SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
Kattankulathur, Chennai-603203.

FACULTY OF ENGINEERING AND TECHNOLOGY

School of Computing

Department of Data Science and Business Systems

Academic Year (2022–2023)

18CSE366J – DATAMINING AND ANALYTICS

SEMESTER – VI

SRM INSTITUTE OF SCIENCE
ANDTECHNOLOGY
Kattankulathur, Chennai-603203.

FACULTY OF ENGINEERING AND TECHNOLOGY

18CSE366J – DATA MINING AND ANALYTICS

BONAFIDE CERTIFICATE

Certified that this is the Bonafide record of work done by **Puneet Madan RA2011042010002** of **VI** semester B.Tech COMPUTER SCIENCE AND ENGINEERING WITH BUSINESS SYSTEMS during the academic year2022-2023 in the 18CSE366J – DATA MINING AND ANALYTICS

Dr. JEEVA.S.
Staff -Incharge

Dr. M. Lakshmi
Head of the Department

Submitted for the practical examination held on___________at

SRM institute of science and Technology

EXAMINER – 1

EXAMINER-2

INDEX

EXPERIMENT – 1

AIM: Demonstration of Preprocessing on Dataset student.arff

Dataset student.arff:

@relation student
@attribute age {<30,30-40,>40}
@attribute income {low, medium,
high}@attribute student {yes, no}
@attribute credit-rating {fair,
excellent}@attribute buyspc {yes, no}
@data
%
<30, high, no, fair, no
<30, high, no, excellent, no
30-40, high, no, fair, yes
>40, medium, no, fair, yes
>40, low, yes, fair, yes
>40, low, yes, excellent,
no30-40, low, yes, excellent,
yes<30, medium, no, fair,
no <30, low, yes, fair, no
>40, medium, yes, fair, yes
<30, medium, yes, excellent,
yes30-40, medium, no, excellent,
yes 30-40, high, yes, fair, yes
>40, medium, no, excellent, no
%

PROCEDURE:

Step1: Loading the data. We can load the dataset into weka by clicking on open button in preprocessing interface and selecting theappropriate file.

Step2: Once the data is loaded, weka will recognize the attributes andduring the scan of the data weka will compute some basic strategies on each attribute.

The left panel in the above figure shows the list of recognized attributes while the top panel indicates the names of the base relationor table and the current working relation (which are same initially).

Step3: Clicking on an attribute in the left panel will show the basic

statistics on the attributes for the categorical attributes the frequency of each attribute value is shown, while for continuous attributes we can obtain min, max, mean, standard deviation and deviation etc.,
Step4: The visualization in the right button panel in the form of cross-tabulation across two attributes.
Step5: Selecting or filtering attributes Removing an attribute-When we need to remove an attribute, we can do this by using the attribute filters in weka. In the filter model panel, click on choose button, Thiswill show a popup window with a list of available filters.

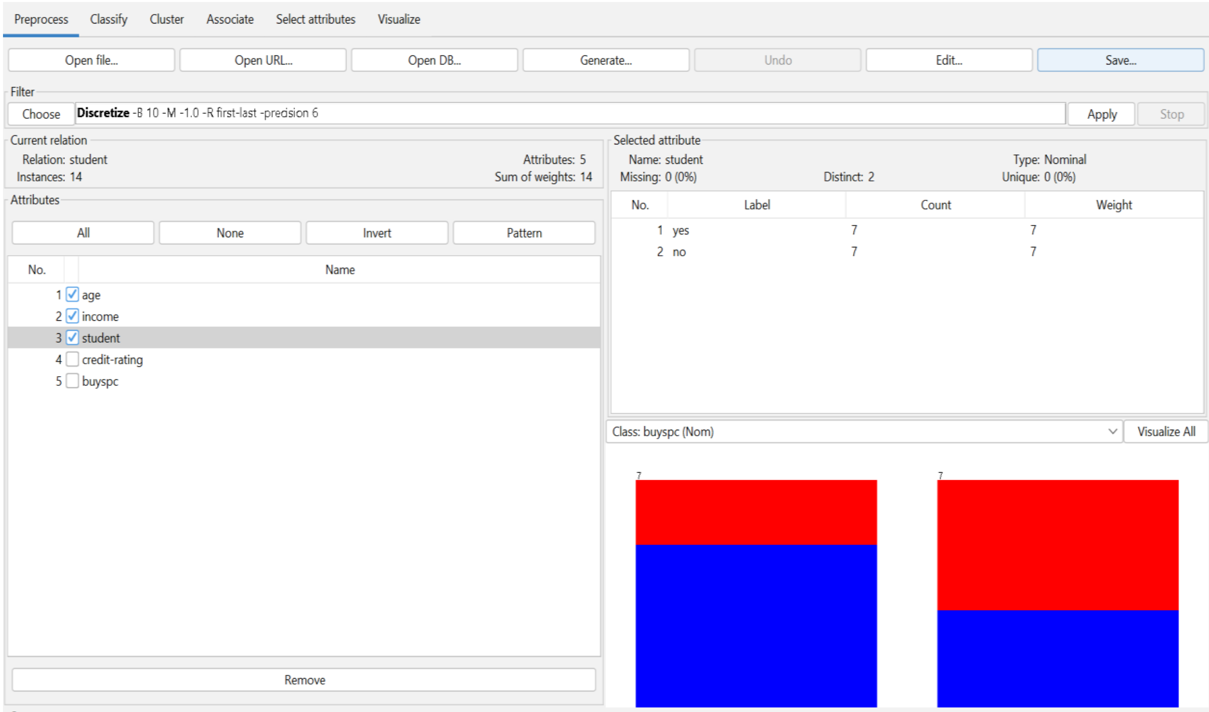
FILTER APPLIED:

*** DISCRETIZATION:**
Sometimes association rule mining can only be performed on categorical data. This requires performing discretization on numeric or continuous attributes. In the following example let us discretize ageattribute.

a. Let us divide the values of age attribute into three bins(intervals).
b. First load the dataset into weka(student.arff)
c. Select the age attribute.
d. Activate filter-dialog box and select "WEKA.filters.unsupervised.attribute.discretize"from the list.
e. We enter the index for the attribute to be discretized. In this case the attribute is age. So, we must enter '1' corresponding to the age attribute.
f. Enter '3' as the number of bins. Leave the remaining field values asthey are. Click OK button.
g. Click apply in the filter panel. This will result in a new workingrelation
with the selected attribute partition into 3 bins.
h. Save the new working relation in a file called student-data-discretized.arff

RESULT:

Shows the effect of discretization.



A demonstration of the preprocessing of the dataset student.arff hasbeen successfully completed.

EXPERIMENT – 2

AIM: Demonstration of Preprocessing on labor. arff

Labor.arff Dataset:

%

%

@relation labor

@attribute 'duration' real

@attribute 'wage-increase-first-year' real

@attribute 'wage-increase-second-year' real

@attribute 'wage-increase-third-year' real

@attribute 'cost-of-living-adjustment' {'none','tcf','tc'}

@attribute 'working-hours' real

@attribute 'pension' {'none','ret_allw','empl_contr'}

@attribute 'standby-pay' real

@attribute 'shift-differential' real

@attribute 'education-allowance' {'yes','no'}

@attribute 'statutory-holidays' real

@attribute 'vacation' {'below_average','average','generous'}

@attribute 'longterm-disability-assistance' {'yes','no'}

@attribute 'contribution-to-dental-plan' {'none','half','full'}

@attribute 'bereavement-assistance' {'yes','no'}

@attribute 'contribution-to-health-plan' {'none','half','full'}

@attribute 'class' {'bad','good'}

@data

1,5,?,?,?,40,?,?,2,?,11,'average',?,?,'yes',?,'good'

2,4.5,5.8,?,?,35,'ret_allw',?,?,'yes',11,'below_average',?,'full',?,'full','good'

?,?,?,?,?,38,'empl_contr',?,5,?,11,'generous','yes','half','yes','half','good'

3,3.7,4,5,'tc',?,?,?,?,'yes',?,?,?,?,'yes',?,'good'

3,4.5,4.5,5,?,40,?,?,?,?,12,'average',?,'half','yes','half','good'

2,2,2.5,?,?,35,?,?,6,'yes',12,'average',?,?,?,?,'good'

3,4,5,5,'tc',?,'empl_contr',?,?,?,12,'generous','yes','none','yes','half','good'

3,6.9,4.8,2.3,?,40,?,?,3,?,12,'below_average',?,?,?,?,'good'

2,3,7,?,?,38,?,12,25,'yes',11,'below_average','yes','half','yes',?,'good'

1,5.7,?,?,'none',40,'empl_contr',?,4,?,11,'generous','yes','full',?,?,'good'

3,3.5,4,4.6,'none',36,?,?,3,?,13,'generous',?,?,'yes','full','good'

2,6.4,6.4,?,?,38,?,?,4,?,15,?,?,'full',?,?,'good'

2,3.5,4,?,'none',40,?,?,2,'no',10,'below_average','no','half',?,'half','bad'

3,3.5,4,5.1,'tcf',37,?,?,4,?,13,'generous',?,'full','yes','full','good'

1,3,?,?,'none',36,?,?,10,'no',11,'generous',?,?,?,?,'good'

2,4.5,4,?,'none',37,'empl_contr',?,?,?,11,'average',?,'full','yes',?,'good'

1,2.8,?,?,?,35,?,?,2,?,12,'below_average',?,?,?,?,'good'

1,2.1,?,?,'tc',40,'ret_allw',2,3,'no',9,'below_average','yes','half',?,'none','ba
d'

1,2,?,?,'none',38,'none',?,?,'yes',11,'average','no','none','no','none','bad'

2,4,5,?,'tcf',35,?,13,5,?,15,'generous',?,?,?,?,'good'

2,4.3,4.4,?,?,38,?,?,4,?,12,'generous',?,'full',?,'full','good'

2,2.5,3,?,?,40,'none',?,?,?,11,'below_average',?,?,?,?,'bad'

3,3.5,4,4.6,'tcf',27,?,?,?,?,?,?,?,?,?,?,?,'good'

2,4.5,4,?,?,40,?,?,4,?,10,'generous',?,'half',?,'full','good'

1,6,?,?,?,38,?,8,3,?,9,'generous',?,?,?,?,'good'

3,2,2,2,'none',40,'none',?,?,?,10,'below_average',?,'half','yes','full','bad'

2,4.5,4.5,?,'tcf',?,?,?,?,'yes',10,'below_average','yes','none',?,'half','good'

2,3,3,?,'none',33,?,?,?,'yes',12,'generous',?,?,'yes','full','good'

2,5,4,?,'none',37,?,?,5,'no',11,'below_average','yes','full','yes','full','good'

3,2,2.5,?,?,35,'none',?,?,?,10,'average',?,?,'yes','full','bad'

3,4.5,4.5,5,'none',40,?,?,?,'no',11,'average',?,'half',?,?,'good'

3,3,2,2.5,'tc',40,'none',?,5,'no',10,'below_average','yes','half','yes','full','bad'

2,2.5,2.5,?,?,38,'empl_contr',?,?,?,10,'average',?,?,?,?,'bad'

2,4,5,?,'none',40,'none',?,3,'no',10,'below_average','no','none',?,'none','bad'

3,2,2.5,2.1,'tc',40,'none',2,1,'no',10,'below_average','no','half','yes','full','bad'

2,2,2,?,'none',40,'none',?,?,'no',11,'average','yes','none','yes','full','bad'

1,2,?,?,'tc',40,'ret_allw',4,0,'no',11,'generous','no','none','no','none','bad'

1,2.8,?,?,'none',38,'empl_contr',2,3,'no',9,'below_average','yes','half',?,'none','ba
d'

3,2,2.5,2,?,37,'empl_contr',?,?,?,10,'average',?,?,'yes','none','bad'

2,4.5,4,?,'none',40,?,?,4,?,12,'average','yes','full','yes','half','good'

1,4,?,?,'none',?,'none',?,?,'yes',11,'average','no','none','no','none','bad'

2,2,3,?,'none',38,'empl_contr',?,?,'yes',12,'generous','yes','none','yes','full','bad'

2,2.5,2.5,?,'tc',39,'empl_contr',?,?,?,12,'average',?,?,'yes',?,'bad'

2,2.5,3,?,'tcf',40,'none',?,?,?,11,'below_average',?,?,'yes',?,'bad'

2,4,4,?,'none',40,'none',?,3,?,10,'below_average','no','none',?,'none','bad'

2,4.5,4,?,?,40,?,?,2,'no',10,'below_average','no','half',?,'half','bad'

2,4.5,4,?,'none',40,?,?,5,?,11,'average',?,'full','yes','full','good'

2,4.6,4.6,?,'tcf',38,?,?,?,?,?,?,'yes','half',?,'half','good'

2,5,4.5,?,'none',38,?,14,5,?,11,'below_average','yes',?,?,'full','good'

2,5.7,4.5,?,'none',40,'ret_allw',?,?,?,11,'average','yes','full','yes','full','goo
d'

2,7,5.3,?,?,?,?,?,?,?,11,?,'yes','full',?,?,'good'

3,2,3,?,'tcf',?,'empl_contr',?,?,'yes',?,?,'yes','half','yes',?,'good'

3,3.5,4,4.5,'tcf',35,?,?,?,?,13,'generous',?,?,'yes','full','good'

3,4,3.5,?,'none',40,'empl_contr',?,6,?,11,'average','yes','full',?,'full','good'

3,5,4.4,?,'none',38,'empl_contr',10,6,?,11,'generous','yes',?,?,'full','good'

3,5,5,5,?,40,?,?,?,?,12,'average',?,'half','yes','half','good'

3,6,6,4,?,35,?,?,14,?,9,'generous','yes','full','yes','full','good'

%

%

PROCEDURE:

Step1: Loading the data. We can load the dataset into weka by clicking on open button in preprocessing interface and selecting the appropriate file.
Step2: Once the data is loaded, weka will recognize the attributes and during the scan of the data weka will compute some basic strategies on each attribute. The left panel in the above figure shows the list of recognized attributes while the top panel indicates the names of the base relation or table and the current working relation (which are same initially).
Step3: Clicking on an attribute in the left panel will show the basic statistics onthe attributes for the categorical attributes the frequency of each attribute valueis shown, while for continuous attributes we can obtain min, max, mean, standard deviation and deviation etc.,
Step4: The visualization in the right button panel in the form of cross-tabulationacross two attributes.
Step5: Selecting or filtering attributes Removing an attribute-When we need to

remove an attribute, we can do this by using the attribute filters in weka. In thefilter model panel, click on choose button, This will show a popup window with a list of available filters.

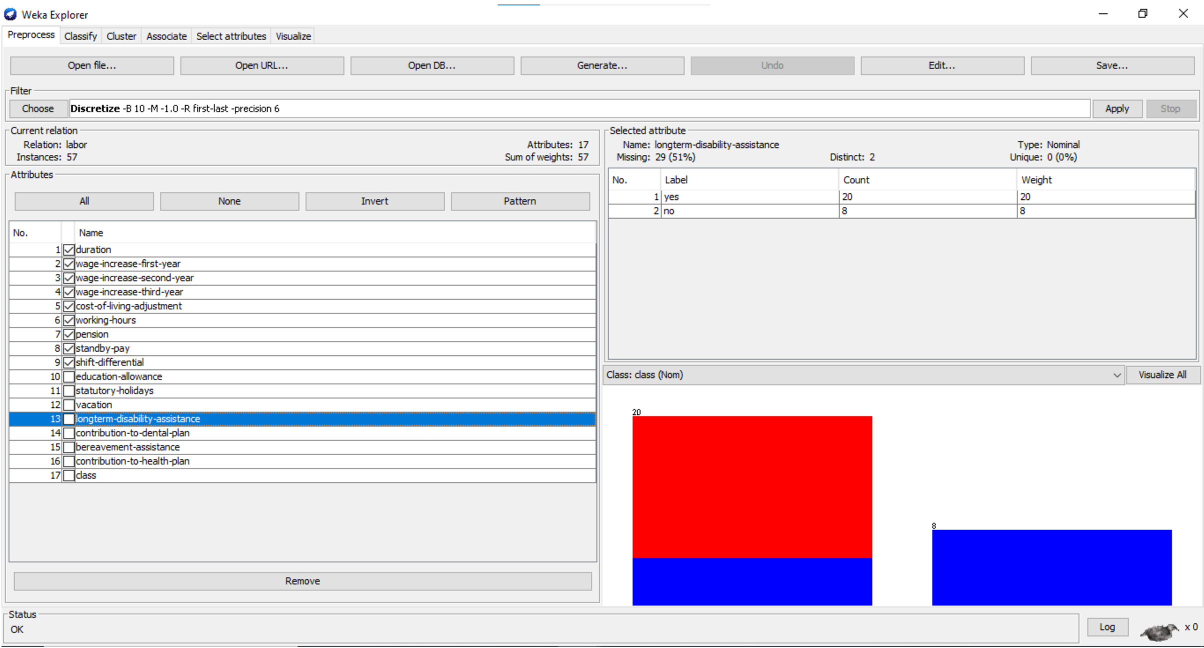
FILTER APPLIED:

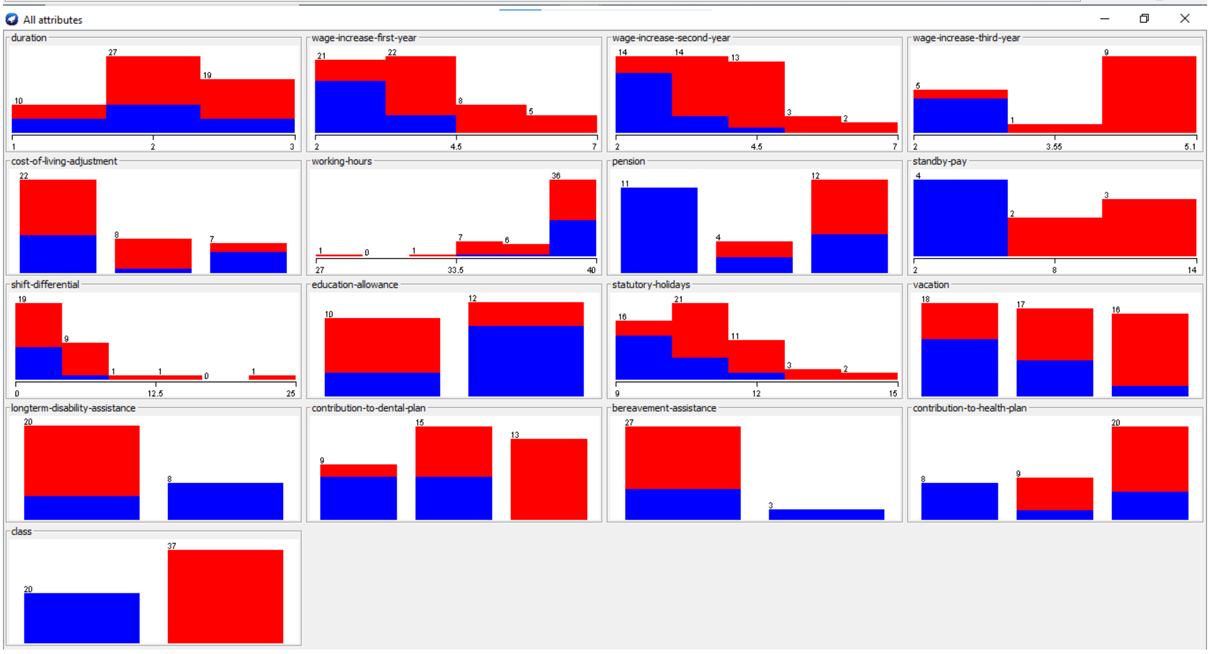
- **DISCRETIZATION:**
  Sometimes association rule mining can only be performed on categorical data. This requires performing discretization on numeric or continuous attributes. In the following example let us discretize age attribute.

  a. Let us divide the values of age attribute into three bins(intervals).
  b. First load the dataset into weka(labor.arff)
  c. Select the age attribute.
  d. Activate filter-dialog box and select "WEKA.filters.unsupervised.attribute.discretize"from the list.
  e. We enter the index for the attribute to be discretized.In this case theattribute is age. So we must enter '1' corresponding to the age attribute.

  f. Enter '3' as the number of bins. Leave the remaining field values asthey are.
  Click OK button.
  g. Click apply in the filter panel. This will result in a new workingrelation with the selected attribute partition into 3 bins.
  h. Save the new working relation in a file called labor-data-discretized.arff

OUTPUT:





RESULT:

A demonstration of the preprocessing of the dataset labor.arff has been successfully completed.

EXPERIMENT – 3

AIM: Demonstration of Association rule process on dataset contactlenses.arff using apriori algorithm.

Contactlenses.arff Dataset:

@relation contact-lenses
@attribute age {young, pre-presbyopic,
presbyopic}@attribute spectacle-prescrip {myope, hypermetrope}
@attribute astigmatism {no, yes}
@attribute tear-prod-rate {reduced,
normal}@attribute contact-lenses {soft, hard,
none} @data
%
% 24 instances
%
young,myope,no,reduced,none
young,myope,no,normal,soft
young,myope,yes,reduced,none
young,myope,yes,normal,hard
young,hypermetrope,no,reduced,none
young,hypermetrope,no,normal,soft
young,hypermetrope,yes,reduced,none
young,hypermetrope,yes,normal,hard
pre-presbyopic,myope,no,reduced,none
pre-presbyopic,myope,no,normal,soft
pre-
presbyopic,myope,yes,reduced,nonepre-
presbyopic,myope,yes,normal,hard
pre-presbyopic,hypermetrope,no,reduced,none
pre-presbyopic,hypermetrope,no,normal,soft
pre-
presbyopic,hypermetrope,yes,reduced,nonepre-
presbyopic,hypermetrope,yes,normal,none
presbyopic,myope,no,reduced,none
presbyopic,myope,no,normal,none
presbyopic,myope,yes,reduced,none
presbyopic,myope,yes,normal,hard
presbyopic,hypermetrope,no,reduced,none
presbyopic,hypermetrope,no,normal,soft
presbyopic,hypermetrope,yes,reduced,none
presbyopic,hypermetrope,yes,normal,none

PROCEDURE:

Step1: Open the data file in Weka Explorer. It is presumed that the required data fields have been discretized. In this example it is age attribute.
Step2: Clicking on the associate tab will bring up the interface for association

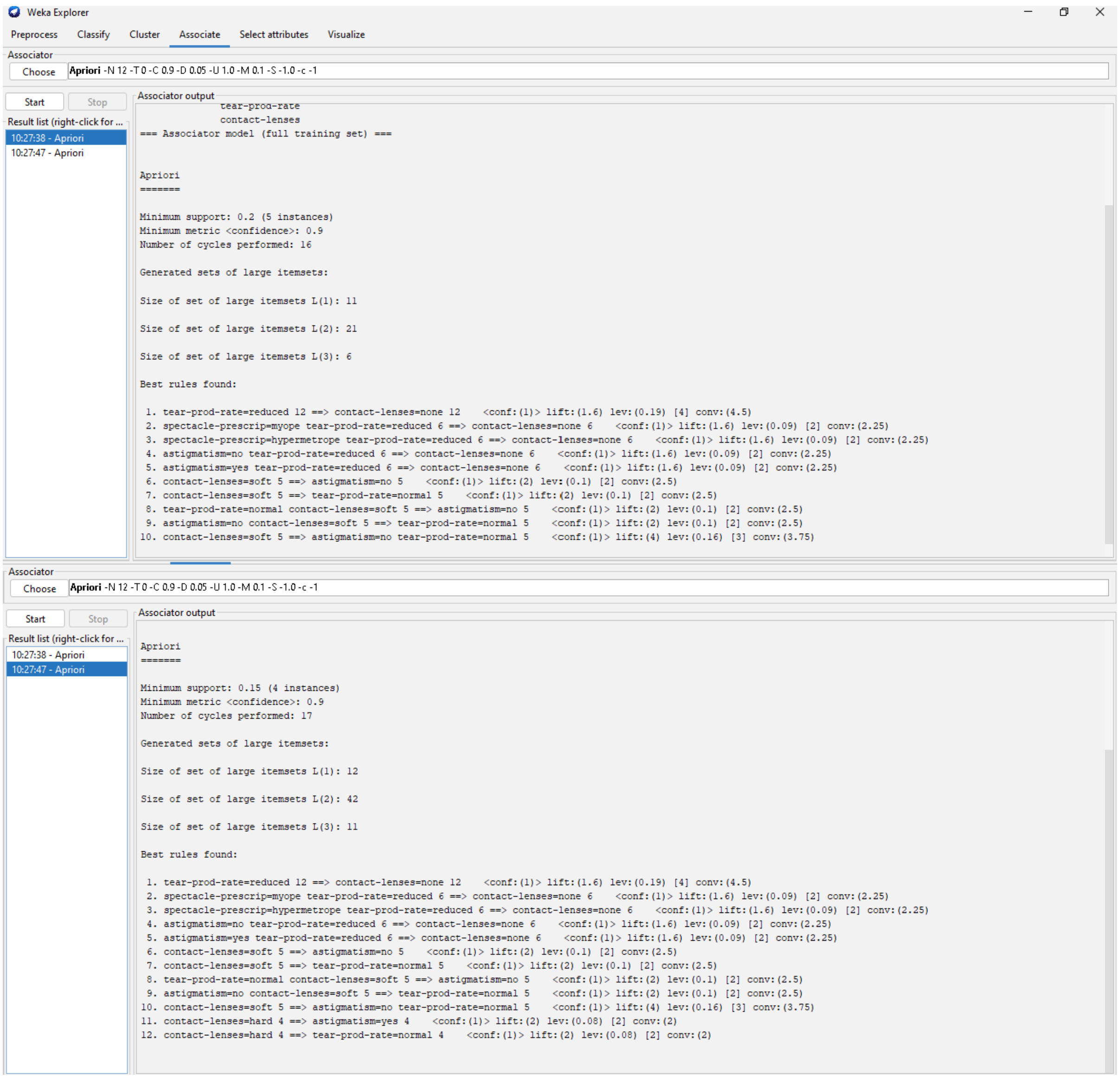
rule algorithm.
Step3: We will use apriori algorithm. This is the default algorithm.
Step4: Inorder to change the parameters for the run (example support, confidence etc) we click on the text box immediately to the right of the choose button.

Apriori algorithm:

Apriori Algorithm is a sequence of steps to be followed to find the most frequent itemset in the given database. This data mining technique follows the join and the prune steps iteratively until the most frequent itemset is achieved.

OUTPUT:

The following screenshot shows the association rules that were generated when apriori algorithm is applied on the given dataset



RESULT:

A demonstration of Association rule process on dataset contactlenses.arff usingapriori algorithm has been successfully completed.

EXPERIMENT – 4

AIM: Demonstration of classification rule process on dataset student.arff using j48 algorithm

Student.arff Dataset:

@relation student
@attribute age {<30,30-40,>40}
@attribute income {low, medium,
high}@attribute student {yes, no}
@attribute credit-rating {fair,
excellent}@attribute buyspc {yes, no}
@data
%
<30, high, no, fair, no
<30, high, no, excellent,
no30-40, high, no, fair, yes
>40, medium, no, fair, yes
>40, low, yes, fair, yes
>40, low, yes, excellent, no
30-40, low, yes, excellent, yes
<30, medium, no, fair, no
<30, low, yes, fair, no
>40, medium, yes, fair, yes
<30, medium, yes, excellent, yes
30-40, medium, no, excellent,
yes30-40, high, yes, fair, yes
>40, medium, no, excellent, no
%

PROCEDURE:

Step-1: We begin the experiment by loading the data (student.arff) into weka.
Step2: Next we select the “classify” tab and click “choose” button to select the“j48”classifier.
Step3: Now we specify the various parameters. These can be specified by clicking in the text box to the right of the chose button. In this example, we accept the default values. The default version does perform some pruning butdoes not perform error pruning.
Step4: Under the “text” options in the main panel. We select the 10-fold cross validation as our evaluation approach. Since we don’t have separate evaluation data set, this is necessary to get a reasonable idea of accuracy of generated model.
Step-5: We now click ”start” to generate the model .the Ascii version of the treeas

well as evaluation statistic will appear in the right panel when the model construction is complete.
Step-6: Note that the classification accuracy of model is about 69%.this indicates that we may find more work. (Either in preprocessing or in selecting current parameters for the classification)
Step-7: Now weka also lets us a view a graphical version of the classificationtree. This can be done by right clicking the last result set and selecting "visualize tree" from the pop-up menu.
Step-8: We will use our model to classify the new instances.
Step-9: In the main panel under "text" options click the "supplied test set" radiobutton and then click the "set" button. This wills pop-up a window which will allow you to open the file containing test instances

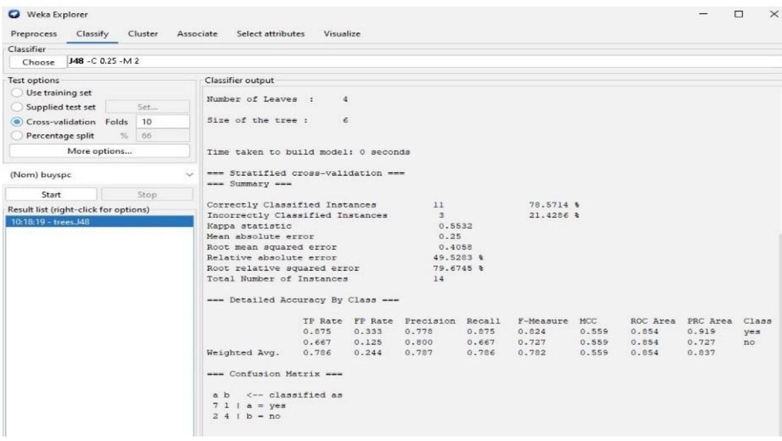
FILTERS USED :

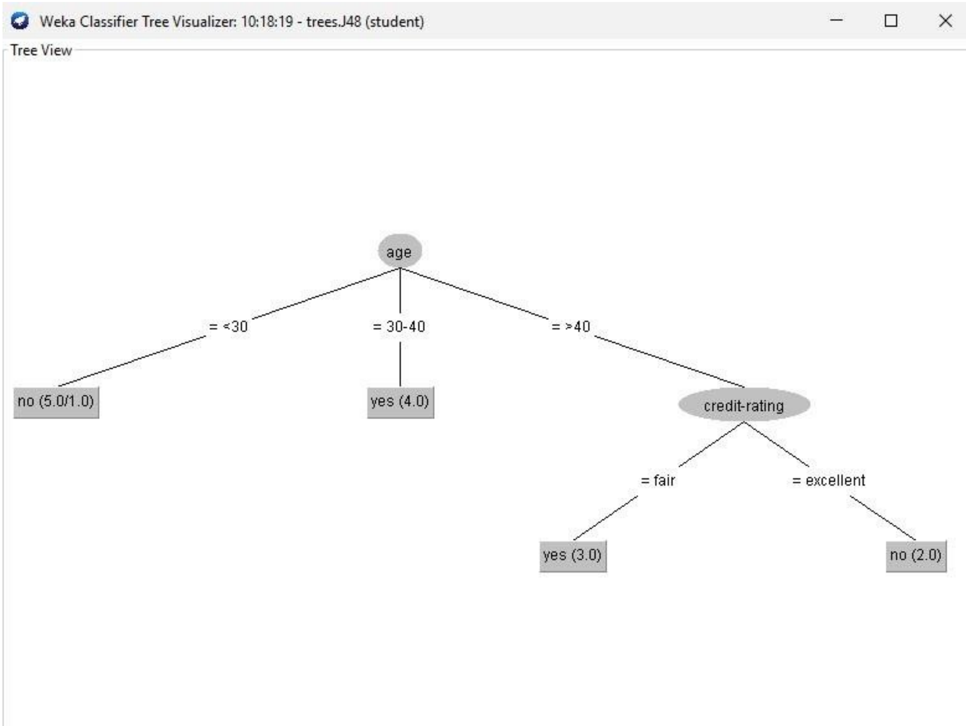
CLASSIFICATION J48 ALGORITHM :

J48 develops a decision node utilizing the expected estimations of the class. J48decision tree can deal with particular characteristics, lost or missing attribute estimations of the data and varying attribute costs. Here accuracy can be expanded by pruning

The Algorithm -
Stage 1: The leaf is labeled with a similar class if the instances belong to similar class.
Stage 2: For each attribute, the potential data will be figured and the gain in the data will be taken from the test on the attribute.
Stage 3: Finally the best attribute will be chosen depending upon the current selection parameter.

OUTPUT:





RESULT: The Demonstration of classification rule process on dataset student.arff using j48 algorithm has been done successfully.

EXPERIMENT – 5

AIM: Demonstration of classification rule process any dataset usingSMO algorithm.

Student.arff Dataset:

@relation student
@attribute age {<30,30-40,>40}
@attribute income {low, medium,
high}@attribute student {yes, no}
@attribute credit-rating {fair,
excellent}@attribute buyspc {yes, no}
@data
%
<30, high, no, fair, no
<30, high, no, excellent,
no30-40, high, no, fair, yes
>40, medium, no, fair, yes
>40, low, yes, fair, yes
>40, low, yes, excellent, no
30-40, low, yes, excellent, yes
<30, medium, no, fair, no
<30, low, yes, fair, no
>40, medium, yes, fair, yes
<30, medium, yes, excellent, yes
30-40, medium, no, excellent,
yes30-40, high, yes, fair, yes
>40, medium, no, excellent, no
%

PROCEDURE:

Step-1: We begin the experiment by loading the data (student.arff) into weka.
Step2: Next we select the "classify" tab and click "choose" button to select the "j48"classifier.
Step3: Now we specify the various parameters. These can be specified by clicking in the text box to the right of the chose button. In this example, we accept the default values. The default version does perform some pruning butdoes not perform error pruning.
Step4: Under the "text" options in the main panel. We select the 10-fold cross validation as our evaluation approach. Since we don't have separate evaluation data set, this is necessary to get a reasonable idea of accuracy of generated

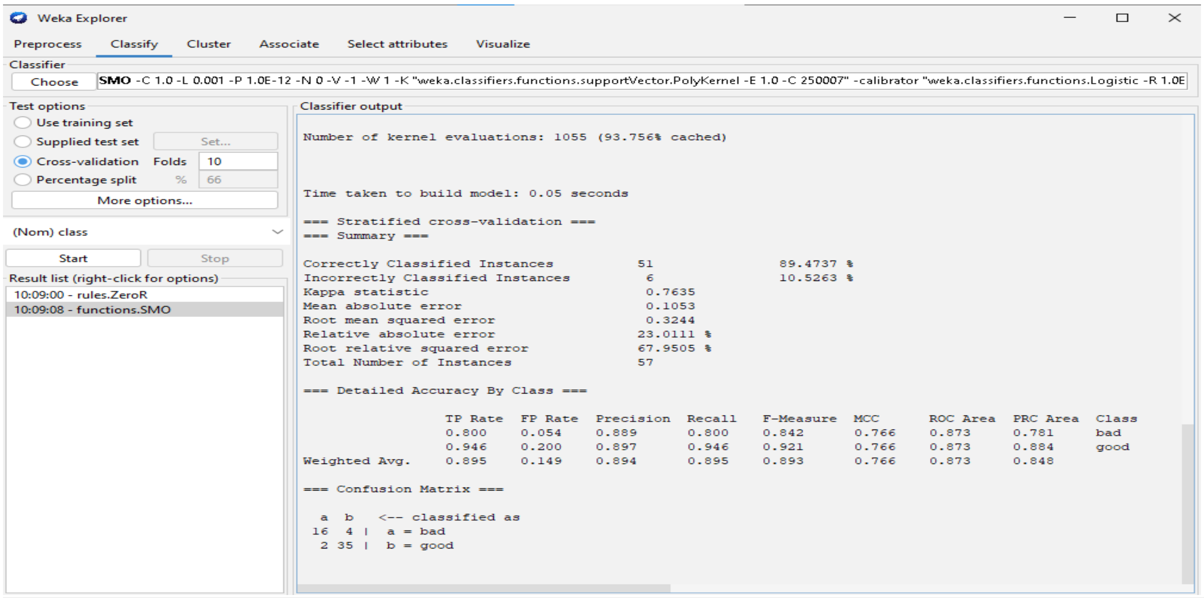
model.
Step-5: We now click ”start” to generate the model .the Ascii version of the treeas well as evaluation statistic will appear in the right panel when the model construction is complete.
Step-6: Note that the classification accuracy of model is about 69%.this indicates that we may find more work. (Either in preprocessing or in selecting current parameters for the classification)
Step-7: Now weka also lets us a view a graphical version of the classification tree. This can be done by right clicking the last result set and selecting “SMOagorithm” from the pop-up menu.
Step-8: We will use our model to classify the new instances.
Step-9: In the main panel under “text” options click the “supplied test set” radiobutton and then click the “set” button. This wills pop-up a window which will allow you to open the file containing test instances

FILTERS USED :

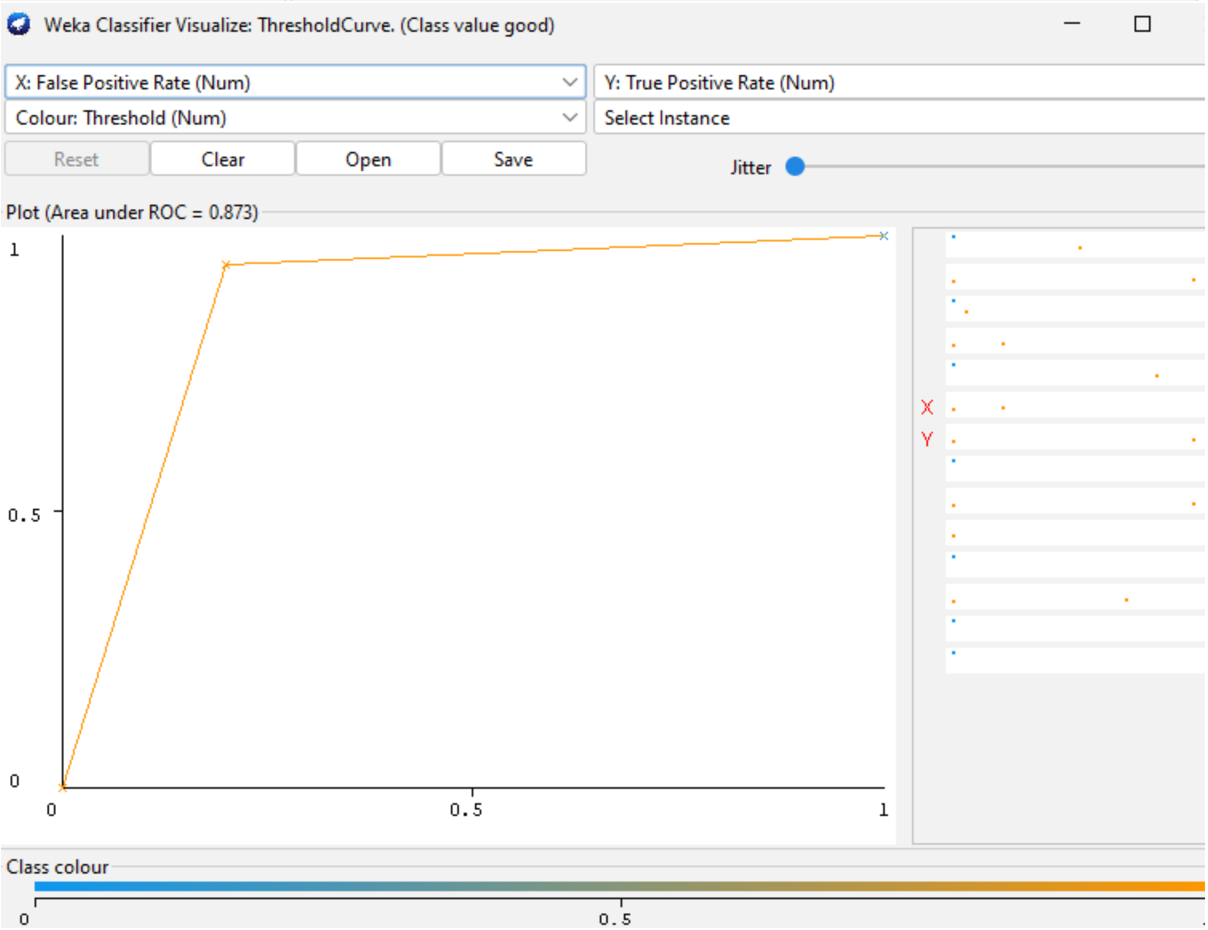
CLASSIFICATION OF SMO ALGORITHM:

*Sequential Minimal Optimization (SMO) is used for training a support vector classifier using polynomial or RBF kernels. It replaces all missing the values and transforms nominal attributes into binary ones. A single hidden layer neural network uses exactly the same form of model as an SVM.
*Training a Support Vector Machine (SVM) requires the solution of a very large quadratic programming (QP) optimization problem. SMO breaks this large QP problem into a series of smallest possible QP problems. These small QP problems are solved analytically, which avoids using a time-consuming numerical QP optimization as an inner loop.
*The amount of memory required for SMO is linear in the training set size, which allows SMO to handle very large training sets. Because large matrix computation is avoided, SMO scales somewhere between linear and quadratic in the training set size for various test problems.

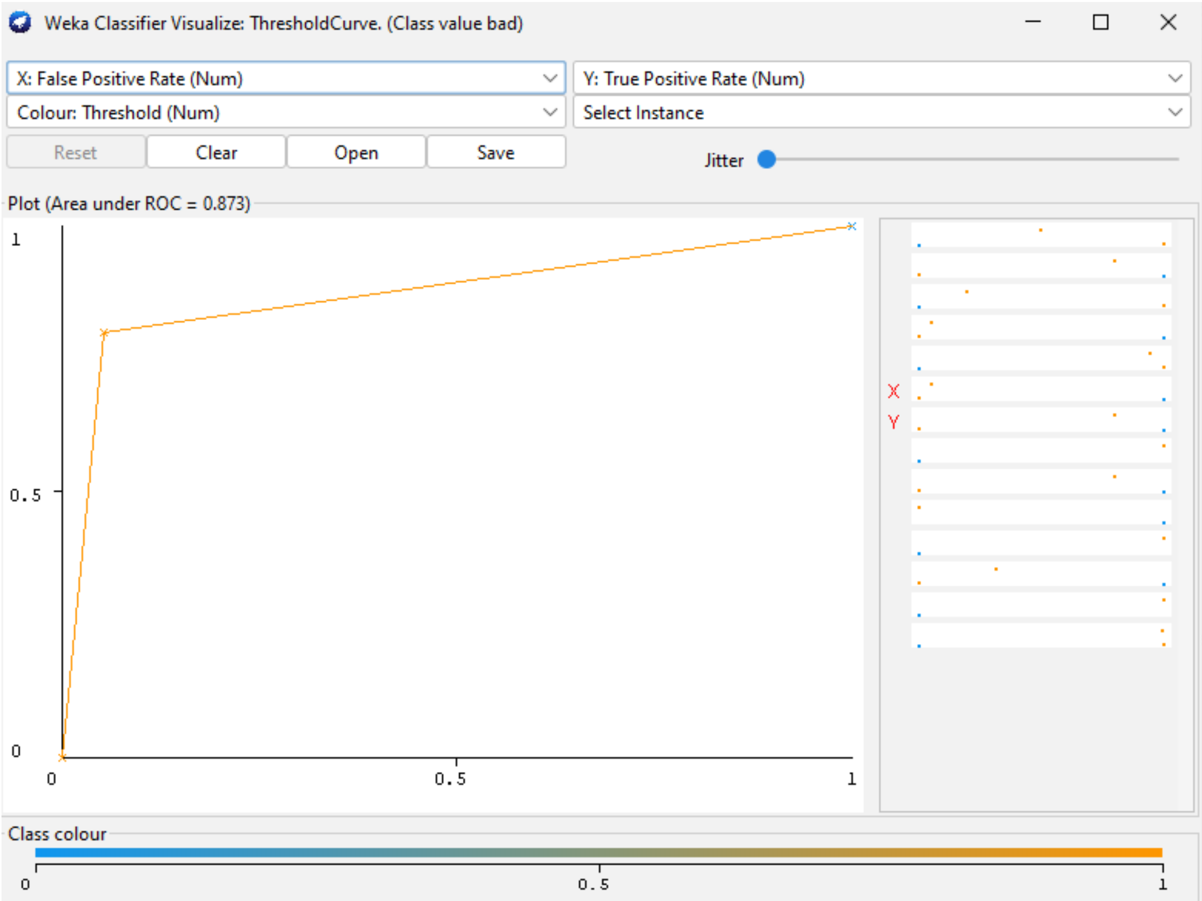
OUTPUT:



Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
Choose
SMO -C 1.0 -L 0.001 -P 1.0E-12 -N 0 -V -1 -W 1 -K "weka.classifiers.functions.supportVector.PolyKernel -E 1.0 -C 250007" -calibrator "weka.classifiers.functions.Logistic -R 1.0E
Test options
Use training set
Supplied test set
Set...
Cross-validation
Folds
10
Percentage split
%
66
More options...
(Nom) class
Start
Stop
Result list (right-click for options)
10:09:00 - rules.ZeroR
10:09:08 - functions.SMO
Classifier output
Number of kernel evaluations: 1055 (93.756% cached)
Time taken to build model: 0.05 seconds
=== Stratified cross-validation ===
=== Summary ===
Correctly Classified Instances 51 89.4737 %
Incorrectly Classified Instances 6 10.5263 %
Kappa statistic 0.7635
Mean absolute error 0.1053
Root mean squared error 0.3244
Relative absolute error 23.0111 %
Root relative squared error 67.9505 %
Total Number of Instances 57
=== Detailed Accuracy By Class ===
TP Rate FP Rate Precision Recall F-Measure MCC ROC Area PRC Area Class
0.800 0.054 0.889 0.800 0.842 0.766 0.873 0.781 bad
0.946 0.200 0.897 0.946 0.921 0.766 0.873 0.884 good
Weighted Avg. 0.895 0.149 0.894 0.895 0.893 0.766 0.873 0.848
=== Confusion Matrix ===
a b <-- classified as
16 4 | a = bad
2 35 | b = good



Weka Classifier Visualize: ThresholdCurve. (Class value good)
X: False Positive Rate (Num)
Y: True Positive Rate (Num)
Colour: Threshold (Num)
Select Instance
Reset
Clear
Open
Save
Jitter
Plot (Area under ROC = 0.873)
1
0.5
0
0
0.5
1
X
Y
Class colour
0
0.5



RESULT: The Demonstration of classification rule process on dataset student.arff using SMO algorithm has been done successfully.

EXPERIMENT – 6

AIM: Demonstration of classification rule process on dataset employee.arff using naive bayes algorithm

Employee.arff Dataset:

@relation employee
@attribute age {25, 27, 28, 29, 30, 35, 48}
@attribute
salary{10k,15k,17k,20k,25k,30k,35k,32k}@attribute
performance {good, avg, poor}
@data
%
25, 10k, poor
27, 15k, poor
27, 17k, poor
28, 17k, poor
29, 20k, avg
30, 25k, avg
29, 25k, avg
30, 20k, avg
35, 32k, good
48, 34k, good
48, 32k, good
%

PROCEDURE:

Step 1. We begin the experiment by loading the data (employee.arff) into weka.Step2: next we select the "classify" tab and click "choose" button to select the "Naïve Bayes"classifier.
Step3: now we specify the various parameters. These can be specified by clicking in the text box to the right of the chose button. In this example, we accept the default values his default version does perform some pruning but does not perform error pruning.
Step4: under the "text "options in the main panel. We select the 10-fold cross validation as our evaluation approach. Since we don't have separate evaluationdata set, this is necessary to get a reasonable idea of accuracy of generated model.
Step-5: we now click"start"to generate the model .the ASCII version of the tree as well as evaluation statistic will appear in the right panel when the model

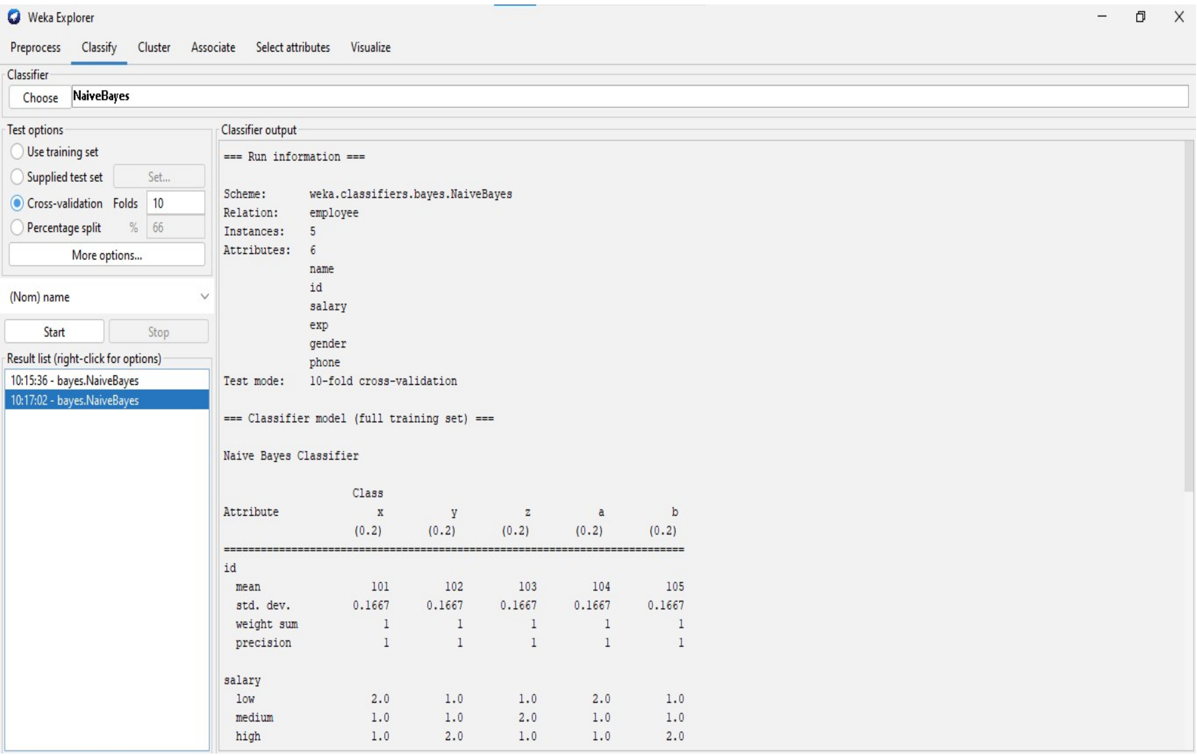
construction is complete.
Step-6: note that the classification accuracy of model is about 69%.this indicates that we may find more work. (Either in preprocessing or in selecting current parameters for the classification)
Step-7: now weka also lets us a view a graphical version of the classificationtree. This can be done by right clicking the last result set and selecting "visualize tree" from the pop-up menu.
Step-8: we will use our model to classify the new instances.
Step-9: In the main panel under "text "options click the "supplied test set" radiobutton and then click the "set" button. This will show pop-up window which will allow you to open the file containing test instances.

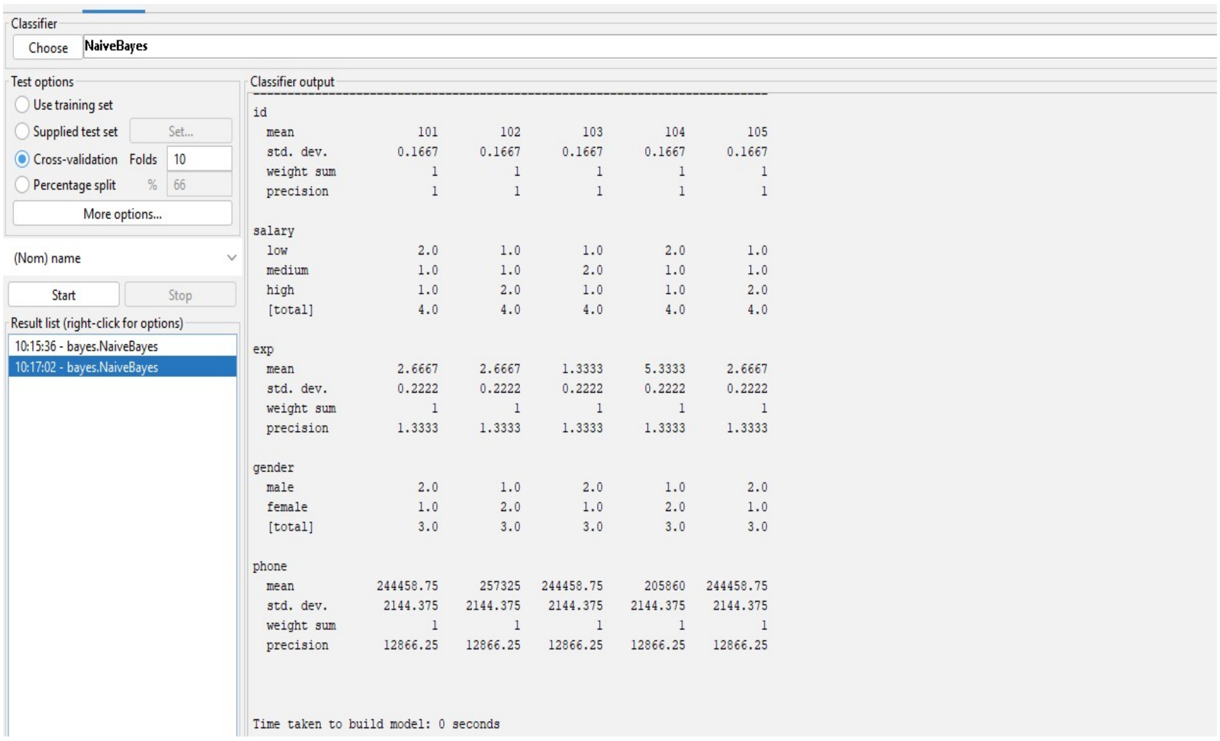
FILTERS USED :

NAÏVE BAYES Algorithm :

- Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.
- It is mainly used in *text classification* that includes a high-dimensional training dataset.
- Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.
- It is a probabilistic classifier, which means it predicts on the basisof the probability of an object.
- Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

OUTPUT :





RESULT : Demonstration of classification rule process on dataset employee.arff using naive bayes algorithm has been done successfully.

EXPERIMENT - 7

AIM: Demonstration of clustering rule process on dataset iris.arffusing simple k-means

PROCEDURE:

Step 1: Run the Weka explorer and load the data file iris.arff in preprocessing interface.
Step 2: Inorder to perform clustering select the ‘cluster’ tab in the explorer and click on the choose button. This step results in a dropdown list of available clustering algorithms.
Step 3 : In this case we select ‘simple k-means’.
Step 4: Next click in text button to the right of the choose button to get popup window shown in the screenshots. In this window we enter six on the number of clusters and we leave the value of the seed on as it is. The seed value is used in generating a random number which is used for making the internal assignments of instances of clusters.
Step 5 : Once of the option have been specified. We run the clustering algorithm there we must make sure that they are in the ‘cluster mode’ panel. The use of training set option is selected and then we click ‘start’ button. This process and resulting window are shown in the following screenshots.
Step 6 : The result window shows the centroid of each cluster as well as statistics on the number and the percent of instances assigned to different clusters. Here clusters centroid are means vectors for each clusters. This clusterscan be used to characterized the cluster.For eg, the centroid of cluster1 shows the class iris.versicolor mean value of the sepal length is 5.4706, sepal width 2.4765, petal width 1.1294, petal length 3.7941.
Step 7: Another way of understanding characterstics of each cluster through visualization ,we can do this, try right clicking the result set on the result. List panel and selecting the visualize cluster assignments.

Interpretation of the above visualization

From the above visualization, we can understand the distribution of sepal lengthand petal length in each cluster. For instance, for each cluster is dominated by petal length. In this case by changing the color dimension to other attributes we can see their distribution with in each of the cluster.
Step 8: We can assure that resulting dataset which included each instance alongwith its assign cluster. To do so we click the save button in the visualization window and save the result iris k-mean .

.

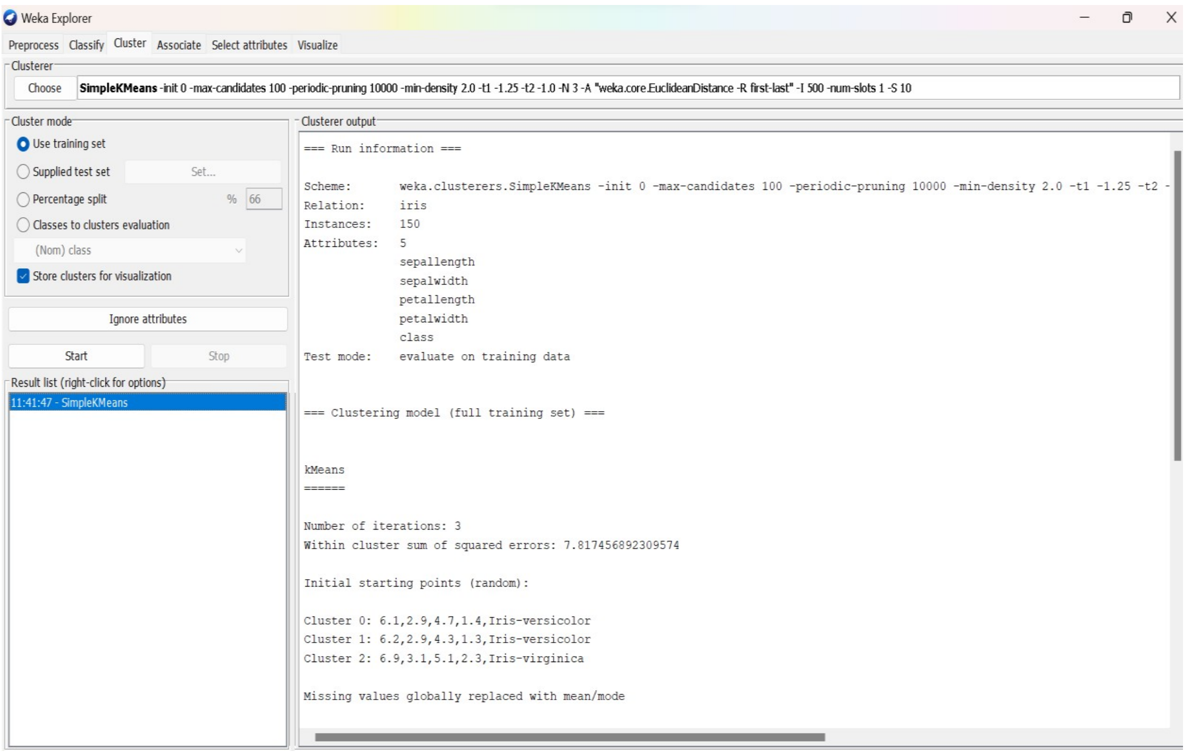
FILTERS USED:

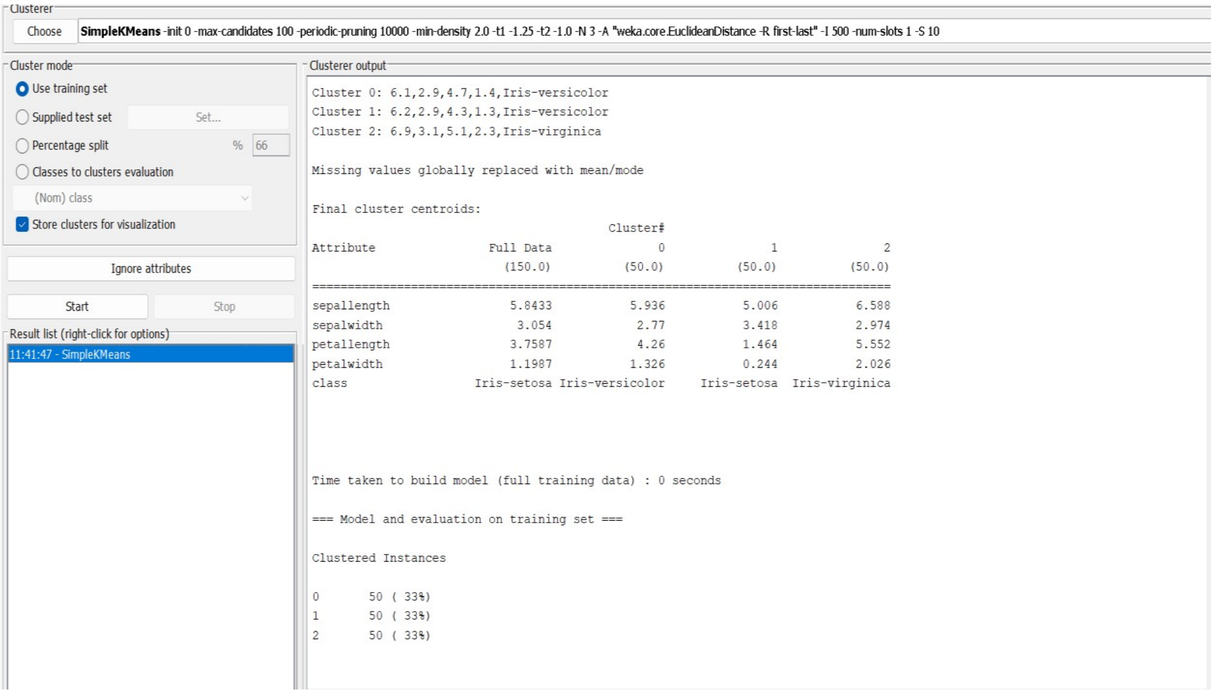
Simple K-Means Algorithm:

K-means clustering is a simple unsupervised learning algorithm. In this, the data objects ('n') are grouped into a total of 'k' clusters, with each observation belonging to the cluster with the closest mean. It defines 'k' sets, one for each cluster k n (the point can be thought of as the center of a one or two-dimensional figure). The clusters are separated by a large distance.
The data is then organized into acceptable data sets and linked to the nearest collection. If no data is pending, the first stage is more difficult to complete; in this case, an early grouping is performed. The 'k' new set must be recalculated as the barycenters of the clusters from the previous stage.

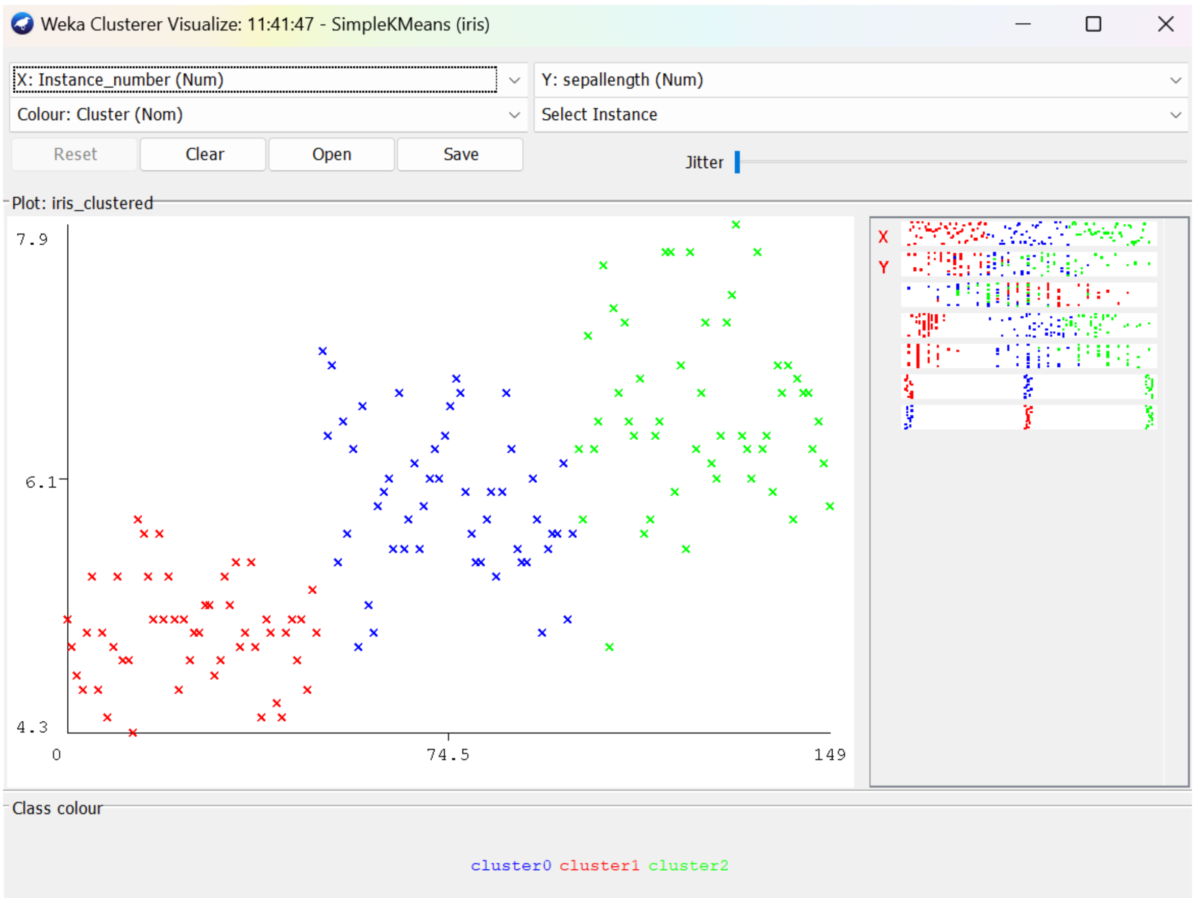
The same data set points and the nearest new sets are bound together after these 'k' new sets have been created. After that, a loop is created. The 'k' sets change their position step by step until no further changes are made as a resultof this loop.

OUTPUT:





VISUALISATION:



RESULT: Demonstration of clustering rule process on dataset iris.arff using simple k-means has been completed successfully.

EXPERIMENT – 8

AIM: This experiment illustrates demonstration of the classification rule process using the Logistic Regression algorithm for the classifier in weka. Thesample data set used in this experiment is “student” data available in arff format.

PROCEDURE:

Steps involved in this experiment:

Step 1: We begin the experiment by loading the data (student. arff) into weka.

Step 2: Next we select the “classify” tab and click the “choose” button to select the “Logistic” classifier.

Step 3: Now we specify the various parameters. These can be specified by clicking in the text box to the right of the choose button. In this example, we accept the default values. The default version does perform some pruning butdoes not perform error pruning.

Step 4: Under the “text” options in the main panel. We select the 10-fold cross-validation as our evaluation approach. Since we don’t have a separate evaluation data set, this is necessary to get a reasonable idea of the accuracy of the generated model.

Step 5: We now click ”start” to generate the model. The evaluation statistic will appear in the right panel when the model construction is complete.

Step 6: Note that the classification accuracy of the model is about 80%.this indicates that we may find more work. (Either in preprocessing or in selecting current parameters for the classification)

Step 7: We will use our model to classify the new instances.

Step 8: In the main panel under “text” options click the “supplied test set” radio button and then click the “set” button. These wills pop up a window that will allow you to open the file containing test instances.

DATASET:

student. arff

```
@relation student
@attribute age {<30,30-40,>40}
@attribute income {low, medium,
high}@attribute student {yes, no}
@attribute credit-rating {fair,
excellent}@attribute buyspc {yes, no}
@data
%
<30, high, no, fair, no
<30, high, no, excellent,
no30-40, high, no, fair, yes
>40, medium, no, fair, yes
>40, low, yes, fair, yes
>40, low, yes, excellent, no
30-40, low, yes, excellent, yes
<30, medium, no, fair, no
<30, low, yes, fair, no
>40, medium, yes, fair, yes
<30, medium, yes, excellent, yes
30-40, medium, no, excellent,
yes30-40, high, yes, fair, yes
>40, medium, no, excellent, no
%
```

FILTER USED:

LOGISTIC REGRESSION –

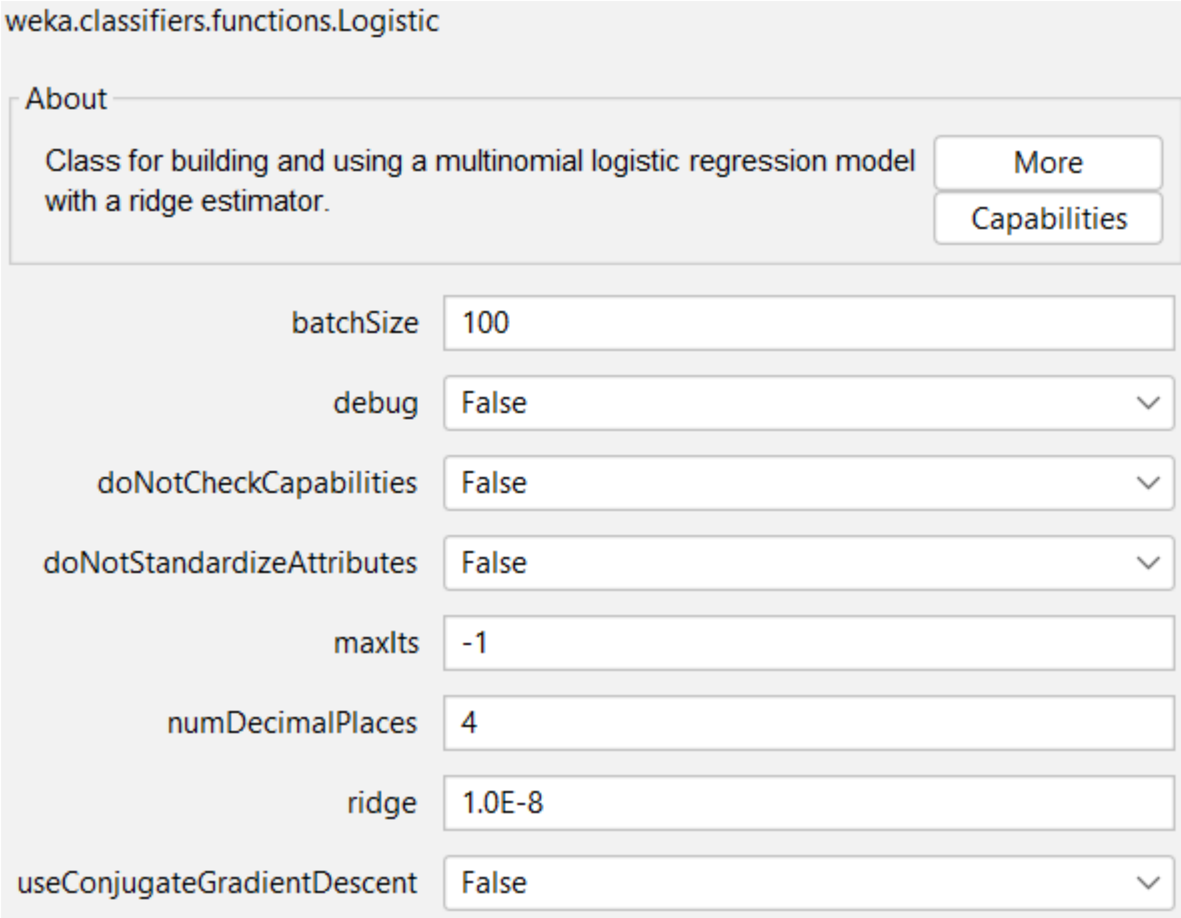
This type of statistical model (also known as logit model) is often used for classification and predictive analytics. Logistic regression estimates the probability of an event occurring, such as voted or didn't vote, based on a given dataset of independent variables.

The following screenshot shows the classification rules that were generated when the logistic regression algorithm is applied to the given dataset.

Classifier output

```
=== Run information ===

Scheme:       weka.classifiers.functions.Logistic -R 1.0E-8 -M -1 -num-decimal-places 4
Relation:     student
Instances:    14
Attributes:   5
              age
              income
              student
              credit-rating
              buyspc
Test mode:    10-fold cross-validation

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

Logistic Regression with ridge parameter of 1.0E-8
Coefficients...
                                         Class
Variable                                   yes
===============================================
age=<30                               -28.8116
age=30-40                              32.0198
age=>40                                 0.3495
income=low                            -32.8936
income=medium                          29.0923
income=high                            -2.0173
student=no                            -63.3112
credit-rating=excellent               -30.8618
Intercept                              48.0599


Odds Ratios...
                                         Class
Variable                                   yes
===============================================
age=<30                                      0
age=30-40                 8.054031041279456E13
```

```
===============================================
age=<30                                      0
age=30-40                 8.054031041279456E13
age=>40                                 1.4184
income=low                                   0
income=medium             4.311636736716433E12
income=high                              0.133
student=no                                   0
credit-rating=excellent                      0


Time taken to build model: 0.02 seconds

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

Correctly Classified Instances          12               85.7143 %
Incorrectly Classified Instances         2               14.2857 %
Kappa statistic                          0.72
Mean absolute error                      0.1431
Root mean squared error                  0.3768
Relative absolute error                 28.3564 %
Root relative squared error             73.9849 %
Total Number of Instances               14

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.750    0.000    1.000      0.750    0.857      0.750    0.896     0.952     yes
                 1.000    0.250    0.750      1.000    0.857      0.750    0.917     0.815     no
Weighted Avg.    0.857    0.107    0.893      0.857    0.857      0.750    0.905     0.893

=== Confusion Matrix ===

 a b   <-- classified as
 6 2 | a = yes
 0 6 | b = no
```



RESULT: Hence Classification Rule processed on the student dataset is completed.

EXPERIMENT – 9

AIM: This experiment illustrates the Implementation of Non-Linear Regression in weka. The sample data set used in this experiment is “CPU” dataavailable at arff format. This document assumes that appropriate data pre-processing has been performed.

Dataset CPU.arff:

@relation 'cpu'
@attribute vendor { adviser, amdahl, apollo, basf, bti, burroughs, c.r.d, cdc, cambex, dec, dg,formation, four-phase, gould, hp, harris, honeywell, ibm, ipl, magnuson, microdata, nas, ncr, nixdorf, perkin-elmer, prime, siemens, sperry, sratus, wang}
@attribute MYCT real
@attribute MMIN real
@attribute MMAX real
@attribute CACH real
@attribute CHMIN real
@attribute CHMAX
real@attribute class real
@data
adviser,125,256,6000,256,16,128,199
amdahl,29,8000,32000,32,8,32,253
amdahl,29,8000,32000,32,8,32,253
amdahl,29,8000,32000,32,8,32,253
amdahl,29,8000,16000,32,8,16,132
amdahl,26,8000,32000,64,8,32,290
amdahl,23,16000,32000,64,16,32,381
amdahl,23,16000,32000,64,16,32,381
amdahl,23,16000,64000,64,16,32,749
amdahl,23,32000,64000,128,32,64,123
8apollo,400,1000,3000,0,1,2,23
apollo,400,512,3500,4,1,6,24
basf,60,2000,8000,65,1,8,70
basf,50,4000,16000,65,1,8,117
bti,350,64,64,0,1,4,15
bti,200,512,16000,0,4,32,64
burroughs,167,524,2000,8,4,15,23
burroughs,143,512,5000,0,7,32,29

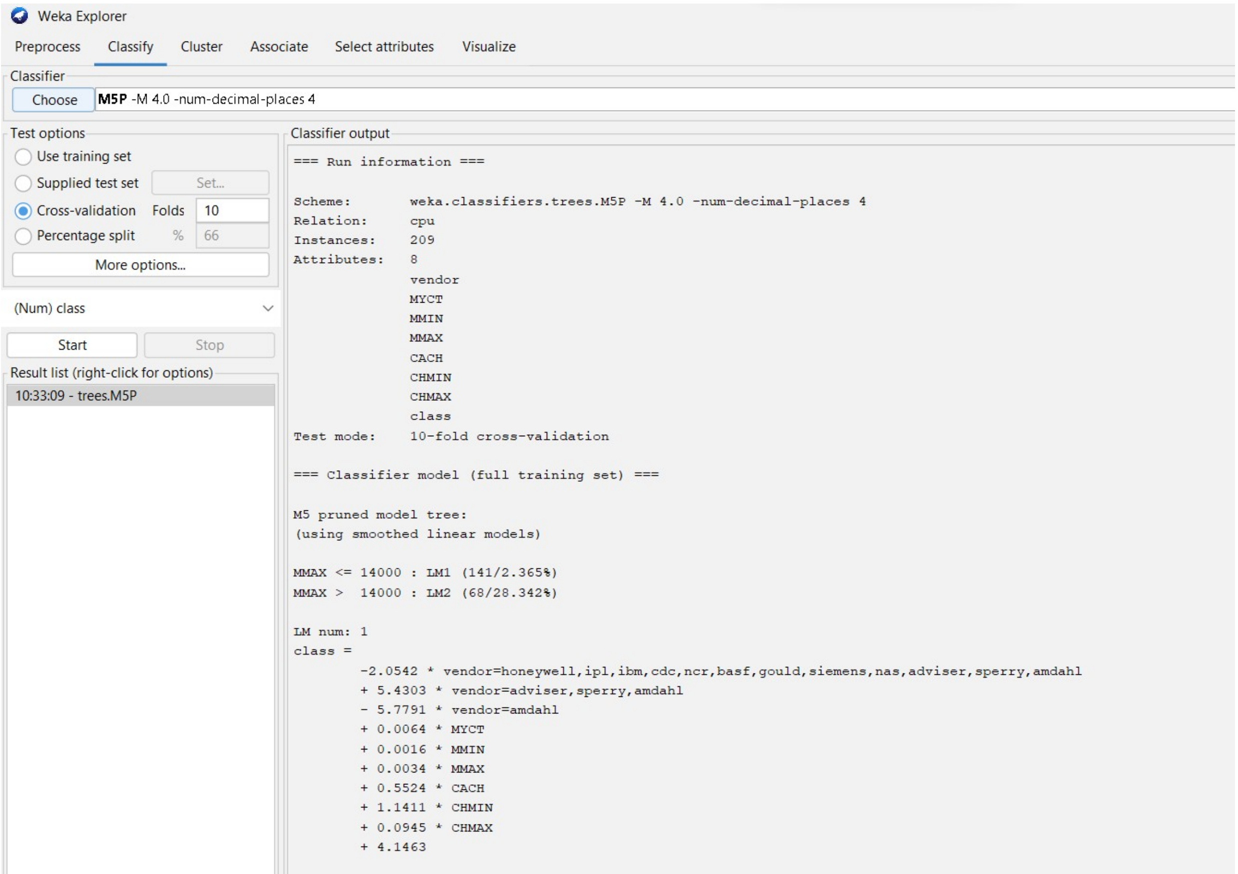
PROCEDURE:

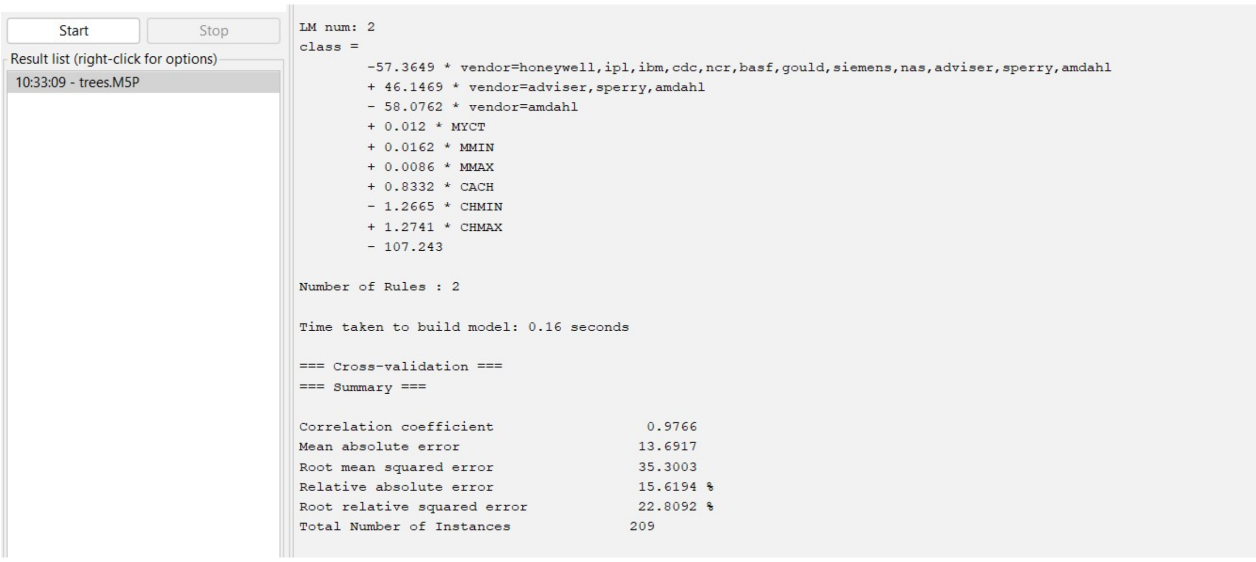
Step-1: We begin the experiment by loading the data (CPU.arff)into weka.
Step2: Next we select the “classify” tab and click “choose” button to select the “Tree” classifier under which we select “M5P”.

Step3: Now we specify the various parameters. These can be specified by clicking in the text box to the right of the chose button. In this example, we accept the default values. The default version does perform some pruning butdoes not perform error pruning.
Step4: Under the "text" options in the main panel. We select the 10-fold cross validation as our evaluation approach. Since we don't have separate evaluationdata set, this is necessary to get a reasonable idea of accuracy of generated model.
Step-5: We now click "start" to generate the model. the Ascii version of the tree as well as evaluation statistic will appear in the right panel when the model construction is complete.
Step-6: We will use our model for performing non-linear regression the newinstances.

OUTPUT:

The following screenshot shows the classifier output that were generated whennon-linear regression algorithm is applied on the given dataset.





RESULT: We have successfully implemented non-linear regression algorithm in weka.

EXPERIMENT – 10

AIM: This experiment illustrates the Implementation of Simple Linear Regression in weka. The sample data set used in this experiment is “CPU” data available at arff format. This document assumes that appropriate data pre-processing has been performed.

Dataset CPU.arff:

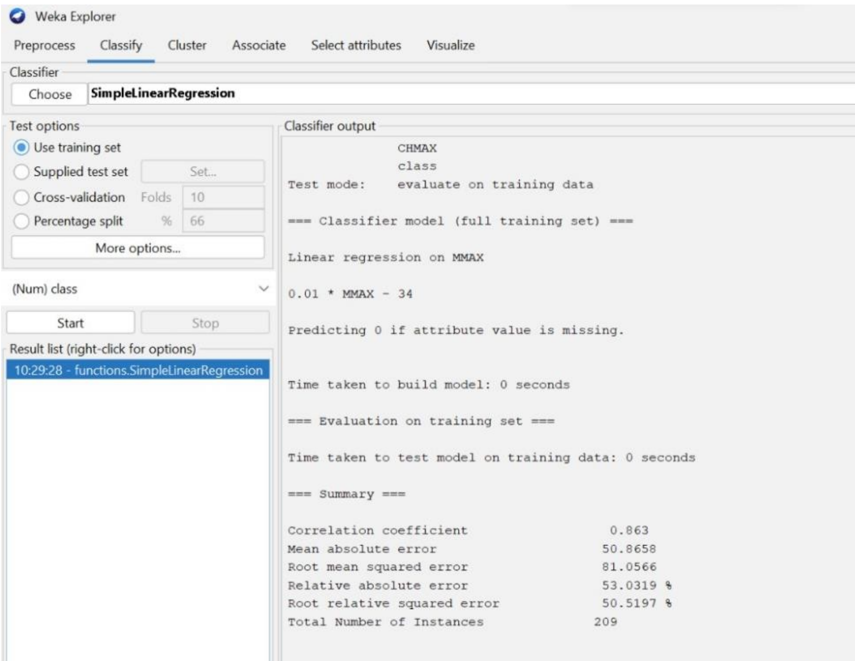
@relation 'cpu'
@attribute vendor { adviser, amdahl, apollo, basf, bti, burroughs, c.r.d, cdc, cambex, dec, dg,formation, four-phase, gould, hp, harris, honeywell, ibm, ipl, magnuson, microdata, nas, ncr, nixdorf, perkin-elmer, prime, siemens, sperry, sratus, wang}
@attribute MYCT real
@attribute MMIN real
@attribute MMAX real
@attribute CACH real
@attribute CHMIN real
@attribute CHMAX
real@attribute class real
@data
adviser,125,256,6000,256,16,128,199
amdahl,29,8000,32000,32,8,32,253
amdahl,29,8000,32000,32,8,32,253
amdahl,29,8000,32000,32,8,32,253
amdahl,29,8000,16000,32,8,16,132
amdahl,26,8000,32000,64,8,32,290
amdahl,23,16000,32000,64,16,32,381
amdahl,23,16000,32000,64,16,32,381
amdahl,23,16000,64000,64,16,32,749
amdahl,23,32000,64000,128,32,64,123
8apollo,400,1000,3000,0,1,2,23
apollo,400,512,3500,4,1,6,24
basf,60,2000,8000,65,1,8,70
basf,50,4000,16000,65,1,8,117
bti,350,64,64,0,1,4,15
bti,200,512,16000,0,4,32,64
burroughs,167,524,2000,8,4,15,23
burroughs,143,512,5000,0,7,32,29

PROCEDURE:

Step-1: We begin the experiment by loading the data (CPU.arff) into weka.
Step2: Next we select the "classify" tab and click "choose" button to select the "Functions" classifier under which we select "SimpleLinearRegression".
Step3: Now we specify the various parameters. These can be specified by clicking in the text box to the right of the chose button. In this example, we accept the default values. The default version does perform some pruning butdoes not perform error pruning.
Step4: Under the "text" options in the main panel. We select the 10-fold cross validation as our evaluation approach. Since we don't have separate evaluation data set, this is necessary to get a reasonable idea of accuracy of generated model.
Step-5: We now click "start" to generate the model. the Ascii version of the tree as well as evaluation statistic will appear in the right panel when the model construction is complete.
Step-6: We will use our model for performing Simple Linear Regression the new instances.

OUTPUT:

The following screenshot shows the classifier output that were generated when Simple linear regression algorithm is applied on the given dataset.



RESULT: We have successfully implemented Simple Linear Regression algorithm in weka.