

EXPERIMENT1

Aim: Demonstrate the following data preprocessing tasks.

- ❖ Loading the dataset
- ❖ Adding Attribute
- ❖ Add expression
- ❖ Copy attribute
- ❖ Remove attribute

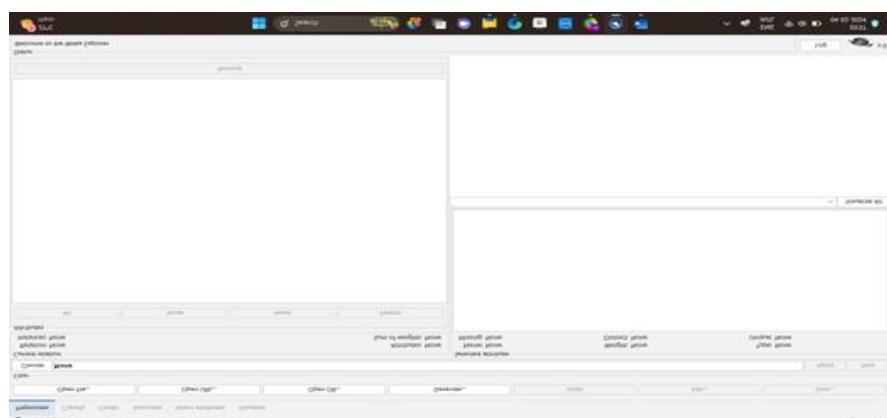
Description: WEKA - an open source software provides tools for data preprocessing, implementation of several Machine Learning algorithms, and visualization tools so that you can develop machine learning techniques and apply them to real-world data mining problems

1) Loading the dataset:

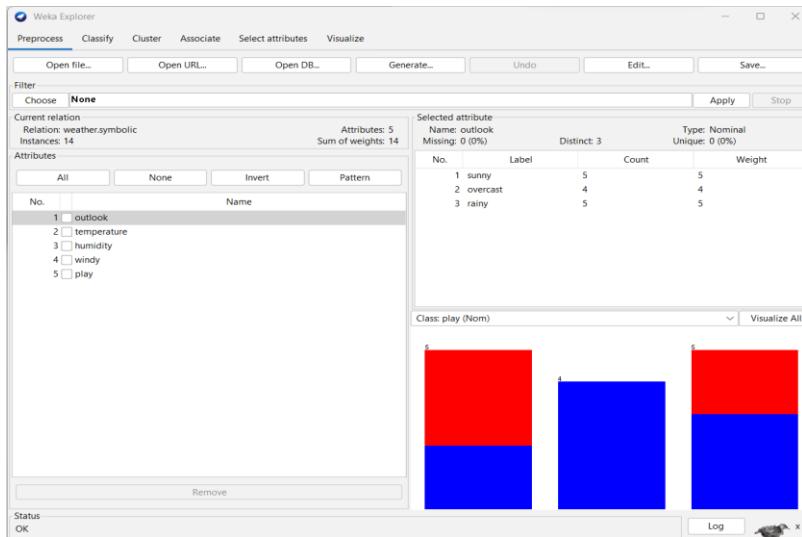
- Go to weka 3.8.6



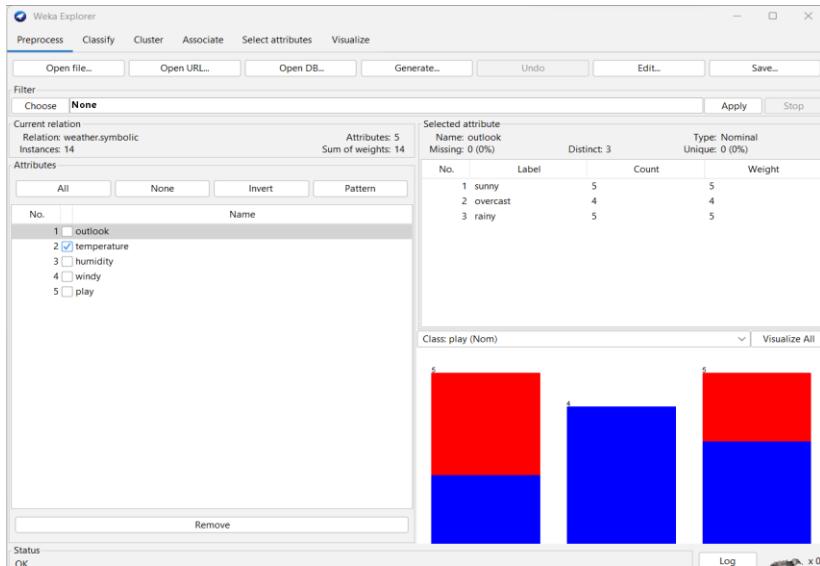
- Select "Explorer."



- Click on "Open File", then select "Look in" and navigate to Local Disk (C:) -> Program Files -> Weka-3-8-6 -> data -> weather.nominal (any data can be selected). Finally, click on "Open."
- The below data is about the “outlook data”



- Select or tick the "Temperature" box. It should appear as follows:



2) Adding Attribute:

- Go to excel and create the excel with some data and save it as .csv(delimiter)

Sno	Item	No of products	Price
1	Laptop	2	800
2	Smartphone	1	300
3	Tablet	3	200
4	Headphone	1	100
5	Bluetooth	2	150

- Go to 'Open File'. This is where the Excel file is saved. Provide the file name and location. The file type should be 'csv'. Click 'Open'.

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Open file... Open URL... Open DB... Generate... Undo Edit... Save...

Filter Choose Add -N unnamed -C last -W 1.0

Current relation Relation 1 Instances: 5 Attributes: 4 Sum of weights: 5

Attributes All None Invert Pattern

No. Name

1 Sno
2 Item
3 No of products
4 Price

Selected attribute Name: Sno Missing: 0 (0%) Distinct: 5 Type: Numeric Unique: 5 (100%)

Statistic Value

Minimum 1
Maximum 5
Mean 3
StdDev 1.581

Class: Price (Num) Visualize All

Remove

Status OK Log x 0

Information

NAME
weka.filters.unsupervised.attribute.Add

SYNOPSIS
An instance filter that adds a new attribute to the dataset. The new attribute will contain all missing values.

OPTIONS

nominalLabels -- The list of value labels (nominal attribute creation only). The list must be comma-separated, e.g: "red,green,blue". If this is empty, the created attribute will be numeric.

debug -- If set to true, filter may output additional info to the console.

attributeName -- Set the new attribute's name.

attributeIndex -- The position (starting from 1) where the attribute will be inserted (first and last are valid indices).

doNotCheckCapabilities -- If set, the filter's capabilities are not checked before it is built. (Use with caution to reduce runtime.)

weight -- The weight for the new attribute.

dateFormat -- The format of the date values (see ISO-8601).

attributeType -- Defines the type of the attribute to generate.

- Select choose ->filter->unsupervised->attribute->ADD and click ok.
- we have an option labeled 'Add-n'. Click on that. It will prompt you with 'Attribute Index': choose from 'Last', 'First', or 'Middle'. Next, input the 'AttributeName' which is the name of the attribute.

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Open file... Open URL... Open DB... Generate... Undo Edit... Save...

Filter Choose Add -N Total-C last -W 1.0

Current relation Relation 1 weka.filters.unsupervised.attribute.Add -N Total-Clst-W1.0-weka.filters.unsupervised.attribute.Remove-R5 Instances: 5 Attributes: 4 Selected attribute Name: Sno

weka.filters.unsupervised.attribute.Add

About An instance filter that adds a new attribute to the dataset.

More Capabilities

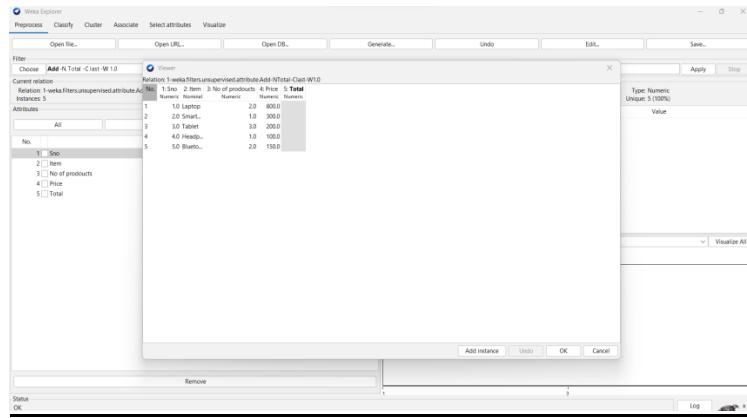
attributeIndex: last
attributeName: Total
attributeType: Numeric attribute
dateFormat: yyyy-MM-dd'T'HHmmss
debug: False
doNotCheckCapabilities: False
nominalLabels:
weight: 1.0

Open... Save... OK Cancel

Remove

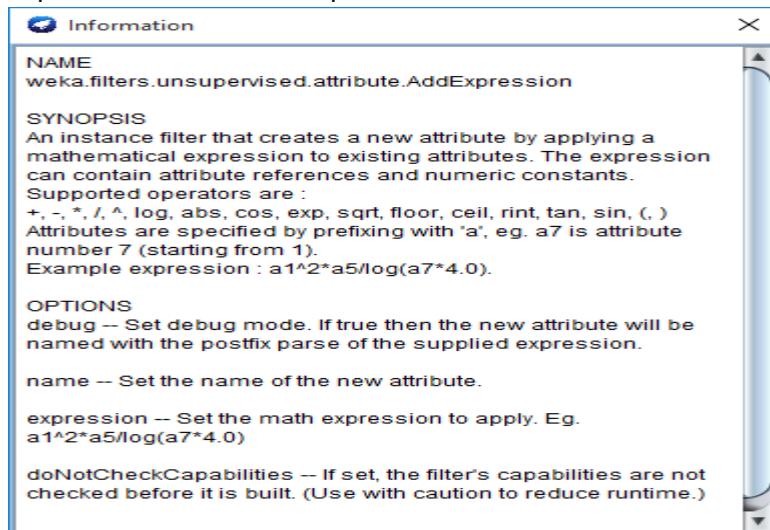
Status OK Log x 0

- After that, click on 'Apply' and then 'Edit'. Finally, it will display as follows

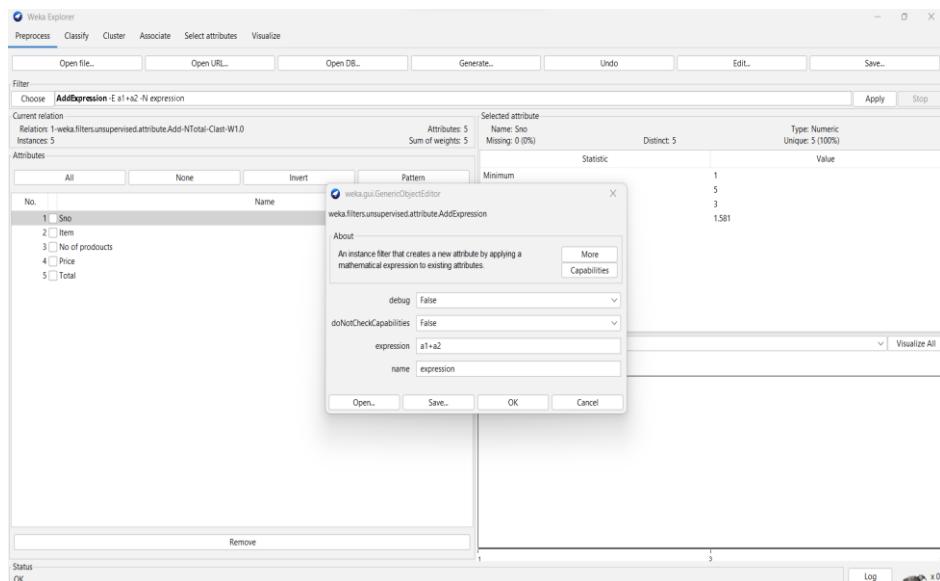


3)Add Expression:

- "To add an expression, follow the same steps as adding an attribute. Select 'Choose', then navigate to 'Filter', followed by 'Unsupervised'. Next, select 'Attribute' and then choose 'Add Expression'. Click 'OK' to proceed."



- Add Expression name as "a3+a4" and Name as "Total".



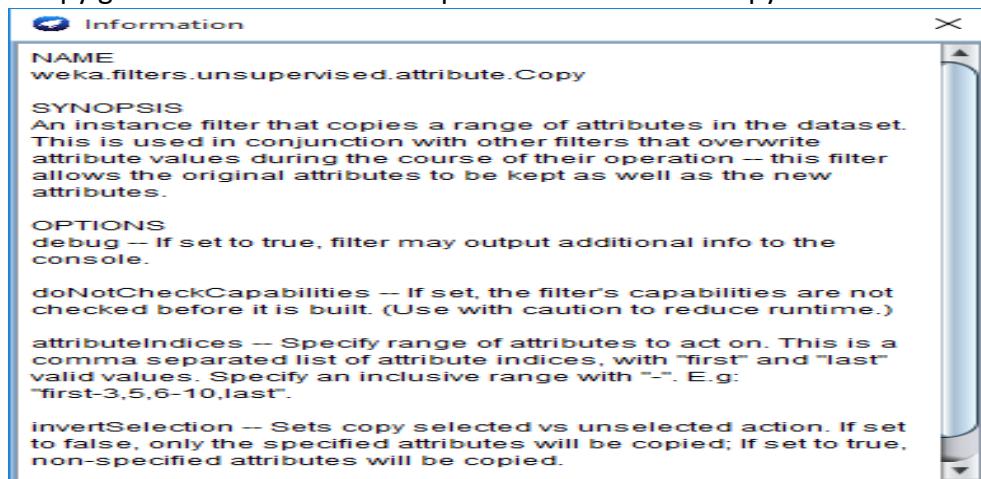
- Click the “Apply” button to apply the filter and edit

The screenshot shows the Weka Explorer interface with a dataset loaded. The dataset has five rows and five columns. The columns are labeled: No., Sno, Item, No. of products, Price, and Total. The data is as follows:

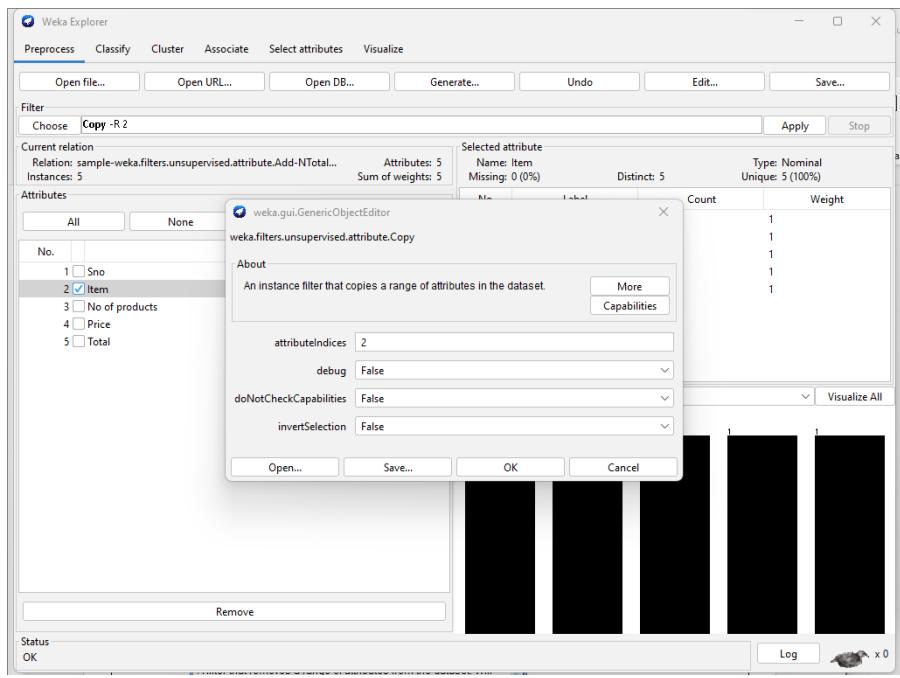
No.	Sno	Item	No. of products	Price	Total
1	1.0	Laptop	2.0	800.0	1600.0
2	2.0	Smart...	1.0	300.0	300.0
3	3.0	Tablet	3.0	200.0	600.0
4	4.0	Head...	1.0	100.0	100.0
5	5.0	Bluet...	2.0	150.0	300.0

4)Copy attribute:

- For copy go to choose->filter->unsupervised->attribute->copy->ok



- copy attribute at last indices to the dataset

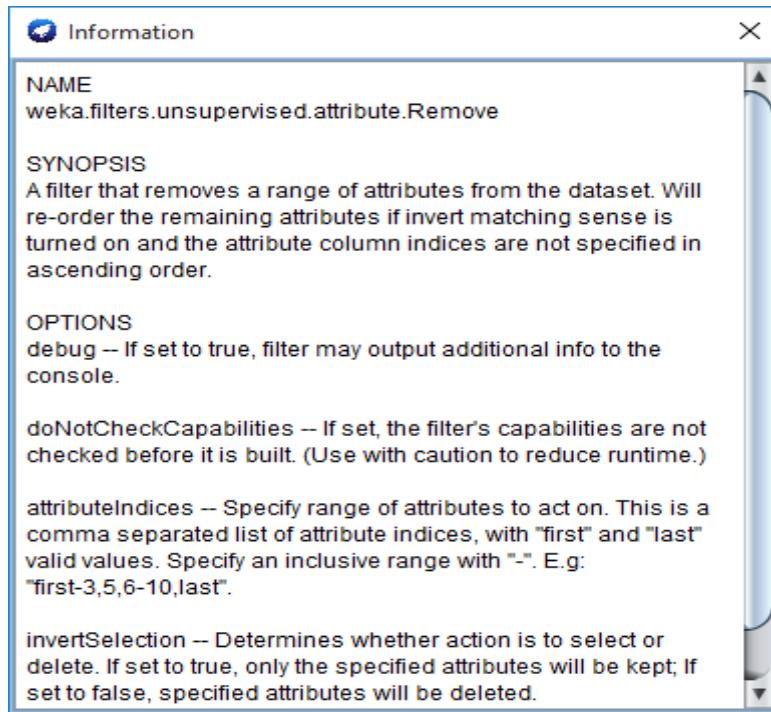


➤ Apply and edit

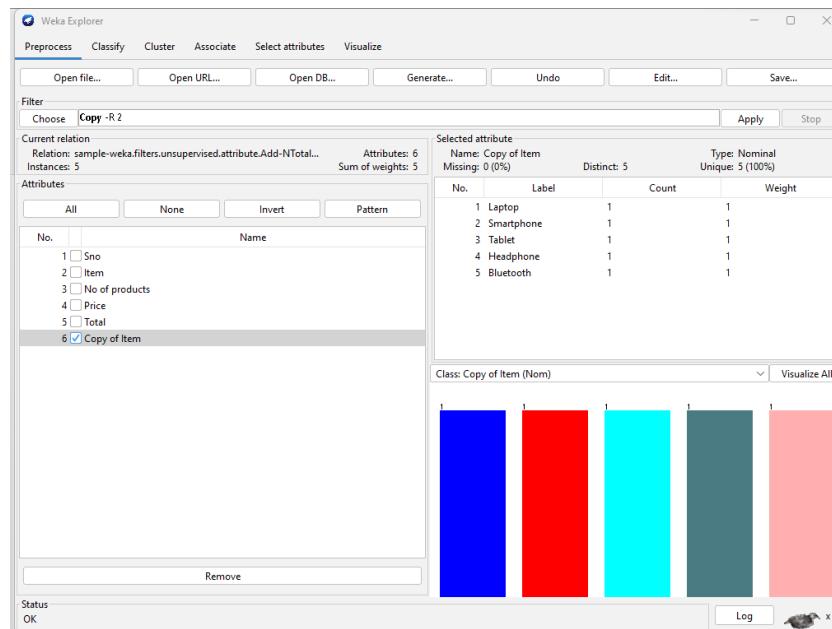
No.	1: Sno	2: Item	3: No of products	4: Price	5: Total	6: Copy of Item
1	1.0 Laptop	2.0	800.0	1600.0	Laptop	
2	2.0 Smart...	1.0	300.0	300.0	Smartphone	
3	3.0 Tablet	3.0	200.0	600.0	Tablet	
4	4.0 Head...	1.0	100.0	100.0	Headphone	
5	5.0 Bluet...	2.0	150.0	300.0	Bluetooth	

5) Remove attribute:

- To remove the attribute select/tick the attribute and select remove



- Select the attribute "Copy of Item" and remove the attribute (click on remove).



- Click apply and edit.

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Viewer

Relation: sample-weka.filters.unsupervised.attribute.Add-NTotal-Clst-W1.0-weka.filters.unsupervised.attribute.AddExpression-Ea3+a4-Nexpression-weka.filters.unsupervised.attribute.C...

No.	1: S...	2: Item	3: No of products	4: Price	5: Total
	Numeric	Nominal	Numeric	Numeric	Numeric
1	1.0	Laptop	2.0	800.0	1600.0
2	2.0	Smart...	1.0	300.0	300.0
3	3.0	Tablet	3.0	200.0	600.0
4	4.0	Head...	1.0	100.0	100.0
5	5.0	Bluet...	2.0	150.0	300.0

Add instance Undo OK Cancel

Remove 1 3

Status Problem filtering instances Log x 0

EXPERIMENT – 2

AIM:

Perform operations on any dataset using ReplaceMissingValues and ReplaceMissingValues by user content and Normalize attributes.

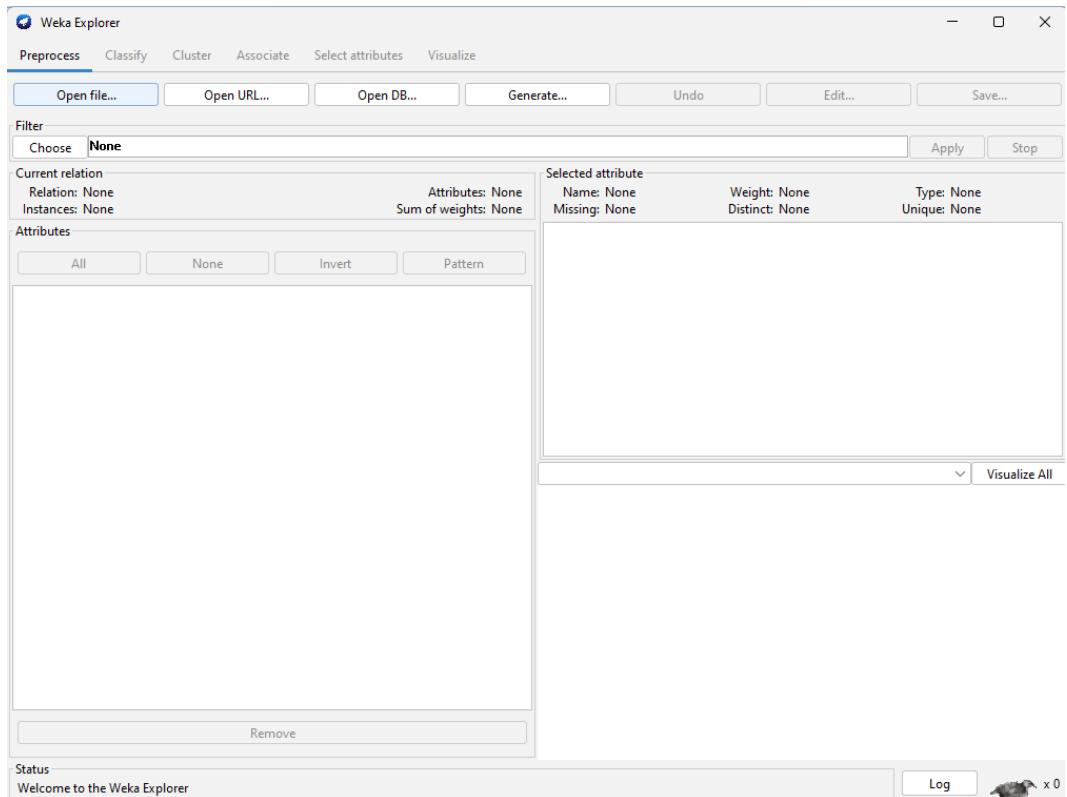
Description:

In weka, replacing missing values is a crucial preprocessing step. Utilize the “Replace Missing Values” filter to handle missing data, selecting either default strategies like mean or median imputation or opting for user specified content replacement. This ensures a complete dataset for more accurate and reliable machine learning model training.

Procedure:

- ❖ Step-1: open weka tool and select the explorer



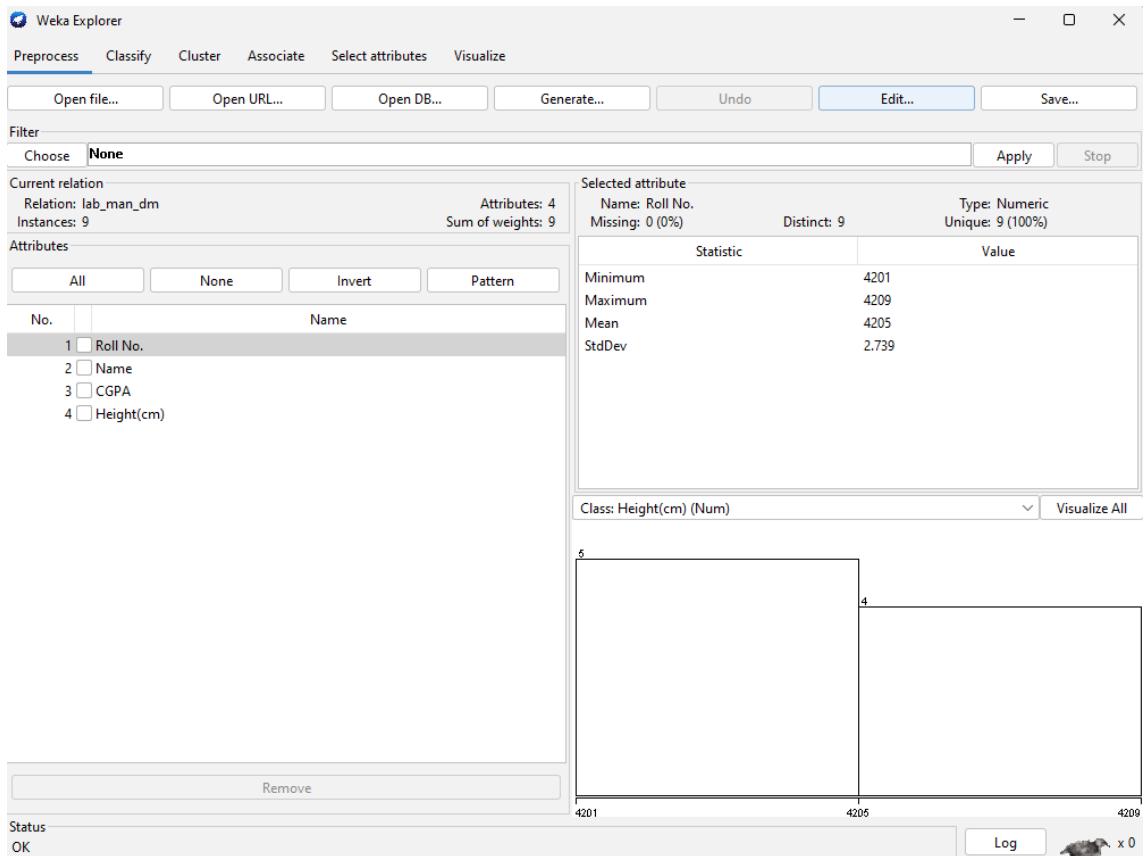


❖ Step-2: load the dataset by selecting the “open file” option.

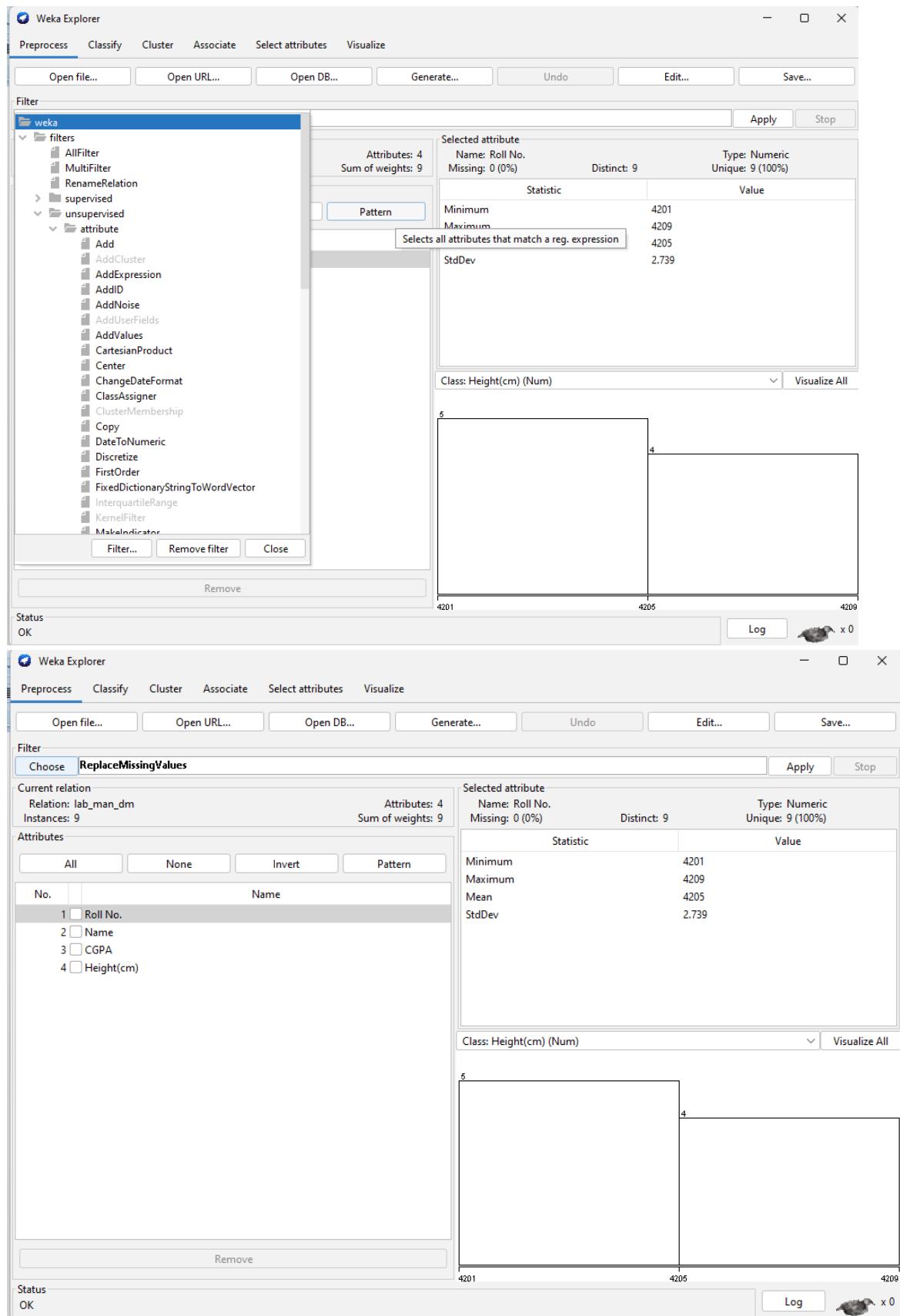
Select the “.csv(comma delimited)” saved file.

1	Roll No.	Name	CGPA	Height(cm)
2	4201	kiran kumar	9	175
3	4202	ayush	8.2	180
4	4203	manoj	8.5	165
5	4204	naveen	8.28	150
6	4205	vyshu	9.35	165
7	4206	sai	7.5	180
8	4207	kiran kumar	8	192
9	4208			
10	4209	nikhil	8.6	174

❖ Upon successful opening.



- ❖ Step-3: since the dataset is intentionally filled with missing values, we will fill it up using filters.
- ❖ ReplaceMissingValues:
 - ➔ To utilize this option, follow accordingly
 - ➔ Select choose filter > filters > unsupervised > attribute > ReplaceMissingValues



→ Next, select “apply” and click on “edit” to see the result table.

Viewer

Relation: lab_man_dm-weka.filters.unsupervised.attribute.ReplaceMissingValues

No.	1: Roll No.	2: Name	3: CGPA	4: Height(cm)
	Numeric	Nominal	Numeric	Numeric
1	4201.0	kiran k...	9.0	175.0
2	4202.0	ayush	8.2	180.0
3	4203.0	manoj	8.5	165.0
4	4204.0	naveen	8.28	150.0
5	4205.0	vyshu	9.35	165.0
6	4206.0	sai	7.5	180.0
7	4207.0	kiran k...	8.0	192.0
8	4208.0	kiran k...	8.4287499	
9	4209.0	nikhil	8.6	174.0



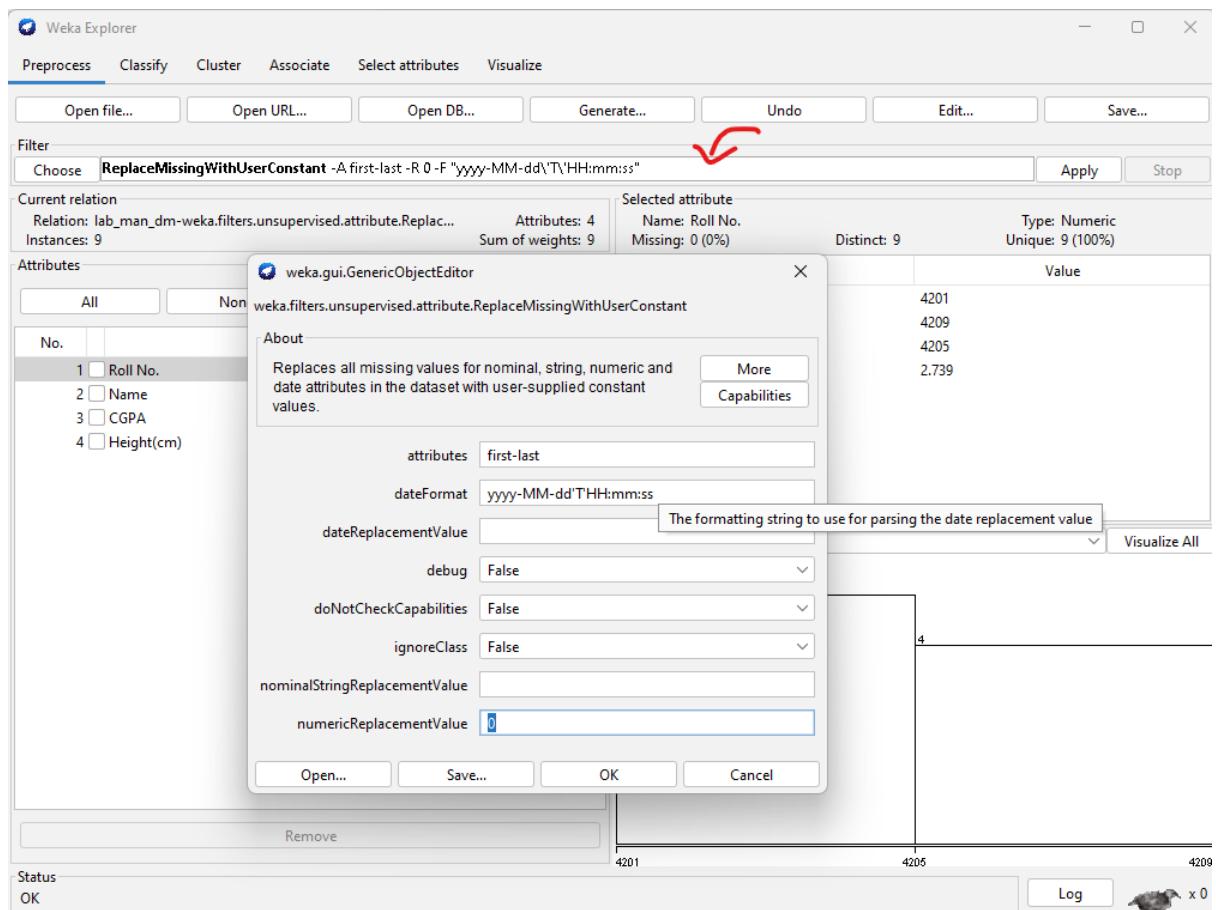
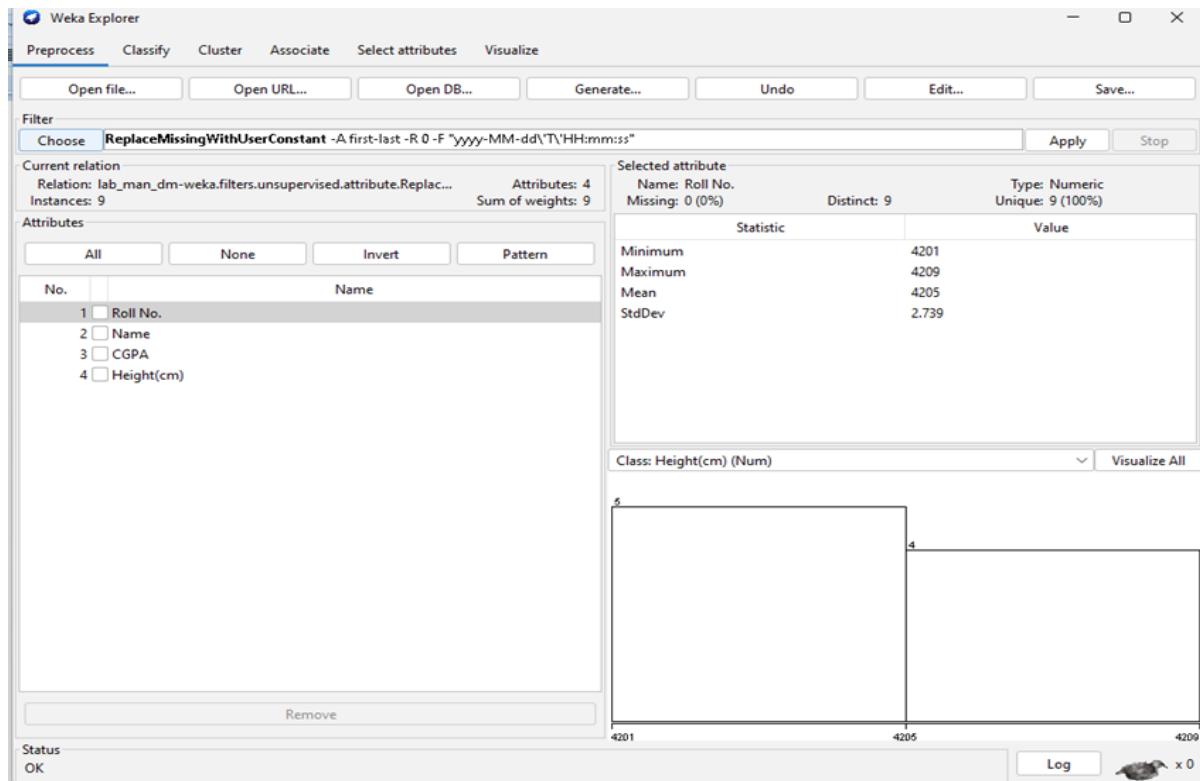
Add instance Undo OK Cancel

→ Here, we can see that the 8th instance is filled with some values.

→ NOTE:

- I. While replacing the Categorical data the data is replaced with Mode of the data.
- II. While replacing the Numerical data the data is replaced with the Mean of the data.
- III. Also, the last column is not updated because it is the decision column and no modifications are done in it.

❖ ReplaceMissingWithUserConstant:



→ click below the arrow mark and modify the “numericReplacementValue” accordingly.

Viewer

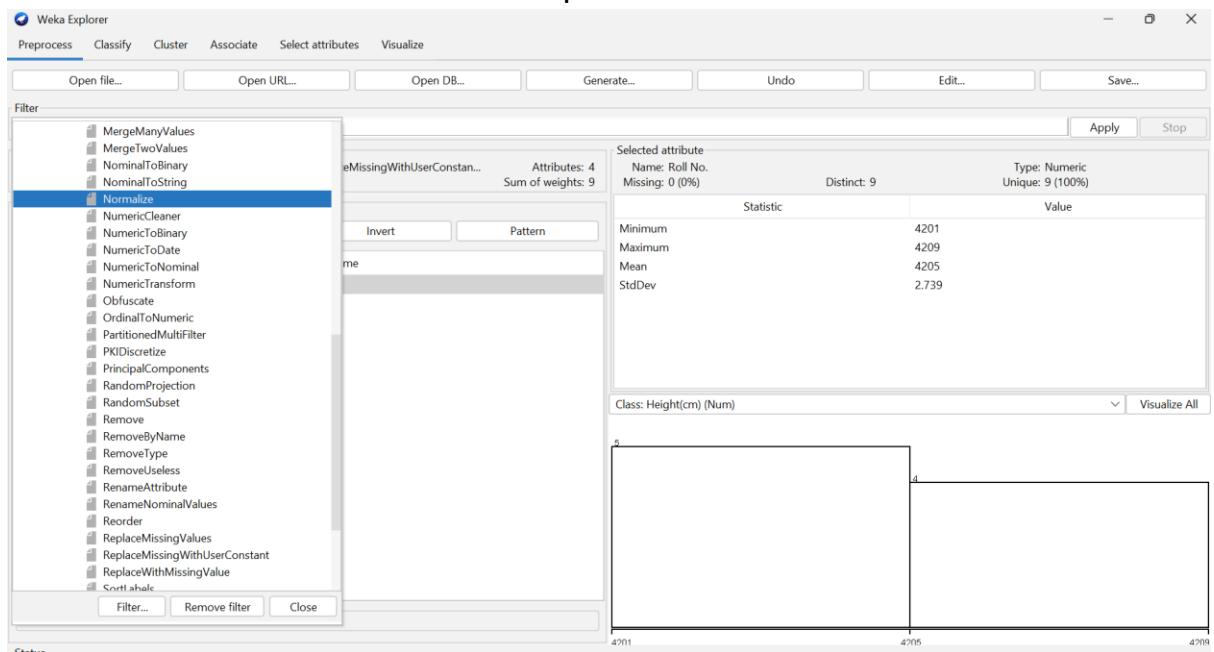
Relation: lab_man_dm-weka.filters.unsupervised.attribute.ReplaceMissingWithUserConstant

No.	1: Roll No.	2: Name	3: CGPA	4: Height(cm)
	Numeric	Nominal	Numeric	Numeric
1	4201.0	kiran k...	9.0	175.0
2	4202.0	ayush	8.2	180.0
3	4203.0	manoj	8.5	165.0
4	4204.0	naveen	8.28	150.0
5	4205.0	vyshu	9.35	165.0
6	4206.0	sai	7.5	180.0
7	4207.0	kiran k...	8.0	192.0
8	4208.0	0.999	0.999	174.0
9	4209.0	nikhil	8.6	174.0

❖ Normalization:

→ To utilize this option, follow accordingly

→ Select choose filter > filters > unsupervised > attribute >normalize



→ After applying the filter, results are shown below

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Open file... Open URL... Open DB... Generate... Undo Edit... Save...

Filter Choose Normalize -S 1.0 -T 0.0 Apply Stop

Current relation Relation: lab_man_dm-weka.filters.unsupervised.attribute.ReplaceMissingWithUserConstant-Afirst-last-R0.999-Fyyyy-MM-dd'T'HH:mm:ss-weka.filters.unsupervised...

Instances: 9

Attributes

All None

No.

1 Roll No.
2 Name
3 CGPA
4 Height(cm)

Viewer

No.	1: Roll No.	2: Name	3: CGPA	4: Height(cm)
1	0.0	kiran k...	0.9580...	175.0
2	0.125	ayush	0.8622...	180.0
3	0.25	manoj	0.8982...	165.0
4	0.375	naveen	0.8718...	150.0
5	0.5	vyshu	1.0	165.0
6	0.625	sai	0.7784...	180.0
7	0.75	kiran k...	0.8383...	192.0
8	0.875	0.999	0.0	
9	1.0	nikhil	0.9101...	174.0

Add instance Undo OK Cancel

The screenshot shows the Weka Explorer interface with a 'Viewer' window open. The 'Viewer' window displays a table of 9 instances with 4 attributes: Roll No., Name, CGPA, and Height(cm). The Height(cm) column contains values ranging from 0.0 to 175.0. A filter 'Normalize -S 1.0 -T 0.0' is applied, which typically scales the data to a range between 0 and 1. The 'Attributes' panel on the left lists the four attributes, and the 'Instances' panel shows the 9 rows of data. The 'Preprocess' tab is selected in the top menu bar.

→ All attributes are updated to be ranged between the interval [0,1]

EXPERIMENT - 3

AIM:

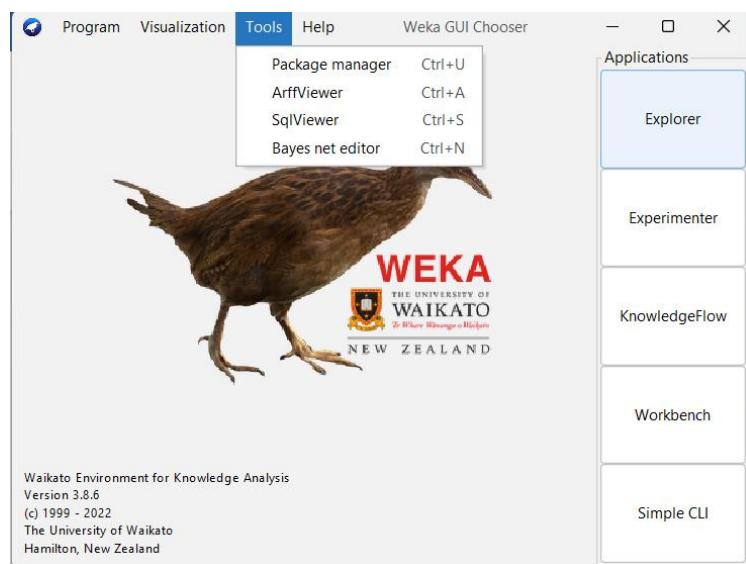
Build a Classification Model using ID3 Decision Tree Algorithm on any dataset.

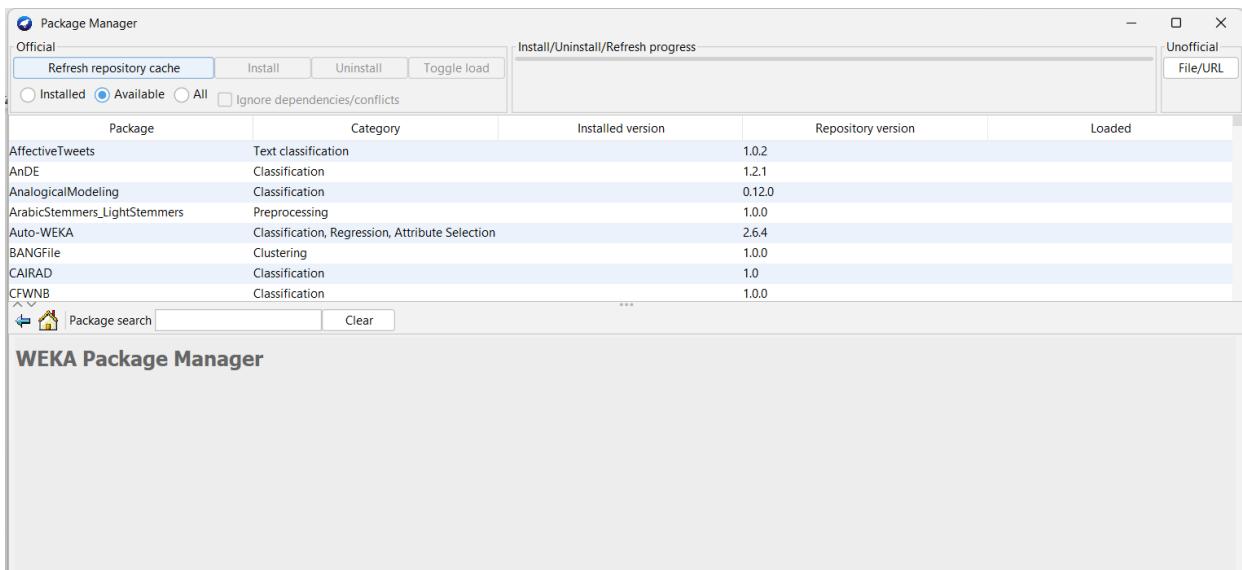
Description:

A decision tree is a flowchart-like tree structure, where each internal node (non-leaf node) denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node (or *terminal node*) holds a class label. ID3 stands for **Iterative Dichotomiser 3** and is named such because the algorithm iteratively (repeatedly) dichotomizes (divides) features into two or more groups at each step.

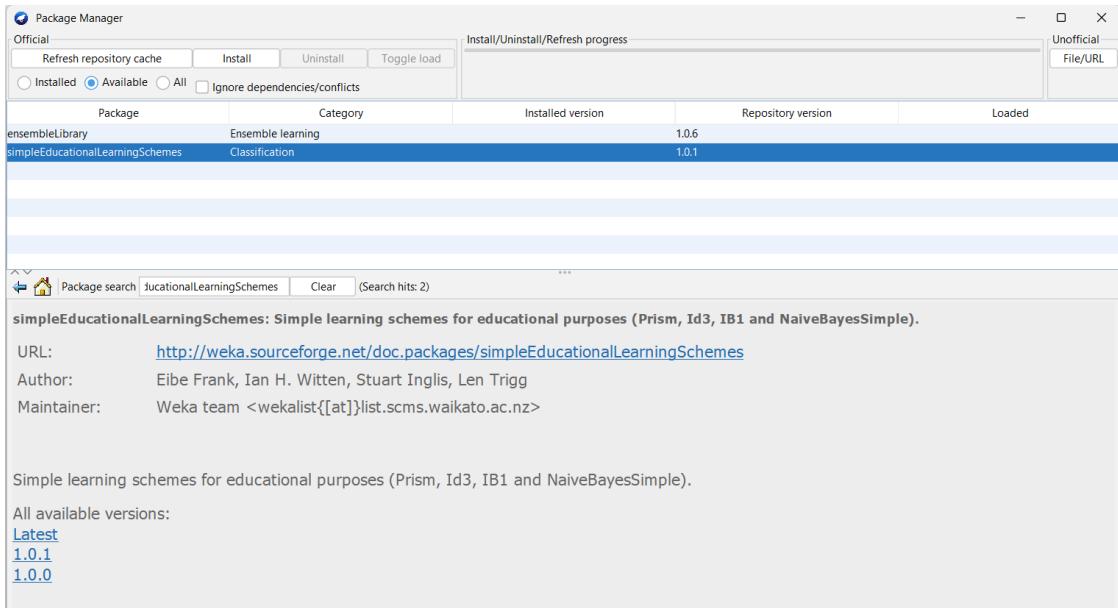
→Installing a Package

1. Select Tools > Package manager from WEKA GUI Chooser.

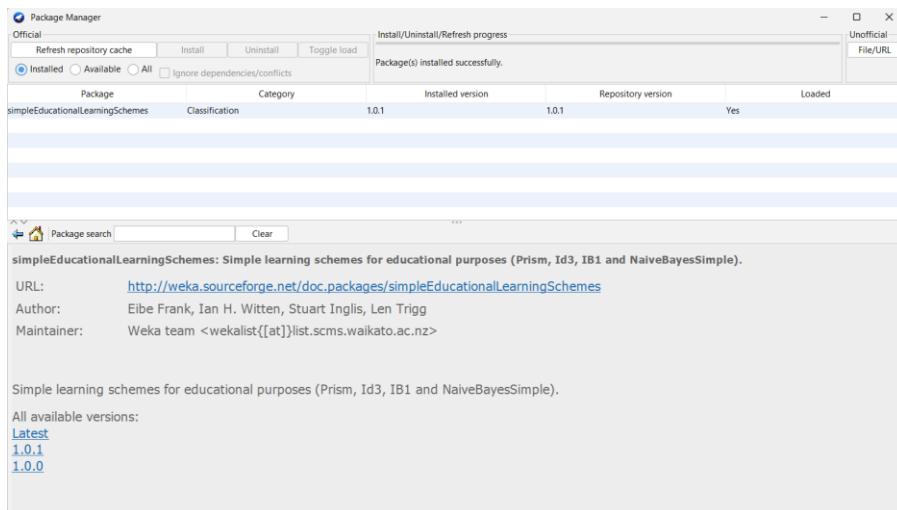




2. To download the ID3 Decision Tree package, you'll need to search for the package name "**simpleEducationalLearningSchemes**" in the Package Search. Make sure to keep the window in the "Available" mode while searching.
3. Please click the "install" button to proceed with package installation.



4. The installation process has been completed. Please check the "Installed" window to verify the package.



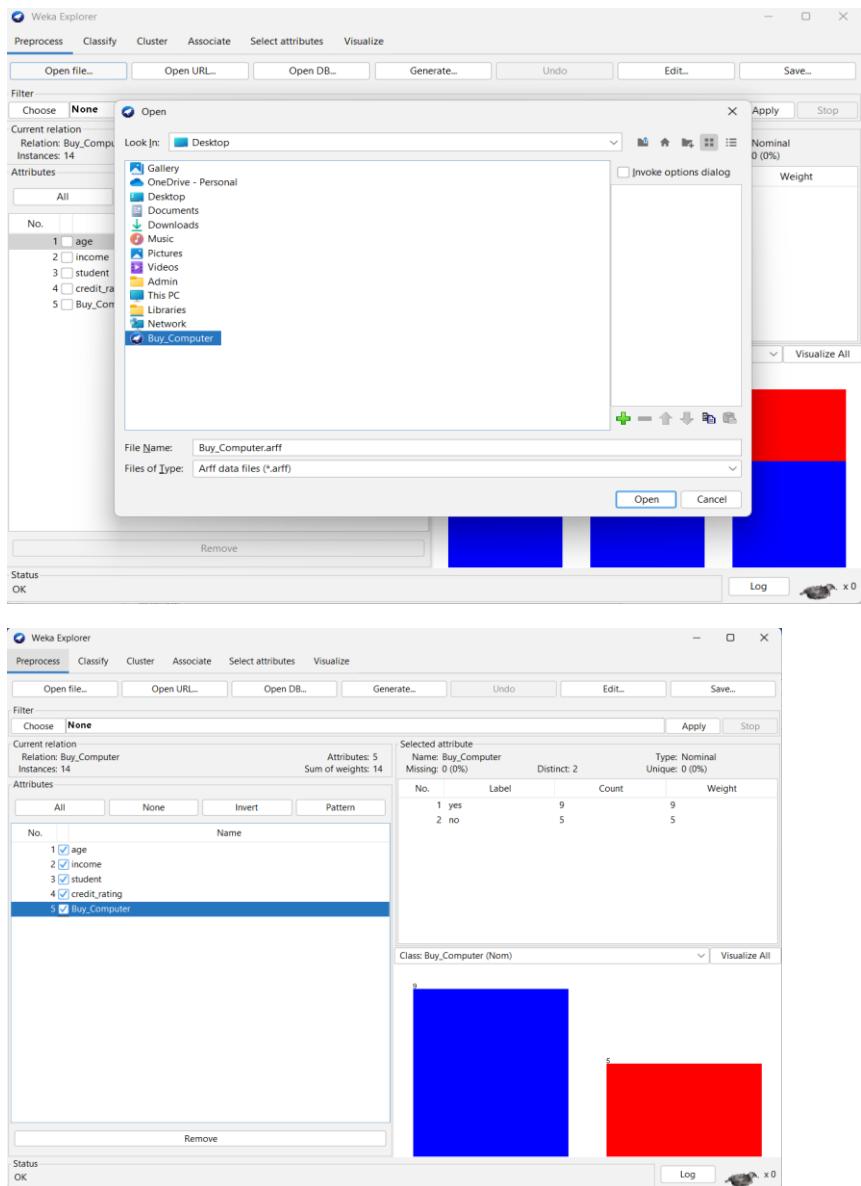
→Classification Model using ID3 Decision Tree Algorithm.

1. Generate a dataset containing attributes such as Age, Income, Student, Credit_Rating, and Buy_Computer, formatted in Notepad with .arff file format.

```
@relation Buy_Computer
@attribute age {youth, middle_age, senior}
@attribute income {low, medium, high}
@attribute student {yes, no}
@attribute credit_rating {excellent, fair}
@attribute Buy_Computer {yes, no}

@data
youth,high,no,fair,no
youth,high,no,excellent,no
middle_age,high,no,fair,yes
senior,medium,no,fair,yes
senior,low,yes,excellent,yes
senior,low,yes,excellent,no
middle_age,low,yes,fair,yes
youth,medium,no,fair,no
youth,low,yes,fair,yes
senior,medium,yes,fair,yes
youth,medium,yes,excellent,yes
middle_age,medium,no,excellent,yes
middle_age,high,yes,fair,yes
senior,medium,no,excellent,no
```

2. Select the dataset that we created earlier.

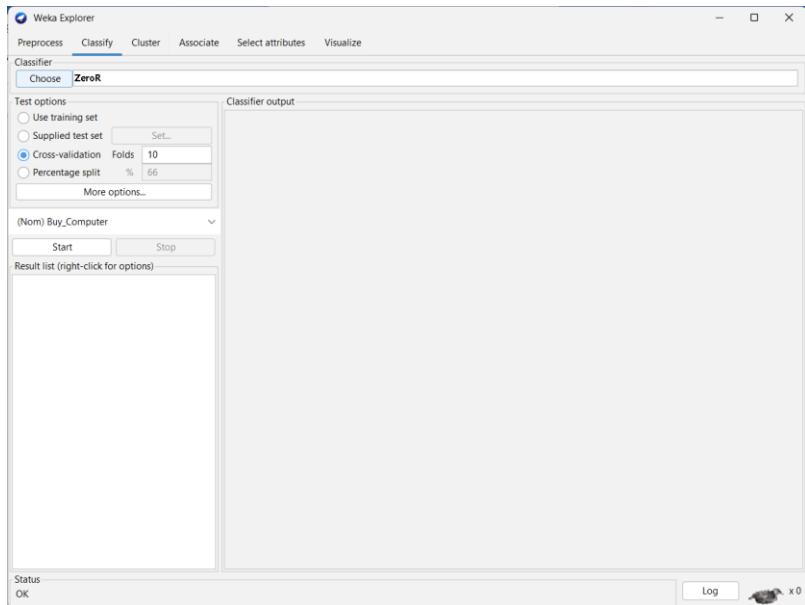


3. Please click "Edit" to view the Dataset in tabular format.

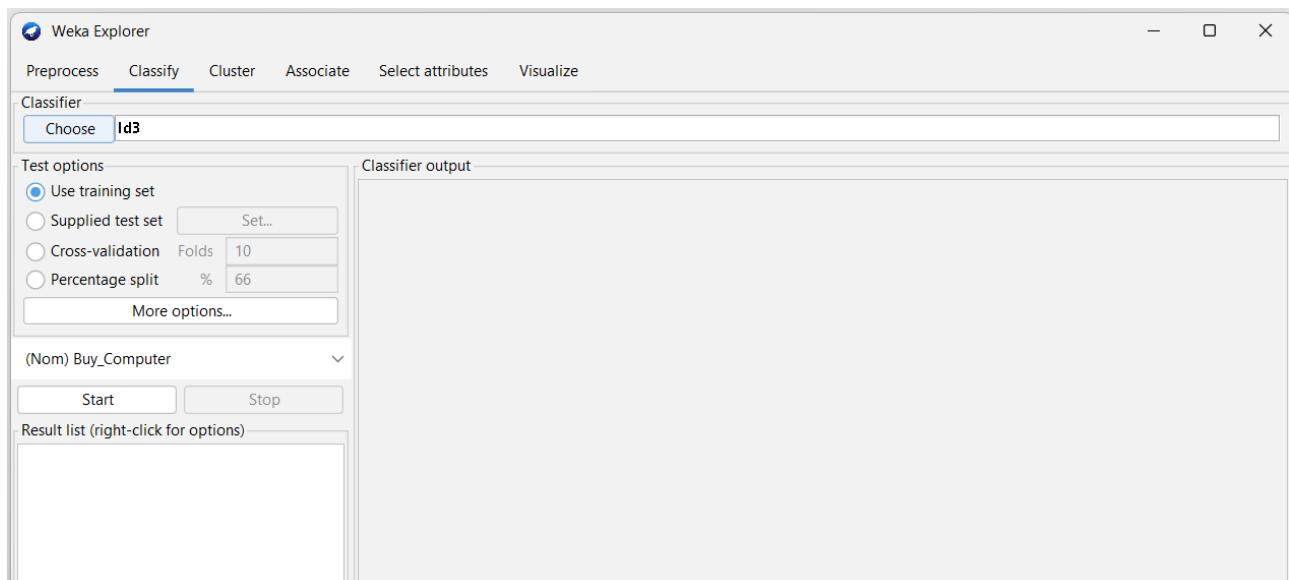
This screenshot shows the 'Viewer' interface displaying the 'Buy_Computer' dataset in tabular form. The table has 14 rows and 5 columns, corresponding to the 14 instances and 5 attributes.

No.	1: age	2: income	3: student	4: credit_rating	5: Buy_Computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	excellent	yes
6	senior	low	yes	excellent	no
7	middle	low	yes	fair	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle	medium	no	excellent	yes
13	middle	high	yes	fair	yes
14	senior	medium	no	excellent	no

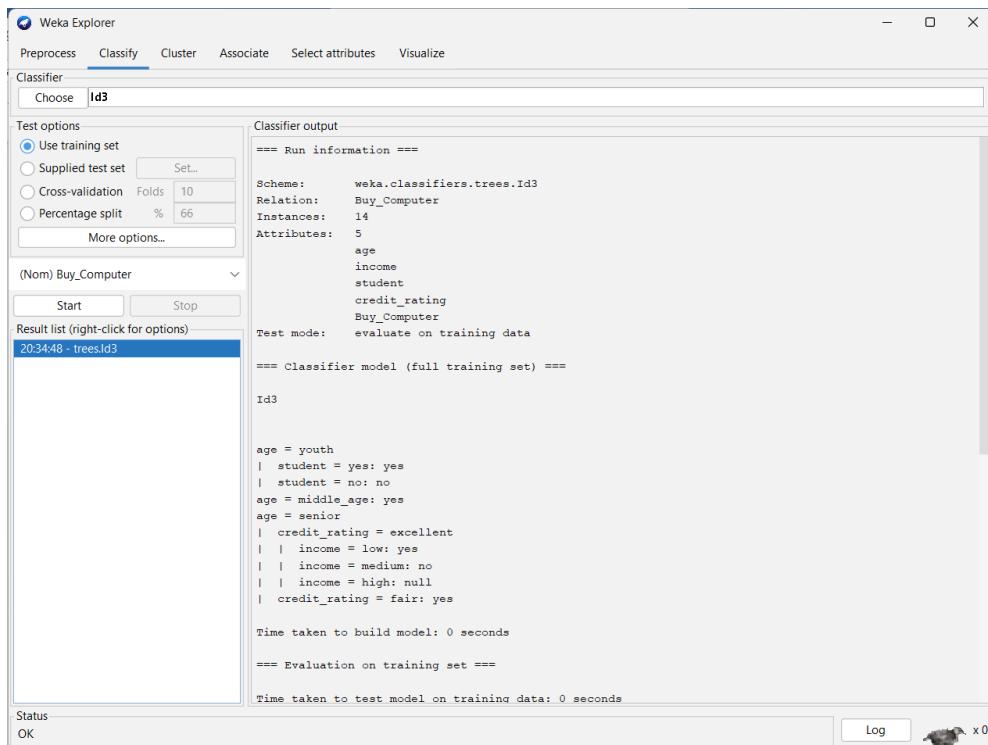
4. Please open the Classify window to utilize the **ID3 Decision Tree algorithm** for classifying the dataset.



5. Click on "Choose" and select "trees > ID3" to choose the package for the ID3 Decision Tree algorithm.



6. Initiate the classification process by clicking on the "Start" button.



7. The Classifier Output window will display the following information:

- Run information
- Classifier model
- Stratified cross-validation results
- Detailed Accuracy By Class
- Confusion Matrix

Classifier output

```

| | income = high: null
| credit_rating = fair: yes

Time taken to build model: 0 seconds

==== Evaluation on training set ====

Time taken to test model on training data: 0 seconds

==== Summary ====
Correctly Classified Instances      13          92.8571 %
Incorrectly Classified Instances   1           7.1429 %
Kappa statistic                   0.8372
Mean absolute error               0.0714
Root mean squared error          0.189
Relative absolute error          15.3846 %
Root relative squared error     39.4132 %
Total Number of Instances        14

==== Detailed Accuracy By Class ====

```

TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
1.000	0.200	0.900	1.000	0.947	0.849	0.989	0.989	yes
0.800	0.000	1.000	0.800	0.889	0.849	0.989	0.967	no
Weighted Avg.	0.929	0.129	0.936	0.929	0.926	0.989	0.981	

```

==== Confusion Matrix ====
a b    <-- classified as
9 0 | a = yes
1 4 | b = no

```

Experiment – 4

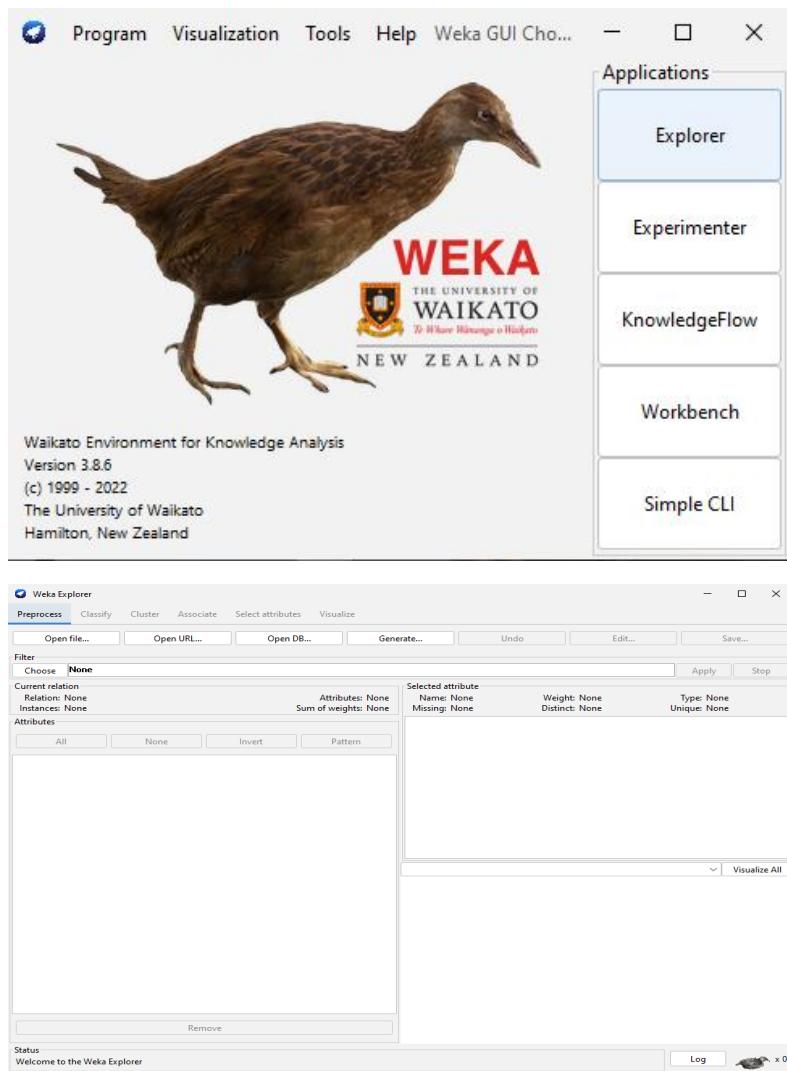
Aim:

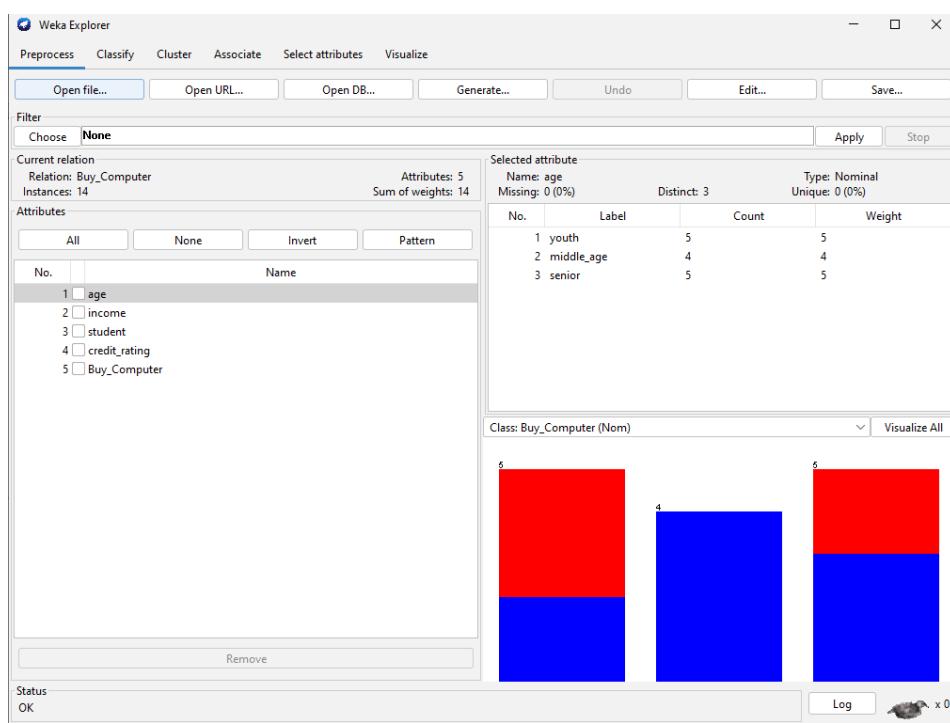
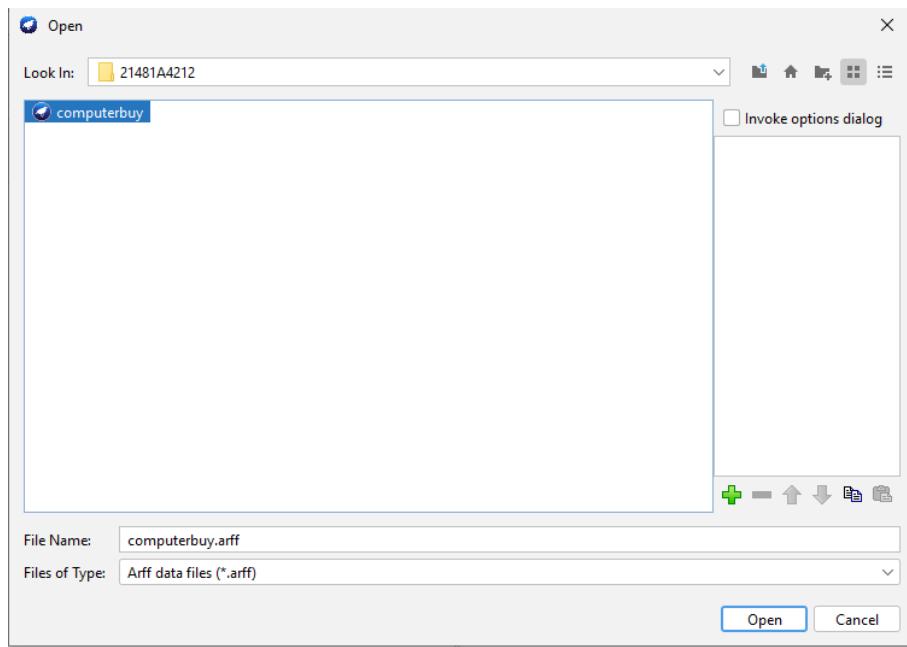
Build a Classification Model using J48 Decision Tree Algorithm on any dataset.

Description: J48 is a machine learning decision tree classification algorithm based on Iterative Dichotomiser 3. It is very helpful in examine the data categorically and continuously.

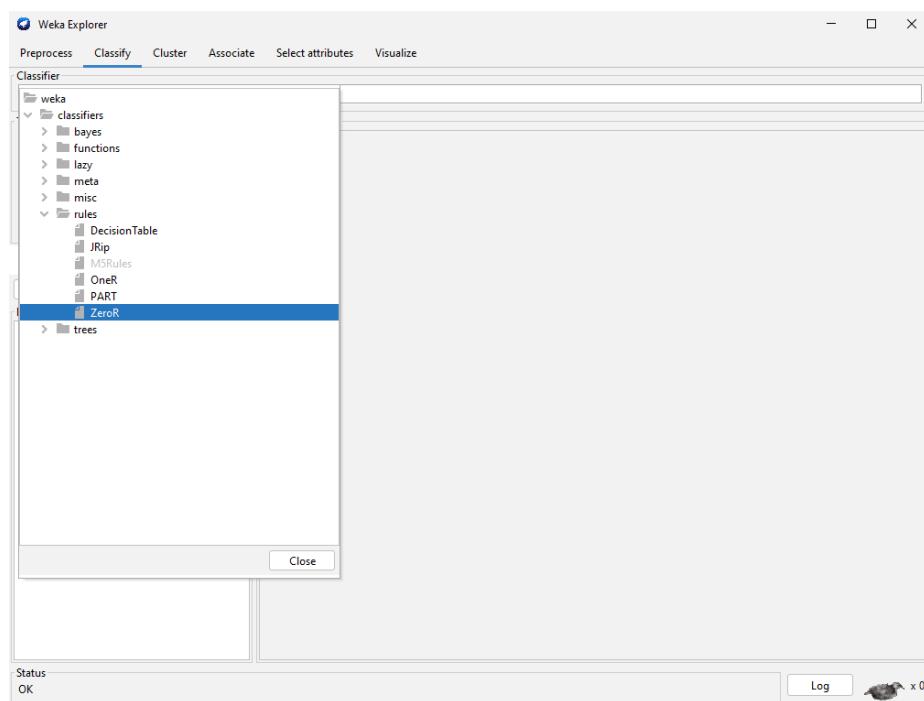
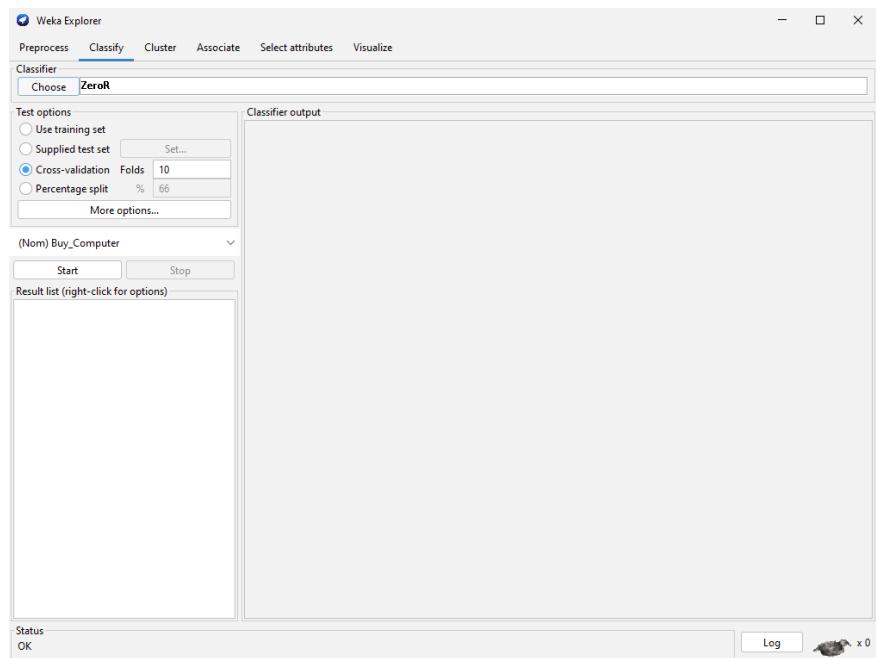
Process:

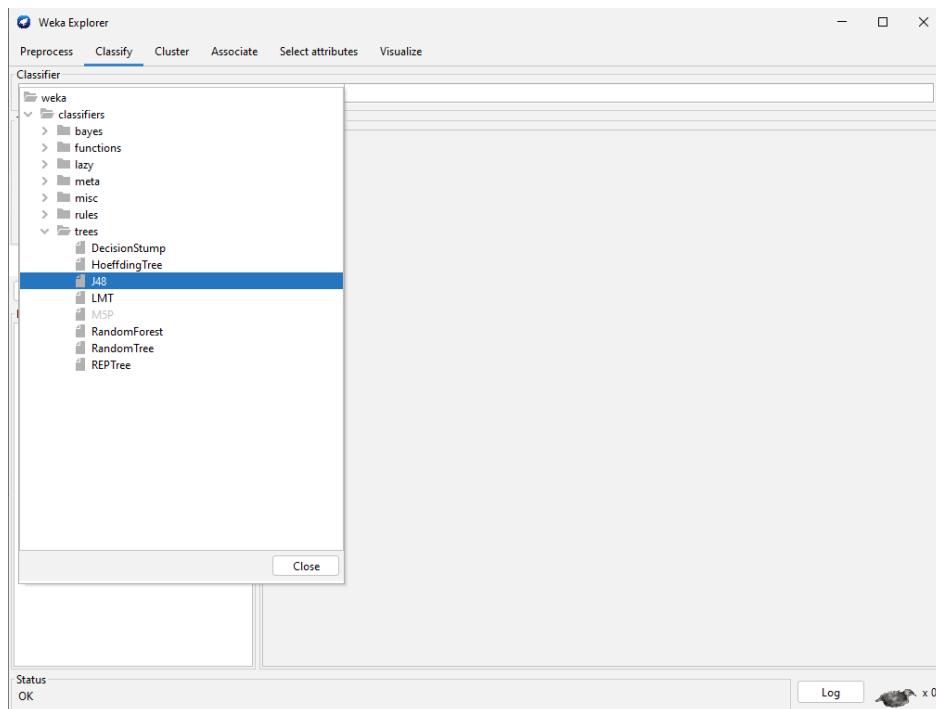
1)open weka tool > explorer and load the .arff file



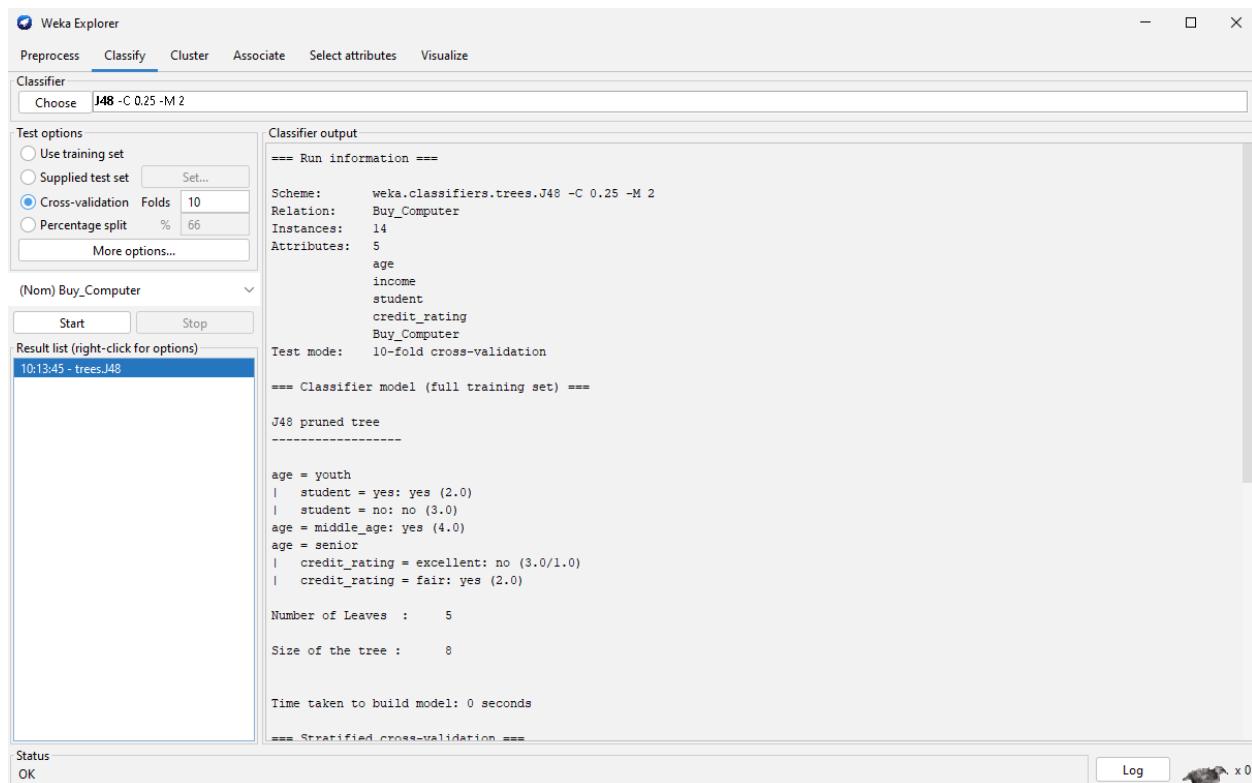


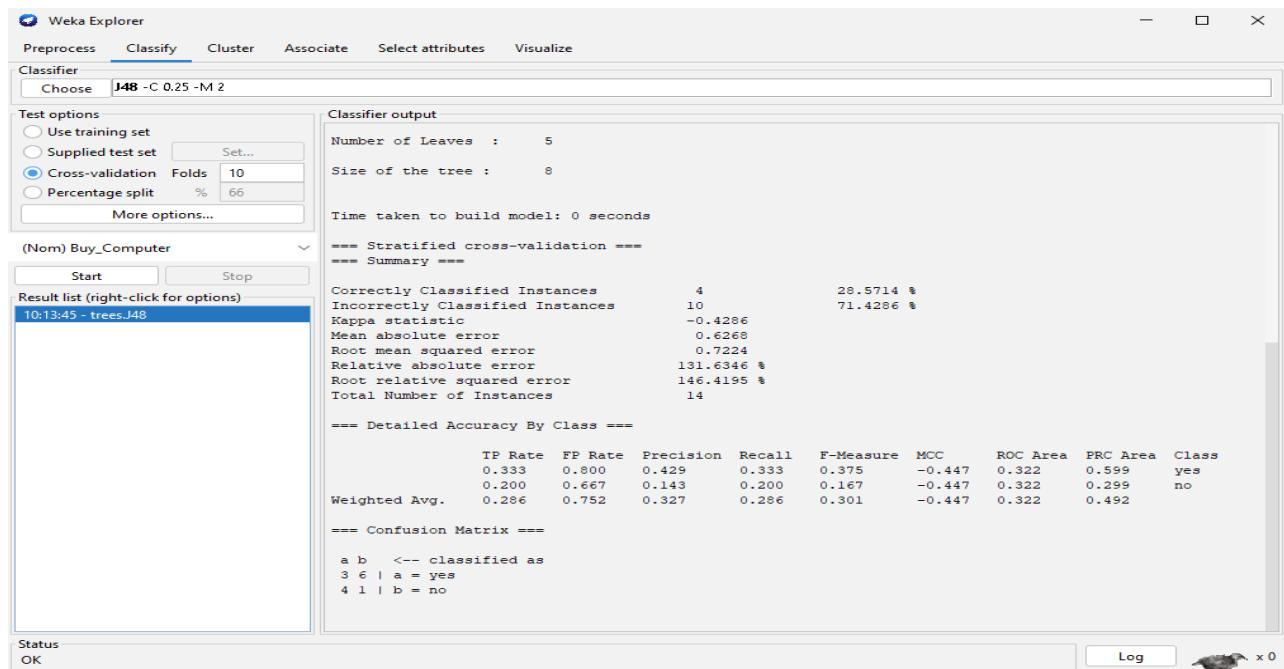
2)click on classify > choose> trees > and select J48.





3)click start and the output is observed in screen.

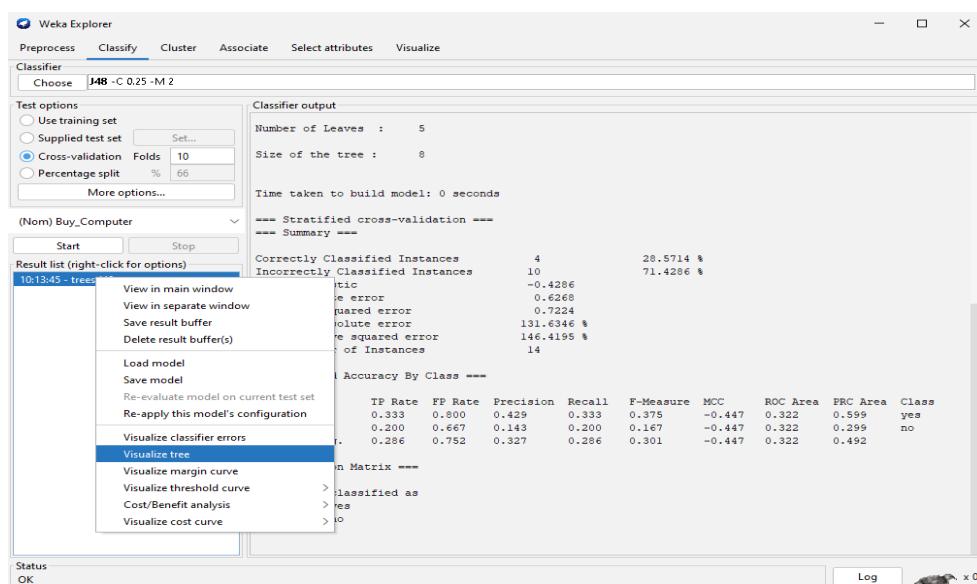


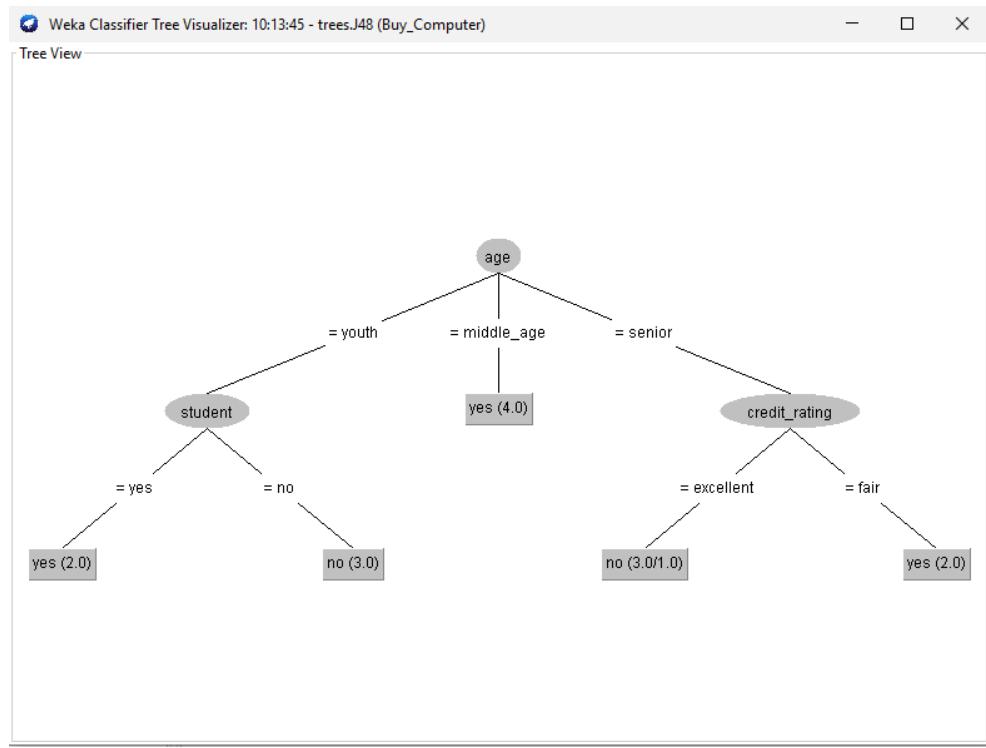


4) The classifier output window will display the following information

- I. Run Information
- II. Classifier model
- III. Stratified cross –validation results
- IV. Detailed Accuracy by class
- V. Confusion matrix

5)To get the decision tree move cursor to the highlighted line and right click it. Now select visualise tree form the given options.





Experiment -5

Aim:

Build a Classification Model using Naive Bayes Algorithm on any dataset

Description:

A naive Bayes classifier estimates the class-conditional probability by assuming that the attributes are conditionally independent, given the class label y.

The conditional independence assumption can be formally stated as follows:

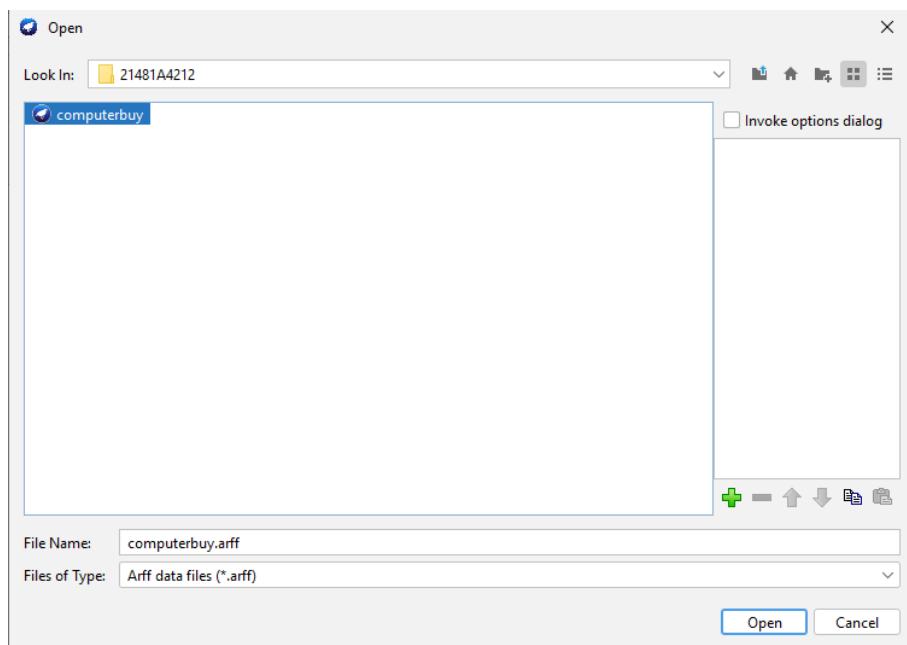
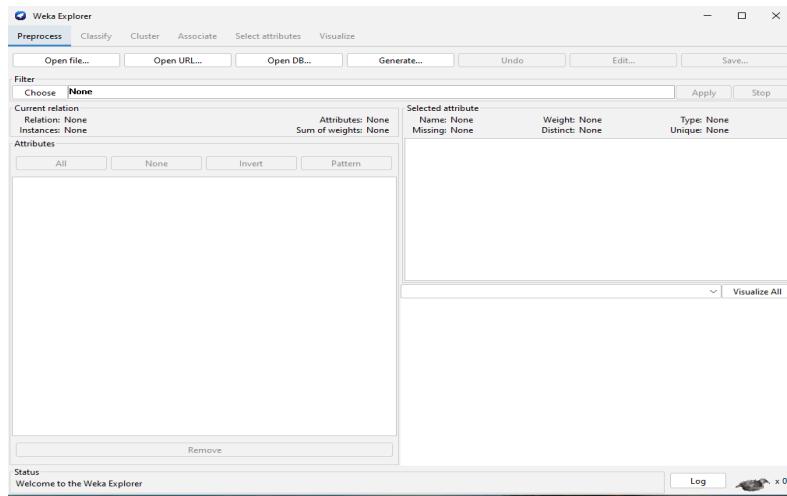
$$P(X/Y=y) = \prod_{i=1}^d p(\frac{X_i}{Y} = y)$$

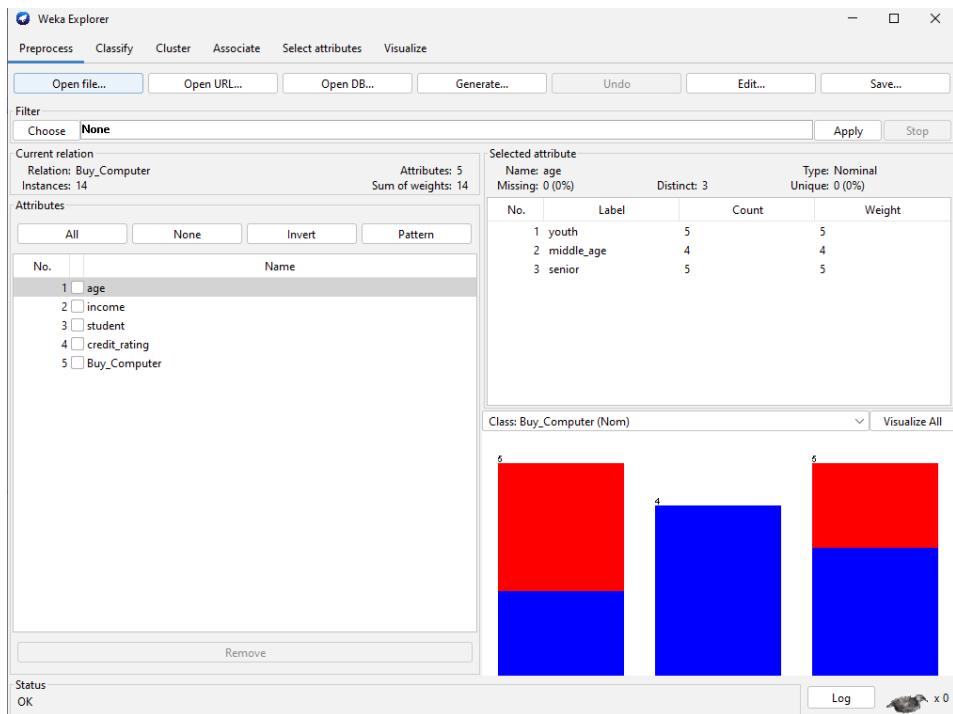
Where each attribute set $X = \{X_1, X_2, \dots, X_d\}$ consists of d attributes.

Process:

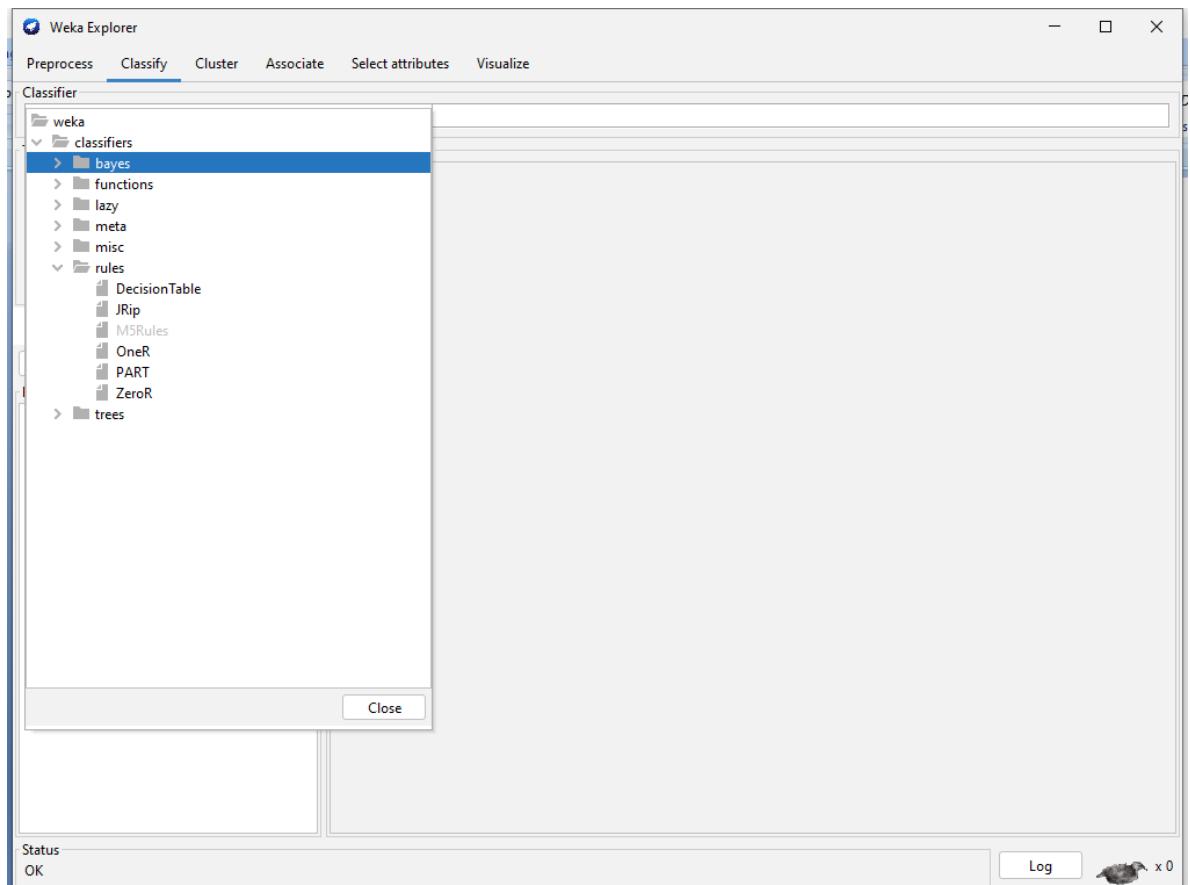
1. open weka tool > explorer and load the .arff file

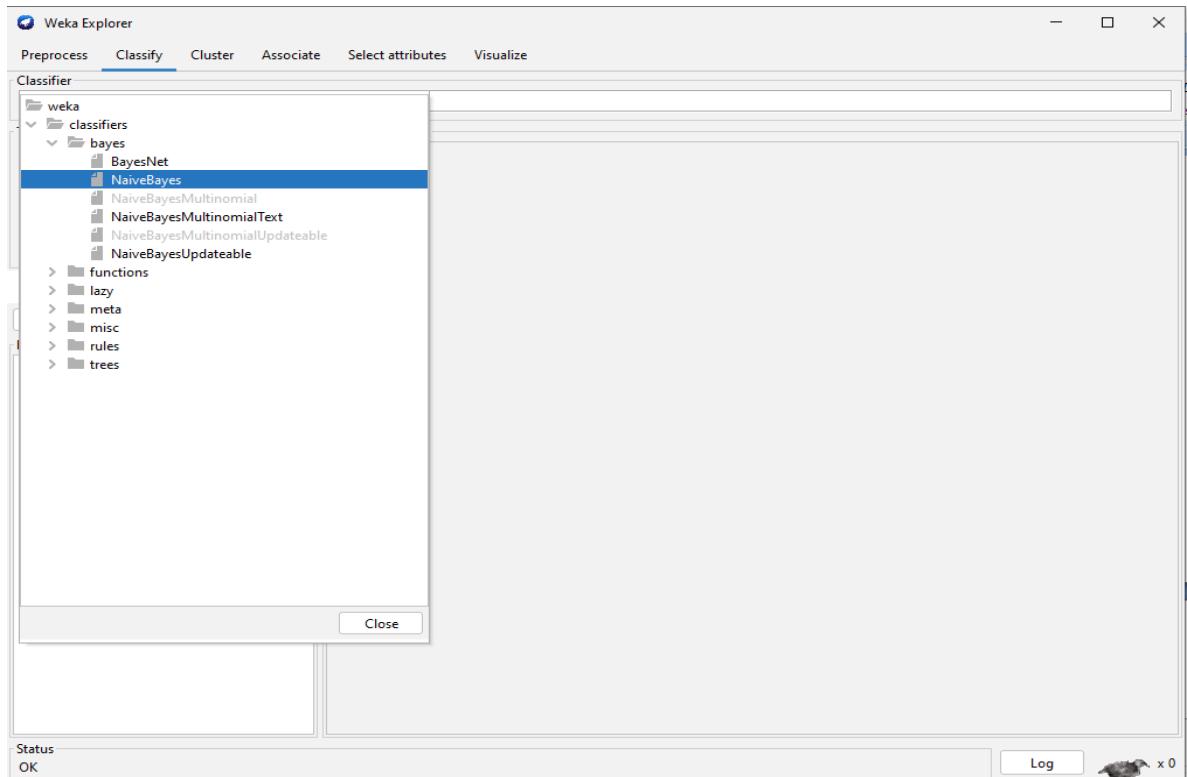




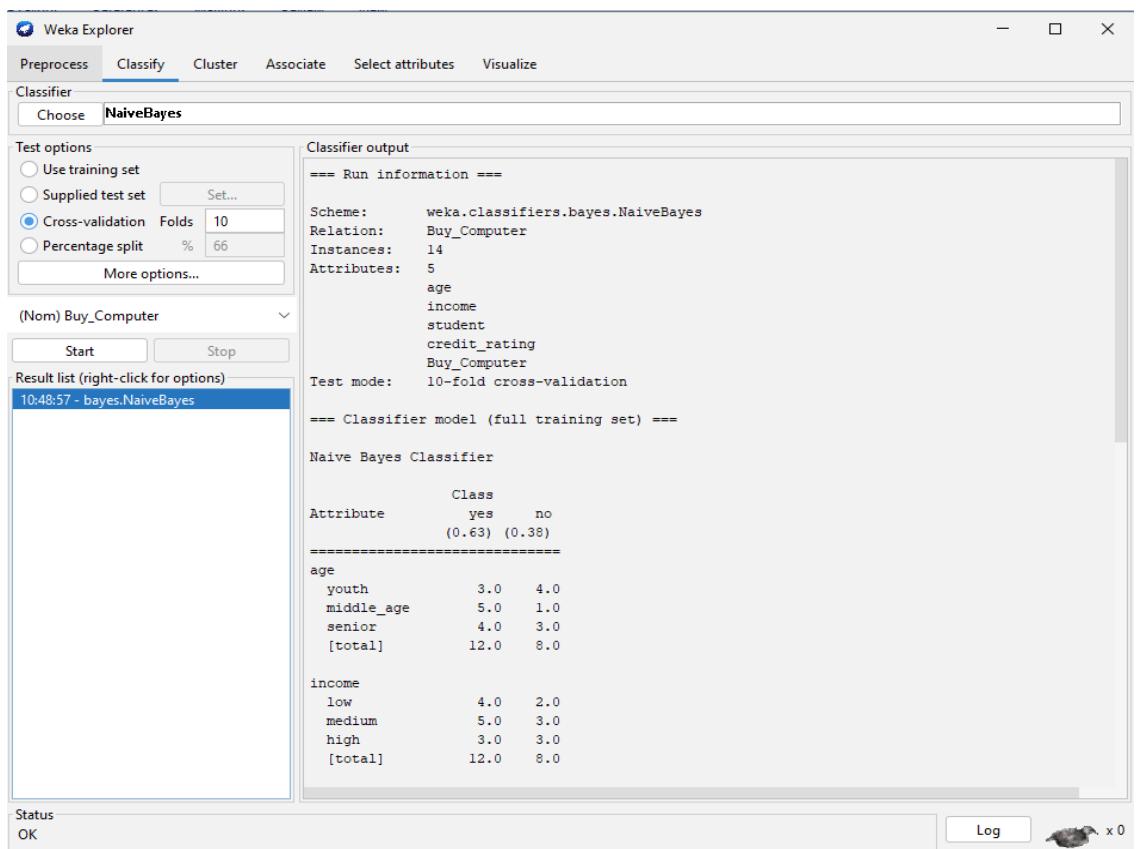


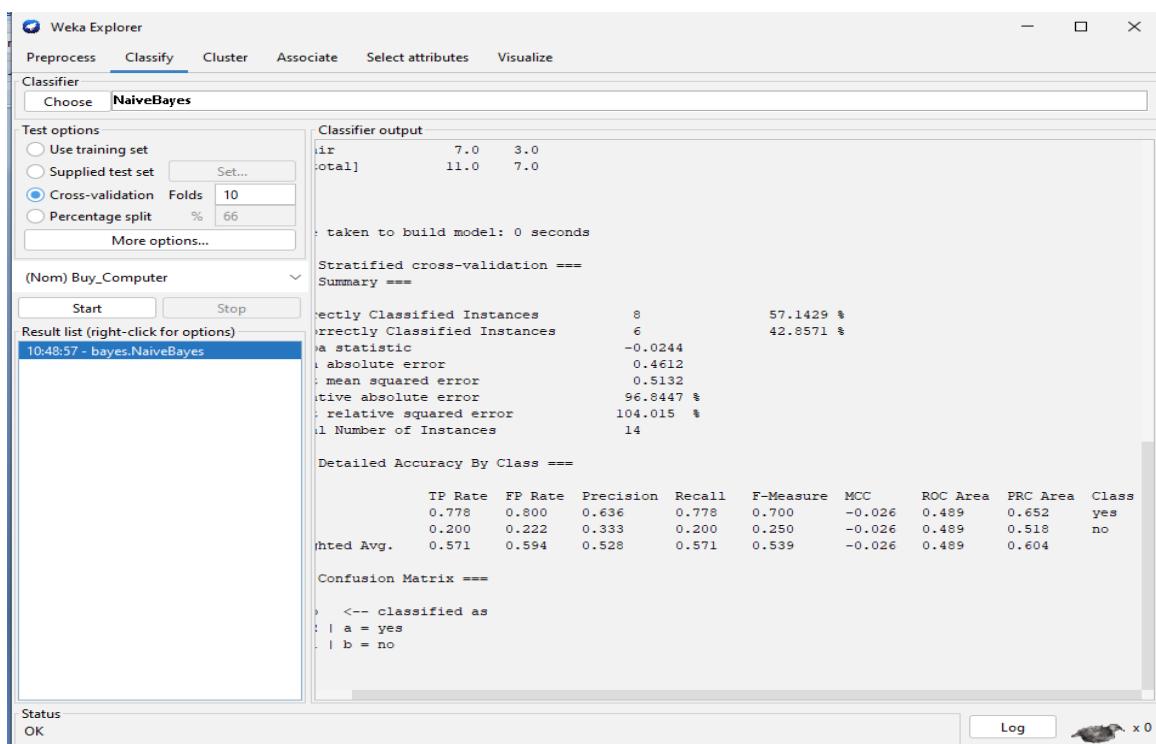
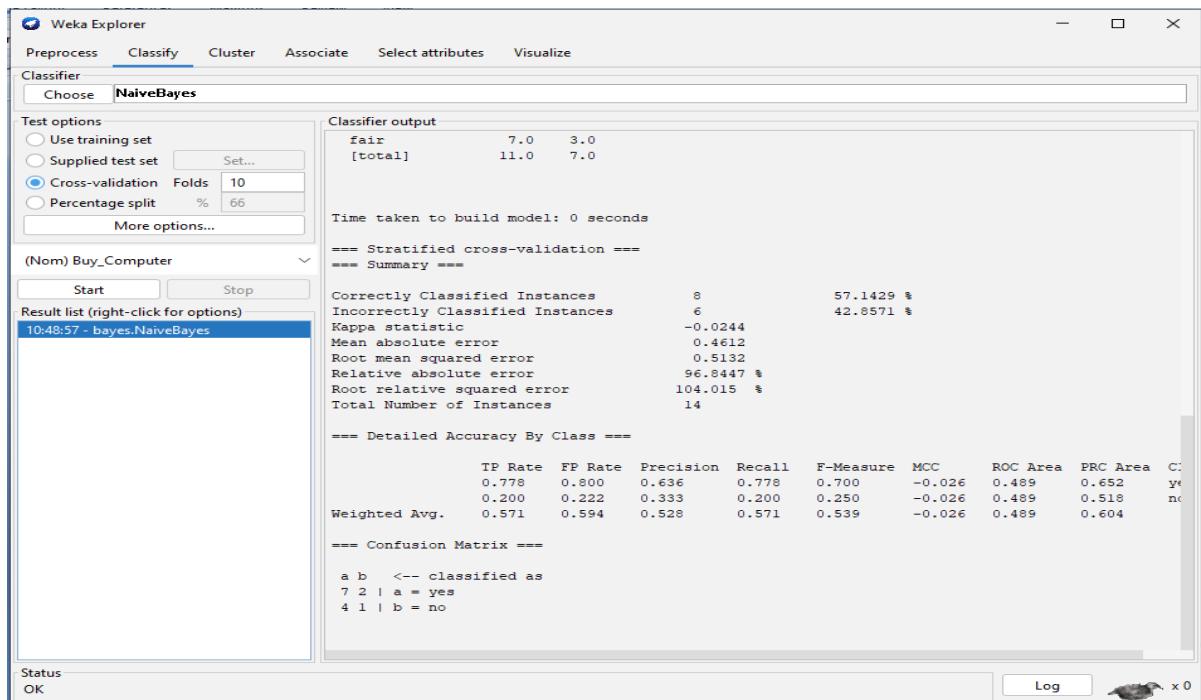
2. Click on classify > choose>bayes> and select naviebayes





3. Click start and the output is observed in screen.





4. The classifier output window will display the following information

- VI. Run Information
- VII. Classifier model
- VIII. Stratified cross –validation results
- IX. Detailed Accuracy by class
- X. Confusion matrix

EXPERIMENT - 6

AIM:

Apply K-Cross Validation on any dataset.

DESCRIPTION:

To apply K-Cross Validation as previously done in Experiment 3, we need to download the ID3 package.

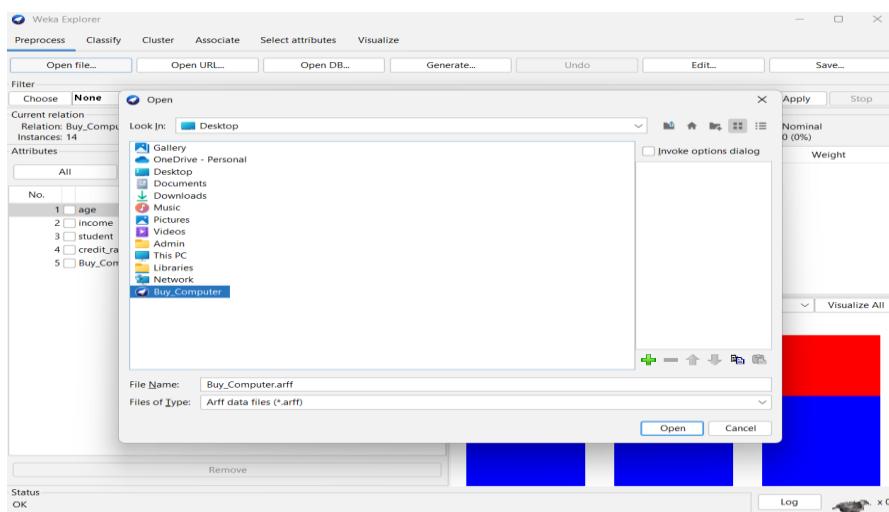
→Apply K-Cross Validation

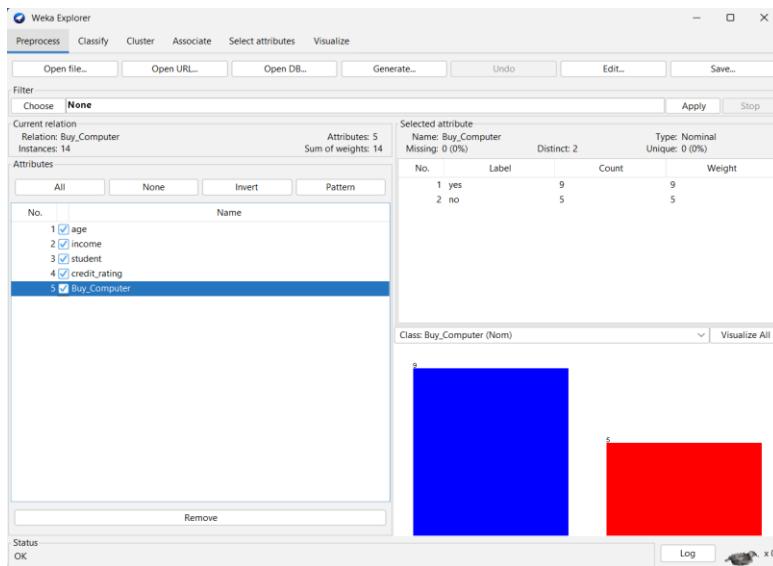
1. Generate a dataset containing attributes such as Age, Income, Student, Credit_Rating, and Buy_Computer, formatted in Notepad with .arff file format.

```
@relation Buy_Computer
@attribute age {youth, middle_age, senior}
@attribute income {low, medium, high}
@attribute student {yes, no}
@attribute credit_rating {excellent, fair}
@attribute Buy_Computer {yes, no}

@data
youth,high,no,fair,no
youth,high,no,excellent,no
middle_age,high,no,fair,yes
senior,medium,no,fair,yes
senior,low,yes,excellent,yes
senior,low,yes,excellent,no
middle_age,low,yes,fair,yes
youth,medium,no,fair,no
youth,low,yes,fair,yes
senior,medium,yes,fair,yes
youth,medium,yes,excellent,yes
middle_age,medium,no,excellent,yes
middle_age,high,yes,fair,yes
senior,medium,no,excellent,no
```

2. Select the dataset that we created earlier.





3. Please click "Edit" to view the Dataset in tabular format.

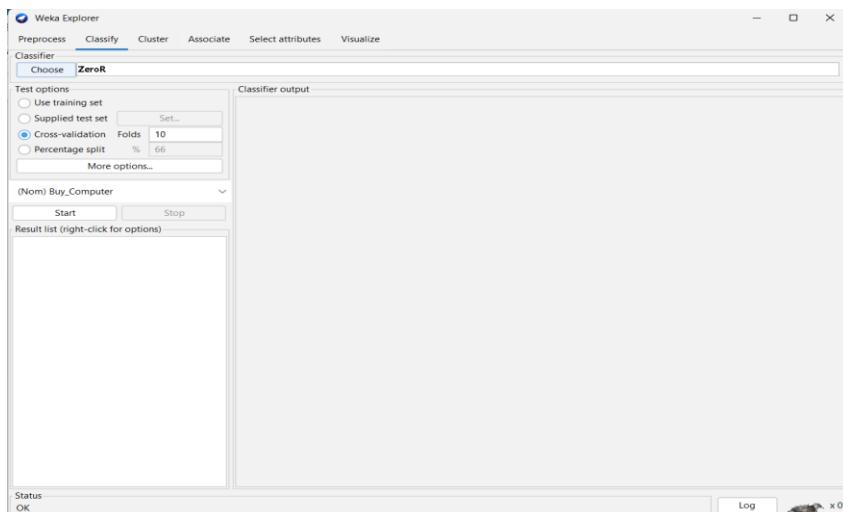
Viewer

Relation: Buy_Computer

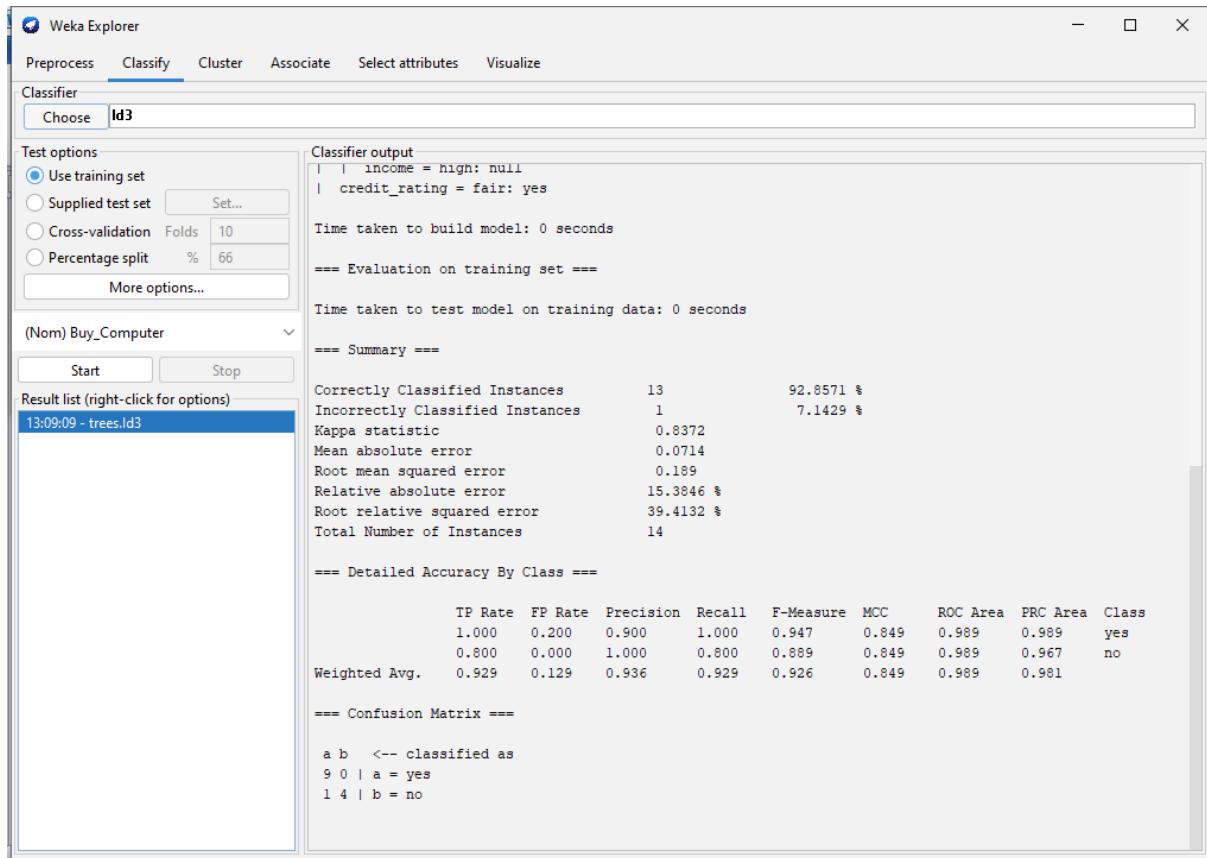
No.	1: age	2: income	3: student	4: credit_rating	5: Buy_Computer
	Nominal	Nominal	Nominal	Nominal	Nominal
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle...	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	excellent	yes
6	senior	low	yes	excellent	no
7	middle...	low	yes	fair	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle...	medium	no	excellent	yes
13	middle...	high	yes	fair	yes
14	senior	medium	no	excellent	no

Add instance Undo OK Cancel

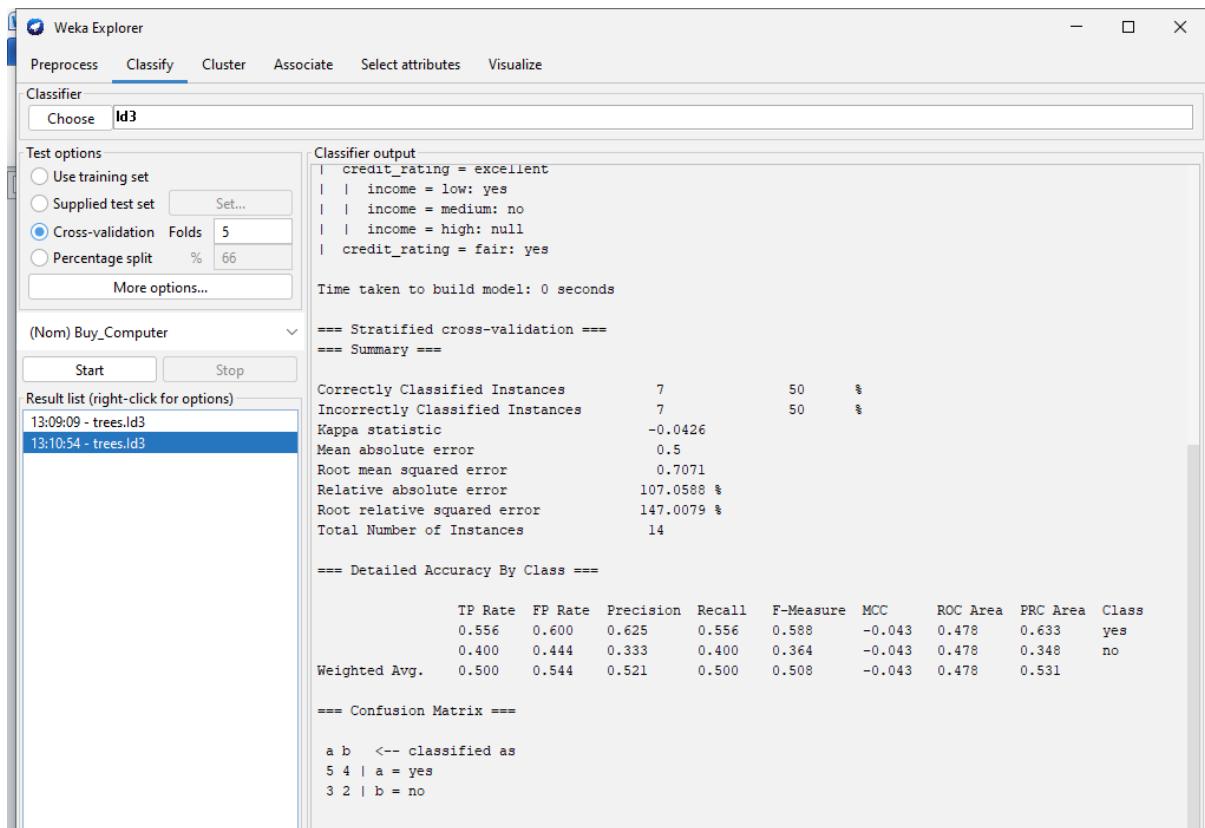
4. Please open the Classify window to utilize the ID3 Decision Tree algorithm for classifying the dataset.



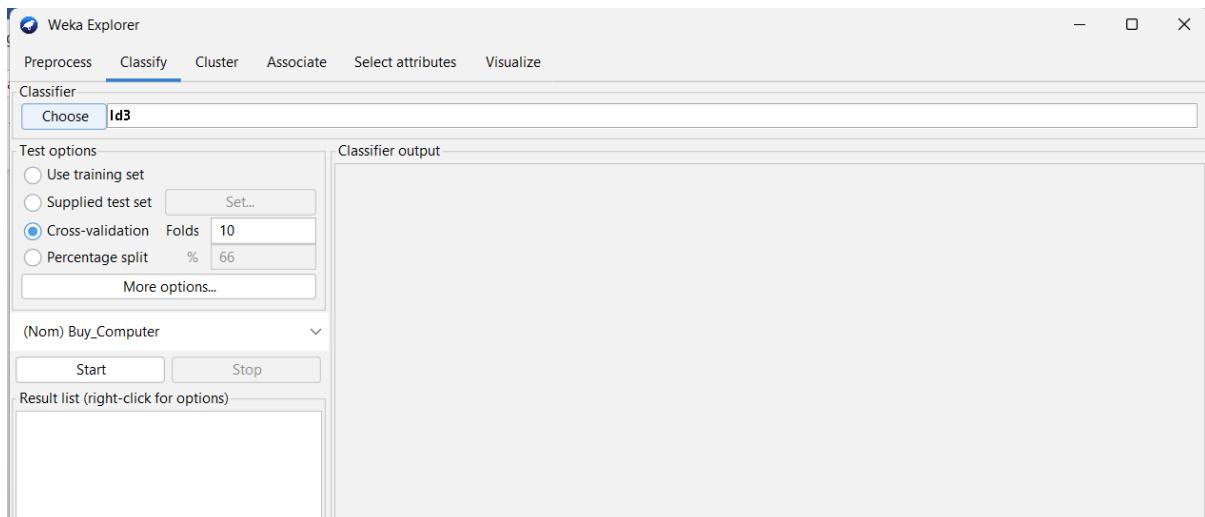
5. Click on "Choose" and select "trees > ID3" to choose the package for the ID3 Decision Tree algorithm.



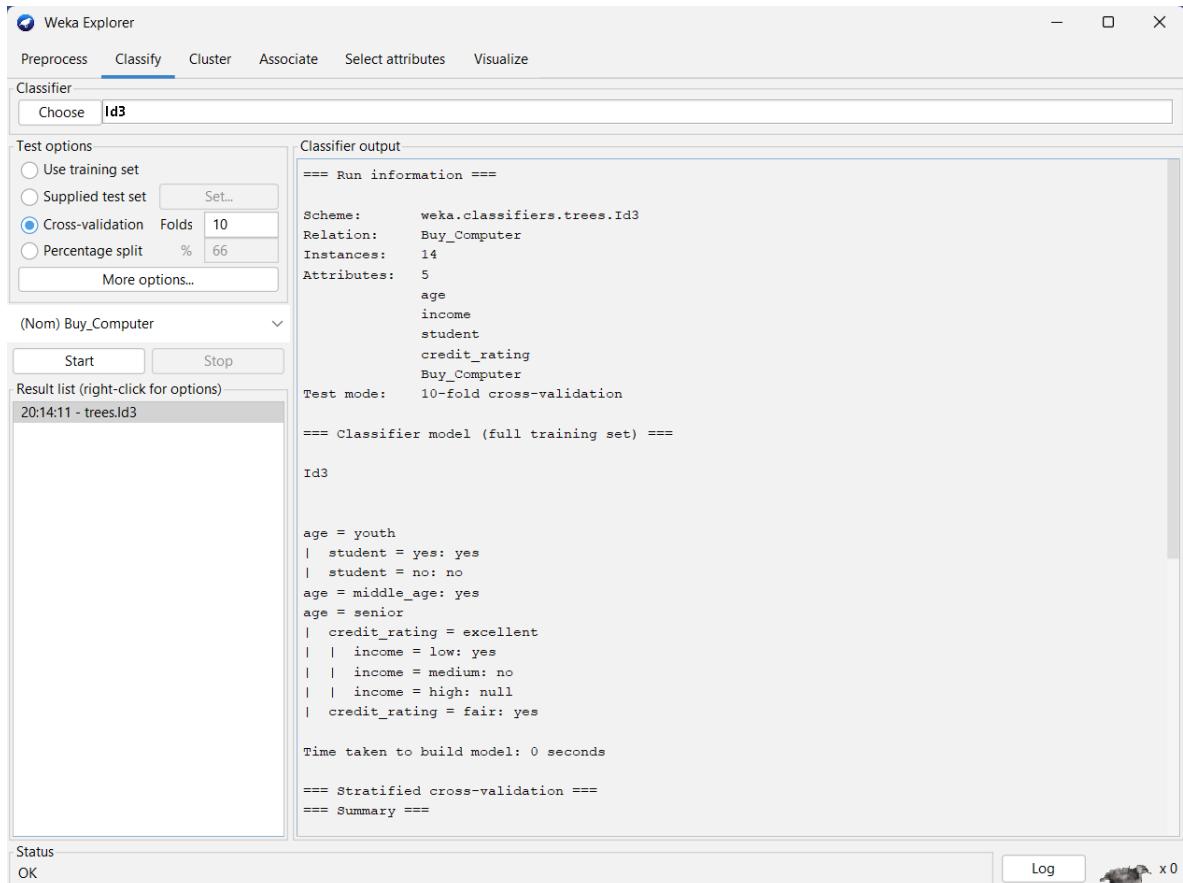
6. Switch the test options to Cross Validation mode (Folds : 5).



7. Switch the test options to Cross Validation mode (Folds : 10 Default).



8. Initiate the classification process by clicking on the "Start" button.



```
Classifier output
| credit_rating = excellent
| | income = low: yes
| | income = medium: no
| | income = high: null
| credit_rating = fair: yes

Time taken to build model: 0 seconds

== Stratified cross-validation ==
== Summary ==

Correctly Classified Instances          8           57.1429 %
Incorrectly Classified Instances        6           42.8571 %
Kappa statistic                         0.0667
Mean absolute error                     0.4286
Root mean squared error                 0.6547
Relative absolute error                  90          %
Root relative squared error            132.6919 %
Total Number of Instances               14

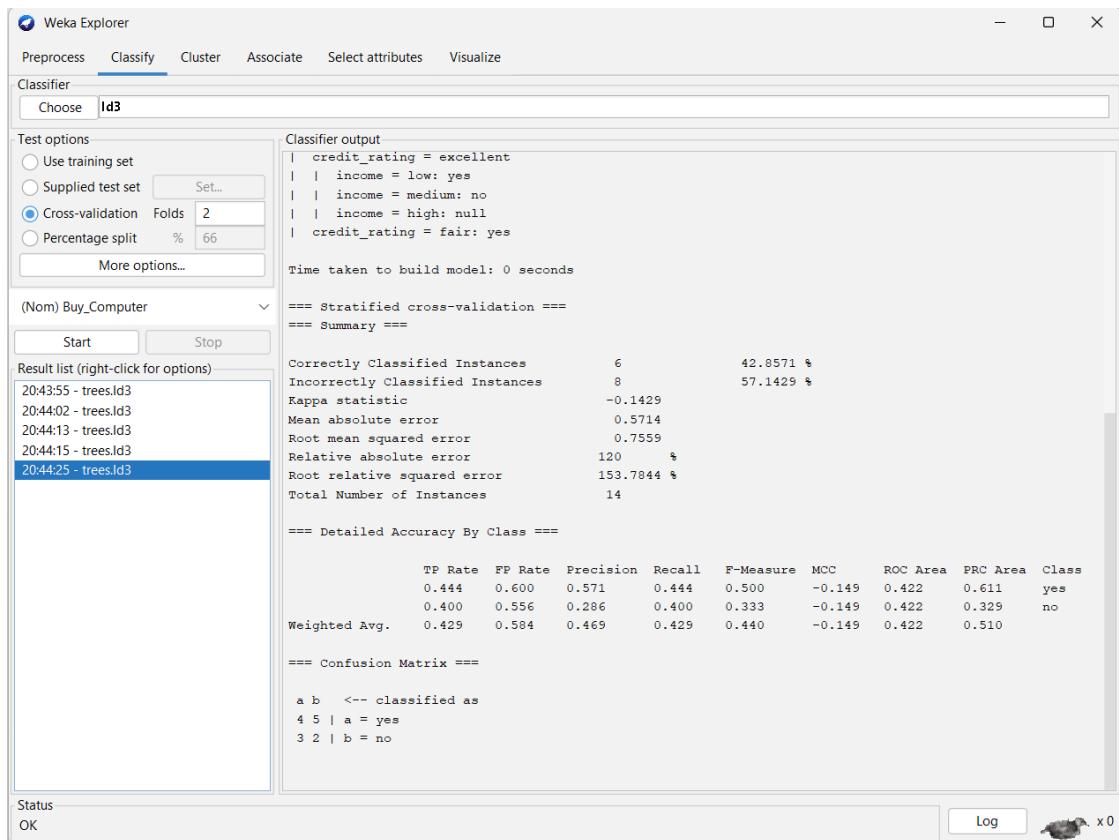
== Detailed Accuracy By Class ==

      TP Rate   FP Rate   Precision   Recall    F-Measure   MCC     ROC Area   PRC Area   Class
      0.667     0.600     0.667     0.667     0.667     0.067     0.533     0.659     yes
      0.400     0.333     0.400     0.400     0.400     0.067     0.533     0.374     no
Weighted Avg.   0.571     0.505     0.571     0.571     0.571     0.067     0.533     0.557

== Confusion Matrix ==

a b    <-- classified as
6 3 | a = yes
3 2 | b = no
```

9. Modify the fold parameter from 10 to any random number between 1 and 14, considering that there are 14 instances in the dataset.



10. To obtain varying outputs, alter the fold parameter multiple times.

Experiment-7

Aim: Generate frequent item set using Apriori Algorithm and also generate association rules for any market basket data

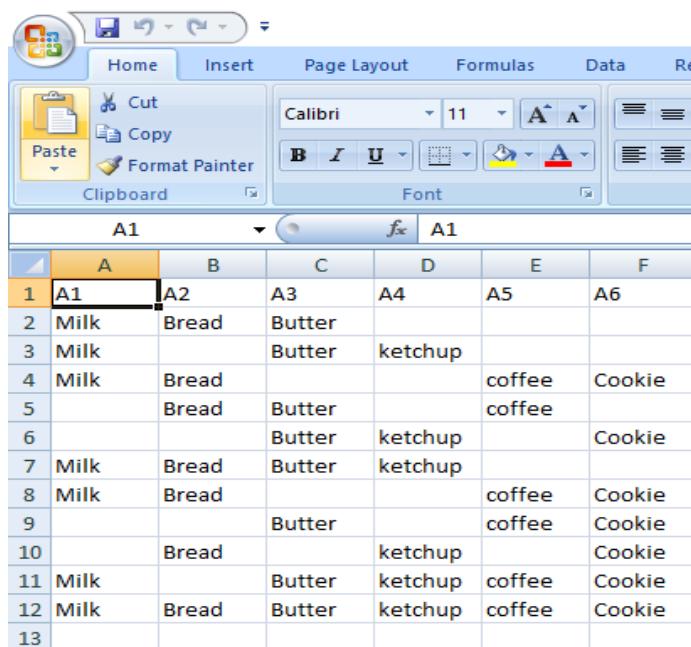
Description:

Apriori is the first association rule mining algorithm that uses support-based pruning to systematically control the exponential growth of candidate item sets.

And the apriori principle states that "If an item set is frequent, then all of its subsets must also be frequent".

Process:

1. First create the market basket data(transaction data set) and save it in the CSV form

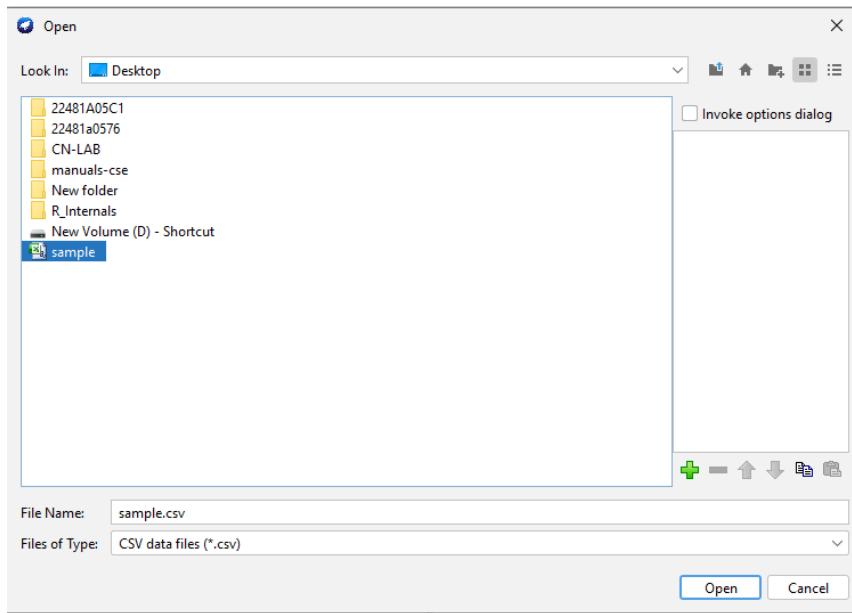


The screenshot shows a Microsoft Excel spreadsheet with data in rows 1 through 13. The columns are labeled A through F. The data represents a market basket dataset with the following entries:

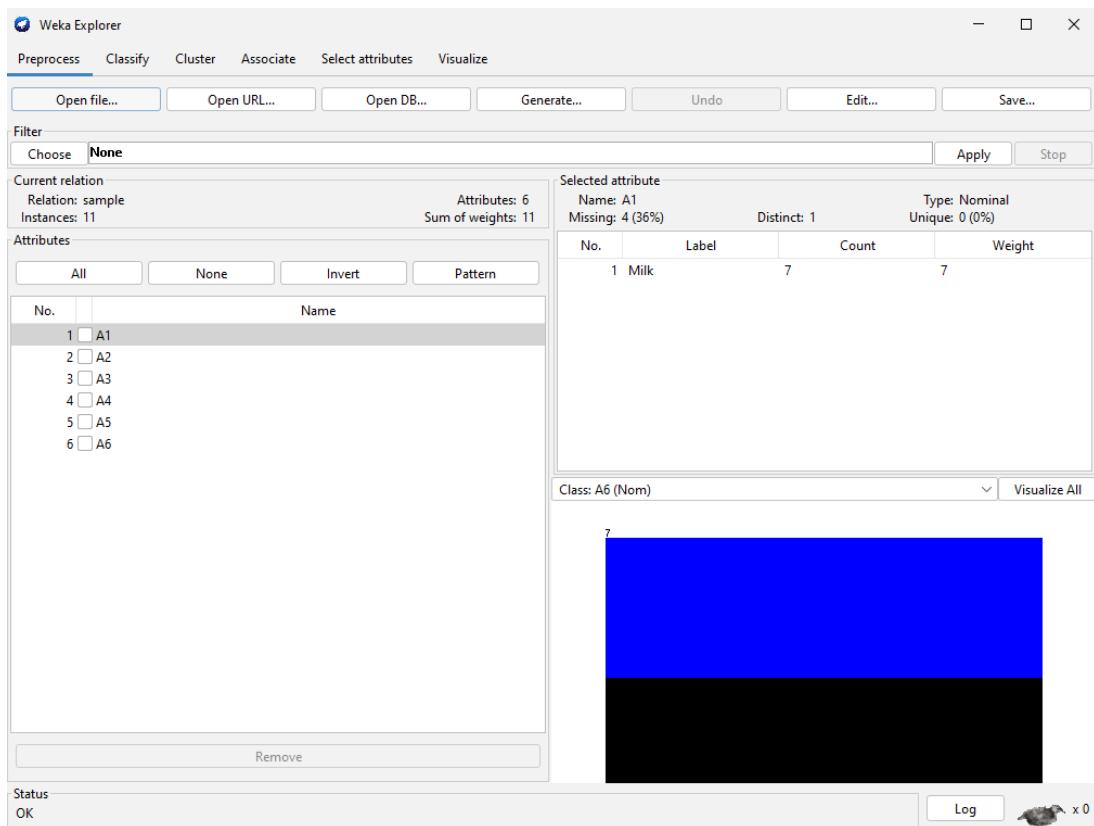
	A	B	C	D	E	F
1	A1	A2	A3	A4	A5	A6
2	Milk	Bread	Butter			
3	Milk		Butter	ketchup		
4	Milk	Bread			coffee	Cookie
5		Bread	Butter		coffee	
6			Butter	ketchup		Cookie
7	Milk	Bread	Butter	ketchup		
8	Milk	Bread			coffee	Cookie
9			Butter		coffee	Cookie
10		Bread		ketchup		Cookie
11	Milk		Butter	ketchup	coffee	Cookie
12	Milk	Bread	Butter	ketchup	coffee	Cookie
13						

2. Open weka tool>explore>and load the dataset which has been created

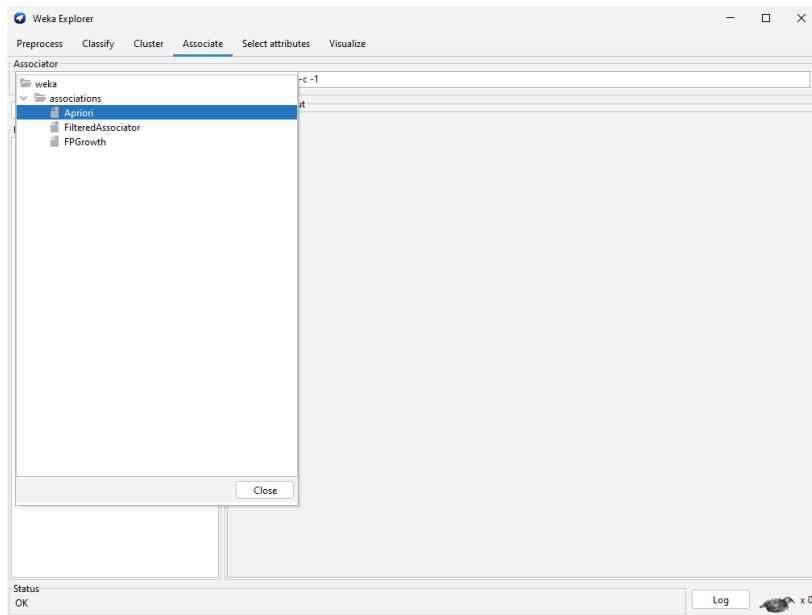




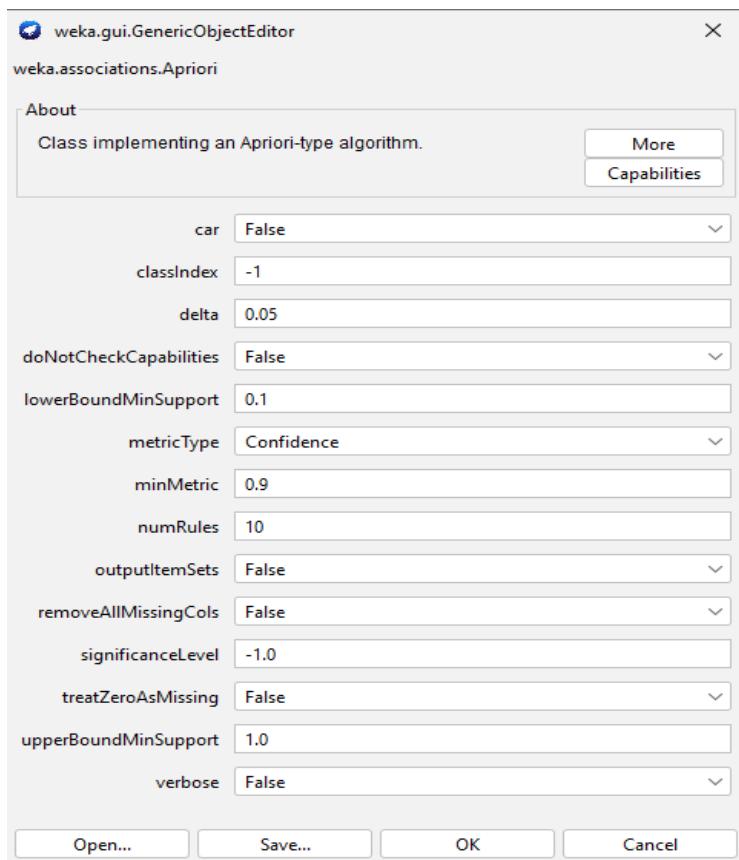
3. Once the dataset is loaded you will observe the window as shown below

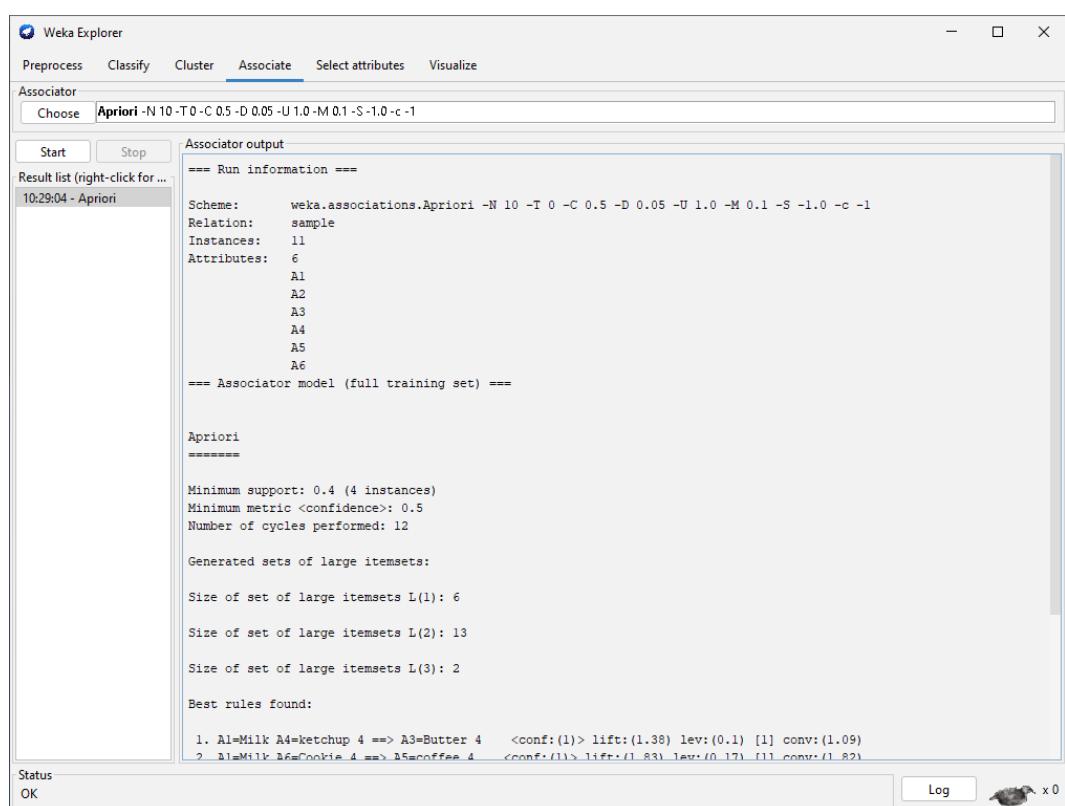
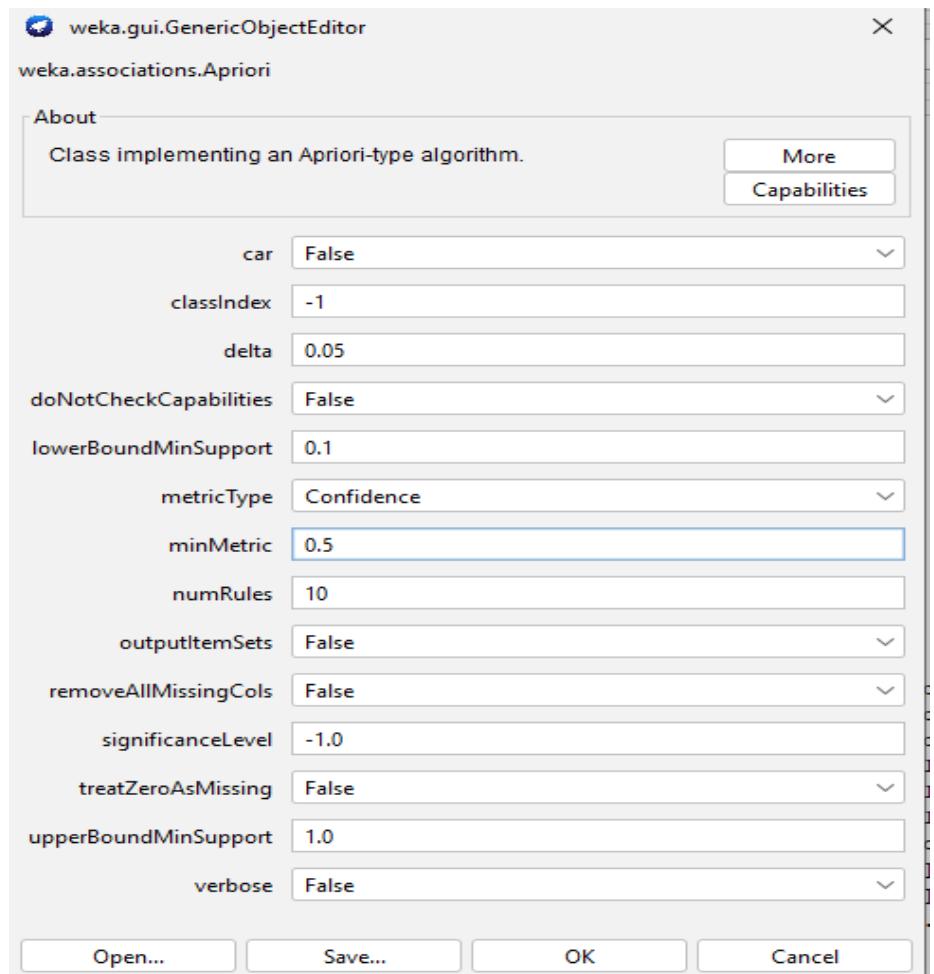


4. After this as we have to apply the association rules go to associate and click on choose and select the Apriori algorithm



- Once you click on the Apriori you get a window where you can adjust the features such as support and confidence. Adjust these features and now apply to see the output.





Weka Explorer

Preprocess Classify Cluster Associate **Select attributes** Visualize

Associate

Choose **Apriori -N 10 -T 0 -C 0.5 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1**

Start Stop

Result list (right-click for ...)
10:29:04 - Apriori

Associate output

```
A5
A6
==> Associate model (full training set) ==>

Apriori
=====

Minimum support: 0.4 (4 instances)
Minimum metric <confidence>: 0.5
Number of cycles performed: 12

Generated sets of large itemsets:

Size of set of large itemsets L(1): 6
Size of set of large itemsets L(2): 13
Size of set of large itemsets L(3): 2

Best rules found:

1. A1=Milk A4=ketchup 4 ==> A3=Butter 4    <conf:(1)> lift:(1.38) lev:(0.1) [1] conv:(1.09)
2. A1=Milk A5=Cookie 4 ==> A5=coffee 4    <conf:(1)> lift:(1.83) lev:(0.17) [1] conv:(1.82)
3. A1=Milk A5=coffee 4 ==> A6=Cookie 4    <conf:(1)> lift:(1.57) lev:(0.13) [1] conv:(1.45)
4. A4=ketchup 6 ==> A3=Butter 5    <conf:(0.83)> lift:(1.15) lev:(0.06) [0] conv:(0.82)
5. A5=coffee 6 ==> A6=Cookie 5    <conf:(0.83)> lift:(1.31) lev:(0.11) [1] conv:(1.09)
6. A3=Butter A4=ketchup 5 ==> A1=Milk 4    <conf:(0.8)> lift:(1.26) lev:(0.07) [0] conv:(0.91)
7. A1=Milk A3=Butter 5 ==> A4=ketchup 4    <conf:(0.8)> lift:(1.47) lev:(0.12) [1] conv:(1.14)
8. A5=coffee A6=Cookie 5 ==> A1=Milk 4    <conf:(0.8)> lift:(1.26) lev:(0.07) [0] conv:(0.91)
9. A2=Bread 7 ==> A1=Milk 5    <conf:(0.71)> lift:(1.12) lev:(0.05) [0] conv:(0.85)
10. A1=Milk 7 ==> A2=Bread 5    <conf:(0.71)> lift:(1.12) lev:(0.05) [0] conv:(0.85)
```

Status OK

Log x 0

6. The output window results the below sections

- I. Run information
- II. Associate Model(apriori)
- III. Best rules found

Experiment -8

Aim: Generate frequent item set using FP growth Algorithm

Description:

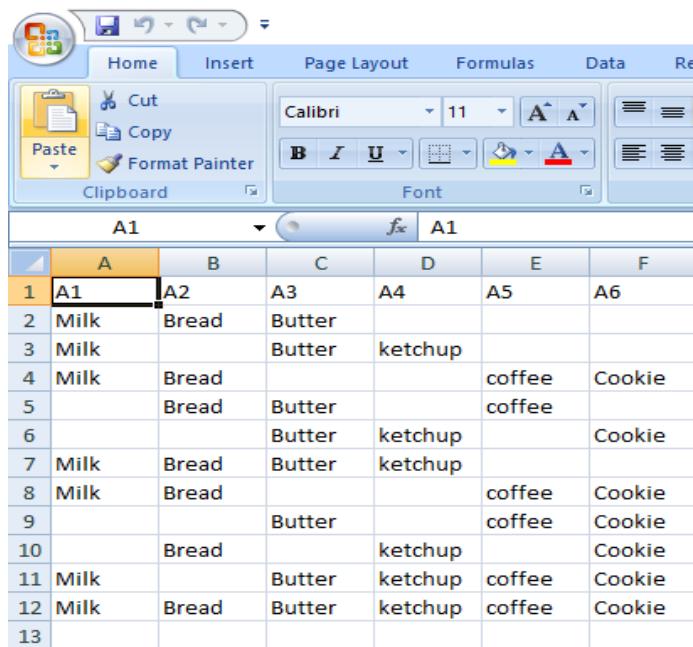
FP growth encodes the data set using a **compact data structure** called an FP-tree and extracts **frequent itemsets** directly from this structure.

And the representation of the FP tree is as follows:

- An FP-tree is a compressed representation of the input data.
- It is constructed by reading the data set one transaction at a time and mapping each transaction onto a path in the FP-tree.
- As different transactions have several items in common, **their paths may overlap**

Process:

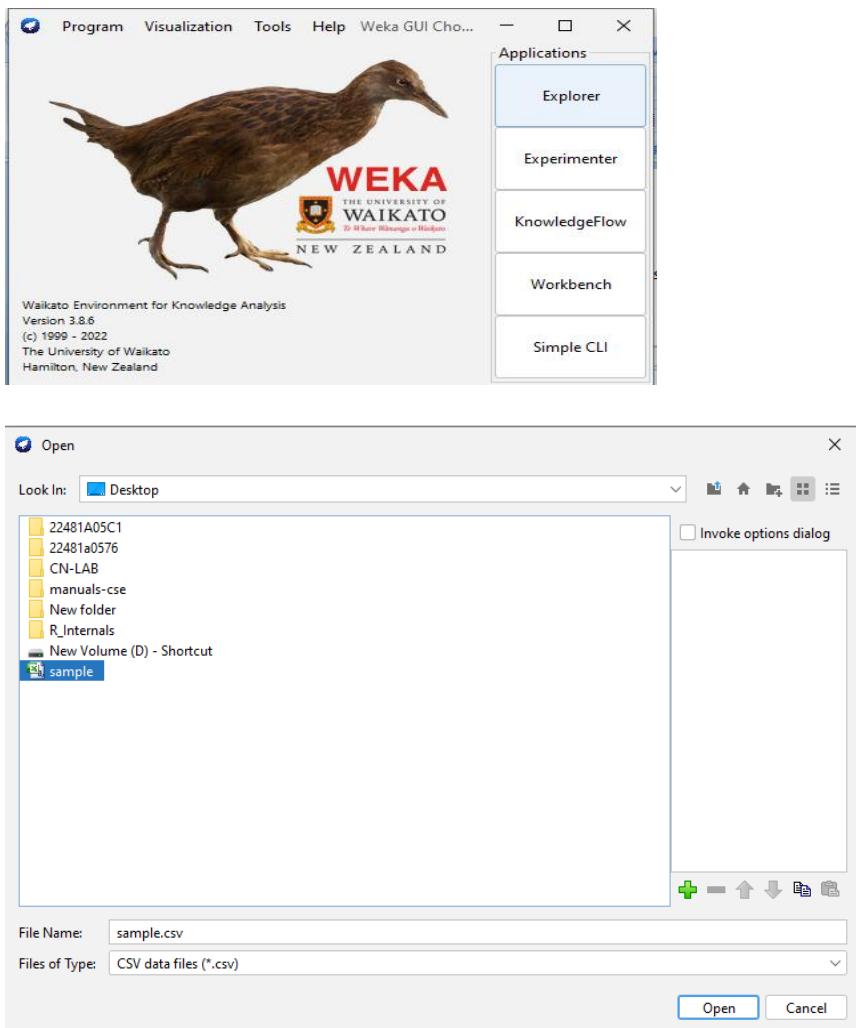
7. First create the market basket data(transaction data set) and save it in the CSV form



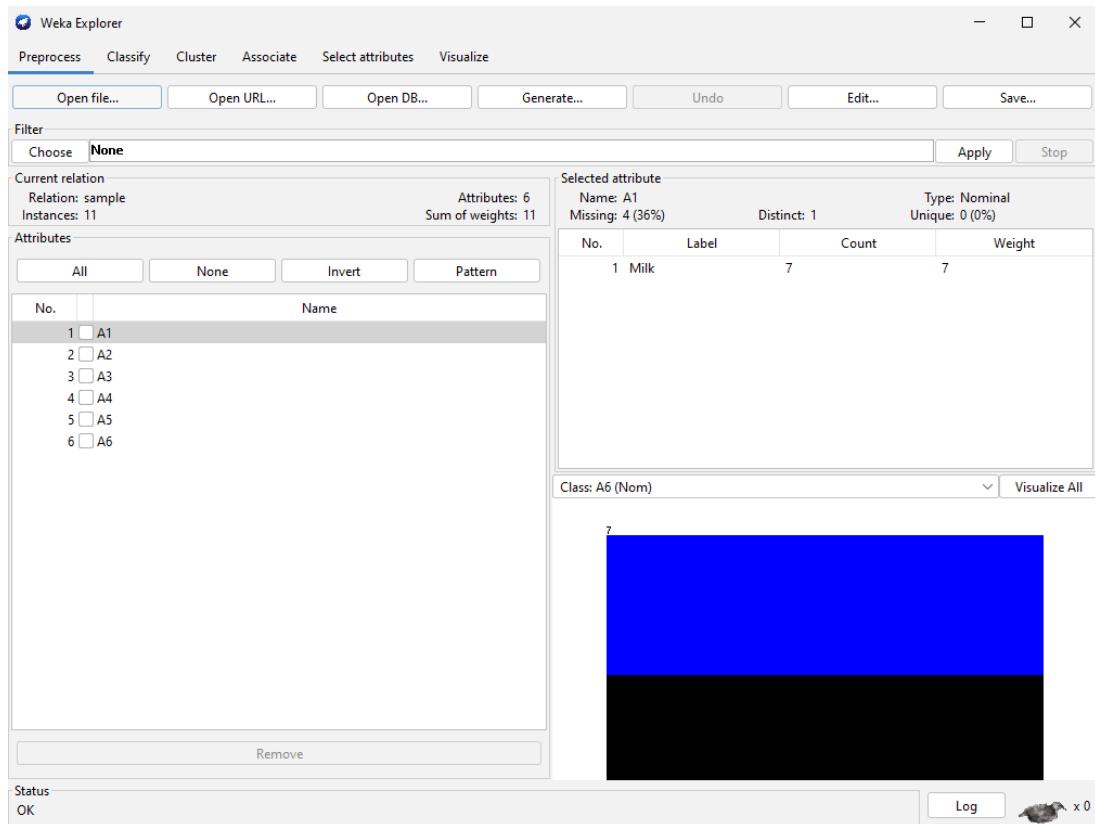
The screenshot shows a Microsoft Excel spreadsheet with the following data:

	A	B	C	D	E	F
1	A1	A2	A3	A4	A5	A6
2	Milk	Bread	Butter			
3	Milk		Butter	ketchup		
4	Milk	Bread			coffee	Cookie
5		Bread	Butter		coffee	
6			Butter	ketchup		Cookie
7	Milk	Bread	Butter	ketchup		
8	Milk	Bread			coffee	Cookie
9			Butter		coffee	Cookie
10		Bread		ketchup		Cookie
11	Milk		Butter	ketchup	coffee	Cookie
12	Milk	Bread	Butter	ketchup	coffee	Cookie
13						

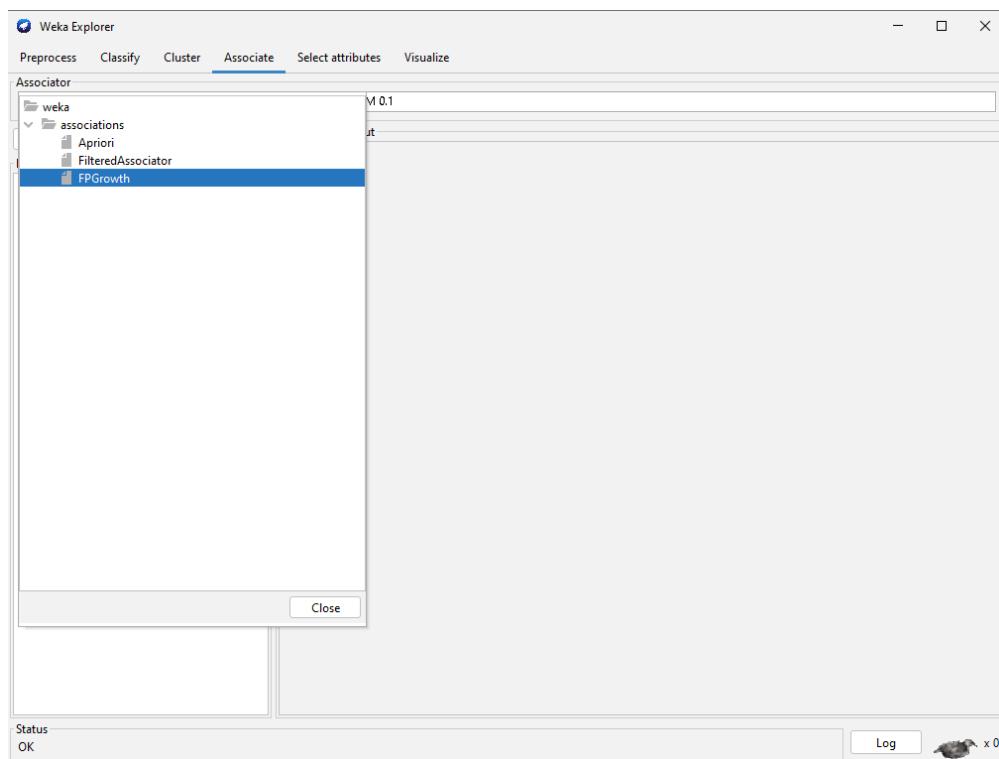
8. Open weka tool>explore>and load the dataset which has be created



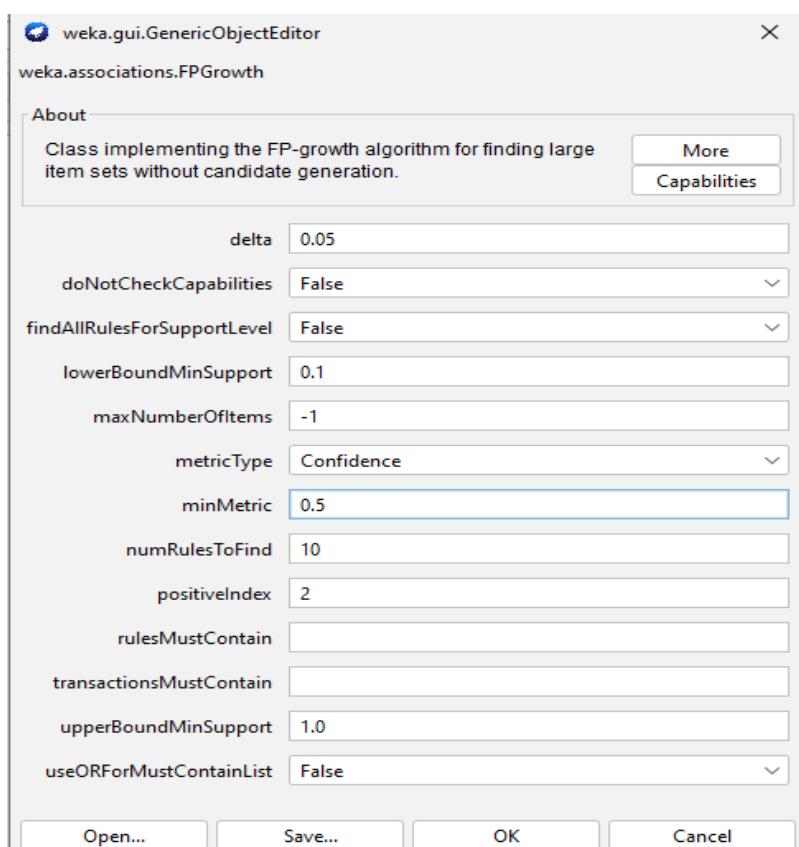
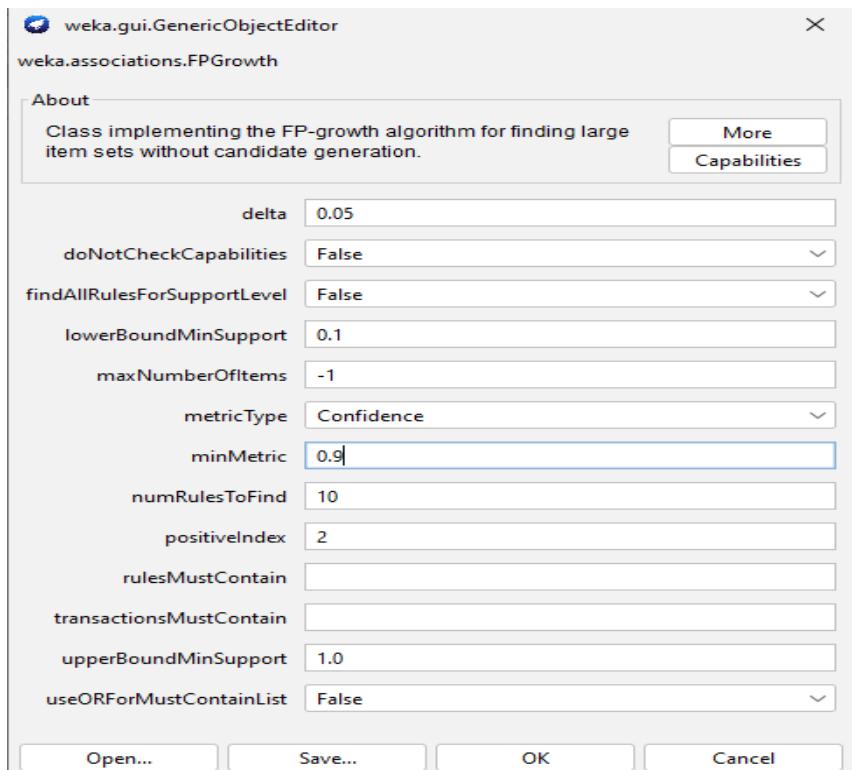
9. Once the dataset is loaded you will observe the window as shown below



- After this as we have to apply the association rules go to associate and click on choose and select the FP growth algorithm



- Once you click on the FP growth you get a window where you can adjust the features such as support and confidence. Adjust these features and now apply to see the output.



Weka Explorer

Preprocess Classify Cluster **Associate** Select attributes Visualize

Associator
Choose **FPGrowth -P 2 -I -1 -N 10 -T 0 -C 0.5 -D 0.05 -U 1.0 -M 0.1**

Start Stop

Result list (right-click for ...)
10:43:44 - FPGrowth

Associator output

```
==== Run information ====
Scheme: weka.associations.FPGrowth -P 2 -I -1 -N 10 -T 0 -C 0.5 -D 0.05 -U 1.0 -M 0.1
Relation: sample
Instances: 11
Attributes: 6
A1
A2
A3
A4
A5
A6
==== Associator model (full training set) ====
FPGrowth found 38 rules (displaying top 10)

1. [A1=Milk, A4=ketchup]: 4 ==> [A3=Butter]: 4 <conf:(1)> lift:(1.38) lev:(0.1) conv:(1.09)
2. [A6=Cookie, A1=Milk]: 4 ==> [A5=coffee]: 4 <conf:(1)> lift:(1.83) lev:(0.17) conv:(1.82)
3. [A1=Milk, A5=coffee]: 4 ==> [A6=Cookie]: 4 <conf:(1)> lift:(1.57) lev:(0.13) conv:(1.45)
4. [A4=ketchup]: 6 ==> [A3=Butter]: 5 <conf:(0.83)> lift:(1.15) lev:(0.06) conv:(0.82)
5. [A5=coffee]: 6 ==> [A6=Cookie]: 5 <conf:(0.83)> lift:(1.31) lev:(0.11) conv:(1.09)
6. [A3=Butter, A1=Milk]: 5 ==> [A4=ketchup]: 4 <conf:(0.8)> lift:(1.47) lev:(0.12) conv:(1.14)
7. [A3=Butter, A4=ketchup]: 5 ==> [A1=Milk]: 4 <conf:(0.8)> lift:(1.26) lev:(0.07) conv:(0.91)
8. [A6=Cookie, A5=coffee]: 5 ==> [A1=Milk]: 4 <conf:(0.8)> lift:(1.26) lev:(0.07) conv:(0.91)
9. [A1=Milk]: 7 ==> [A3=Butter]: 5 <conf:(0.71)> lift:(0.98) lev:(-0.01) conv:(0.64)
10. [A6=Cookie]: 7 ==> [A5=coffee]: 5 <conf:(0.71)> lift:(1.31) lev:(0.11) conv:(1.06)
```

Status OK Log  x 0

12. The output window results the below sections

- IV. Run information
- V. Associator Model(FP growth)
- VI. Best rules found

Experiment -9

Aim: Experiment to compare the performance of various data mining algorithm on any data set

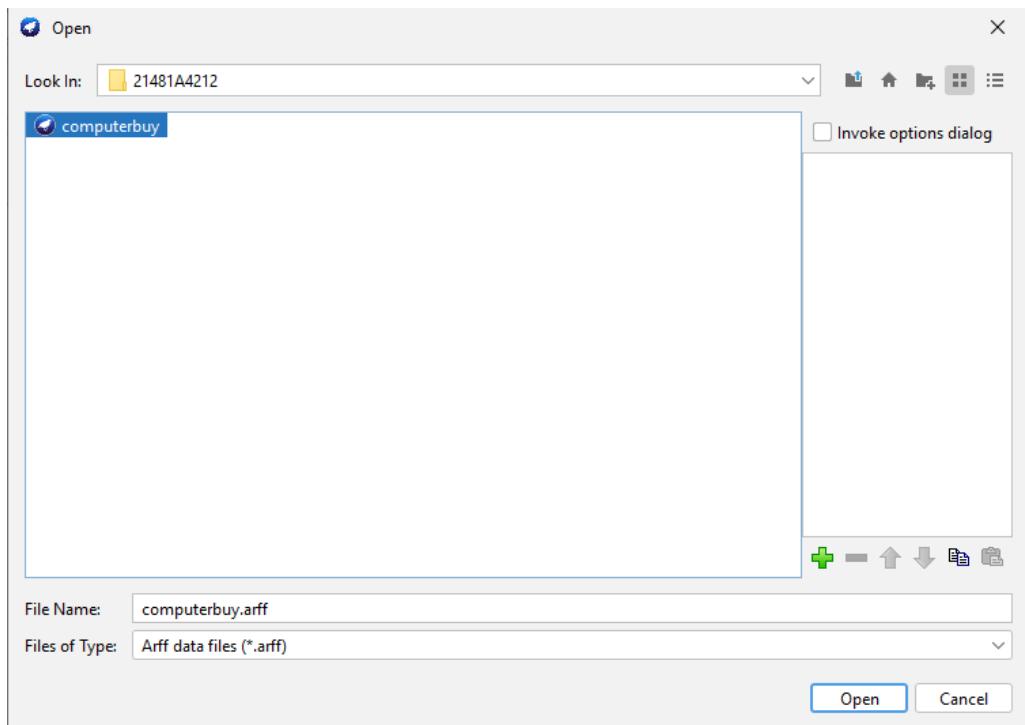
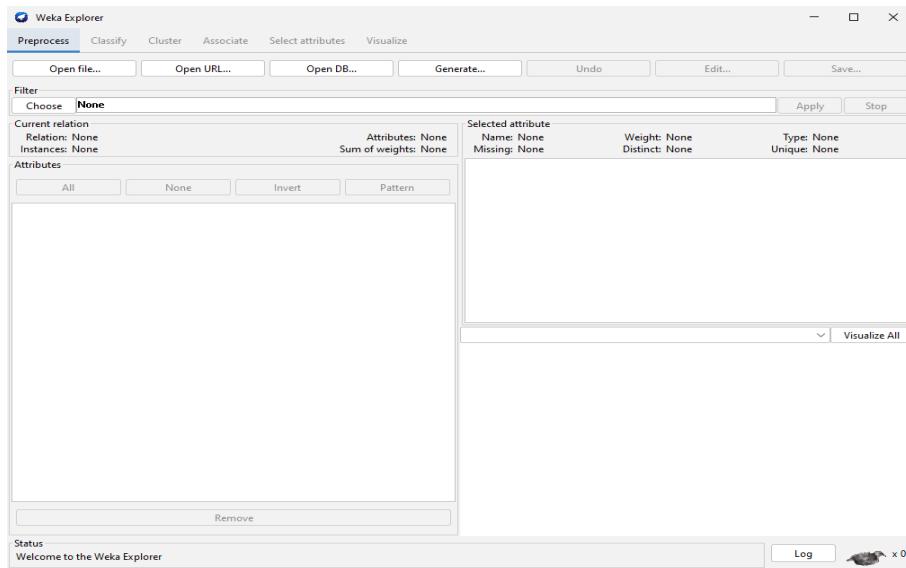
Description:

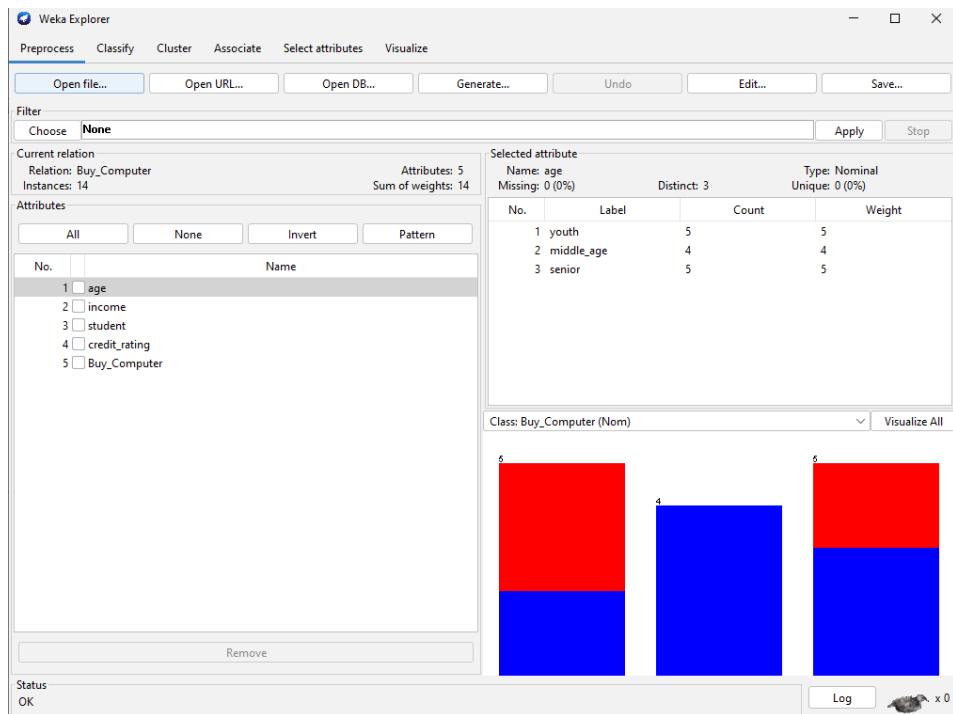
Data Mining Algorithms are a particular category of algorithms useful for analyzing data and developing data models to identify meaningful patterns. These are part of machine learning algorithms. These algorithms are implemented through various programming like R language, Python, and data mining tools to derive the optimized data models. Some of the data mining algorithms are C4.5 for decision trees, K-means for cluster data analysis, Naive Bayes Algorithm and many more and these algorithms are based upon statistical and mathematical formulas which applied to the data set.

Process:

1. Open weka tool >explore>choose file and load the .arrf file





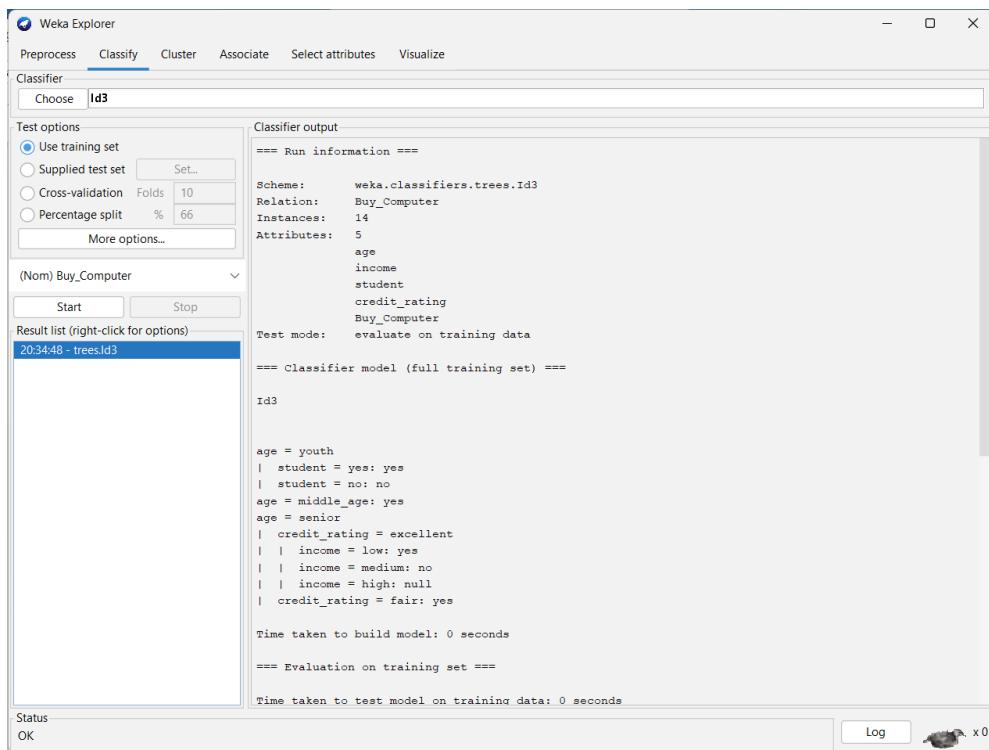


2. One of the outputs of the classifications algorithm is confusion matrix and the general structure looks like as shown below

		Predicted condition	
		Positive (PP)	Negative (PN)
Total population = P + N			
Actual condition	Positive (P)	True positive (TP)	False negative (FN)
	Negative (N)	False positive (FP)	True negative (TN)

3. Now perform the below classification algorithms on the data set
- ID3 classification
 - J48 classification
 - Navies Bayes classification

4. The output window of the ID3 classification is shown below



The screenshot shows the Weka Explorer interface with the 'Classify' tab selected. Under 'Classifier', 'Choose' is set to 'Id3'. In the 'Test options' section, 'Use training set' is checked. The 'Relation' dropdown shows '(Nom) Buy_Computer'. The 'Result list' pane displays '20:34:48 - treesId3'. The 'Classifier output' pane contains the following text:

```

    === Run information ===
    Scheme: weka.classifiers.trees.Id3
    Relation: Buy_Computer
    Instances: 14
    Attributes: 5
    age
    income
    student
    credit_rating
    Buy_Computer
    Test mode: evaluate on training data

    === Classifier model (full training set) ===

    id3

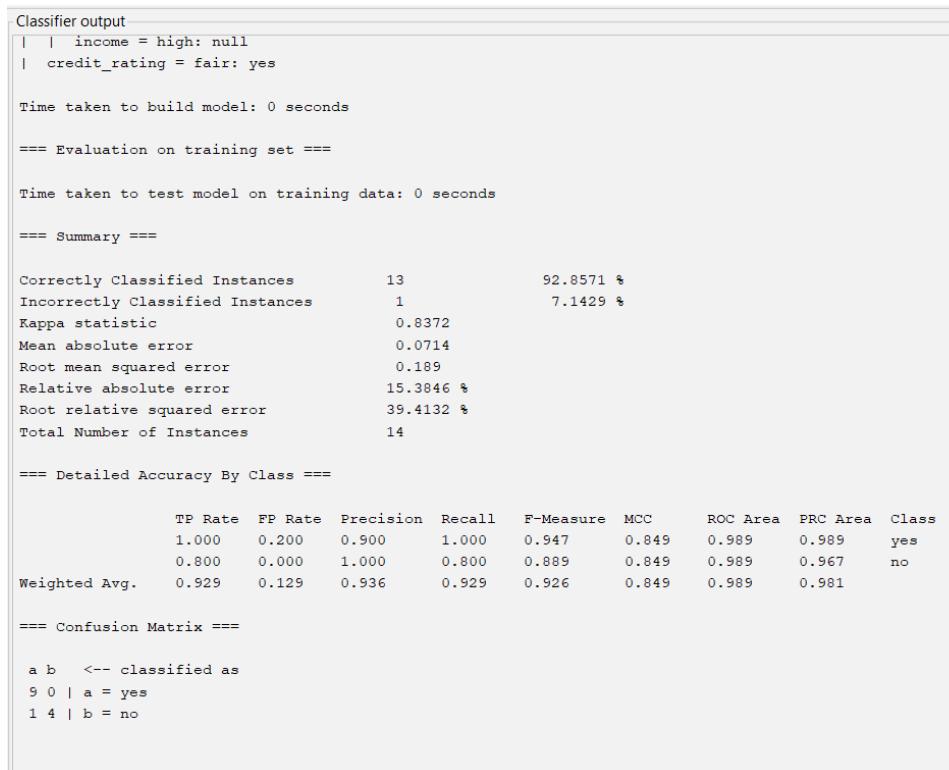
    age = youth
    | student = yes: yes
    | student = no: no
    age = middle_age: yes
    age = senior
    | credit_rating = excellent
    | | income = low: yes
    | | income = medium: no
    | | income = high: null
    | credit_rating = fair: yes

    Time taken to build model: 0 seconds

    === Evaluation on training set ===

    Time taken to test model on training data: 0 seconds
  
```

Status: OK



The screenshot shows the 'Classifier output' window with the following content:

```

    Classifier output
    | | income = high: null
    | credit_rating = fair: yes

    Time taken to build model: 0 seconds

    === Evaluation on training set ===

    Time taken to test model on training data: 0 seconds

    === Summary ===

    Correctly Classified Instances      13          92.8571 %
    Incorrectly Classified Instances   1          7.1429 %
    Kappa statistic                   0.8372
    Mean absolute error               0.0714
    Root mean squared error          0.189
    Relative absolute error           15.3846 %
    Root relative squared error      39.4132 %
    Total Number of Instances        14

    === Detailed Accuracy By Class ===

    TP Rate   FP Rate   Precision  Recall   F-Measure  MCC     ROC Area  PRC Area  Class
    1.000     0.200     0.900     1.000    0.947     0.849    0.989    0.589    yes
    0.800     0.000     1.000     0.800    0.889     0.849    0.989    0.967    no
    Weighted Avg.  0.929     0.129     0.936     0.929    0.926     0.849    0.989    0.981

    === Confusion Matrix ===

    a b  <- classified as
    9 0 | a = yes
    1 4 | b = no
  
```

5. The output window of the J48 classification is shown below

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier Choose J48 - C 0.25 -M 2

Test options

- Use training set
- Supplied test set Set...
- Cross-validation Folds 10
- Percentage split % 66

More options...

(Nom) Buy_Computer

Start Stop

Result list (right-click for options)

10:13:45 - trees.J48

Classifier output

```
== Run information ==
Scheme: weka.classifiers.trees.J48 -C 0.25 -M 2
Relation: Buy_Computer
Instances: 14
Attributes: 5
age
income
student
credit_rating
Buy_Computer
Test mode: 10-fold cross-validation

== Classifier model (full training set) ==
J48 pruned tree
-----
age = youth
| student = yes: yes (2.0)
| student = no: no (3.0)
age = middle_age: yes (4.0)
age = senior
| credit_rating = excellent: no (3.0/1.0)
| credit_rating = fair: yes (2.0)

Number of Leaves : 5
Size of the tree : 8

Time taken to build model: 0 seconds

== Stratified cross-validation ==

```

Status OK Log x 0

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize

Classifier Choose J48 - C 0.25 -M 2

Test options

- Use training set
- Supplied test set Set...
- Cross-validation Folds 10
- Percentage split % 66

More options...

(Nom) Buy_Computer

Start Stop

Result list (right-click for options)

10:13:45 - trees.J48

Classifier output

```
Number of Leaves : 5
Size of the tree : 8
Time taken to build model: 0 seconds
== Stratified cross-validation ==
== Summary ==
Correctly Classified Instances 4 28.5714 %
Incorrectly Classified Instances 10 71.4286 %
Kappa statistic -0.4286
Mean absolute error 0.6268
Root mean squared error 0.7224
Relative absolute error 131.6346 %
Root relative squared error 146.4195 %
Total Number of Instances 14

== Detailed Accuracy By Class ==

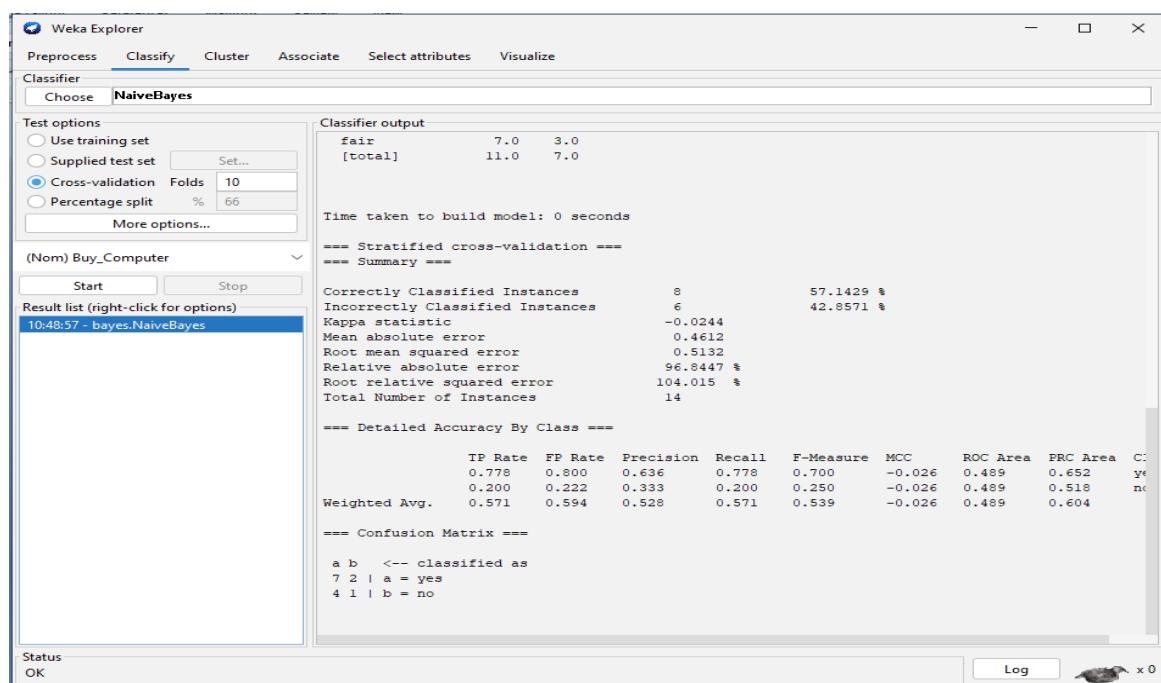
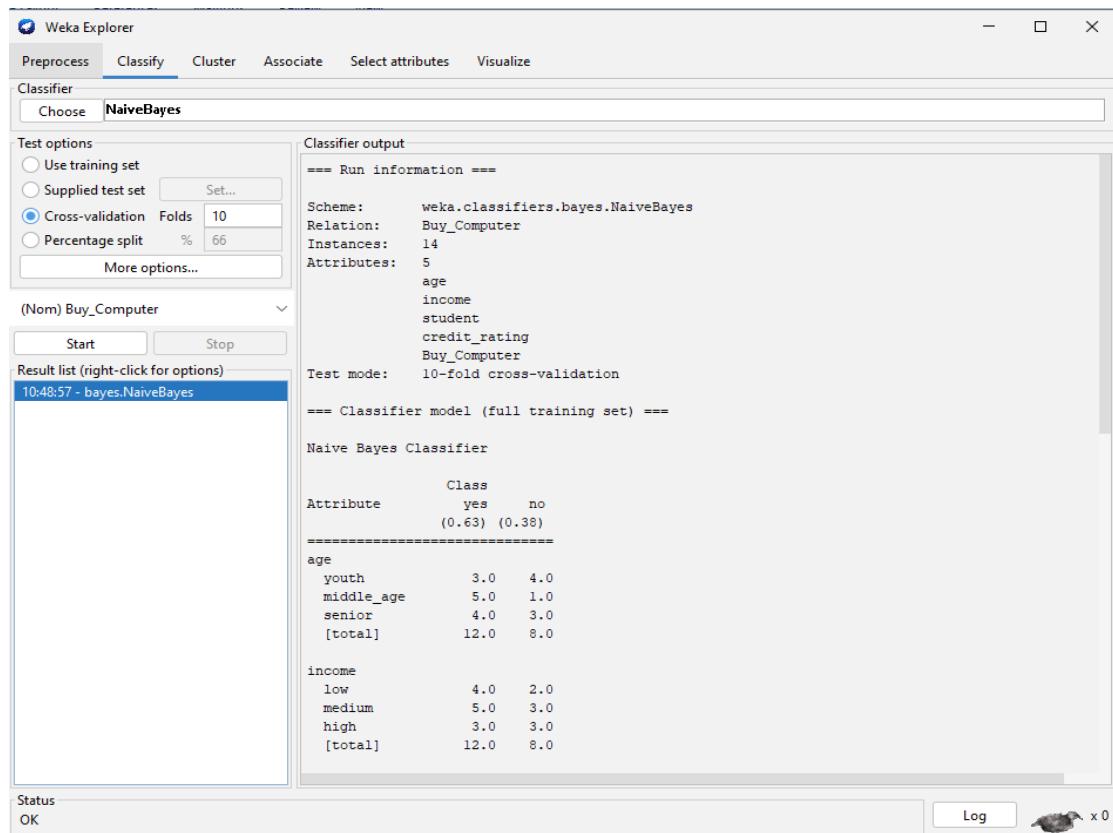
```

	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
0	0.333	0.800	0.429	0.333	0.375	-0.447	0.322	0.599	yes
1	0.200	0.667	0.143	0.200	0.167	-0.447	0.322	0.299	no
Weighted Avg.	0.286	0.752	0.327	0.286	0.301	-0.447	0.322	0.492	

```
== Confusion Matrix ==
a b  <-- classified as
3 6 | a = yes
4 1 | b = no
```

Status OK Log x 0

6. The output window of the J48 classification is shown below



- Now compare the accuracy of the three algorithms and the accuracy of ID3 has maximum compared to all classification algorithms that we had performed.

Experiment – 10

Aim:

Apply K-Means clustering algorithm on any data set by passing different k values.

Description:

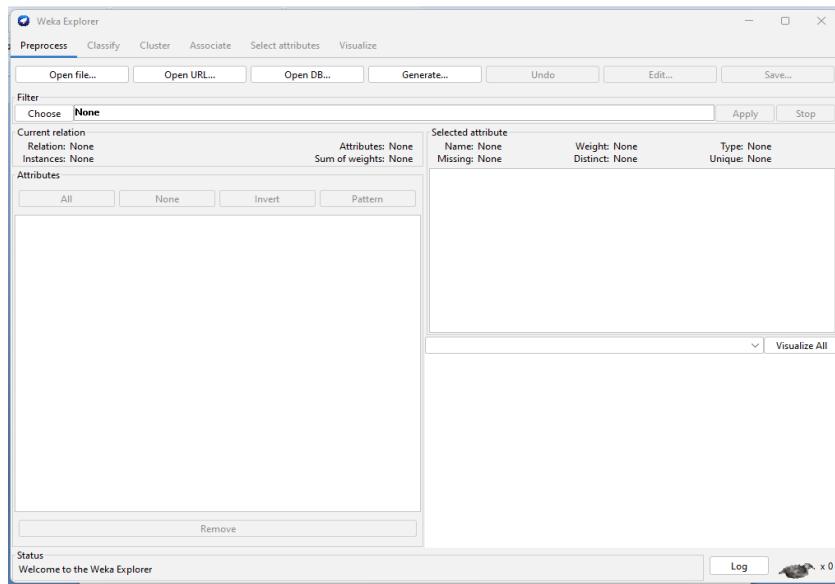
K means clustering, assigns data points to one of the K clusters depending on their distance from the center of the clusters. It starts by randomly assigning the clusters centroid in the space. Then each data point assign to one of the cluster based on its distance from centroid of the cluster. After assigning each point to one of the cluster, new cluster centroids are assigned. This process runs iteratively until it finds good cluster. In the analysis we assume that number of cluster is given in advanced and we have to put points in one of the group.

Process:

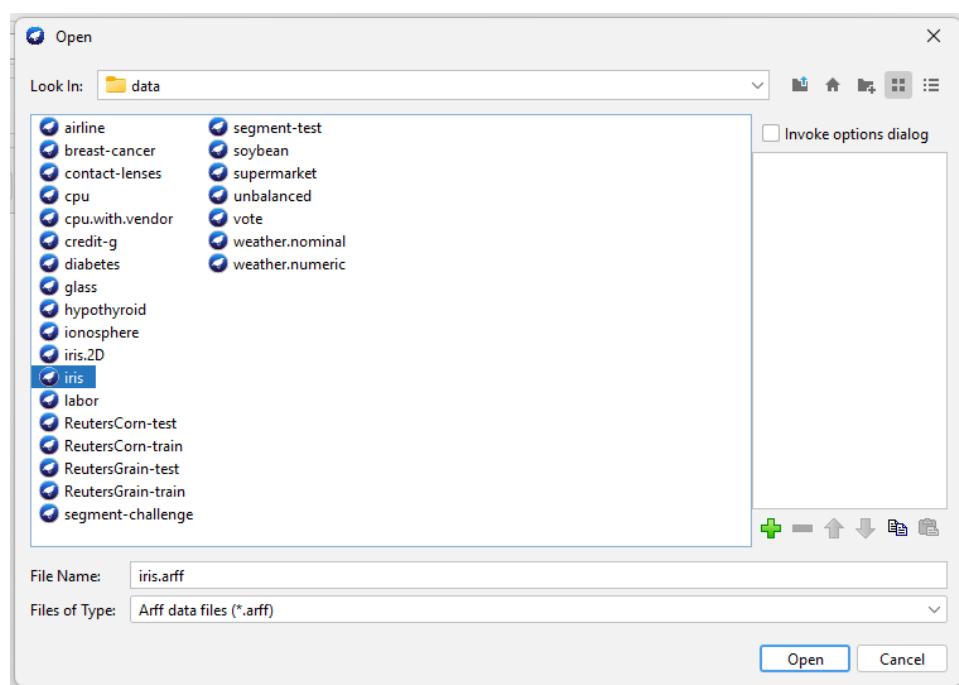
1. Open Weka tool



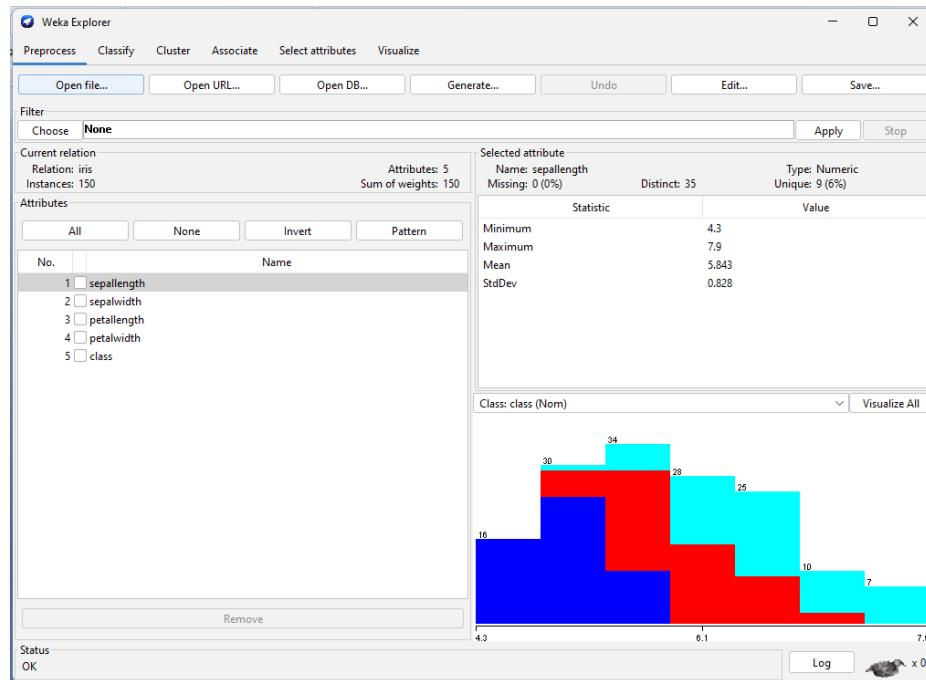
2. Goto Explorer



3. Click on "Open file"
4. Select the dataset which we have downloaded earlier (iris dataset)



5. Click on "Open"

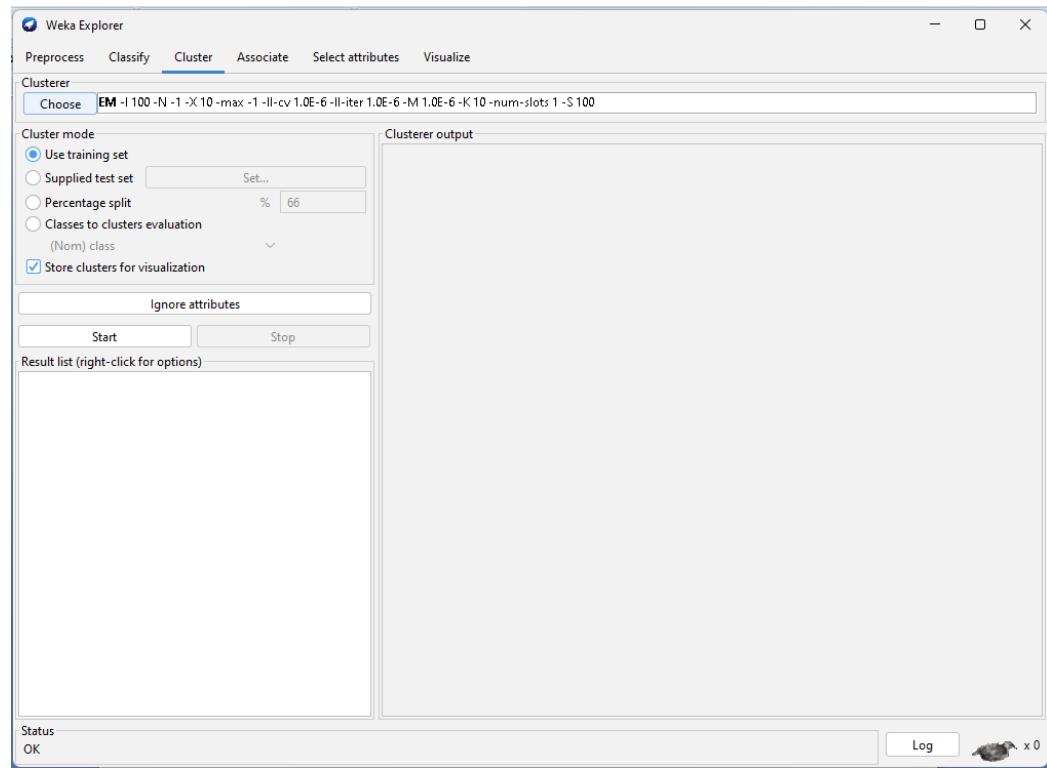


6. Click on “Edit” to view the dataset

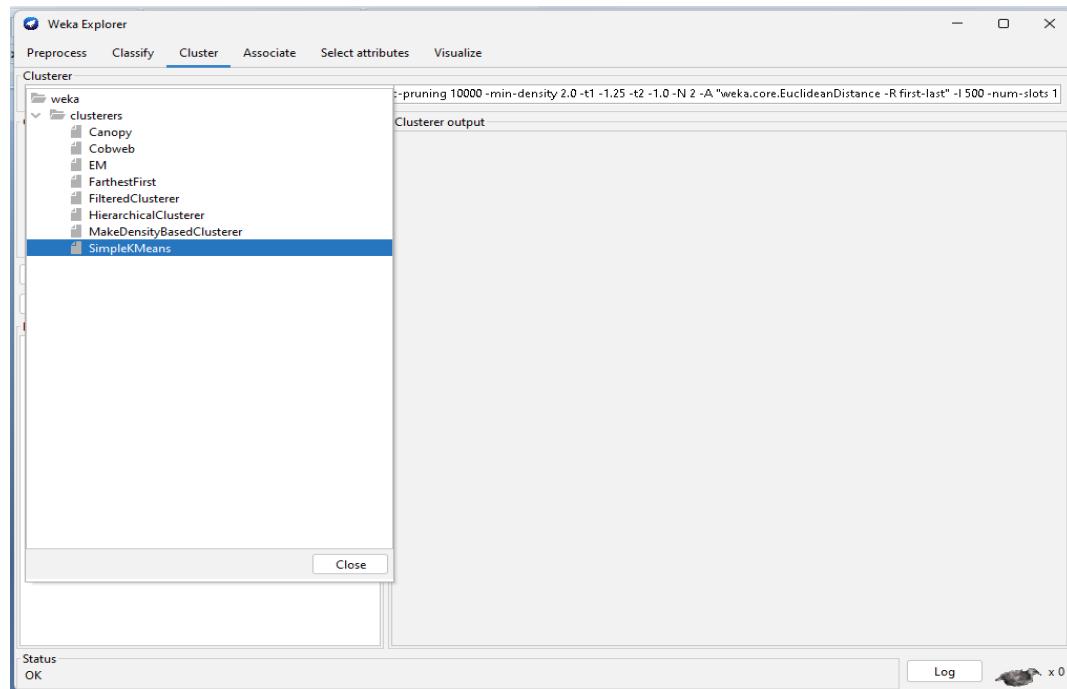
No.	1: sepal length	2: sepal width	3: petal length	4: petal width	5: class
	Numeric	Numeric	Numeric	Numeric	Nominal
1	5.1	3.5	1.4	0.2	Iris-setosa
2	4.9	3.0	1.4	0.2	Iris-setosa
3	4.7	3.2	1.3	0.2	Iris-setosa
4	4.6	3.1	1.5	0.2	Iris-setosa
5	5.0	3.6	1.4	0.2	Iris-setosa
6	5.4	3.9	1.7	0.4	Iris-setosa
7	4.6	3.4	1.4	0.3	Iris-setosa
8	5.0	3.4	1.5	0.2	Iris-setosa
9	4.4	2.9	1.4	0.2	Iris-setosa
10	4.9	3.1	1.5	0.1	Iris-setosa
11	5.4	3.7	1.5	0.2	Iris-setosa
12	4.8	3.4	1.6	0.2	Iris-setosa
13	4.8	3.0	1.4	0.1	Iris-setosa
14	4.3	3.0	1.1	0.1	Iris-setosa
15	5.8	4.0	1.2	0.2	Iris-setosa
16	5.7	4.4	1.5	0.4	Iris-setosa
17	5.4	3.9	1.3	0.4	Iris-setosa
18	5.1	3.5	1.4	0.3	Iris-setosa
19	5.7	3.8	1.7	0.3	Iris-setosa
20	5.1	3.8	1.5	0.3	Iris-setosa
21	5.4	3.4	1.7	0.2	Iris-setosa
22	5.1	3.7	1.5	0.4	Iris-setosa
23	4.6	3.6	1.0	0.2	Iris-setosa
24	5.1	3.3	1.7	0.5	Iris-setosa

Add instance Undo OK Cancel

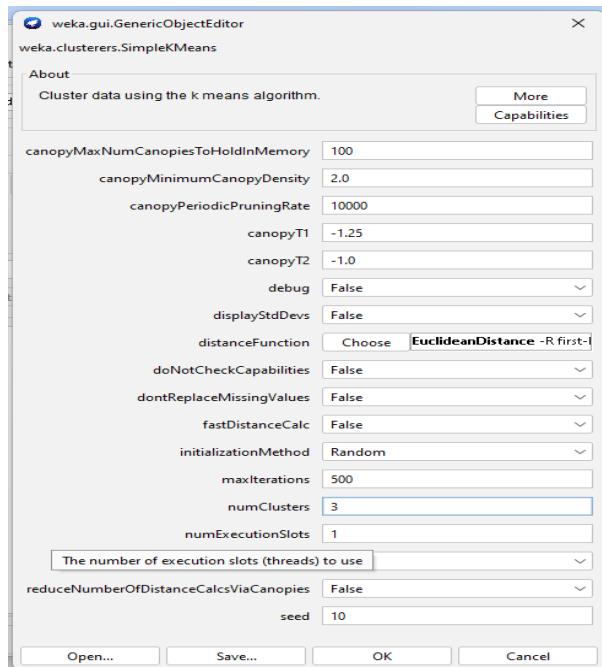
7. Now, click on “Cluster”



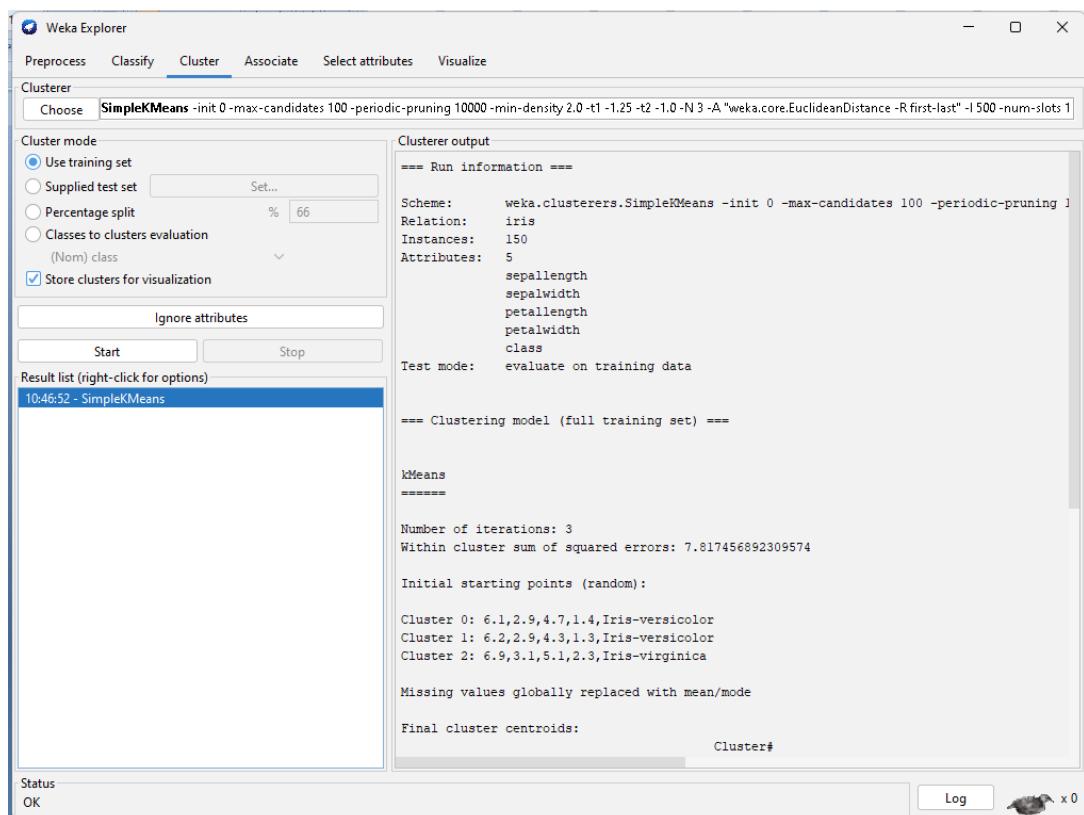
8. Goto “Choose” and select “SimpleKMeans”

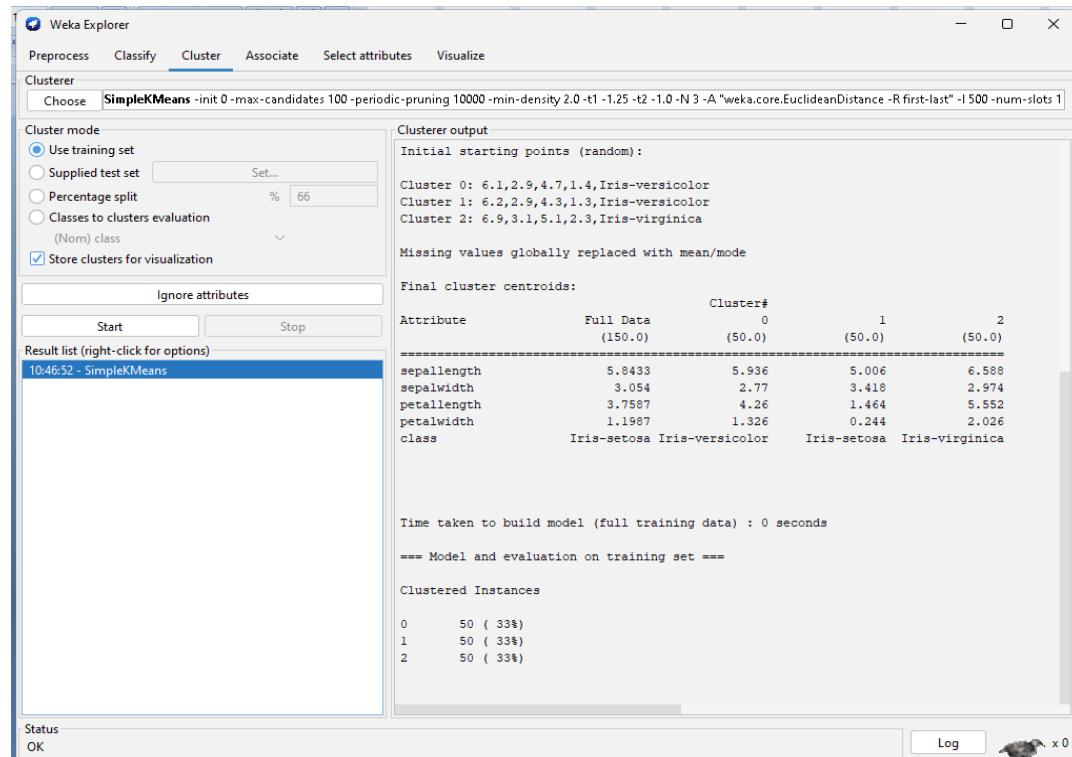


9. Now, click on “SimpleKMeans” and change the numClusters value to “3” to get 3 clusters

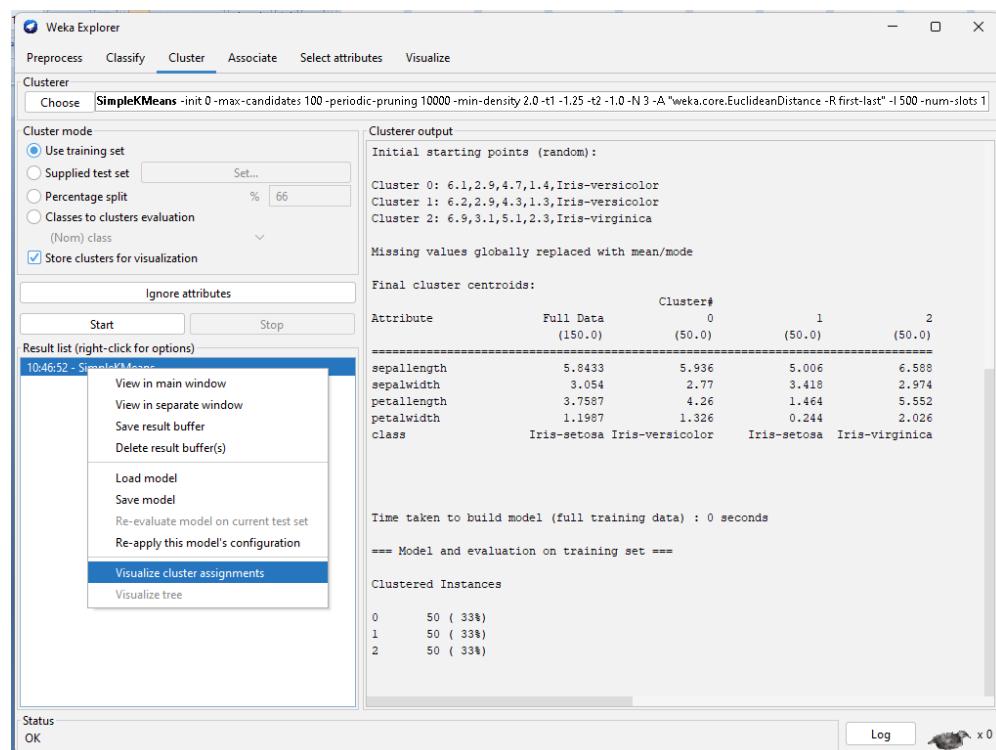


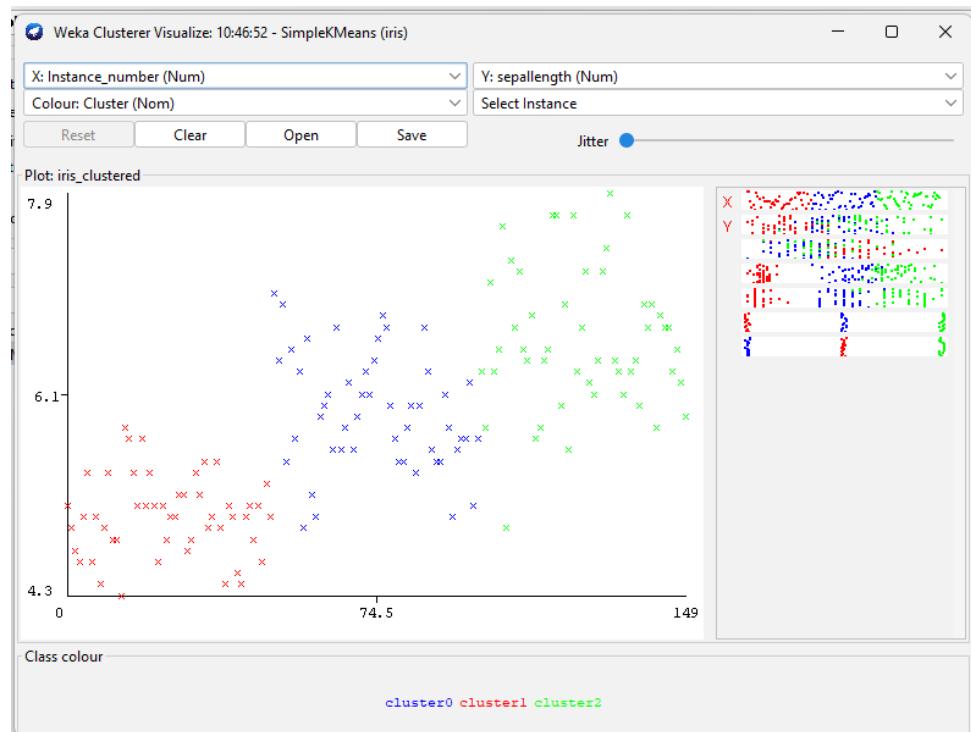
10. Click on “Start” to get the Cluster output



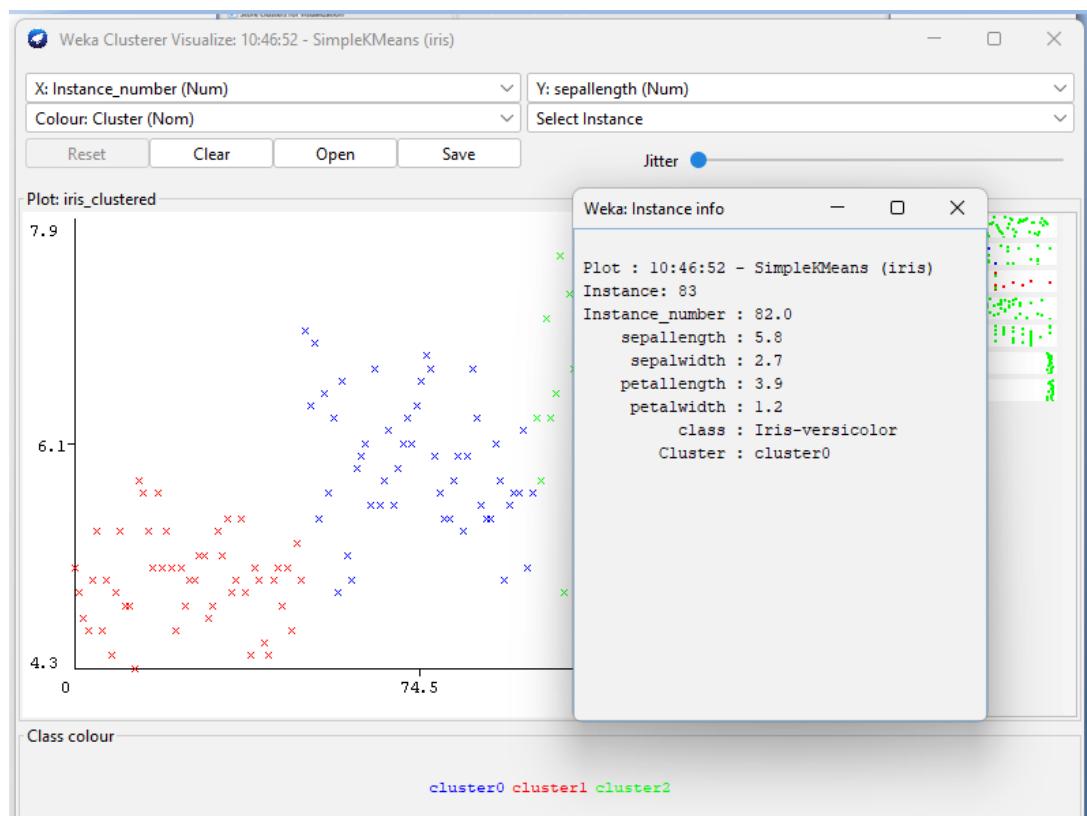


11. Now, right click on SimpleKMeans which is on Result list and then select the “Visualize cluster assignments”





- Click on any instance on the graph to get the information of that instance



EXPERIMENT – 11

AIM:

Utilize the Visualize option and tinker with the datapoints, available in it.

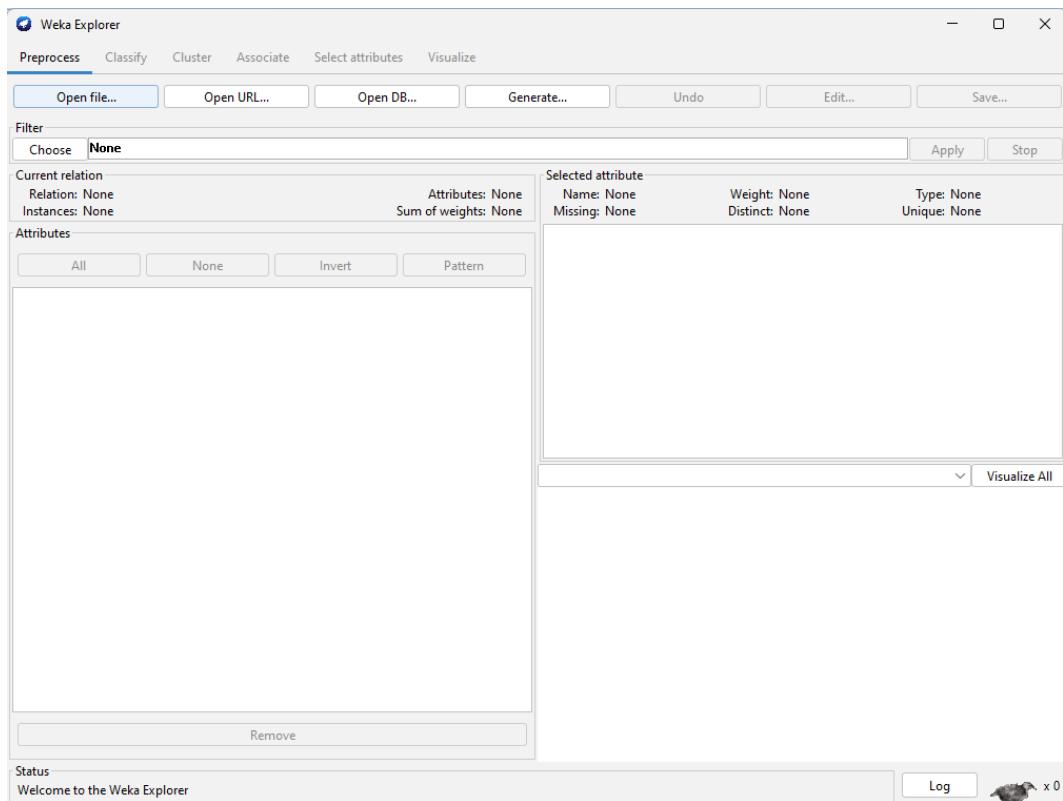
Description:

The visualization option in Weka tool allows users to create visual representations of their data through various charts and graphs. These visualizations can help users better understand their data, identify patterns, trends, and outliers. The tool provides a range of visualization techniques such as scatter plots, line charts, histogram, bar charts, and more. Users can customize the format, color, size, and other options of the visualizations to suit their needs.

Procedure:

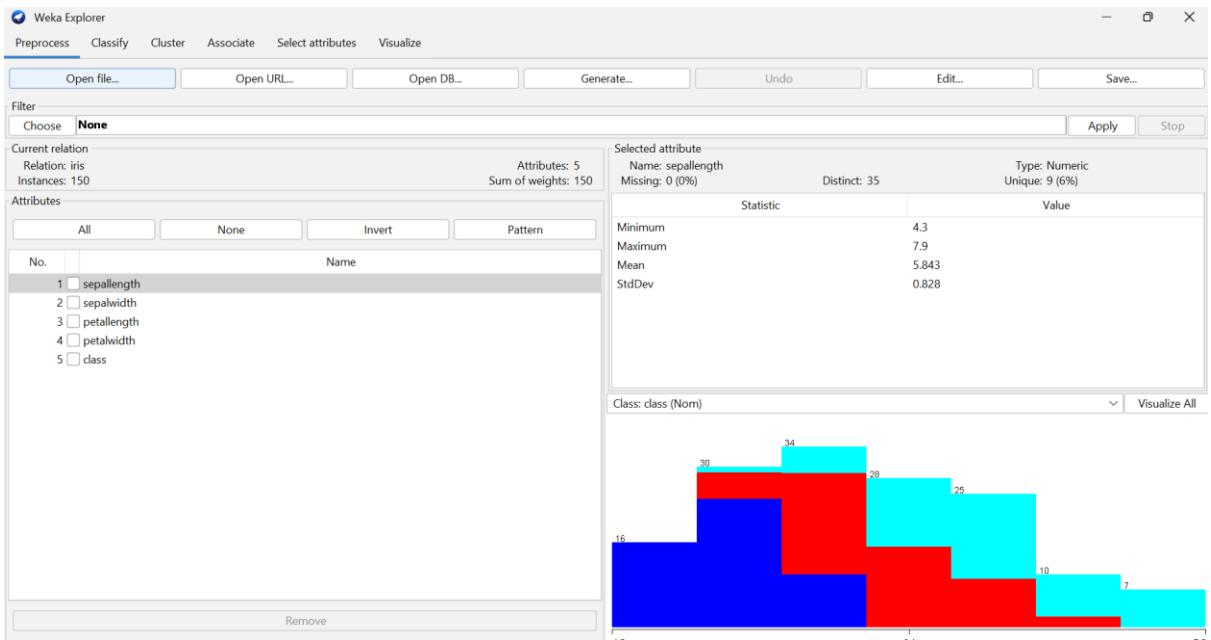
- ❖ open weka tool and select the explorer



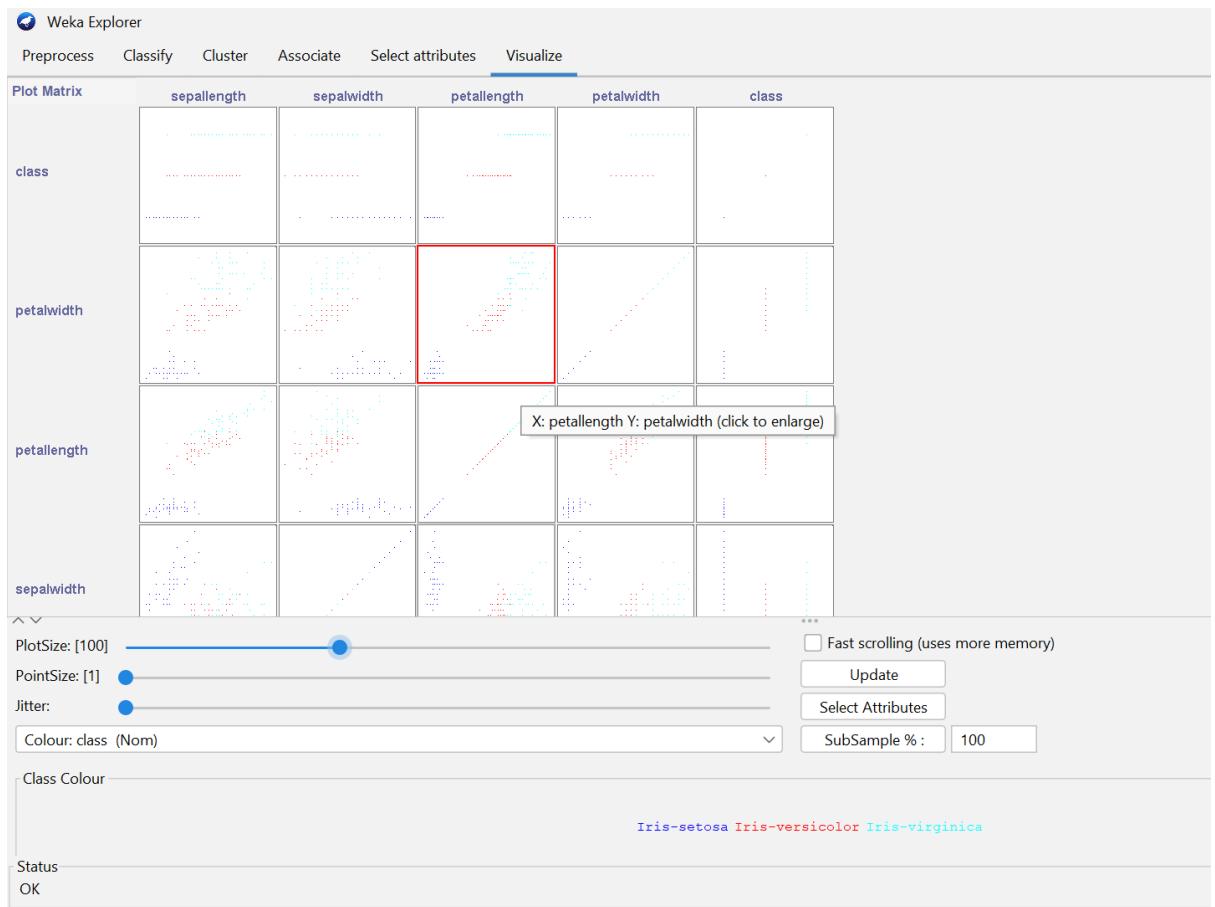


- ❖ load the iris dataset by selecting the “open file” option.

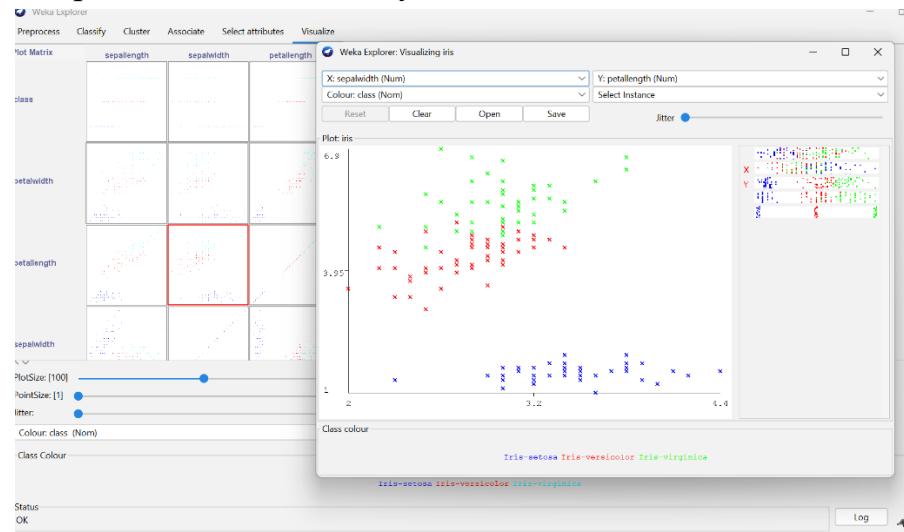
→upon successful opening



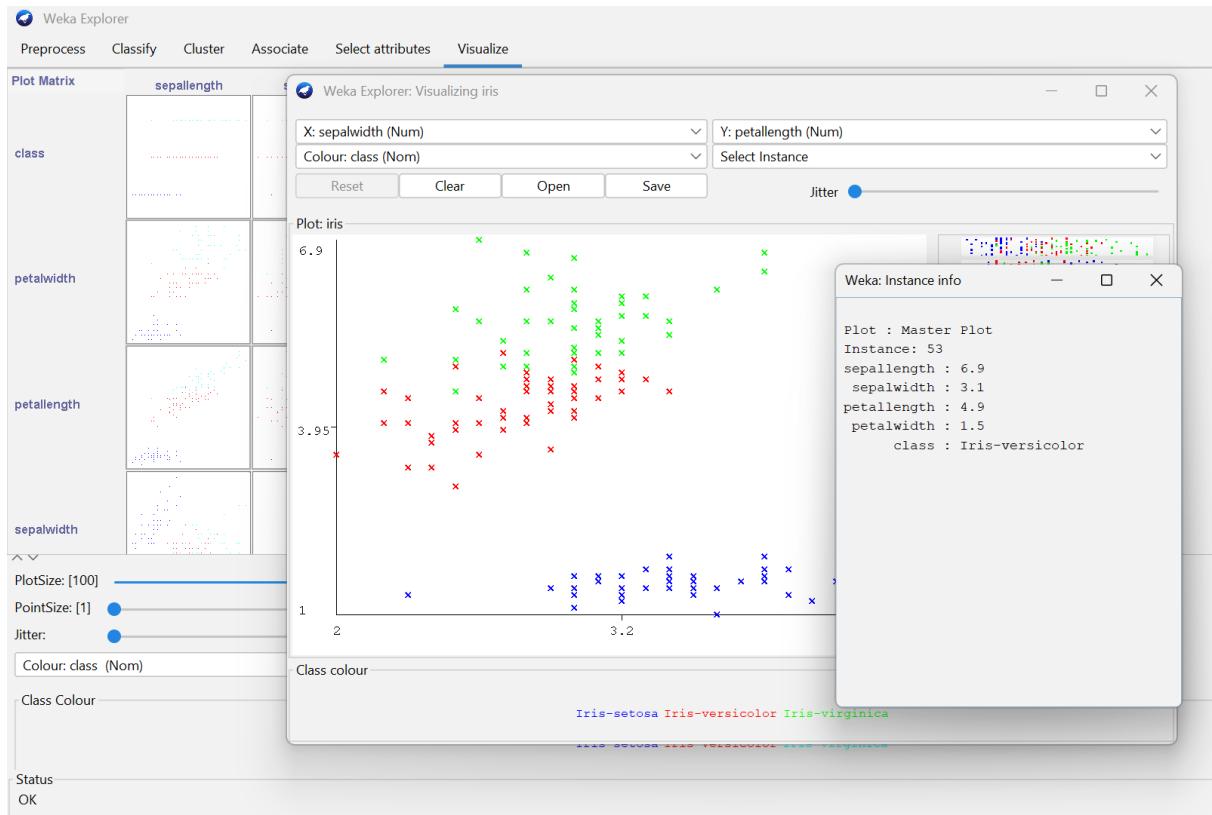
- ❖ Select visualize option from the toolbar.



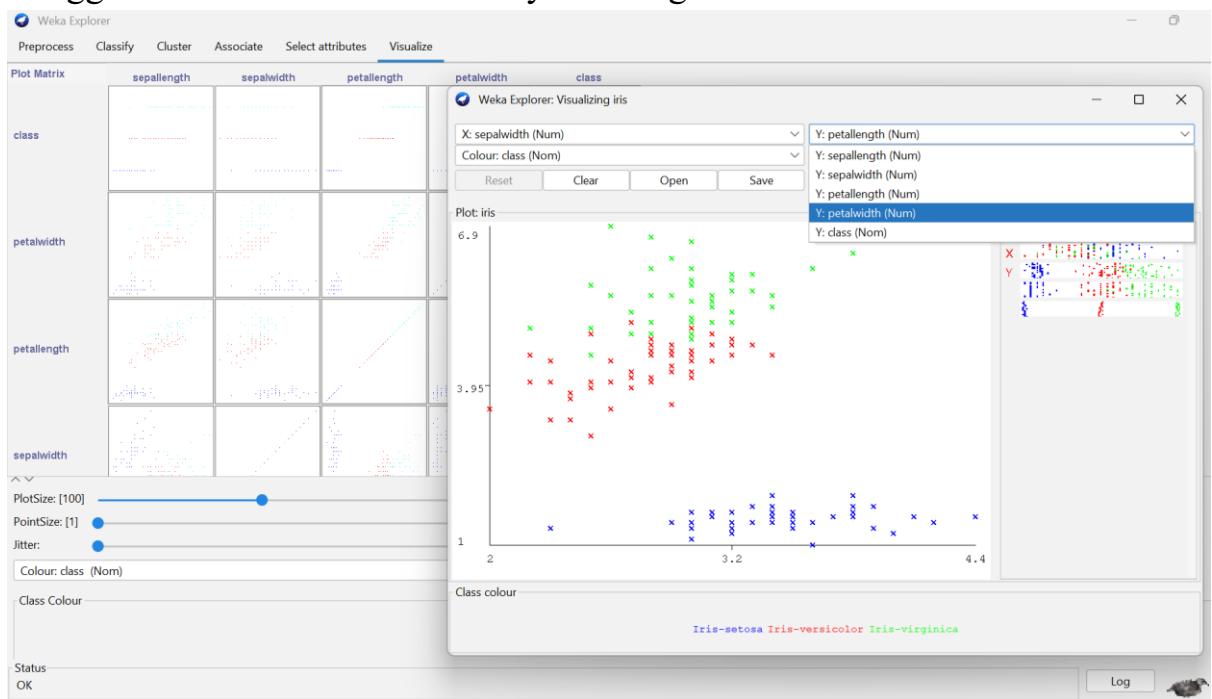
❖ open a selected box of your choice to enhance the view.



❖ select a single data point to dvelve into the details of it.



- ❖ Toggle the X or Y Co-ordinates by selecting the different features.



Experiment – 12

Aim: Apply Clustering technique on any dataset using Knowledge Flow.

Description: Cluster analysis, also known as clustering, is a method of data mining that groups similar data points together. The goal of cluster analysis is to divide a dataset into groups (or clusters) such that the data points within each group are more similar to each other than to data points in other groups. This process is often used for exploratory data analysis and can help identify patterns or relationships within the data that may not be immediately obvious. There are many different algorithms used for cluster analysis, such as k-means, hierarchical clustering, and density-based clustering. The choice of algorithm will depend on the specific requirements of the analysis and the nature of the data being analysed.

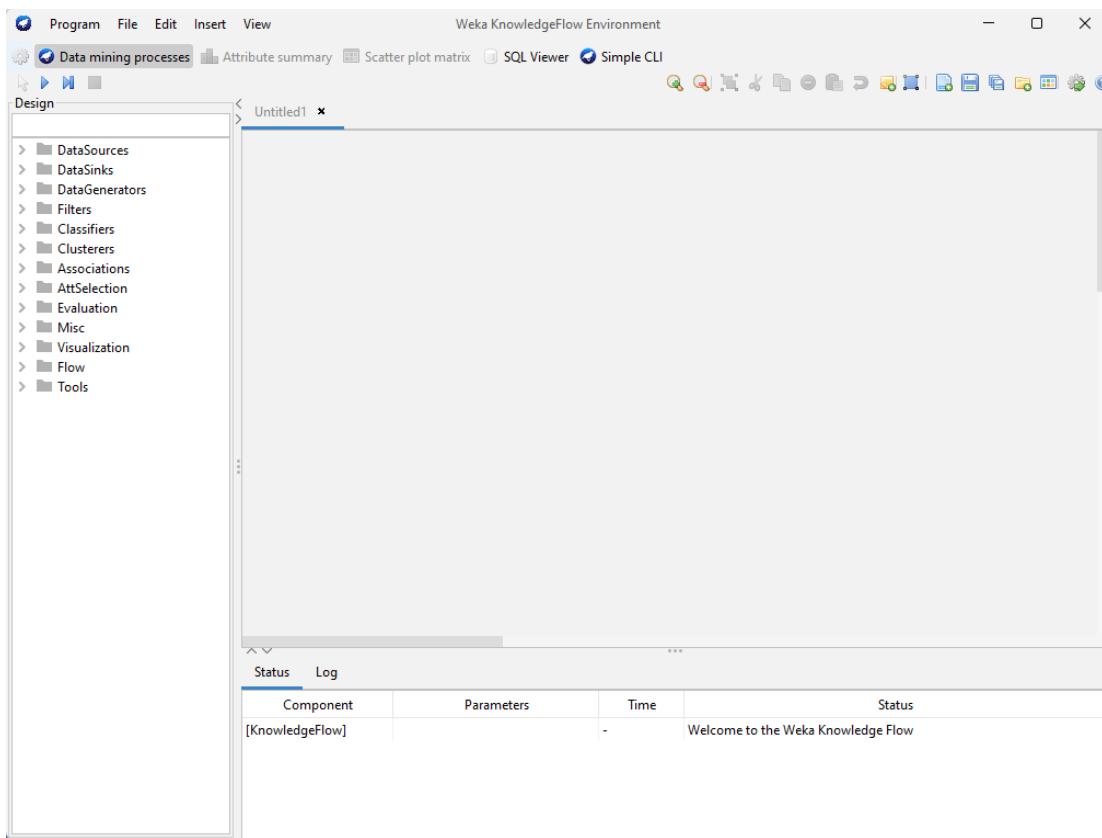
Step by Step Process:

→ To Add Attribute

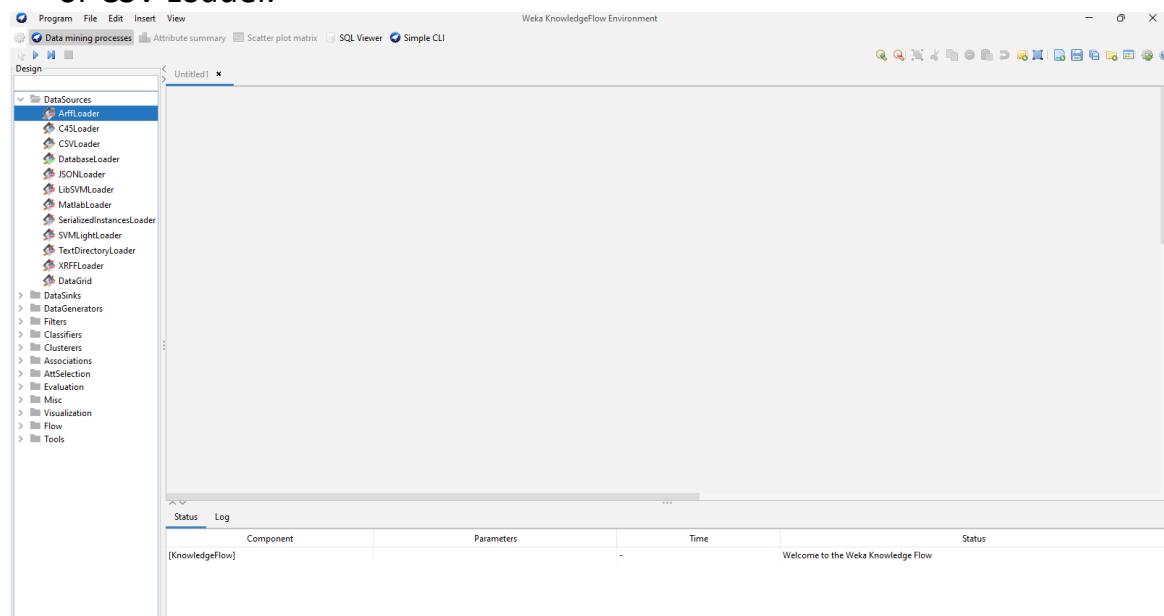
1. Open WEKA Tool and navigate to Knowledge Flow.



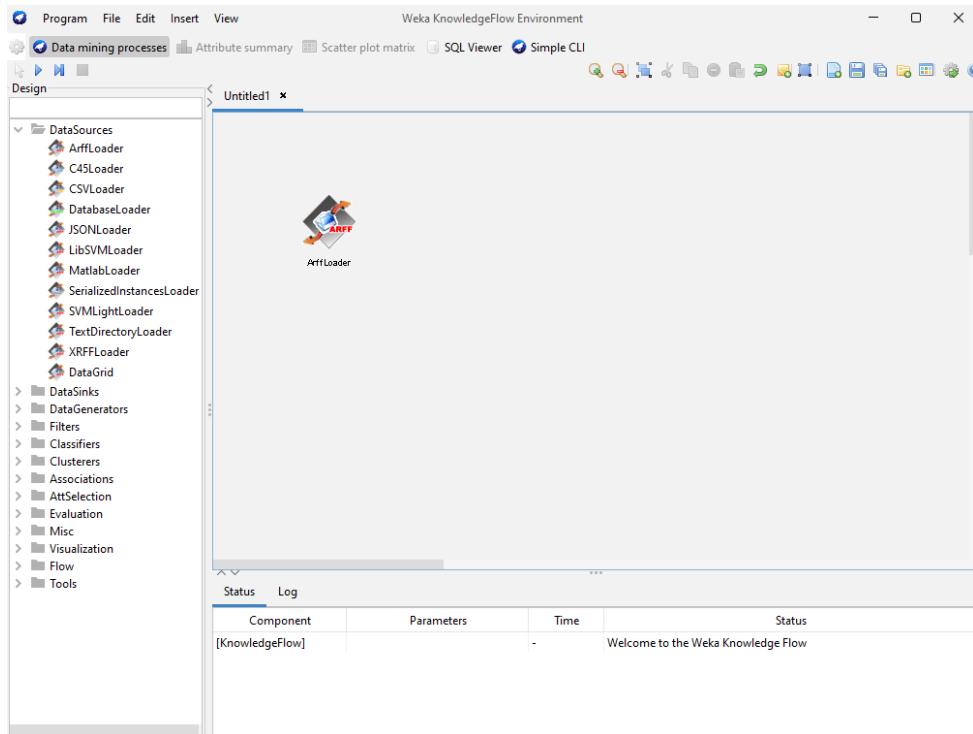
2. Knowledge Flow Environment Views



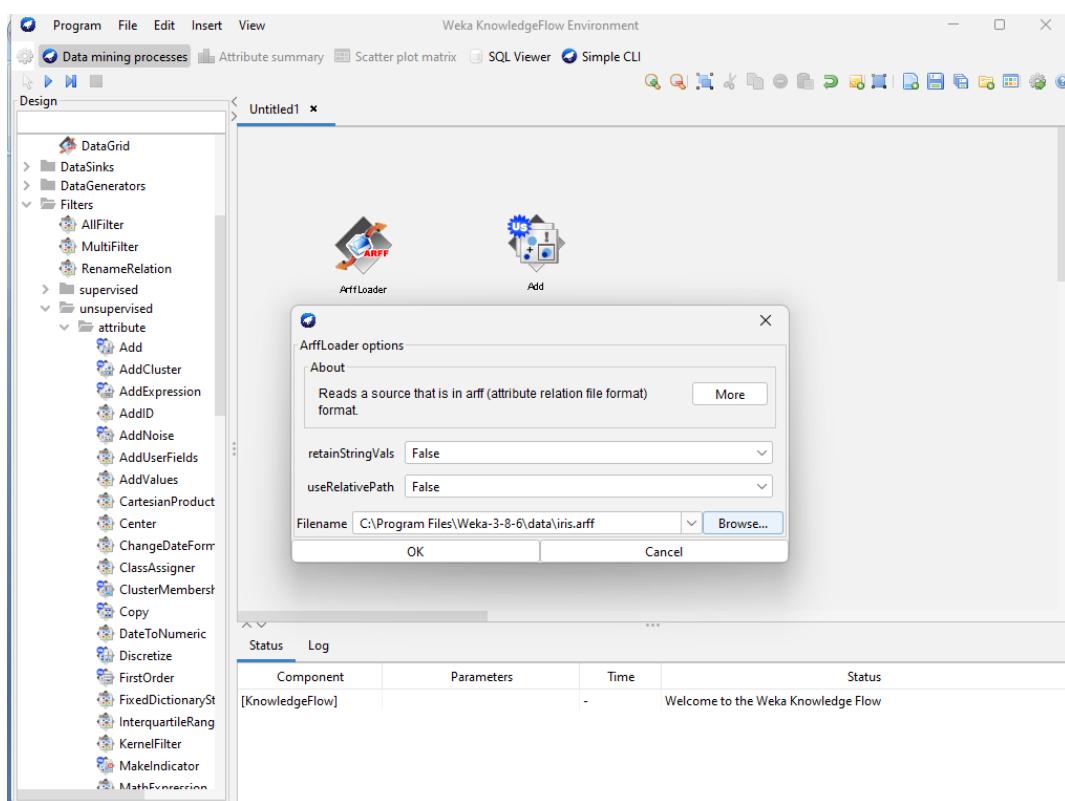
3. To create a model's Knowledge Flow, pick **DataSources** in the Design window. Expand data sources. To pick the dataset, use either ArffLoader or CSV Loader.



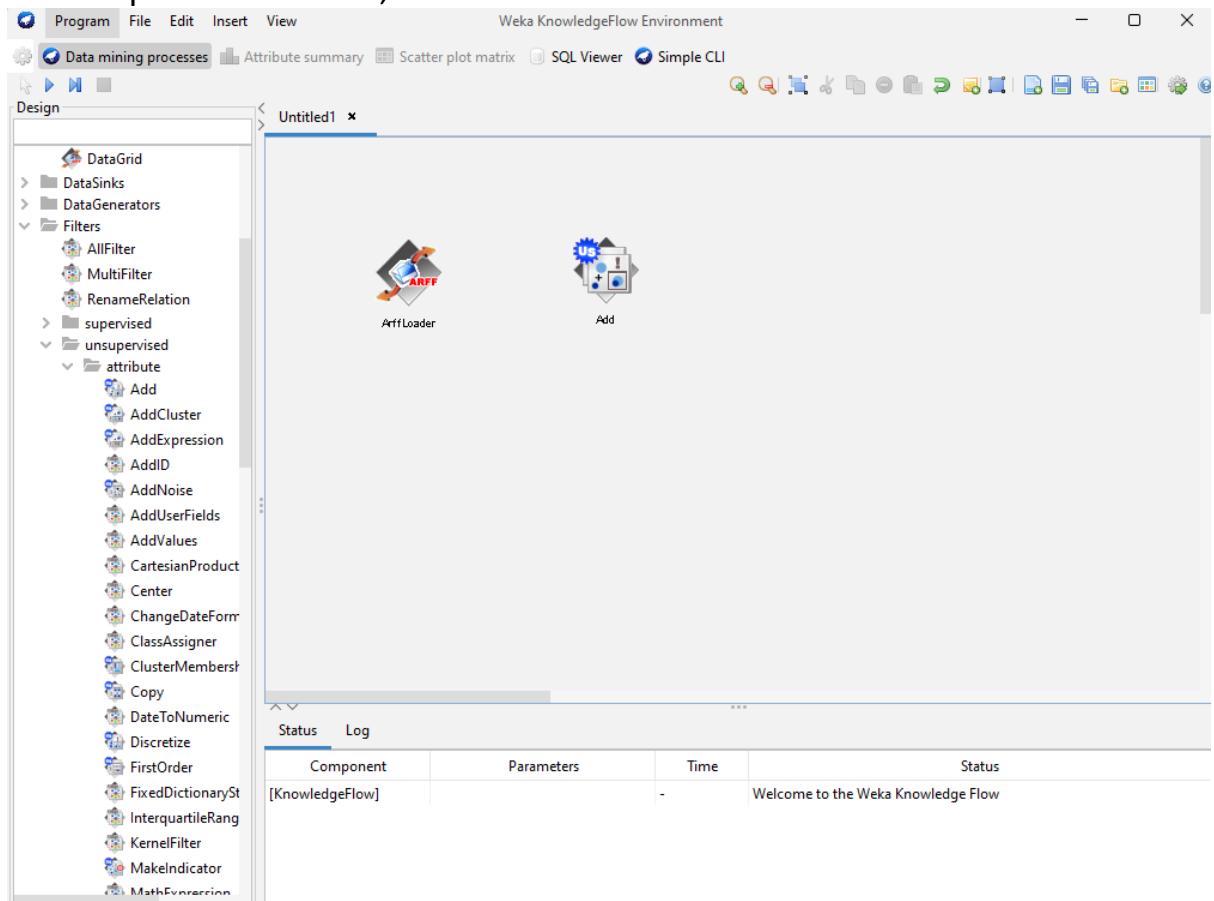
4. Select the Loader and set it in the Knowledge Flow by clicking on the canvas window.



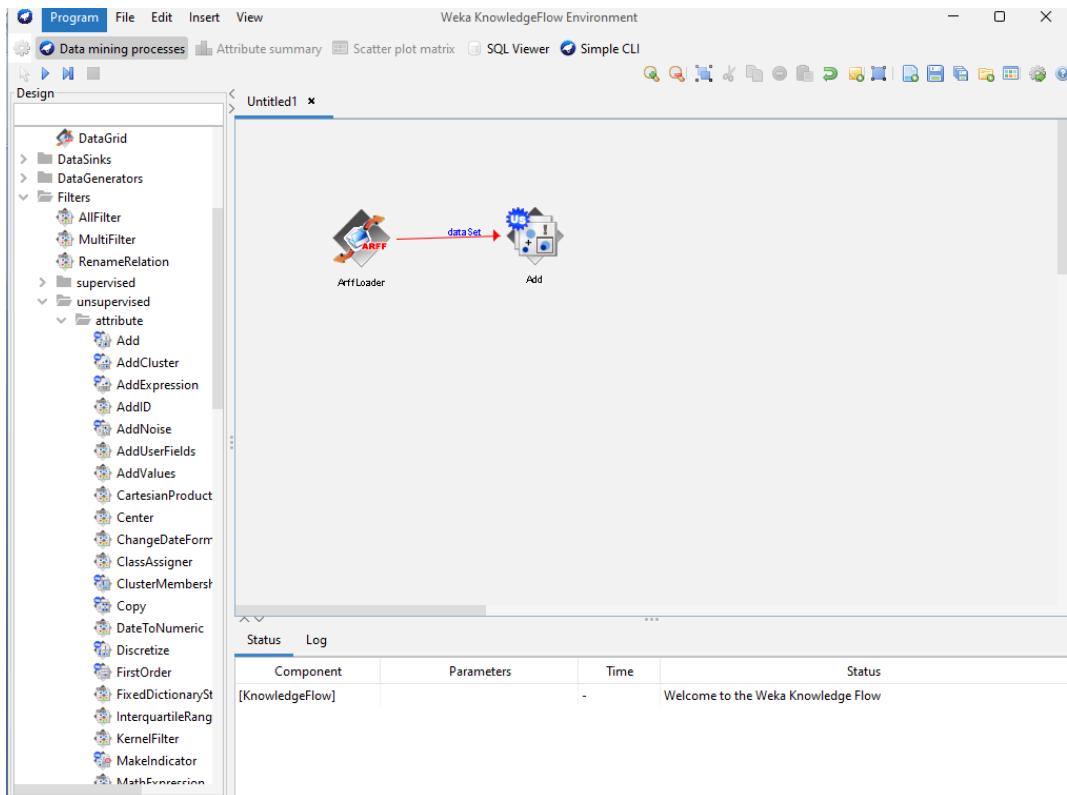
5. To load the dataset in ArffLoader, right-click on it, select Configure, and then pick the dataset.



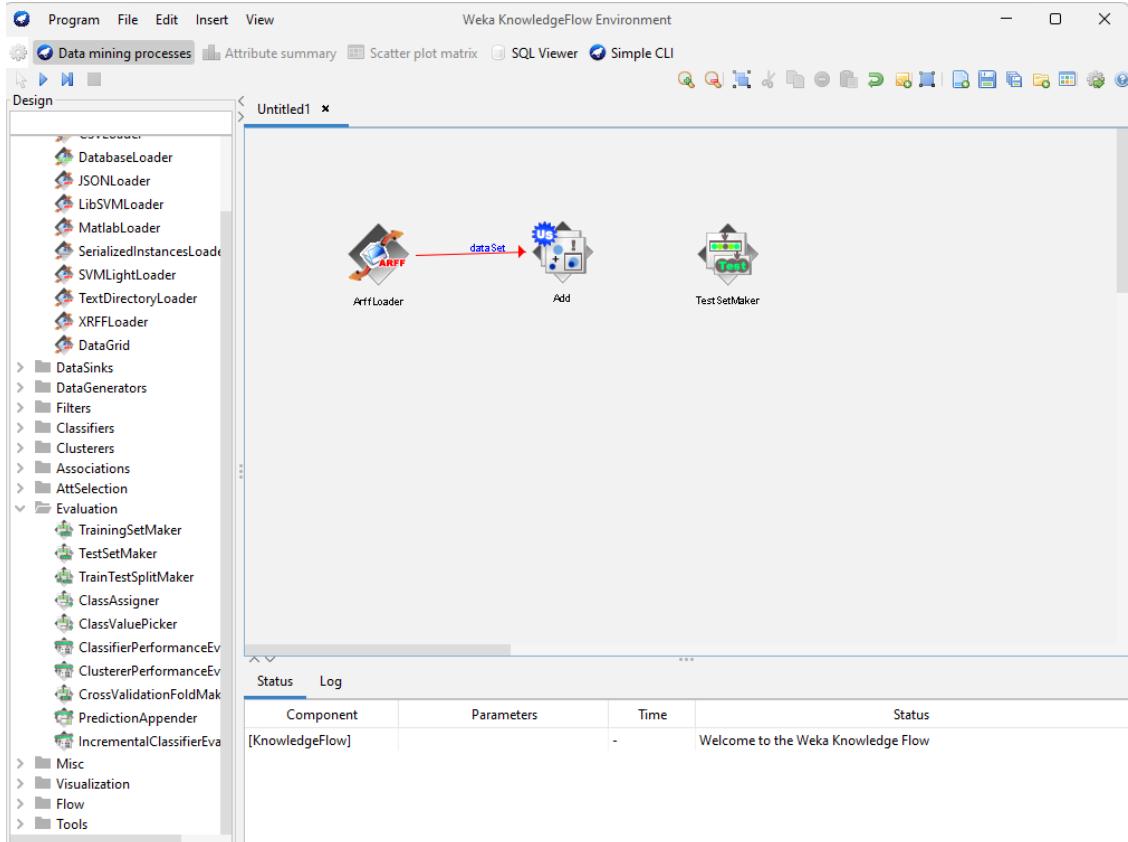
6. To add an attribute, navigate to Filters > Unsupervised > Attributes > Add. To position the Add, select it and then click on the canvas window.



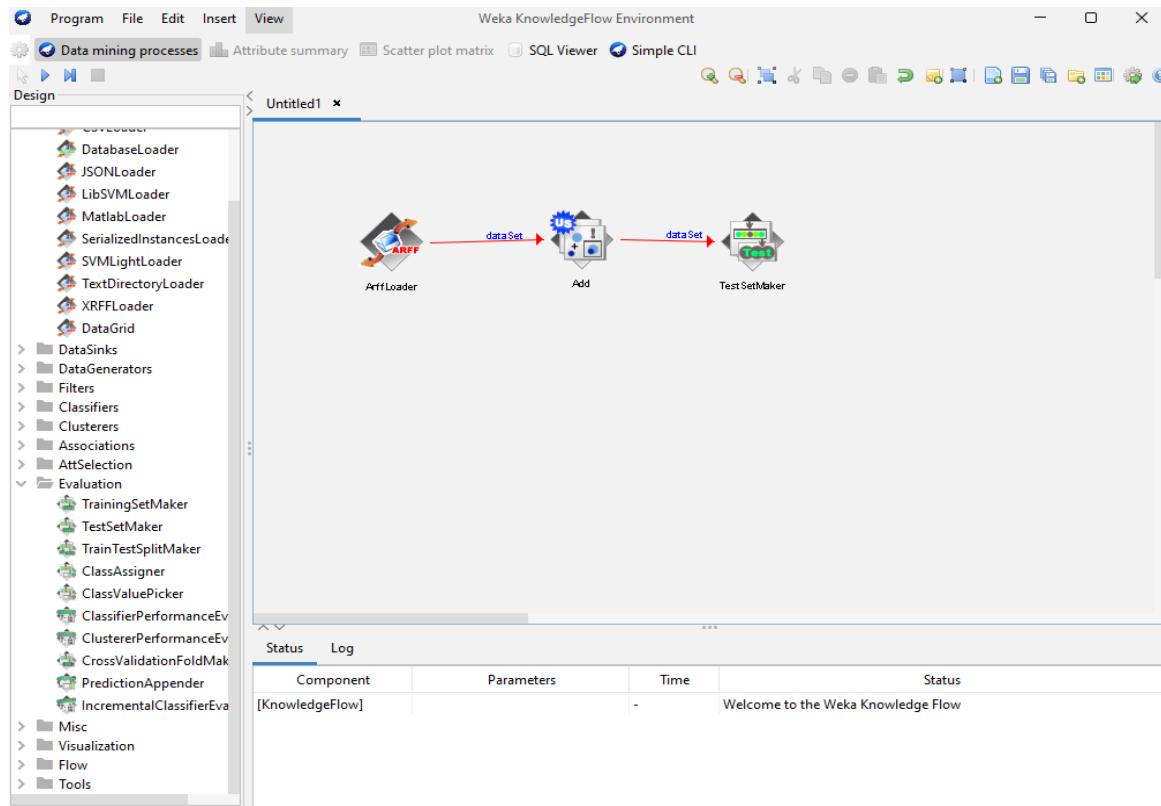
7. To link ArffLoader and Add, right-click on Arffloader>dataSet and choose Add.



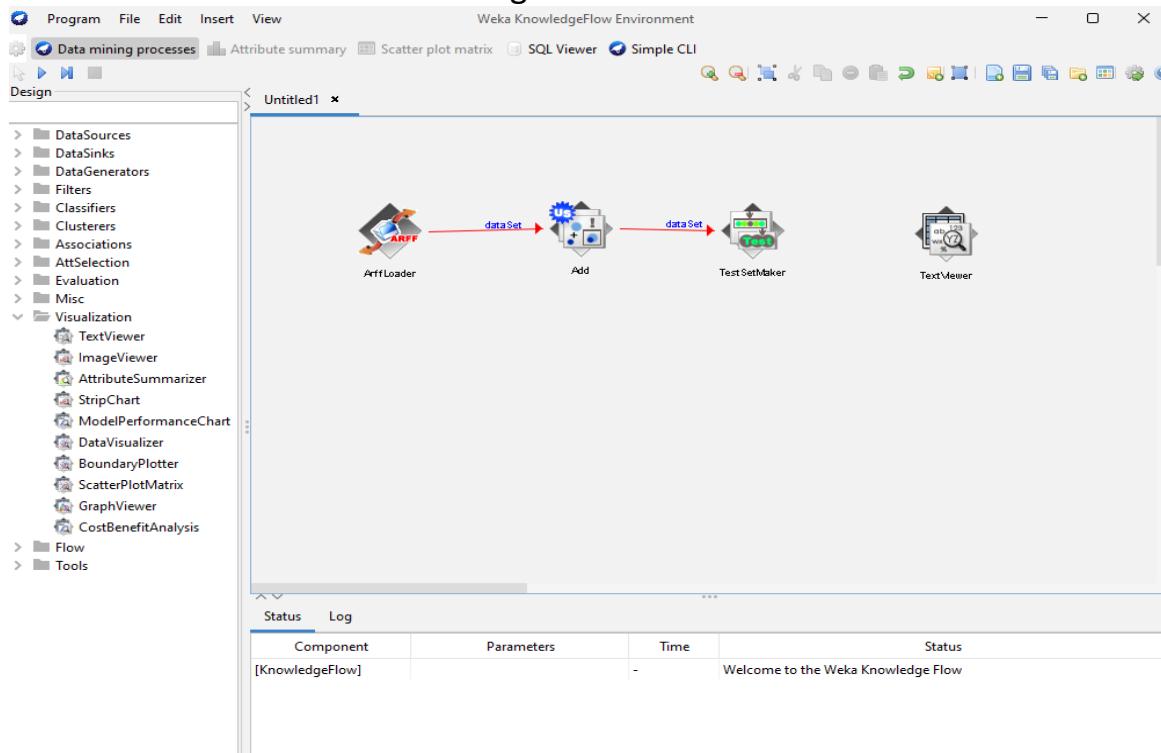
- To evaluate the DataSet, choose Evaluation > TestSetMaker. To position it, simply click on the canvas window.



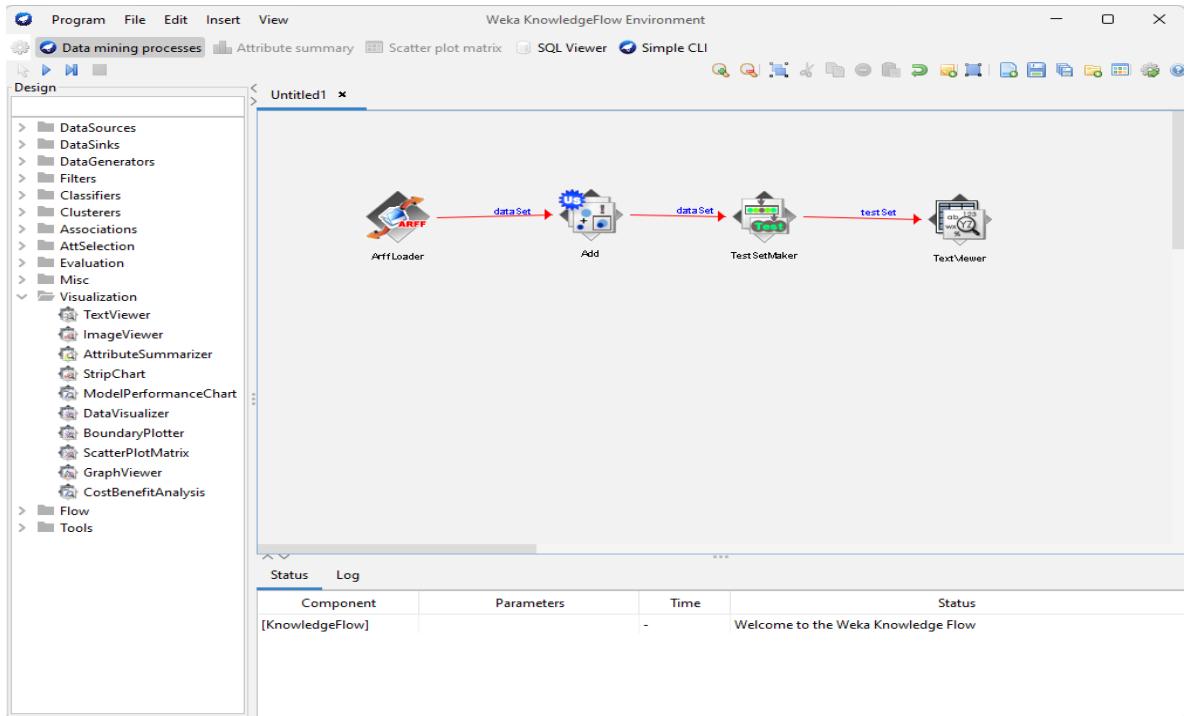
9. To link Add and TestSetMaker, right-click on Add>dataSet and then choose TestSetMaker.



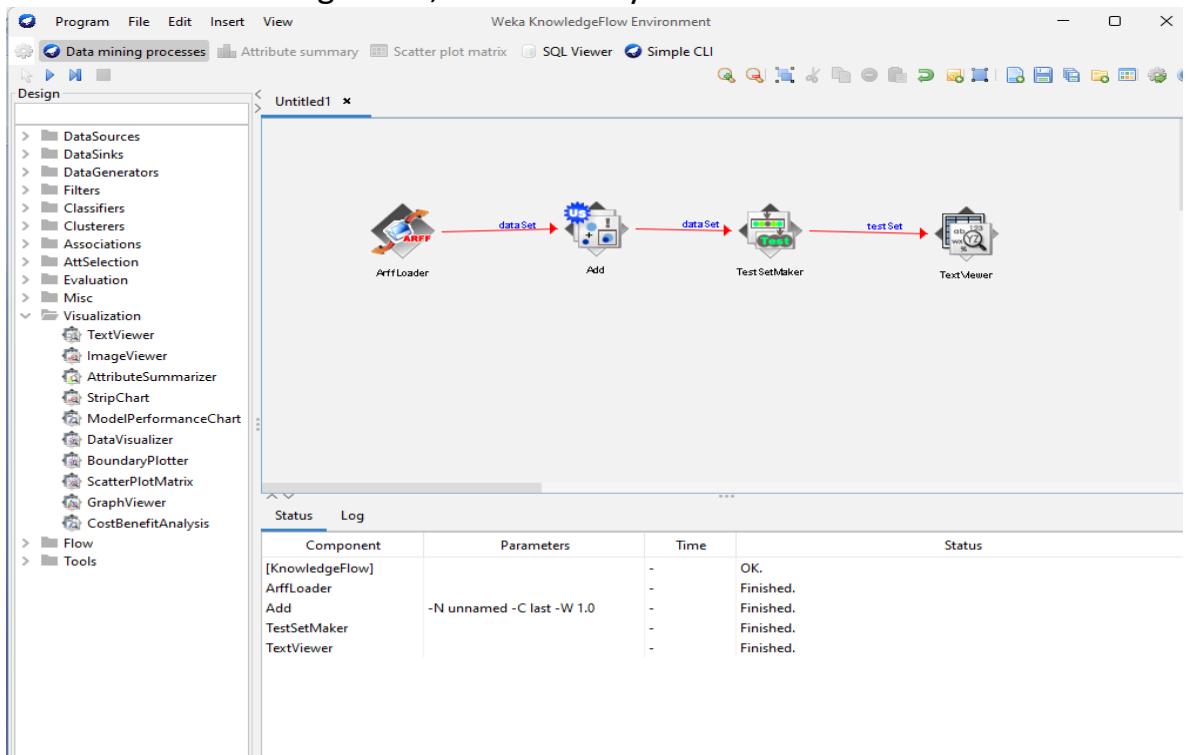
10. To view the findings, we must utilize Visualization. Choose Visualization>TextViewer and drag it onto the canvas window.



11. To create a link between TestSetMaker and TextViewer, right-click on TestSetMaker, then choose TestSet and drop it on TextViewer.



12. To start the Knowledge Flow, click the Play button in the Menu bar.



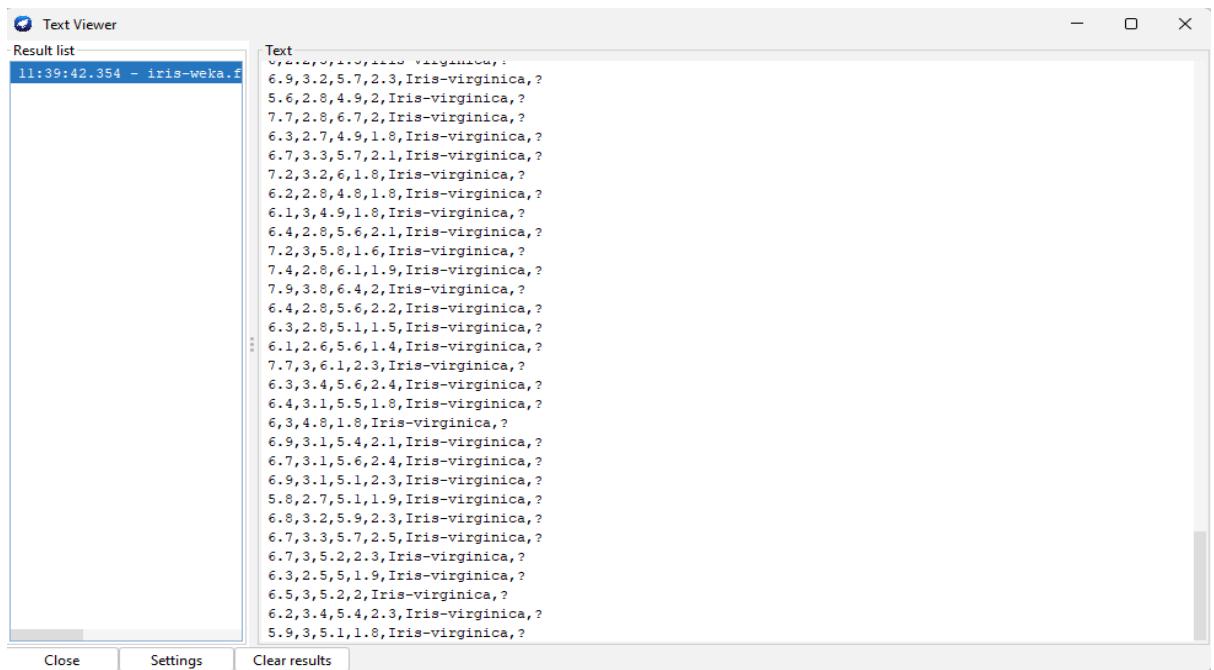
We can see the execution status in Status Window.

13. To view the results, right-click on TextViewer and select Show Results.

```
@relation iris-weka.filters.unsupervised.attribute.Add-Unnamed-Clas-W1.0

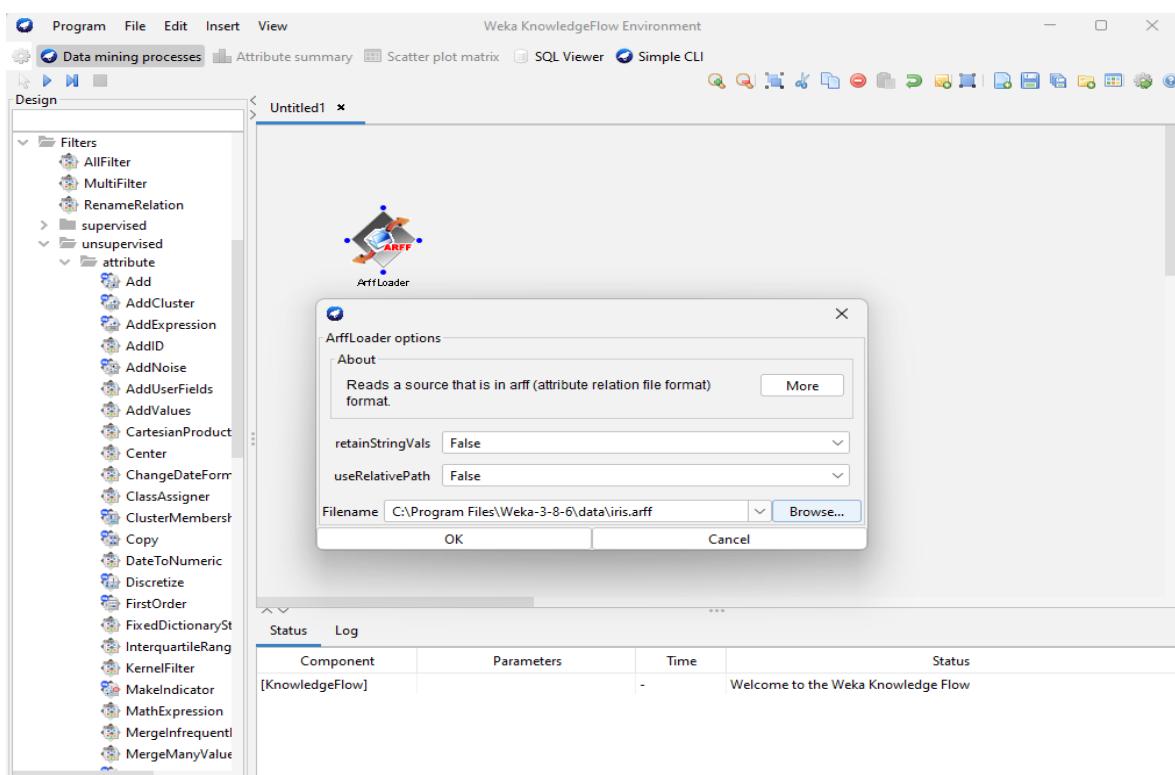
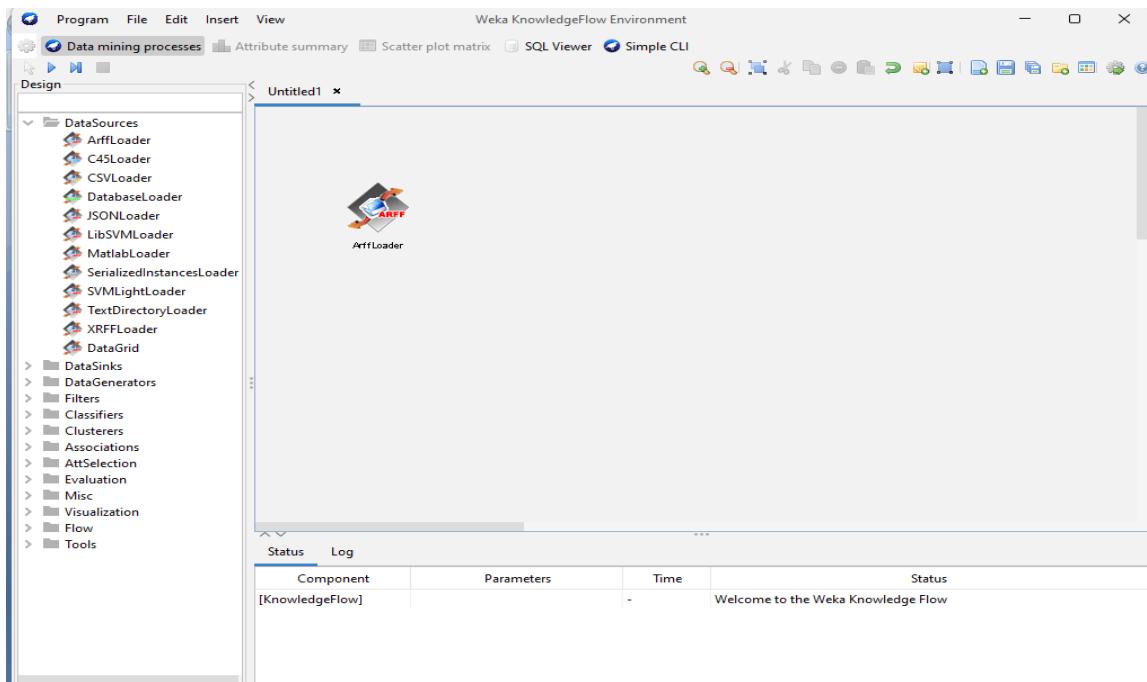
@attribute sepallength numeric
@attribute sepalwidth numeric
@attribute petallength numeric
@attribute petalwidth numeric
@attribute class {Iris-setosa,Iris-versicolor,Iris-virginica}
@attribute unnamed numeric

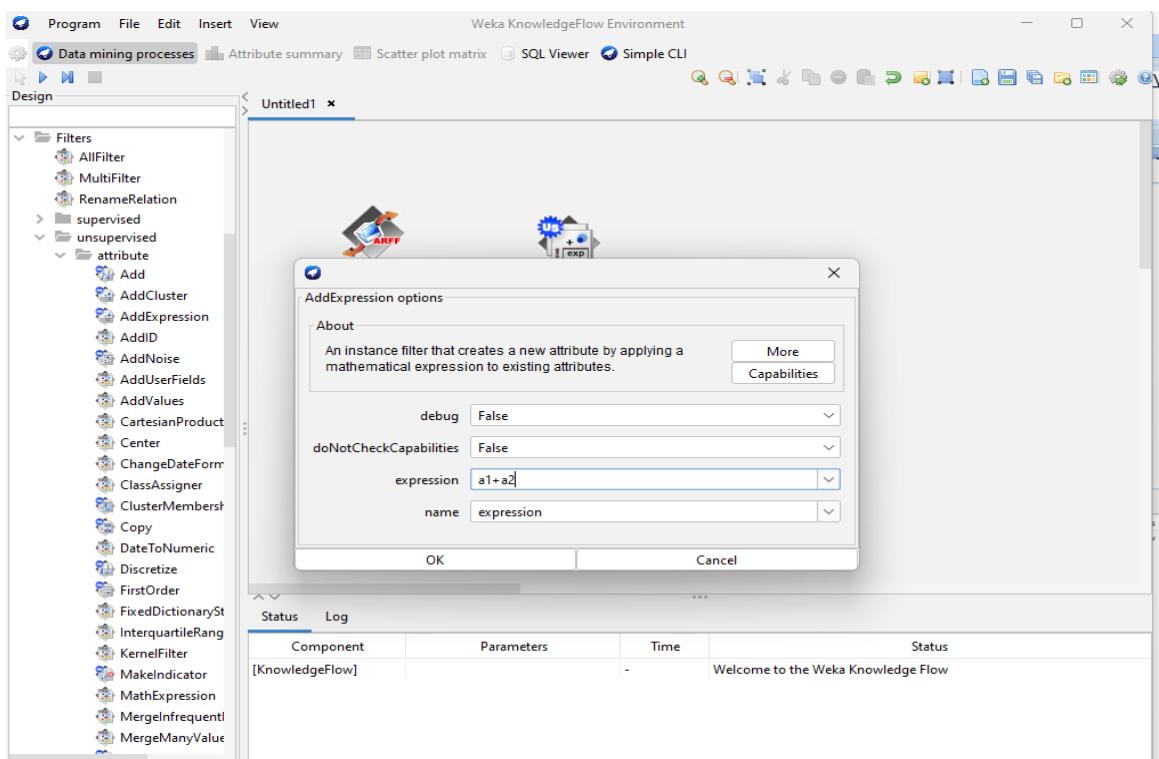
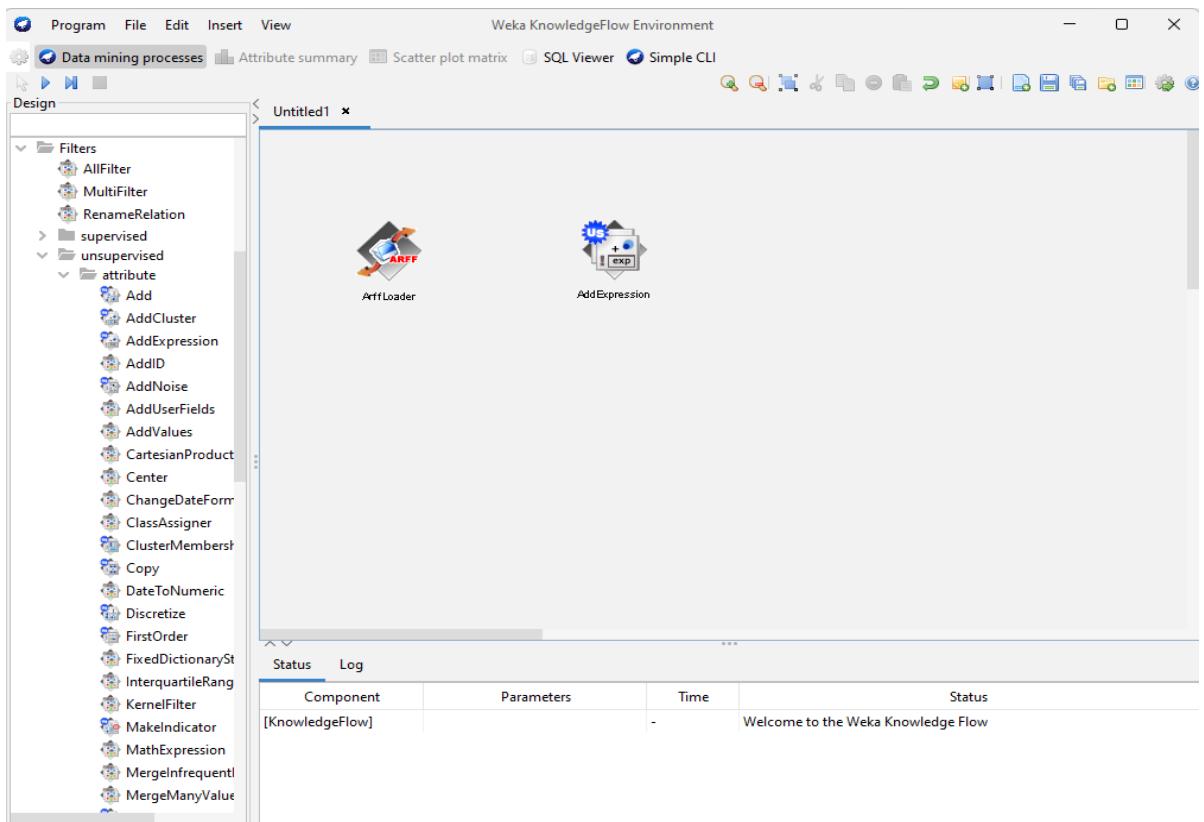
@data
5.1,3.5,1.4,0.2,Iris-setosa,?
4.9,3.1,1.4,0.2,Iris-setosa,?
4.7,3.2,1.3,0.2,Iris-setosa,?
```

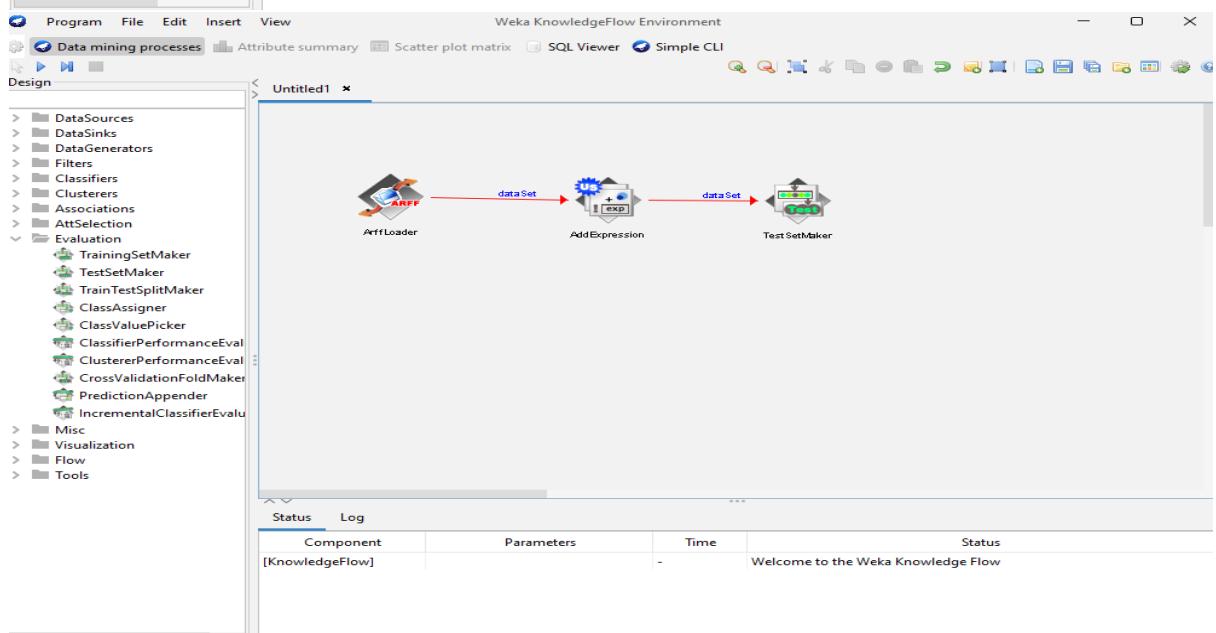
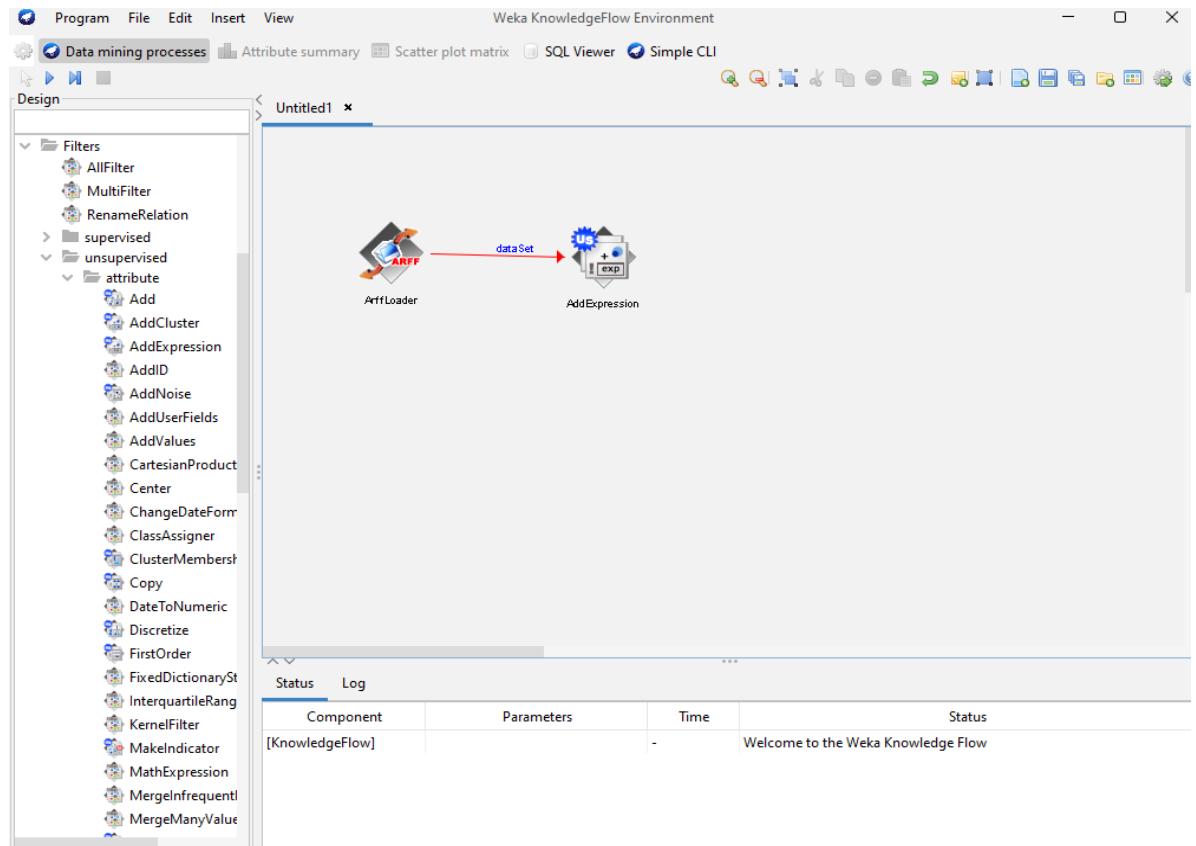


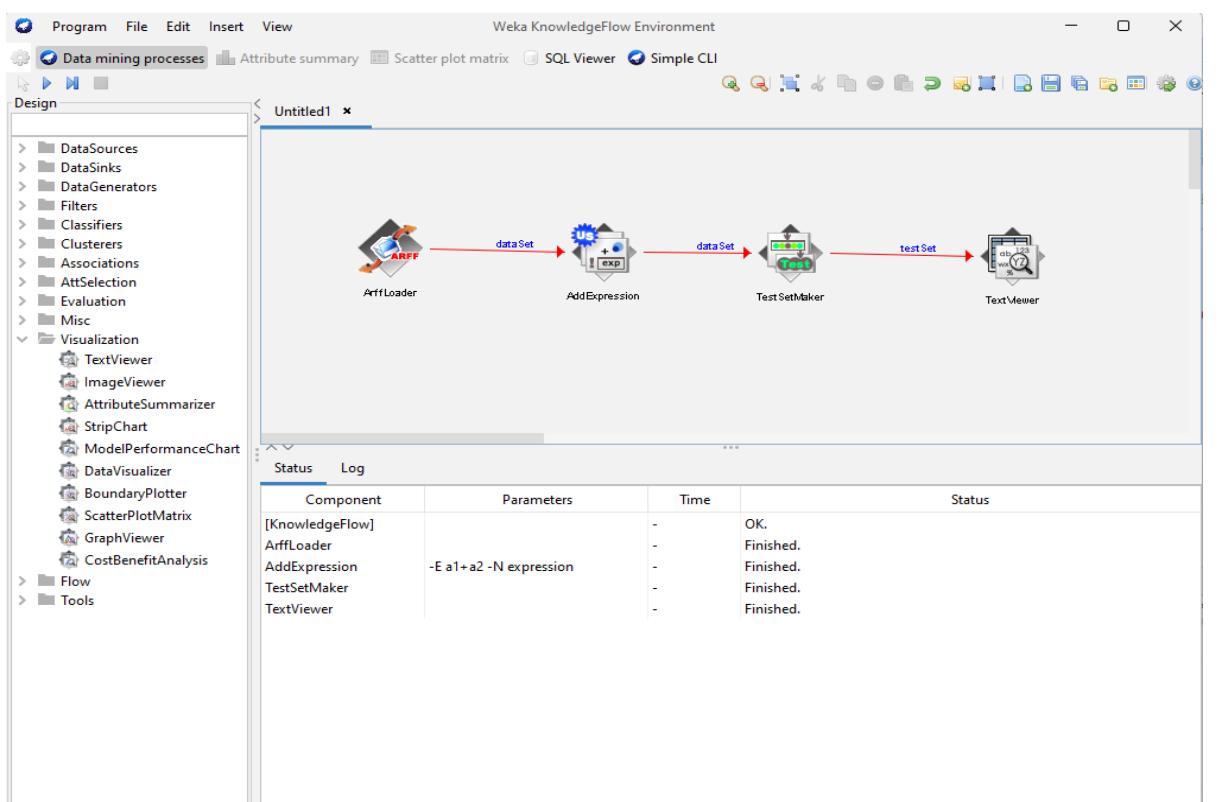
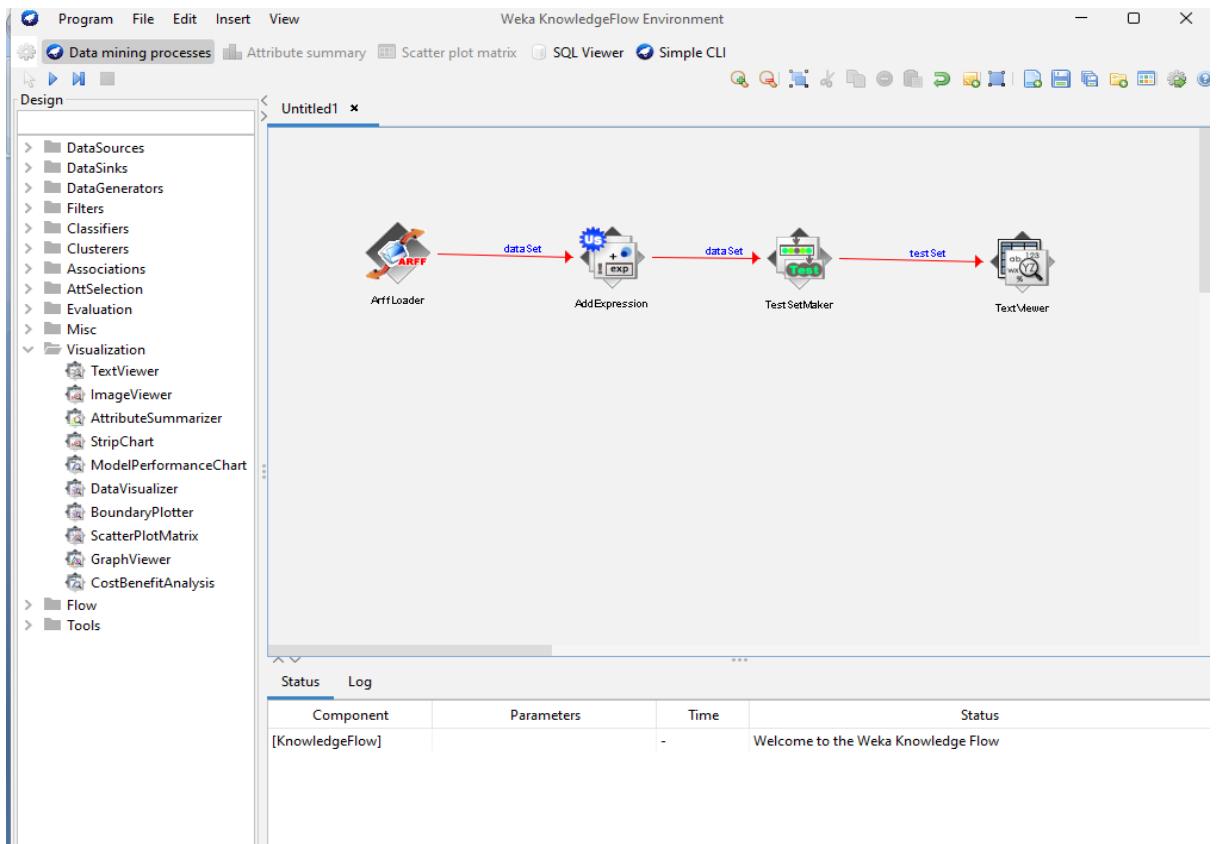
The ? symbol symbolizes the un-inserted values in the newly introduced attribute.

→ *To Add Expression:*









Result :

```

@relation iris-weka.filters.unsupervised.attribute.AddExpression-E1+a2-Nexpression

@attribute sepallength numeric
@attribute sepalwidth numeric
@attribute petallength numeric
@attribute petalwidth numeric
@attribute class {Iris-setosa,Iris-versicolor,Iris-virginica}
@attribute al+a2 numeric

```

Text Viewer

Result list

12:09:54.246 - iris-weka.f

Text

```

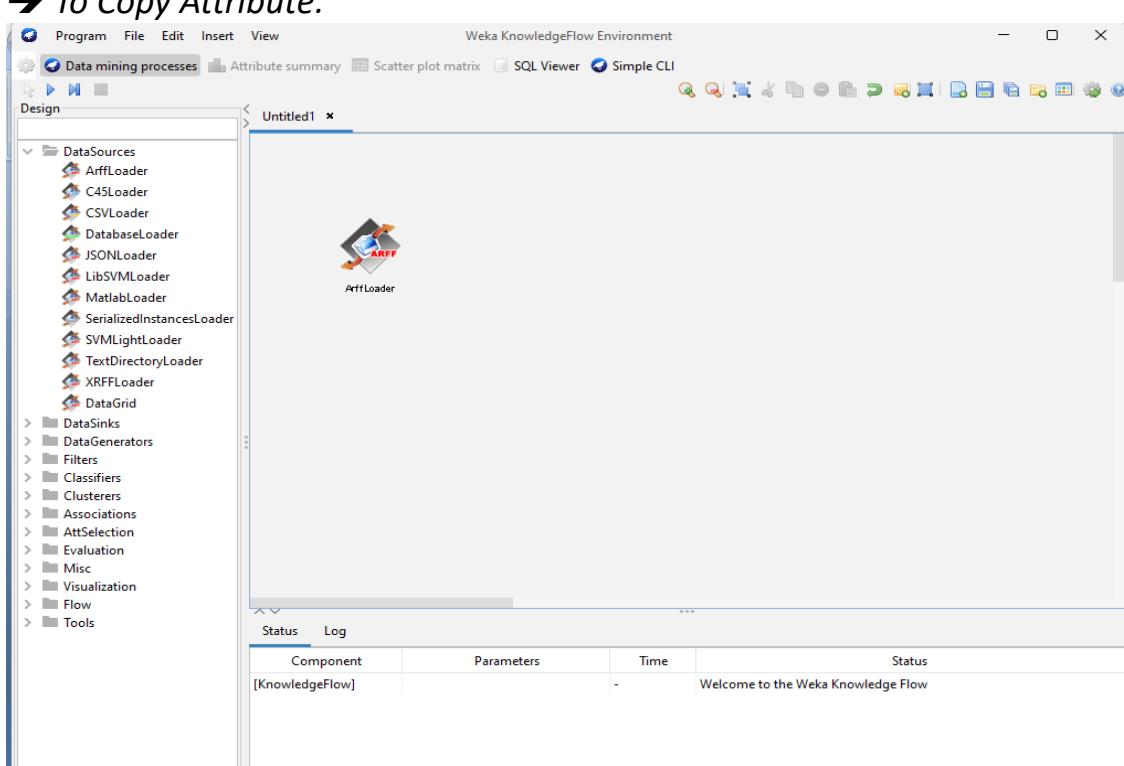
@relation iris-weka.filters.unsupervised.attribute.AddExpression-E1+a2-Nexpression

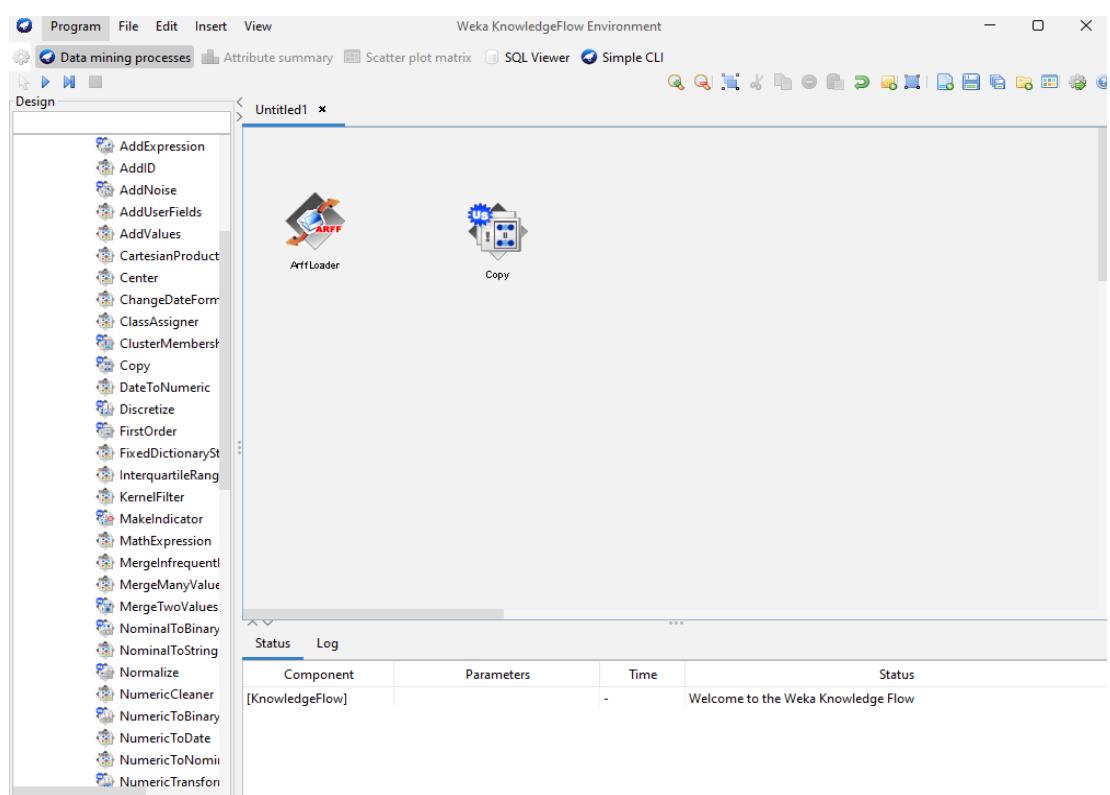
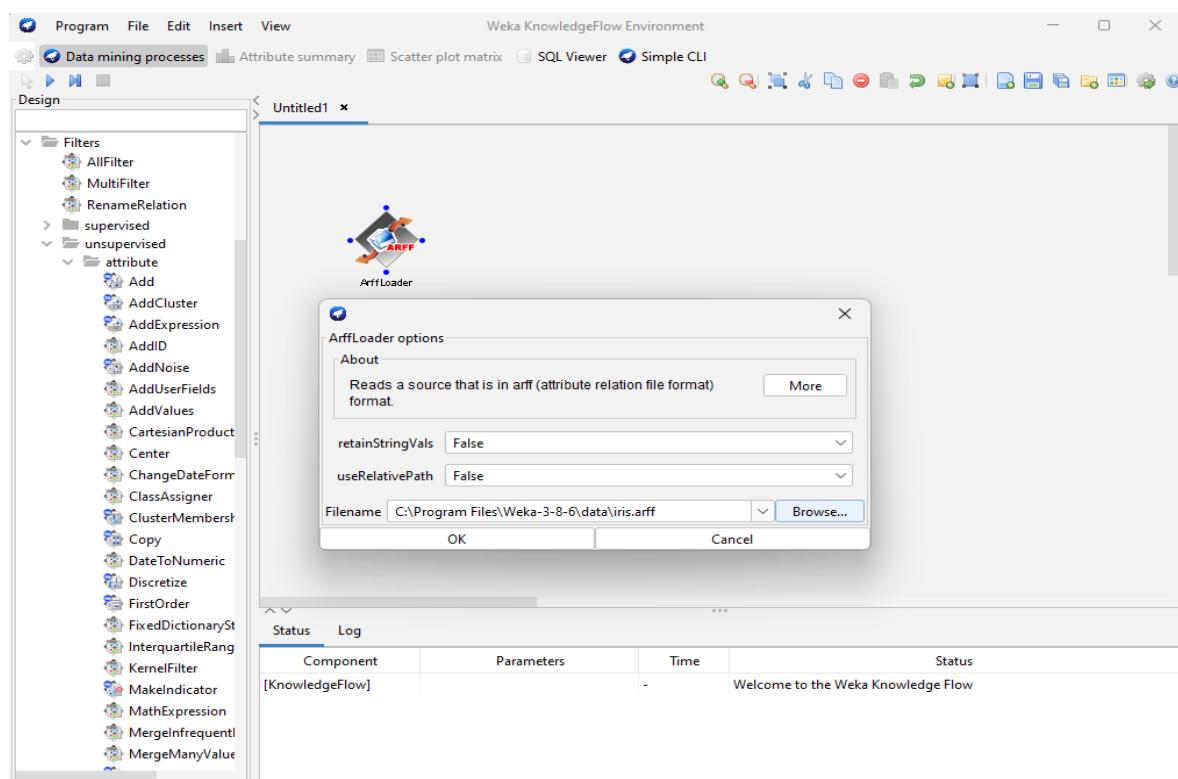
@attribute sepallength numeric
@attribute sepalwidth numeric
@attribute petallength numeric
@attribute petalwidth numeric
@attribute class {Iris-setosa,Iris-versicolor,Iris-virginica}
@attribute al+a2 numeric

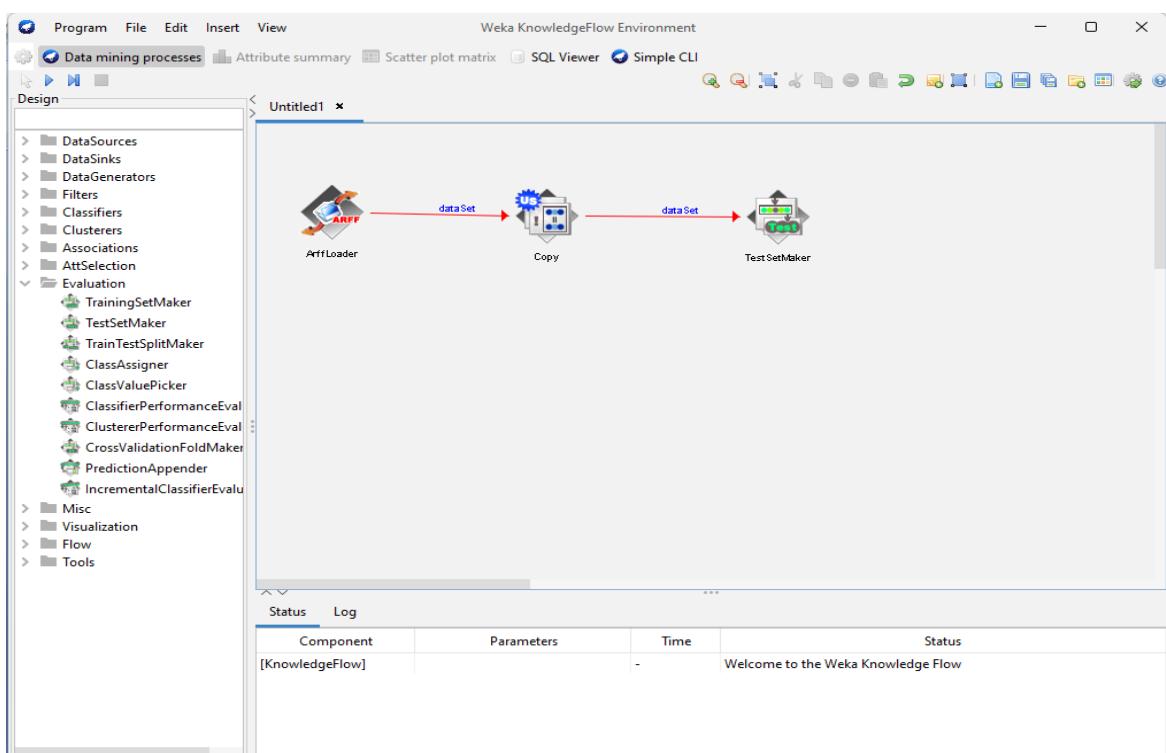
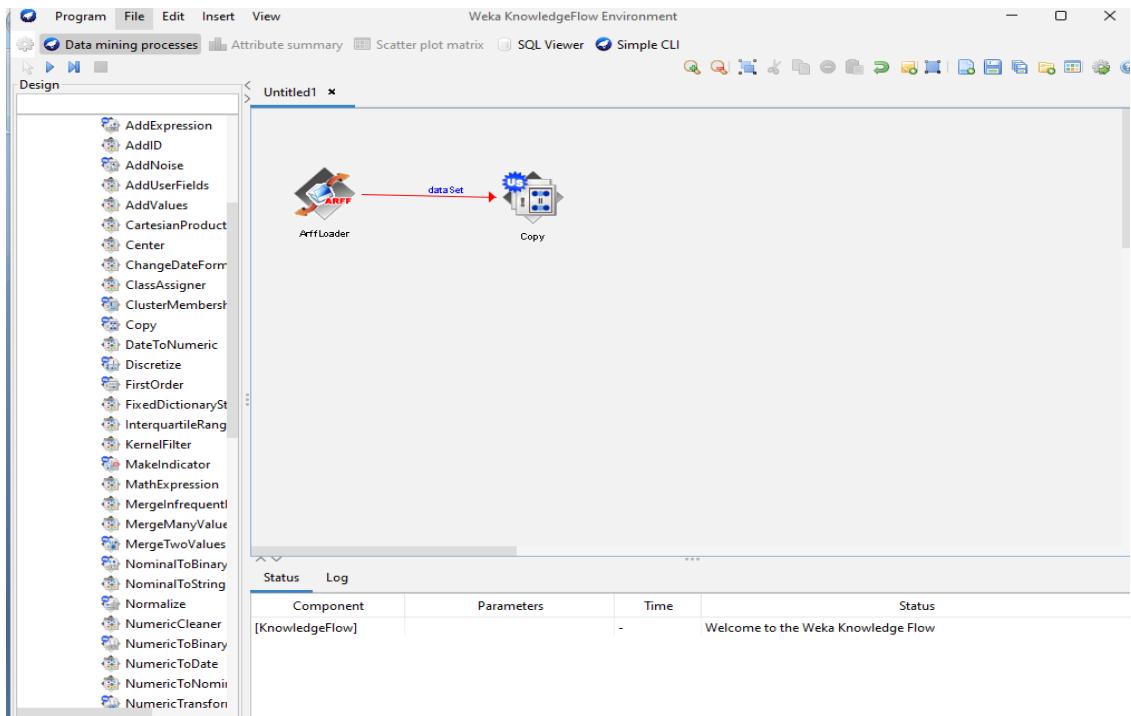
@data
5.1,3.5,1.4,0.2,Iris-setosa,8.6
4.9,3,1.4,0.2,Iris-setosa,7.9
4.7,3.2,1.3,0.2,Iris-setosa,7.9
4.6,3.1,1.5,0.2,Iris-setosa,7.7
5.3,6,1.4,0.2,Iris-setosa,8.6
5.4,3.9,1.7,0.4,Iris-setosa,9.3
4.6,3.4,1.4,0.3,Iris-setosa,8
5.3,4,1.5,0.2,Iris-setosa,8.4
4.4,2.9,1.4,0.2,Iris-setosa,7.3
4.9,3.1,1.5,0.1,Iris-setosa,8
5.4,3.7,1.5,0.2,Iris-setosa,9.1
4.8,3.4,1.6,0.2,Iris-setosa,8.2
4.8,3,1.4,0.1,Iris-setosa,7.8
4.3,3,1.1,0.1,Iris-setosa,7.3
5.8,4,1.2,0.2,Iris-setosa,9.8
5.7,4.4,1.5,0.4,Iris-setosa,10.1
5.4,3.9,1.3,0.4,Iris-setosa,9.3
5.1,3.5,1.4,0.3,Iris-setosa,8.6
5.7,3.8,1.7,0.3,Iris-setosa,9.5
5.1,3.8,1.5,0.3,Iris-setosa,8.9

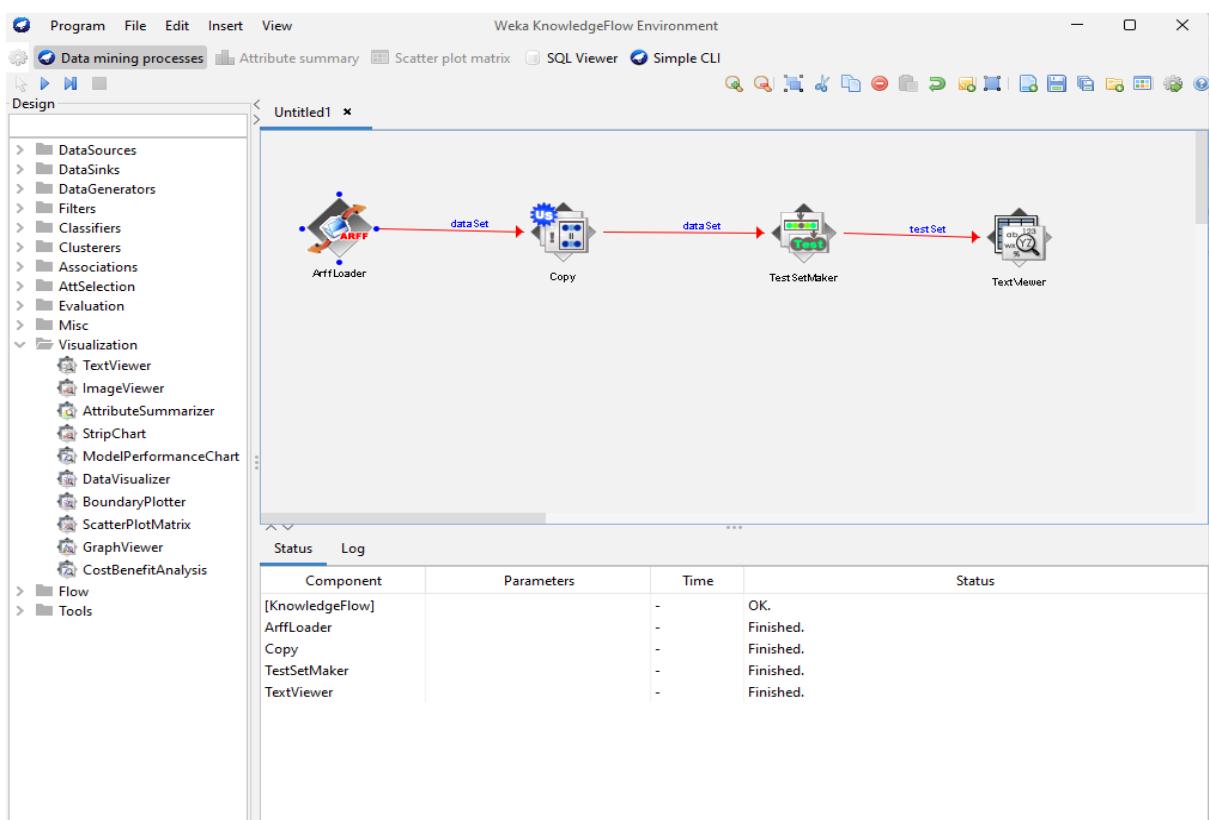
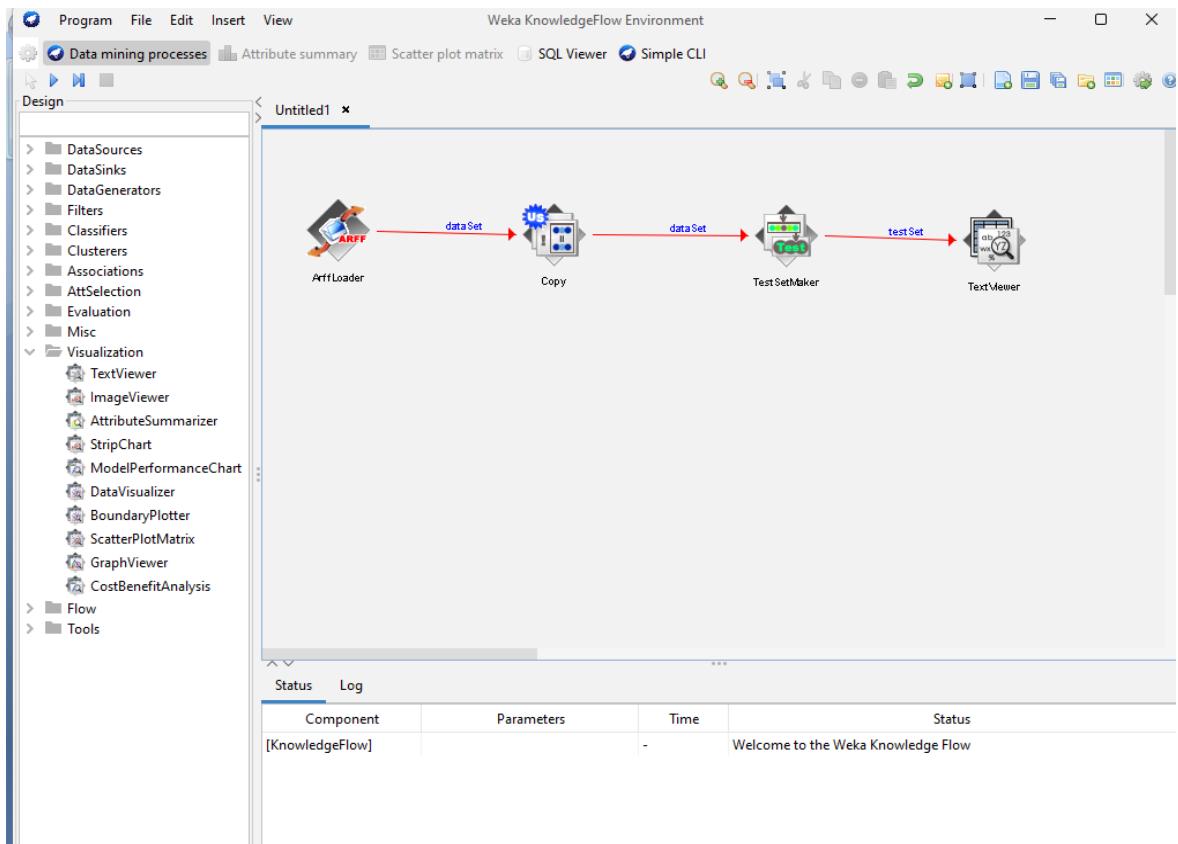
```

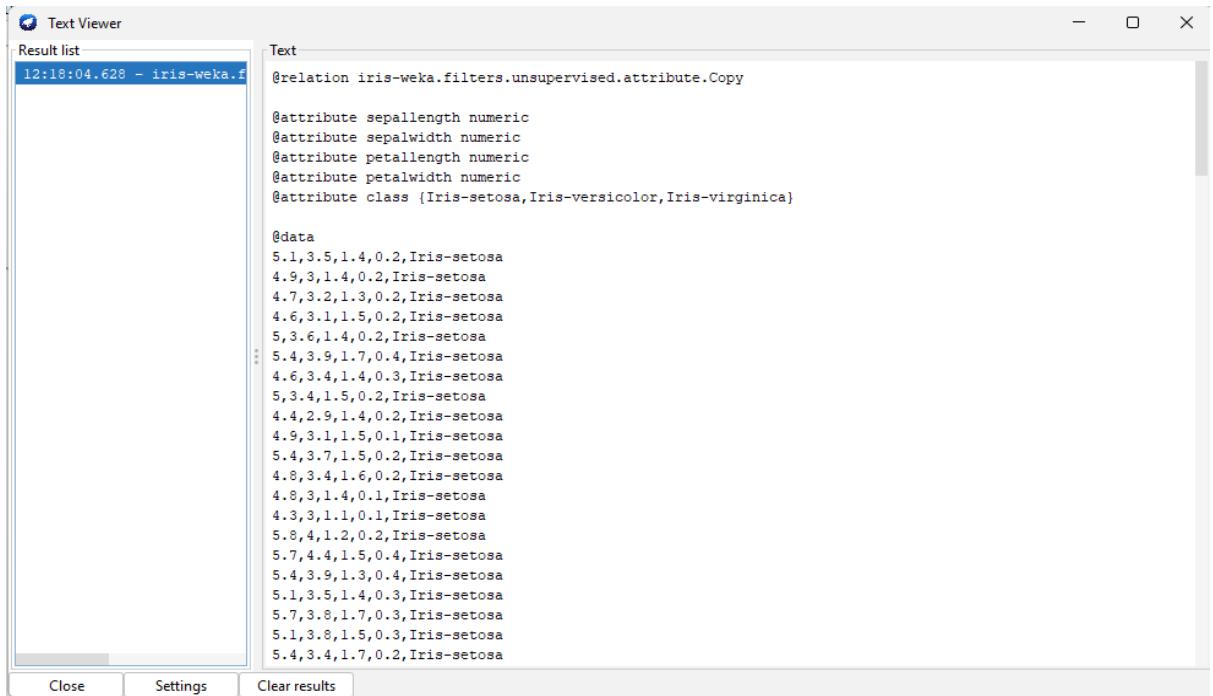
To Copy Attribute:











The screenshot shows a 'Text Viewer' window from the WEKA tool. The title bar says 'Text Viewer'. The main area is titled 'Text' and contains the following code:

```
@relation iris-weka.filters.unsupervised.attribute.Copy  
  
@attribute sepallength numeric  
@attribute sepalwidth numeric  
@attribute petallength numeric  
@attribute petalwidth numeric  
@attribute class {Iris-setosa,Iris-versicolor,Iris-virginica}  
  
@data  
5.1,3.5,1.4,0.2,Iris-setosa  
4.9,3,1.4,0.2,Iris-setosa  
4.7,3.2,1.3,0.2,Iris-setosa  
4.6,3.1,1.5,0.2,Iris-setosa  
5,3.6,1.4,0.2,Iris-setosa  
5.4,3.9,1.7,0.4,Iris-setosa  
4.6,3.4,1.4,0.3,Iris-setosa  
5,3.4,1.5,0.2,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  
4.8,3,1.4,0.1,Iris-setosa  
4.3,3,1.1,0.1,Iris-setosa  
5.8,4,1.2,0.2,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
```

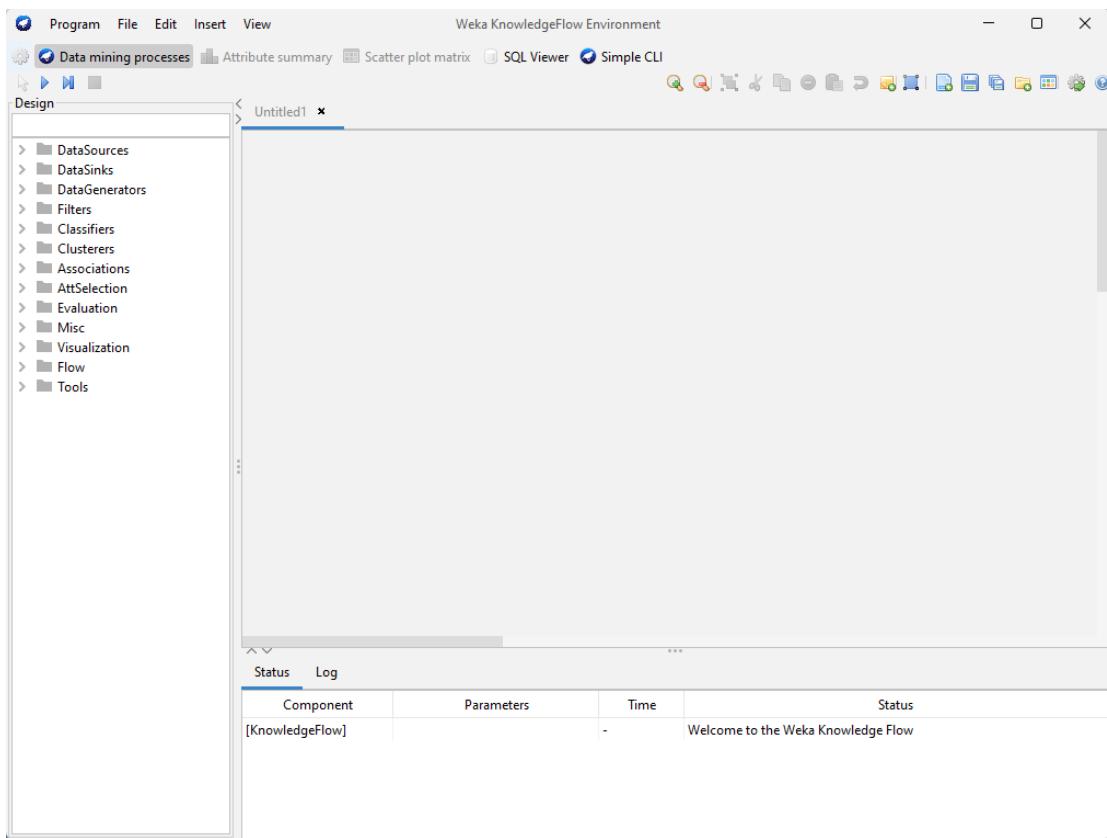
At the bottom of the window are three buttons: 'Close', 'Settings', and 'Clear results'.

→ To Add Attribute

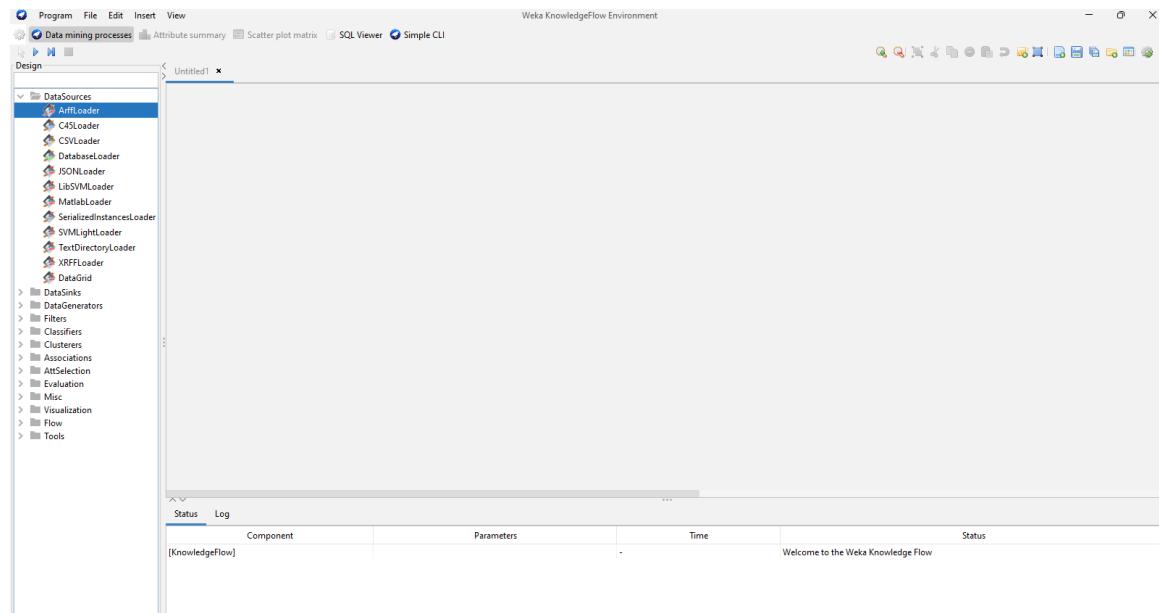
1. Open WEKA Tool and navigate to KnowledgeFlow.



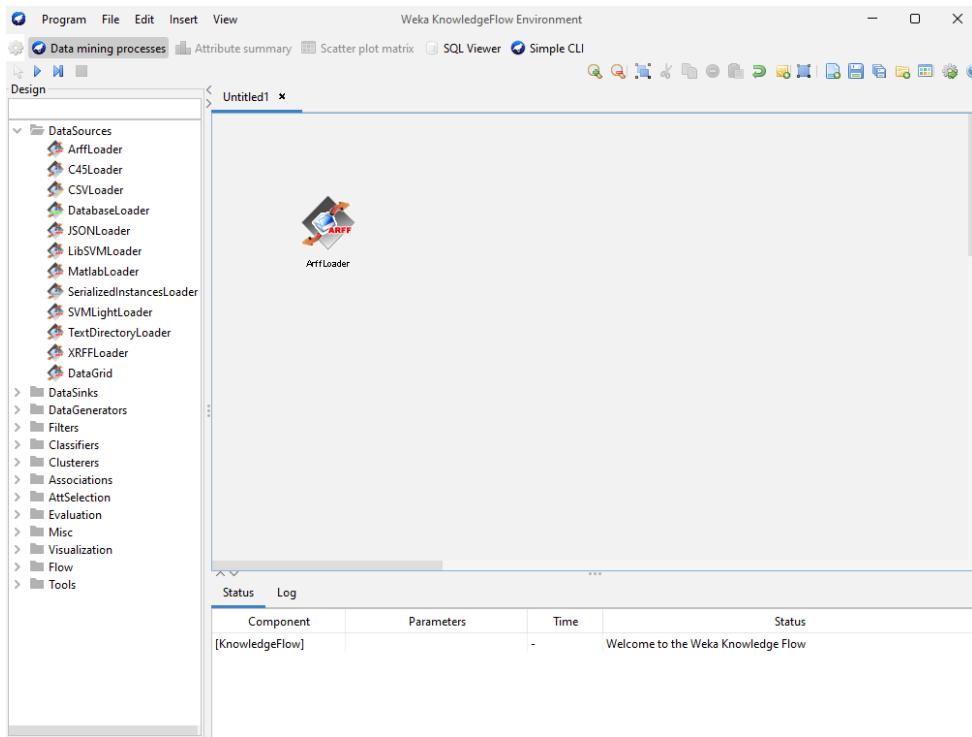
2. Knowledge Flow Environment views :,



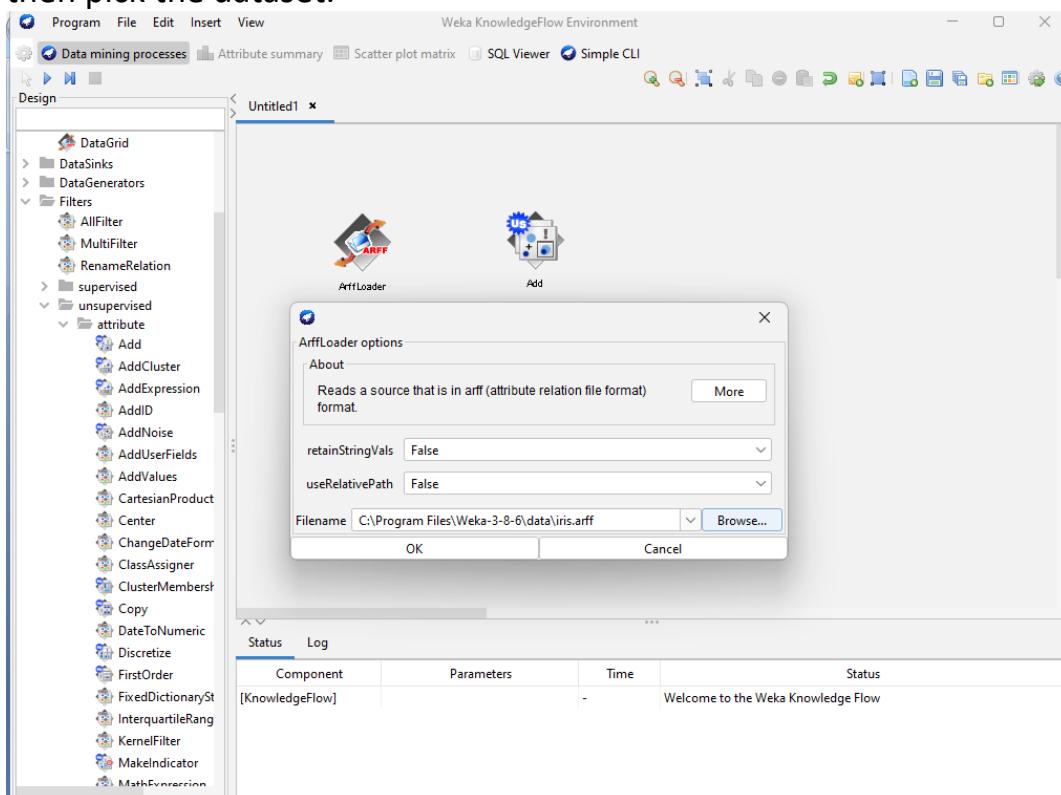
3. To create a model's Knowledge Flow, pick DataSources in the Design box. Expand data sources. To pick the dataset, use either ArffLoader or CSV Loader.



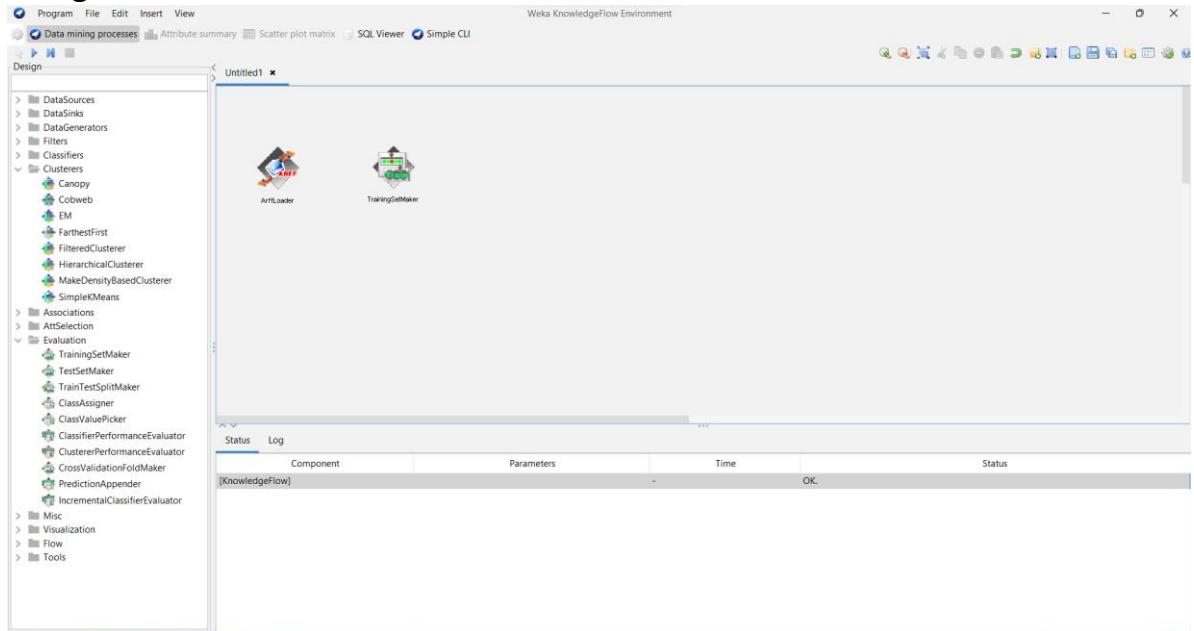
4. Click on the canvas window to add the Loader to the Knowledge Flow.



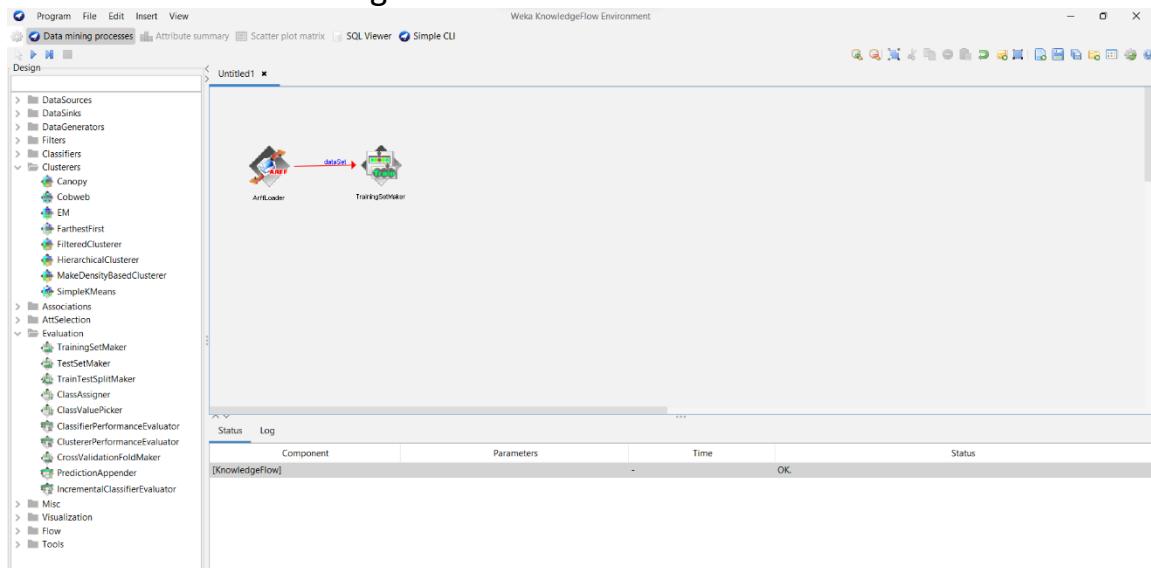
5. To load the dataset in ArffLoader, right-click on it, select Configure, and then pick the dataset.



6. To evaluate the dataset, select Evaluation > TrainingSetMaker in the Design Window.



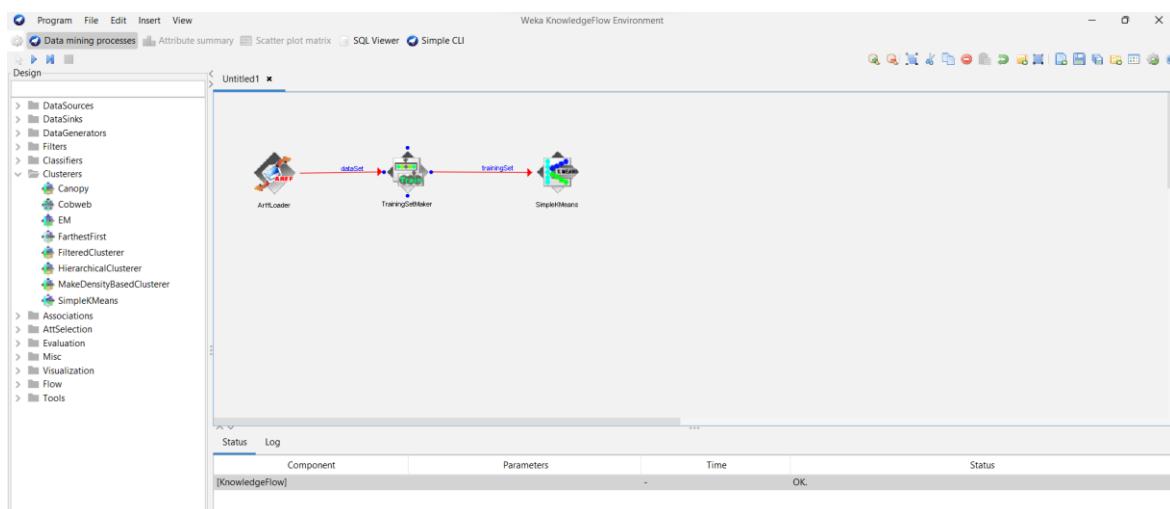
7. To link ArffLoader and TrainingSetMaker, right-click on Arffloader>dataSet and then choose TrainingSetMaker.



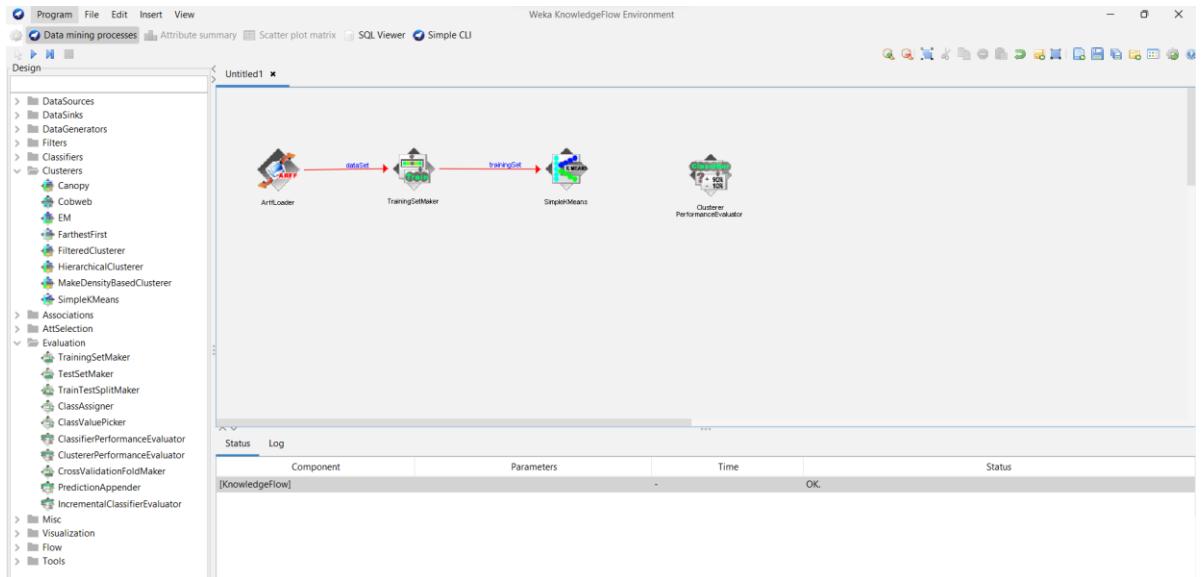
8. To add a cluster, navigate to Clusters>SimpleKMeans and set it on the Canvas Window.



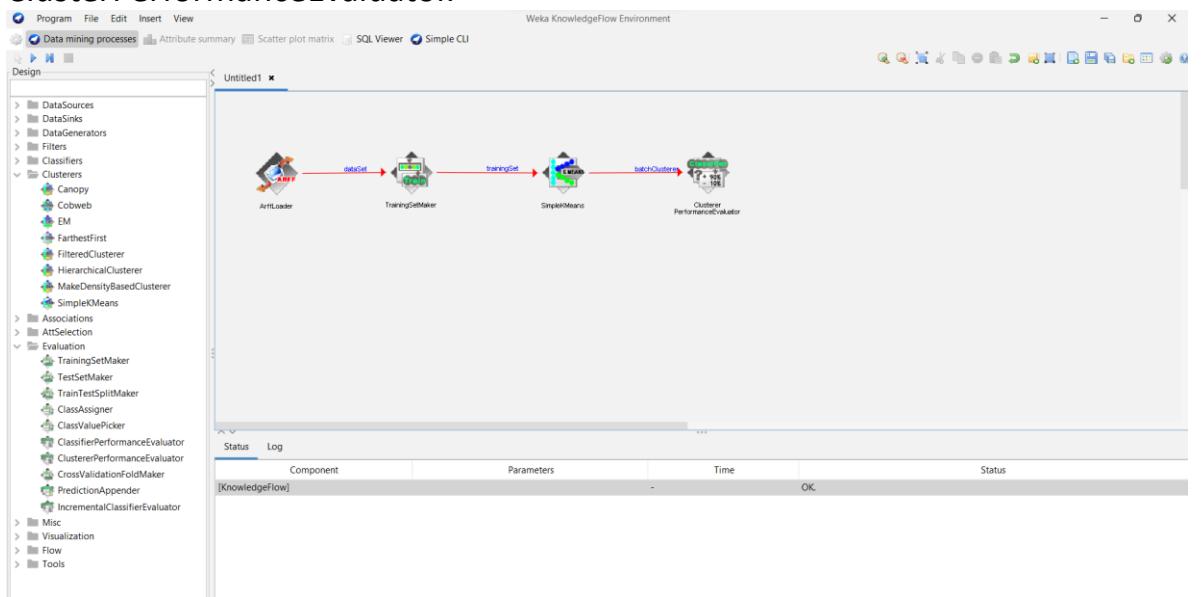
9. To link SimpleKMeans and TrainingSetMaker, right-click on TrainingSetMaker >trainingSet and then select TrainingSetMaker.



10. To assess the Cluster's performance, add a ClusterPerformanceEvaluator from the Design Window, choose Evaluation, and place it on Canvas.



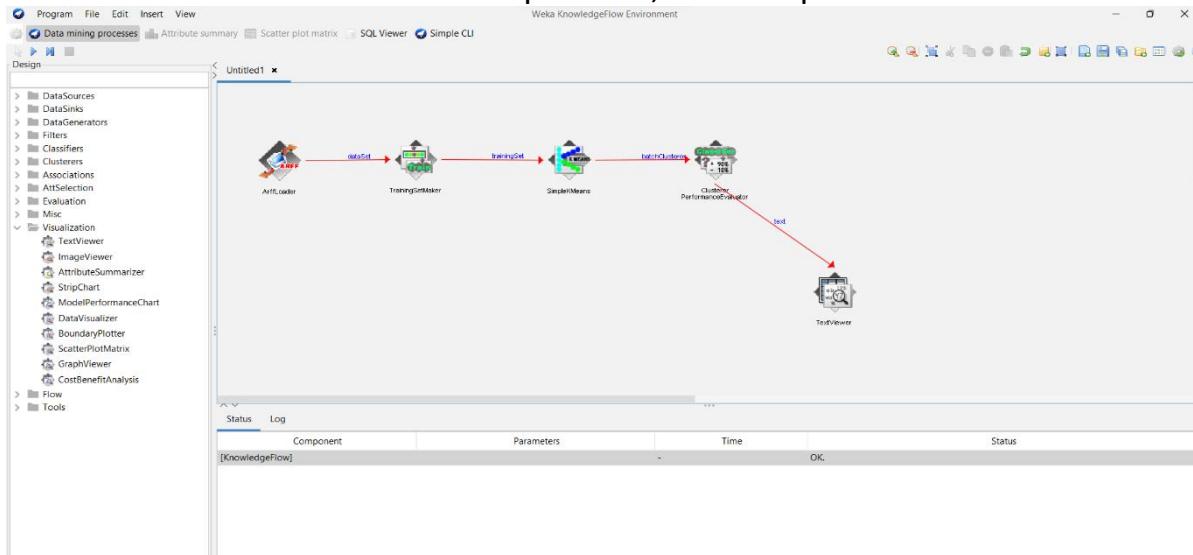
11. To connect Simple K Means and ClusterPerformanceEvaluator, right-click on SimpleKMeans and choose batchCluster, then link it to ClusterPerformanceEvaluator.



12. To view the results, we need to use Visualization. Select Visualization>TextViewer and place it on the canvas window.



13. To connect ClusterPerformanceEvaluator and TextViewer, right-click on ClusterPerformanceEvaluator and pick text, then drop it onto TextViewer.



14. In the beginning the Knowledge Flow, click the Play button in the Menu bar.



15. To examine the results, right-click on TextViewer and select Show Results.

The image contains two side-by-side screenshots of a "Text Viewer" application window. Both windows have a title bar "Text Viewer", a menu "Result list", and buttons "Close", "Settings", and "Clear results".

Screenshot 1: This window shows the full evaluation result for training instances. It includes the scheme (SimpleKMeans), relation (iris), kMeans parameters, iteration count (7), within cluster sum of squared errors (62.1436882815797), initial starting points, cluster assignments, missing value handling, final cluster centroids, and clustered instances.

```
Text
==== Evaluation result for training instances ====
Scheme: SimpleKMeans-init 0 -max-candidates 100 -periodic-pruning 10000 -min-density 2.0 -t1 -1.25 -t2 -1
Relation: iris

kMeans
=====

Number of iterations: 7
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          Cluster#
                  (150.0)           0             1
sepallength     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

Clustered Instances
0      100 ( 67%)
1      50 ( 33%)
```

Screenshot 2: This window shows a subset of the evaluation results, specifically the kMeans section. It includes the number of iterations (7), within cluster sum of squared errors (62.1436882815797), initial starting points, cluster assignments, missing value handling, final cluster centroids, and clustered instances.

```
Text
kmeans
=====

Number of iterations: 7
Within cluster sum of squared errors: 62.1436882815797

Initial starting points (random):

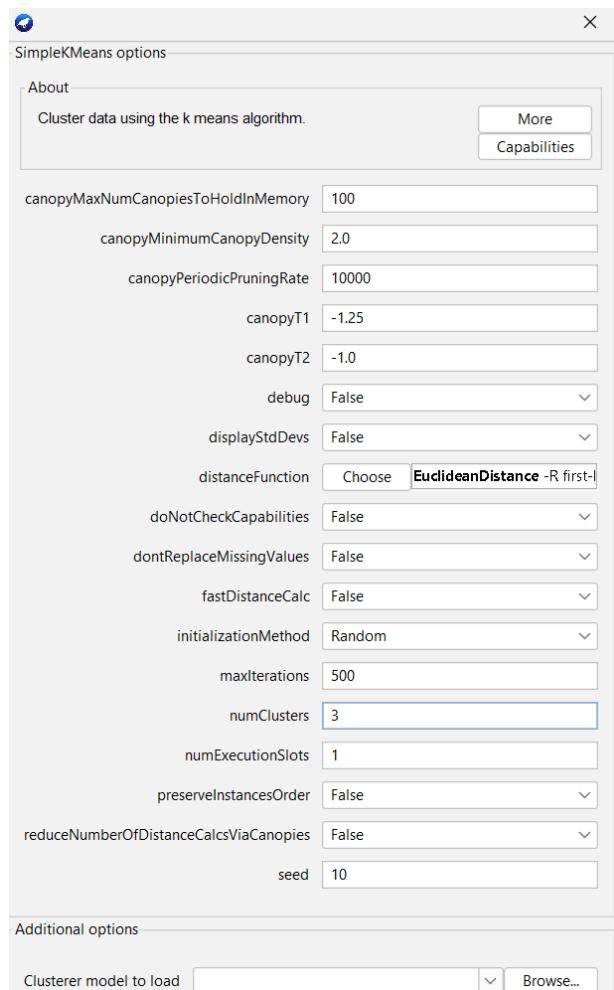
Cluster 0: 6.1,2.9,4.7,1.4,Iris-versicolor
Cluster 1: 6.2,2.9,4.3,1.3,Iris-versicolor

Missing values globally replaced with mean/mode

Final cluster centroids:
Attribute      Full Data          Cluster#
                  (150.0)           0             1
sepallength     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

Clustered Instances
0      100 ( 67%)
1      50 ( 33%)
```

15. We can configure SimpleKMeans to obtain Clustered Instances, right click on SimpleKMeans modify the number at numClusters



16. To view findings, re-run the KnowledgeFlow and right-click on the TextViewer.

The TextViewer window displays the following output from the SimpleKMeans run:

```

23:24:58.980 - SimpleKMeans
23:30:48.227 - SimpleKMeans
23:31:55.386 - SimpleKMeans

Number of iterations: 3
Within cluster sum of squared errors: 7.817456892309574

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    Cluster#
                  (150.0)      0          1          2
=====
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

Clustered Instances

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

```

The window also includes "Close", "Settings", and "Clear results" buttons at the bottom.

The numClusters changed to 3. So these are the Clustered Instances findings.