

## LAB PROGRAM - 2

### 2. Consider the following data set

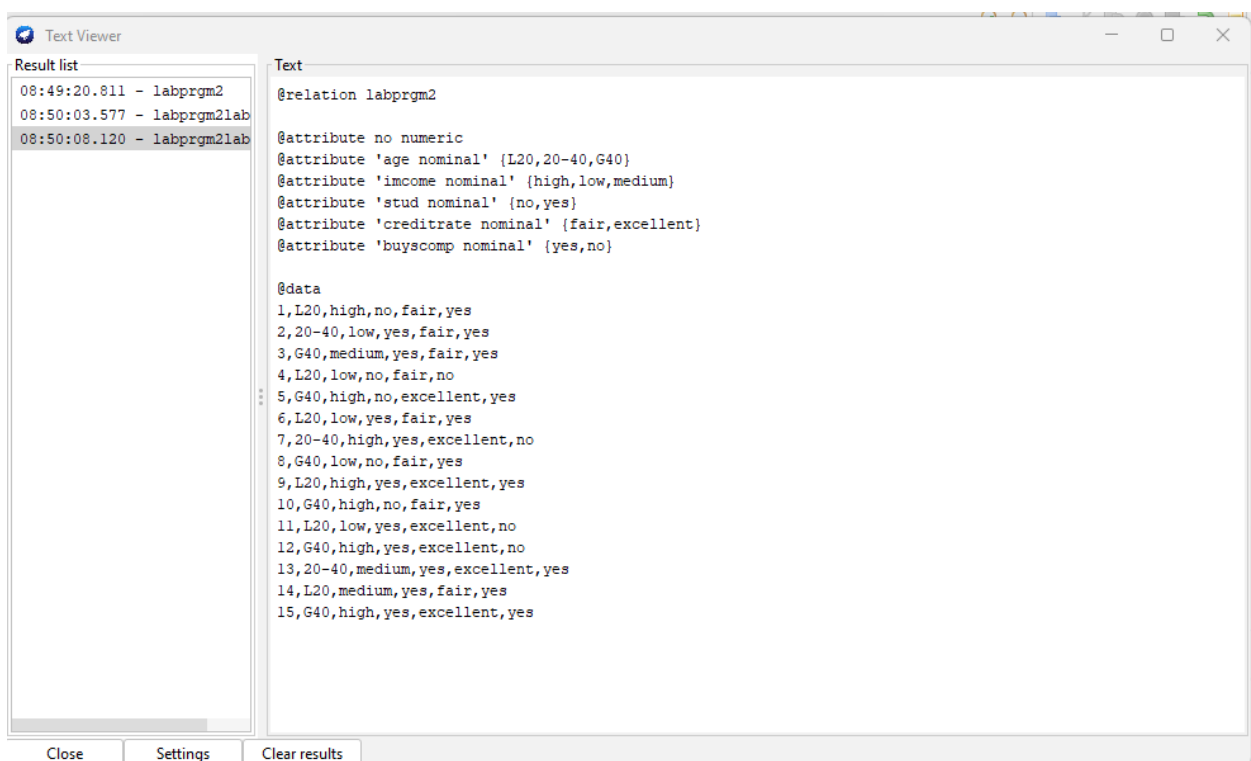
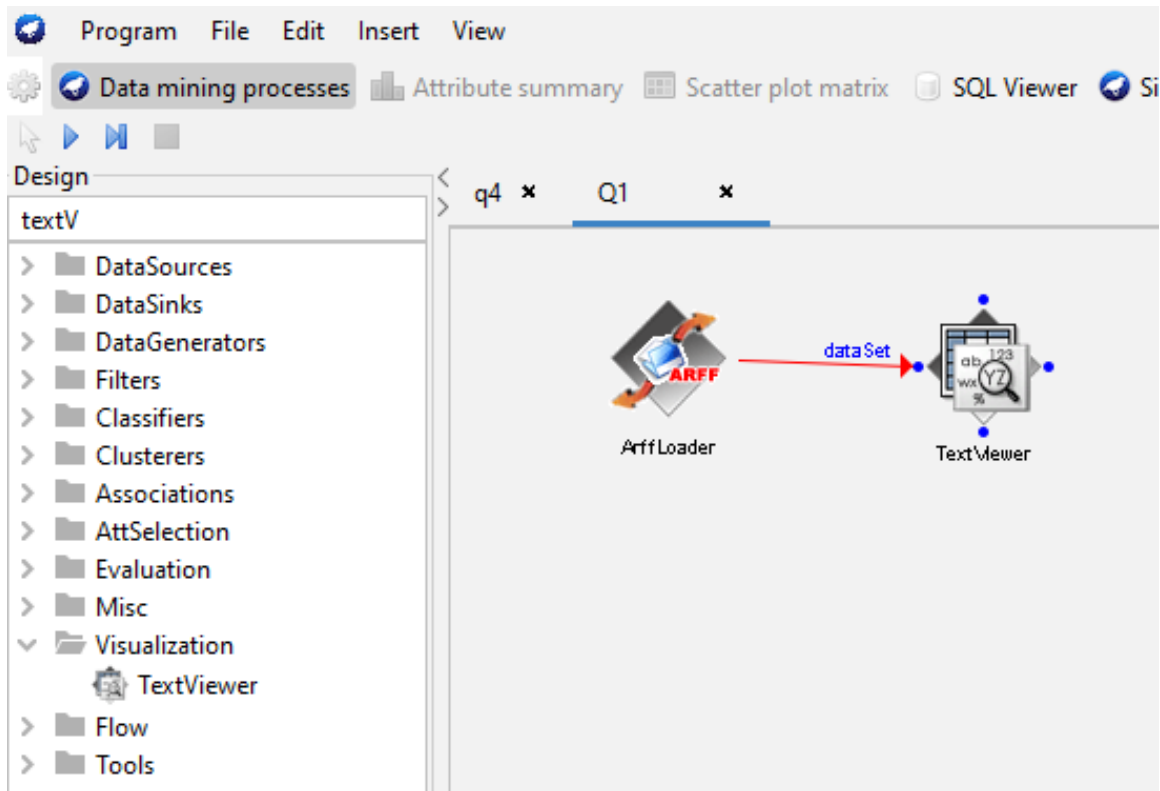
Relation: employee

No.	age Nominal	income Nominal	stud Nominal	creditrte Nominal	buyscomp Nominal
1	L20	high	no	fair	yes
2	20-40	low	yes	fair	yes
3	G40	medium	yes	fair	yes
4	L20	low	no	fair	no
5	G40	high	no	excellent	yes
6	L20	low	yes	fair	yes
7	20-40	high	yes	excellent	no
8	G40	low	no	fair	yes
9	L20	high	yes	excellent	yes
10	G40	high	no	fair	yes
11	L20	low	yes	excellent	no
12	G40	high	yes	excellent	no
13	20-40	medium	yes	excellent	yes
14	L20	medium	yes	fair	yes
15	G40	high	yes	excellent	yes

- Load ARFF file and explore knowledge flow interface
- configure the data source , check the status area after executing the configuration
- Perform operations such as Attribute Selection, Filter, Classify, Data Sink, Visualization and Evaluation
- Apply incremental learning and analyze the result
- do clustering : use generator properties, two clustering schemes, and result panel
- Generate classification Matrix and Construct Decision tree
- Perform Linear Regresssion and Analyze , Validate and Visualize the data

## i) Load ARFF File and Explore Knowledge Flow Interface

- Open Weka 3.9.6 and choose **Knowledge Flow** from the main window.
- In the Knowledge Flow interface, go to the toolbar and click on the **Data Sources** tab.
- Drag the **ARFFLoader** component from the list to the main canvas.
- Double-click the ARFFLoader icon, select the ARFF file you want to load, and click **OK**.
- Connect the **ARFFLoader** to a **DataSource** or **Dataset** component to view data flow.
- Explore the various options and panels (like visualization, preprocessing, etc.) in the interface.



## ii) Configure the Data Source and Check the Status Area

- Once the ARFFLoader is connected, right-click it and select **Start Loading**.
- You will see updates in the **Status Area** (usually located at the bottom of the Knowledge Flow window).
- Make sure the data loads successfully by viewing the progress bar or status log.

Status Log			
Component	Parameters	Time	Status
[KnowledgeFlow]		-	OK.
ArffLoader		-	Finished.
TextViewer		-	Finished.

## iii) Perform Operations: Attribute Selection, Filter, Classify, Data Sink, Visualization, and Evaluation

### 1. Attribute Selection:

- Add the **Attribute Selection** component from the list (under "Filter").
- Connect the ARFFLoader or Dataset to the **Attribute Selection** component.
- Double-click to configure and select the desired selection method (e.g., CFS, InfoGain, etc.).
- Start the process and view the selected attributes.

### 2. Filter:

- Add a **Filter** component (such as **Normalize** or **Discretize**) and connect it to the Dataset.
- Configure the filter and apply it.

### 3. Classify:

- Add a **Classifier** (e.g., J48 for decision trees, or NaiveBayes).
- Connect the **Filtered Dataset** to the **Classifier**.
- Configure the classifier, run it, and check the results.

### 4. Data Sink:

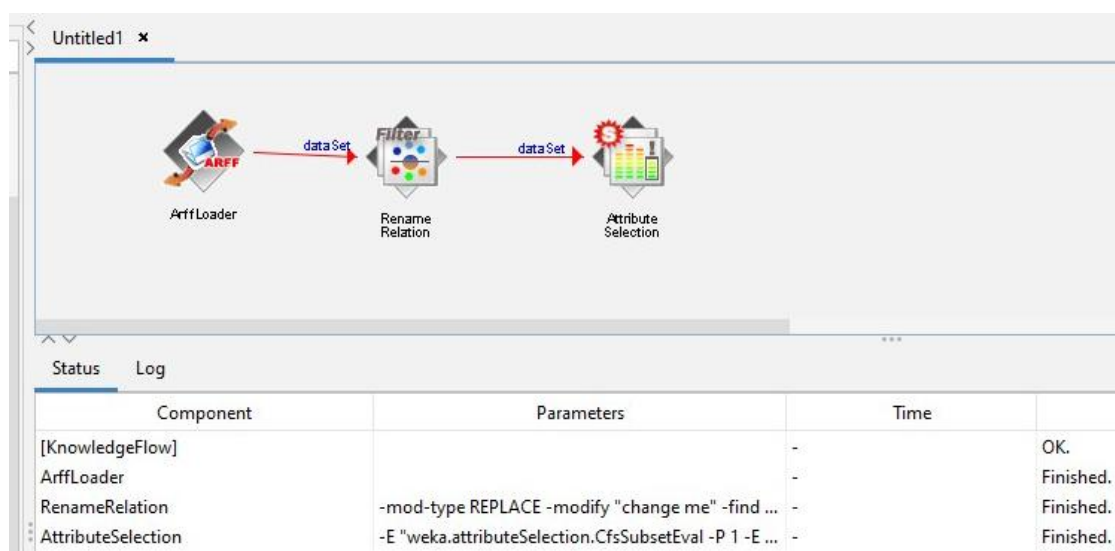
- Use **DataSink** to output results to a file.
- Connect the Classifier's output to **DataSink**, specify the output file, and run the process.

### 5. Visualization:

- Use **Graph Viewer** to view data plots.
- Connect a **DataSource** or **Classifier** output to it for visualization.

### 6. Evaluation:

- Add an **Evaluation** component (e.g., **Cross Validation** or **Percentage Split**).
- Connect the classifier to it, configure, and evaluate the model.
- View results such as confusion matrix and performance metric.



Program File Edit Insert View Weka KnowledgeFlow Environment

Data mining processes Attribute summary Scatter plot matrix SQL Viewer Simple CLI

Design

treeV

- DataSourcees
- DataSinks
- DataGenerators
- Filters
- Classifiers
- Clusterers
- Associations
- AttSelection
- Evaluation
- Misc
- Visualization
- Flow
- Tools

q4 x Q1 x q3 x

```

graph LR
    ARFFLoader[ARFF Loader] -- "dataSet" --> ClassAssigner[ClassAssigner]
    ClassAssigner -- "dataSet" --> CrossValidationFoldMaker[CrossValidation FoldMaker]
    CrossValidationFoldMaker -- "testSet trainingSet" --> J48[J48]
    J48 -- "batchClassifier" --> ClassifierPerformanceEvaluator[Classifier Performance Evaluator]
    ClassifierPerformanceEvaluator -- "text" --> TextViewer[TextViewer]
  
```

Status Log

Component	Parameters	Time	
ARFFLoader		-	finished.
ClassAssigner	-C last	-	Finished.
CrossValidationFoldMaker		-	Finished.
J48	-C 0.25 -M 2	-	Finished.
ClassifierPerformanceEvaluator		-	Finished.
TextViewer		-	Finished.

Text Viewer

Result list

09:16:43.268 - J48

Text

```

=== Evaluation result ===

Scheme: J48
Options: -C 0.25 -M 2
Relation: labprgm2-weka.filters.unsupervised.attribute.ClassAssigner-Clas

=== Summary ===

Correctly Classified Instances      10           66.6667 %
Incorrectly Classified Instances     5           33.3333 %
Kappa statistic                    -0.1194
Mean absolute error                 0.4339
Root mean squared error             0.4938
Relative absolute error             104.4801 %
Root relative squared error         107.8336 %
Total Number of Instances          15

=== Detailed Accuracy By Class ===

                TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                0.909    1.000    0.714     0.909    0.800     -0.161   0.182    0.621    yes
                0.000    0.091    0.000     0.000    0.000     -0.161   0.182    0.267    no
Weighted Avg.   0.667    0.758    0.524     0.667    0.587     -0.161   0.182    0.526

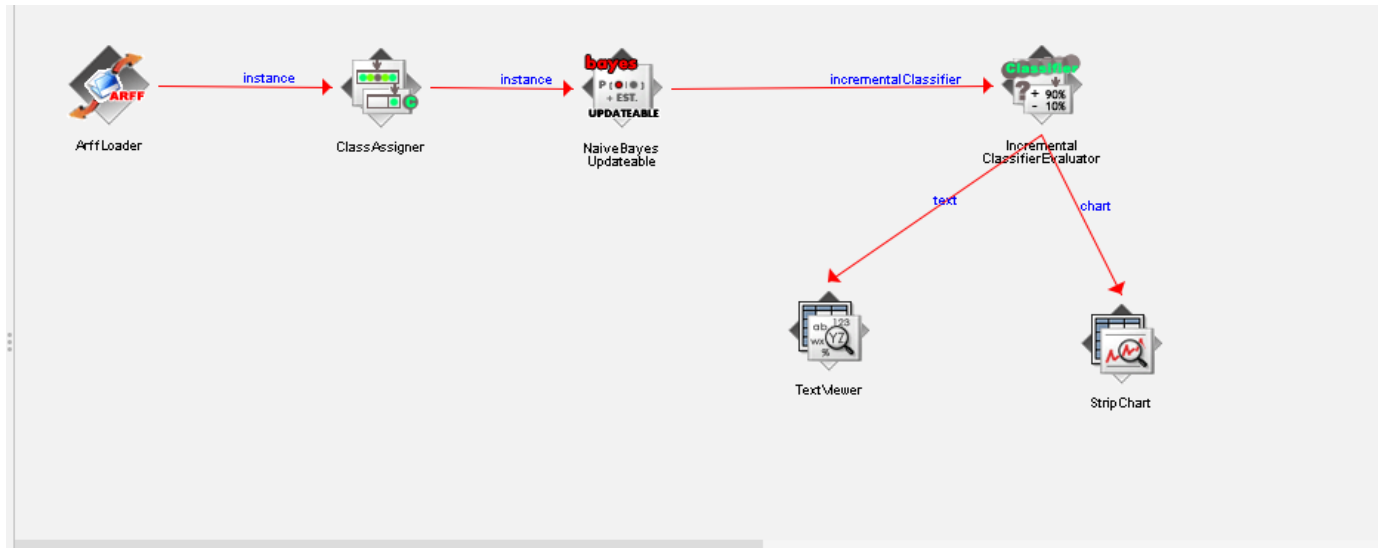
=== Confusion Matrix ===

 a  b  <-- classified as
10  1  |  a = yes
 4  0  |  b = no
  
```

Close Settings Clear results

#### iv) Apply Incremental Learning and Analyze the Result

- Use an Incremental Classifier (like NaiveBayesUpdateable).
- Add the **IncrementalClassifier** component and connect it to the data source.
- Configure the classifier and enable incremental updates.
- Execute the process and monitor how the model is updated with new data chunks.
- Analyze performance by using an evaluation component to track learning over time.



Status		Log		
Component	Parameters	Time	Status	
ArffLoader	overall flow throughput -	-	finished - 13 insts @ 1305 insts/sec	
ClassAssigner	-	-	Finished - 15 insts @ 535 insts/sec	
NaiveBayesUpdateable	-	-	Finished - 15 insts @ 833 insts/sec	
IncrementalClassifierEvaluator	-	-	Finished - 15 insts @ 15000 insts/sec	
StripChart	-	-	Finished - 15 insts @ 15000 insts/sec	
TextViewer	-	-	Finished.	

Text Viewer

Result list

08:32:58.631 - Results: Na  
08:40:43.989 - Results: Na

Text

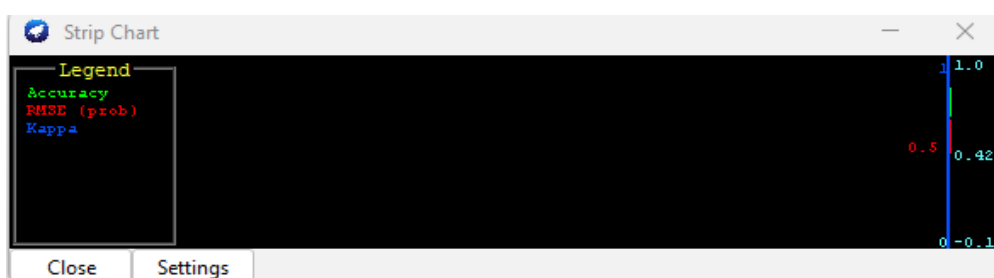
```
=== Performance information ===

Scheme: NaiveBayesUpdateable
Relation: labprgm2

Correctly Classified Instances      9      60 %
Incorrectly Classified Instances    6      40 %
Kappa statistic                    -0.2162
Mean absolute error                 0.4936
Root mean squared error             0.6116
Total Number of Instances          15

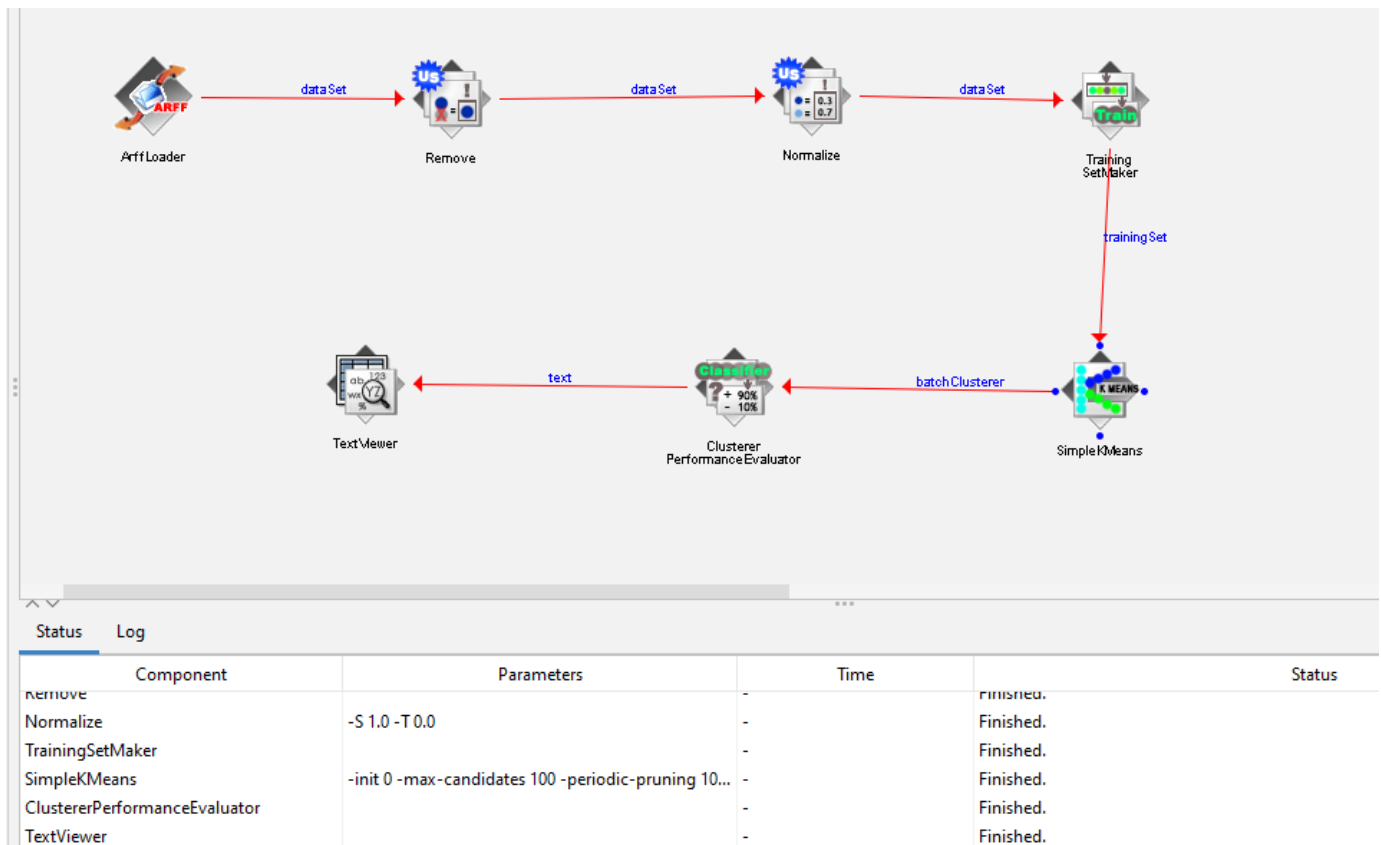
=== Confusion Matrix ===

a b  <-- classified as
9 2 | a = yes
4 0 | b = no
```



## v) Do Clustering: Use Generator Properties, Two Clustering Schemes, and Result Panel

- **Remove:** Likely used to remove unwanted attributes.
- **Normalize:** Applies normalization to the data.
- **TrainingSetMaker:** Prepares the dataset for training.
- **SimpleKMeans:** This is where you are applying a clustering algorithm (K-Means).
- **ClusterPerformanceEvaluator:** Evaluates the results of clustering.
- **TextViewer:** Outputs the evaluation result.



Text Viewer

Result list

09:48:57.560 - SimpleKMean

Text

```

=====
Number of iterations: 3
Within cluster sum of squared errors: 23.1890589569161

Initial starting points (random):

Cluster 0: 0.214286,L20,low,no,fair,no
Cluster 1: 0.571429,L20,high,yes,excellent,yes

Missing values globally replaced with mean/mode

Final cluster centroids:

Attribute          Full Data      Cluster#
                   (15.0)        (6.0)        (9.0)
=====
no                  0.5          0.3452      0.6032
age nominal         L20          L20          G40
income nominal      high         low          high
stud nominal        yes          no           yes
creditrate nominal  fair         fair         excellent
buyscomp nominal    yes          yes          yes

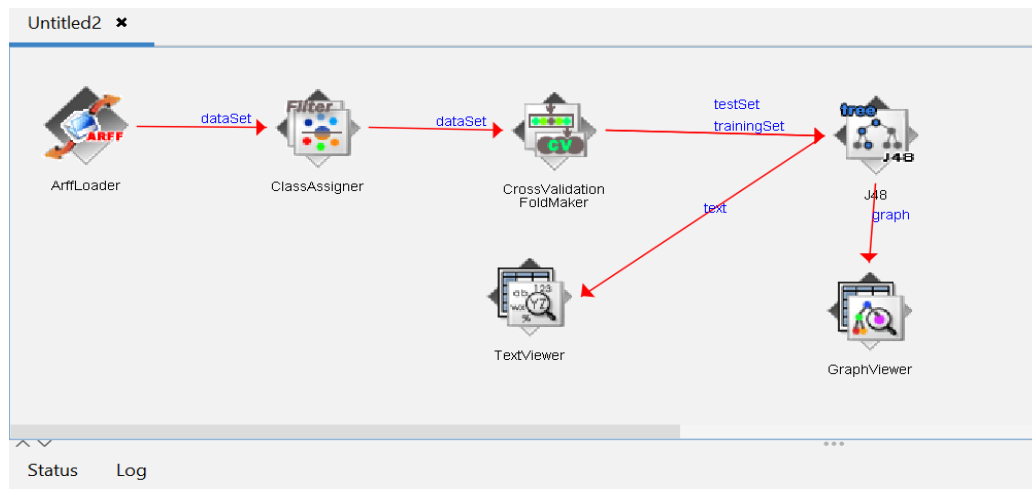
Clustered Instances

0          6 ( 40%)
1          9 ( 60%)
    
```

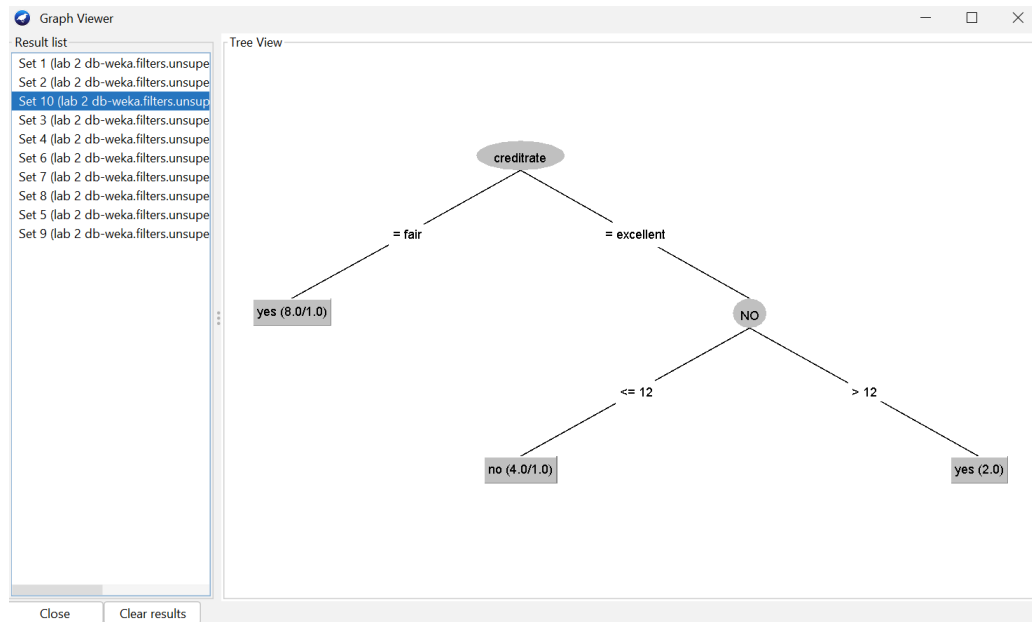
Close Settings Clear results

## vi) Generate classification Matrix and Construct Decision tree

- **ArffLoader** : Load your dataset.
- **J48** :Construct the decision tree classifier using the training data.
- **GraphViewer** :Visualize the results of the classification.
- **TextViewer** : Output the evaluation results for review.



Component	Parameters	Time	
[KnowledgeFlow]		-	OK.
ArffLoader		-	Finished.
ClassAssigner	-C last	-	Finished.
CrossValidationFoldMaker		-	Finished.
J48	-C 0.25 -M 2	-	Finished.
TextViewer		-	Finished.
GraphViewer		-	Finished.



```
Text
=== Classifier model ===
Scheme: J48
Relation: lab 2 db-weka.filters.unsupervised.attribute.ClassAssigner-Clas

J48 pruned tree
-----
creditrates = fair: yes (8.0/1.0)
creditrates = excellent
| NO <= 12: no (4.0/1.0)
| NO > 12: yes (2.0)

Number of Leaves : 3
Size of the tree : 5
```

## vii) Perform Linear Regression and Analyze , Validate and Visualize the data

- **ArffLoader** : Load your dataset.
- **Remove** : Clean the dataset by removing unwanted attributes.
- **Normalize** : Apply normalization to prepare the data for regression analysis.
- **TrainingSetMaker**: Create a training set from your data.
- **LinearRegression** : Perform linear regression on the training data.
- **Evaluation** : Evaluate the regression model's performance.
- **TextViewer** : Output the evaluation results for review.

