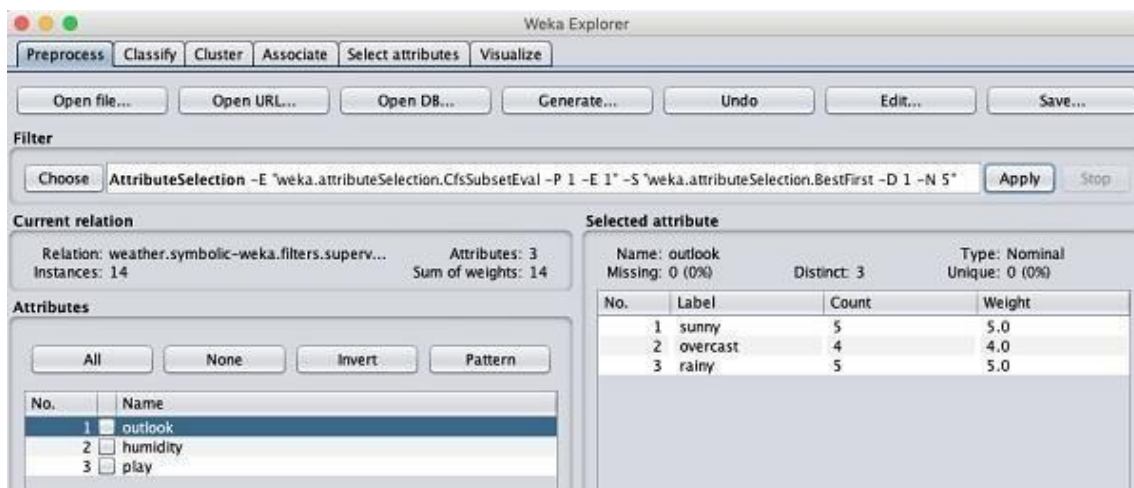
Haga clic en el botón Aplicar y examine el atributo de temperatura y/o humid. Notará que estos han cambiado de tipos numéricos a nominales.

Name: temperature		Type: Nominal	
Missing: 0 (0%)		Distinct: 1	
		Unique: 0 (0%)	
No.	Label	Count	Weight
1	'All'	14	14.0

Echemos un vistazo a otro filtro ahora. Suponga que desea seleccionar los mejores atributos para decidir play. Seleccione y aplique el siguiente filtro:

weka→filters→supervised→attribute→AttributeSelection

Notará que elimina los atributos de temperature y humid de la base de datos.



Otros tipos de filtros podemos encontrar en

weka→filters→unsupervised→attribute

donde podremos encontrar varios filtros que nos pueden ser útiles para el preproceso.

Una vez que esté satisfecho con el preprocesamiento de sus datos, guárdalos haciendo clic en el botón Guardar. Utilizará este archivo guardado para la construcción de modelos.

Clasificadores

Muchas aplicaciones de aprendizaje automático están relacionadas con la clasificación. Por ejemplo, puede clasificar un tumor como maligno o benigno. Es posible que desee decidir si jugar un juego al aire libre dependiendo de las condiciones climáticas.

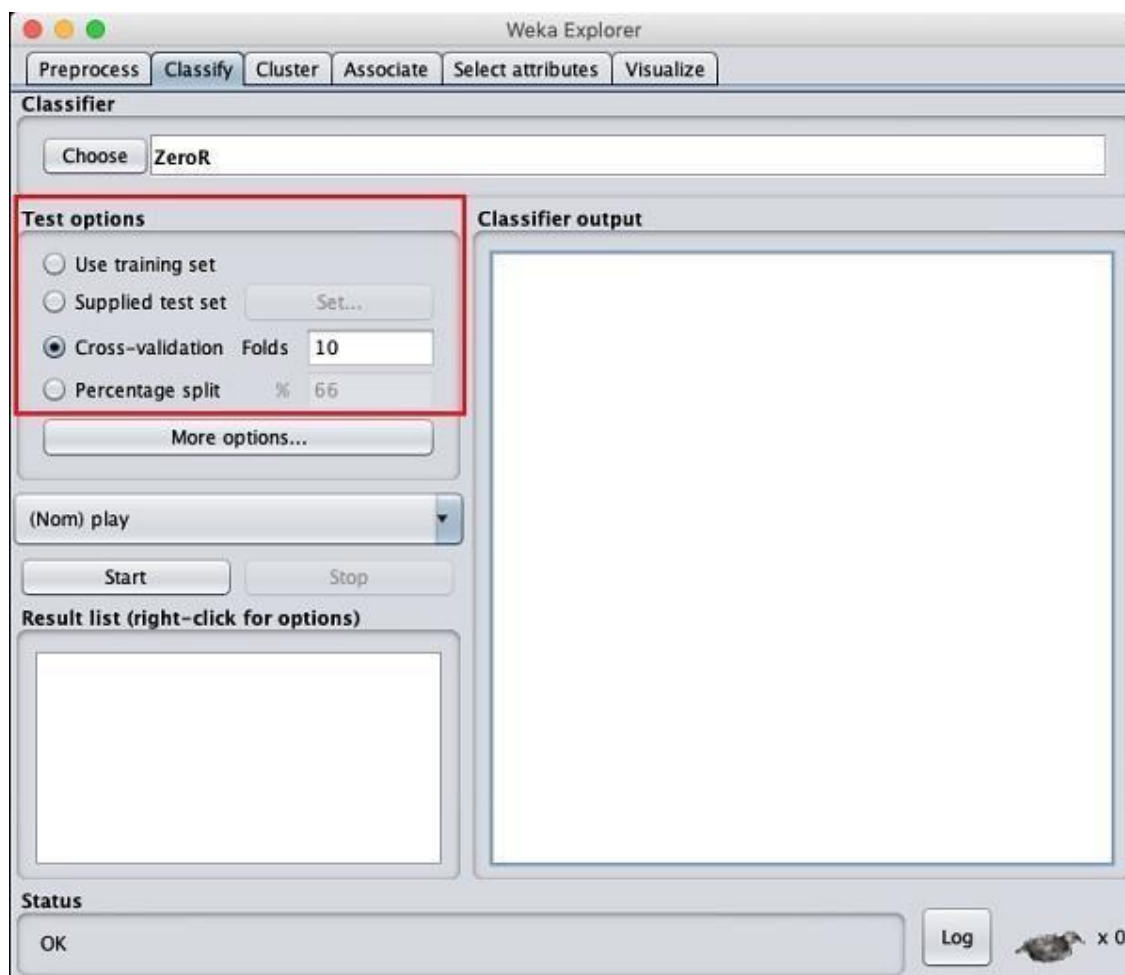
Generalmente, esta decisión depende de varias características/condiciones del clima.

Por lo tanto, es posible que prefiera usar un clasificador de árbol para tomar su decisión de jugar o no.

Vamos a generar un árbol clasificador de este tipo con datos meteorológicos para decidir las condiciones de juego.

Configuración de datos de prueba

Usaremos el archivo de datos meteorológicos preprocesados. Abra el archivo guardado usando la opción Abrir archivo... en la pestaña Preprocesar, haga clic en la pestaña Clasificar y verá la siguiente pantalla:

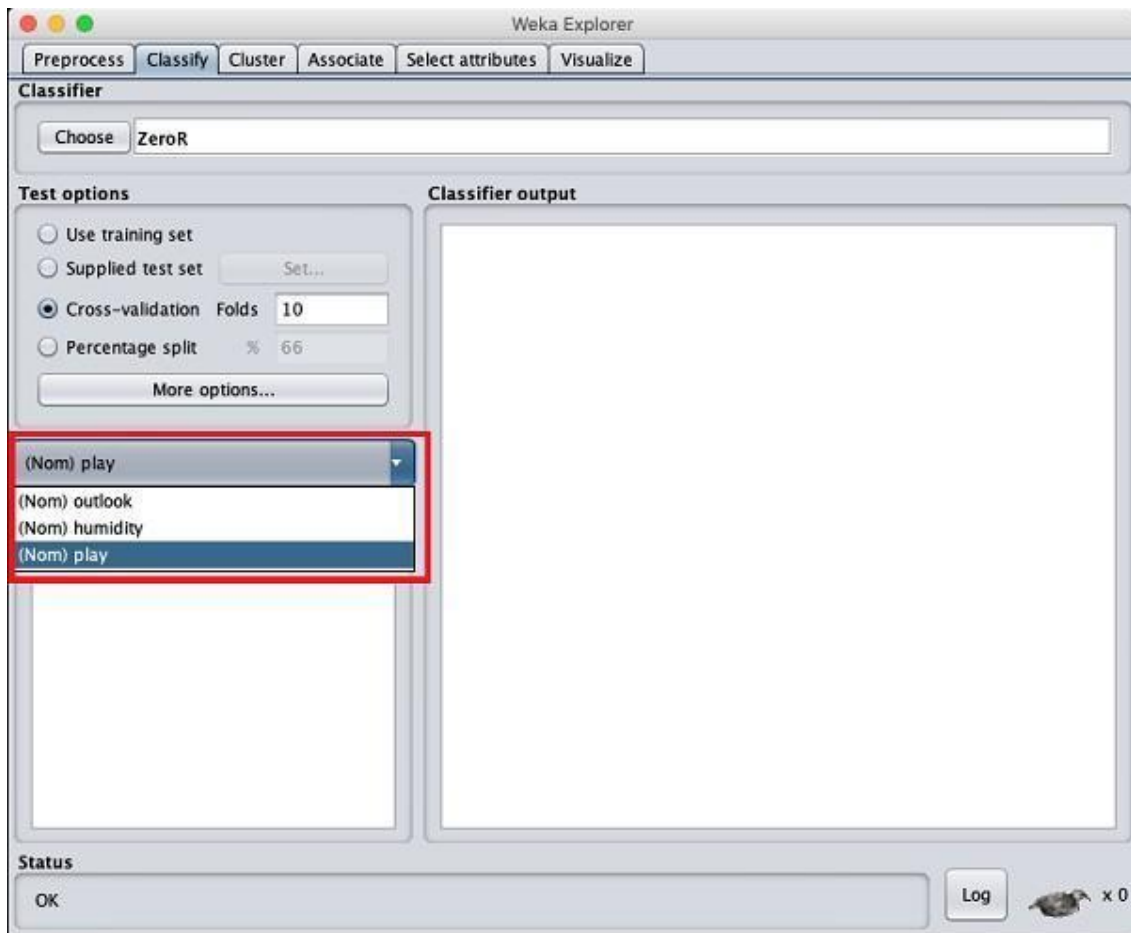


Antes de conocer los clasificadores disponibles, examinemos las opciones de prueba. Notará cuatro opciones de prueba que se enumeran a continuación:

- Training set
- Supplied test set
- Cross-validation
- Percentage split

A menos que tenga su propio conjunto de entrenamiento o un conjunto de prueba proporcionado por el cliente, usaría opciones de Cross-validation o percentage split. Con Cross-validation, puede establecer el número de folds en los que se dividirán y utilizarán los datos completos durante cada iteración del entrenamiento. En la percentage split., dividirá los datos entre entrenamiento y prueba utilizando el percentage Split establecido.

Ahora, mantenga la opción de play como predeterminada para la clase de salida:



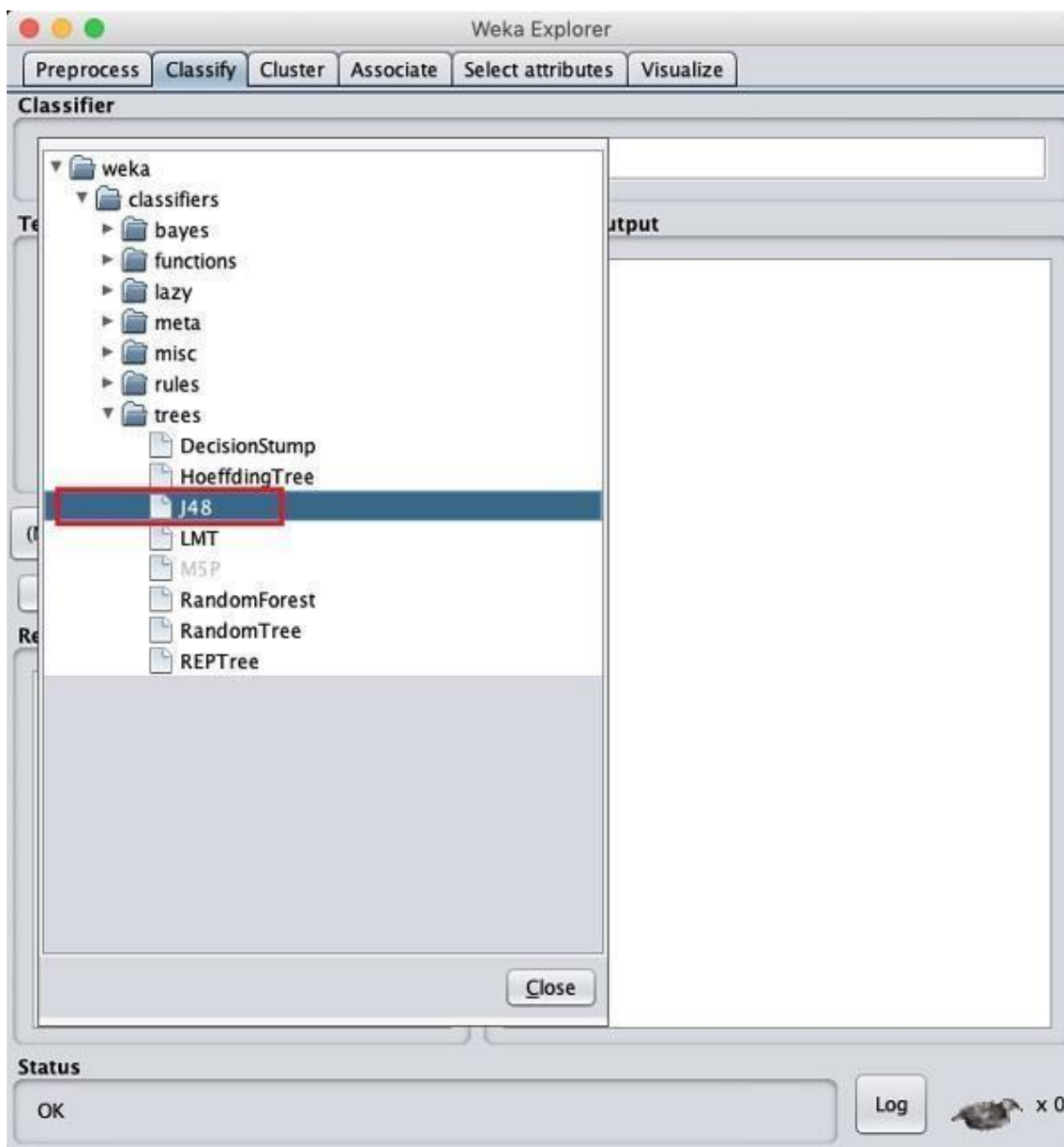
A continuación, seleccionará el clasificador.

Seleccionar clasificador

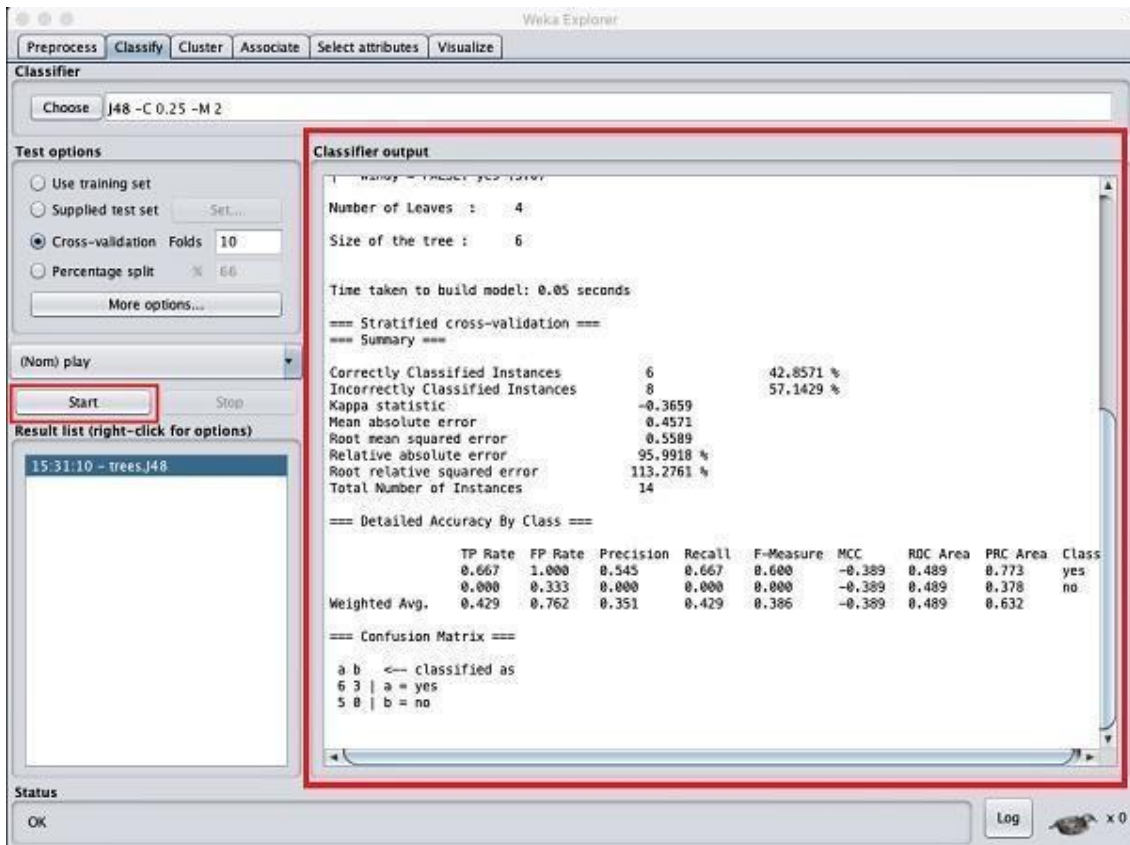
Haga clic en el botón Elegir y seleccione el siguiente clasificador:

weka→classifiers>trees>J48

Esto se muestra en la siguiente captura de pantalla:



Haga clic en el botón Start para iniciar el proceso de clasificación. Después de un tiempo, los resultados de la clasificación se presentarán en su pantalla como se muestra aquí:

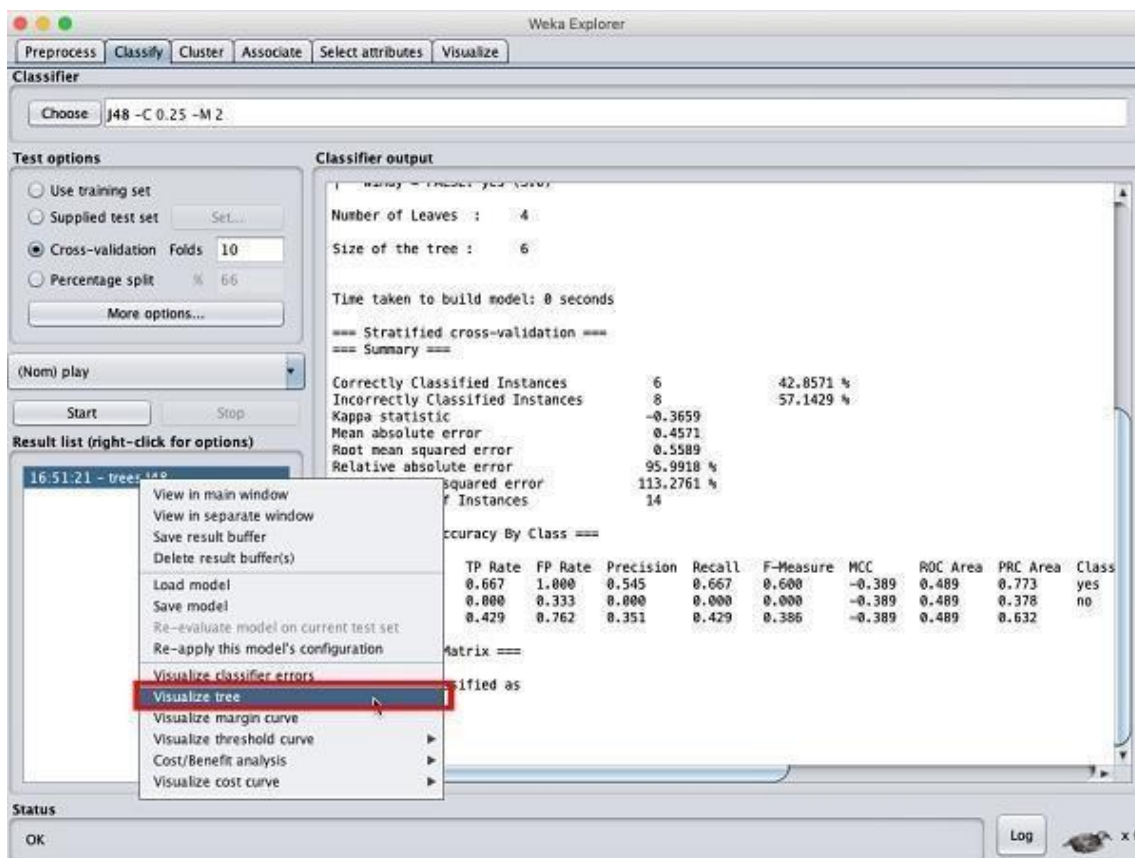


Examinemos la salida que se muestra en el lado derecho de la pantalla.

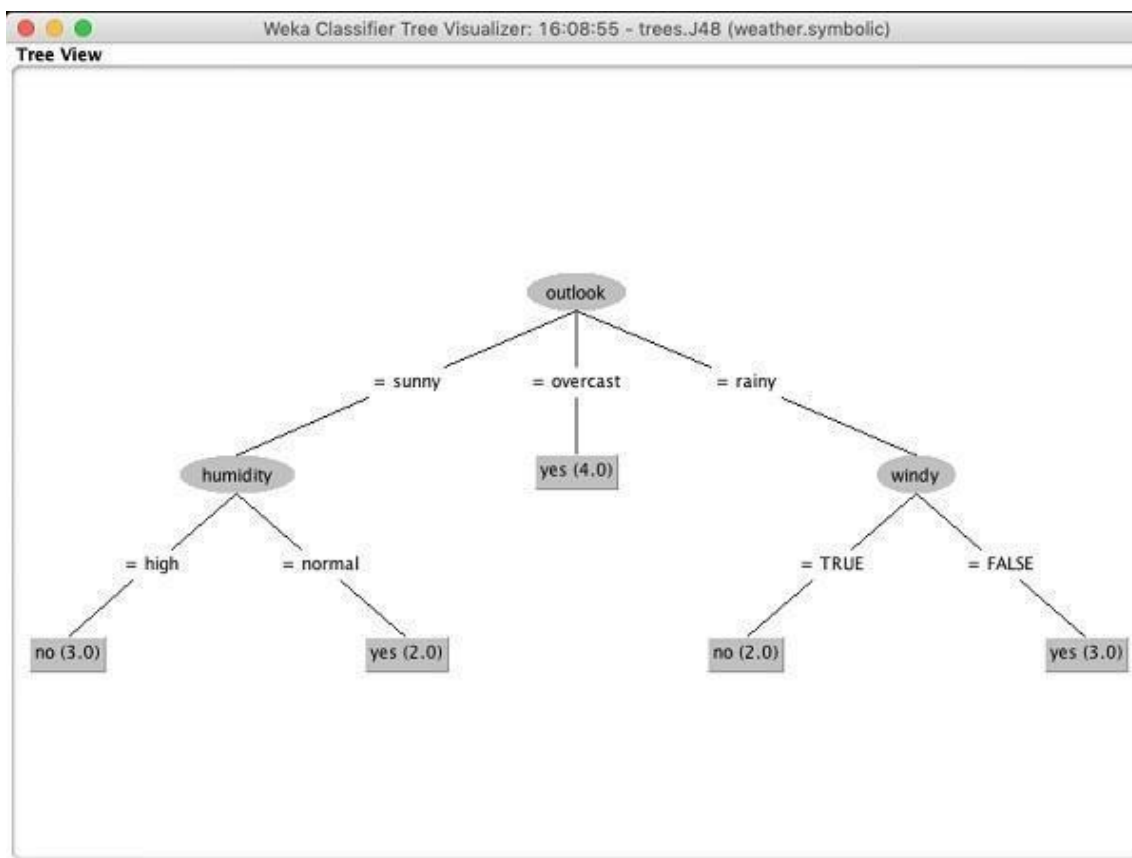
Dice que el tamaño del árbol es 6. En el Resumen, dice que las instancias clasificadas correctamente como 6 y las instancias clasificadas incorrectamente como 8, también dice que el error relativo absoluto es 95%. También muestra la Matriz de Confusión. Se puede deducir fácilmente a partir de estos resultados que la clasificación no es aceptable y que necesitará más datos para el análisis, para refinar su selección de características, reconstruir el modelo, etc. hasta que esté satisfecho con la precisión del modelo. De todos modos, de eso se trata WEKA. Le permite probar sus ideas rápidamente.

Visualizar resultados

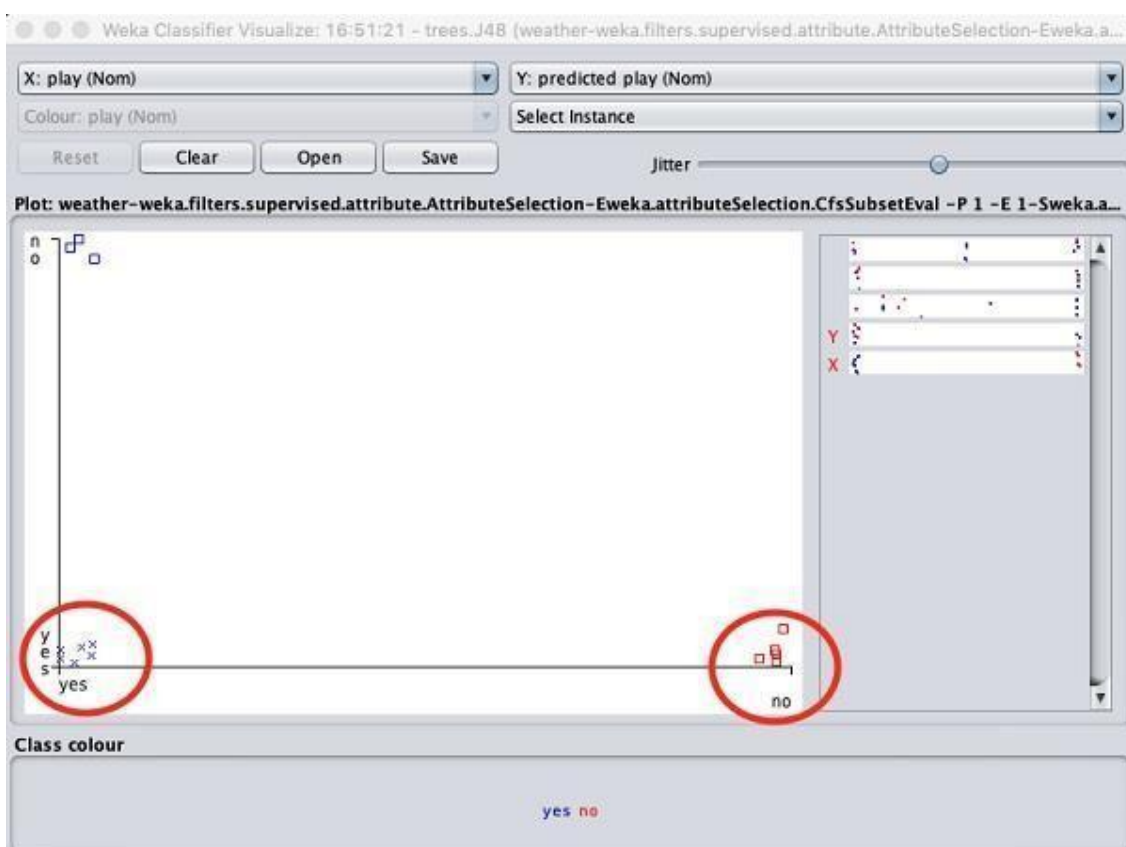
Para ver la representación visual de los resultados, haga clic derecho en el resultado en el cuadro de lista Resultado. Varias opciones aparecerían en la pantalla como se muestra aquí:



Seleccione Visualize tree para obtener una representación visual del árbol transversal como se ve en la siguiente captura de pantalla:

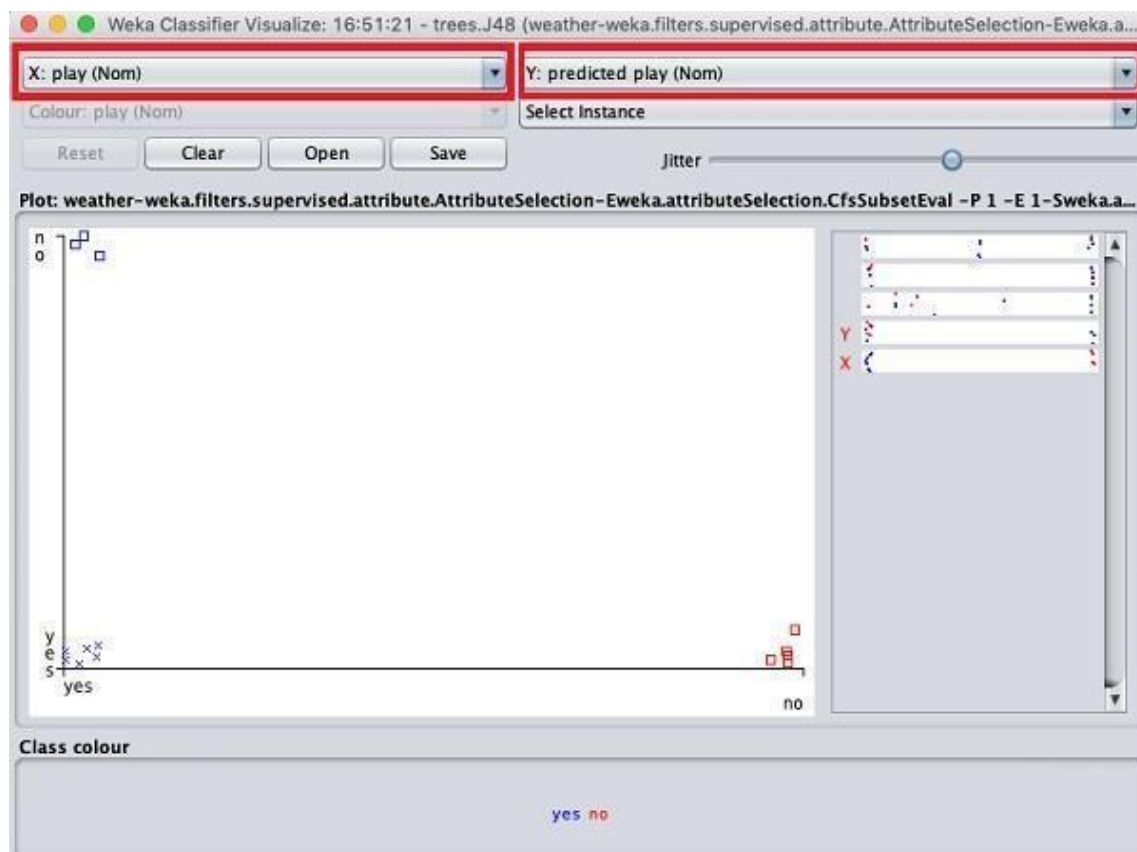


Seleccionando Visualize classifier errors



Una cruz representa una instancia clasificada correctamente, mientras que los cuadrados representan instancias clasificadas incorrectamente. En la esquina inferior izquierda de la gráfica, verá una cruz que indica si el outlook es sunny y luego juegue. Así que esta es una instancia clasificada correctamente. Para ubicar instancias, puede introducir algo de jitter deslizando la barra deslizante de jitter(fluctuación)

La trama actual es Outlook versus Play. Estos se indican mediante los dos cuadros de lista desplegable en la parte superior de la pantalla.



Ahora, intentemos una selección diferente en cada uno de estos cuadros y observe cómo cambian los ejes X e Y. Lo mismo se puede lograr utilizando las franjas horizontales en el lado derecho. Cada tira representa un atributo. El clic izquierdo en la tira establece el atributo seleccionado en el eje X, mientras que un clic derecho lo establecería en el eje Y.

Output Confusion Matrix

Muestra la matriz de confusión del clasificador. Esta tabla donde el número de columnas es el número de atributos muestra de la clasificación de las instancias. Da una información muy útil porque no solo da la información de los errores sino que también da el tipo.

Si tuviéramos el problema de clasificar vacas gordas y flacas y corriéramos el clasificador la matriz nos podría dar:

Gordas	Flacas	
32	4	Gordas
4	43	Flacas

Donde las columnas indican las categorías clasificadas por el clasificador y las filas las categorías reales de los datos. Entonces en la diagonal principal están los elementos que ha acertado el clasificador y los demás son los errores.

Calculo de indice de confianza, captura y soporte

La confianza, está dada por la relación que existe entre la totalidad de las observaciones que fueron afectados por la regla (47 vacas) y la cantidad de observaciones que fueron afectadas por la clase mayoritaria (43 realmente son vacas flacas que captó la regla) con esta misma regla.

$$\% \text{confianza} = 43/47 * 100 = 91.5\%$$

El porcentaje de captura, está dado por la relación que existe entre la cantidad de observaciones de la clase mayoritaria que fueron afectadas por esta regla (43 realmente viven que capto la regla) y la cantidad total de observaciones procesadas pertenecientes a esta misma clase (47 sobreviven global).

$$\% \text{captura} = 43/47 * 100 = 91.5\%$$

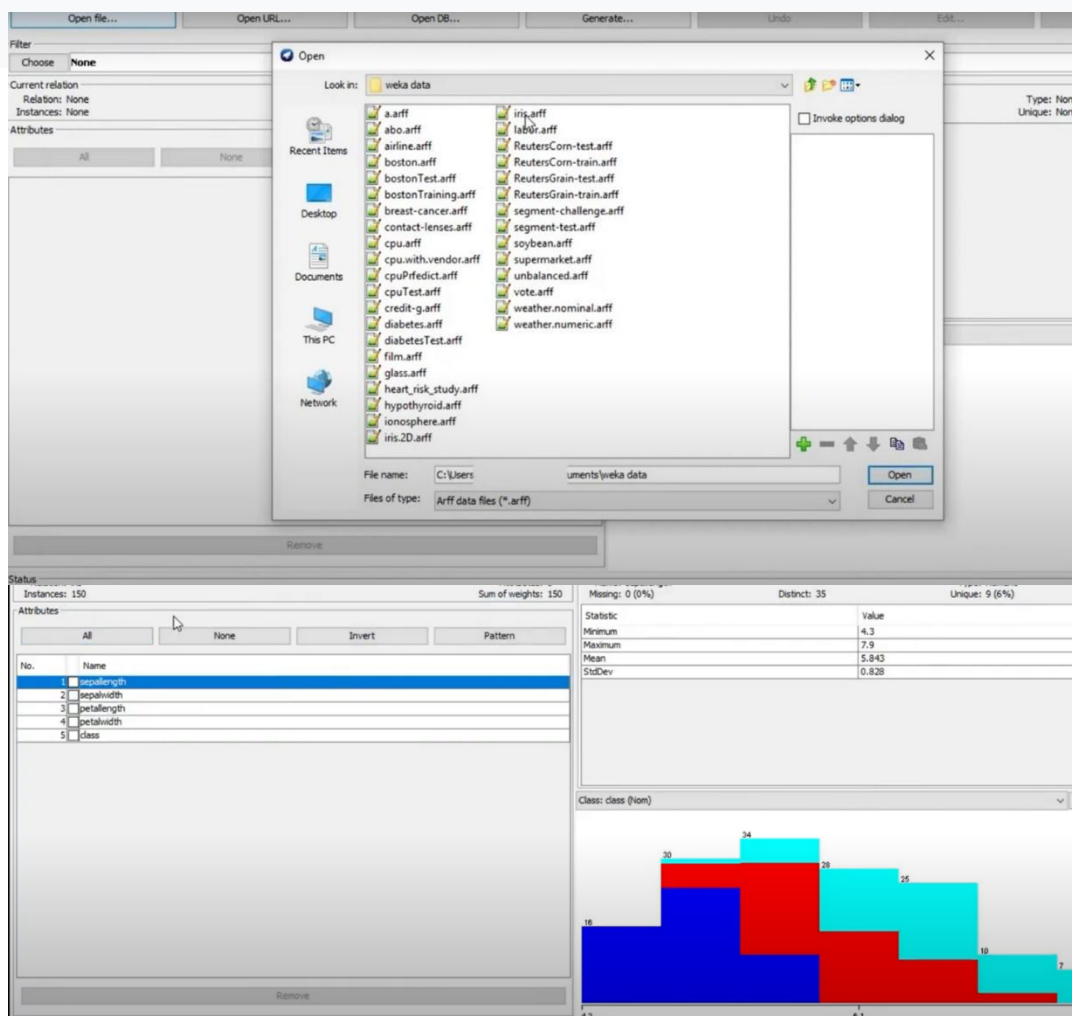
La cantidad de observaciones que soporta la regla, esta dada por la relación que existe entre la totalidad de las observaciones que afecta la regla (47 vacas flacas segun el arbol) y la totalidad de observaciones procesadas (83 casos).

$$\text{Soporte: } 47/83 * 100 = 56.62\%$$

Hay varias otras funcionalidades provistas para su análisis más profundo.

Multilayer Perceptron (MPL)

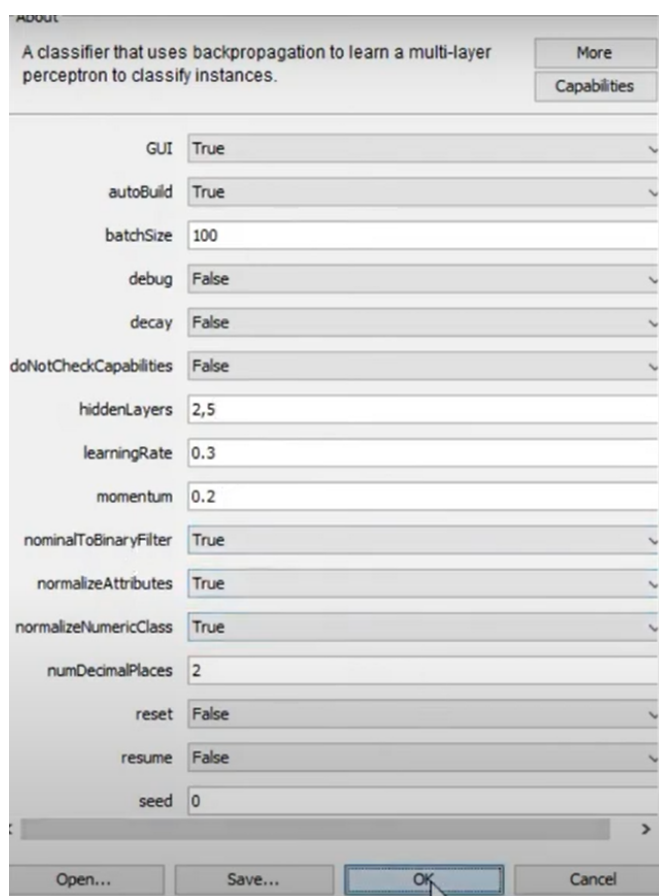
- Es una red neuronal artificial profunda conectada a múltiples capas en un grafo directo.
 - Es una técnica básica que utiliza el concepto de cerebro biológico
 - MPL utiliza la técnica de aprendizaje supervisado llamada backpropagation para el training MPL es utilizada en:
 1. Conjuntos de datos tabulados
 2. Clasificación.
 - Se utiliza en aplicaciones que realizan reconocimiento de voz, imagen, etc.
- Para la utilización de este algoritmo levantaremos de vuelta el archivo iris.arf



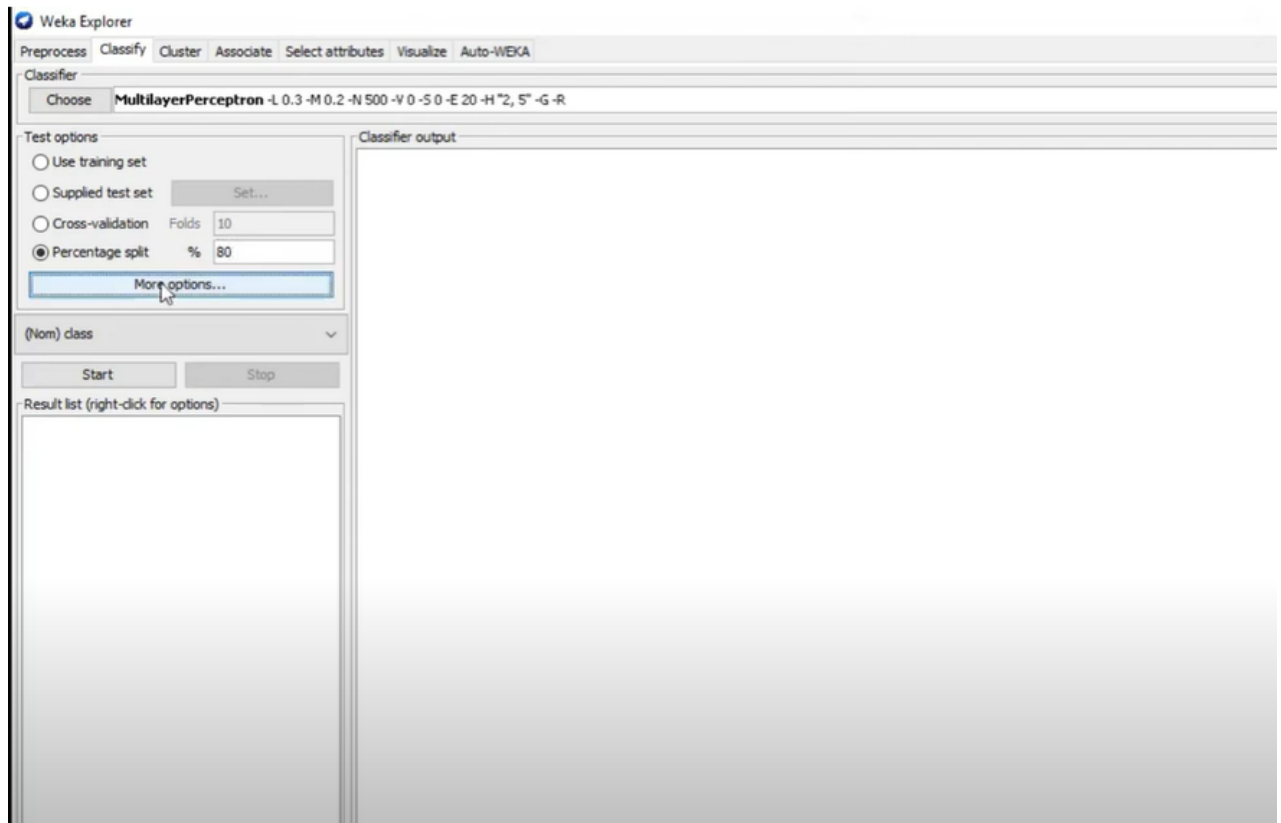
En este caso iremos a **Classifiers->functions->MultilayerPerceptron**



Los parámetros que utilizamos son:

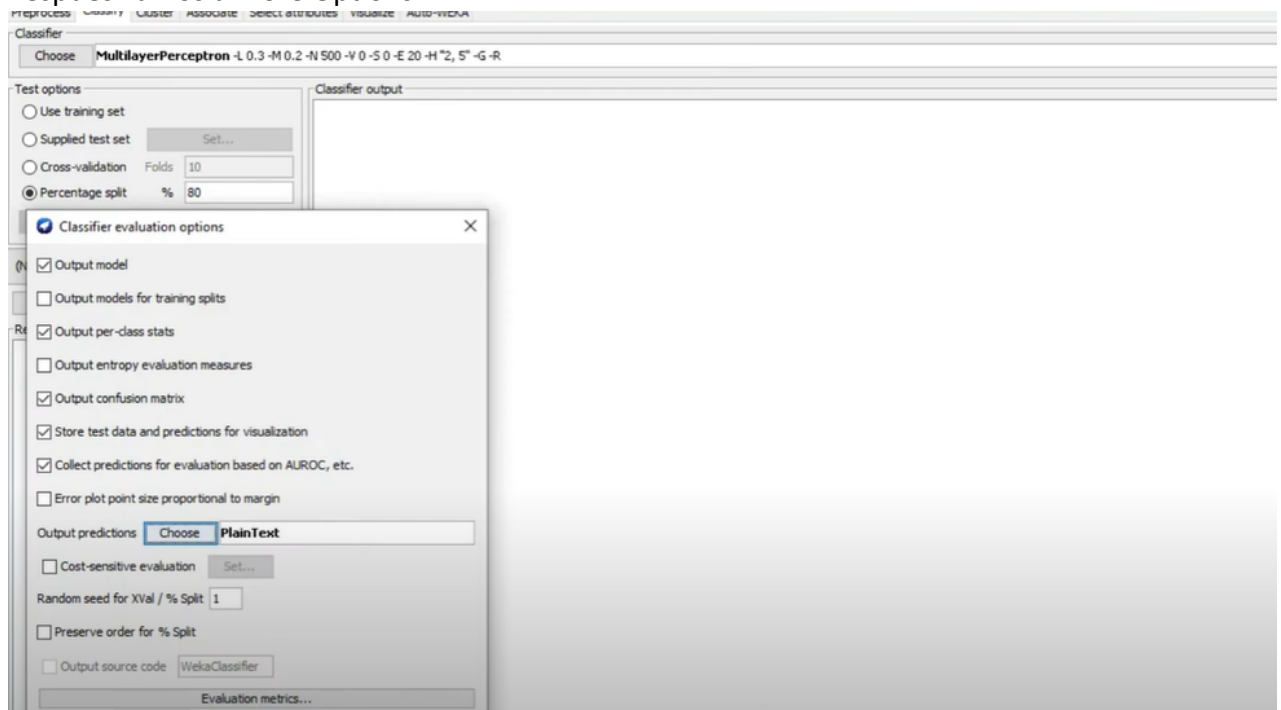


GUI: True, Hidden Layers: 2, 5 neuronas en las respectivas franjas.

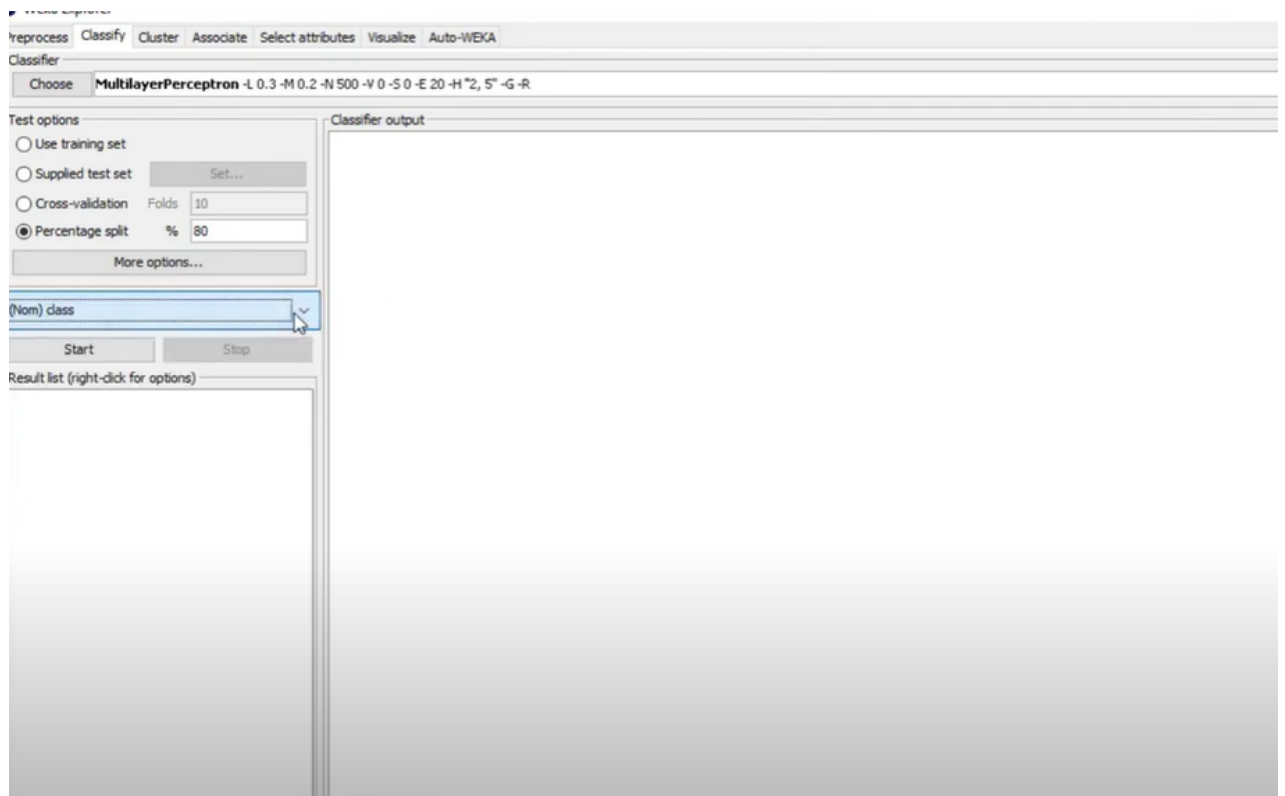


Percentage Split: 80%

Después vamos a more Options.



En Output Predictors: PlainText

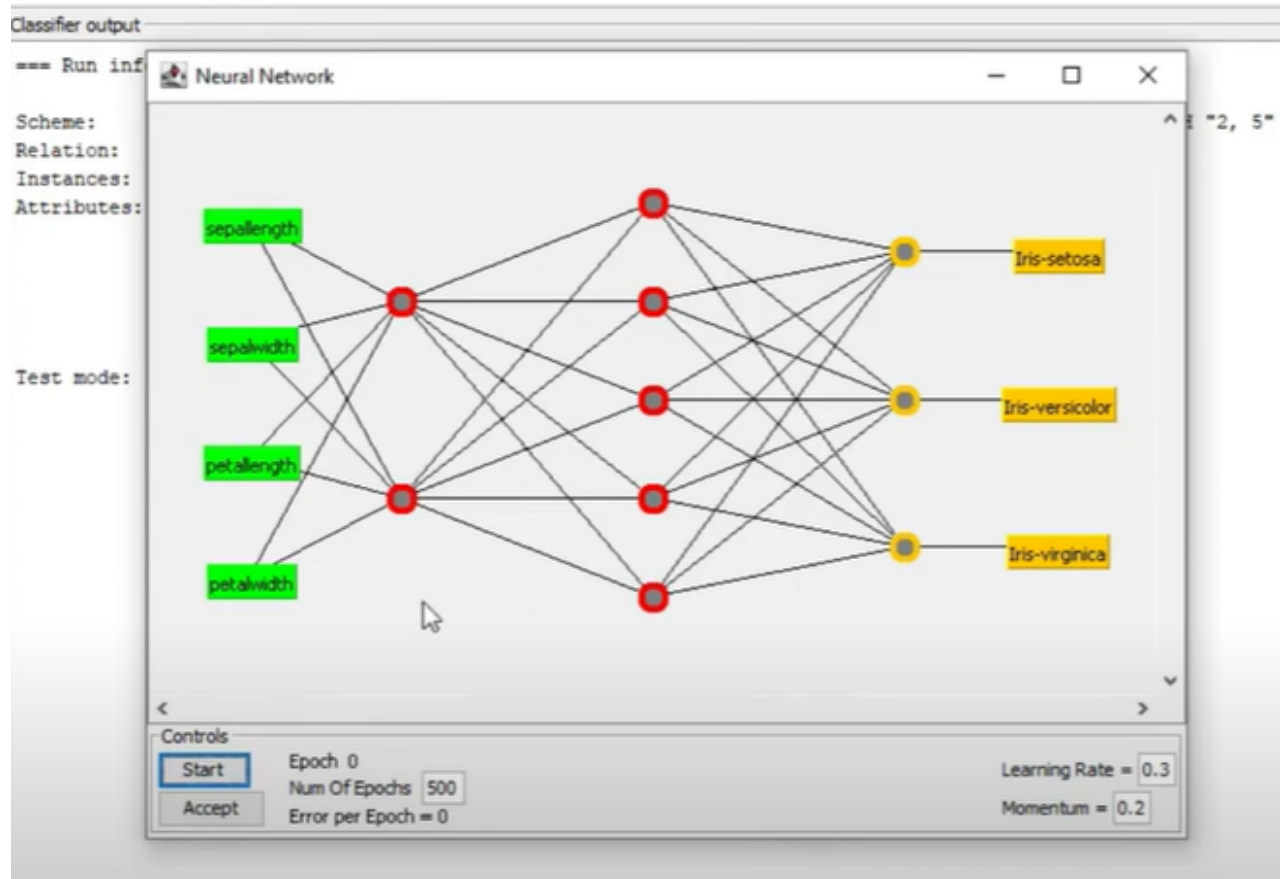


No hay que olvidarse de poner el atributo class en este caso como target y corremos start.

Toda red neuronal tiene dos pasos:

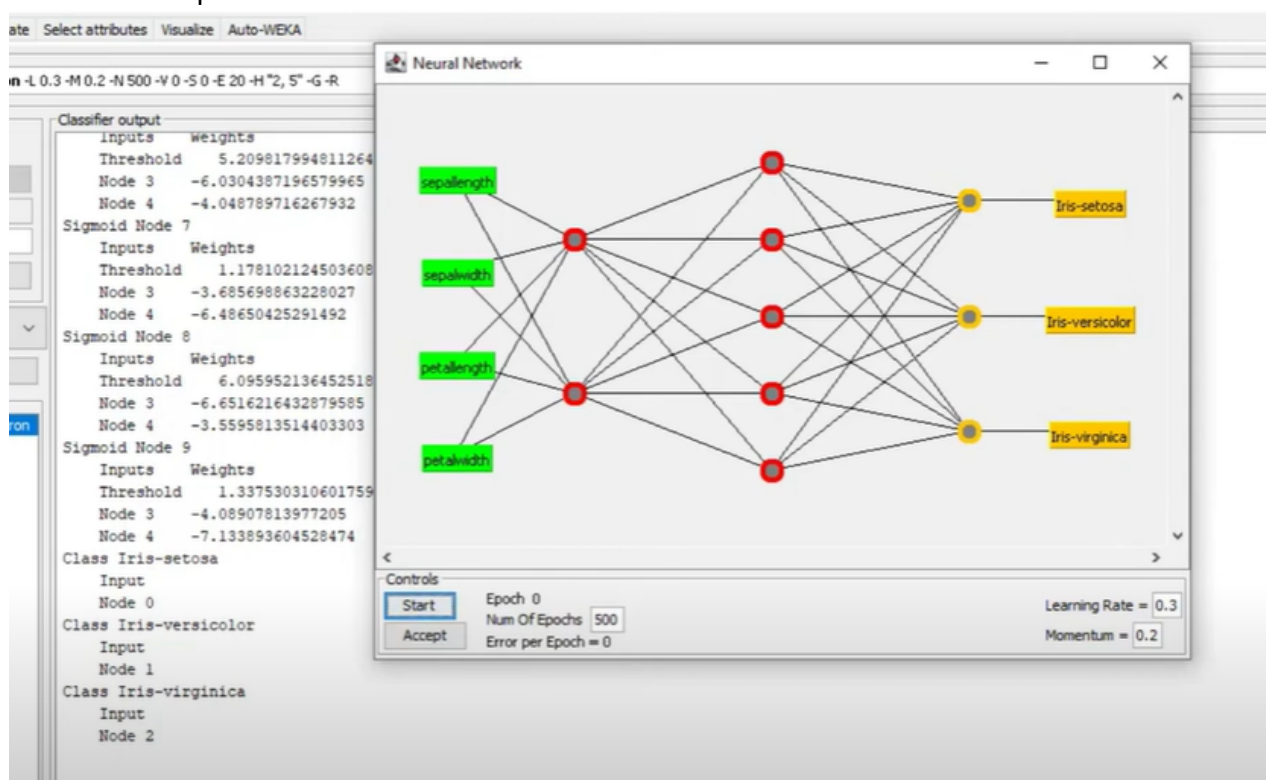
- Fast generation y entrenamiento: Generación rápida del modelo y entrenamiento
- Test: que utiliza el set de datos para testear el modelo.

Al correr Start dijimos nos aparecerá el primer paso.



En esta primera instancia se puede ver las cuatro entradas, las dos neuronas próximas, y las cinco, después se puede ver las 3 salidas del modelo. Apretamos Start y después Accept.

Después de esto ya entrenamos el modelo y lo vamos a testear, nos aparecerá la misma pantalla con el modelo para testearlo:



Oprimimos Start y Accept y vamos a la parte a la prediction área:

Classifier output

Time taken to build model: 76.13 seconds

=== Predictions on test split ===

inst#	actual	predicted	error	prediction
1	3:Iris-virginica	3:Iris-virginica	0.982	
2	2:Iris-versicolor	2:Iris-versicolor	0.991	
3	2:Iris-versicolor	2:Iris-versicolor	0.97	
4	1:Iris-setosa	1:Iris-setosa	0.989	
5	3:Iris-virginica	3:Iris-virginica	0.982	
6	1:Iris-setosa	1:Iris-setosa	0.99	
7	2:Iris-versicolor	2:Iris-versicolor	0.995	
8	1:Iris-setosa	1:Iris-setosa	0.989	
9	1:Iris-setosa	1:Iris-setosa	0.989	
10	2:Iris-versicolor	2:Iris-versicolor	0.989	
11	3:Iris-virginica	3:Iris-virginica	0.982	
12	2:Iris-versicolor	2:Iris-versicolor	0.993	
13	1:Iris-setosa	1:Iris-setosa	0.99	
14	1:Iris-setosa	1:Iris-setosa	0.989	
15	3:Iris-virginica	3:Iris-virginica	0.982	
16	1:Iris-setosa	1:Iris-setosa	0.989	
17	1:Iris-setosa	1:Iris-setosa	0.989	
18	3:Iris-virginica	2:Iris-versicolor	+	0.993
19	1:Iris-setosa	1:Iris-setosa	0.989	
20	2:Iris-versicolor	2:Iris-versicolor	0.995	
21	1:Iris-setosa	1:Iris-setosa	0.989	
22	3:Iris-virginica	3:Iris-virginica	0.982	
23	3:Iris-virginica	3:Iris-virginica	0.982	
24	2:Iris-versicolor	2:Iris-versicolor	0.991	
25	2:Iris-versicolor	2:Iris-versicolor	0.993	
26	2:Iris-versicolor	2:Iris-versicolor	0.995	

Si analizamos las distintas instancias podemos ver que predispone muy bien.
Si vamos a la Summery section

=== Summary ===

Correctly Classified Instances	29	96.6667 %
Incorrectly Classified Instances	1	3.3333 %
Kappa statistic	0.9497	
Mean absolute error	0.0306	
Root mean squared error	0.1486	
Relative absolute error	6.871 %	
Root relative squared error	31.4903 %	
Total Number of Instances	30	

=== Detailed Accuracy By Class ===

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	Iris-setosa
	1.000	0.050	0.909	1.000	0.952	0.929	0.975	0.943	Iris-versicolor
	0.889	0.000	1.000	0.889	0.941	0.921	0.995	0.989	Iris-virginica
Weighted Avg.	0.967	0.017	0.970	0.967	0.966	0.953	0.990	0.978	

=== Confusion Matrix ===

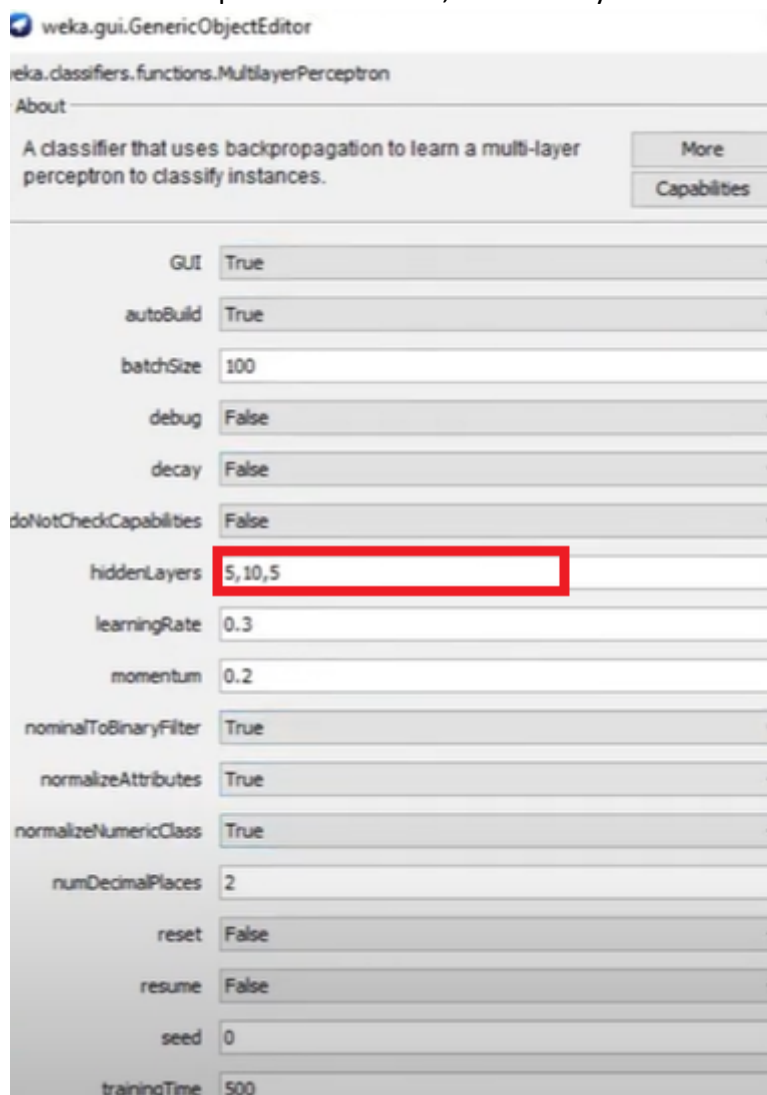
a	b	c	<-- classified as
11	0	0	a = Iris-setosa
0	10	0	b = Iris-versicolor
0	1	8	c = Iris-virginica

Podemos ver que las instancias correctamente clasificadas son 97% (29) y 3% mal clasificadas (1)
También podemos ver la matriz de confusión, donde once para Iris-setosa están bien clasificadas,

diez de Iris-versicolor también bien clasificadas y para el caso de Iris-virginica ocho bien clasificadas y una mal clasificada.

Después de este análisis podemos correr el mismo mpl model con distinta combinaciones de Hidden layer (neuronas)

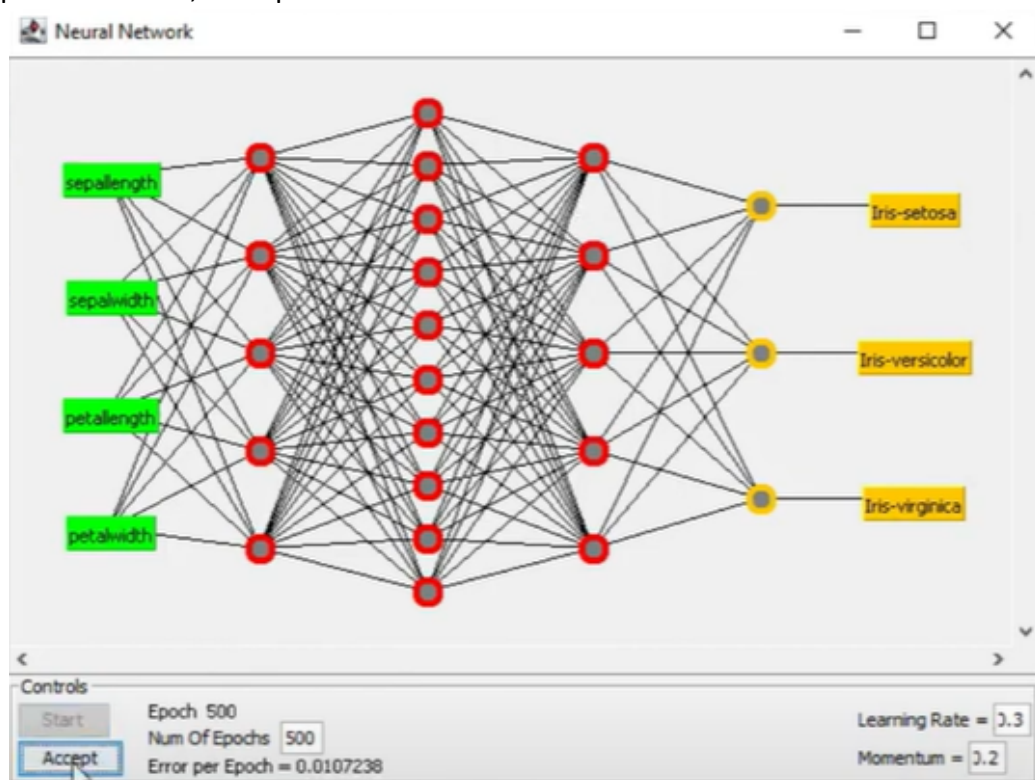
En este caso lo haremos con una capa de 5 neuronas, una de 10 y otra de 5



The screenshot shows the Weka GUI's GenericObjectEditor for the MultilayerPerceptron classifier. The 'hiddenLayers' field is highlighted with a red box and contains the value '5,10,5'. Other settings include GUI (True), autoBuild (True), batchSize (100), debug (False), decay (False), doNotCheckCapabilities (False), learningRate (0.3), momentum (0.2), nominalToBinaryFilter (True), normalizeAttributes (True), normalizeNumericClass (True), numDecimalPlaces (2), reset (False), resume (False), seed (0), and trainingTime (500).

Property	Value
GUI	True
autoBuild	True
batchSize	100
debug	False
decay	False
doNotCheckCapabilities	False
hiddenLayers	5,10,5
learningRate	0.3
momentum	0.2
nominalToBinaryFilter	True
normalizeAttributes	True
normalizeNumericClass	True
numDecimalPlaces	2
reset	False
resume	False
seed	0
trainingTime	500

Cuando ponemos Start, nos aparece el modelo



Lo entrenamos y lo testeamos en este caso con esta configuración la salida nos dará:

```

=== Summary ===
Correctly Classified Instances      28      93.3333 %
Incorrectly Classified Instances    2       6.6667 %
Kappa statistic                    0.8997
Mean absolute error                 0.0562
Root mean squared error             0.1784
Relative absolute error             12.6315 %
Root relative squared error         37.8166 %
Total Number of Instances          30

=== Detailed Accuracy By Class ===

```

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	FRC Area	Class
	1.000	0.000	1.000	1.000	1.000	1.000	1.000	1.000	Iris-setosa
	0.900	0.050	0.900	0.900	0.900	0.850	0.990	0.983	Iris-versicolor
	0.889	0.048	0.889	0.889	0.889	0.841	0.989	0.977	Iris-virginica
Weighted Avg.	0.933	0.031	0.933	0.933	0.933	0.902	0.993	0.987	

```

=== Confusion Matrix ===
  a  b  c  <-- classified as
11  0  0 | a = Iris-setosa
 0  9  1 | b = Iris-versicolor
 0  1  8 | c = Iris-virginica

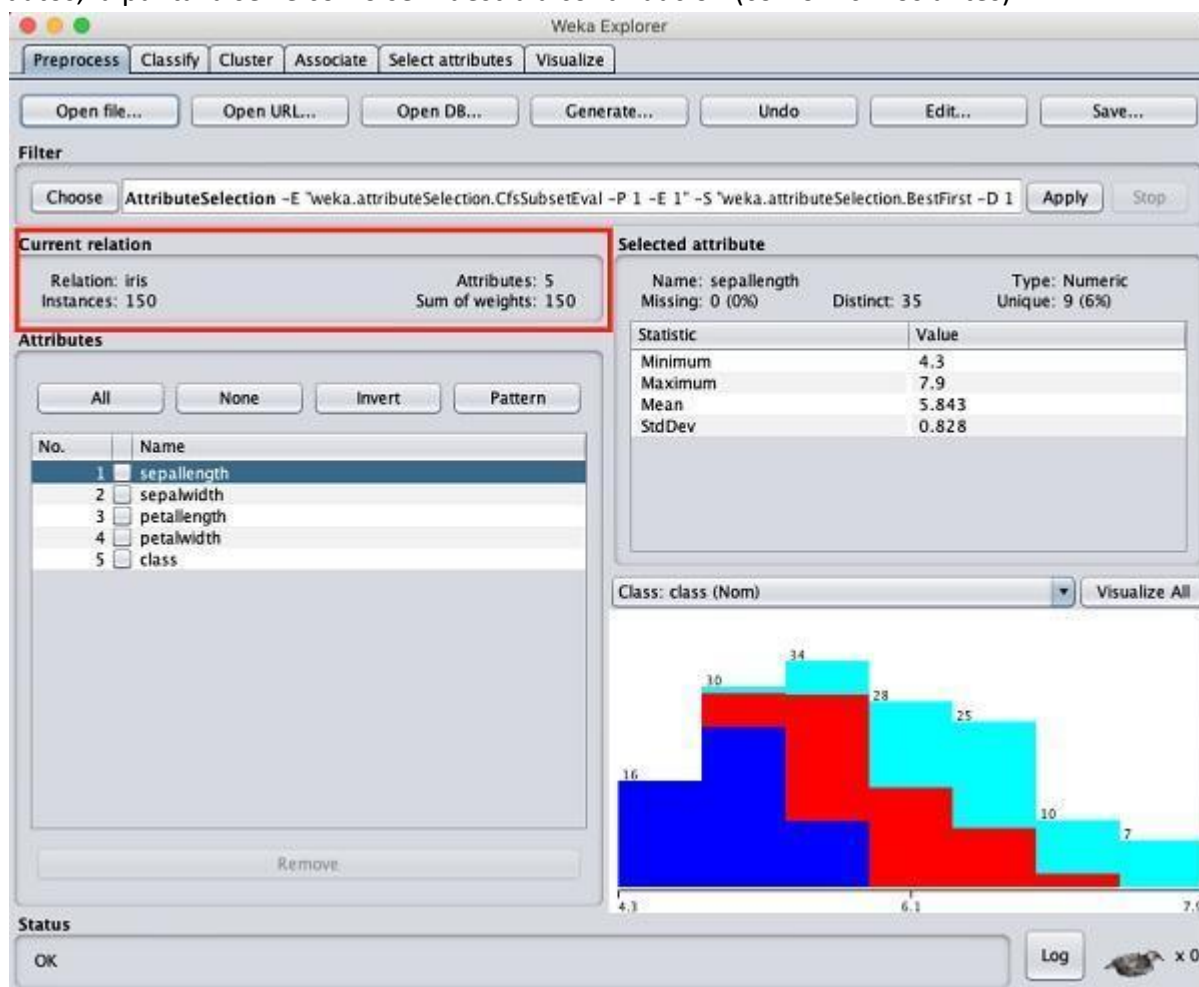
```

Podemos observar que las instancias correctas están en un 93%, de esto podemos concluir que el agregar franjas de neuronas no nos asegura un porcentaje mayor de instancias bien clasificadas.

Un algoritmo de agrupamiento encuentra grupos de instancias similares en todo el conjunto de datos. WEKA admite varios algoritmos de agrupamiento, como EM, FilteredClusterer, HierarchicalClusterer, SimpleKMeans, etc. Debe comprender completamente estos algoritmos para aprovechar al máximo las capacidades

Cargando datos

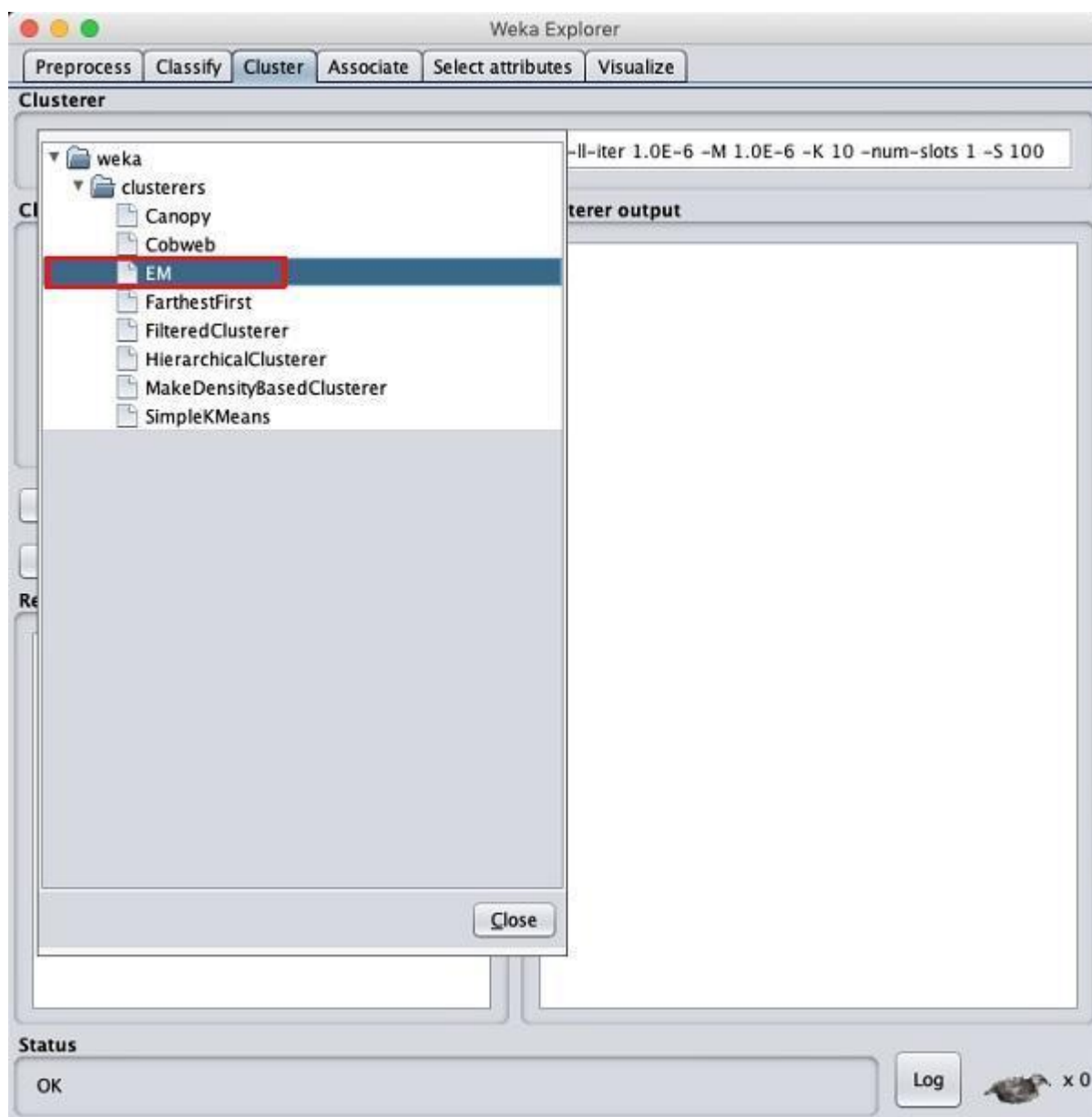
En el explorador WEKA, seleccione la pestaña Preprocesar. Haga clic en la opción Abrir archivo... y seleccione el archivo iris.arff en el cuadro de diálogo de selección de archivos. Cuando carga los datos, la pantalla se ve como se muestra a continuación (como hicimos antes):



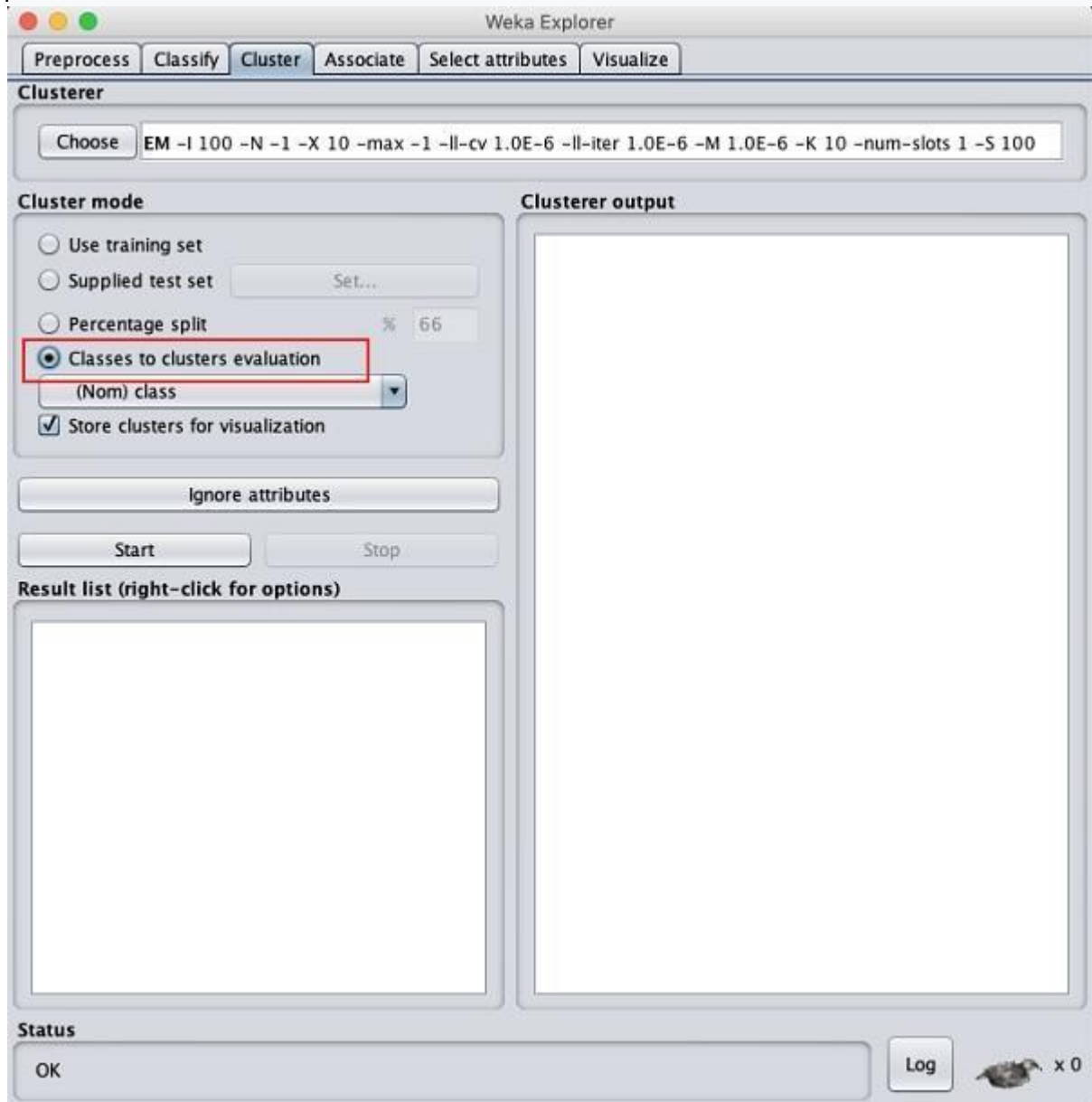
Puede observar que hay 150 instancias y 5 atributos. Los nombres de los atributos se enumeran como longitud de sépalo, ancho de sépalo, longitud de pétalo, ancho de pétalo y clase. Los primeros cuatro atributos son de tipo numérico mientras que la clase es de tipo nominal con 3 valores distintos. Siempre debemos examinar cada atributo para comprender las características de la base de datos. No haremos ningún procesamiento previo de estos datos y procederemos directamente a la construcción del modelo.

Clustering

Haga clic en la PESTAÑA Clúster para aplicar los algoritmos de agrupamiento a nuestros datos cargados. Haga clic en el botón Elegir. Verá la siguiente pantalla:



Ahora, seleccione EM como el algoritmo de agrupación. En la subventana del modo de clúster, seleccione la opción de evaluación Clases a clústeres como se muestra en la siguiente captura de pantalla:

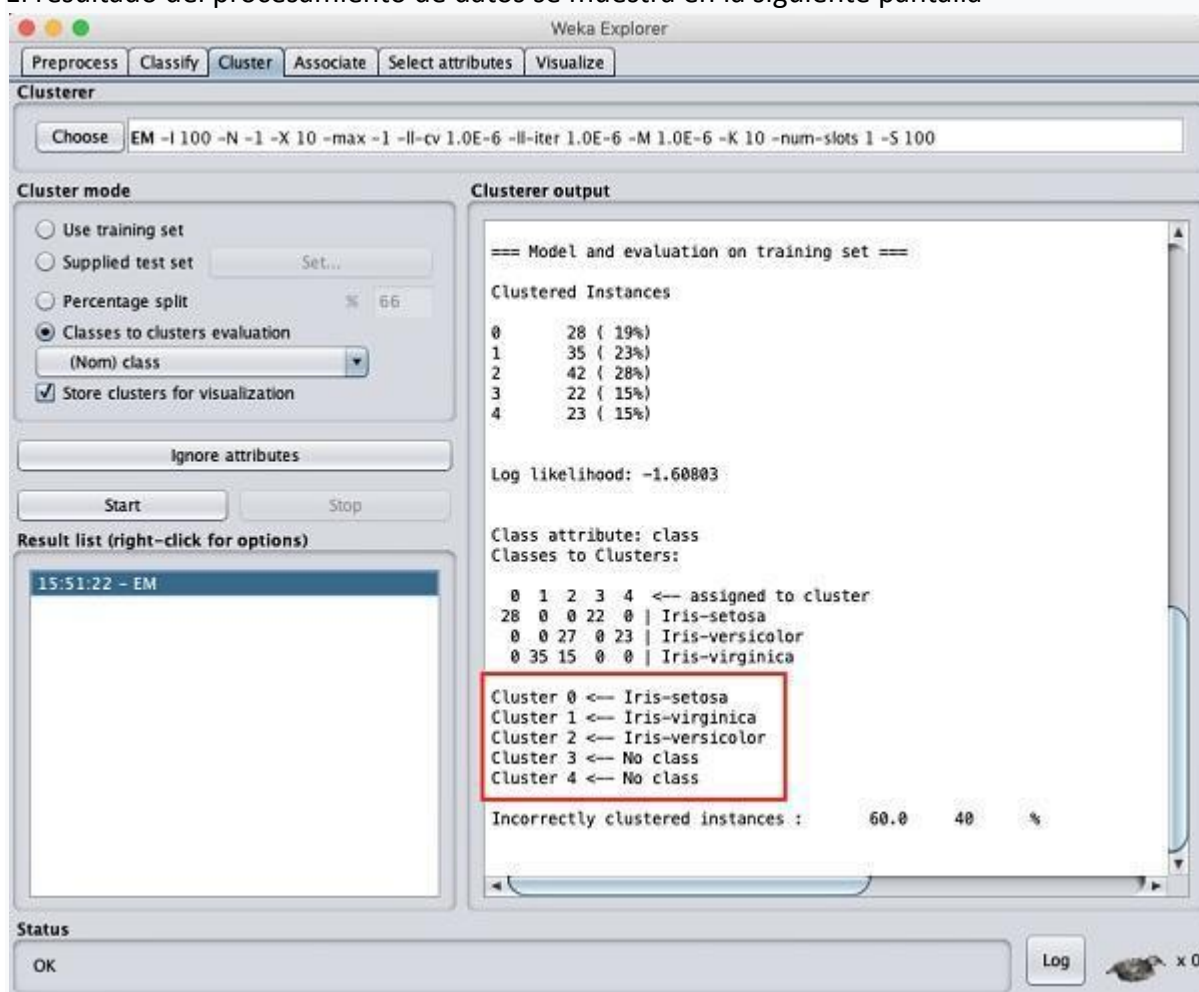


Haga clic en el botón Start para procesar los datos. Después de un tiempo, los resultados se presentarán en la pantalla.

A continuación, estudiemos los resultados.

Examinando el output

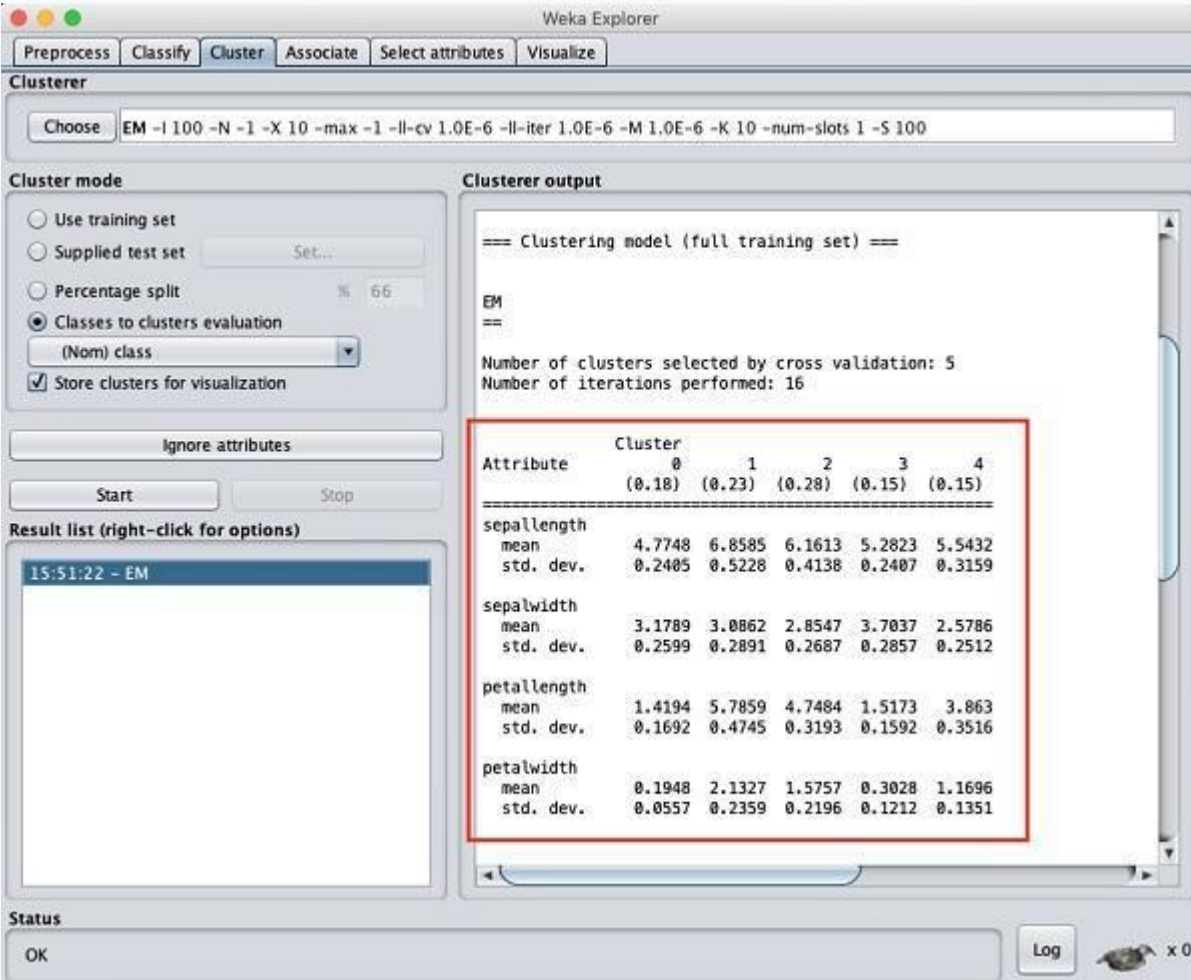
El resultado del procesamiento de datos se muestra en la siguiente pantalla



Desde la salida, puede observar que;

- Hay 5 instancias agrupadas detectadas en la base de datos.
- El Grupo 0 representa setosa, el Grupo 1 representa virginica, el Grupo 2 representa versicolor, mientras que los dos últimos grupos no tienen ninguna clase asociada con ellos.

Si nos desplazamos hacia arriba en la ventana de resultados, también verá algunas estadísticas que brindan la media y la desviación estándar para cada uno de los atributos en los diversos grupos detectados. Esto se muestra en la captura de pantalla que se muestra a continuación:



Clusterer

Choose `EM -I 100 -N -1 -X 10 -max -1 -ll-cv 1.0E-6 -ll-iter 1.0E-6 -M 1.0E-6 -K 10 -num-slots 1 -S 100`

Cluster mode

- ☐ Use training set
- ☐ Supplied test set `Set...`
- ☐ Percentage split `% 66`
- ☒ Classes to clusters evaluation
 - (Nom) class
- ☒ Store clusters for visualization

`Ignore attributes`

`Start` `Stop`

Result list (right-click for options)

15:51:22 - EM

Clusterer output

=== Clustering model (full training set) ===

EM

Number of clusters selected by cross validation: 5
Number of iterations performed: 16

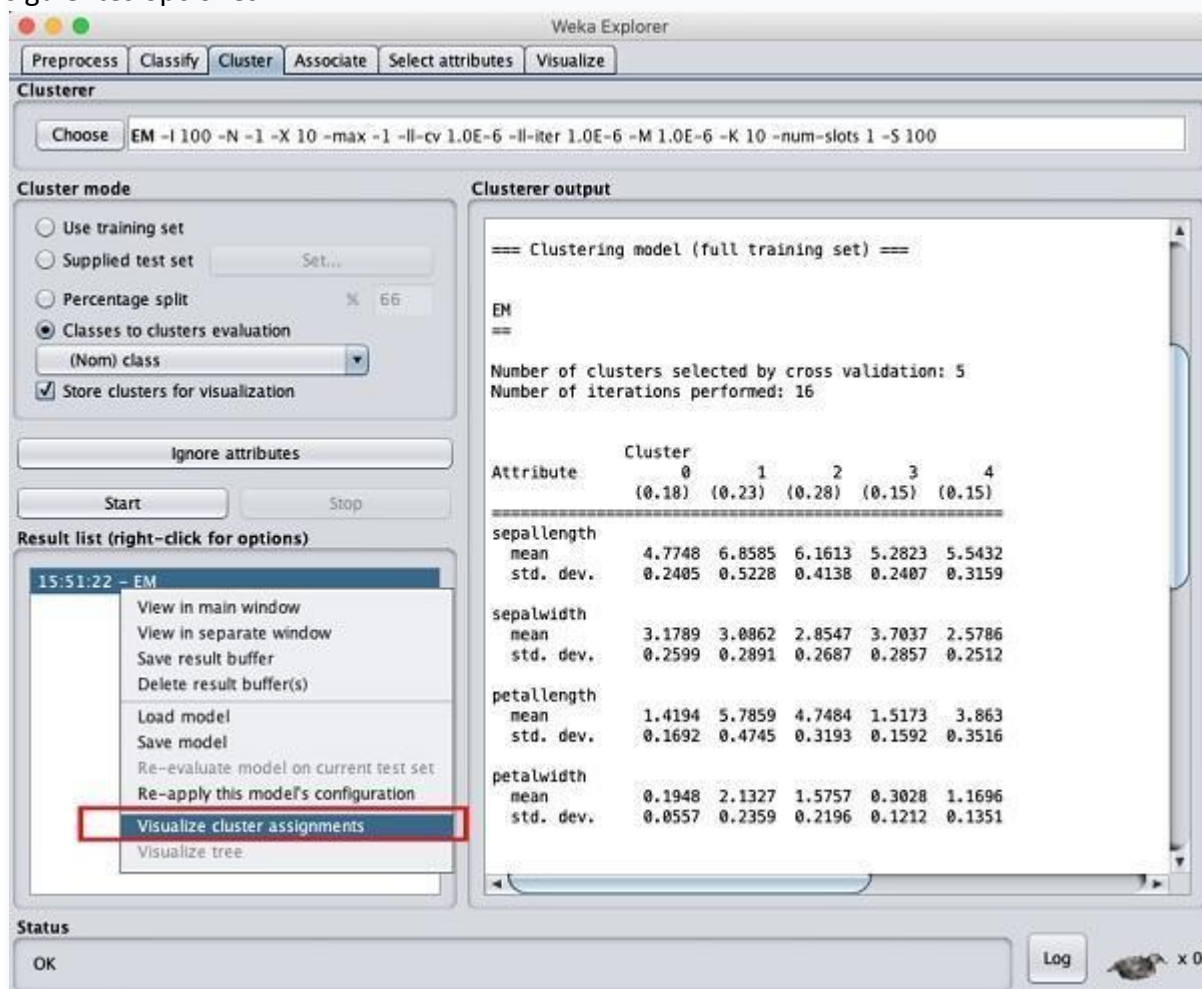
Attribute	Cluster 0 (0.18)	Cluster 1 (0.23)	Cluster 2 (0.28)	Cluster 3 (0.15)	Cluster 4 (0.15)
sepal length					
mean	4.7748	6.8585	6.1613	5.2823	5.5432
std. dev.	0.2405	0.5228	0.4138	0.2407	0.3159
sepal width					
mean	3.1789	3.0862	2.8547	3.7037	2.5786
std. dev.	0.2599	0.2891	0.2687	0.2857	0.2512
petal length					
mean	1.4194	5.7859	4.7484	1.5173	3.863
std. dev.	0.1692	0.4745	0.3193	0.1592	0.3516
petal width					
mean	0.1948	2.1327	1.5757	0.3028	1.1696
std. dev.	0.0557	0.2359	0.2196	0.1212	0.1351

Status

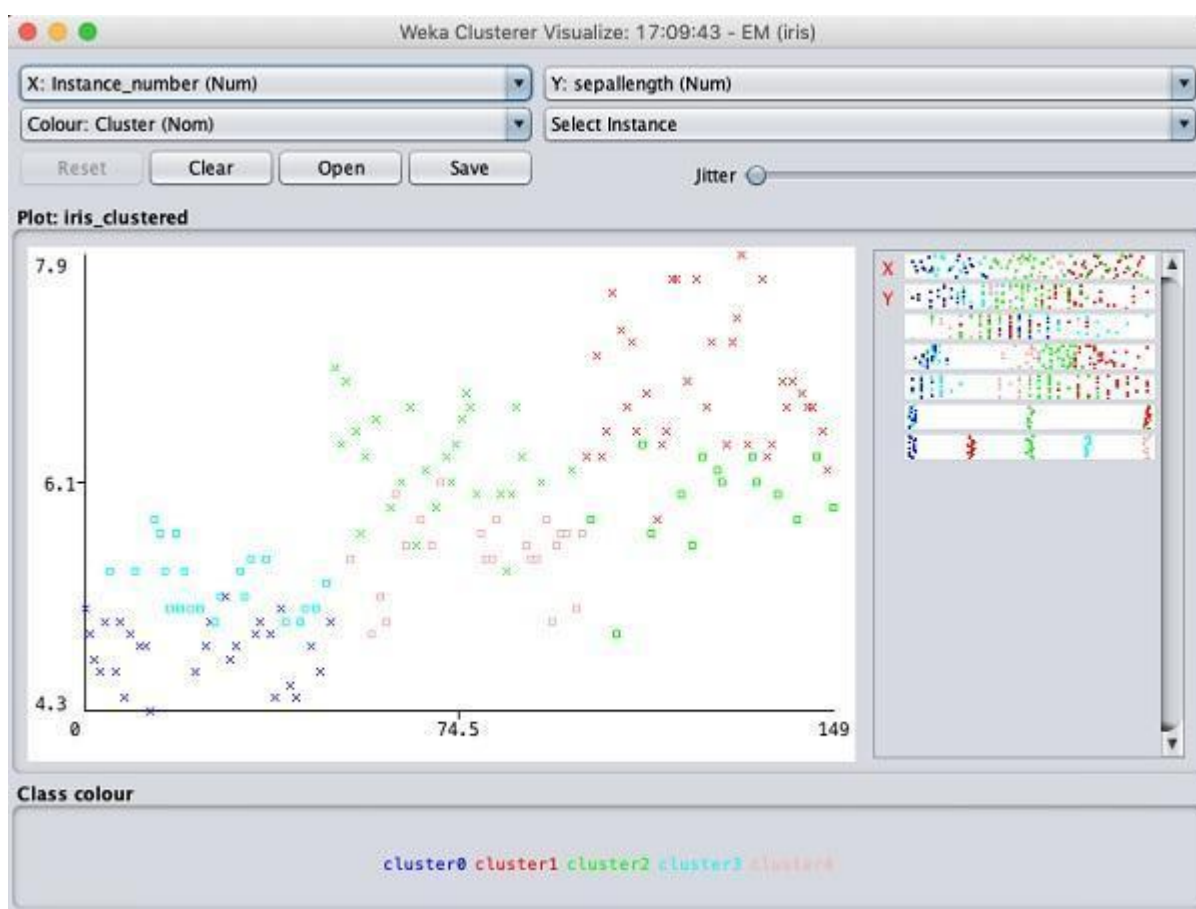
OK `Log` x 0

Visualizando Clusters

Para visualizar los grupos, haga clic con el botón derecho en el resultado de EM. Verá las siguientes opciones:



Seleccione Visualize cluster assingments. Verá el siguiente resultado:



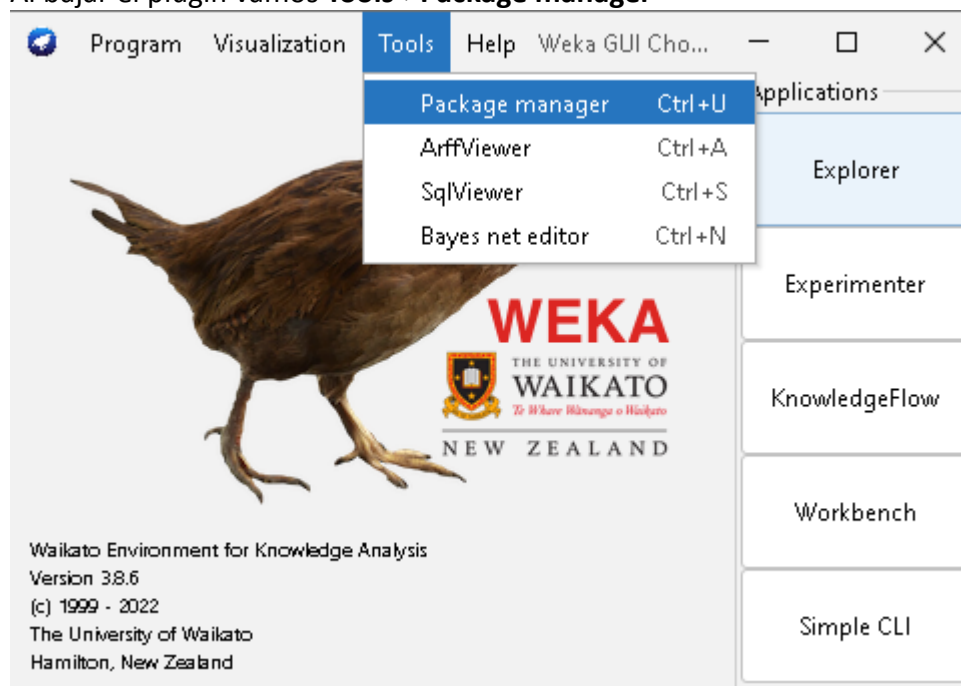
Como en el caso de la clasificación, notará la distinción entre las instancias identificadas correctamente e incorrectamente. Puede jugar cambiando los ejes X e Y para analizar los resultados. Puede utilizar el jitter como en el caso de la clasificación para averiguar la concentración de instancias correctamente identificadas

Redes Neuronales con Clustering

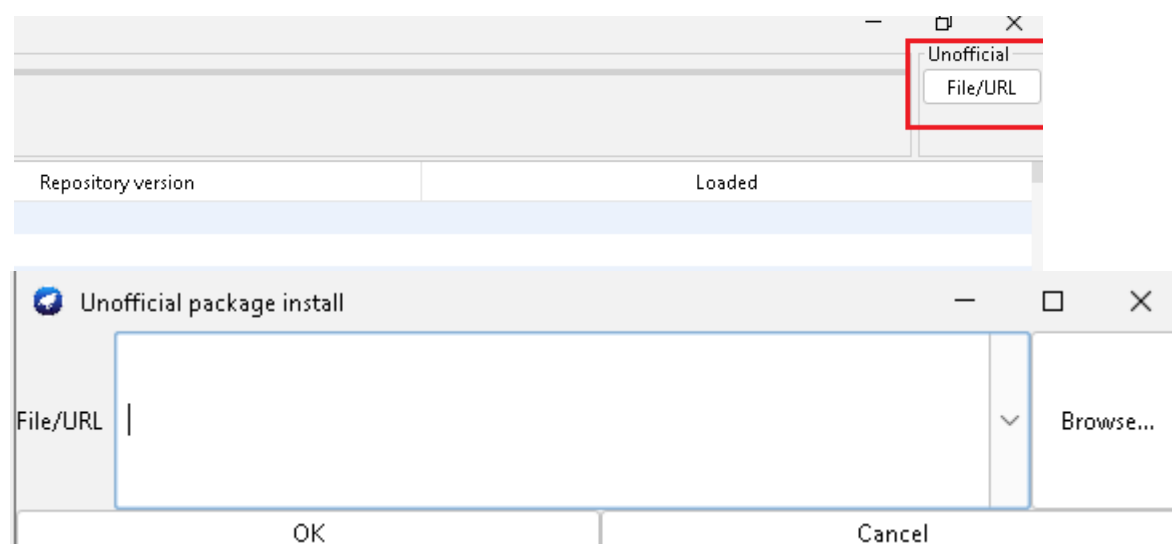
será necesario descargar e instalar en la aplicación el plugin:

https://sourceforge.net/projects/wekann/files/SelfOrganizingMap/SelfOrganizingMap1.0.3.zip/download?use_mirror=ufpr&download=

Al bajar el plugin vamos **Tools->Package manager**

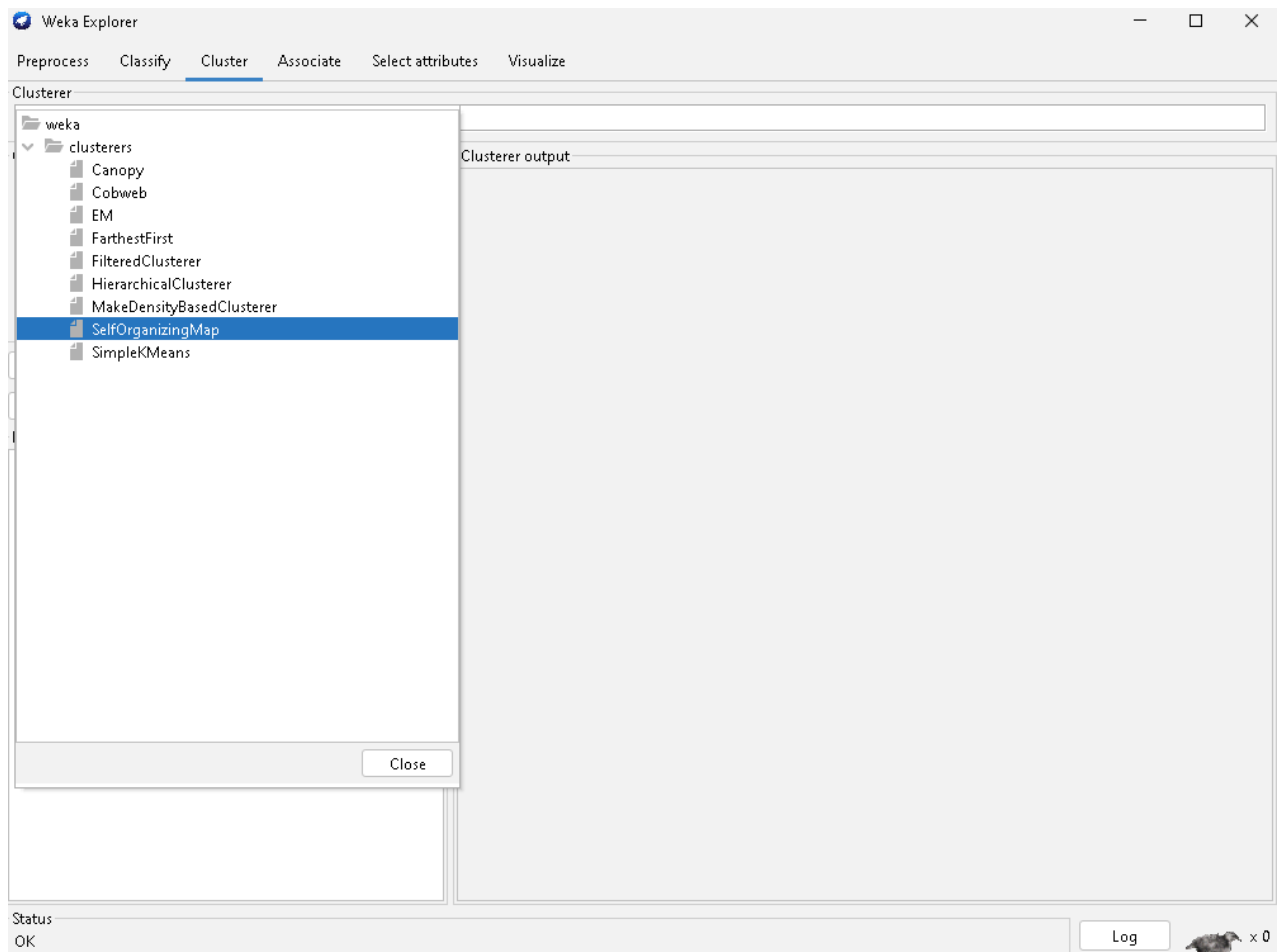


Después en el package manager buscamos el botón de Unofficial y cargamos el plugin



Y cargamos el plugin que está en el zip: SelfOrganizingMap.zip, después nos pide reiniciar weka.

Para encontrar el algoritmo, vamos a **cluster->SelfOrganizingMap**



¿Qué es un SOM?

Un SOM (Self Organizing Maps) es una clase especial de red neuronal usada como herramientas de clustering y visualización en el análisis exploratorio de datos. El objetivo principal de este es el reducir las dimensiones de los datos preservando la topología de los mismos, pero no así las distancias. Es un algoritmo de aprendizaje no supervisado.

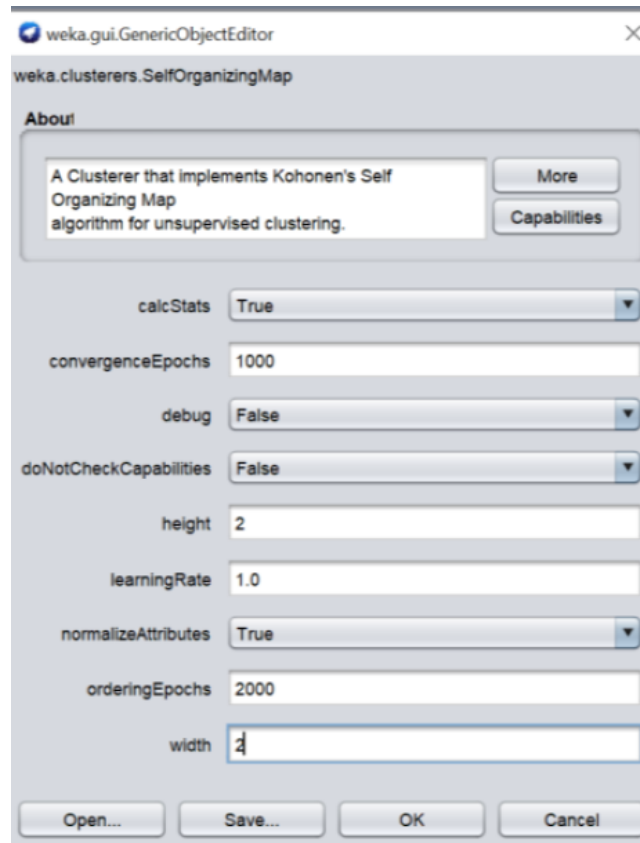
Tips para ejecutar el algoritmo en weka:

Para aplicar el algoritmo correctamente eliminamos las pocas filas que tienen valores nulos y cambiamos los atributos de texto a valor numérico.

Luego estandarizamos los valores para que todas las columnas tengan promedio cero y varianza igual a 1. Esto se debe hacer porque el algoritmo calcula las distancias de los distintos puntos del set original a las neuronas por lo que, si las distribuciones de los valores que toman cada variable difieren mucho, el algoritmo no va dar buenos resultados.

No olvidarse de sacar la columna de clase para correr el algoritmo.

Para ejecutar el algoritmo lo configuramos en primera instancia como muestra la siguiente imagen:



Los parámetros en esta ejecución son:

Cantidad de neuronas del mapa

En total usamos 4 neuronas en un mapa de 2 filas por dos columnas.

Cantidad de ciclos de entrenamientos

La cantidad de ciclos de entrenamiento que usamos fue 1000.

Learning rate

El learning rate lo configuramos en 1%.

El resultado de la corrida del algoritmo, se analiza similar a clustering