Dalhousie University

Heart Disease Prediction Using Data Mining Techniques

CS 6405: Data Mining Project

April 11, 2018

Submitted to:

Dr. Qigang Gao

Submitted by:

Supriya Patel, B00791627

Oyshee Saha Roy, B00790417

Contents

Abstract

Heart disease is increasing day by day in today's world due to lifestyle, heredity.[1]It is one of the leading causes of death in the world. Heart disease prediction will help to detect the disease faster and help to reduce the number of deaths. Here we will be using decision tree classifiers as well as a probability based classifier. The algorithms that we will be using are (i) Naïve Bayes classifier (ii) J48 algorithm (iii) Random forest classifier. We will be using an existing patient dataset of Cleveland from UCI repository to test and analyze the algorithms. Experiments result have been found to reach an accuracy up to 59%.

1 Introduction

Heart disease is one of the leading cause of death in the world. [2] It has been observed that about 2,300 Americans die of cardiovascular disease each day, an average of 1 death every 38 seconds.[1]Predicting Heart disease using patients databases can be compared to the real-life application. It is not possible to determine which attributes have a higher impact on disease prediction. Therefore the already available dataset can be used for diagnosing. Over the years a significant amount of patient data has been collected. Those data can be mined to find hidden patterns which can eventually help in decision making. With the alarming increase in the number of heart disease a large number of patients, researches using data mining techniques have been done using the vast amount of available data. Data mining is the process of finding hidden patterns or relationships with large sets of data, and it also helps to predict outcomes based on that analysis. Thus by building a heart disease prediction system using data mining technique can aid in early –detection or prevention of the disease. The accuracy of prediction depends on the selection of parameters to be used. Many papers have tried to do heart disease prediction using many different datasets available.

2 Background and related works

Millions of people are getting heart disease every year. [1]Medical diagnosis play a vital role. But it is expensive. Data mining technique can be used to find patterns and consistency in sets of data. Thus helping in reducing the cost of diagnosis. Data mining algorithms can be used to construct and classify the different attributes or classes. Risk factors of heart disease can be identified using factors like age, blood pressure, total cholesterol, diabetes, hypertension, family history of heart disease, obesity and lack of physical exercise, fasting blood sugar etc[3]. Different data mining techniques like Neural network, naïve Bayes, decision tree etc. are used in the diagnosis of heart disease. Improvement in accuracy of diagnosis help to improve health outcomes. [4]Decision tree is a supervised learning algorithm. The trained data can be used to predict the class to which the data belongs. The output of the decision tree is easy to understand and the execution time is less. The algorithm generates decision tree classification rules. The leaves of the decision tree are the predicted decisions. Again Probability classifier is able to predict using a probabilistic distribution over a set of classes. Decision tree in WEKA has are easy to implement. It can handle input data like Nominal, Numeric&Text. The general approach took after for Decision Tree classification for satisfying the objective is:

Training => Algorithm => Model => Testing => Evaluation.

3 Application Scenario, Solution and Theory

The application scenario referred in this project is to predict heart disease using decision tree algorithms. The study was conducted by Technique Jaymin Patel, Prof.TejalUpadhyay, Dr. Samir Patel Department of Computer Science and Engineering, Nirma University, Gujarat, India. The primary objectives set for the study was to :

1. The prediction system should not have prior knowledge of the patient records.
2. The chosen system must be scalable to run against an extensive database with thousands of data.

This study uses the open source software tool, WEKA to do prediction using different algorithms. In this study they have used three algorithms J48 algorithm, Random Forest algorithm and Logistic Model Tree Algorithm. Cleveland dataset from UCI repository has been used for the study. It is available at http://archive.ics.uci.edu/ml/datasets/Heart+Disease. The dataset has 76 attributes and 303 records. Only 13 attributes are used for the study and testing. The data mining technique used in the study is the "Classification.

"[5] Classification is a data mining function that assigns items in a collection to target categories or classes. The goal of classification is to accurately predict the target class for each case in the data. For example, a classification model could be used to identify heart disease prediction such as no presence, likely to have, surely have. [6] There are two main steps in classification.

1. Model Construction: construct a classification model based on training data.
2. Model Usage: before using the training data, the accuracy of the data is checked using test data.

Before building training data, the data cleaning is performed to remove noise and redundant data. Classification is then done on the preprocessed data using different algorithms. The rules generated using the algorithms can be used for prediction of the class labels.

4 Algorithms :

The algorithm used in the project are :

1. Naïve Bayes Classifier
2. Random forest algorithm
3. J48 algorithm

Below we will be explaining these algorithms in details:

4.1 Naïve Bayes Classifier:

4.1 Naïve Bayes Classifier: This classifier comes from the family of a simple probabilistic classifier. The general formula used for the algorithm is:

$$P(A|B) = \frac{P(A,B)}{P(B)} = \frac{P(B|A)P(A)}{P(B)}$$

Which in laymen language can be explained as,

$$\frac{\text{Likelihood} * \text{prior}}{evidence}$$

P(A|B) is the posterior probability of the class. (i.e. target attribute)

P(A) is the prior probability of the class

P(B|A) is the likelihood which is the probability of predictor given class

P(B) is the prior probability of predictor.

The classifier can be trained very efficiently in a supervised learning setting. The model uses the method of maximum likelihood. The steps to perform the algorithm :

1. Convert the data set into a frequency table
2. Create Likelihood table by finding the probabilities
3. Calculate the posterior probability for each class using Naive Bayesian equation.
4. The class with the highest posterior probability is the outcome of prediction.

4.2 Random Forest Algorithm :[1] Random forest is an ensemble classifier that consists of many decision trees. Individual trees represent the output of the classes. Higher the number of trees, higher the rate of accuracy. [9] The process of finding the root node and splitting the feature nodes will happen randomly. This classifier can handle missing values and won't overfit the model.

[5] Random Forest pseudocode:

```
1. Randomly select "k" features from total "m" features.
    a. Where k << m
2. Among the "k" features, calculate the node "d" using the best split point.
3. Split the node into daughter nodes using the best split.
4. Repeat 1 to 3 steps until "l" number of nodes has been reached.
5. Build forest by repeating steps 1 to 4 for "n" number times to create "n" number of trees.
```

Random forest prediction pseudocode:

To perform prediction using the trained random forest algorithm uses the below pseudocode.

```
1. Takes the test features and use the rules of each randomly created decision tree to predict the outcome and stores the predicted outcome (target)
2. Calculate the votes for each predicted target.
3. Consider the high voted predicted target as the final prediction from the random forest algorithm.
```

4.3 J48 algorithm :[1] J48 is an open source Java implementation of the C4.5 algorithm in the WEKA tool. The algorithm builds the tree from the dataset using the information entropy. Basic steps to construct tree are

a. Check whether all cases belong to the same class, then the tree is a leaf and is labelled with that class.
b. For each attribute, calculate the information and information gain.
c. Find the best splitting attribute (depending upon current selection criterion).

5 Data Source and Preparation :

The dataset used for the project is the Cleveland dataset of the UCI repository. The dataset is available at http://archive.ics.uci.edu/ml/machine-learning-databases/heart-disease/.

The dataset has 14 attributes and 294 records. The missing value is denoted by –9 or ?.

Selected Dataset attributes are :

Name	Type	Description
Age	Continuous	Age in years
Sex	Discrete	0=female 1 =male
cp	Discrete	Chest pain type: 1 = typical angina, 2 = atypical angina, 3 = non-anginal pain 4 =asymptom
trestbps	Continuous	Resting blood pressure (in mm Hg)
chol	Continuous	Serum cholesterol in mg/dl
fbs	Discrete	Fasting blood sugar>120 mg/dl: 1-true 0=False
restecg	Discrete	Value 0: normal Value 1: having ST-T wave abnormality (T wave inversions and/or ST elevation or depression of > 0.05 mV) Value 2: showing probable or definite

		left ventricular hypertrophy by Estes' criteria
thalach	Continuous	Maximum heart rate achieved
exang	Discrete	Exercise induced angina: 1 = Yes 0 = No
oldpeak	Continuous	Depression induced by exercise relative to rest
slope	Discrete	The slope of the peak exercise segment : 1 = up sloping 2 = flat 3 = down sloping
ca	Continuous	Number of major vessels coloured by fluoroscopy that ranged between 0 and 3.
thal	Discrete	3 = normal 6 = fixed defect 7= reversible defect
num	Discrete	Diagnosis classes: 0 = No Presence 1=Least likely to have heart disease 2= >1 3= >2 4=More likely have heart disease

5.1 Data cleaning :

The data cleaning has been implemented in Python. The code is saved in "noise.py".The following steps were followed by cleaning up the dataset :

1. Major noise is detected in the form ? or -9. These values are replaced with "NaN" values which follows IEEE standard.
2. A column-wise check is performed to check the validity of 0 value as column data. For example, then checked column-wise to see if any of the columns have 0 that should not have it. For example, age column or bp column cannot have 0 values. If such columns have 0 values, replace with "NaN."
3. All "NaN" values were replaced with dummy values.
4. Further optimization can be done using several formulas to replace these NaN values.
5. Check each row and column so that they have at least minimum valid values. Then check that each row and column should have at least some valid values. If they do not have, those rows or columns are dropped.

5.2 Training and Test Dataset: After data cleaning, the columns ca and that are dropped. The total number is 205. The training and test dataset have been split into 70% and 30% respectively.

Training dataset: clev_train

Test dataset : clev_test

In case of using the dataset in WEKA, the file name "total" was uploaded and was split there.

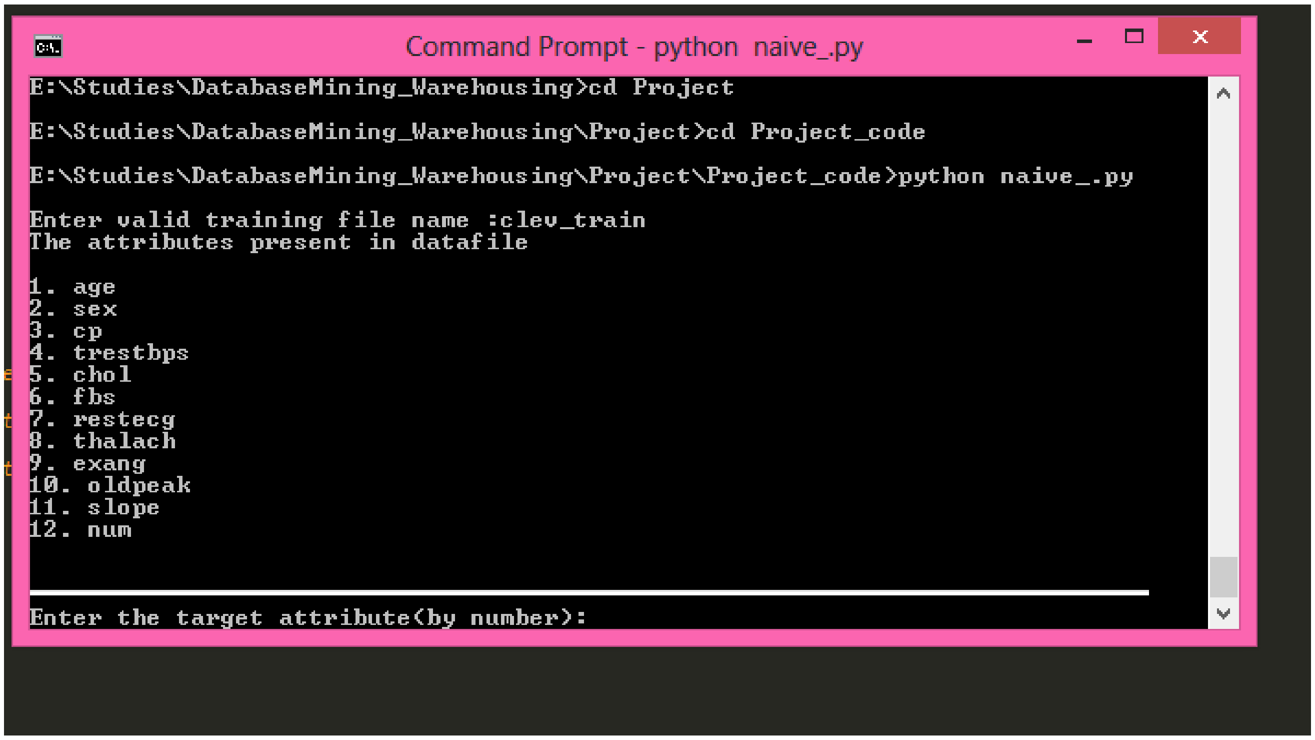
6 System Design and Implementation :

6.1 Naïve Bayes Algorithm (Implemented in Python): We have implemented the algorithm using python. The following functions were implemented. The file was saved as naïve_.py.

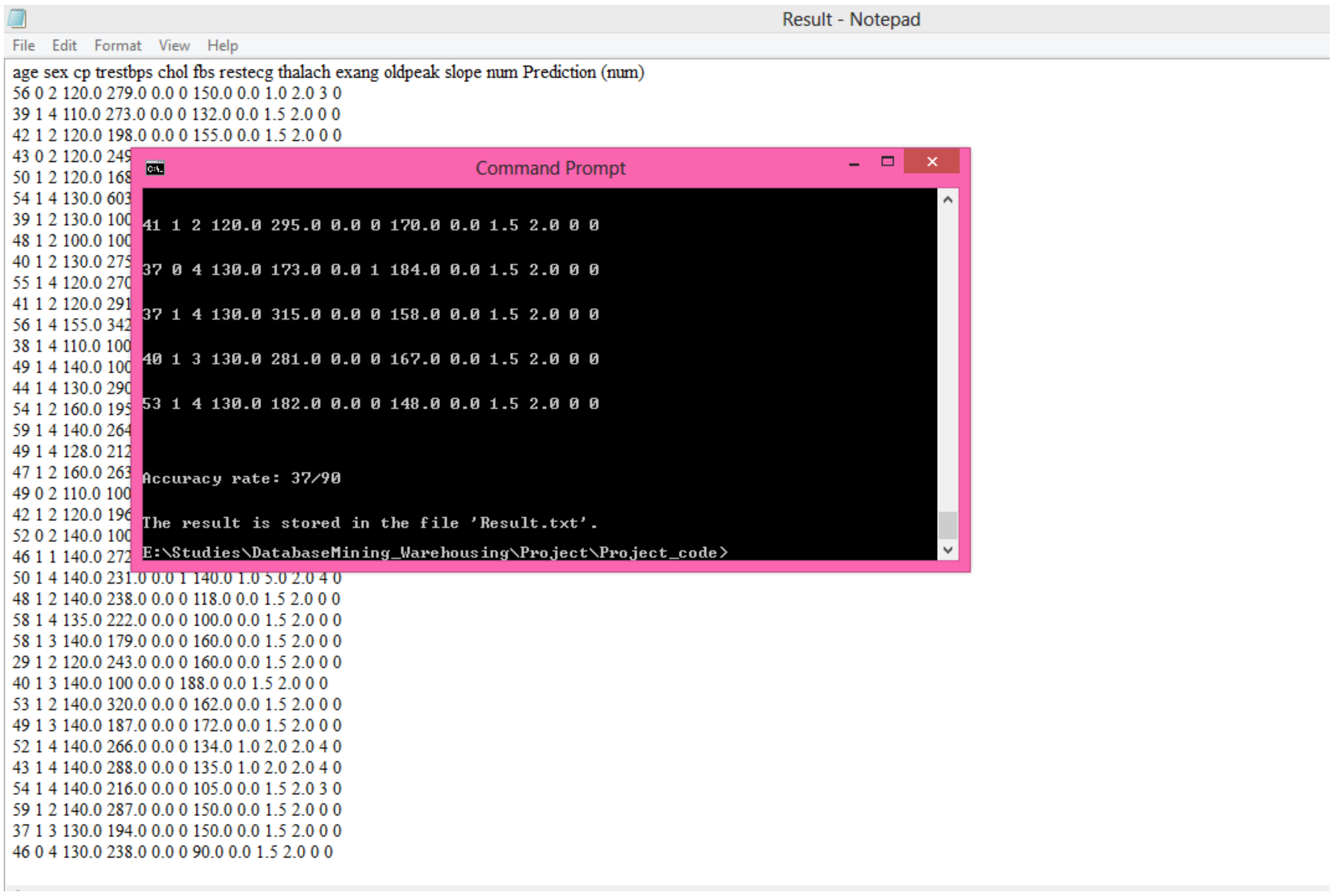
Function name	**Usage**
inputFile()	Enter training file name and check if it exists
readFile()	read training file
inputs()	take user input to get target attribute
dataCleaning()	divide dataset into target and others
calculateProbability()	calculate probability of each factor
calculatePrediction()	calculate prediction of test data
readTestfile()	read test filename
getTestdataset()	get testdata set
findProbability()	to calculate probabilty
findTarget()	unique value of target list
displayResult()	display result
calculateAccuracy()	calculate accuracy
readTestfile()	read testfile
findFactorlist()	find factor value for attribute
findUniqueList()	find unique value of list

Validations: Input filename validation for both test and training exists. Validation to check the target attribute name is also there. Probability value 0 cases have also been handled.

Demo :



```
E:\Studies\DatabaseMining_Warehousing>cd Project

E:\Studies\DatabaseMining_Warehousing\Project>cd Project_code

E:\Studies\DatabaseMining_Warehousing\Project\Project_code>python naive_.py

Enter valid training file name :clev_train
The attributes present in datafile

1. age
2. sex
3. cp
4. trestbps
5. chol
6. fbs
7. restecg
8. thalach
9. exang
10. oldpeak
11. slope
12. num


Enter the target attribute(by number):
```



```
age sex cp trestbps chol fbs restecg thalach exang oldpeak slope num Prediction (num)
56 0 2 120.0 279.0 0.0 0 150.0 0.0 1.0 2.0 3 0
39 1 4 110.0 273.0 0.0 0 132.0 0.0 1.5 2.0 0 0
42 1 2 120.0 198.0 0.0 0 155.0 0.0 1.5 2.0 0 0
43 0 2 120.0 249
50 1 2 120.0 168
54 1 4 130.0 603
39 1 2 130.0 100
48 1 2 100.0 100
40 1 2 130.0 275
55 1 4 120.0 270
41 1 2 120.0 291
56 1 4 155.0 342
38 1 4 110.0 100
49 1 4 140.0 100
44 1 4 130.0 290
54 1 2 160.0 195
59 1 4 140.0 264
49 1 4 128.0 212
47 1 2 160.0 263
49 0 2 110.0 100
42 1 2 120.0 196
52 0 2 140.0 100
46 1 1 140.0 272
50 1 4 140.0 231.0 0.0 1 140.0 1.0 5.0 2.0 4 0
48 1 2 140.0 238.0 0.0 0 118.0 0.0 1.5 2.0 0 0
58 1 4 135.0 222.0 0.0 0 100.0 0.0 1.5 2.0 0 0
58 1 3 140.0 179.0 0.0 0 160.0 0.0 1.5 2.0 0 0
29 1 2 120.0 243.0 0.0 0 160.0 0.0 1.5 2.0 0 0
40 1 3 140.0 100 0.0 0 188.0 0.0 1.5 2.0 0 0
53 1 2 140.0 320.0 0.0 0 162.0 0.0 1.5 2.0 0 0
49 1 3 140.0 187.0 0.0 0 172.0 0.0 1.5 2.0 0 0
52 1 4 140.0 266.0 0.0 0 134.0 1.0 2.0 2.0 4 0
43 1 4 140.0 288.0 0.0 0 135.0 1.0 2.0 2.0 4 0
54 1 4 140.0 216.0 0.0 0 105.0 0.0 1.5 2.0 3 0
59 1 2 140.0 287.0 0.0 0 150.0 0.0 1.5 2.0 0 0
37 1 3 130.0 194.0 0.0 0 150.0 0.0 1.5 2.0 0 0
46 0 4 130.0 238.0 0.0 0 90.0 0.0 1.5 2.0 0 0
```

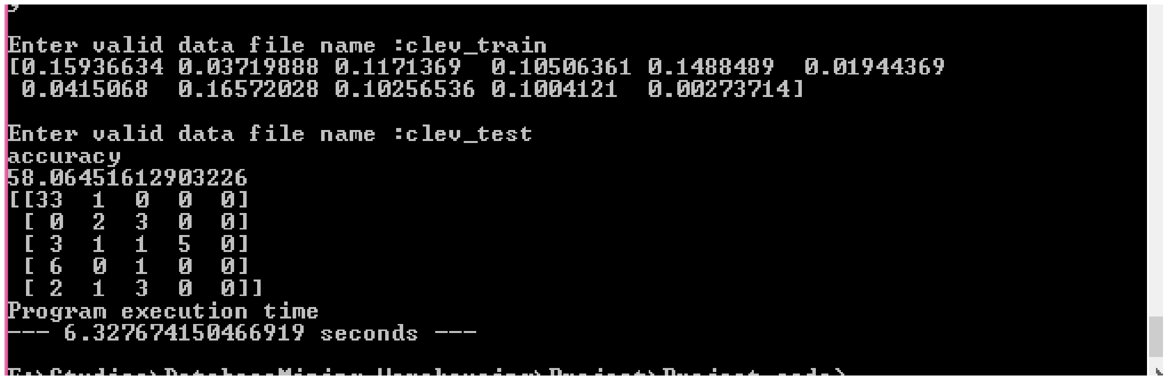
```
41 1 2 120.0 295.0 0.0 0 170.0 0.0 1.5 2.0 0 0

37 0 4 130.0 173.0 0.0 1 184.0 0.0 1.5 2.0 0 0

37 1 4 130.0 315.0 0.0 0 158.0 0.0 1.5 2.0 0 0

40 1 3 130.0 281.0 0.0 0 167.0 0.0 1.5 2.0 0 0

53 1 4 130.0 182.0 0.0 0 148.0 0.0 1.5 2.0 0 0


Accuracy rate: 37/90

The result is stored in the file 'Result.txt'.
E:\Studies\DatabaseMining_Warehousing\Project\Project_code>
```

6.2 Random forest Algorithm(Implemented in Python): The program has been implemented sklearn library using python. The functions used :

Functions	Usage
InputFile	input training file name
ReadFile	read training dataset
Classifier	implement the classifier; read test file
Confusionmatrix	print the confusion matrix
InputTestfile	input test file name

Validations: Input filename validation for both test and training exists. Validation to check the target attribute name is also there.

Output :



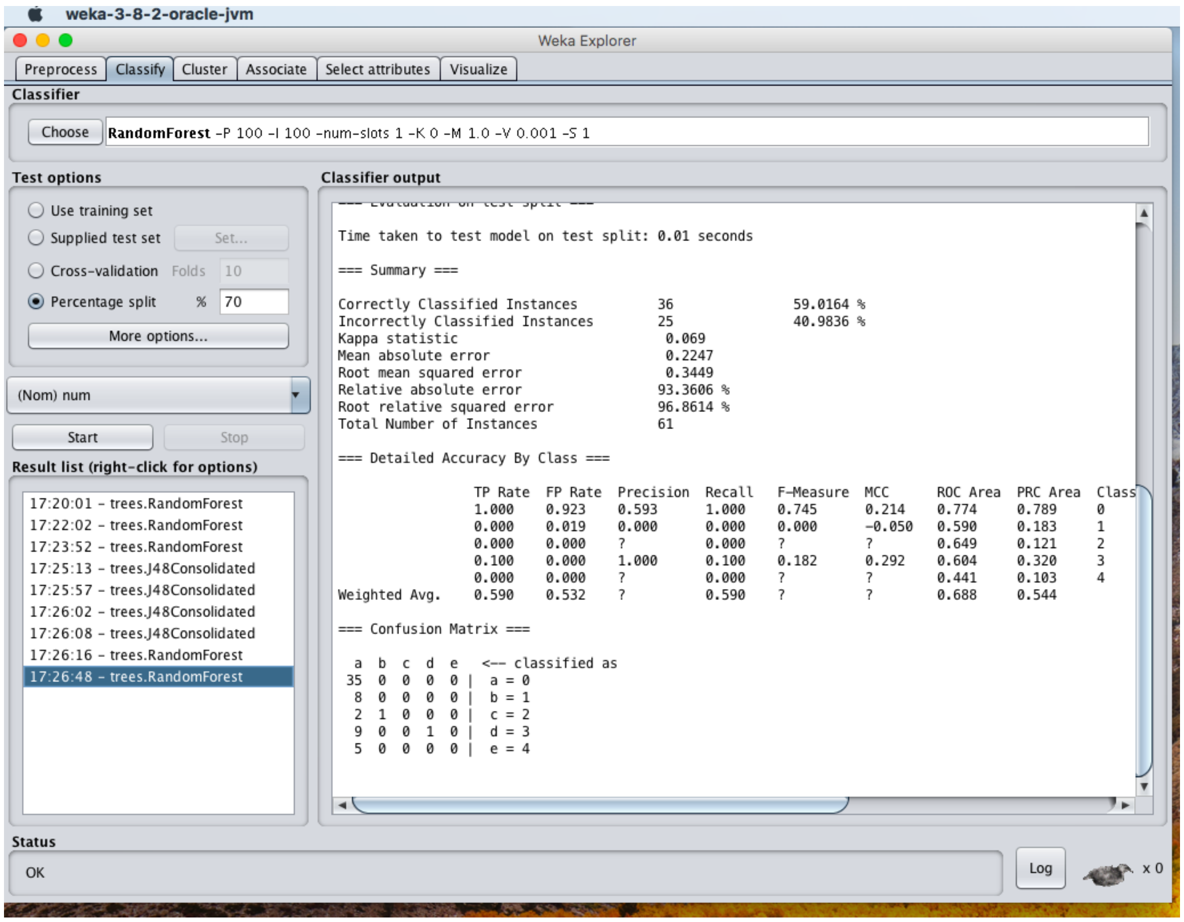
```
Enter valid data file name :clev_train
[0.15936634 0.03719888 0.1171369  0.10506361 0.1488489  0.01944369
 0.0415068  0.16572028 0.10256536 0.1004121  0.00273714]

Enter valid data file name :clev_test
accuracy
58.06451612903226
[[33  1  0  0  0]
 [ 0  2  3  0  0]
 [ 3  1  1  5  0]
 [ 6  0  1  0  0]
 [ 2  1  3  0  0]]
Program execution time
--- 6.327674150466919 seconds ---
```

7 Results in WEKA :

The three algorithms have been implemented in Weka. The steps to execute the algorithms in WEKA are as follows :

1. Start the WEKA Explorer.
2. Uploaded the dataset file.
3. Select filter- numerical to nominal for attribute filtering.
4. select the classifier
5. chose the slitting option(70%)
6. Click on start.

Weka Output for Random Forest



Classifier: RandomForest -P 100 -I 100 -num-slots 1 -K 0 -M 1.0 -V 0.001 -S 1

Test options: Cross-validation Folds 10 [✓ Percentage split % 70]

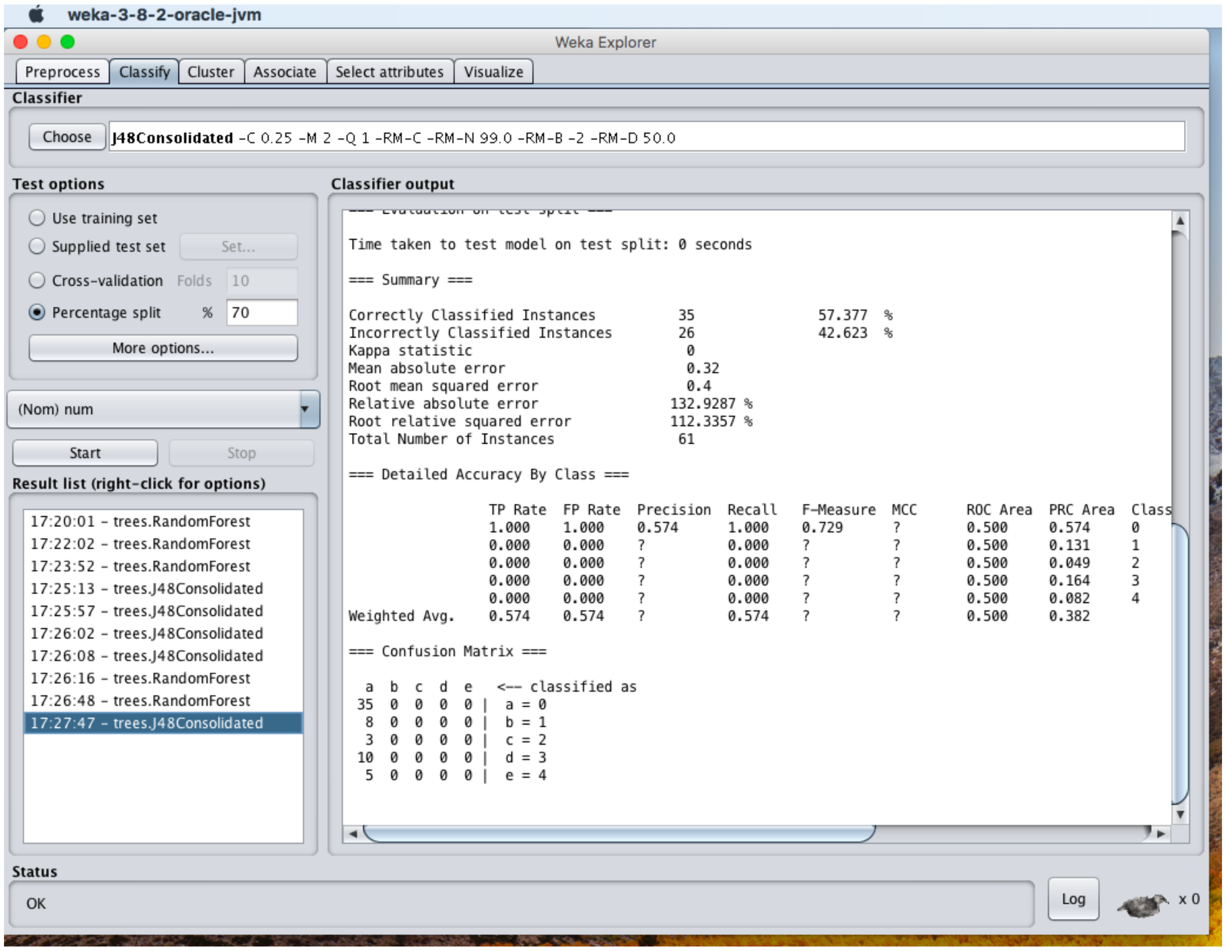
(Nom) num

Result list:
17:20:01 - trees.RandomForest
17:22:02 - trees.RandomForest
17:23:52 - trees.RandomForest
17:25:13 - trees.J48Consolidated
17:25:57 - trees.J48Consolidated
17:26:02 - trees.J48Consolidated
17:26:08 - trees.J48Consolidated
17:26:16 - trees.RandomForest
17:26:48 - trees.RandomForest

```
Time taken to test model on test split: 0.01 seconds

=== Summary ===

Correctly Classified Instances          36               59.0164 %
Incorrectly Classified Instances        25               40.9836 %
Kappa statistic                          0.069
Mean absolute error                      0.2247
Root mean squared error                  0.3449
Relative absolute error                 93.3606 %
Root relative squared error             96.8614 %
Total Number of Instances               61

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.923    0.593      1.000    0.745      0.214    0.774     0.789     0
                 0.000    0.019    0.000      0.000    0.000      -0.050   0.590     0.183     1
                 0.000    0.000    ?          0.000    ?          ?        0.649     0.121     2
                 0.100    0.000    1.000      0.100    0.182      0.292    0.604     0.320     3
                 0.000    0.000    ?          0.000    ?          ?        0.441     0.103     4
Weighted Avg.    0.590    0.532    ?          0.590    ?          ?        0.688     0.544

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
 35  0  0  0  0 |  a = 0
  8  0  0  0  0 |  b = 1
  2  1  0  0  0 |  c = 2
  9  0  0  1  0 |  d = 3
  5  0  0  0  0 |  e = 4
```

Status: OK

Weka Output for J48



J48Consolidated -C 0.25 -M 2 -Q 1 -RM-C -RM-N 99.0 -RM-B -2 -RM-D 50.0

Test options:
- ( ) Use training set
- ( ) Supplied test set — Set...
- ( ) Cross-validation — Folds 10
- (●) Percentage split — % 70
- More options...

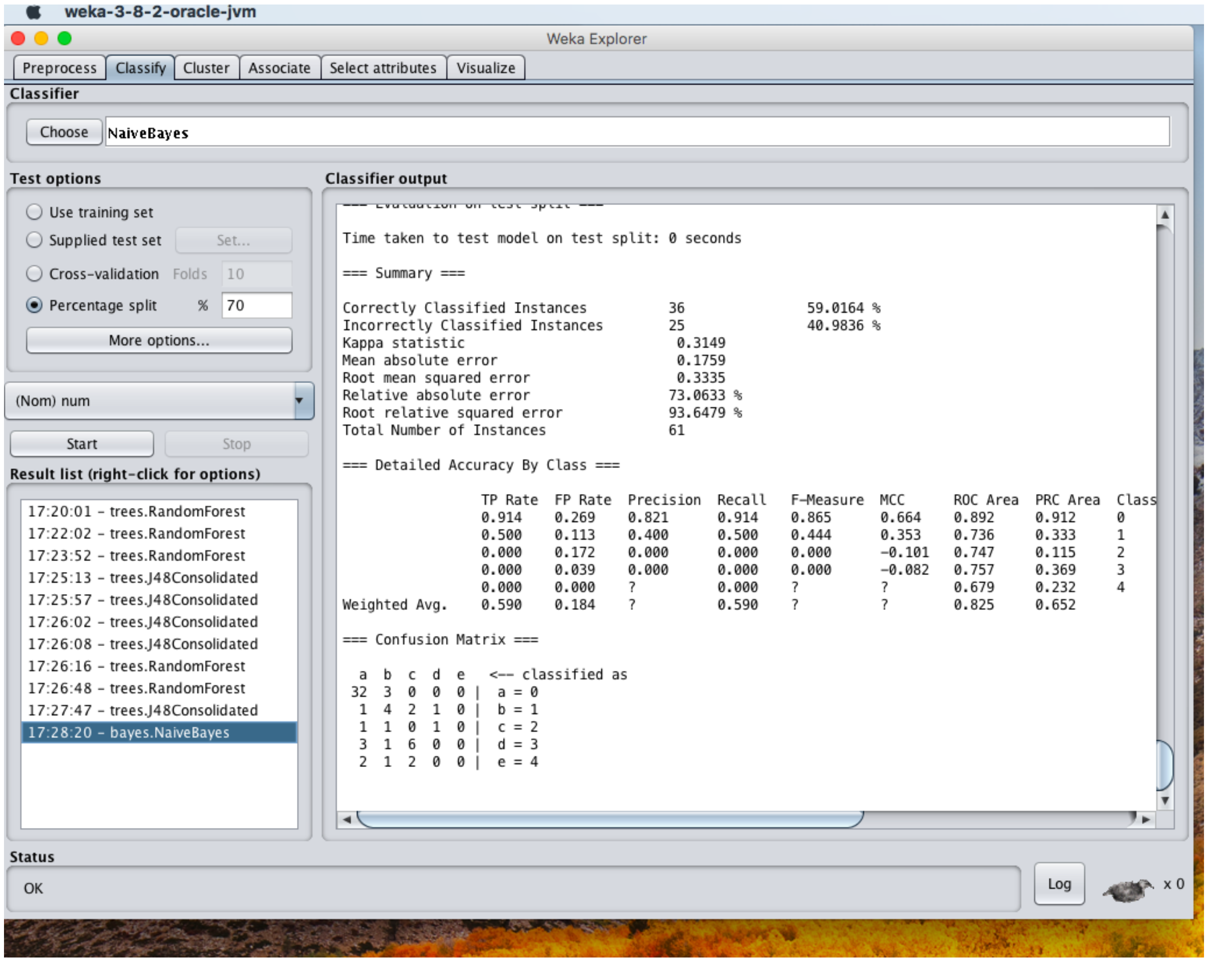
(Nom) num

Result list (right-click for options)
- 17:20:01 - trees.RandomForest
- 17:22:02 - trees.RandomForest
- 17:23:52 - trees.RandomForest
- 17:25:13 - trees.J48Consolidated
- 17:25:57 - trees.J48Consolidated
- 17:26:02 - trees.J48Consolidated
- 17:26:08 - trees.J48Consolidated
- 17:26:16 - trees.RandomForest
- 17:26:48 - trees.RandomForest
- 17:27:47 - trees.J48Consolidated

```
Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances          35               57.377  %
Incorrectly Classified Instances        26               42.623  %
Kappa statistic                          0
Mean absolute error                      0.32
Root mean squared error                  0.4
Relative absolute error                132.9287 %
Root relative squared error            112.3357 %
Total Number of Instances               61

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    1.000    0.574      1.000    0.729      ?        0.500     0.574     0
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.131     1
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.049     2
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.164     3
                 0.000    0.000    ?          0.000    ?          ?        0.500     0.082     4
Weighted Avg.    0.574    0.574    ?          0.574    ?          ?        0.500     0.382

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
 35  0  0  0  0 |  a = 0
  8  0  0  0  0 |  b = 1
  3  0  0  0  0 |  c = 2
 10  0  0  0  0 |  d = 3
  5  0  0  0  0 |  e = 4
```

OK

Weka Output for Naïve Bayes



Classifier: NaiveBayes

Test options: Percentage split [✓] % 70

Result list:

```
17:20:01 - trees.RandomForest
17:22:02 - trees.RandomForest
17:23:52 - trees.RandomForest
17:25:13 - trees.J48Consolidated
17:25:57 - trees.J48Consolidated
17:26:02 - trees.J48Consolidated
17:26:08 - trees.J48Consolidated
17:26:16 - trees.RandomForest
17:26:48 - trees.RandomForest
17:27:47 - trees.J48Consolidated
17:28:20 - bayes.NaiveBayes
```

```
Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances          36               59.0164 %
Incorrectly Classified Instances        25               40.9836 %
Kappa statistic                          0.3149
Mean absolute error                      0.1759
Root mean squared error                  0.3335
Relative absolute error                 73.0633 %
Root relative squared error             93.6479 %
Total Number of Instances               61

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.914    0.269    0.821      0.914    0.865      0.664    0.892     0.912     0
                 0.500    0.113    0.400      0.500    0.444      0.353    0.736     0.333     1
                 0.000    0.172    0.000      0.000    0.000      -0.101   0.747     0.115     2
                 0.000    0.039    0.000      0.000    0.000      -0.082   0.757     0.369     3
                 0.000    0.000    ?          0.000    ?          ?        0.679     0.232     4
Weighted Avg.    0.590    0.184    ?          0.590    ?          ?        0.825     0.652

=== Confusion Matrix ===

  a  b  c  d  e   <-- classified as
 32  3  0  0  0 |  a = 0
  1  4  2  1  0 |  b = 1
  1  1  0  1  0 |  c = 2
  3  1  6  0  0 |  d = 3
  2  1  2  0  0 |  e = 4
```

8 Performance Evaluation :

Algorithm	Research Paper	Our Result
Naïve Bayes	-	41%(python) ; 59.01%(weka)
Random Forest	Between 55-57%(weka)	58.06%(python); 59.06%(weka)
J48	56.76%(weka)	57.37%(weka)

	Time taken to test model on test split up
Naïve Bayes	0 sec
Random Forest	0.01 sec
J48	0 sec

	Mean Absolute Error
Naïve Bayes	0.17
Random Forest	0.22
J48	0.32

So from our evaluation, it can be easily seen that Naïve Bayes is faster and more efficient among these algorithms followed by random forest and then J48.

9 Conclusion & Future Work :

[1]By analyzing the experimental results, it is concluded that Naïve Bayes technique turned out to be the best classifier for heart disease prediction because it contains more accuracy and least time taken to test model on test split up. We can see that highest accuracy belongs to Random forest algorithm followed by Naïve Bayes algorithm and J48 algorithm respectively. The best algorithm Random forest algorithm based on UCI data has the highest accuracy, i.e. 59.06%(weka) 58.06%(python), while J48 algorithm has the lowest accuracy, i.e. 57.37 %.In conclusion, although we have received a better accuracy rate than the referred paper, there is a need for combinational and more complex models to increase the accuracy of predicting the early onset of heart disease.

Due to time limitation, different other data mining techniques were not approached for a better result. Different rules such as Association, Clustering, K-means can be taken into consideration for future work.

12 References :

[1] Heart Disease Prediction Using Machine learning and Data Mining Technique Jaymin Patel, Prof.TejalUpadhyay, Dr. Samir Patel Department of Computer Science and Engineering, Nirma University, Gujarat, India Principal, Grow More Faculty of Engineering, Ahmedabad, Gujarat, India

[2] https://healthmetrics.heart.org/wp-content/uploads/2018/02/At-A-Glance-Heart-Disease-and-Stroke-Statistics-2018.pdf

[3] [3] Y. E. Shao, C.-D. Hou, and C.-C. Chiu, "Hybrid intelligent modelling schemes for heart disease classification," Applied Soft Computing,vol. 14, pp. 47–52, 2014.

[4] https://www.analyticsvidhya.com/blog/2015/01/decision-tree-simplified/2/

[5] https://medium.com/@Synced/how-random-forest-algorithm-works-in-machine-learning-3c0fe15b6674

[6] http://www.inf.unibz.it/dis/teaching/DWDM/slides2012/lesson9-Classification1.pdf

[7] https://www.cs.cmu.edu/~epxing/Class/10701-08s/Lecture/lecture3.pdf

[8] https://www.analyticsvidhya.com/blog/2017/09/naive-bayes-explained/

[9] http://dataaspirant.com/2017/05/22/random-forest-algorithm-machine-learing/

[10] https://docs.oracle.com/cd/B28359_01/datamine.111/b28129/classify.htm#i1005746

13 Appendixes :

Appendix A : Example ofDataset

age sex cp trestbps chol fbs restecg thalach exang oldpeak slope num

40 1 2 140.0 289.0 0.0 0 172.0 0.0 1.5 2.0 0

49 0 3 160.0 180.0 0.0 0 156.0 0.0 1.0 2.0 1

37 1 2 130.0 283.0 0.0 1 98.0 0.0 1.5 2.0 0

48 0 4 138.0 214.0 0.0 0 108.0 1.0 1.5 2.0 3

54 1 3 150.0 100 0.0 0 122.0 0.0 1.5 2.0 0

39 1 3 120.0 339.0 0.0 0 170.0 0.0 1.5 2.0 0

45 0 2 130.0 237.0 0.0 0 170.0 0.0 1.5 2.0 0

54 1 2 110.0 208.0 0.0 0 142.0 0.0 1.5 2.0 0

37 1 4 140.0 207.0 0.0 0 130.0 1.0 1.5 2.0 1

48 0 2 120.0 284.0 0.0 0 120.0 0.0 1.5 2.0 0

37 0 3 130.0 211.0 0.0 0 142.0 0.0 1.5 2.0 0

58 1 2 136.0 164.0 0.0 1 99.0 1.0 2.0 2.0 3

39 1 2 120.0 204.0 0.0 0 145.0 0.0 1.5 2.0 0

49 1 4 140.0 234.0 0.0 0 140.0 1.0 1.0 2.0 3

42 0 3 115.0 211.0 0.0 1 137.0 0.0 1.5 2.0 0

54 0 2 120.0 273.0 0.0 0 150.0 0.0 1.5 2.0 0

38 1 4 110.0 196.0 0.0 0 166.0 0.0 1.5 2.0 1

43 0 2 120.0 201.0 0.0 0 165.0 0.0 1.5 2.0 0

60 1 4 100.0 248.0 0.0 0 125.0 0.0 1.0 2.0 1

36 1 2 120.0 267.0 0.0 0 160.0 0.0 3.0 2.0 1

43 0 1 100.0 223.0 0.0 0 142.0 0.0 1.5 2.0 0

44 1 2 120.0 184.0 0.0 0 142.0 0.0 1.0 2.0 0

49 0 2 124.0 201.0 0.0 0 164.0 0.0 1.5 2.0 0

44 1 2 150.0 288.0 0.0 0 150.0 1.0 3.0 2.0 3

40 1 3 130.0 215.0 0.0 0 138.0 0.0 1.5 2.0 0

36 1 3 130.0 209.0 0.0 0 178.0 0.0 1.5 2.0 0