BVRIT HYDERABAD
College of Engineering for Women
Department of Computer Science & Engineering

DATA MINING LAB

IV Year – I Semester
(R16 Regulation)
Academic Year: 2020-21

BVRIT HYDERABAD

College of Engineering for Women

Rajiv Gandhi Nagar, Bachupally, Hyderabad -90

Introduction To WEKA

WEKA - Waikato **E**nvironment for **K**nowledge **A**nalysis

- A collection of open source of many data mining and machine learning algorithms, including
  - pre-processing on data
  - Classification
  - clustering
  - Association rule extraction
- It's a data mining/machine learning tool developed by Department of Computer Science, University of Waikato, New Zealand by Ian H. Witten , Eibe Frank and Mark A. Hall

- WEKA is a state-of-the-art facility for developing Machine Learning Techniques and their application to real-world data mining problems.

- Java based (also open source) **issued** under the GNU General Public License.
- Used for Research, Education, and Applications.
- It can be run on **Windows, Linux and Mac.**
- Weka is also a bird found only on the islands of New Zealand, is its symbol.

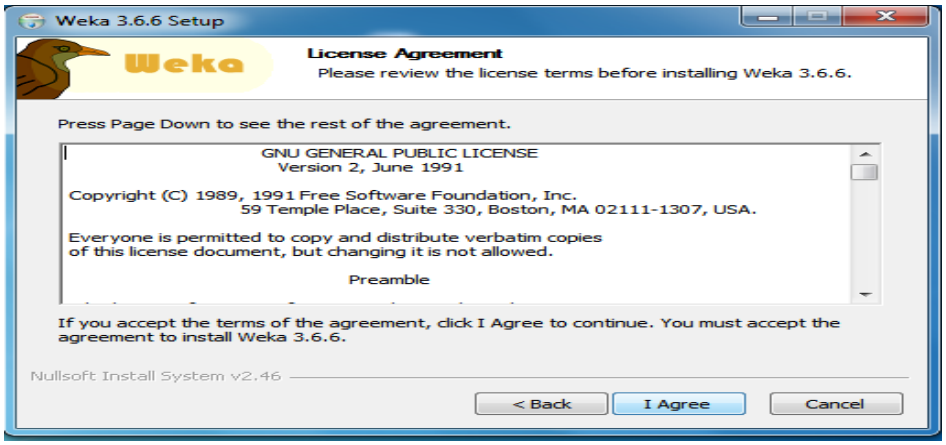
Weka Main Features:

- 49 data preprocessing tools
- 76 classification/regression algorithms
- 8 clustering algorithms
- 3 algorithms for finding association rules
- 15 attribute/subset evaluators
- 10 search algorithms for feature selection
- 3 algorithms for finding association rules.

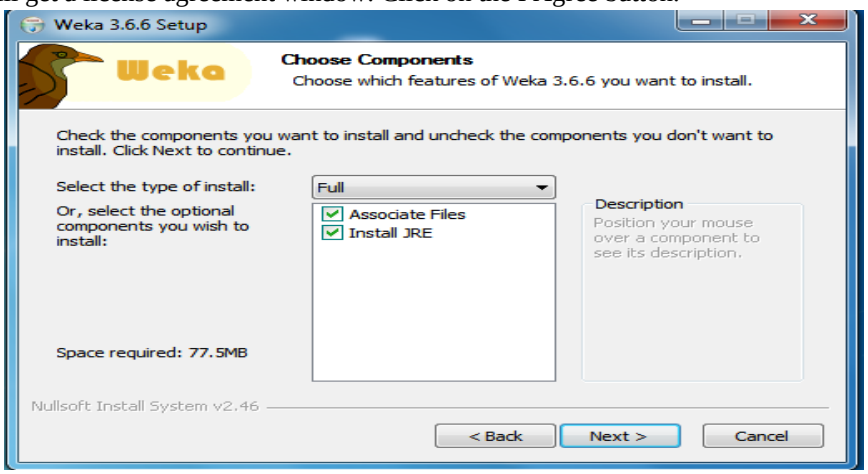
i) Downloading and/or installing of WEKA data mining toolkit.

- Download Weka (the stable version) from http://www.cs.waikato.ac.nz/ml/weka/
- Choose a self-extracting executable (including Java VM)
- (If you are interested in modifying/extending weka there is a developer version that includes the source code)
- After download is completed, run the self- extracting file to install Weka, and use the default set-ups.

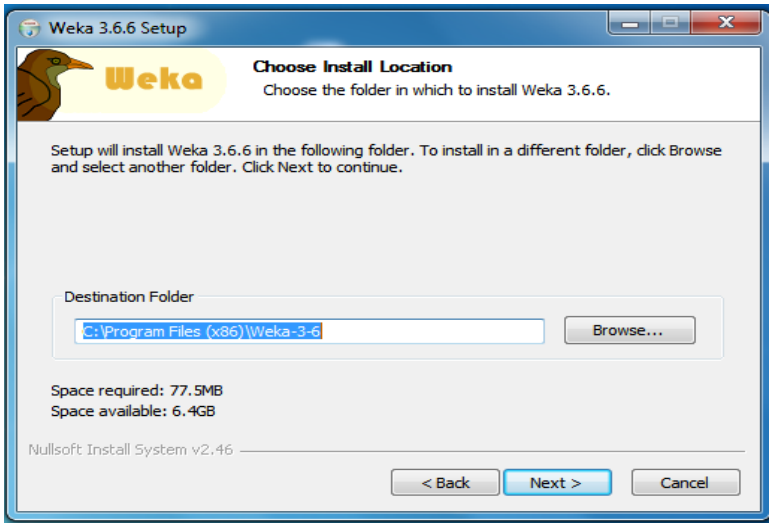
Step-1: Double click on the weka .exe file. We will then get a weka setup wizard window. Click on the next button.



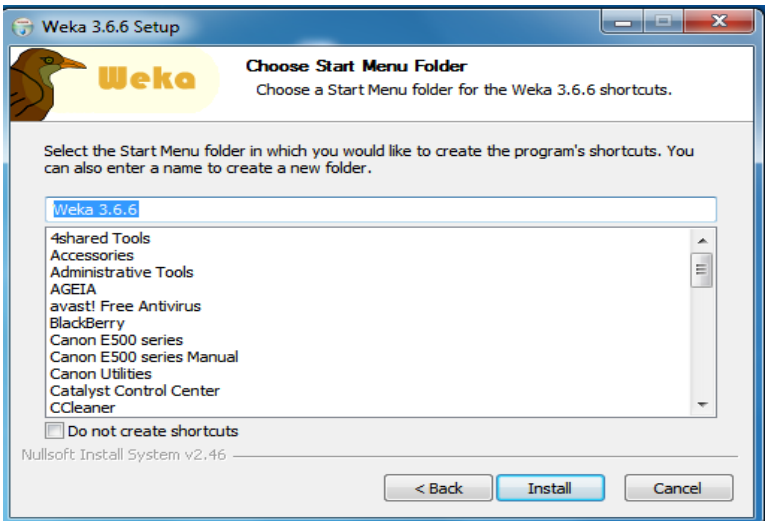
Step-2: Now we will get a license agreement window. Click on the I Agree button.



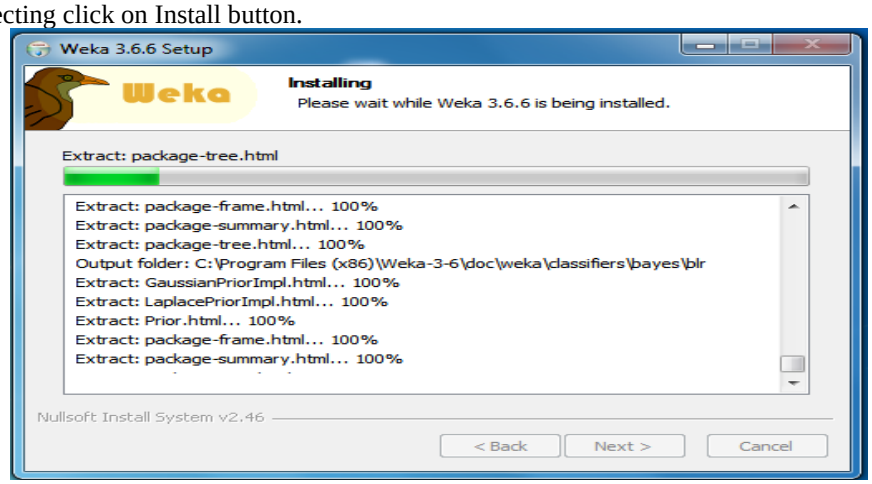
Step-3: Now we will get a choose components window. Select the Associate Files and Install JRE checkboxes and then click on the next button.



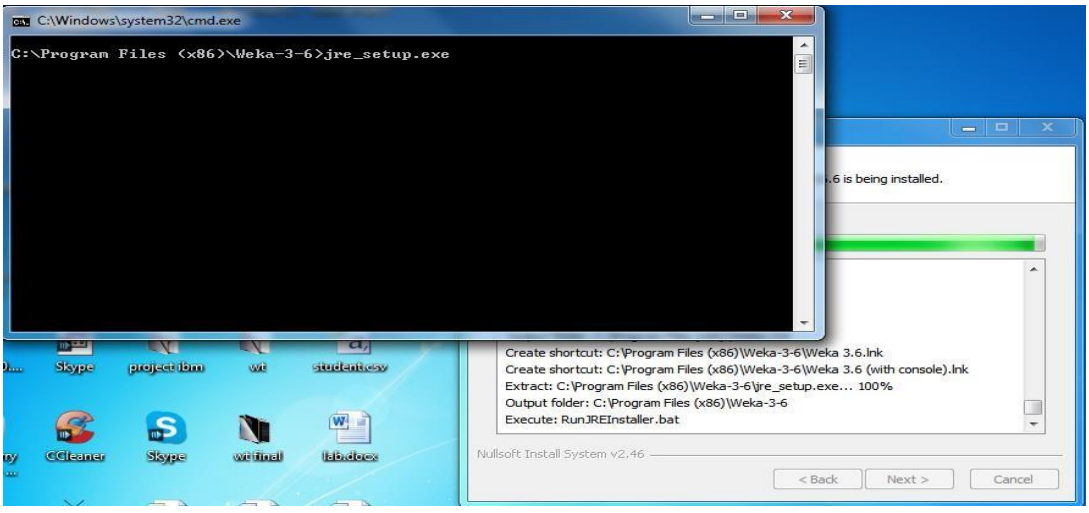
Step-4: Now we will get a choose install location window where we have to give the destination folder address for the installation of the weka software. After selecting the destination folder, click on the next button.



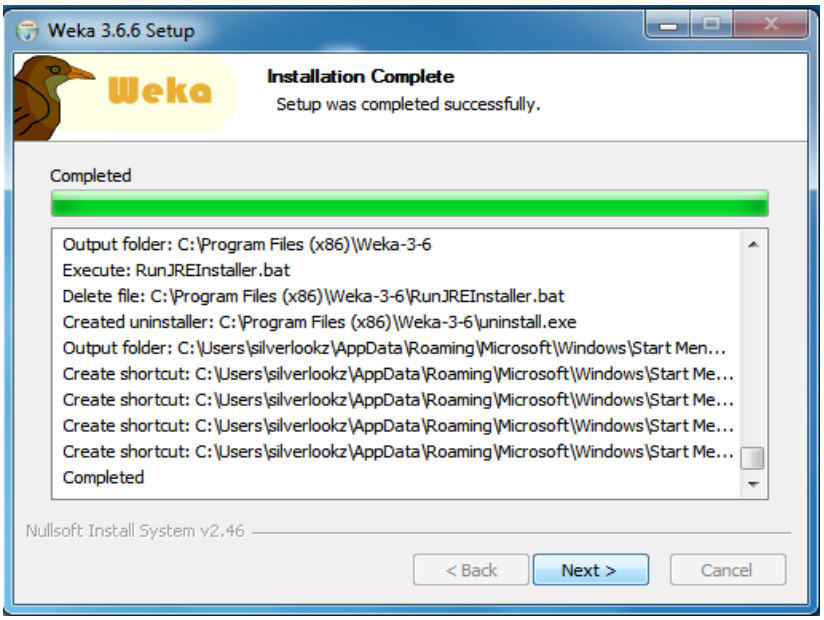
Step-5: Now we will get a choose start menu folder window where we should select the start menu folder for creating the shortcut for the weka software. We can also select the do not create shortcuts option if you wish to not have any shortcut. After selecting click on Install button.



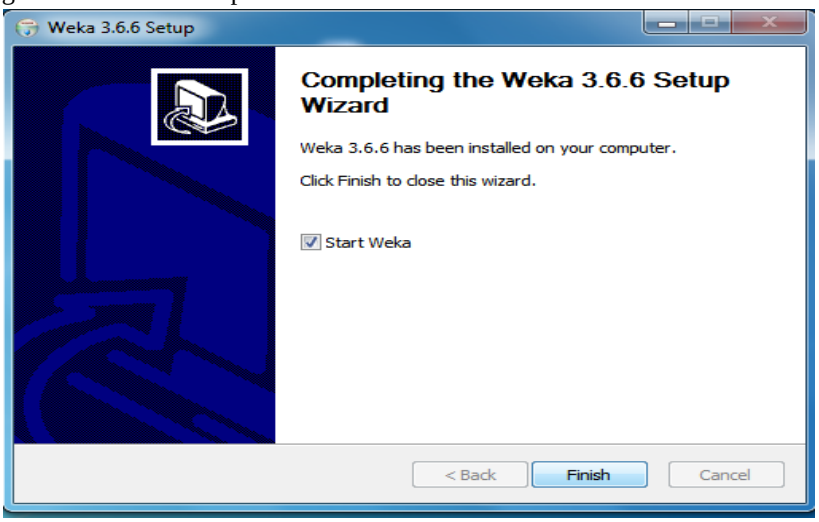
Now we will get a installing window where all the required packages and files are installed.



Now we will get a command prompt window as part of the installation process.



Step-6: Now we will get a installation complete window. Click on the next button.



Step-7: Now we will get a completing the weka setup window. This indicates the successful installation of the weka software. Now click on the finish button.

Step-8: Now we will get a weka GUI chooser with four applications here: Explorer, Experimenter, KnowledgeFlow, and simple CLI.

Launching WEKA Explorer:

1. You can launch Weka from C:\Program Files directory, from your desktop selecting Icon **(or)**

2. You can launch from the Windows task bar click 'Start' , choose 'All Programs' , choose 'Weka 3.6' to start the WEKA.

Then 'WEKA GUI Chooser' window appears on the screen, you can select one of the four options at the bottom of the window.

ii) Understanding the features of WEKA tool kit such as Explorer, Knowledge flow interface , Experimenter, Command line interface.

- It is an Explorer for exploring Data.
- Preprocessing, Classification, Clustering, Attribute selection, Learning, Visualization.

The Experimenter (experimental environment):

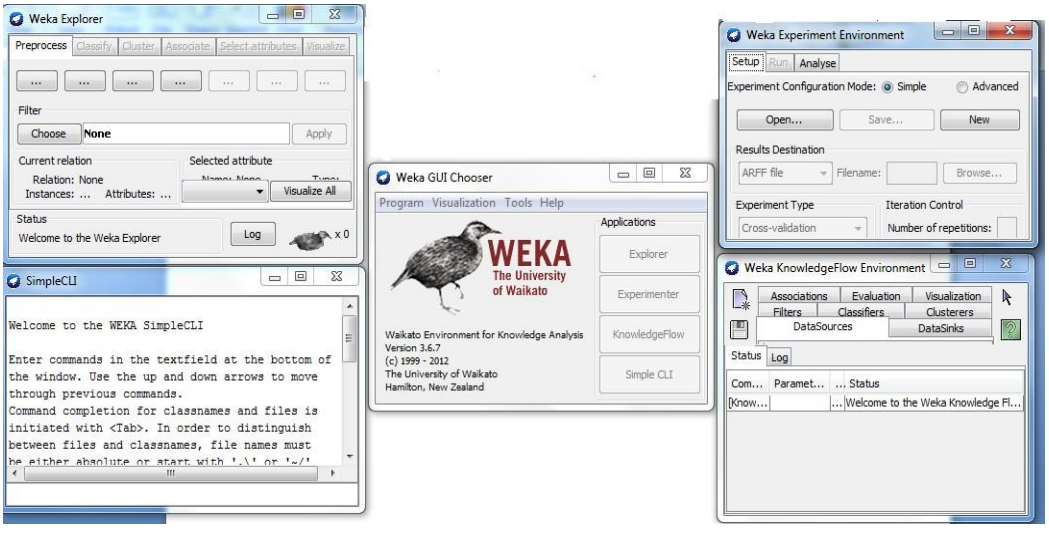
- It is an environment for performing experiments and conducting statistical tests in between learning schemes.
- Testing and evaluating machine learning algorithms

The Knowledge Flow (new process model inspired interface):

- It is a java-beans –based interface for setting up and running machine learned experiments.
- Visual design of KDD process

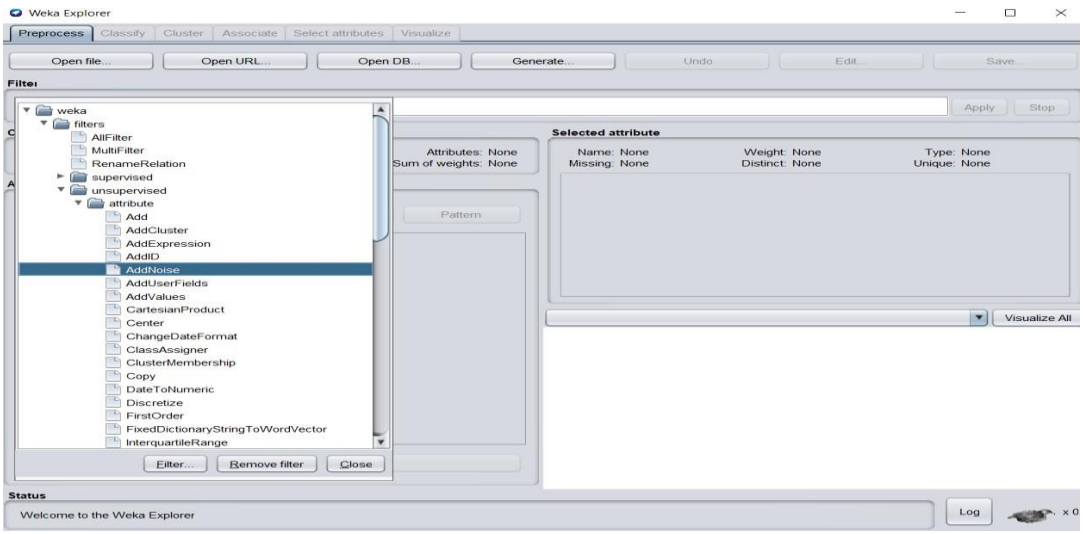
Simple Command-Line Interface:

- A simple interface for typing commands
- Allows direct execution of WEKA commands.

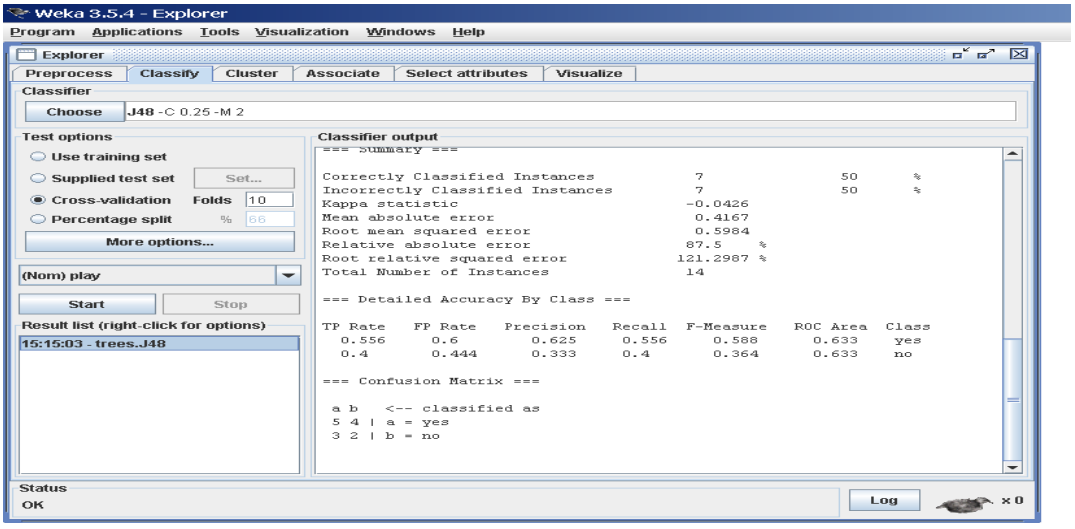


iii) Navigate the options available in the WEKA.

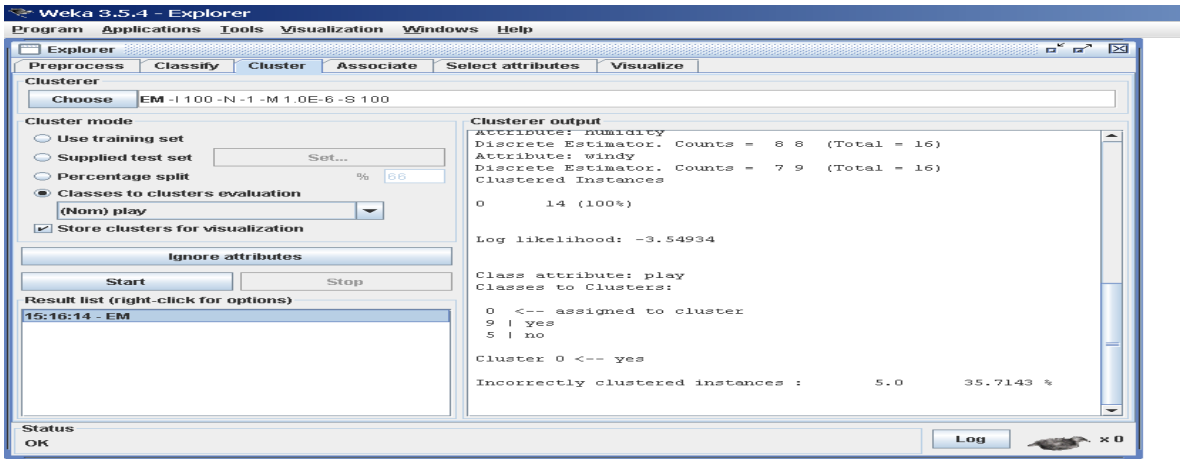
- **Preprocessing Filters**



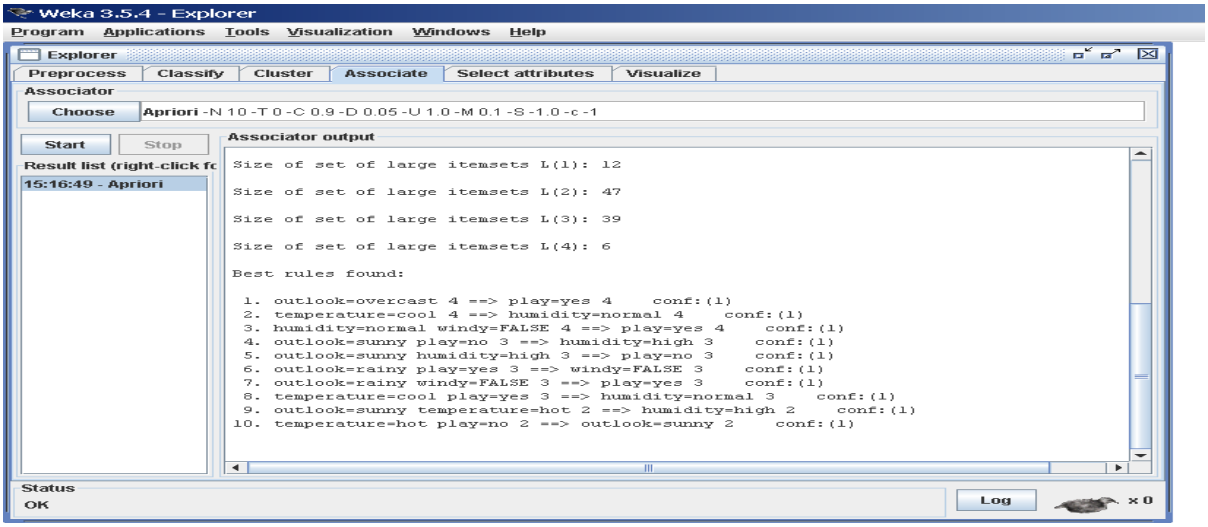
- **Classification/Regression**



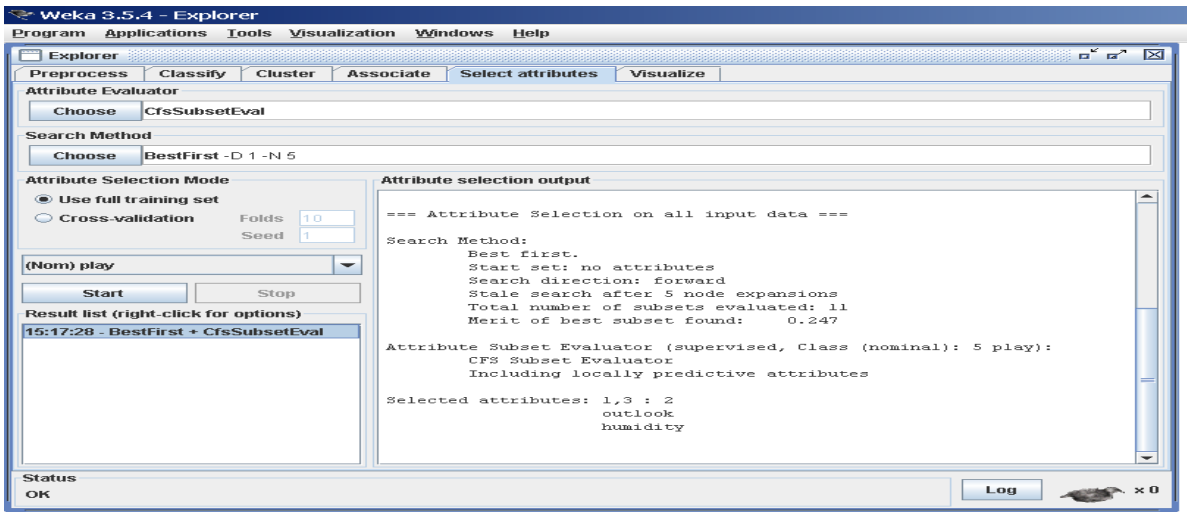
- **Clustering**



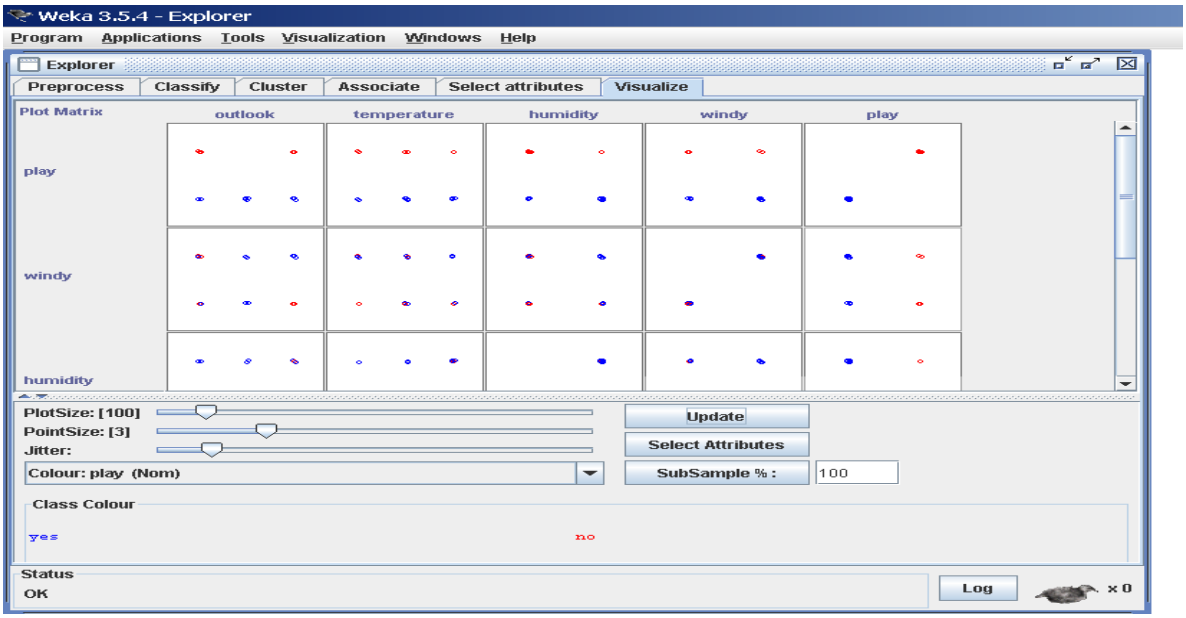
- **Association**



- **Select Attributes**



- **Visualization**



iv)Study the ARFF file format.

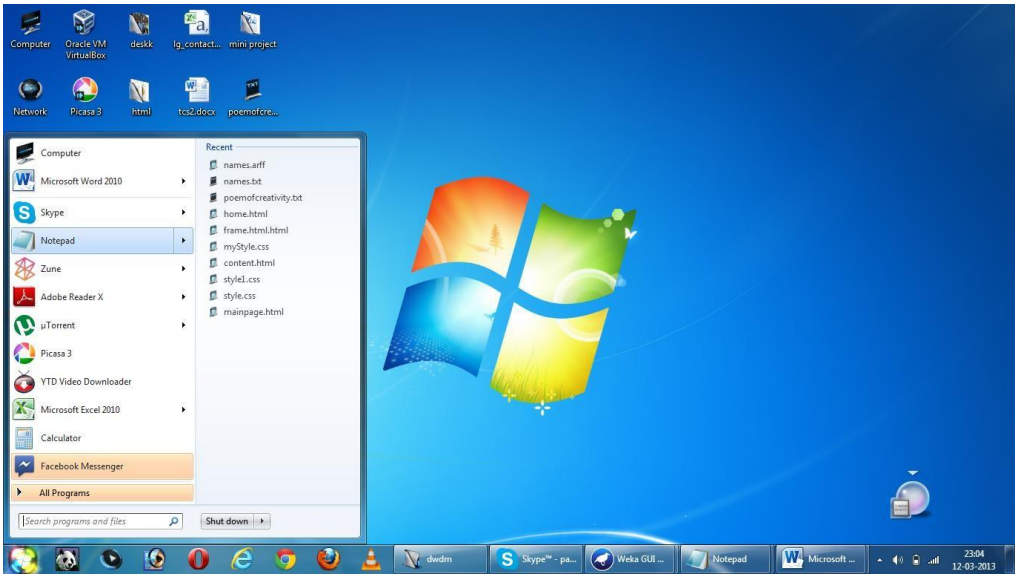
An ARFF file consists of two distinct sections:

- the Header section defines attribute name, type and relations, start with a keyword.
  @Relation <data-name>
  @attribute <attribute-name> <type> or {range}
- The Data section lists the data records, starts with @Data list of data instances.
- Any line start with % is the comments.

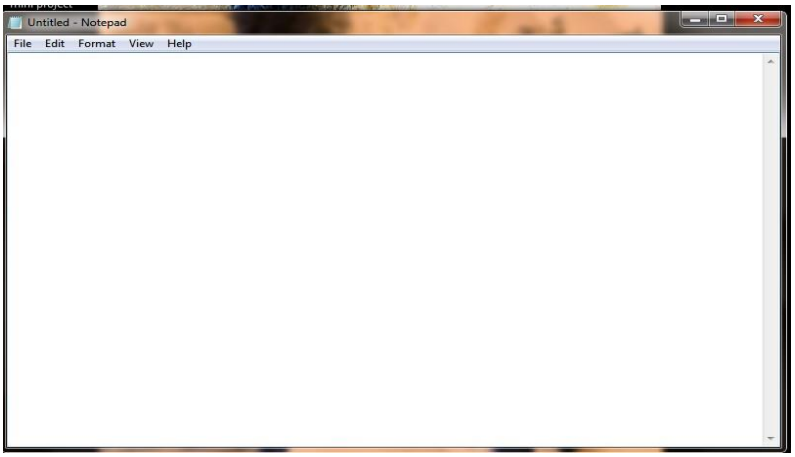
Data types supported by ARFF:

- numeric
- string
- nominal specification
- date

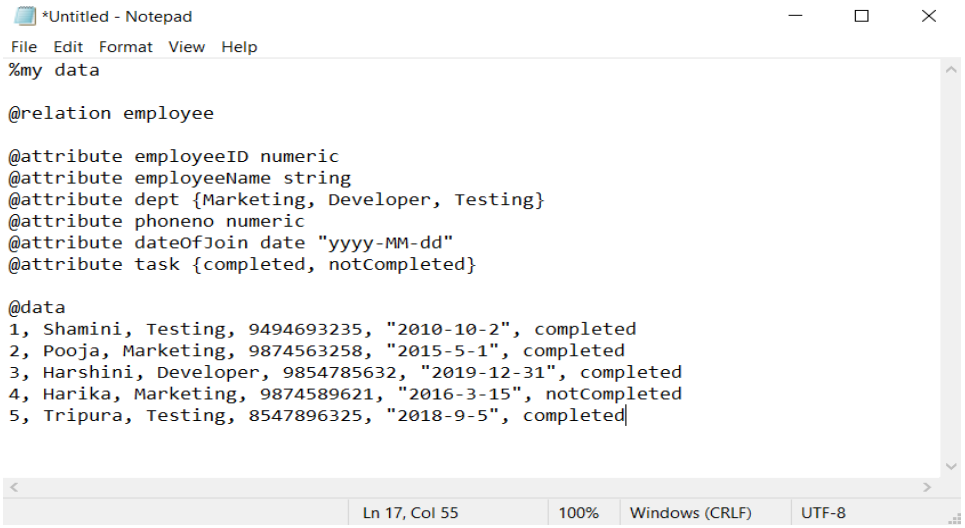
Creation Of ARFF File



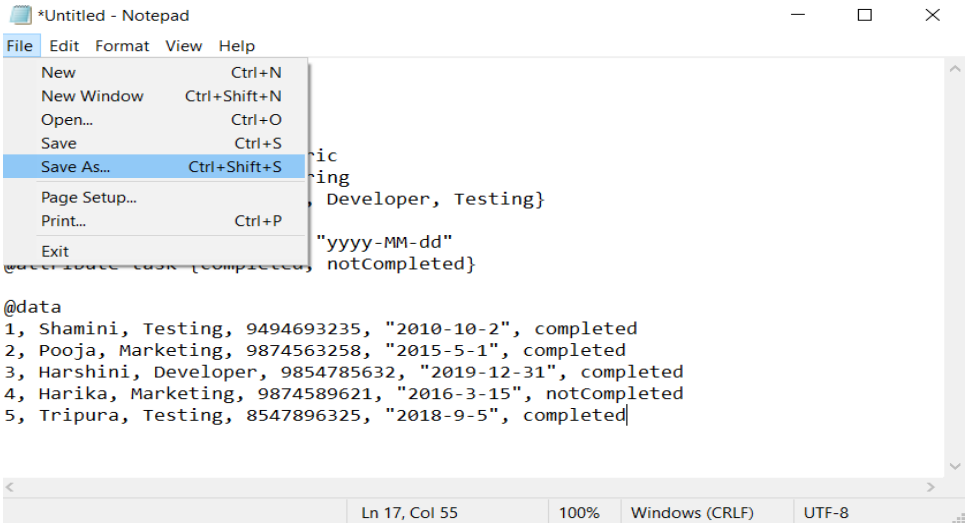
Step-1: To create an ARFF file, first open a notepad by the following way.
Program➋Accessories➋Notepad.



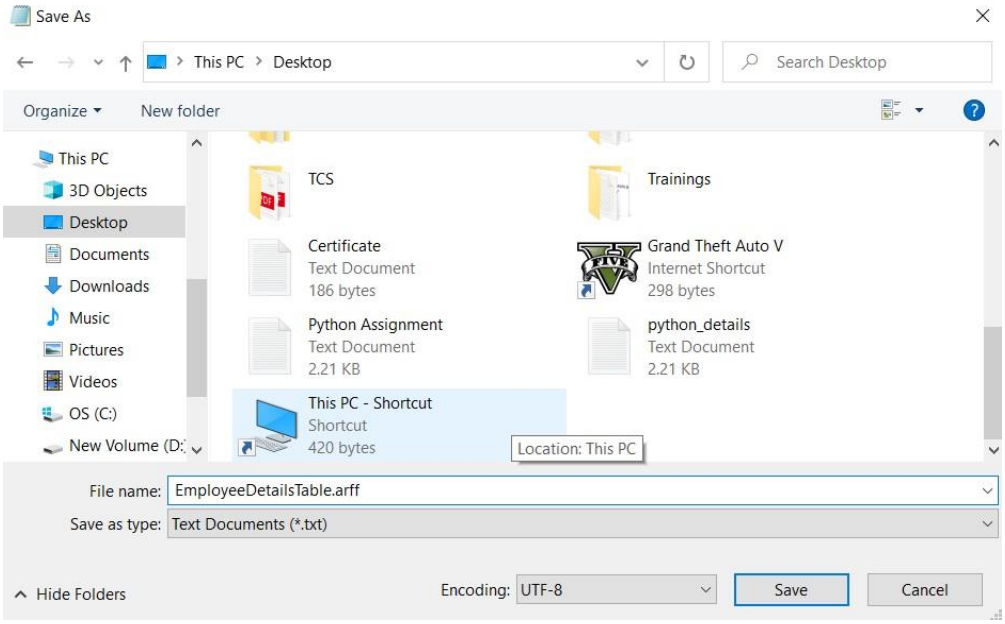
Step-2: Then click on Notepad, we will get a window having new page.



Step-3: Creating ARFF file in notepad .



Step-4: After creation of ARFF file, save by following way.File➋Save
After opening the file then click on save.–



Step-5: After clicking on save, then we will get save window and save ARFF file with .arff extension.

v) Explore the available data sets in WEKA.
vi) Load a data set (ex: Weather dataset, Iris dataset, etc.)
vii) Load each dataset and observe the following:

(i) List the attribute names and their types

(ii) Number of records in each dataset

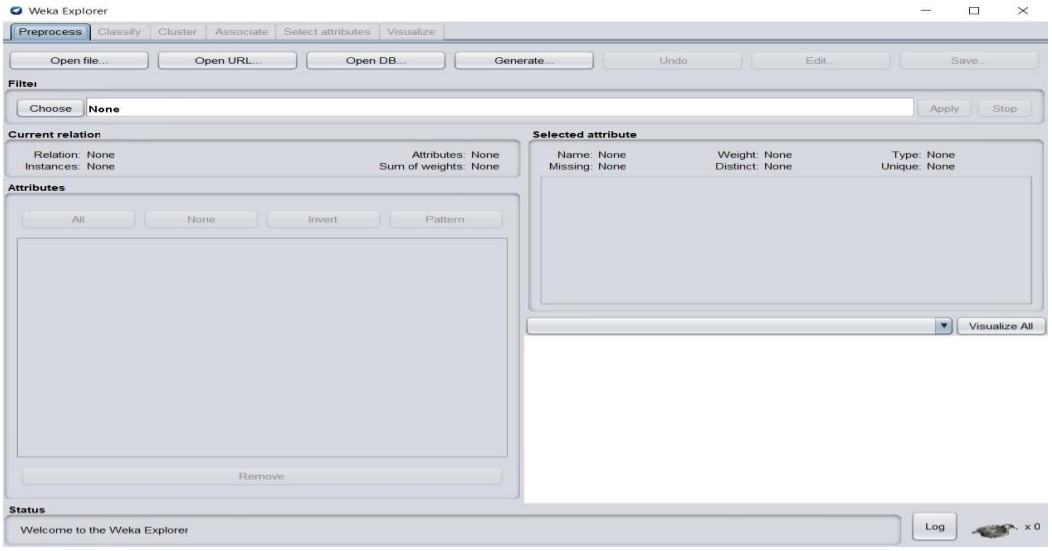
(iii)Identify the class attribute (if any)

(iv) Plot Histogram

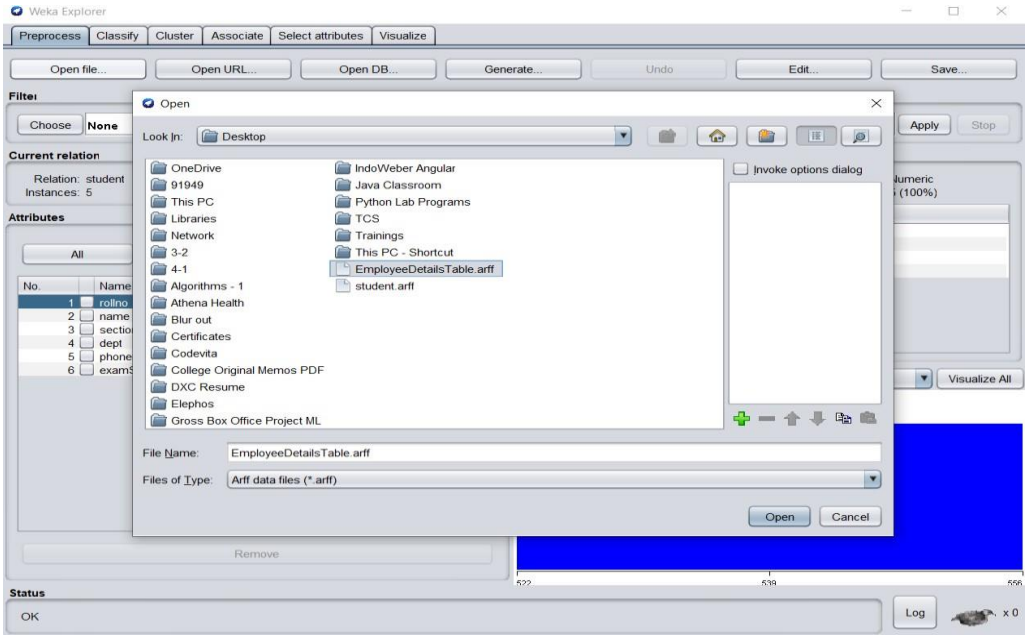
(v) Determine the number of records for each class

(vi) Visualize the a data in various dimensions

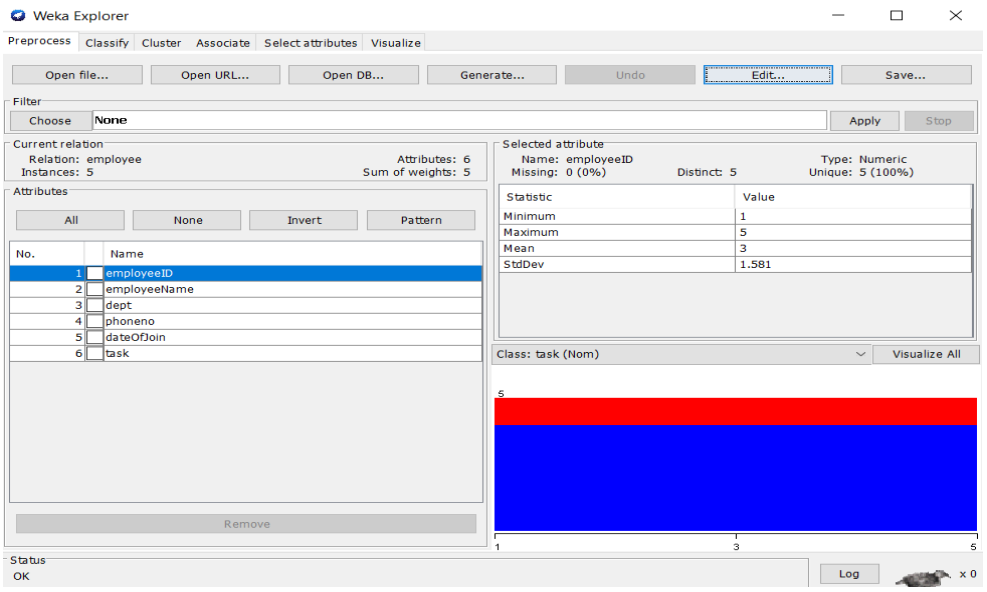
Step-1: After completion of creation of ARFF file, open Weka GUI user by clicking the weka icon and select the Explorer option from the list of applications



Step-2: Now we will get a window called Weka explorer where we should open the ARFF file. Now click on open file button.



Step-3: Then we will get one window named open. Using this window we have to open the ARFF file which we created in the above steps and then click on open button.



Step-4: After opening the file, we will get number of attributes and names of those attributes. The type and additional details are also displayed for the selected attribute at the right side of the preprocess window.

Name : is the name of an attribute,

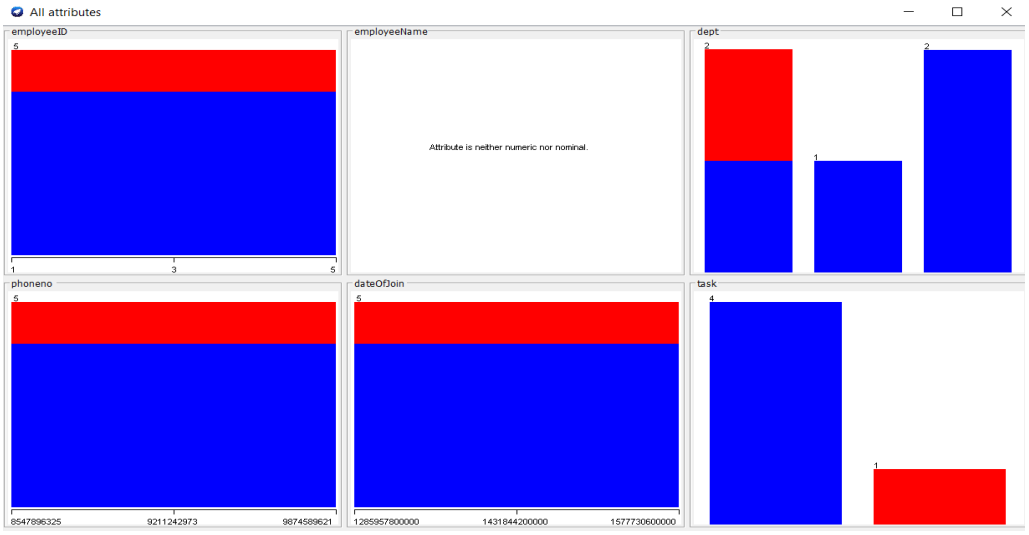
Type : is most commonly Nominal or Numeric.

Missing : is the number (percentage) of instances in the data for which this attribute is Unspecified.

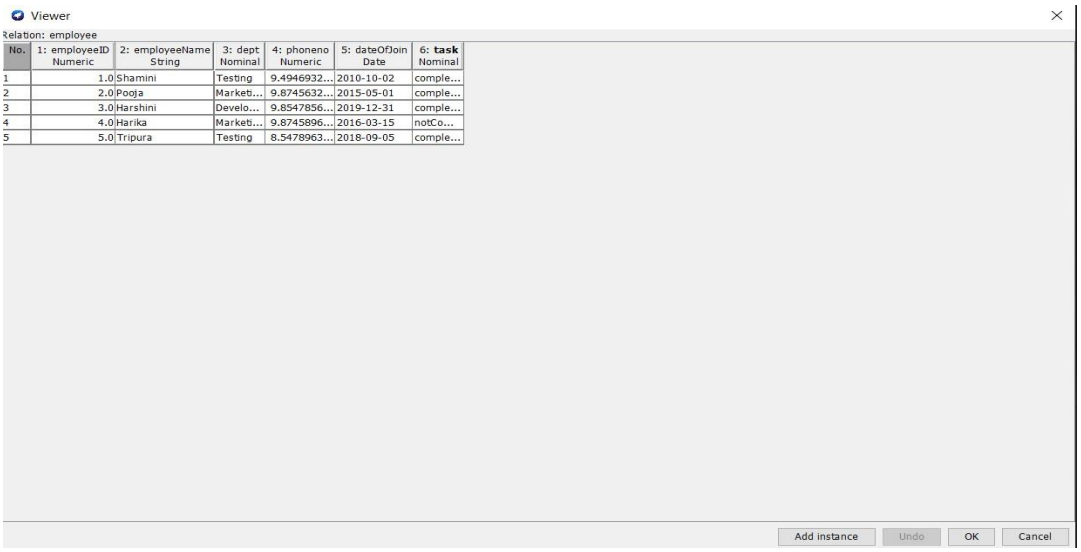
Distinct : is the number of different values that the data contains for this attribute.

Unique : is the number (percentage) of instances in the data having a value for this attribute that no other instances have.

Step-5: When we click on the visualize all button, we can see the details of all the attributes at one place(plot histogram).



This is the visualize all window.



Step-6: Click on the edit option to open the viewer. In this, we can see all the data that we saved in the ARFF file.

Programs:

Program 1: List all the categorical (or nominal) attributes and the real-valued attributes separately.

SOLUTION:

Count all qualitative and numerical attributes

Number of Attributes german: 21 (7 numerical, 14 categoric

Categorical or Nominal attributes:-

1. checking_status
2. credit history
3. purpose
4. savings_status
5. employment_since
6. personal status
7. debtors
8. property
9. installment plans
10. housing
11. job
12. telephone
13. foreign worker
14. class label

Real valued attributes:-

1. duration
2. credit amount
3. installment rate
4. residence_since
5. age

6. existing credits
7. num_dependents

Program 2: What attributes do you think might be crucial in making the credit assessment? Come up with some simple rules in plain English using your selected attributes.

SOLUTION:

According to me the following attributes may be crucial in making the credit risk assessment.

1. Credit history: if the credit history of the applicant is good, then we are assured that the applicant will pay the installments of the loan regularly.

2. Employment: if the applicant is employed, he has a source of income to repay the loan. So, there is less risk.

3. Property magnitude: if the magnitude of the property is more, even if the applicant fails to pay the loan, the bank can seize the property.

4. Job: if the applicant is employed and highly qualified, then the salary of applicant will be high and he can easily repay the loan.

5. Credit amount: If the credit amount is less, then the risk associated with it is less as compared to huge credit amounts.

6. Installment commitment: if the installment commitment of the applicant is high, then the bank can grant the loan.

7. Existing credits: if the number of existing credits of the applicant is low, then the bank can grant the loan as the applicant will have less commitments.

Basing on the above attributes, we can make a decision whether to give credit or not.

Program 3: One type of model that you can create is a Decision Tree - train a Decision Tree using the complete dataset as the training data. Report the model obtained after training.

SOLUTION:

German Data set –

Step-1: Select Explorer from weak GUI chooser then click on open file option in preprocess tab.

Step-2: select the credit_g arff file and click on open.

Step-3: select the class attribute and go to classify tab.

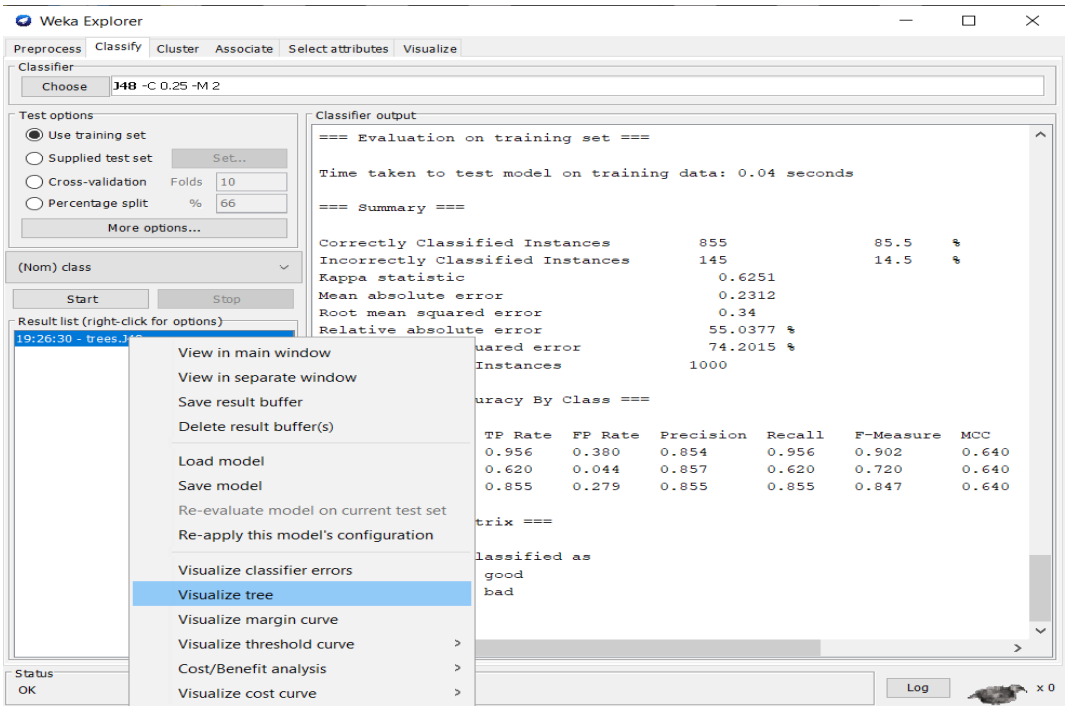
Step-4: select the use training set option from the list of test options.

Step-5: select the J48 tree from the list of trees at the choose option.

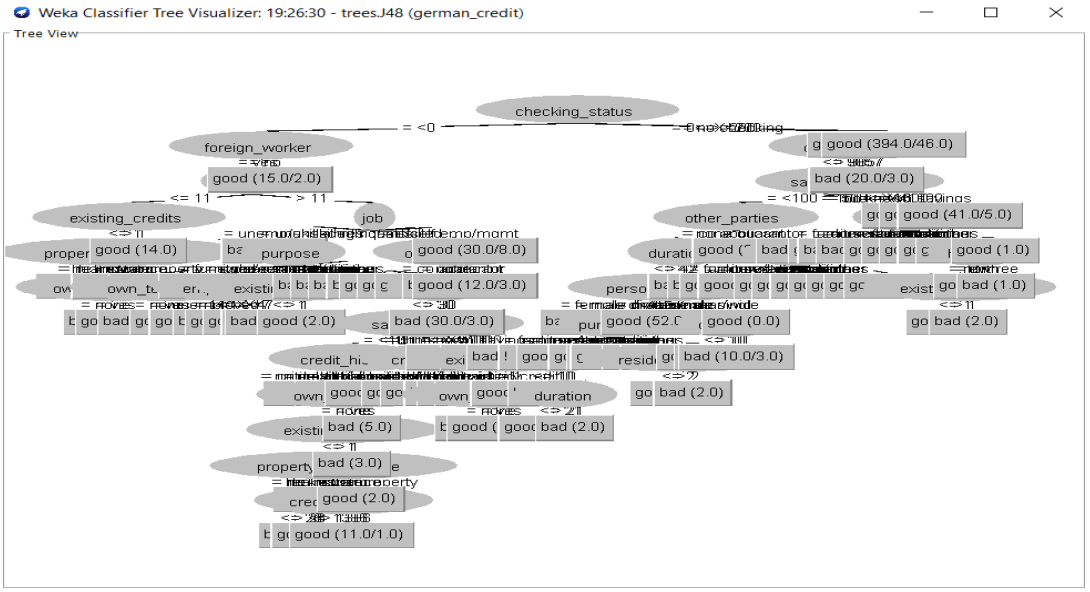
Step-6: Select class label in drop down list then click on start option.

Step-7: The Accuracy results will be displayed at the right side of the window.

Outputs:



Step-8: To visualize the decision tree, right click on the result list and select the visualize tree option.
Visualize Decision Tree:



Step-9: Click on the auto scale and fit to screen options to zoom in to the decision tree.

Calculations:

=== Confusion Matrix ===

a b <-- classified as

669 31 | a = 1

114 186 | b = 2

Number of Leaves : 103

Size of the tree : 140

Time taken to build model: 0.12 seconds

Program 4: Suppose you use your above model trained on the complete dataset, and classify credit good/bad for each of the examples in the dataset. What % of examples can you classify correctly? (This is also called testing on the training set) Why do you think you cannot get 100 % training accuracy?

Predictive Accuracy Evaluation:

The main methods of predictive accuracy evaluations are:

- Resubstitution (N ; N)
- Holdout (2N/3 ; N/3)
- x-fold cross-validation (N-N/x ; N/x)
- Leave-one-out (N-1 ; 1)

Where N is the number of records (instances) in the dataset

Training and Testing:

REMEMBER: we must know the classification (class attribute values) of all instances (records) used in the test procedure.

- **Success:** instance (record) class is predicted correctly
- **Error:** instance class is predicted incorrectly
- **Error rate:** a percentage of errors made over the whole set of instances (records) used for testing
- **Predictive Accuracy:** a percentage of well classified data in the testing data set.

SOLUTION :-

According to the rules, for the maximum accuracy, we have to take 2/3 of the dataset as training set and the remaining 1/3 as test set. But here in the above model we have taken complete dataset as training set which results only 85.5% accuracy correctly classified and remaining 14.5% of examples are incorrectly.

Hence, we can't get 100% training accuracy because out of the 20 attributes, we have Some unnecessary attributes which are also been analyzed and trained.

Program 5: Is testing on the training set as you did above a good idea? Why or Why not?

SOLUTION:

Testing on Training Set (2N/3 ; N/3):

According to the rules, for the maximum accuracy, we have to take 2/3 of the dataset as training set and the remaining 1/3 as test set. But here in the above model we have taken complete dataset as training set which results only 85.5% accuracy.

This is done for the analyzing and training of the unnecessary attributes which does not make a crucial role in credit risk assessment. And by this complexity is increasing and finally it leads to the minimum accuracy.

If some part of the dataset is used as a training set and the remaining as test set then it leads to the accurate results and the time for computation will be less.

This is why, we prefer not to take complete dataset as training set.

In some of the cases it is good, it is better to go with cross validation

X-fold cross-validation (N-N/x; N/x):

The cross-validation is used to prevent the overlap of the test sets
First step: split data into x disjoint subsets of equal size
Second step: use each subset in turn for testing, the remainder for training (repeating cross-validation)
As resulting rules (if applies) we take the sum of all rules.
The error (predictive accuracy) estimates are averaged to yield an overall error (predictive accuracy) estimate Standard cross-validation: 10-fold cross-validation
Why 10?
Extensive experiments have shown that this is the best choice to get an accurate estimate. There is also some theoretical evidence for this. So interesting!

Program 6: One approach for solving the problem encountered in the previous question is using cross-validation? Describe what cross-validation is briefly. Train a Decision Tree again using cross- validation and report your results. Does your accuracy increase/decrease? Why?

Cross validation:-

In k-fold cross-validation, the initial data are randomly portioned into 'k' mutually exclusive subsets or folds D1, D2, D3, ..., Dk. Each of approximately equal size. Training and testing is performed 'k' times. In iteration I, partition Di is reserved as the test set and the remaining partitions are collectively used to train the model. That is in the first iteration subsets D2, D3,....., Dk collectively serve as the training set in order to obtain as first model. Which is tested on Di. The second trained on the subsets D1, D3, ..., Dk and test on the D2 and so on….

SOLUTION:

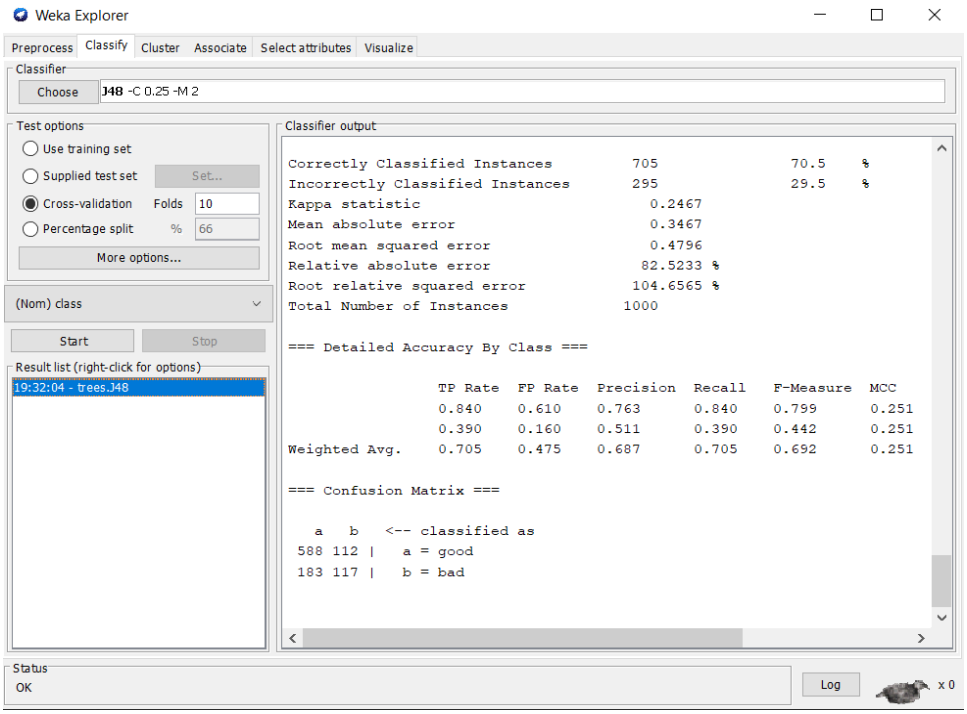
Step-1: Select Explorer from weak GUI chooser then click on open file option in preprocess tab.

Step-2: select credit-g.arff file and click on open option.

Step-3: select the cross validation option from the test options.

Step-4: select the J48 tree from the choose option ad click on start button.

Step-5: Here we can observe the decreased accuracy rate with cross validation.



Calculations:

=== Confusion Matrix ===

a b <-- classified as

588 112 | a = good

183 117 | b = bad

Number of Leaves : 103

Size of the tree : 140

Time taken to build model: 0.07 seconds

Program 7: Check to see if the data shows a bias against "foreign workers" (attribute 20),or "personal-status"(attribute 9). One way to do this (perhaps rather simple minded) is to remove these attributes from the dataset and see if the decision tree created in those cases is significantly different from the full dataset case which you have already done.. To remove an attribute you can use the reprocess tab in Weka's GUI Explorer. Did removing these attributes have any significant effect? Discuss SOLUTION:

This increase in accuracy is because thus two attributes are not much important in training and analyzing by removing this, the time has been reduced to some extent and then it results in increase in the accuracy.

The decision tree which is created is very large compared to the decision tree which we have trained now. This is the main difference between these two decision trees.

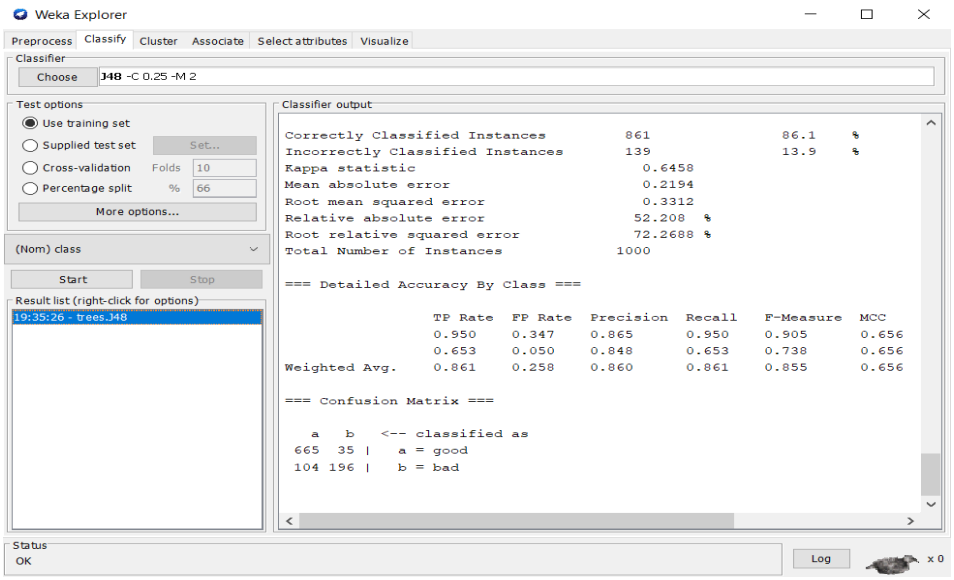
Step-1: Select Explorer from weak GUI chooser then click on open file option in preprocess tab.

Step-2: After file open select the class label to classify the data.

Step-3: select the foreign_worker and personal_status attributes from the list of attributes.

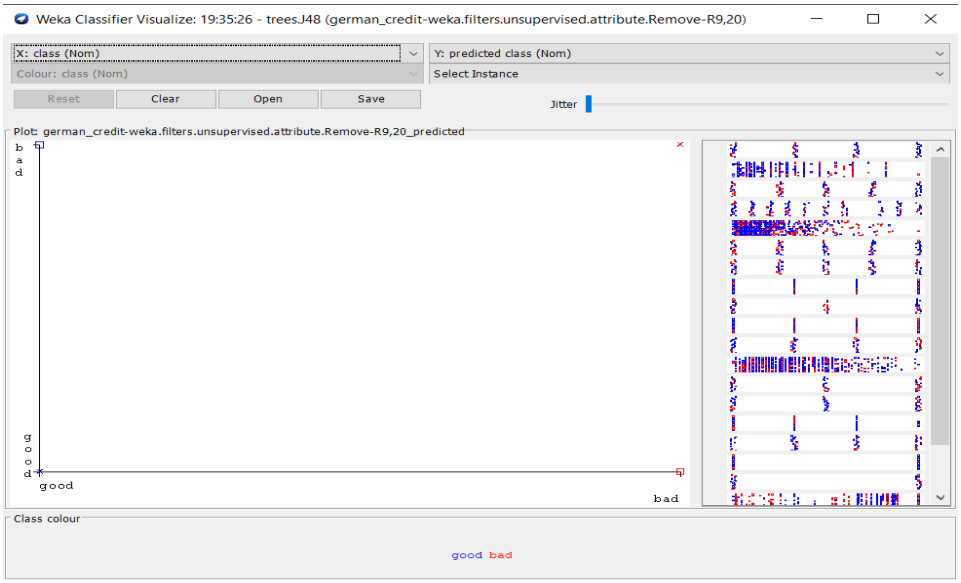
Step-4: now, click the remove button to remove these attributes.

Step-5: now, in the classify tab, select the use training set option and select the J48 tree and click on start button. We can observe the result improvement in accuracy as shown above.

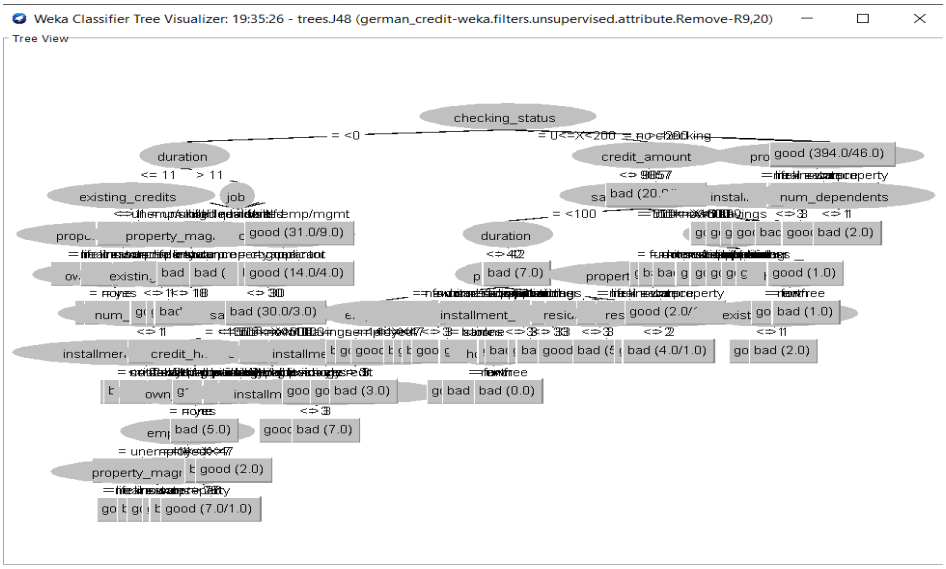


Outputs:

Visualize classifier errors:



Visualize tree



**** The Difference what we observed is accuracy had improved.**

Calculations:

=== Confusion Matrix ===

a b <-- classified as

665 35 | a = good

104 196 | b = bad

Number of Leaves : 97

Size of the tree : 139

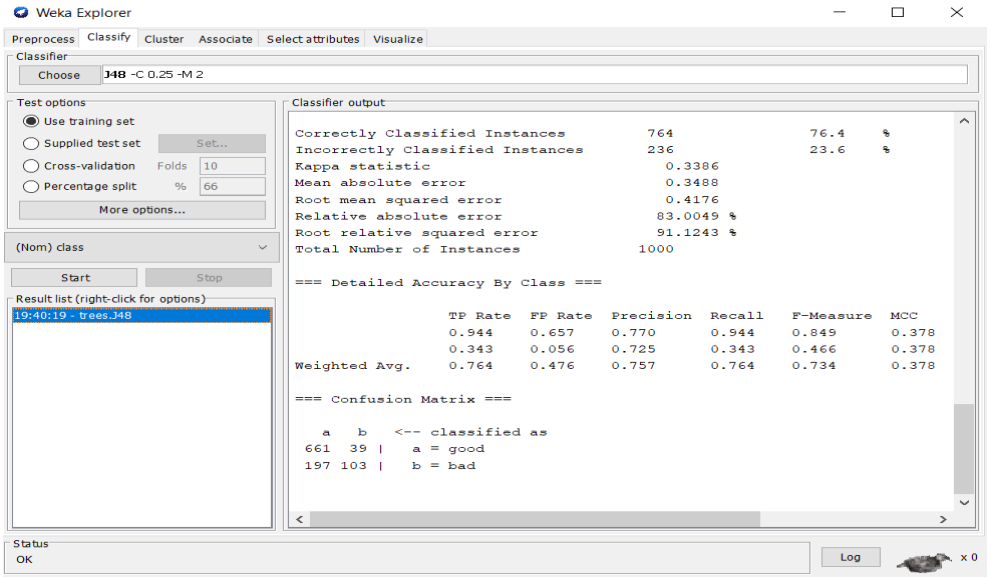
Time taken to build model:0.05 seconds

Program 8: Another question might be, do you really need to input so many attributes to get good results? Maybe only a few would do. For example, you could try just having attributes 2, 3, 5, 7, 10, 17 (and 21, the class attribute (naturally)). Try out some combinations. (You had removed two attributes in problem 7. Remember to reload the arff data file to get all the attributes initially before you start selecting the ones you want.)

Step-1: Select Explorer from weak GUI chooser then click on open file option in preprocess tab.

Step-2: After file opened , select the 1,4,6,8,9,11,12,13,14,15,16,18,19,20 attributes and click on remove.

Step-3: In the classifier, select the use training set option and choose the J48 tree and click on start to view the accuracy results



Calculations:

=== Confusion Matrix ===

a b <-- classified as

661 39 | a = good

197 103 | b = bad

Number of Leaves : 27

Size of the tree : 40

Time taken to build model: 0.01 seconds

Program 9: Sometimes, the cost of rejecting an applicant who actually has a good credit (case 1) might be higher than accepting an applicant who has bad credit (case 2). Instead of counting the misclassifications equally in both cases, give a higher cost to the first case (say cost 5) and lower cost to the second case. You can do this by using a cost matrix in Weka. Train your Decision Tree again and report the Decision Tree and cross-validation results. Are they significantly different from results obtained in problem 6 (using equal cost)?

In the Problem 6, we used equal cost and we trained the decision tree. But here, we consider two cases with different cost.

Let us take Cost Matrix as equal in case1 and unequal in case2.

a). CostSensitiveClassifier (With Equal Matrix)

Cost Matrix

0 1
1 0

Step-1: Select Explorer from weak GUI chooser then click on open file option in preprocess tab.

Step-2: select the credit_g arff file and click on open.

Step-3: Select classify tab then choose CostSensitiveClassifier from meta folder of weka

Step-4: Right click on CostSensitiveClassifier and select show properties.

Step-5: We will get the metaclassifier properties window.

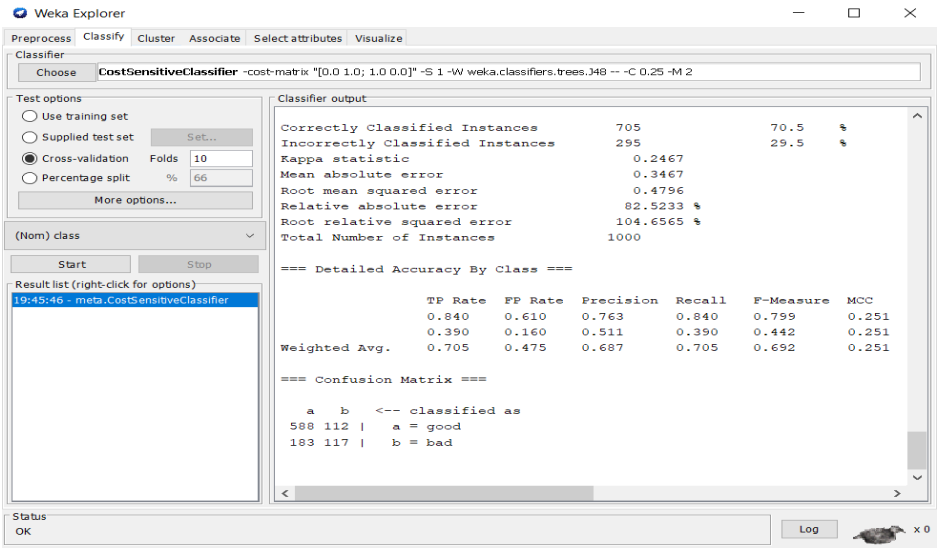
Step-6: Click on classifier choose button and select J48 from trees folder of weka.

Step-7: Double click on costMatrix then change the value of clases to 2 and click on Resize button.

Step-8: We can observe 2X2 equal cost matrix.

Step-9 : Now click on OK.

Step-10: Select cross validation in test options and click on start to view the results.



Calculations:

=== Confusion Matrix ===

a b <-- classified as

588 112 | a = good

183 117 | b = bad

Number of Leaves : 103

Size of the tree : 140

Time taken to build model: 0.14 seconds

CostSensitiveClassifier (With Unequal Matrix)

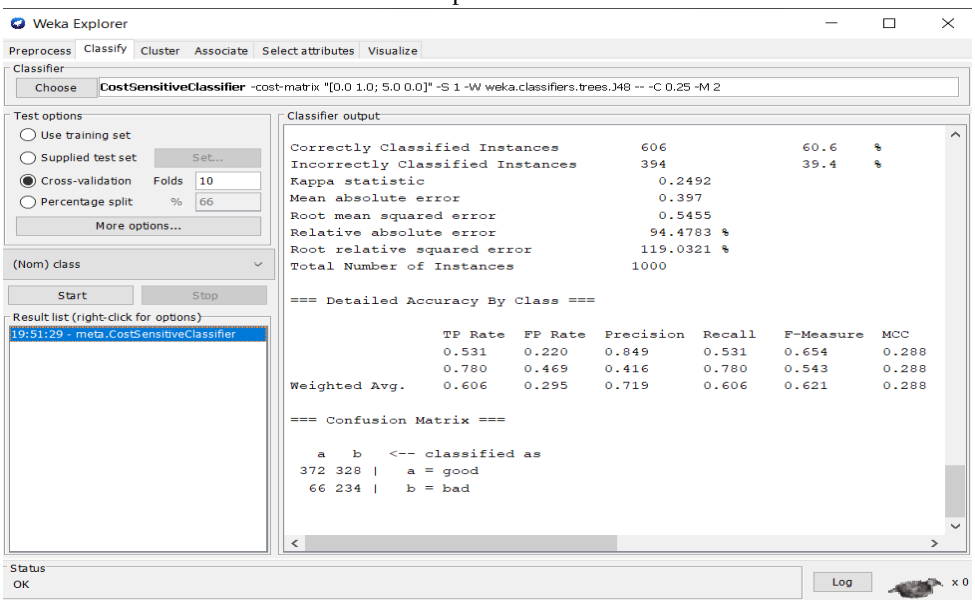
Cost Matrix

0 1

5 0

Step-1: Repeat the first 7 steps of 2X2 equal matrix then change the value of first column and second row is 5 to make it as an unequal matrix

Step-2: Click on OK then select cross validation in test options and click on start to view the results.



Calculations:

=== Confusion Matrix ===

a b <-- classified as

372 328 | a = good

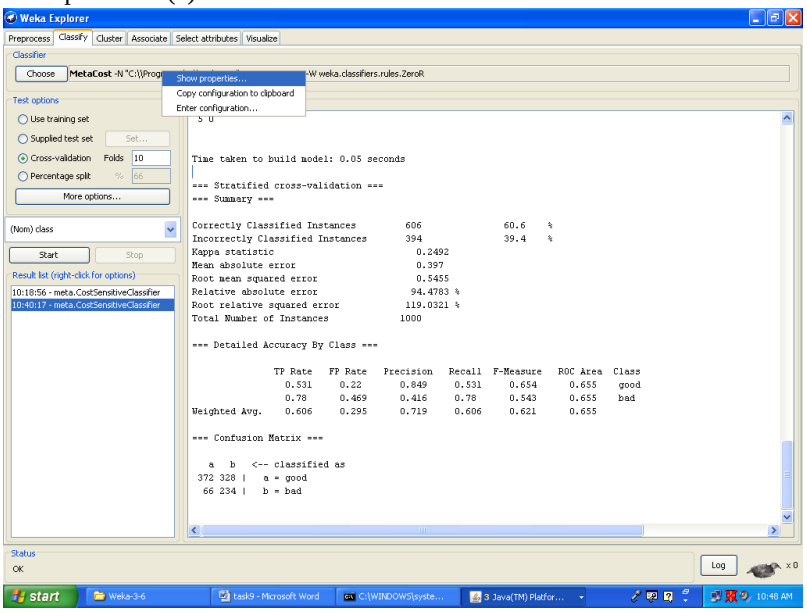
66 234 | b = bad

Number of Leaves : 65
Size of the tree : 94
Time taken to build model: 0.05 seconds

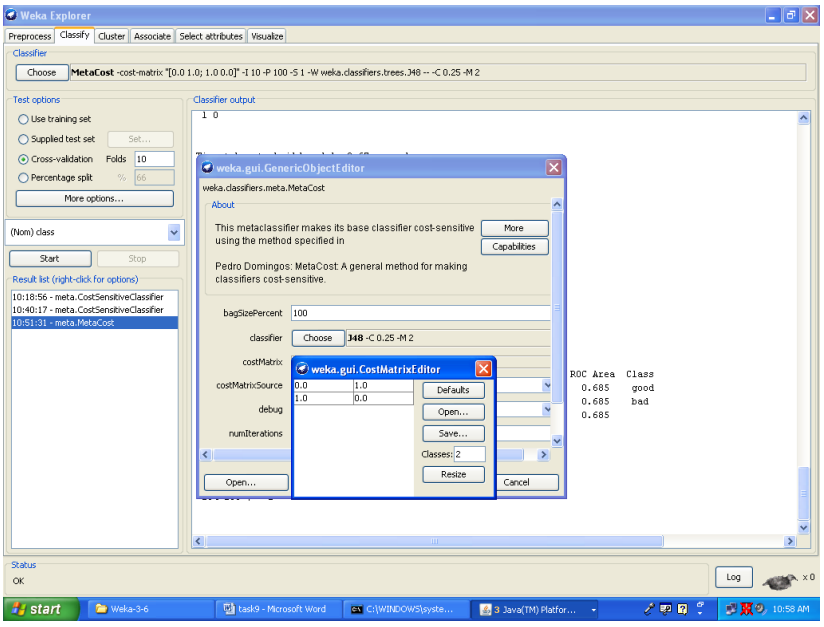
b). MetaCost (With Equal Matrix)

Cost Matrix
0 1
1 0

Step-1: Repeat the first 3 steps Task 9(a) then select MetaCost.

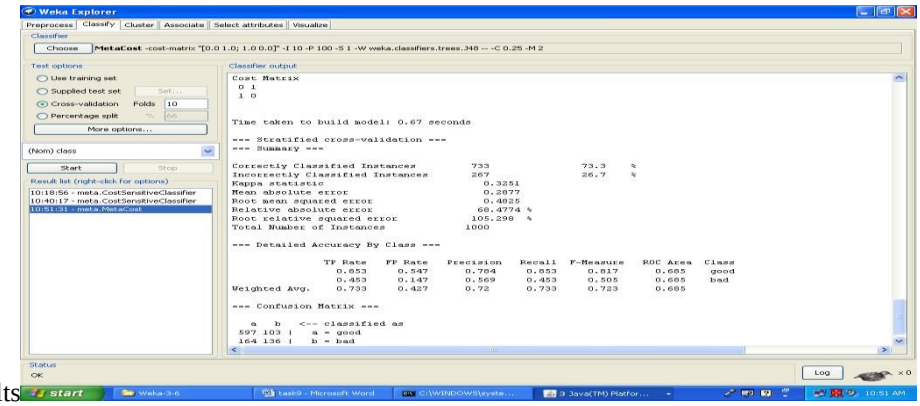


Step-2: Right click on MetaCost and select show properties.



Step-3: Repeat the steps from 5 to 7 of Task 9(a) then double click on cost matrix then we can view the 2X2 equal matrix.

Step-4: Now click on OK and select cross validation in test options and click on start to view the



results

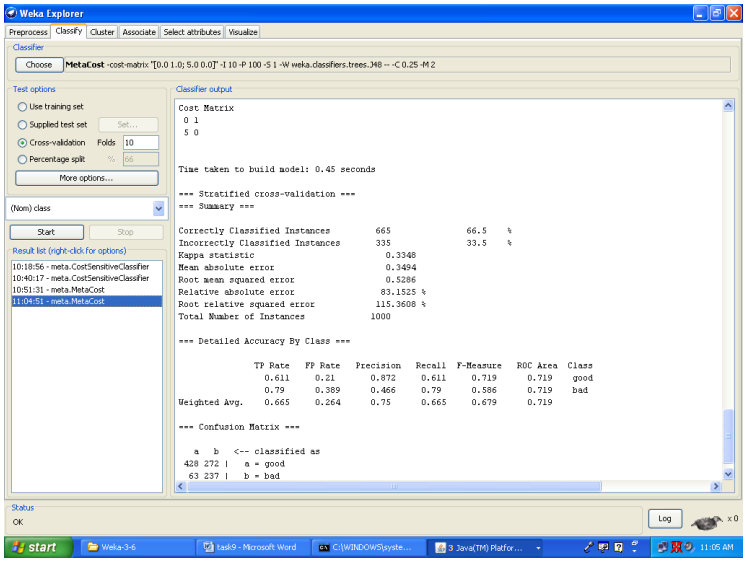
Calculations:

=== Confusion Matrix ===
a b <-- classified as
597 103 | a = good
164 136 | b = bad
Number of Leaves : 104
Size of the tree : 147
Time taken to build model: 0.67 seconds

MetaCost (With Unequal Matrix)

Cost Matrix
0 1
5 0

Step-1: Now change the value of first column and second row is 5 to make it as an unequal matrix
Step-2: Click on OK then select cross validation in test options and click on start to view the results.



Calculations:

=== Confusion Matrix ===
a b <-- classified as
428 272 | a = good
63 237 | b = bad
Number of Leaves : 86
Size of the tree : 123
Time taken to build model: 0.45 seconds

We find some difference in cost factor which is in summary in the difference in cost factor and in time to build the model and also in accuracy.

Program 10: Do you think it is a good idea to prefer simple decision trees instead of having long complex decision trees? How does the complexity of a Decision Tree relate to the bias of the model?

When we consider long complex decision trees, we will have many unnecessary attributes in the tree which results in increase of the bias of the model. Because of this, the accuracy of the model can also effected.

This problem can be reduced by considering simple decision tree. The attributes will be less and it decreases the bias of the model. Due to this the result will be more accurate.

So it is a good idea to prefer simple decision trees instead of long complex trees.

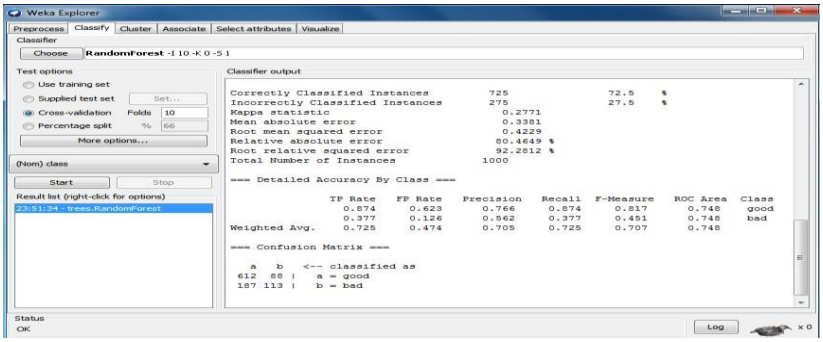
These results are also confirmed by observing the results that the improvement in accuracy compared with task-6.

Step-1: Select Explorer from weak GUI chooser then click on open file option in preprocess tab.

Step-2: select credit-g.arff file and click on open option.

Step-3: select the random forest treefrom trees of weka in the choose option of classify tab.

Step-4: select the cross validation option from the test options and click on start.



From the results we can observe that the accuracy rate with simple tree increased to 72.5% compared to that of Task 6 simple tree i.e., 70.5%.

Calculations:

=== Confusion Matrix ===

a b <-- classified as

612 88 | a = good

187 113 | b = bad

Time taken to build model: 0.11 seconds

Program 11: You can make your Decision Trees simpler by pruning the nodes. One approach is to use Reduced Error Pruning - Explain this idea briefly. Try reduced error pruning for training your Decision Trees using cross-validation (you can do this in Weka) and report the Decision Tree you obtain ? Also, report your accuracy using the pruned model. Does your accuracy increase ?

Reduced-Error Pruning :-

The idea of using a separate pruning set for pruning—which is applicable to decision trees as well as rule sets—is called reduced-error pruning. The variant described previously prunes a rule immediately after it has been grown and is called incremental reduced-error pruning. Another possibility is to build a full, unpruned rule set first, pruning it afterwards by discarding individual tests. However, this method is much slower. Of course, there are many different ways to assess the worth of a rule based on the pruning set. A simple measure is to consider how well the rule would do at discriminating the predicted class from other classes if it were the only rule in the theory, operating under the closed world assumption. If it gets p instances right out of the t instances that it covers, and there are P instances of

this class out of a total T of instances altogether, then it gets p positive instances right. The instances that it does not cover include N - n negative ones, where n = t – p is the number of negative instances that the rule covers and N = T - P is the total number of negative instances. Thus the rule has an overall success ratio of [p +(N - n)] T , and this quantity, evaluated on the test set, has been used to evaluate the success of a rule when using reduced-error pruning.

Step-1: Select Explorer from weak GUI chooser then click on open file option in preprocess tab.

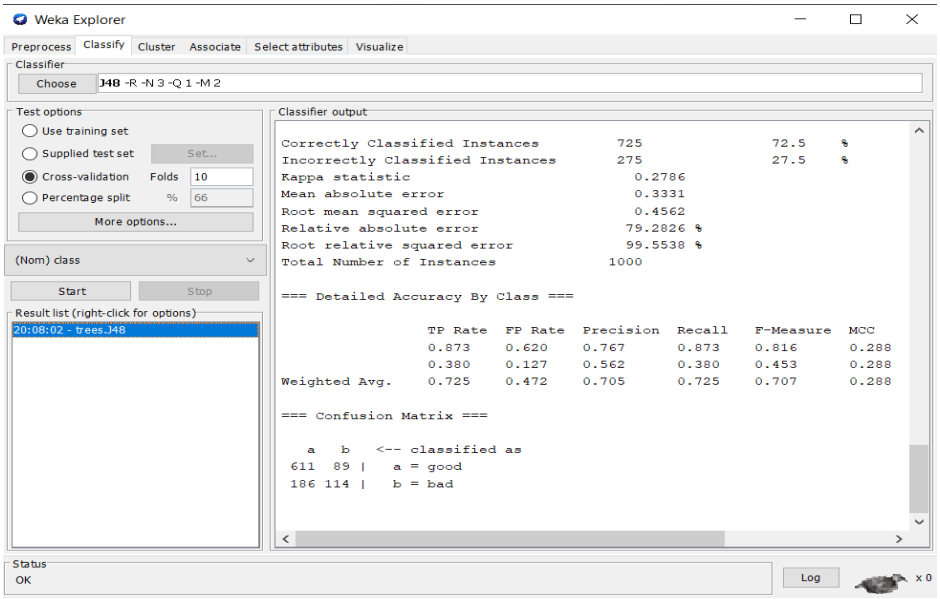
Step-2: select credit-g.arff file and click on open option.

Step-3: After choosing the file in classify tab, choose the J48 tree, right click on the choose bar and select show properties.

Step-4: we will then get a generic object editor. Change the reduced error pruning value from false to true.

Step-5: click on OK after changing the value.

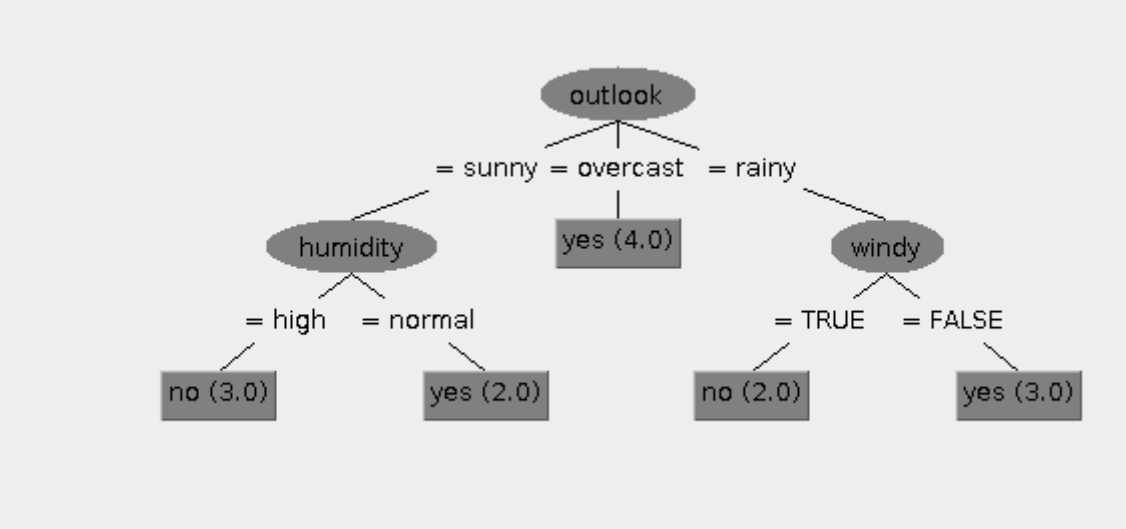
Step-6: select the cross validation option and click on start.
From the results we can observe that the accuracy rate with pruning increased to 72.5% compared to that of without pruning of Task6 i.e., 70.5%.



Calculations:
=== Confusion Matrix ===
a b <-- classified as
611 89 | a = good
186 114 | b = bad
Number of Leaves : 47
Size of the tree : 64
Time taken to build model: 0.49 seconds

Program 12: (Extra Credit): How can you convert a Decision Trees into "if-then-else rules". Make up your own small Decision Tree consisting of 2-3 levels and convert it into a set of rules. There also exist different classifiers that output the model in the form of rules - one such classifier in Weka is rules.PART, train this model and report the set of rules obtained. Sometimes just one attribute can be good enough in making the decision, yes, just one ! Can you predict what attribute that might be in this dataset? OneR classifier uses a single attribute to make decisions (it chooses the attribute based on minimum error). Report the rule obtained by training a one R classifier. Rank the performance of j48, PART and oneR.

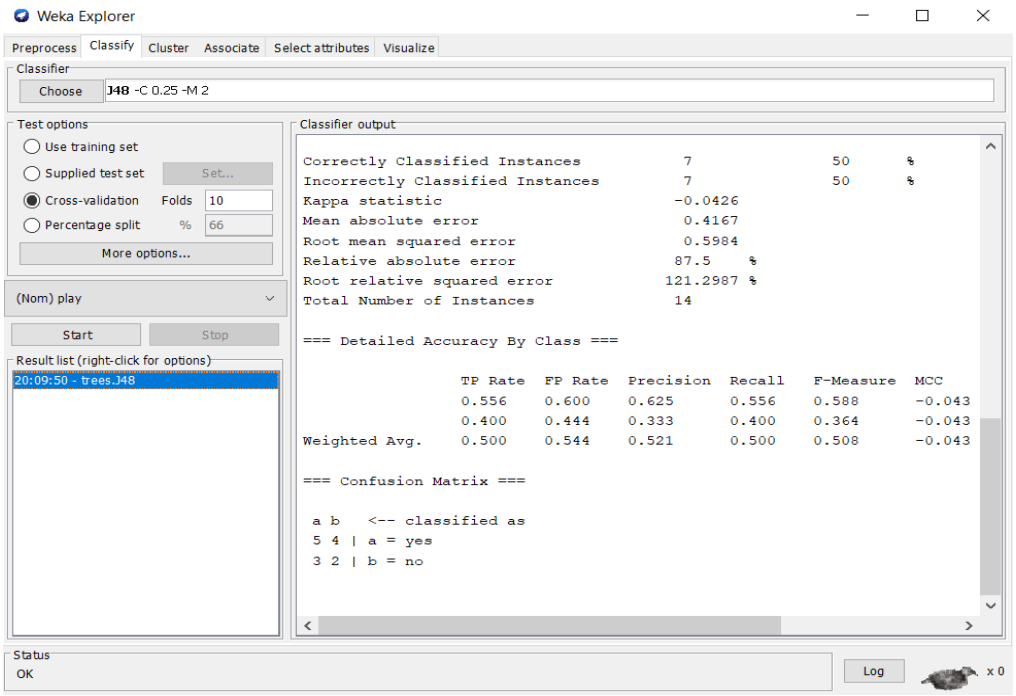
Extra credit file: weather.nominal



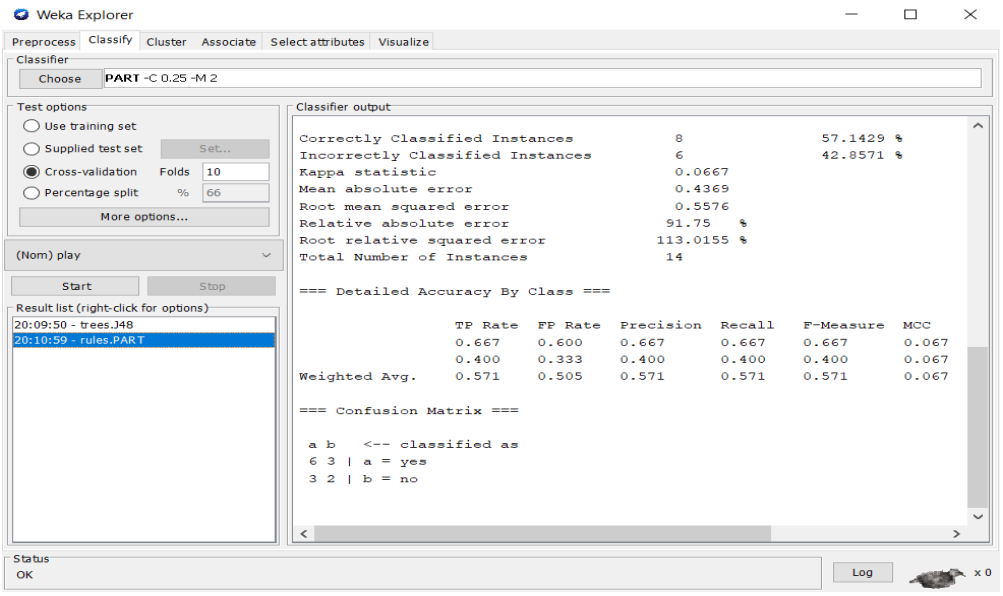
Sample Decision Tree

Converting Decision tree into a set of rules is as follows.
Rule1: If outlook = sunny AND humidity=high THEN play=no
Rule2: If outlook = sunny AND humidity=normal THEN play=yes
Rule3: If outlook = overcast THEN play=yes
Rule4: If outlook = rainy AND windy=true THEN play=no
Rule5: If outlook = rainy AND windy =false THEN play=yes

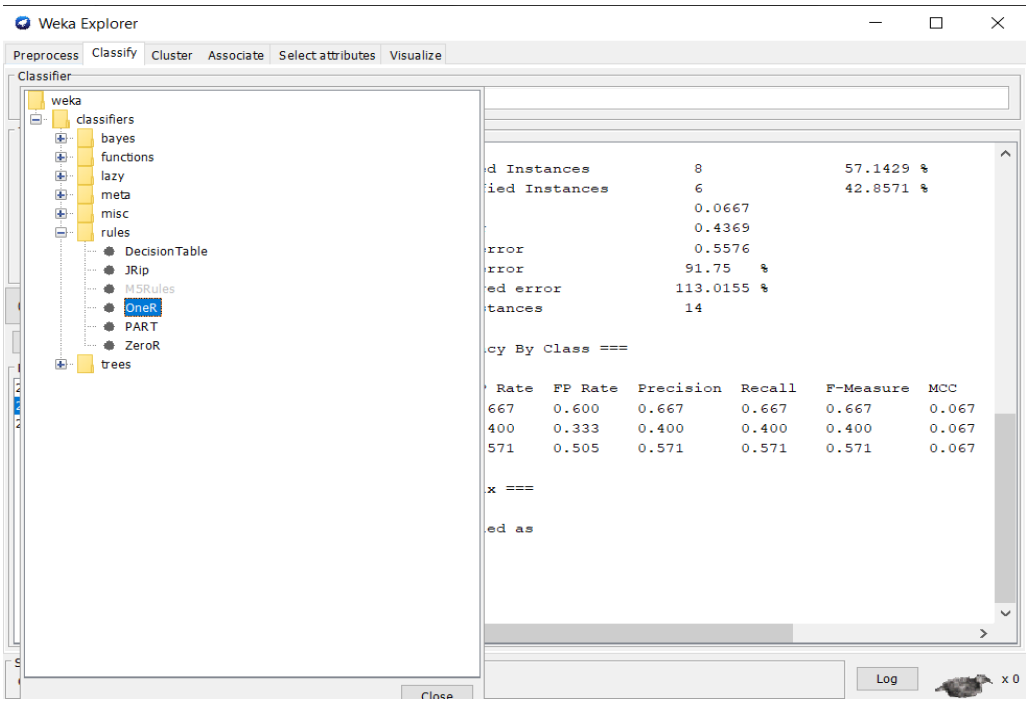
Step-1: open the weather.nominal file by selecting the open file option from weka explorer.
Step-2: select the play(class) attribute to classify the data.
Step-3: in the classify tab, choose the J48 tree from the list of trees.
Step-4: select the cross validation option in the list of test options and click on start and observe the accuracy rate and the time taken to build model.



Step-5: Go back to choose option and select the PART option from the list of rules.

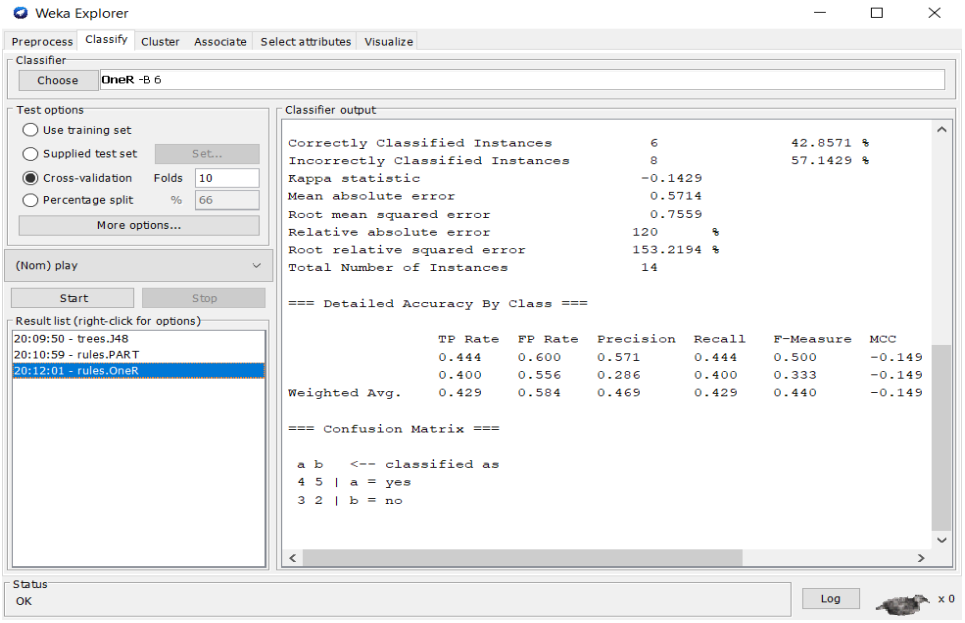


Step-6: select the cross validation option in the list of test options and click on start and observe the accuracy rate and the time taken to build model.



Step-7: Go back to choose option and select the OneR option from the list of rules.

Step-8: select the cross validation option in the list of test options and click on start and observe the accuracy rate and the time taken to build model.



In weka, rules.PART is one of the classifier which converts the decision trees into "IF-THEN-ELSE" rules.

Converting Decision trees into "IF-THEN-ELSE" rules using rules.PART classifier:-

PART decision list

outlook = overcast: yes (4.0)
windy = TRUE: no (4.0/1.0)
outlook = sunny: no (3.0/1.0): yes (3.0)
Number of Rules : 4

Yes, sometimes just one attribute can be good enough in making the decision.

In this dataset (Weather), Single attribute for making the decision is **"outlook"**

outlook:
sunny ➔ no
overcast ➔ yes
rainy ➔ yes

(10/14 instances correct) With respect to the **time**, the oneR classifier has higher ranking and J48 is in 2nd place and PART gets 3rd place.

	J48	PART	oneR
TIME(sec)	0.12	0.14	0.04
RANK	II	III	I

But if you consider the **accuracy,** The J48 classifier has higher ranking, PART gets second place and oneR gets last place.

	J48	PART	oneR
ACCURACY (%)	70.5%	70.2%	66.8%
RANK	I	II	III

Program 13: Design a Hospital Management system data warehouse(TARGET) consist of Dimensions Patient, Medicine, Time, Doctor, Treatment. Assume a Relational database (SOURCE) table schema as follows TIME(Day, Month, Year), PATIENT(Name, Age, Gender, Address), MEDICINE(Medicine_name, Price, Tablets_per_Strip), TREATMENT(Medication, Surgery), DOCTOR(Name, Specification, Experience).

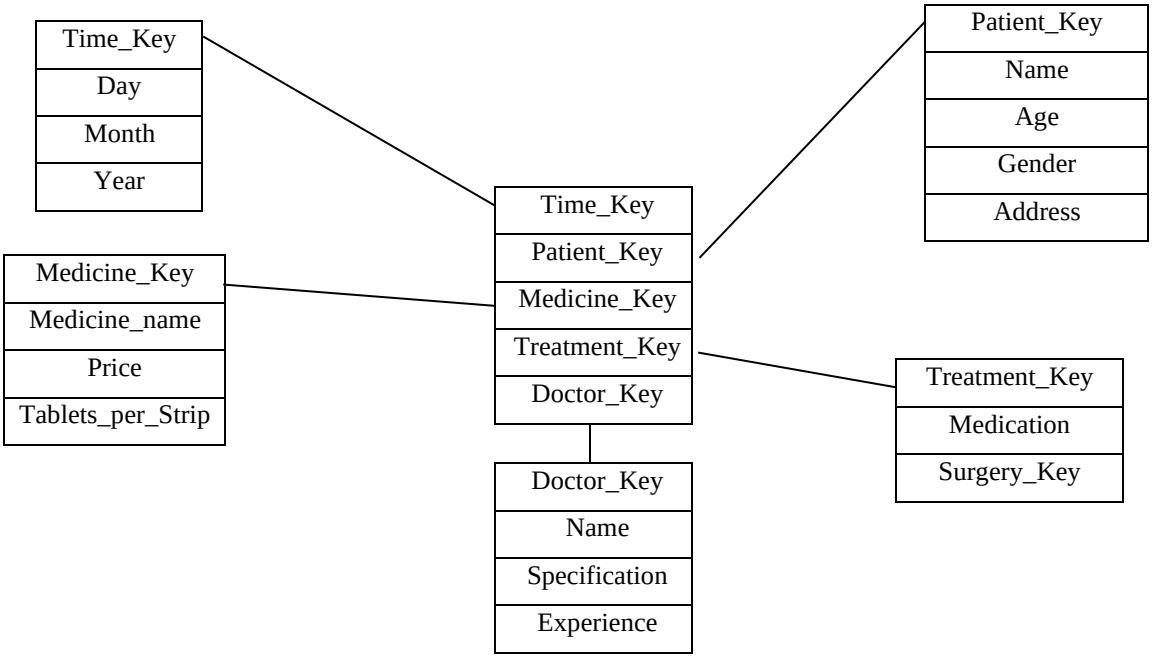
Star Schema in data warehouse, in which the center of the star can have one fact table and a number of associated dimension tables. It is known as star schema as its structure resembles a star. The Star Schema data model is the simplest type of Data Warehouse schema.

Snowflake Schema in data warehouse is a logical arrangement of tables in a multidimensional database such that the ER diagram resembles a snowflake shape. A Snowflake Schema is an extension of a Star Schema, and it adds additional dimensions.

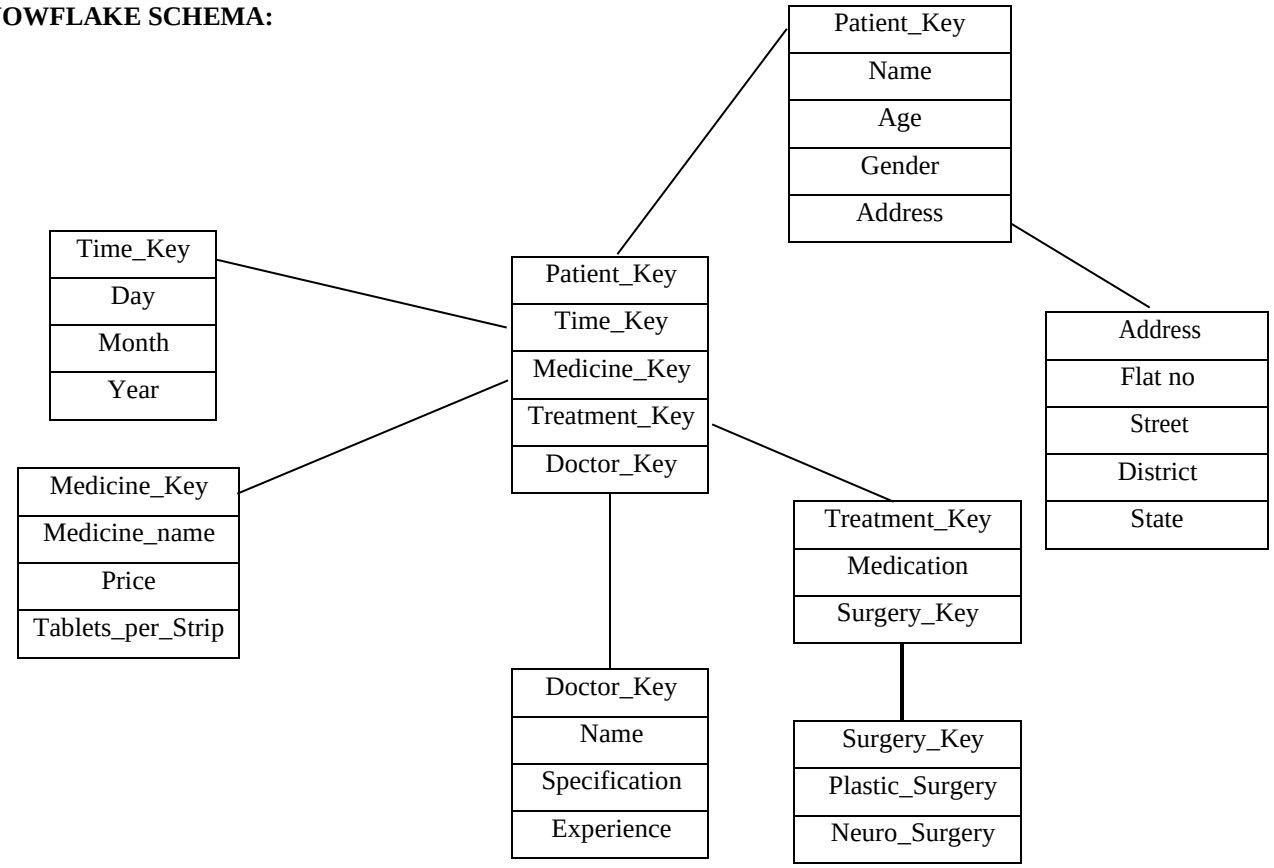
Dimension Table: Dimension tables usually have a relatively small number of records compared to fact tables, but each record may have a very large number of attributes to describe the fact data.

Fact Table: Fact tables record measurements or metrics for a specific event. Fact tables generally consist of numeric values, and foreign keys to dimensional data where descriptive information is kept.[4] Fact tables are designed to a low level of uniform detail (referred to as "granularity" or "grain"), meaning facts can record events at a very atomic level.

STAR SCHEMA REPRESENTATION:



SNOWFLAKE SCHEMA:



Program 14: One type of model that you can create is a Decision Tree - train a Decision Tree using the complete dataset(iris dataset) as the training data. Report the model obtained after training.

SOLUTION:

Iris Data set –

Step-1: Select Explorer from weak GUI chooser then click on open file option in preprocess tab.

Step-2: select the iris arff file and click on open.

Step-3: select the class attribute and go to classify tab.

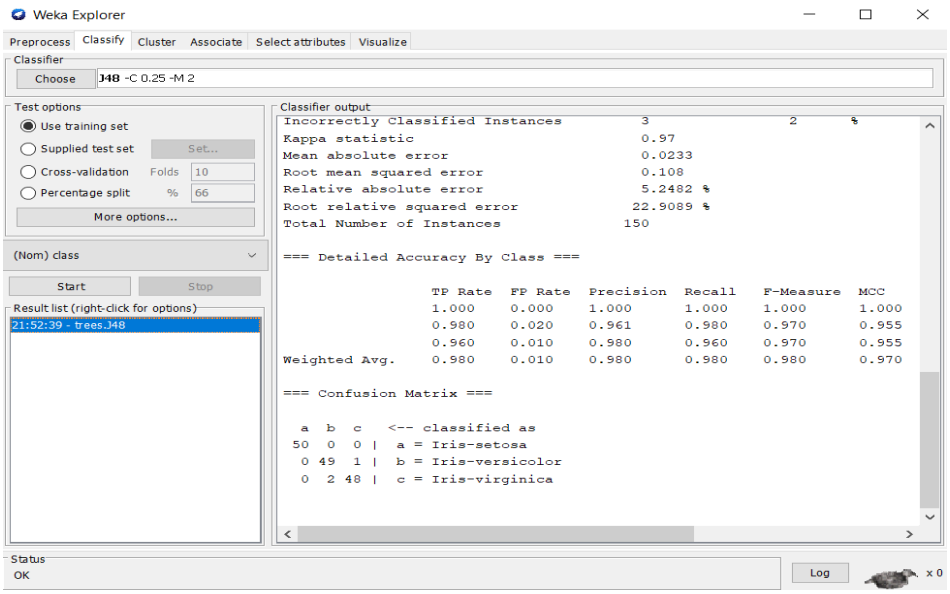
Step-4: select the use training set option from the list of test options.

Step-5: select the J48 tree from the list of trees at the choose option.

Step-6: Select class label in drop down list then click on start option.

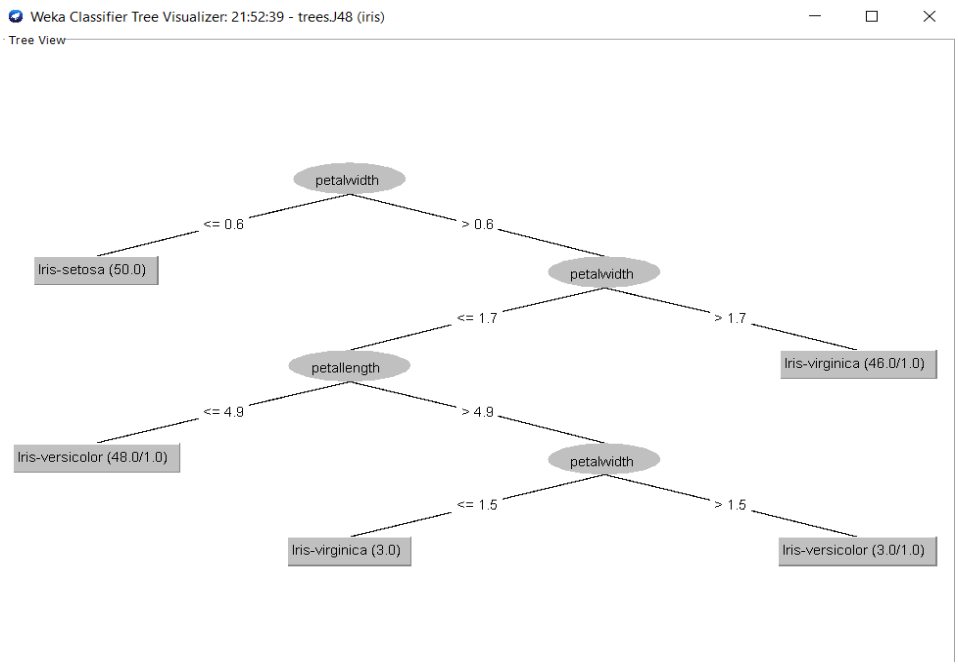
Step-7: The Accuracy results will be displayed at the right side of the window.

Outputs:



Step-8: To visualize the decision tree, right click on the result list and select the visualize tree option.

Visualize Decision Tree:



Step-9: Click on the auto scale and fit to screen options to zoom in to the decision tree.

Calculations:

=== Confusion Matrix ===
a b c <-- classified as
50 0 0 | a = Iris-setosa
0 49 1 | b = Iris-versicolor
0 2 48 | c = Iris-virginica

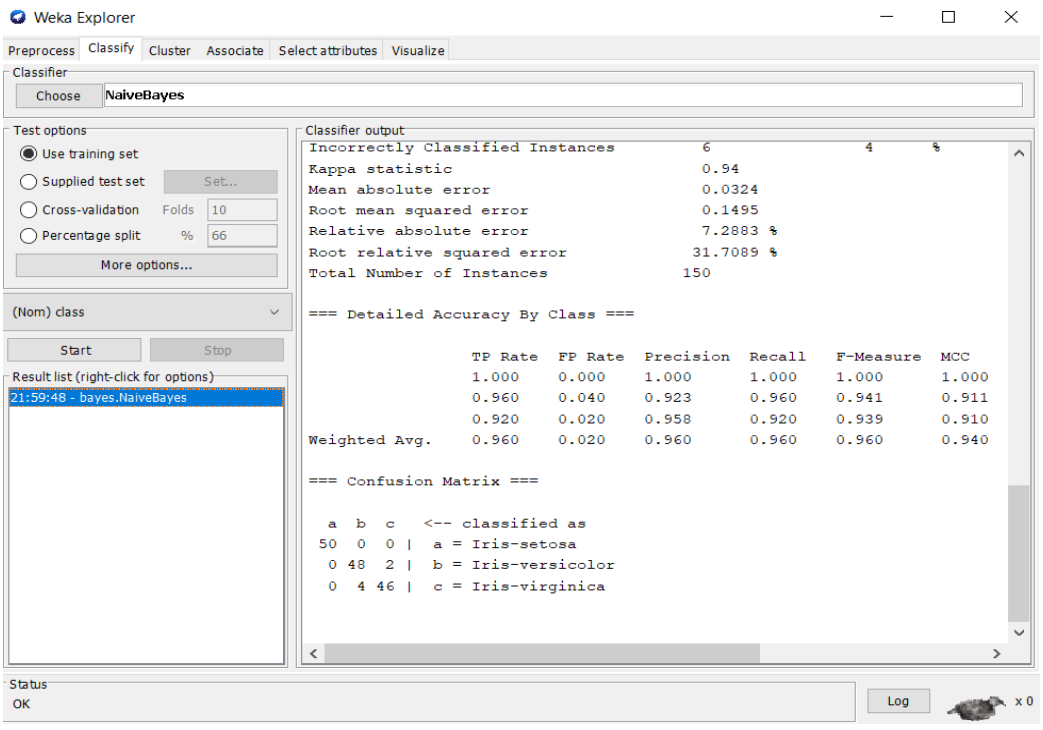
Number of Leaves : 5
Size of the tree : 9
Time taken to build model: 0.02 seconds

Program 15: One type of model that you can create is a Naïve Bayes - dataset(iris dataset) as the training data. Report the model obtained after training.

SOLUTION:

Iris Data set –

Step-1: Select Explorer from weak GUI chooser then click on open file option in preprocess tab.
Step-2: select the iris arff file and click on open.
Step-3: select the class attribute and go to classify tab.
Step-4: select the use training set option from the list of test options.
Step-5: select the NaiveBayes from the list of bayes at the choose option.
Step-6: Select class label in drop down list then click on start option.
Step-7: The Accuracy results will be displayed at the right side of the window.
Outputs:



Calculations:

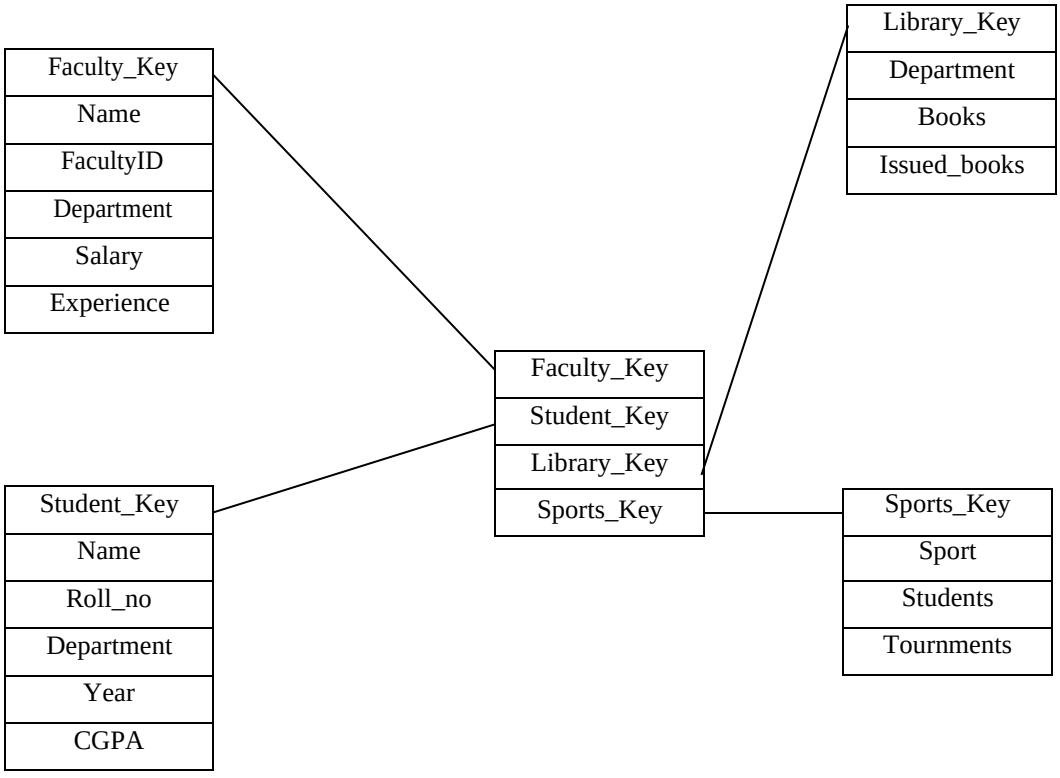
```
=== Confusion Matrix ===
a   b c <-- classified as
50 0 0 | a = Iris-setosa
0 48 2 | b = Iris-versicolor
0   4   46 | c = Iris-virginica

Number of Leaves : 5
Size of the tree : 9
Time taken to build model: 0.02 seconds
```

Program 16: Design a College Management System data warehouse (TARGET) consist of Dimensions: Sports, Library, Students, faculty. Assume the Relational database (SOURCE) table schema as follows FACULTY(Name, FacultyID, Department, Salary, Experience), STUDENT(Name, Roll_no, Department, Year, CGPA), LIBRARY(Department, books, Issued_books), SPORTS(Sport, Students, Tournments).

SOLUTION:

STAR SCHEMA REPRESENTATION:



SNOWFLAKE SCHEMA: