

# WEKA 3

CCS7 – Intelligent Systems



# Table of Contents

- Exploring WEKA 3
- Using WEKA for Classification
- Using WEKA for Clustering
- Other uses of WEKA





# Exploring WEKA 3

WEKA 3



College of  
Information Technology  
and Computer Science

CENTER OF EXCELLENCE  
in Information Technology

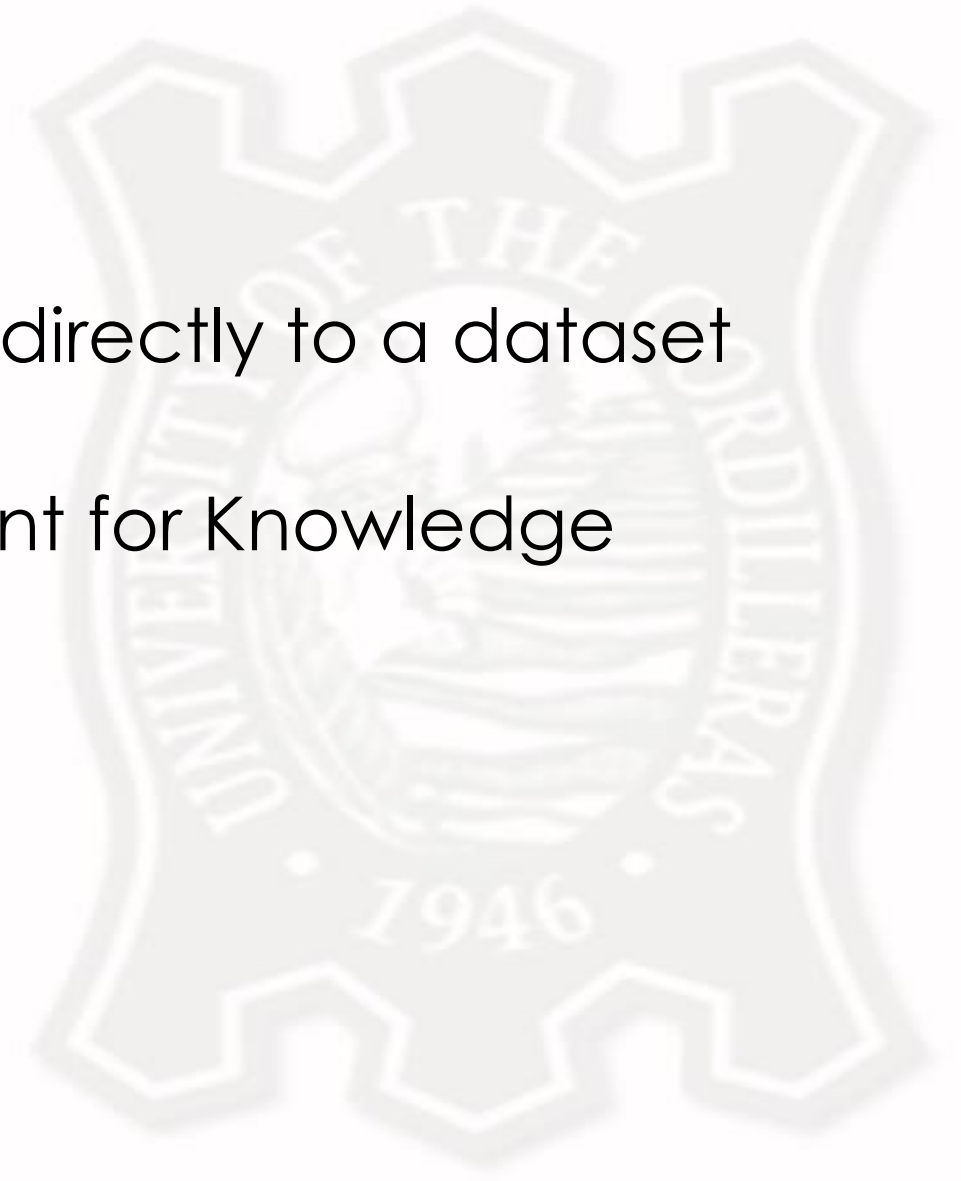
# What is WEKA?

- **WEKA** is an open-source software that provides tools for data preprocessing, machine learning, and visualization.
- It allows you to develop machine learning techniques and apply them to real-world data mining problems.
- **WEKA 3** is the latest version of the software and is developed with the Java programming language.



# What is WEKA?

- Its algorithms can either be applied directly to a dataset or called from a Java code.
- WEKA stands for Waikato Environment for Knowledge Analysis.



# Why use WEKA?

- Why would we want to use WEKA for machine learning?
  - *It is a free open-source tool*
  - *It is portable, since it is implemented in the Java programming language, and thus runs on any Java compatible device*
  - *It has a comprehensive collection of data preprocessing and modeling techniques*
  - *It is easy to use due to its simple user interface*



# Why use WEKA?

- These are some characteristics of the WEKA tool:
  - *WEKA's techniques are predicated on the assumption that the data is available as a single flat file or relation*
  - *Each data point in the file should be described by a fixed number of attributes*
  - *WEKA provides access to SQL databases*



# File Types Supported by WEKA 3

- WEKA uses the **Attribute Relation File Format** for data analysis, by default.
- These are some of the formats that WEKA supports:
  - CSV
  - ARFF
  - *Database using ODBC*

```
@RELATION iris

@ATTRIBUTE sepallength  REAL
@ATTRIBUTE sepalwidth   REAL
@ATTRIBUTE petallength  REAL
@ATTRIBUTE petalwidth   REAL
@ATTRIBUTE class         {Iris-setosa,Iris-versicolor,Iris-virginica}

@DATA
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
5.4,3.7,1.5,0.2,Iris-setosa
4.8,3.4,1.6,0.2,Iris-setosa
4.8,3.0,1.4,0.1,Iris-setosa
4.3,3.0,1.1,0.1,Iris-setosa
5.8,4.0,1.2,0.2,Iris-setosa
5.7,4.4,1.5,0.4,Iris-setosa
5.4,3.9,1.3,0.4,Iris-setosa
5.1,3.5,1.4,0.3,Iris-setosa
5.7,3.8,1.7,0.3,Iris-setosa
5.1,3.8,1.5,0.3,Iris-setosa
5.4,3.4,1.7,0.2,Iris-setosa
5.1,3.7,1.5,0.4,Iris-setosa
4.6,3.6,1.0,0.2,Iris-setosa
```





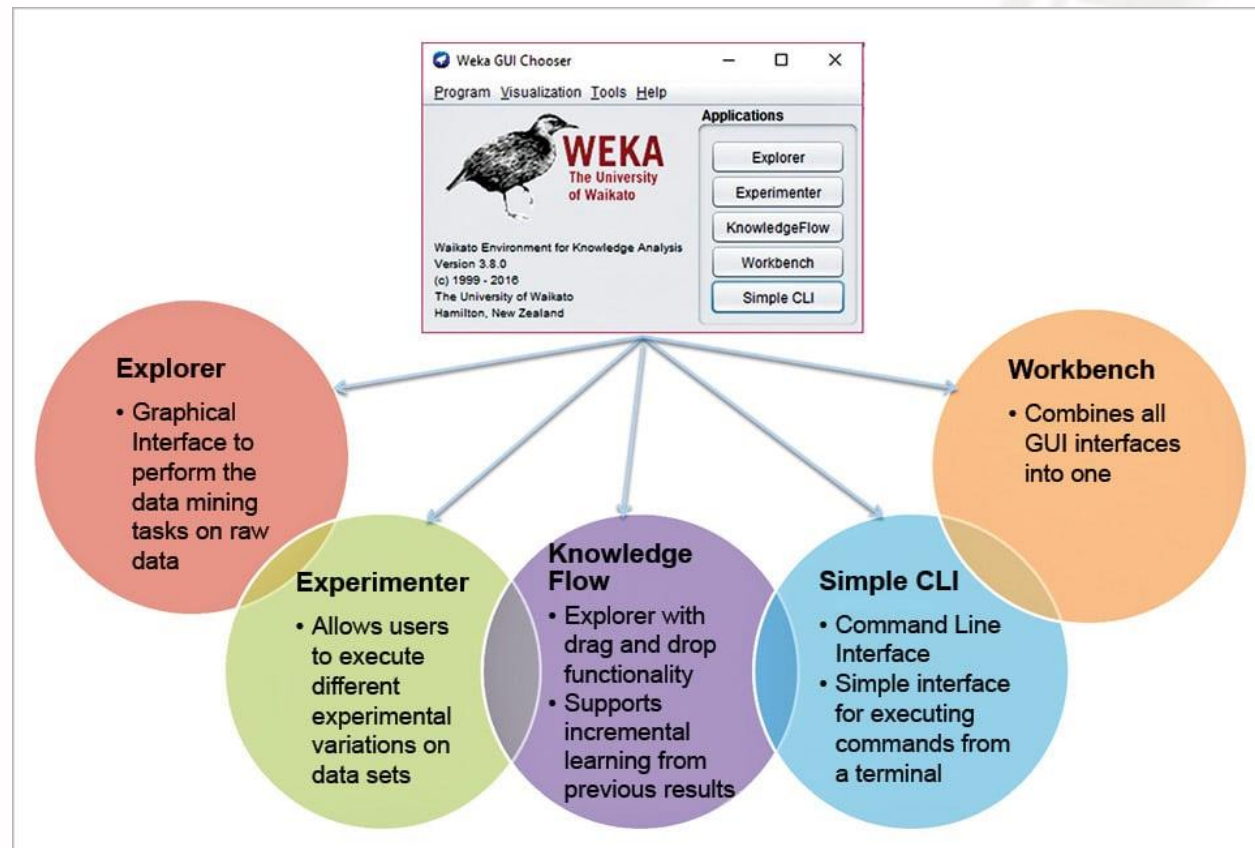
# Installing WEKA

- You can download WEKA from their official website:
  - <https://www.cs.waikato.ac.nz/ml/weka/>
- Make sure that you also install **Java 8 or later** for WEKA to work.
- Once installed, you can then run the application.



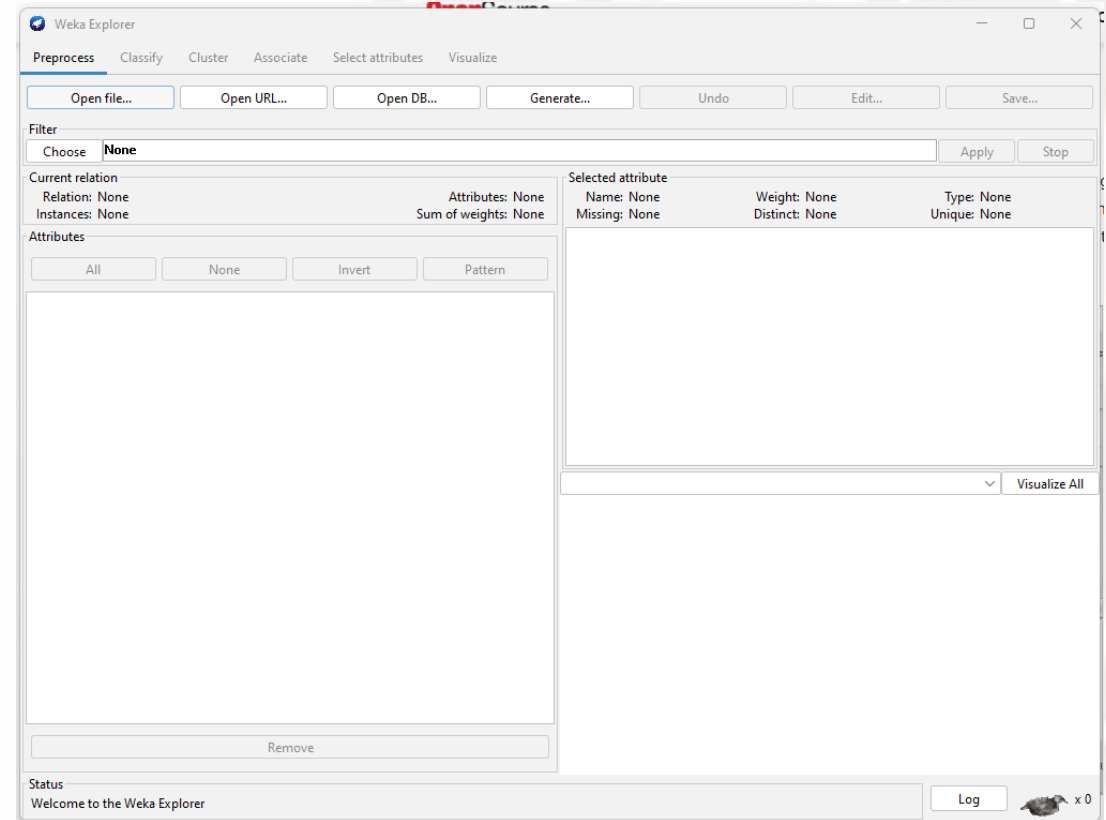
# Exploring the WEKA 3 UI

- Shown below are the parts of the UI for the WEKA software:



# WEKA Explorer

- The main tab that we will be focusing on for the WEKA tool is the **Explorer** tab.
- When clicked, the explorer tab will bring up a new interface, shown on the right.



# WEKA Explorer

- The uses of the different tabs shown are as follows:
  - **Preprocess** – *this allows us to choose the data file*
  - **Classify** – *this allows us to apply and experiment with different algorithms on preprocessed data files*
  - **Cluster** – *this allows us to apply different clustering tools, which identify clusters within the data file*



# WEKA Explorer

- The uses of the different tabs shown are as follows:
  - **Associate** – *this allows us to apply association rules, which identify the association within the data*
  - **Select attributes** – *this allows us to see the changes on the inclusion and exclusion of attributes from the experiment*
  - **Visualize** – *this provides a visualization produced on the data set in a 2D format, in a scatter plot and bar graph output*



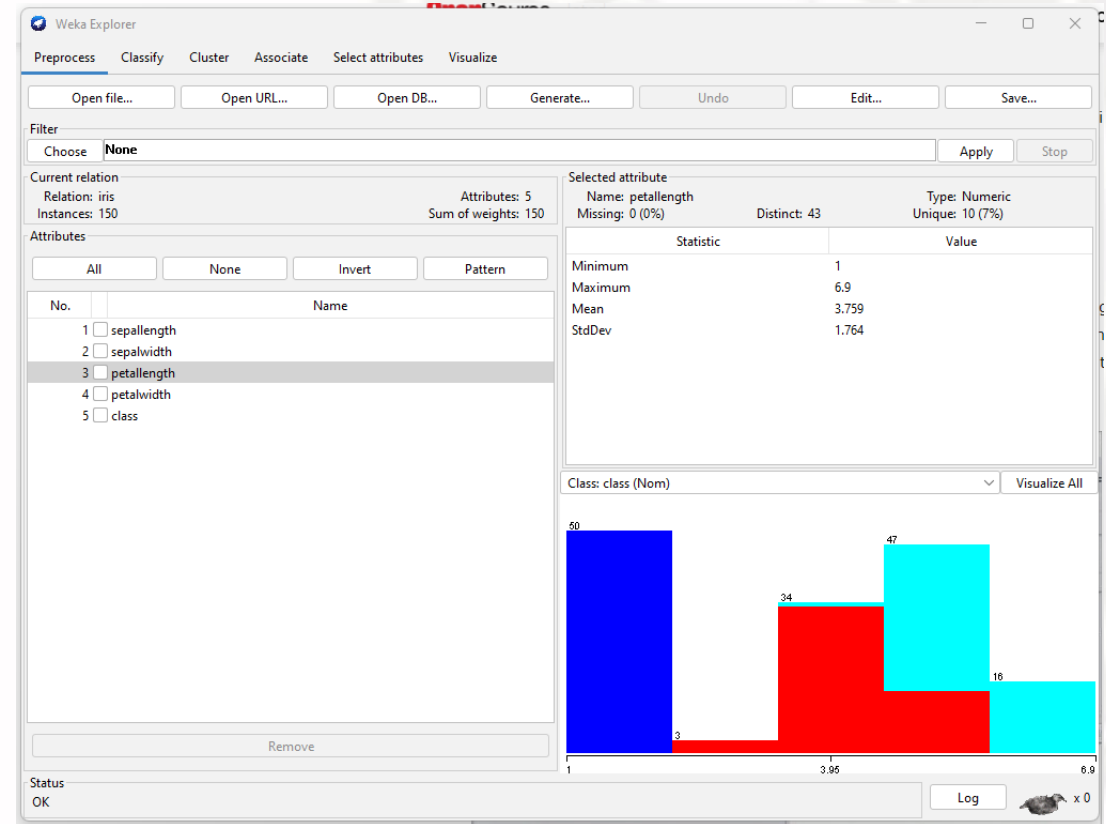
# Preprocessing

- **Preprocessing** refers to manipulating and adjusting data before it is used.
- We can prepare our data for preprocessing in 3 ways:
  - **Open File** – enables the user to select the file from the local machine
  - **Open URL** – enables the user to select the data file from different locations
  - **Open DB** – enables the user to retrieve a data file from a database source



# Preprocessing

- After selecting a file and loading it into the tool, we can then refine the data or use it as is.
- We need to load data first before we can use the other tools in the explorer.





# Using WEKA for Classification

WEKA 3



College of  
Information Technology  
and Computer Science

CENTER OF EXCELLENCE  
in Information Technology



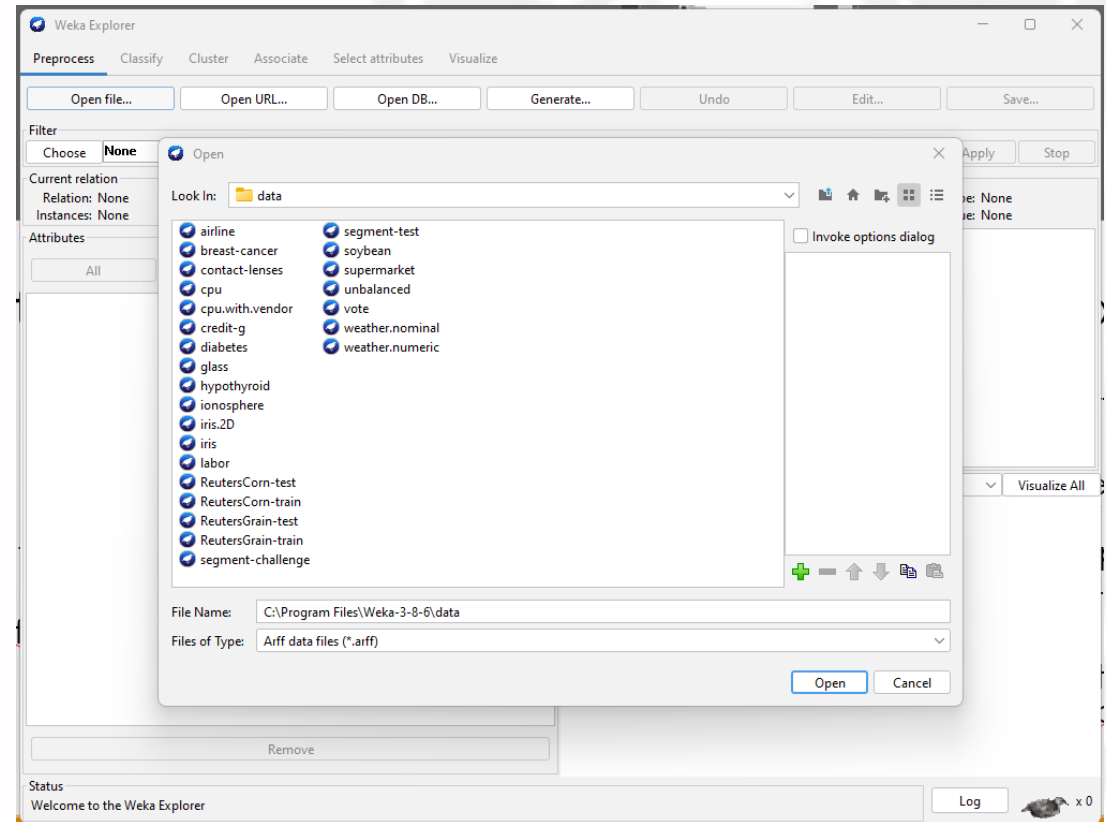
# Classifying with WEKA

- There are many models that we can use for classification in WEKA, such as:
  - ***Decision Trees***
  - ***Support Vector Machines***
  - ***Instance-based Classifiers***
  - ***Logistic Regression***
  - ***Bayes' Nets***



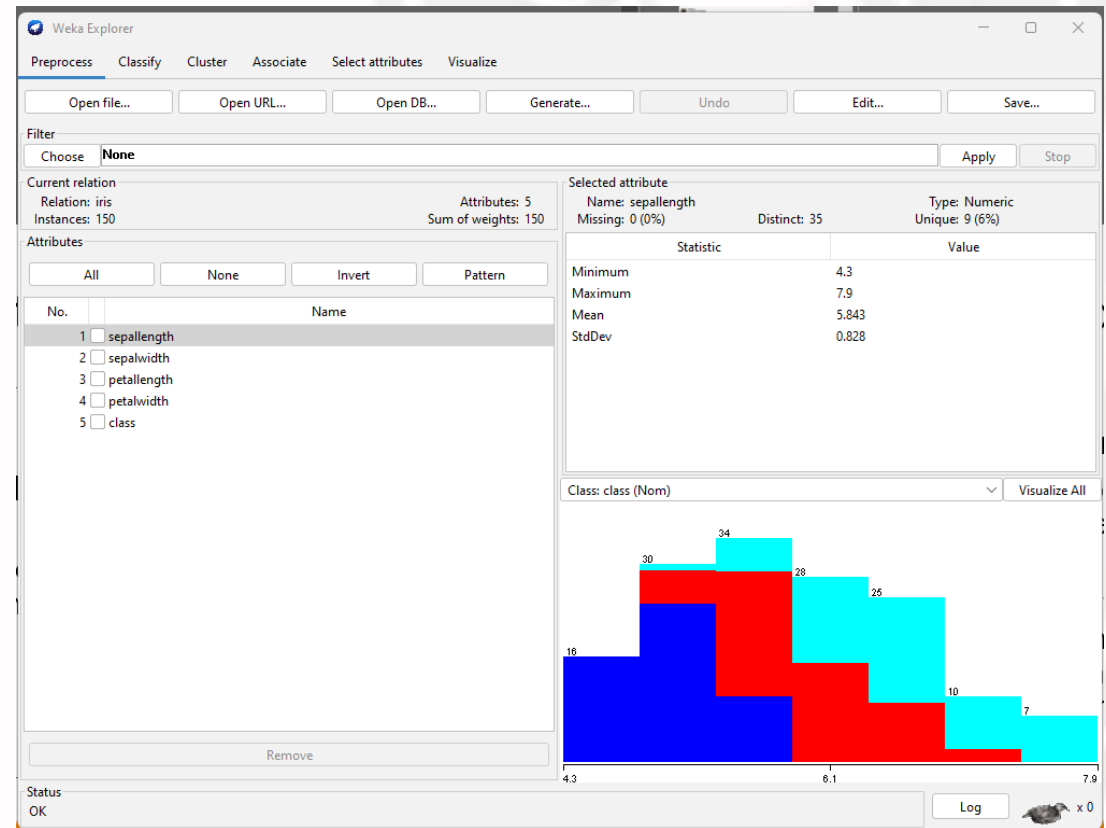
# Preparing our Data

- We start by **loading in the data that we will be classifying.**
- We select the Explorer option in WEKA, and select the preprocess tab.
- We can then select an .arff file or a .csv file.



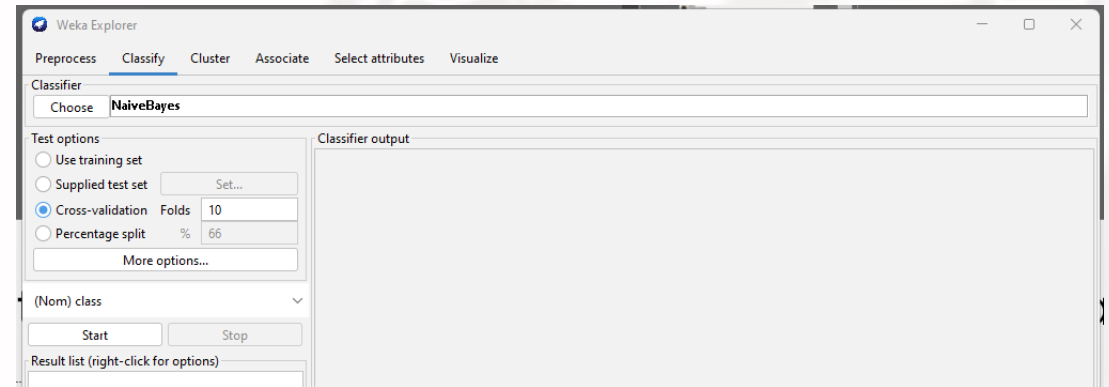
# Preparing our Data

- We will make use of the “iris” data for our example.
- This data has 3 classes, with 50 instances each.
- We will then attempt to see how different classification algorithms perform given this dataset.



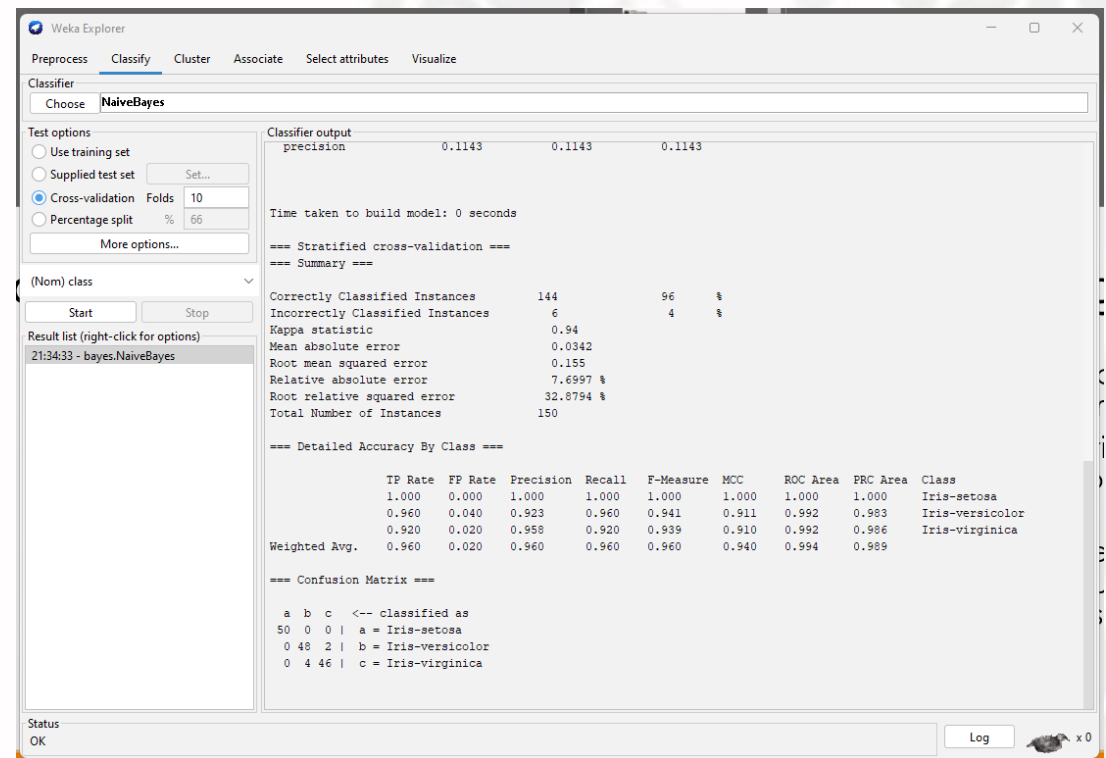
# Preparing our Data

- Once our data has been loaded, let's head over to the “Classify” tab for the next step.
- We are going to make use of the model **NaiveBayes** to classify our data.



# Classifying our Data

- We can then click start to classify our given data.
- After processing our data, we will be shown the number of correctly and incorrectly classified instances.
- It will also give additional metrics, such as the **mean absolute error**.



# Classifying our Data

- Another part of note in the results is the **Confusion Matrix**.
- This serves as an important metric in showing how well the model performs when classifying the different classes.
- It can help reveal biases in the data.

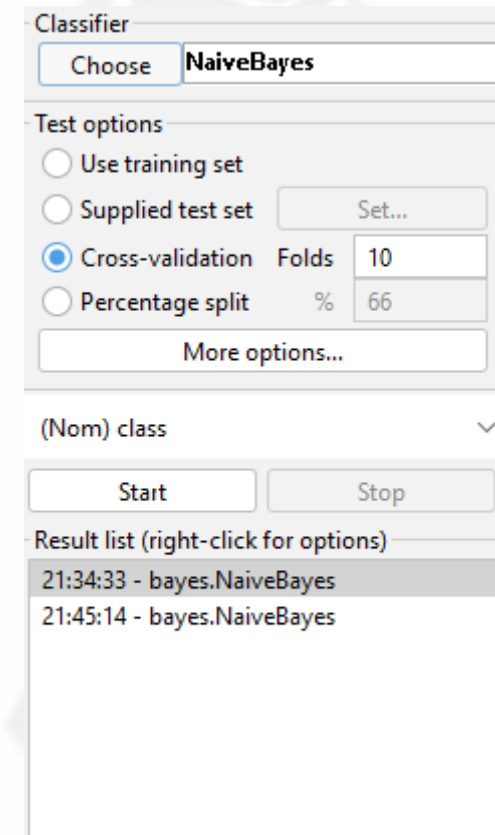
=== Confusion Matrix ===

a	b	c	<-- classified as
50	0	0	a = Iris-setosa
0	48	2	b = Iris-versicolor
0	4	46	c = Iris-virginica



# Testing the Classification Process

- We can also quickly compare the performance of different models by clicking on each result under the **Result list**.
- This allows us to test and evaluate many models on the same data.





# Using WEKA for Clustering

WEKA 3



College of  
Information Technology  
and Computer Science

CENTER OF EXCELLENCE  
in Information Technology



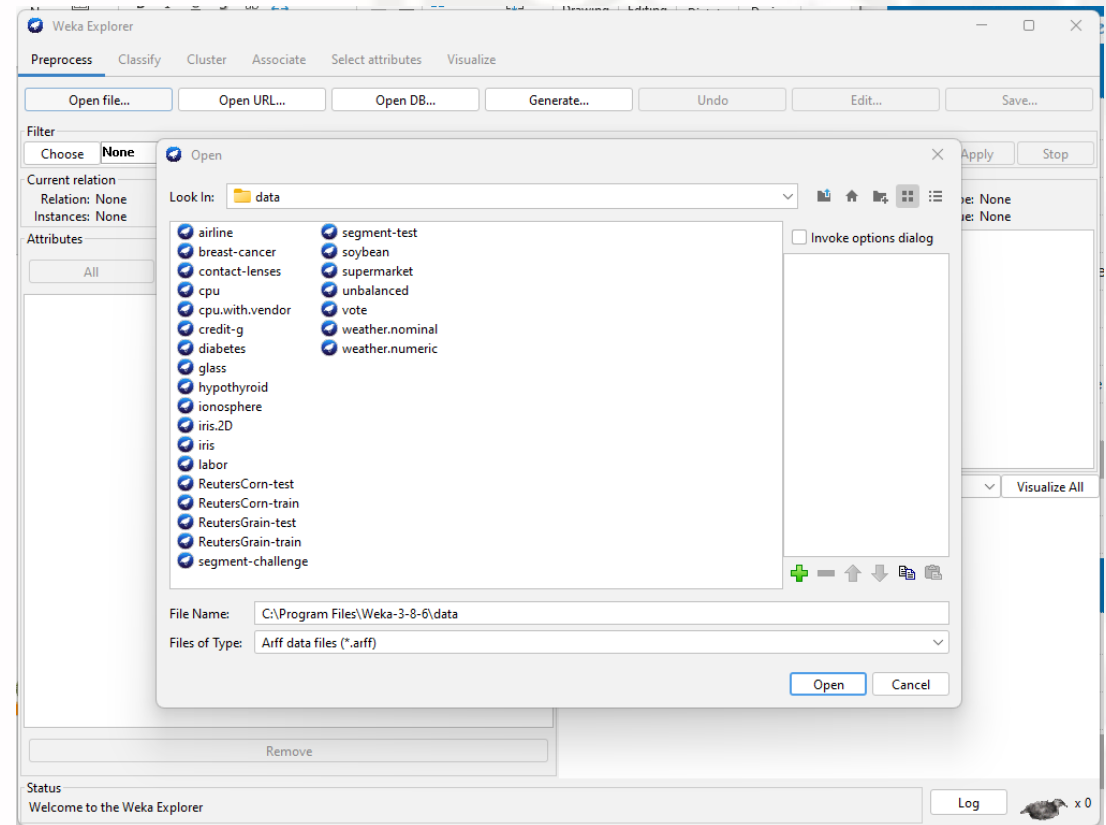
# Clustering with WEKA

- WEKA allows you to use several clustering algorithms such as:
  - ***E-M Algorithm***
  - ***FilteredClusterer***
  - ***HierarchicalClusterer***
  - ***Simple K-Means***
- These clustering algorithms all have their own characteristics and specific uses.



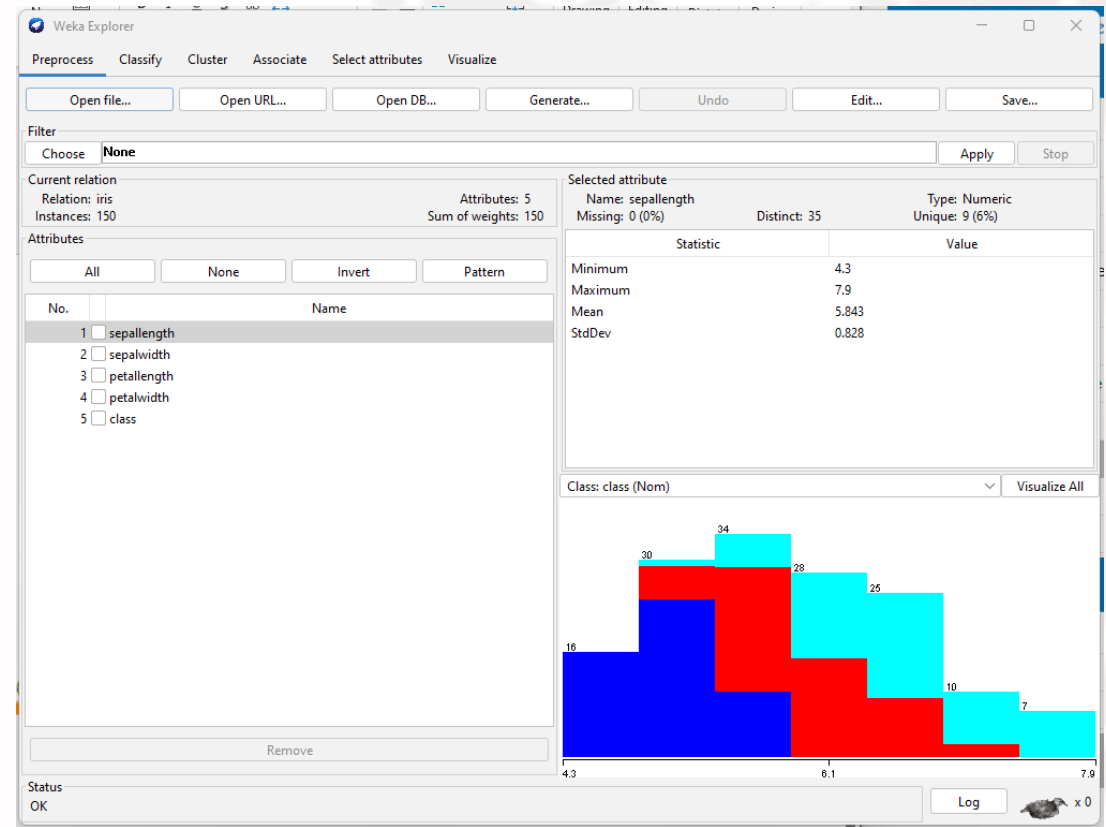
# Preparing our Data

- We again start by **preparing our data to be processed.**
- We select the Explorer option in WEKA, then select the preprocess tab.



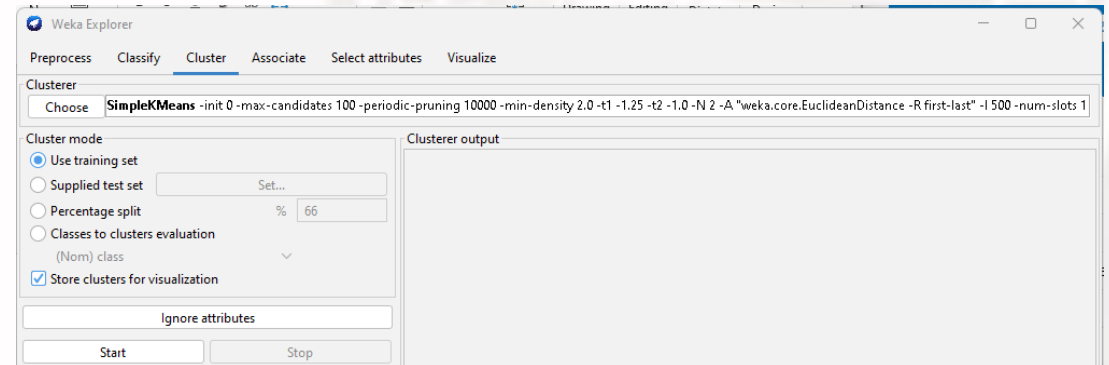
# Preparing our Data

- We will again make use of the “iris” data for our example.
- We know that this data has 3 classes with 50 instances each.
- We will then see how the different clustering algorithms perform when clustering this data.



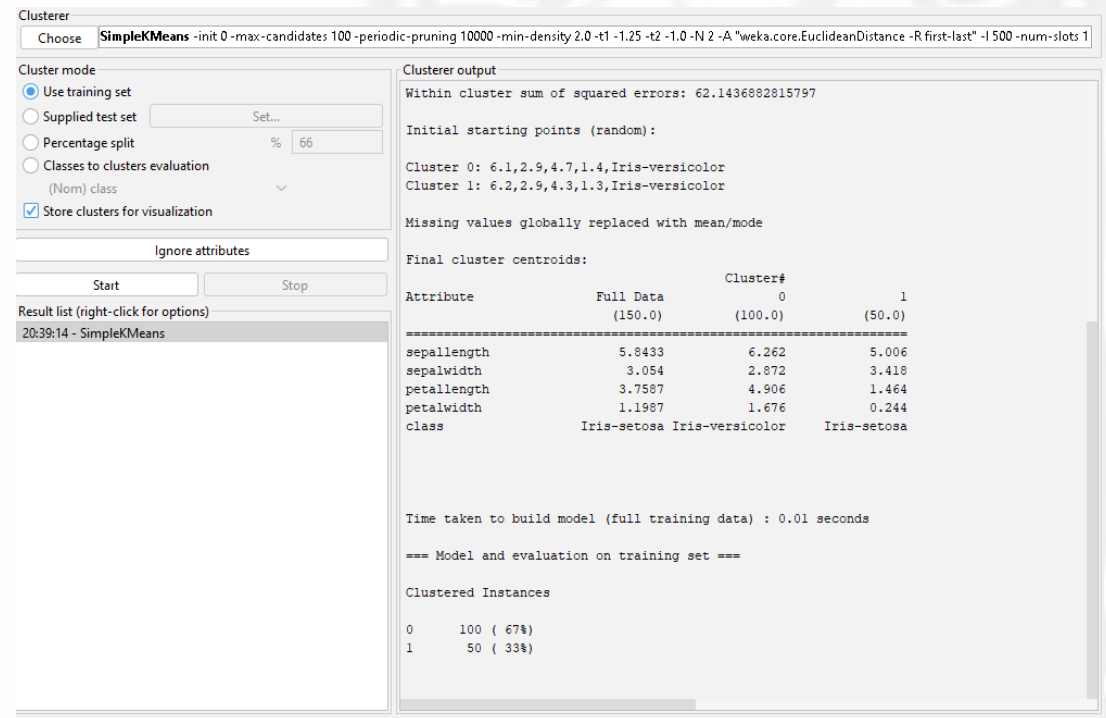
# Preparing our Data

- Once we have loaded our data, we will then select the “Cluster” tab on the header.
- We will then select choose and then select **SimpleKMeans** for our example.



# Clustering our Data

- We can then click start to get our clustering result.
- You might notice that we only have two clusters (0 and 1).
- It would be better if we have three clusters so let us adjust our cluster algorithm.



The screenshot shows the Weka Clusterer window with SimpleKMeans selected. The 'Cluster mode' section has 'Use training set' selected. The 'Cluster output' pane displays the following information:

Within cluster sum of squared errors: 62.1436882815797

Initial starting points (random):

Cluster 0: 6.1,2.9,4.7,1.4,Iris-versicolor  
Cluster 1: 6.2,2.9,4.3,1.3,Iris-versicolor

Missing values globally replaced with mean/mode

Final cluster centroids:

Attribute	Full Data (150.0)	Cluster# 0 (100.0)	1 (50.0)
sepalength	5.8433	6.262	5.006
sepalwidth	3.054	2.872	3.418
petallength	3.7587	4.906	1.464
petalwidth	1.1987	1.676	0.244
class		Iris-setosa Iris-versicolor	Iris-setosa

Time taken to build model (full training data) : 0.01 seconds

=== Model and evaluation on training set ===

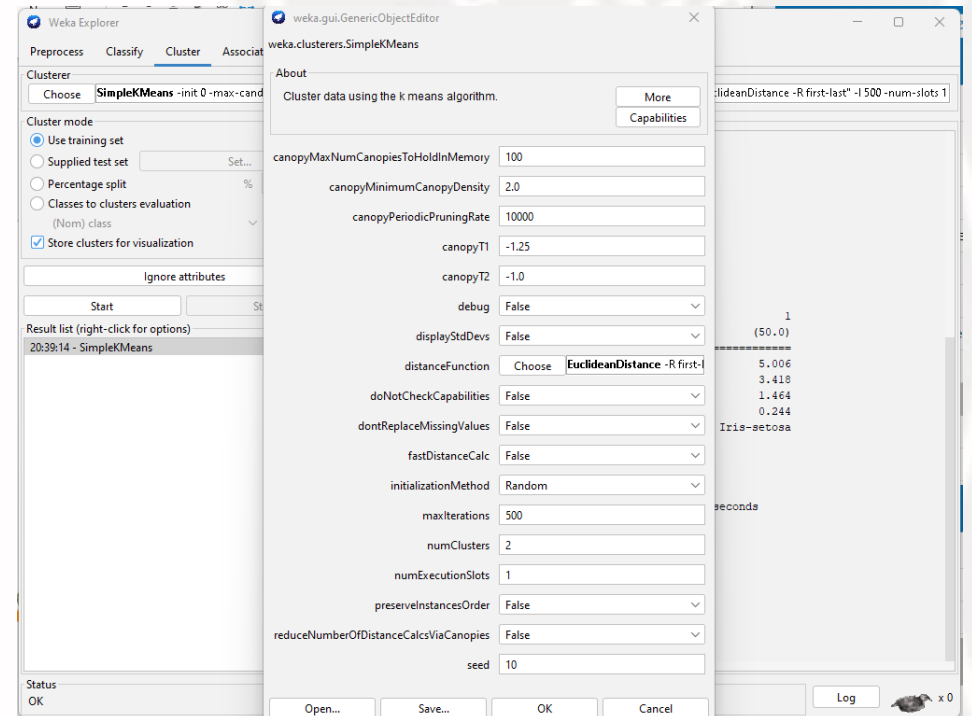
Clustered Instances

Cluster	Count	Percentage
0	100	67%
1	50	33%



# Clustering our Data

- We can access the parameters of the cluster algorithm by right clicking or double clicking it.
- This will allow us to adjust our cluster as needed.
- In this case, we will change the numbers of clusters from 2 to 3.



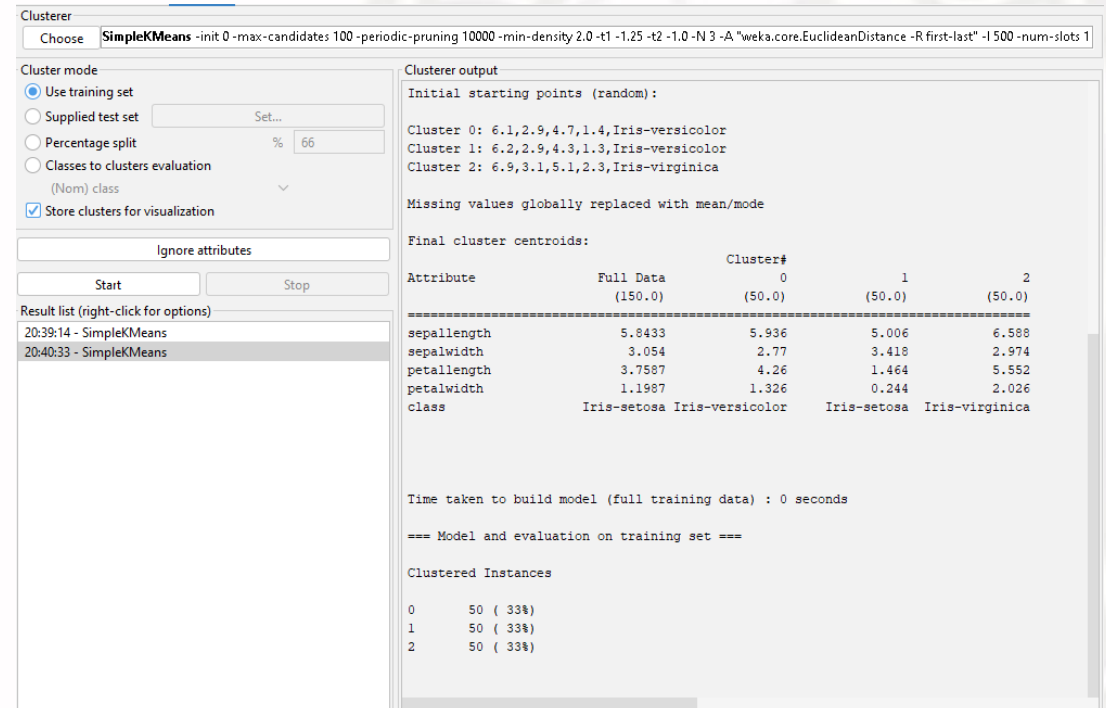
numClusters

3



# Clustering our Data

- We can see that we now have 3 clusters for our data.
- It seems like the algorithm perfectly clustered the three types of flowers.
- There is a problem here, but we will fix it later.



The screenshot shows the Weka Clustering tool interface. The 'Clusterer' dropdown is set to 'SimpleKMeans'. The 'Cluster mode' section has 'Use training set' selected. The 'Cluster output' pane displays the following information:

Initial starting points (random):

- Cluster 0: 6.1,2.9,4.7,1.4,Iris-versicolor
- Cluster 1: 6.2,2.9,4.3,1.3,Iris-versicolor
- Cluster 2: 6.9,3.1,5.1,2.3,Iris-virginica

Missing values globally replaced with mean/mode

Final cluster centroids:

Attribute	Full Data (150.0)	Cluster# 0 (50.0)	1 (50.0)	2 (50.0)
sepalength	5.8433	5.936	5.006	6.588
sepalwidth	3.054	2.77	3.418	2.974
petallength	3.7587	4.26	1.464	5.552
petalwidth	1.1987	1.326	0.244	2.026
class	Iris-setosa Iris-versicolor	Iris-setosa	Iris-virginica	

Time taken to build model (full training data) : 0 seconds

=== Model and evaluation on training set ===

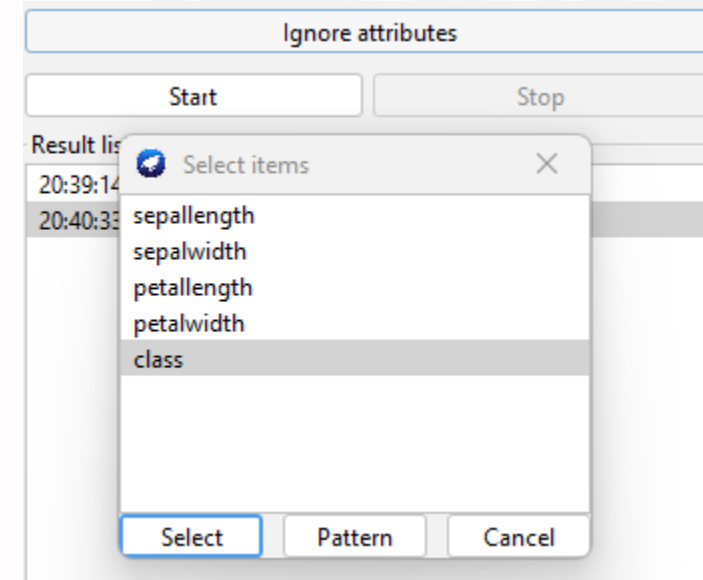
Clustered Instances

0	50 ( 33%)
1	50 ( 33%)
2	50 ( 33%)



# Improving our Clusters

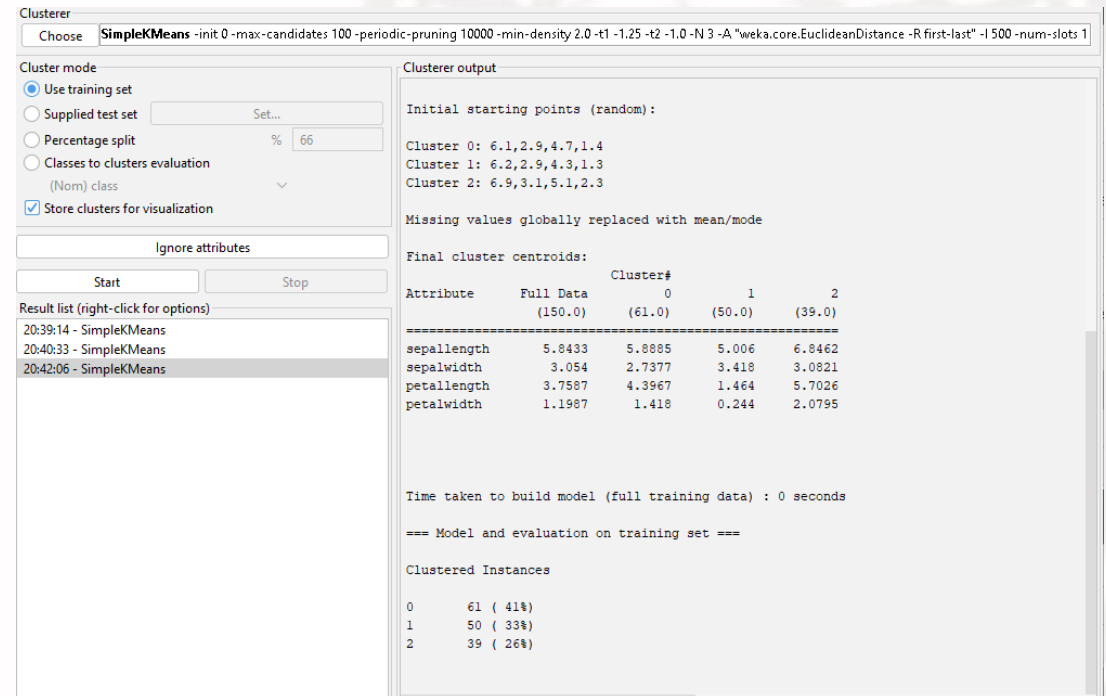
- Now, we need to make some adjustments to our clusters to make it more “realistic”.
- We were getting perfect results due to a “class” attribute which specified the class for our data.
- We will need to ignore this attribute when clustering our data.





# Improving our Clusters

- After ignoring the “class” attribute, we will then cluster our data again.
- We can now see that we have more realistic results for our algorithm.
- We can then choose to visualize this or cluster it with a different algorithm.



The screenshot shows the Weka Clusterer window with SimpleKMeans selected. The 'Cluster mode' section has 'Use training set' selected. The 'Cluster output' section displays the following information:

Initial starting points (random):

Cluster 0: 6.1,2.9,4.7,1.4  
Cluster 1: 6.2,2.9,4.3,1.3  
Cluster 2: 6.9,3.1,5.1,2.3

Missing values globally replaced with mean/mode

Final cluster centroids:

Attribute	Full Data (150.0)	Cluster# 0 (61.0)	1 (50.0)	2 (39.0)
sepal.length	5.8433	5.8885	5.006	6.8462
sepal.width	3.054	2.7377	3.418	3.0821
petal.length	3.7587	4.3967	1.464	5.7026
petal.width	1.1987	1.418	0.244	2.0795

Time taken to build model (full training data) : 0 seconds

=== Model and evaluation on training set ===

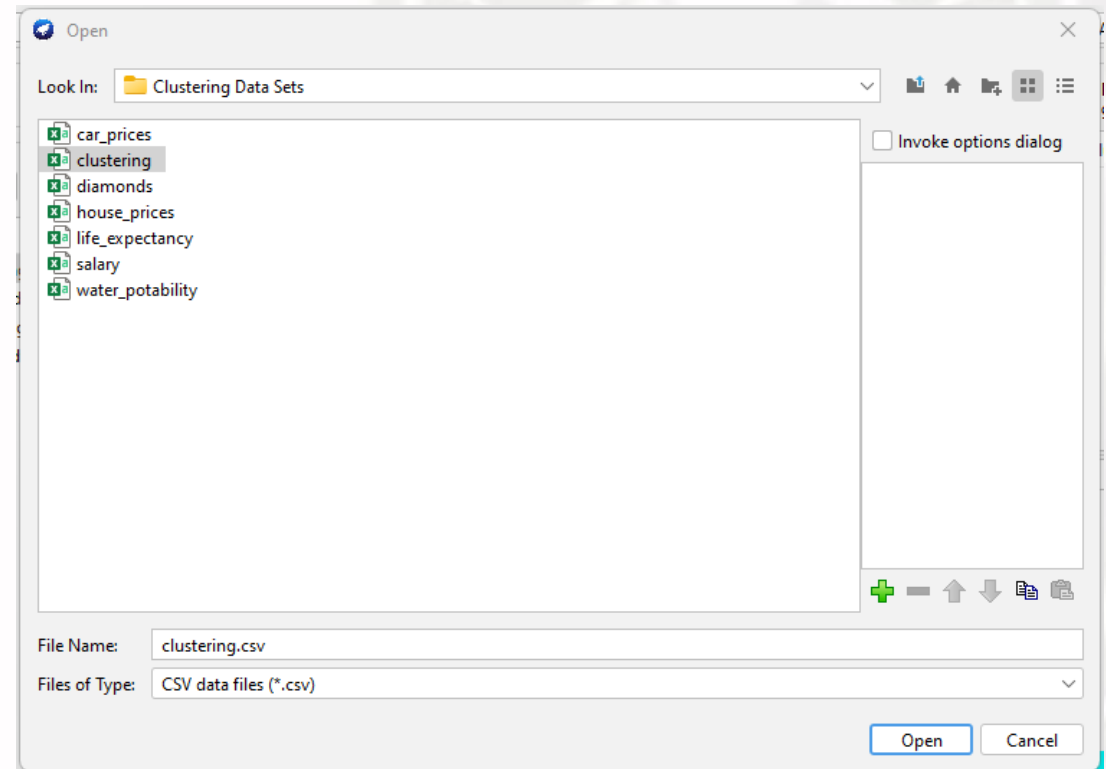
Clustered Instances

Cluster	Count	Percentage
0	61	(41%)
1	50	(33%)
2	39	(26%)



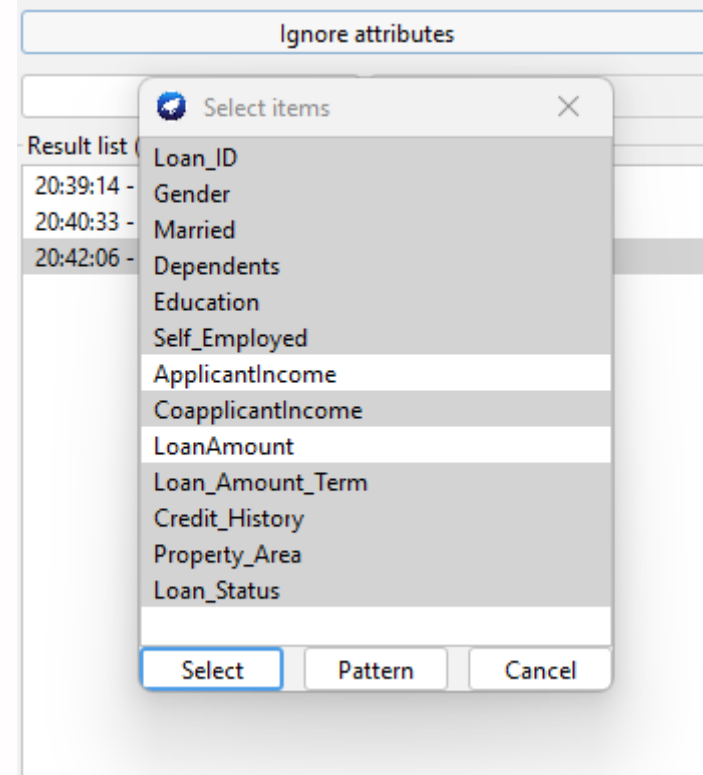
# Clustering other file types

- We can also cluster other file types by selecting to appropriate file type in the preprocess menu.
- This allows us to cluster spreadsheets and other types of data.



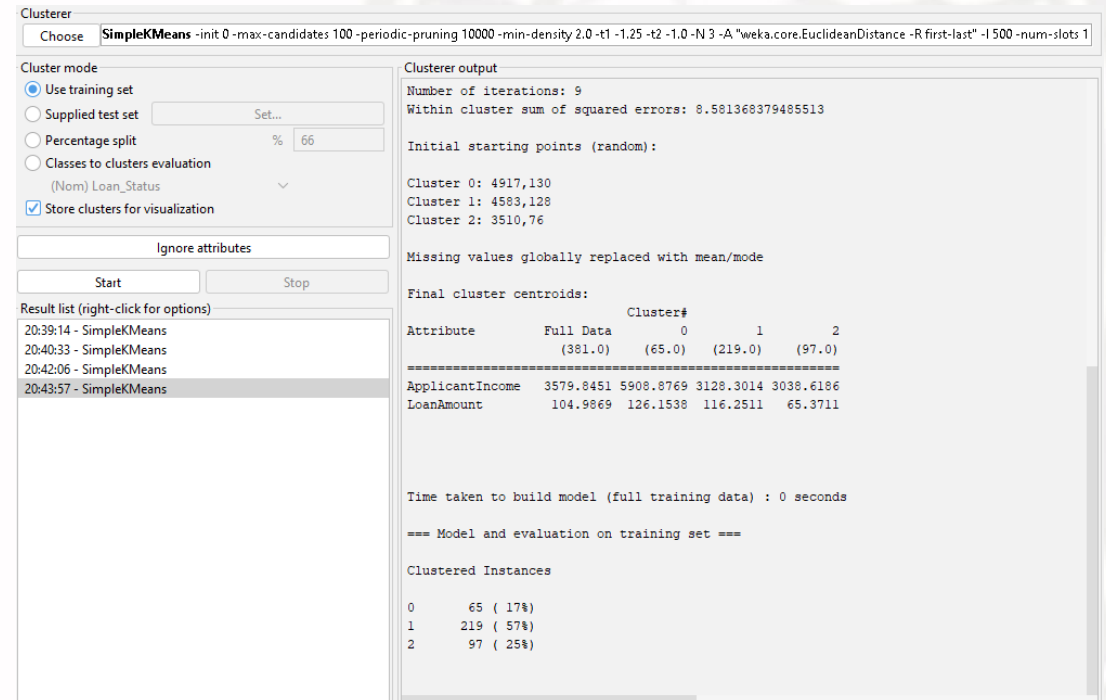
# Clustering other file types

- In the case of the clustering.csv file, we will open it into WEKA.
- We will then ignore the other attributes since most of them are text and will not help with the clustering process.



# Clustering other file types

- We can now see how the algorithm clusters the data using **SimpleKMeans** with 3 clusters.
- We can adjust the algorithm or number of clusters and compare the results.



The screenshot shows the Weka Clusterer window. The 'Choose' dropdown is set to 'SimpleKMeans'. The 'Cluster mode' section has 'Use training set' selected. The 'Store clusters for visualization' checkbox is checked. The 'Result list' on the left shows four entries for SimpleKMeans, with the last one (20:43:57) selected. The 'Cluster output' pane on the right displays the following information:

Clusterer: SimpleKMeans -init 0 -max-candidates 100 -periodic-pruning 10000 -min-density 2.0 -t1 -1.25 -t2 -1.0 -N 3 -A "weka.core.EuclideanDistance -R first-last" -I 500 -num-slots 1

Cluster mode:  
☒ Use training set  
☐ Supplied test set (Set...)  
☐ Percentage split (% 66)  
☐ Classes to clusters evaluation (Nom) Loan\_Status  
☒ Store clusters for visualization

Ignore attributes: [ ]  
Start [ ] Stop [ ]

Result list (right-click for options):  
20:39:14 - SimpleKMeans  
20:40:33 - SimpleKMeans  
20:42:06 - SimpleKMeans  
20:43:57 - SimpleKMeans

Cluster output:  
Number of iterations: 9  
Within cluster sum of squared errors: 8.581368379485513  
Initial starting points (random):  
Cluster 0: 4917,130  
Cluster 1: 4583,128  
Cluster 2: 3510,76  
Missing values globally replaced with mean/mode  
Final cluster centroids:  
Attribute Full Data Cluster# 0 1 2  
(381.0) (65.0) (219.0) (97.0)  
=====

Attribute	Full Data	Cluster# 0	Cluster# 1	Cluster# 2
ApplicantIncome	3579.8451	5908.8769	3128.3014	3038.6186
LoanAmount	104.9869	126.1538	116.2511	65.3711

Time taken to build model (full training data) : 0 seconds  
=== Model and evaluation on training set ===  
Clustered Instances  
0 65 ( 17%)  
1 219 ( 57%)  
2 97 ( 25%)



# Clustering other file types

- We can then visualize our clusters with the same steps as before by right clicking the results.
- This will allow us to see how the algorithm clustered the given data in our data set.



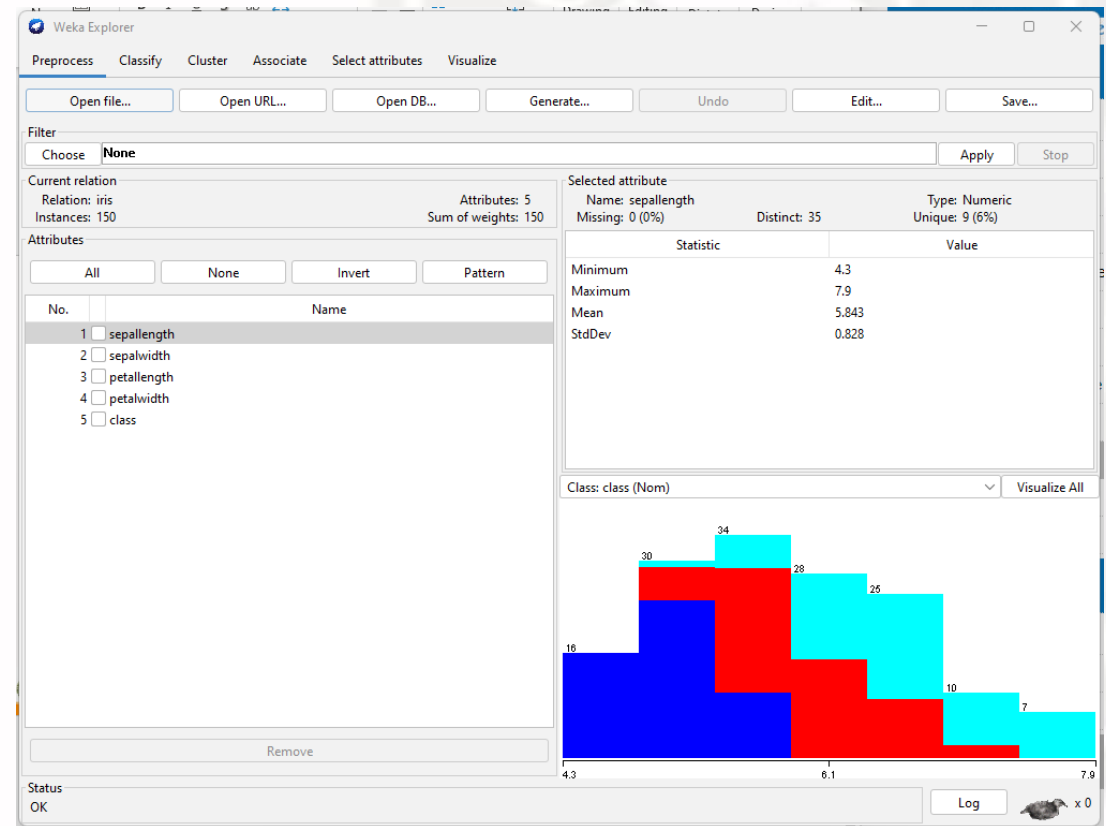
# Classes-to-clusters Evaluation

- **Classes-to-clusters Evaluation** is a tool used to evaluate the accuracy of a given clustering algorithm.
- In this mode, WEKA ignores the class attribute and generates the clusters.
- It then assigns the classes to the clusters and computes the errors based on mismatching clusters and classes.
- It then provides a value for the number of incorrectly clustered instances.



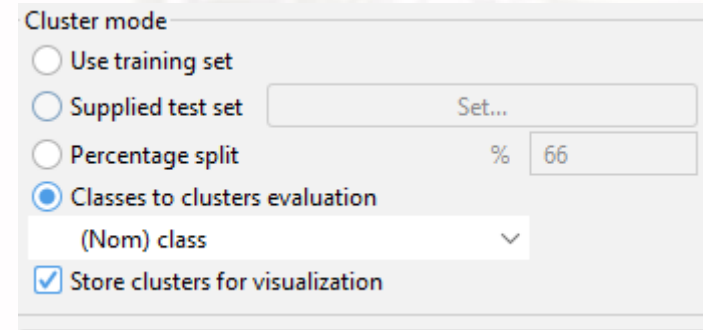
# Classes-to-clusters Evaluation

- We can make use of an option in our cluster to evaluate the **accuracy of the clusters**.
- This will allow us to use the given classes in our data to check the accuracy of our clusters.
- Let us make use of the “iris” dataset for this example.



# Classes-to-clusters Evaluation

- We will head to the “Cluster” page and select the “Classes to clusters evaluation”.
- We will then select class in the dropdown box.
- This will specify that attribute as the class.



Cluster mode

☐ Use training set

☐ Supplied test set

☐ Percentage split %

☒ Classes to clusters evaluation

(Nom) class

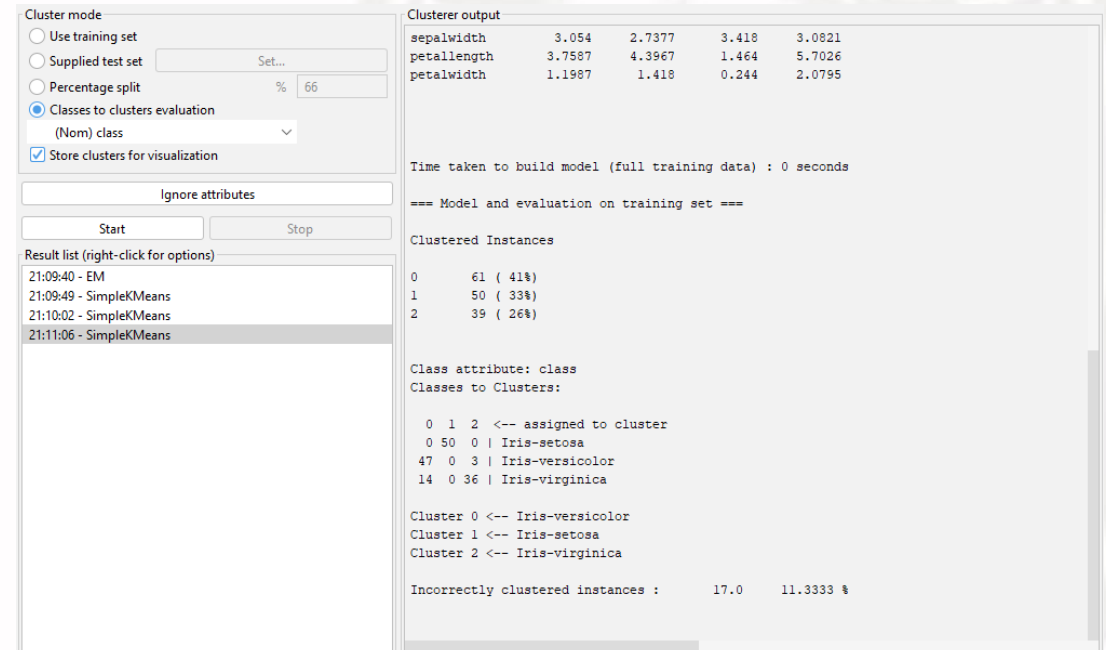
☒ Store clusters for visualization





# Classes-to-clusters Evaluation

- We can now see the results of the evaluation.
- This showcases how accurate the clustering is and what specific clusters had the greatest number of errors.
- Take note that this method does not work without a proper class attribute.



The screenshot displays the Orange3 software interface for the 'Classes to clusters evaluation' widget. The 'Cluster mode' section on the left has 'Classes to clusters evaluation' selected, with '(Nom) class' chosen from the dropdown and 'Store clusters for visualization' checked. The 'Result list' on the left shows four entries: '21:09:40 - EM', '21:09:49 - SimpleKMeans', '21:10:02 - SimpleKMeans', and '21:11:06 - SimpleKMeans', with the last one selected. The 'Clusterer output' panel on the right shows a table of feature values for three clusters, followed by model evaluation statistics and a mapping of clusters to Iris species.

	sepalwidth	petallength	petalwidth
0	3.054	2.7377	3.418
1	3.7587	4.3967	1.464
2	1.1987	1.418	0.244

Time taken to build model (full training data) : 0 seconds  
=== Model and evaluation on training set ===

Clustered Instances

Cluster	Instances	Percentage
0	61	( 41%)
1	50	( 33%)
2	39	( 26%)

Class attribute: class  
Classes to Clusters:

Cluster	Class
0	Iris-versicolor
1	Iris-setosa
2	Iris-virginica

Incorrectly clustered instances : 17.0 11.3333 %





# Other uses of WEKA

WEKA 3



College of  
Information Technology  
and Computer Science

CENTER OF EXCELLENCE  
in Information Technology

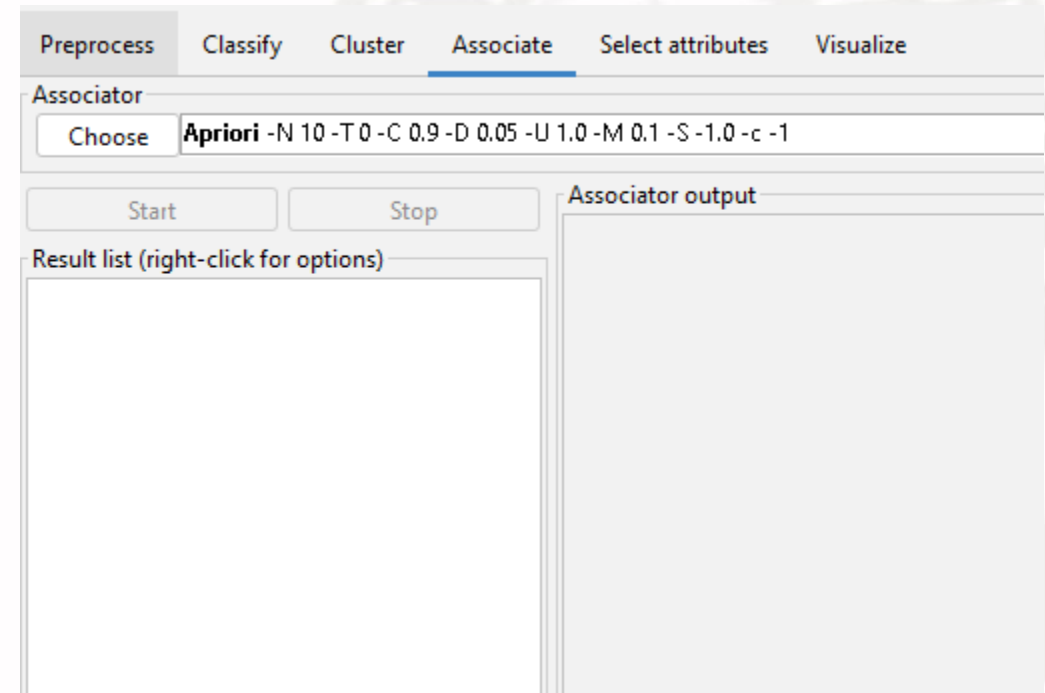
# Other Features of WEKA

- Aside from classification and clustering, there are other tools that you can use in WEKA to process data.
- These are some of the other tools available in WEKA:
  - **Association**
  - **Attribute Selection**
  - **Visualization**



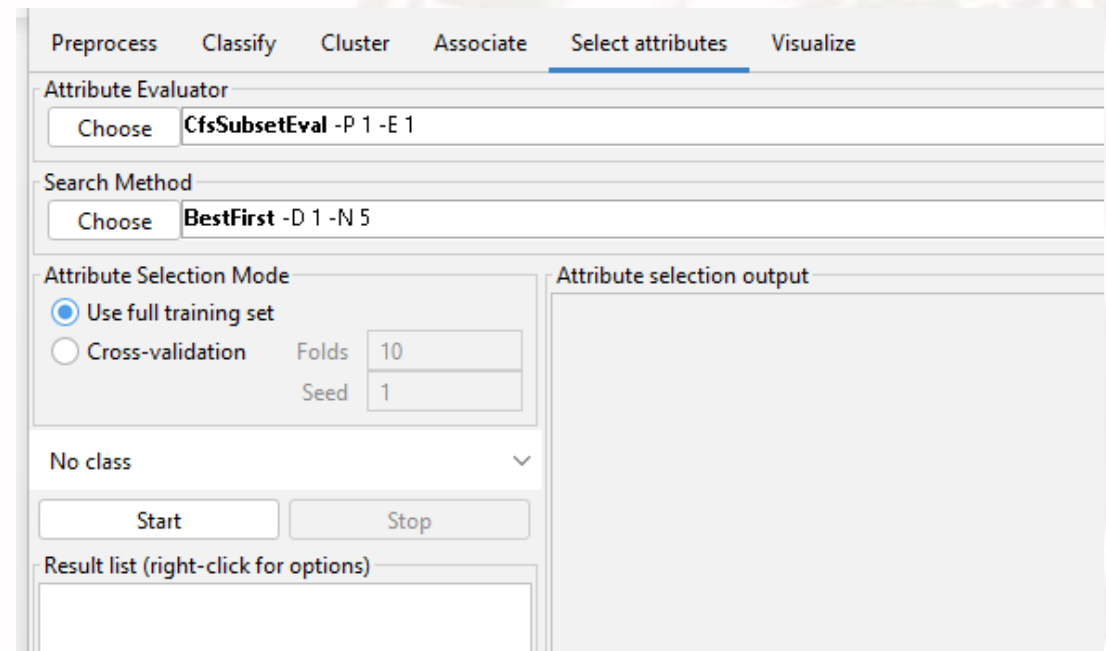
# Association

- **Association** is an unsupervised learning technique that looks for relationships between variables or features in a dataset.
- The main model available in WEKA for association is the Apriori algorithm.



# Attribute Selection

- **Attribute Selection** is a method of choosing the best features for use in a machine learning model.
- This is done to remove redundant or irrelevant data from the dataset.



# Visualization

- **Visualization** allows us to view the different data in a 2D format.
- We can visualize single attributes (1D) and pairs of attributes (2D) in WEKA.

