CPEN405 - ARTIFICIAL INTELLIGENCE COURSE PROJECT - GROUP 13

Adjei-Twum Emmanuel Sefa - 10717695
Twum Penuel Antwi - 10712173
Rebecca Seglah Sesinam - 10722022
Benjamin Powell - 10735628
Akesse Stephen King - 10732467
Osafo Kwadwo Agyeman - 10724754

June 13, 2022

1 INTRODUCTION

Artificial Intelligence is the ability of a digital computer or computer-controlled robot to perform tasks commonly associated with intelligent beings. The term is frequently applied to the project of developing systems endowed with the intellectual processes characteristics of humans such as the ability to reason, discover meaning, generalize or learn from the past. From the Oxford dictionary, Intelligence is defined as the ability to acquire and apply knowledge and skills. Psychologists generally do not characterize human intelligence by just one trait but by the combination of many diverse ability. Research in AI has focused chiefly on the following components of intelligence: learning, reasoning, problem solving, perception and using language. These are the definitive components that have helped shape the face of modern artificial intelligence. The aim of this project is to challenge ourselves to some problems in the fields of machine learning, natural language processing, knowledge representation, optimization and game theory.

2 PART 1 – MACHINE LEARNING USING WEKA

2.1 OBJECTIVES

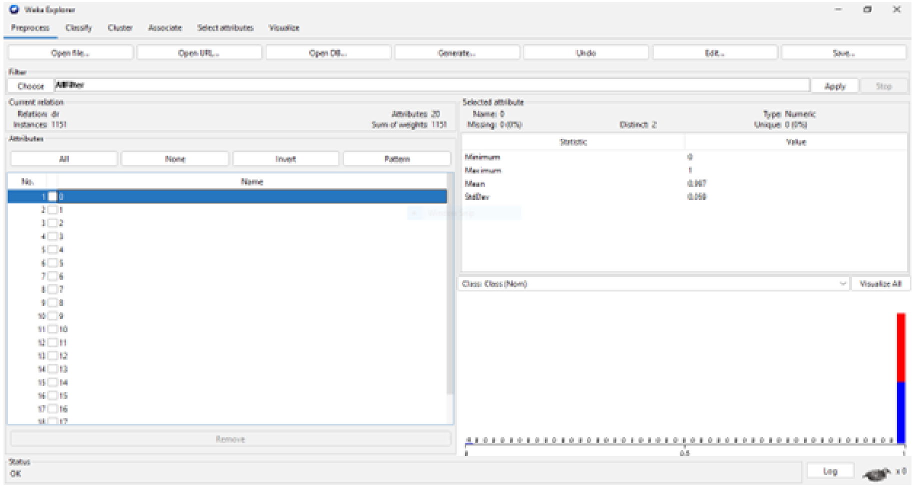
The aim of this is to have some hands-on experience with machine learning algorithms using WEKA for solving real world data-mining problems.

2.2 PROBLEM FORMULATION

To evaluate and analyze the performance of at least three classification schemes on a Diabetic Retinopathy Debrecen data set. The data set was obtained from the open-source UCI Machine Learning Repository. The data set contains features extracted from the Messidor image set to predict whether an image contains signs of diabetic retinopathy or not.

2.3 SOLUTION APPROACH AND ALGORITHMS

The Diabetic Retinopathy Debrecen data set has 1151 instances and 20 attributes as shown in figure 1 below.



Filter: AllFilter

Current relation
Relation: dr
Instances: 1151
Attributes: 20
Sum of weights: 1151

Attributes

No.		Name
1	□	0
2	□	1
3	□	2
4	□	3
5	□	4
6	□	5
7	□	6
8	□	7
9	□	8
10	□	9
11	□	10
12	□	11
13	□	12
14	□	13
15	□	14
16	□	15
17	□	16
18	□	17

Selected attribute
Name: 0
Missing: 0 (0%)
Distinct: 2
Type: Numeric
Unique: 0 (0%)

Statistic	Value
Minimum	0
Maximum	1
Mean	0.997
StdDev	0.059

Class: Class (Nom)

Status
OK

For a data set with over 1000 instances, 2/3 of the data set was used for training and 1/3 was used for testing. The classification schemes applied to the data set were J48 decision tree classification algorithm, the RepTree classification algorithm, Naives Bayes classification algorithm and KNN algorithm. A brief definition of the chosen algorithms and reasons they were chosen is given below.

2.3.1 J48 TREE CLASSIFICATION ALGORITHM

The J48 classification algorithm is one of the best machine learning algorithms to examine the data categorically and continuously. When it is used for instance purposes, it occupies more memory space and depletes the performance and accuracy in classifying medical data.

2.3.2 REDUCED ERROR PRUNING TREE CLASSIFICATION ALGORITHM

Reduced Error Pruning (RepTree) is a fast decision tree learner that builds a decision or regression tree using information gain as the splitting criterion and prunes it using reduced error pruning algorithm. As traversing over the internal nodes from the bottom to the top of a tree, the REP procedure checks for each internal node, whether replacing it with the most repeated class that does not reduce the accuracy of trees. In this case, the node is pruned. The procedure continues until any further pruning will decrease the accuracy.

2.3.3 NAÏVE BAYES CLASSIFICATION ALGORITHM

Naïve Bayes classification calculates explicit probabilities for superposition among the most practical approaches to definite types of learning problems. Even when Bayesian techniques are computationally difficult, they can produce standard and optimal decision making against all other methods of classification.

2.3.4 KNN CLASSIFICATION ALGORITHM

K-nearest neighbor algorithm is a simple, supervised machine learning algorithm that can be used to solve both classification and regression problems. To classify a new instance it searches the training set for one that is “most like” it. KNN algorithm decides a number k which is the nearest neighbor to the data point that is to be classified. If the value of k is 5 it will look for 5 nearest neighbors to that data point. It is easy to implement and understand but has a major drawback of becoming significantly slow as the size of data grows.

2.4 RESULTS AND DISCUSSION

Although having a high percentage of correctly classified instances is good, that alone cannot accurately determine if the model used is good. You can have high classification accuracy and a bad model. For instance, if you have 90 percent of instances all in one class you can say all of them belong to that class. And you’re going to be right 90 percent of the time but the model is not good.

This section discusses and analyzes the statistics of the output of the models of the different algorithms that were applied and the changes in the results as the parameters were varied. In order to do that there are certain concepts we need to understand.

Kappa statistic takes into account the fact that you could randomly guess and correctly classify an instance. It is a statistic of how well you would perform if you take into consideration the fact that you can randomly guess the class of an instance

Kappa = (observed accuracy - expected accuracy)/(1 - expected accuracy

Weighted average gives a general average of how well the algorithm is doing.

True positive is the percentage of classified instances that actually belong to a particular class.

False positive is the percentage of classified instances that do not actually belong to a particular class.

Precision is the ratio of true positives to total data classified as a particular class.

Recall is ratio of true positives to total data that actually belongs to a particular class, say class 0

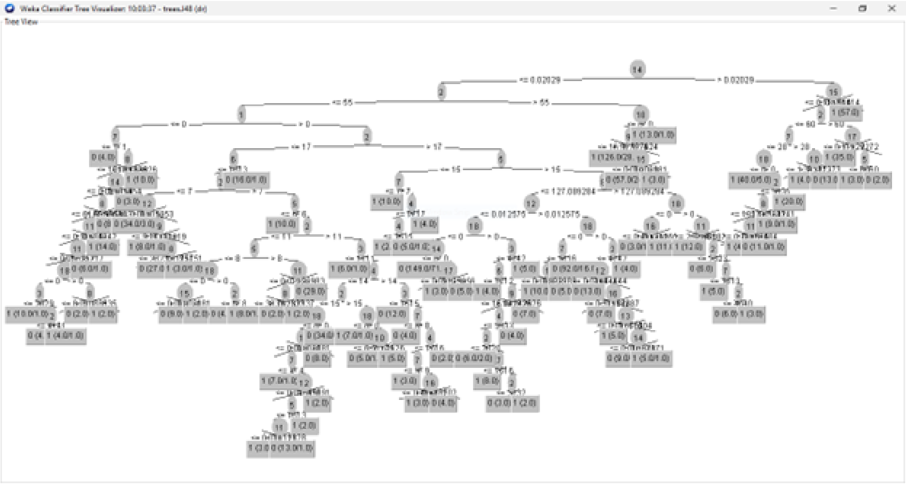
ROC area is the receiver operator characteristic area under the curve. It shows what percentage of the time you are going to correctly classify items if given one of each class.

PRC area is the precision and recall area and it does not account for true positives in the statistics.

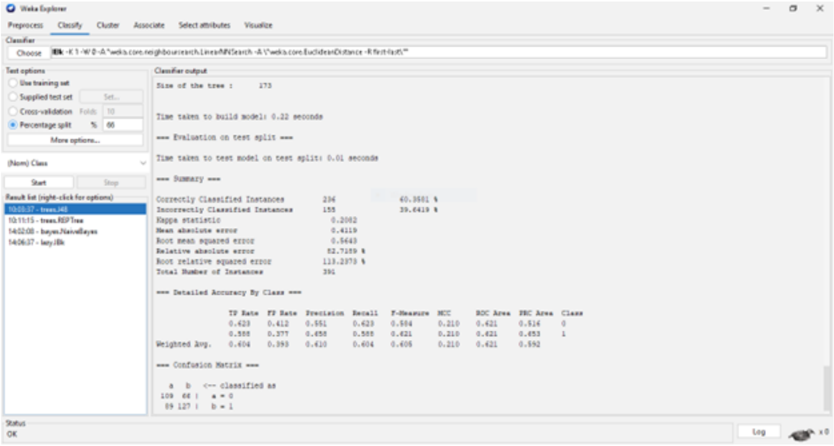
Confusion matrix shows values that are correctly classified for each class and those that are incorrectly classified.

2.4.1 J48 TREE CLASSIFICATION ALGORITHM

The visualized tree for the J48 algorithm on the dataset is given below:



The output for the classifier model is given below:



Classifier: IBk -K 1 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""

Test options:
- ○ Use training set
- ○ Supplied test set
- ○ Cross-validation Folds 10
- ● Percentage split % 66

(Nom) Class

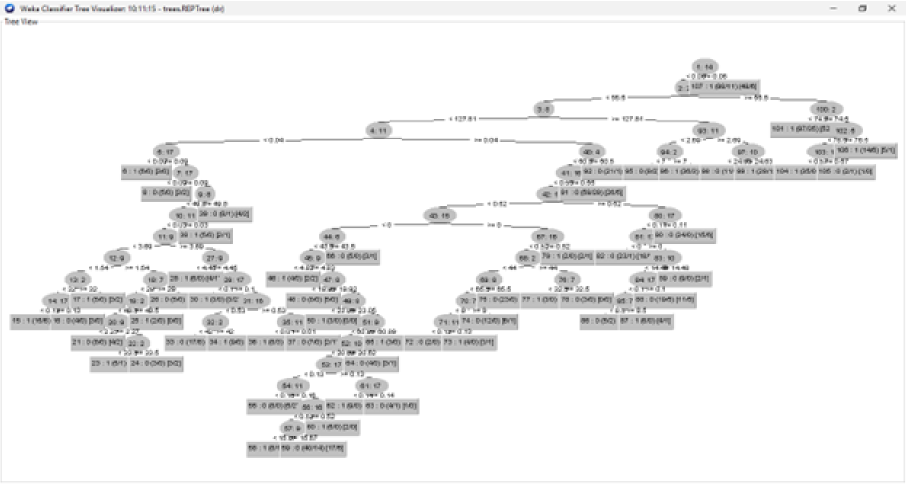
Result list (right-click for options):
- 10:03:37 - trees.J48
- 10:11:15 - trees.REPTree
- 14:02:08 - bayes.NaiveBayes
- 14:06:37 - lazy.IBk

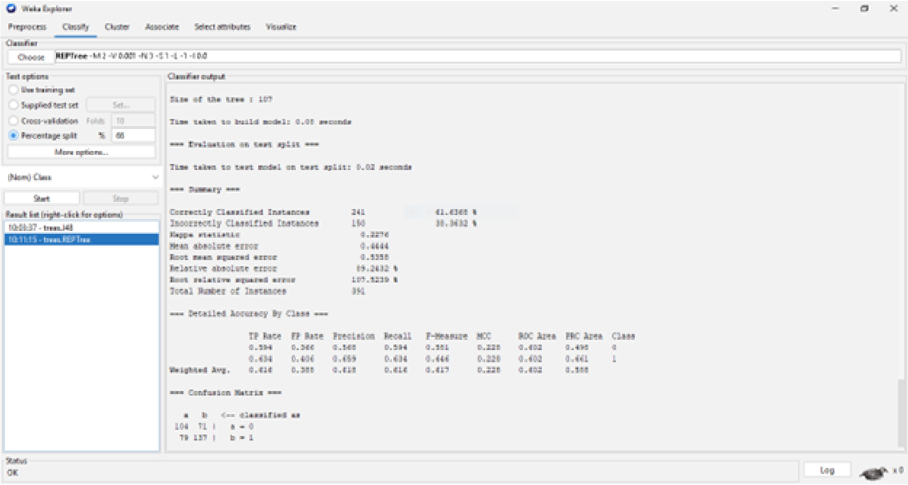
Classifier output:

```
Size of the tree :      173


Time taken to build model: 0.22 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0.01 seconds

=== Summary ===

Correctly Classified Instances         236               60.3581 %
Incorrectly Classified Instances       155               39.6419 %
Kappa statistic                          0.2082
Mean absolute error                      0.4119
Root mean squared error                  0.5643
Relative absolute error                 82.7189 %
Root relative squared error            113.2373 %
Total Number of Instances              391

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.623    0.412    0.551      0.623    0.584      0.210    0.621     0.516     0
                 0.588    0.377    0.658      0.588    0.621      0.210    0.621     0.653     1
Weighted Avg.    0.604    0.393    0.610      0.604    0.605      0.210    0.621     0.592

=== Confusion Matrix ===

   a   b   <-- classified as
 109  66 |   a = 0
  89 127 |   b = 1
```

Status: OK

2.4.2 REDUCED ERROR PRUNING TREE CLASSIFICATION ALGORITHM


Weka Classifier Tree Visualizer: 10:11:15 - trees.REPTree (dr)
Tree View



Classifier: REPTree -M 2 -V 0.001 -N 3 -S 1 -L -1 -I 0.0

Test options
○ Use training set
○ Supplied test set
○ Cross-validation Folds 10
● Percentage split % 66

(Nom) Class

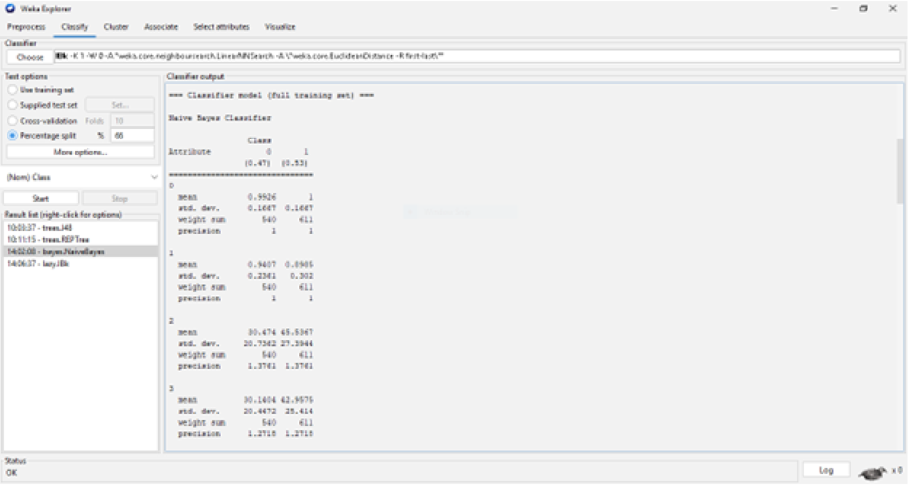
Result list (right-click for options)
10:03:37 - trees.J48
10:11:15 - trees.REPTree

Classifier output

```
Size of the tree : 107

Time taken to build model: 0.05 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0.02 seconds

=== Summary ===

Correctly Classified Instances         241               61.6368 %
Incorrectly Classified Instances       150               38.3632 %
Kappa statistic                          0.2276
Mean absolute error                      0.4644
Root mean squared error                  0.5358
Relative absolute error                 89.2632 %
Root relative squared error            107.5239 %
Total Number of Instances              391

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.594    0.366    0.565      0.594    0.581      0.228    0.602     0.498     0
                 0.634    0.406    0.659      0.634    0.646      0.228    0.602     0.661     1
Weighted Avg.    0.616    0.388    0.618      0.616    0.617      0.228    0.602     0.588

=== Confusion Matrix ===

   a   b   <-- classified as
 104  71 |   a = 0
  79 137 |   b = 1
```

Status
OK

2.4.3 NAÏVE BAYES CLASSIFICATION ALGORITHM



Classifier: IBk -K 1 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""

Test options:
○ Use training set
○ Supplied test set
○ Cross-validation Folds 10
● Percentage split % 66

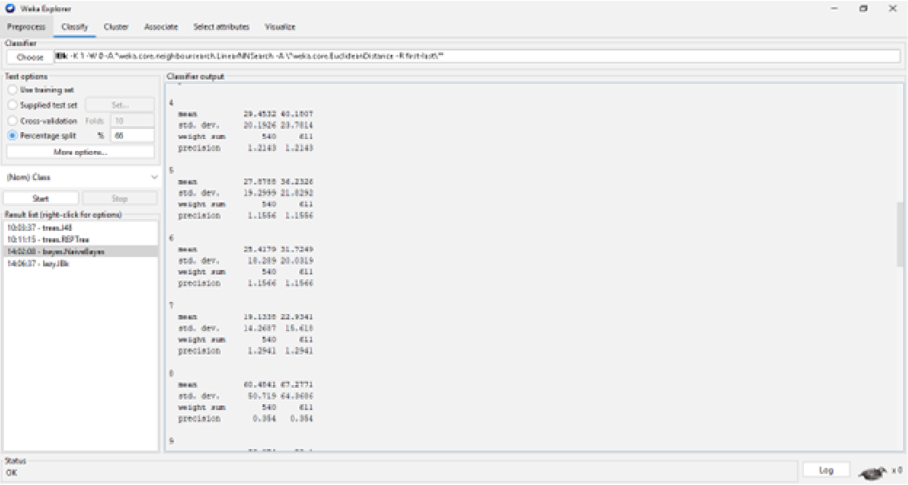
(Nom) Class

Result list (right-click for options)
10:03:37 - trees.J48
10:11:15 - trees.REPTree
14:02:08 - bayes.NaiveBayes
14:06:37 - lazy.IBk

Status: OK

Classifier output

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

Naive Bayes Classifier

                 Class
Attribute            0       1
                (0.47)  (0.53)
================================
0
  mean          0.9926       1
  std. dev.     0.1667  0.1667
  weight sum       540     611
  precision          1       1

1
  mean          0.9407  0.8985
  std. dev.     0.2361   0.302
  weight sum       540     611
  precision          1       1

2
  mean          30.474 65.5367
  std. dev.    20.7362 27.3944
  weight sum       540     611
  precision     1.3761  1.3761

3
  mean         30.1404 42.9575
  std. dev.    20.4472  25.414
  weight sum       540     611
  precision     1.2718  1.2718
```



Classifier: IBk -K 1 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""

Test options:
○ Use training set
○ Supplied test set
○ Cross-validation Folds 10
● Percentage split % 66

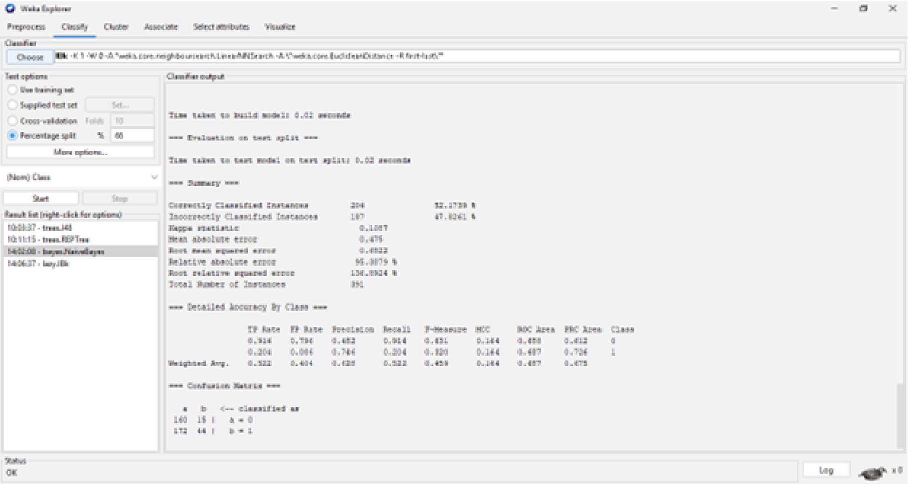
(Nom) Class

Result list (right-click for options)
10:03:37 - trees.J48
10:11:15 - trees.REPTree
14:02:08 - bayes.NaiveBayes
14:06:37 - lazy.IBk

Status: OK

Classifier output

```
4
  mean         29.4532 60.1807
  std. dev.    20.1926 23.7814
  weight sum       540     611
  precision     1.2143  1.2143

5
  mean         27.8788 36.2326
  std. dev.    19.2999 21.8292
  weight sum       540     611
  precision     1.1566  1.1566

6
  mean         25.4179 31.7249
  std. dev.     18.289 20.0319
  weight sum       540     611
  precision     1.1566  1.1566

7
  mean         19.1338 22.9341
  std. dev.    14.2687  15.618
  weight sum       540     611
  precision     1.2941  1.2941

8
  mean         60.4841 87.2771
  std. dev.     50.719 64.0686
  weight sum       540     611
  precision      0.384   0.384

9
```



Classifier: IBk -K 1 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""

Test options
○ Use training set
○ Supplied test set
○ Cross-validation Folds 10
● Percentage split % 66

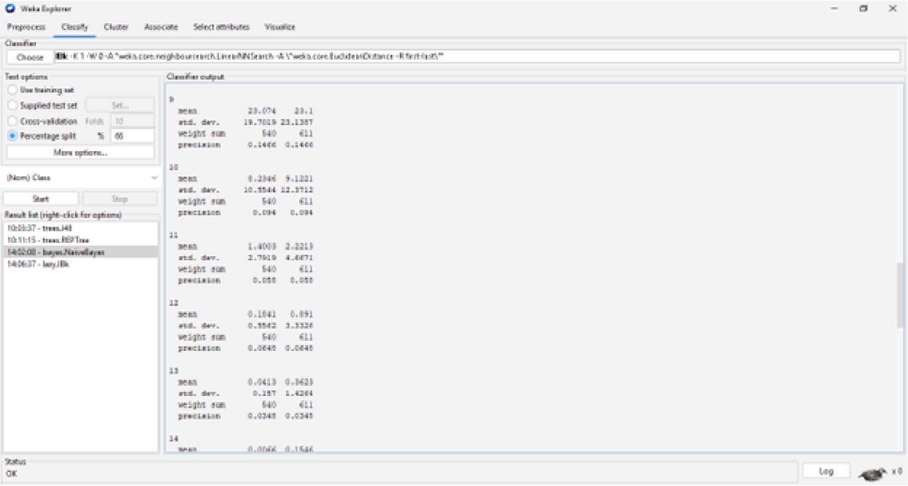
(Nom) Class

Result list (right-click for options)
10:03:37 - trees.J48
10:11:15 - trees.REPTree
14:02:08 - bayes.NaiveBayes
14:06:37 - lazy.IBk

Classifier output

```
Time taken to build model: 0.02 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0.02 seconds

=== Summary ===

Correctly Classified Instances         204               52.1739 %
Incorrectly Classified Instances       187               47.8261 %
Kappa statistic                          0.1067
Mean absolute error                      0.475
Root mean squared error                  0.6822
Relative absolute error                 95.3879 %
Root relative squared error            136.8924 %
Total Number of Instances              391

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.914    0.796    0.482      0.914    0.631      0.164    0.688     0.612     0
                 0.204    0.086    0.746      0.204    0.320      0.164    0.687     0.726     1
Weighted Avg.    0.522    0.404    0.628      0.522    0.459      0.164    0.687     0.675

=== Confusion Matrix ===

   a   b   <-- classified as
 160  15 |   a = 0
 172  44 |   b = 1
```

Status
OK



Classifier: IBk -K 1 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""

Test options
○ Use training set
○ Supplied test set
○ Cross-validation Folds 10
● Percentage split % 66

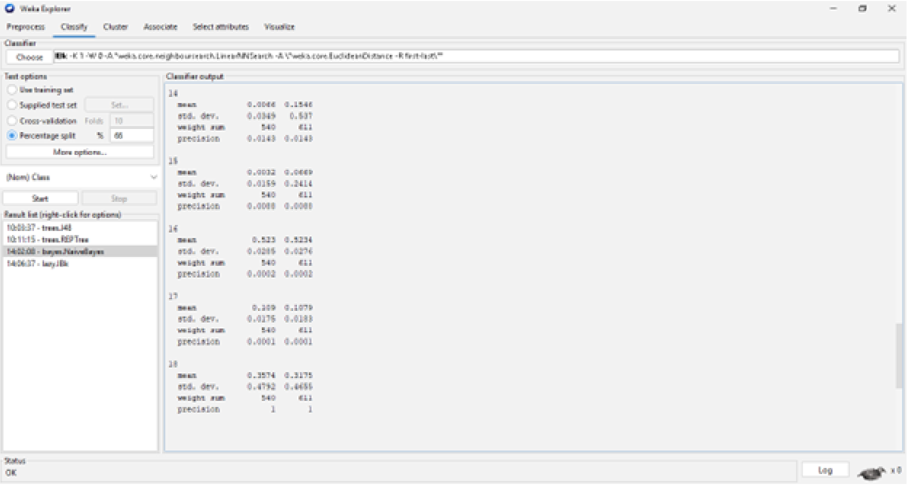
(Nom) Class

Result list (right-click for options)
10:03:37 - trees.J48
10:11:15 - trees.REPTree
14:02:08 - bayes.NaiveBayes
14:06:37 - lazy.IBk

Classifier output

```
9
  mean             20.074    20.1
  std. dev.       19.7019 23.1387
  weight sum          540     611
  precision        0.1466  0.1466

10
  mean             8.2346  9.1221
  std. dev.       10.5544 12.3712
  weight sum          540     611
  precision         0.094   0.094

11
  mean             1.4003  2.2213
  std. dev.        2.7919  4.4671
  weight sum          540     611
  precision         0.058   0.058

12
  mean             0.1841   0.891
  std. dev.        0.5542  3.3326
  weight sum          540     611
  precision        0.0648  0.0648

13
  mean             0.0413  0.3623
  std. dev.         0.157  1.4264
  weight sum          540     611
  precision        0.0348  0.0348

14
  mean             0.0066  0.1546
```

Status
OK



IBk -K 1 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""

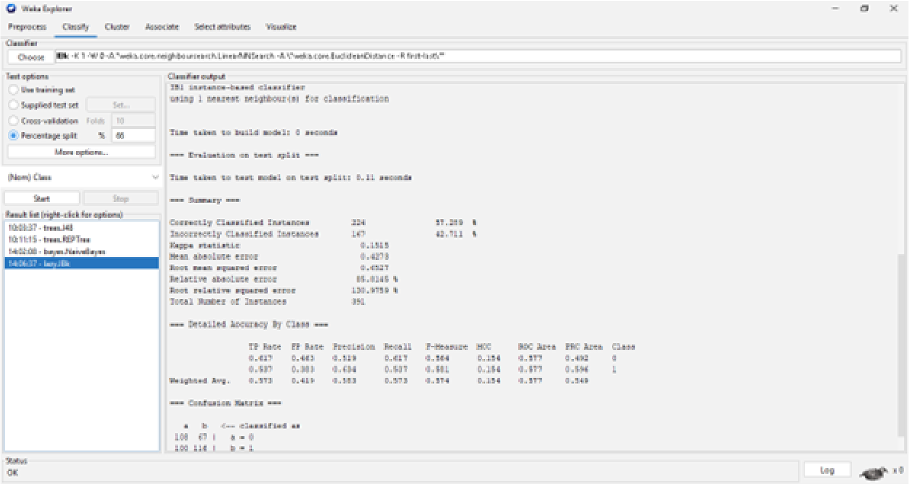
○ Use training set
○ Supplied test set Set...
○ Cross-validation Folds 10
● Percentage split % 66

(Nom) Class

```
14
  mean            0.0066  0.1546
  std. dev.       0.0349   0.537
  weight sum         540     611
  precision       0.0143  0.0143

15
  mean            0.0032  0.0669
  std. dev.       0.0159  0.2414
  weight sum         540     611
  precision       0.0088  0.0088

16
  mean             0.523  0.5234
  std. dev.       0.0285  0.0276
  weight sum         540     611
  precision       0.0002  0.0002

17
  mean             0.109  0.1079
  std. dev.       0.0176  0.0183
  weight sum         540     611
  precision       0.0001  0.0001

18
  mean            0.3574  0.3175
  std. dev.       0.4792  0.4655
  weight sum         540     611
  precision            1       1
```

2.4.4 KNN CLASSIFICATION ALGORITHM

The KNN algorithm has no model. When the algorithm is applied to the dataset with a KNN value of 1, the results are shown in figure below.



IBk -K 1 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""

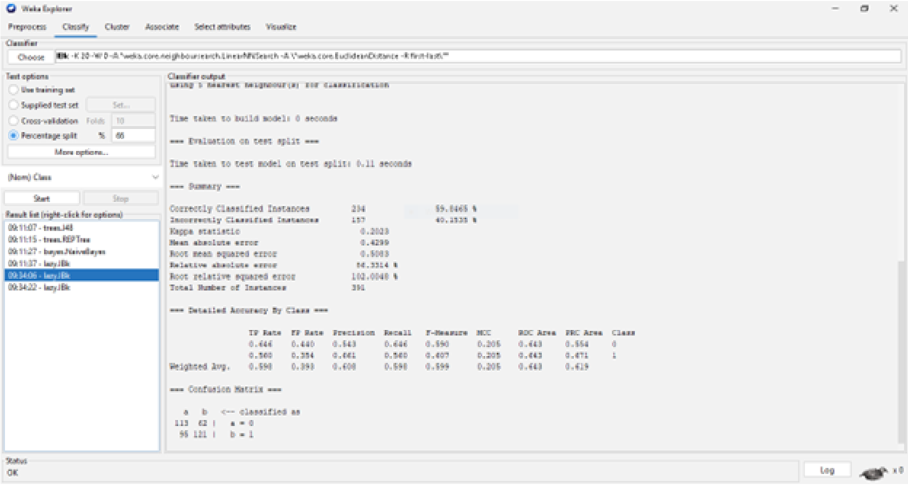
○ Use training set
○ Supplied test set Set...
○ Cross-validation Folds 10
● Percentage split % 66

(Nom) Class

```
IB1 instance-based classifier
using 1 nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0.11 seconds

=== Summary ===

Correctly Classified Instances         224               57.289 %
Incorrectly Classified Instances       167               42.711 %
Kappa statistic                          0.1515
Mean absolute error                      0.4273
Root mean squared error                  0.6527
Relative absolute error                 85.8145 %
Root relative squared error            130.9759 %
Total Number of Instances              391

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.617    0.463    0.519      0.617    0.564      0.154    0.577     0.492     0
                 0.537    0.383    0.634      0.537    0.581      0.154    0.577     0.596     1
Weighted Avg.    0.573    0.419    0.583      0.573    0.574      0.154    0.577     0.549

=== Confusion Matrix ===

   a   b   <-- classified as
 108  67 |   a = 0
 100 116 |   b = 1
```

The number of correctly classified Instances were 224 at a 57.289 percent accuracy and the number of incorrectly classified instances were 167 at a 42.711percent accuracy. It has a kappa statistic of 0.1515. 0.1515 is a poor kappa statistic which is close to random guessing. The algorithm has a weighted true positive rate of 0.573 and a weighted false positive rate of 0.419. It has a weighted precision rate of 0.583 and a recall rate of 0.573. It has an F-measure of 0.573 and an ROC area of 0.577.

For the confusion matrix, 108 instances of class a were correctly classified and 67 instances of class a were incorrectly classified as class b = 1. 100 instances of class b were incorrectly classified as class a and 116 instances of class b were correctly classified. From the above statistics is can be determined that the model is not good and is close to random guessing. The KNN value was then changed to 5 which gave the following results:

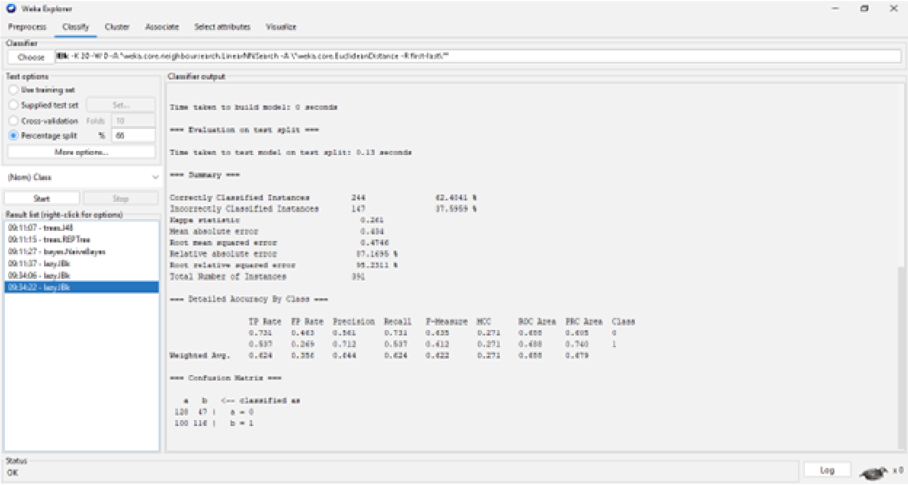


IBk -K 20 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""

Percentage split % 66

```
using 5 nearest neighbour(s) for classification


Time taken to build model: 0 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0.11 seconds

=== Summary ===

Correctly Classified Instances         234               59.8465 %
Incorrectly Classified Instances       157               40.1535 %
Kappa statistic                          0.2023
Mean absolute error                      0.4299
Root mean squared error                  0.5083
Relative absolute error                 86.3314 %
Root relative squared error            102.0048 %
Total Number of Instances              391

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.646    0.440    0.543      0.646    0.590      0.205    0.643     0.554     0
                 0.560    0.354    0.661      0.560    0.607      0.205    0.643     0.671     1
Weighted Avg.    0.598    0.393    0.608      0.598    0.599      0.205    0.643     0.619

=== Confusion Matrix ===

   a   b   <-- classified as
 113  62 |   a = 0
  95 121 |   b = 1
```

The percentage of correctly classified instances improves to 59.8465 percent and the precision, recall, f-measure and ROC area rates improve as well but the model still is not good enough. The KNN value was varied to 20 and the results are shown below:



IBk -K 20 -W 0 -A "weka.core.neighboursearch.LinearNNSearch -A \"weka.core.EuclideanDistance -R first-last\""

Percentage split % 66

```
Time taken to build model: 0 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0.13 seconds

=== Summary ===

Correctly Classified Instances         244               62.4041 %
Incorrectly Classified Instances       147               37.5959 %
Kappa statistic                          0.261
Mean absolute error                      0.434
Root mean squared error                  0.4746
Relative absolute error                 87.1695 %
Root relative squared error             95.2311 %
Total Number of Instances              391

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.731    0.463    0.561      0.731    0.635      0.271    0.688     0.605     0
                 0.537    0.269    0.712      0.537    0.612      0.271    0.688     0.740     1
Weighted Avg.    0.624    0.356    0.644      0.624    0.622      0.271    0.688     0.679

=== Confusion Matrix ===

   a   b   <-- classified as
 128  47 |   a = 0
 100 116 |   b = 1
```

The percentage of correctly classified instances improves to 62.4041 percent and the precision, recall, f-measure and ROC area rates improve significantly. It can then be observed that for such a noisy dataset the accuracy of the model improves as the KNN value increases.

2.5 REFERENCES

anaN, N. S., & thri, V. G. (2018). Performance and Classification Evaluation of J48 Algorithm and Kendall's Based J48 Algorithm (KNJ48). International Journal of Computer Trends and Technology, 59(2), 73–80. https://doi.org/10.14445/22312803/ijctt-v59p112

Bishop, C. M. (2006). Bishop, C. M. (2006). Pattern Recognition and Machine Learning. (M. Jordan, J. Kleinberg, & B. Schölkopf, Eds.)Pattern Recognition (Vol. 4, p. 738). Springer. doi:10.1117/1.2819119Pattern Recognition and Machine Learning. (M. Jordan, J. Kleinberg, & B. Schölkopf, Eds.), Pattern Recognition (Vol. 4, p. 738). Springer. Retrieved from http://www.library.wisc.edu/selectedtocs/bg0137.pdf

artificial intelligence - Evolutionary computing — Britannica. (n.d.). Retrieved from www.britannica.com: https://www.britannica.com/technology/artificial-intelligence/Evolutionary-computing

Weka - classifiers. (n.d.). Retrieved from www.tutorialspoint.com: https://www.tutorialspoint.com/weka/weka

Weka Decision Tree — Build Decision Tree Using Weka. (n.d.). Retrieved from www.analyticsvidhya.com: https://www.analyticsvidhya.com/blog/2020/03/decision-tree-weka-no-coding/