Faculty of Engineering and Technology

Department of Electrical and Computer Engineering

Artificial Intelligence ENCS 3340

Second Semester, 2021/2022

PROJECT#2 Machine Learning for Classification

****

Prepared by: Osama Qutait

Student's No.: 1191072

Partner's name: Mahmoud Samara

Partner's *No.: 1191602*

Instructor: Dr.Adnan Yahya

Date: 12-6-2022

.....

Table of Contents

List of Figures

1. Abstract

First of all, this will be an implementation of Machine Learning for Classification. We will compare different machine learning algorithms for a classification task and test 3 different models. We will use WEKA3.8.6 program to processing, classifying and simulate project results and get the values for the project from (Raisin Dataset.affr) file that have taken from the link. Since the team numbers are 1191072, 1191602, So 2 mode 3 = 2 then we will take group number 2 (Raisin Dataset). We will make the following models:

1. Decision Tree
2. Naïve Bayes
3. Multilayer perceptron

2. Data Set

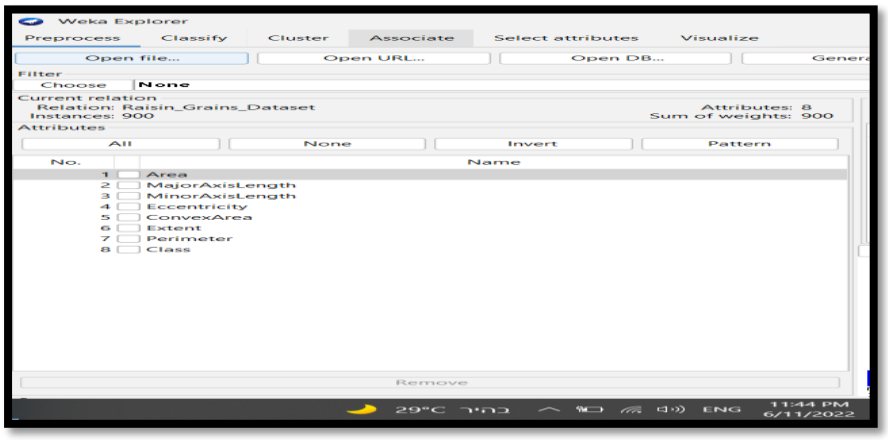

Figure 2-1 Raisin Dataset

As we can see from the previous figure first we have open Raisin Dataset.ARFF file in WEKA program to read it. Now the number of instances and attributes has been initialized and were recorded in the screen shot, so number of instances is 900 and the number of attributes is 8 that are: 1.Area, 2.MajorAxisLength, 3.MinorAxisLength, 4.Eccentricity, 5.ConvexArea, 6.Extent 7.Perimeter (these 7 attributes have Numeric type (continuous) No.8.Class has Nominal type (discrete).

See the following figure that shows the classification for the different data for each attribute:

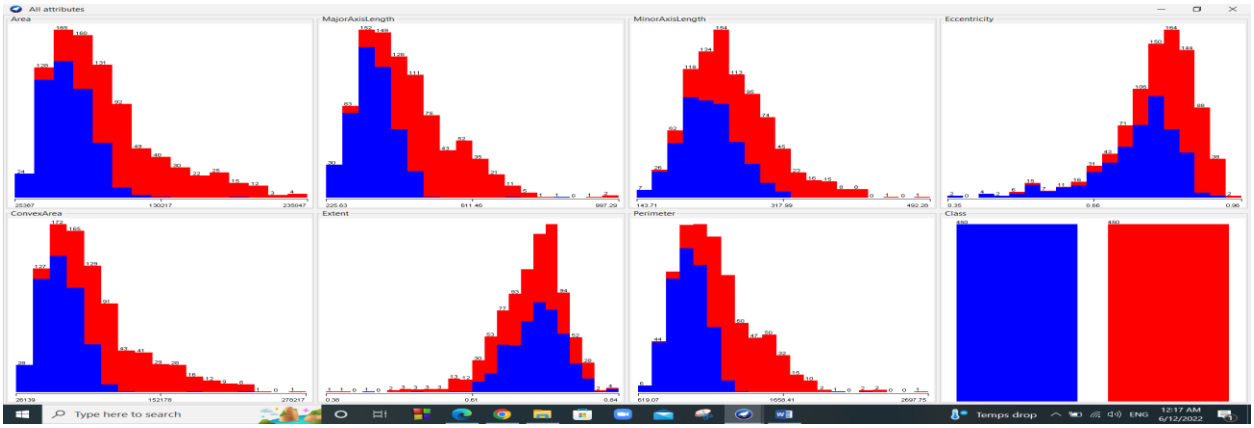

Figure 2-2 distrubtion for all attributes

Also the preprocess page can show the details for each attribute for example: Attribute Area since it is continuous so it will show the different statics like (minimum value, maximum value, Mean and Standard deviation). On the other hand, attribute class it is discrete so it will show the kinds of labels it contains and the weight for each label from the total number.

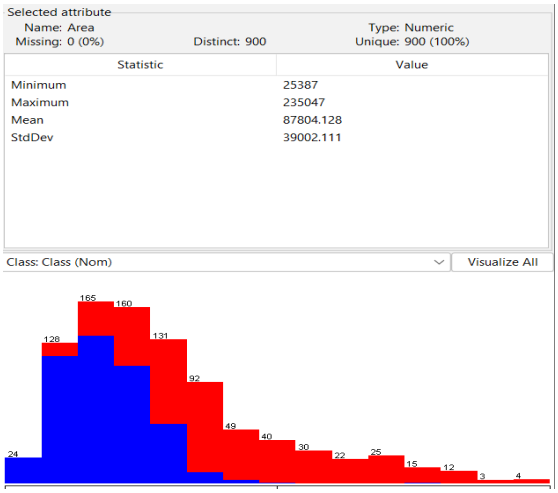

Figure 2-3 Attribute Class information

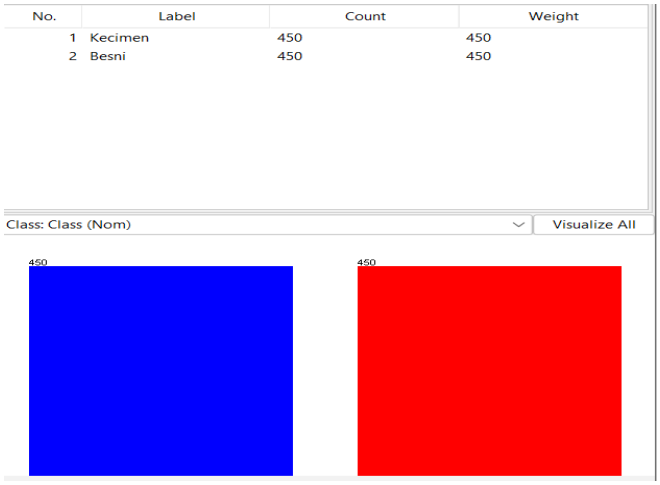

Figure 2-4 Attribute area information

3. First model (Decision tree)

Before starting our work in the decision tree model I will make discretization of two continuous attributes that are (Area and Extent) as the following: From Filter we will choose Discretize then I will choose which attribute I will change and the new number of bins. The following figures shows new distribution for Area and Extent after making discretization so the changed from continuous to discrete(Nominal type).

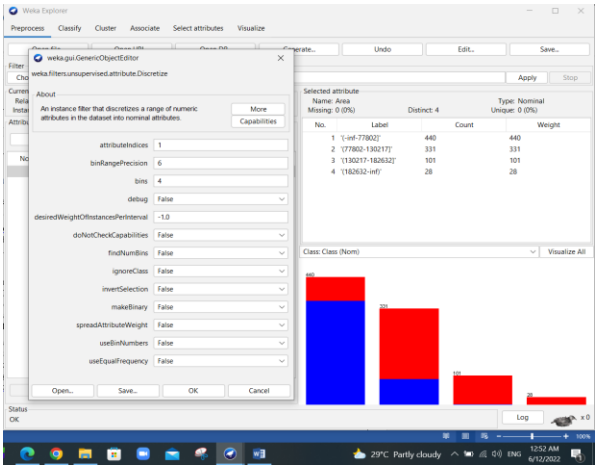

Figure 3-1 new distribution for Area

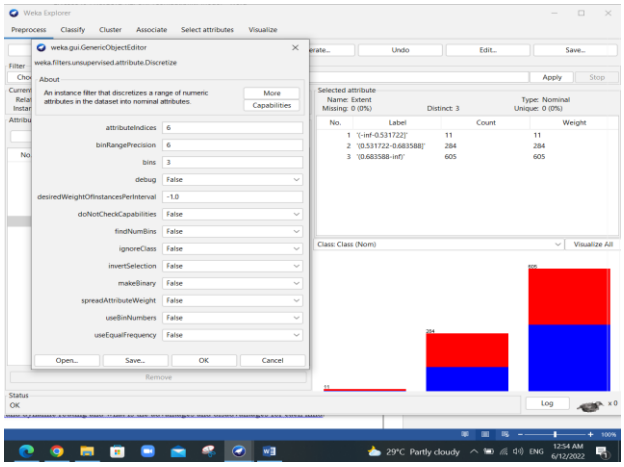

Figure 3-2 new distribution for Extent

We applied the decision tree as the first algorithm for classifying the data set. We chose 5-fold cross validation as a test option:

From the figure 3-3 we see that the total number of leaves was 7 and the size of the tree is 12. In addition, we see that from the test we got 760 instances are correctly classified with a percentage of accuracy of 84.4444% and with 140 incorrectly classified instances with a percentage of accuracy of 15.5556%. Moreover, we see that the Confusion matrix results are initialized as the Class option results, for example Kecimen (a) is true so 390 instances out of 450 is true with rate of 0.876 and this rate is the true positive rate for Keciman and the False Positive is 80 with a rate of 0.178. The precision, recall and F-measure with their weighted averages were shown in the figure. The same thing for Besni with different numbers and rates.

```
Number of Leaves  :     7

Size of the tree :      12


Time taken to build model: 0.01 seconds

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

Correctly Classified Instances         760               84.4444 %
Incorrectly Classified Instances       140               15.5556 %
Kappa statistic                          0.6889
Mean absolute error                      0.2242
Root mean squared error                  0.36
Relative absolute error                 44.849  %
Root relative squared error             72.0037 %
Total Number of Instances              900

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.867    0.178    0.830      0.867    0.848      0.690    0.873     0.827     Kecimen
                 0.822    0.133    0.860      0.822    0.841      0.690    0.873     0.855     Besni
Weighted Avg.    0.844    0.156    0.845      0.844    0.844      0.690    0.873     0.841

=== Confusion Matrix ===

   a   b   <-- classified as
 390  60 |   a = Kecimen
  80 370 |   b = Besni
```

Log x 0

Figure 3-3 Summary of Decision Tree

For The precision we can calculate it as: **Precision = TruePositives / (TruePositives + FalsePositives),** for example for Kecimen Precision it is equal: (0.867 / (0.867+0.178)) = 0.830

For The Recall we can calculate it as: **Recall = TruePositives / (TruePositives + FalseNegatives)** for example for Kecimen Recall it is equal: (0.867 / (0.867+0.133)) = 0.867

For The F-Measure we can calculate it as: **F-Measure = (2 * Precision * Recall) / (Precision + Recall)** for example for Kecimen F-Measure it is equal: (2*0.830*0.867 / (0.867+0.830)) = 0.848

The report shows also all this measures and calculations and confusion matrix for Besni with different numbers and rates.

See figure 3-4 that shows the decision tree

for our test and simulation with the number of leaves was 7 and the size of the tree is 12

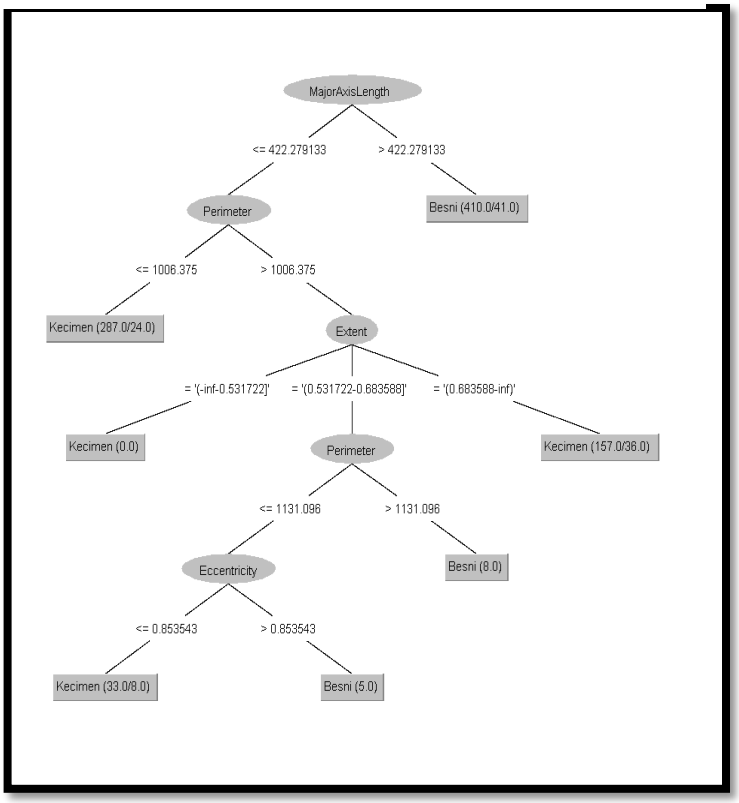

Figure 3-4 The Decision Tree

Now we will change the hyper-parameter for example change binary split to be true and the confidence factor from 0.25 to 0.1. So, the number of leaves, size of tree, precision, recall, F-measure with their weighted averages and percentage of correct/incorrect classification were also affected but the calculation rules will remain the same as shown in the figure below:

```
Number of Leaves  :      6

Size of the tree :       11


Time taken to build model: 0.01 seconds

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

Correctly Classified Instances         761               84.5556 %
Incorrectly Classified Instances       139               15.4444 %
Kappa statistic                          0.6911
Mean absolute error                      0.2389
Root mean squared error                  0.3593
Relative absolute error                 47.7737 %
Root relative squared error             71.8567 %
Total Number of Instances              900

=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
                 0.887     0.196     0.819       0.887   0.852      0.693    0.862     0.818     Kecimen
                 0.804     0.113     0.877       0.804   0.839      0.693    0.862     0.850     Besni
Weighted Avg.    0.846     0.154     0.848       0.846   0.845      0.693    0.862     0.834

=== Confusion Matrix ===

   a   b   <-- classified as
 399  51 |   a = Kecimen
  88 362 |   b = Besni
```

Log x 0

Figure 3-5 Summary of the newDecision Tree

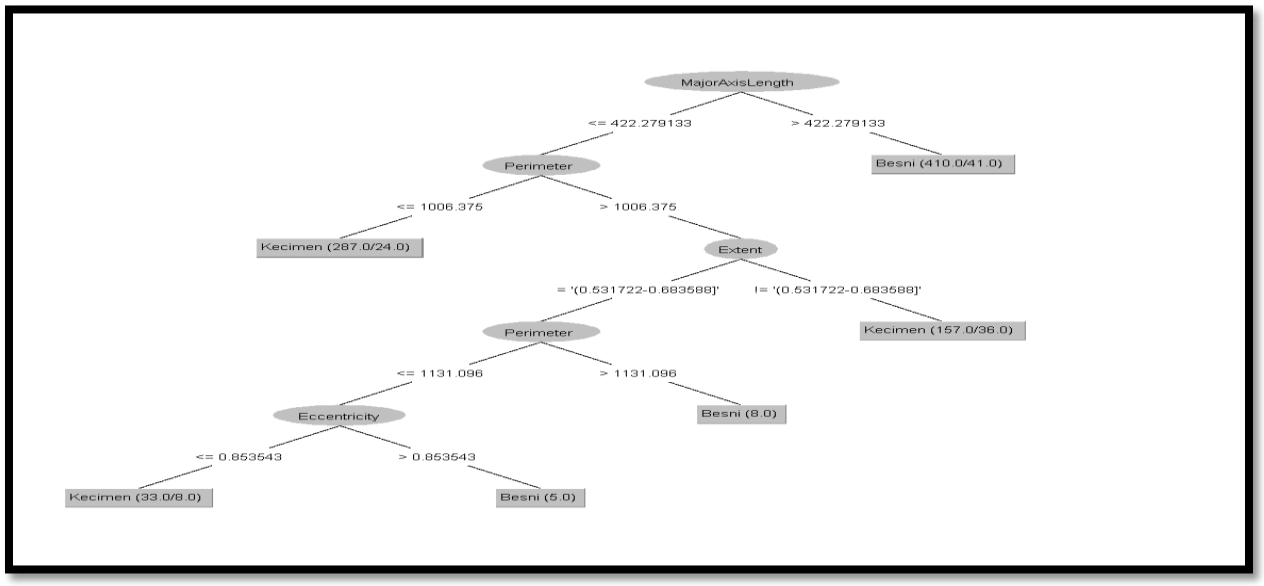

Figure 3-6 New Decision Tree

4. Second model (Naïve Bayes)

I will make discretization to Perimeter attribute as what I did in the first model. The following figure shows new distribution after making discretization so it changed to discrete(Nominal type).

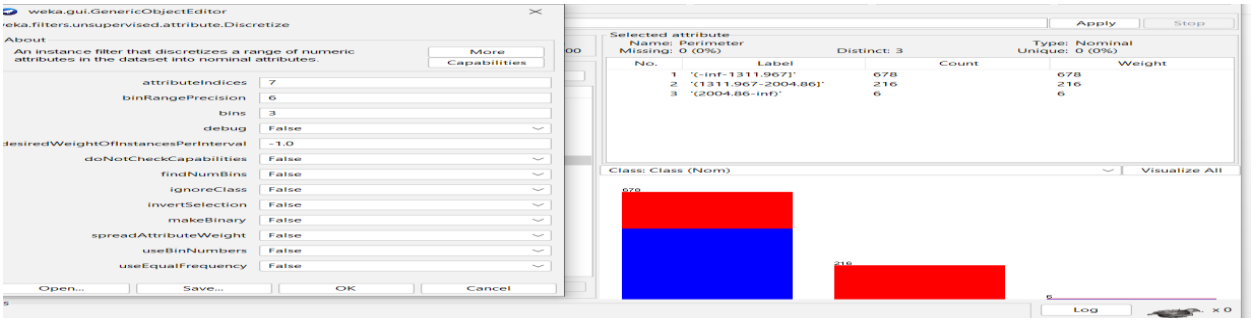

Figure 4-1 new distribution for Perimeter

We applied the Naïve Bayes as the second algorithm for classifying the data set. We chose 5-fold cross validation as a test option:

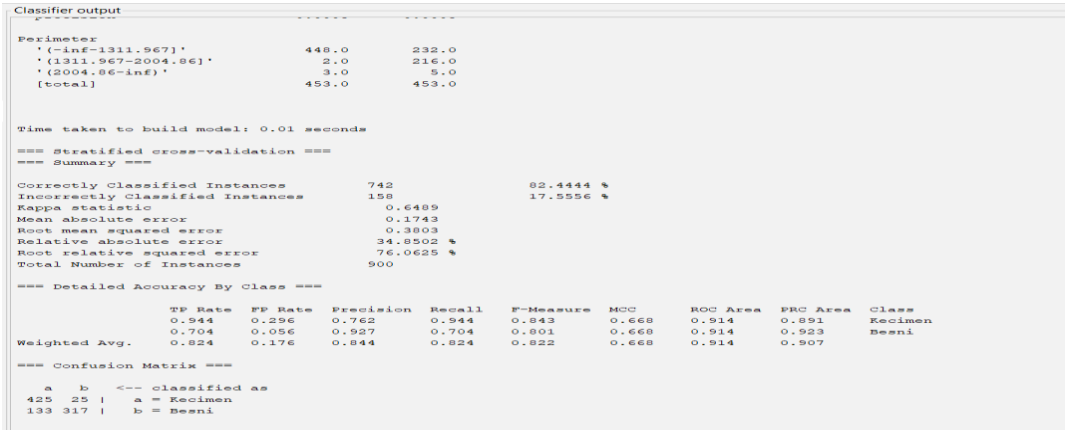

Figure 4-2 Classifier output

From the figure 4-2 we see It produces 5 equal sized sets. In addition, we see that from the test we got 742 instances are correctly classified with a percentage of accuracy of 82.4444% and with 158 incorrectly classified instances with a percentage of accuracy of 17.5556%. Moreover, we see that the Confusion matrix results are initialized as the Class option results, for example Kecimen (a) is true so 425 instances out of 450 is true with rate of 0.944 and this rate is the true positive rate for Keciman and the False Positive is 25 with a rate of 0.296. The precision, recall and F-measure with their weighted averages were shown in the figure. The same thing for Besni with different numbers and rates.

For The precision we can calculate it as: **Precision = TruePositives / (TruePositives + FalsePositives),** for example for Kecimen Precision it is equal: (0.944 / (0.944+0.296)) = 0.762

For The Recall we can calculate it as: **Recall = TruePositives / (TruePositives + FalseNegatives)** for example for Kecimen Recall it is equal: (0.867 / (0.867+0.133)) = 0.944

For The F-Measure we can calculate it as: **F-Measure = (2 * Precision * Recall) / (Precision + Recall)** for example for Kecimen F-Measure it is equal: (2*0.944*0.762 / (0.944+0.762)) = 0.843

The report shows also all this measures and calculations and confusion matrix for Besni with different numbers and rates.

See figure 4-3 that shows the Naïve Bayes curve:

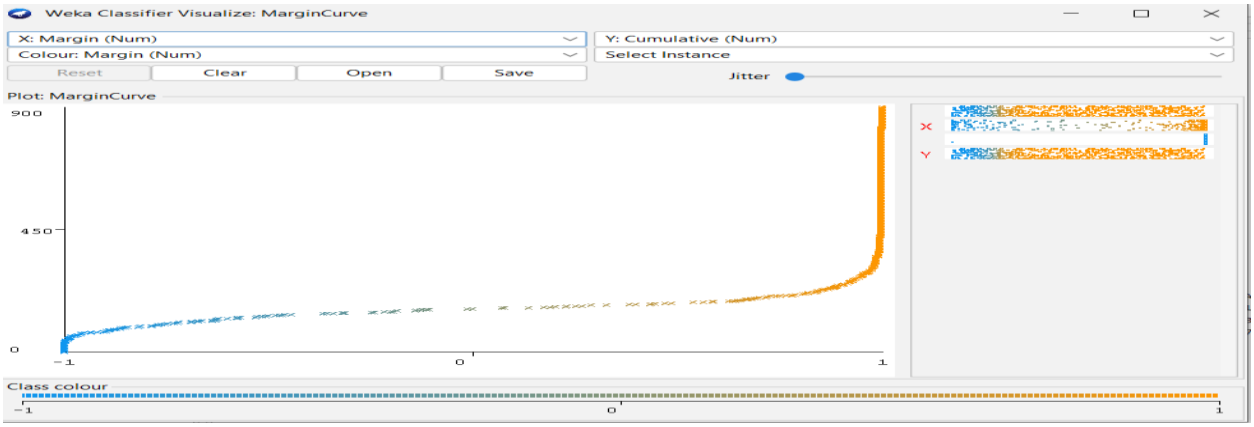

Figure 4-3 Naïve Bayes curve

Now we will change the num of decimal places from 2 to 5, batch size from 100 to 200. So, the precision, recall, F-measure with their weighted averages and percentage of correct/incorrect classification were also affected but the calculation rules will remain the same as shown in the figure below:

Figure 4-4 Classifier output

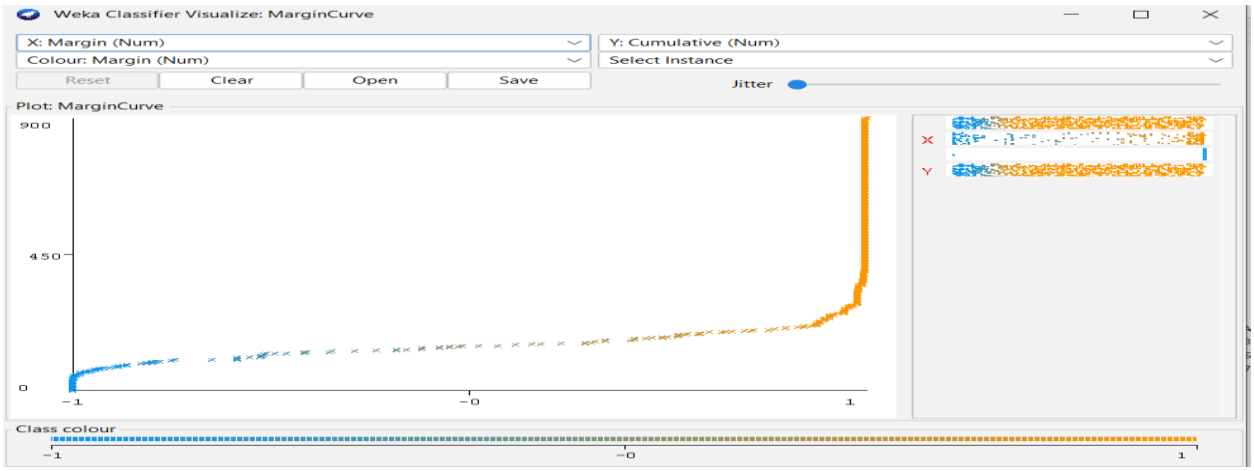

Figure 4-5 New Naïve Bayes curve

5. Third model (multilayer perceptron)

Before starting our work in the multilayer perceptron model I will make discretization of two continuous attributes that are (Area and ConvexArea) as the following: From Filter we will choose Discretize then I will choose which attribute I will change and the new number of bins. The following figures shows new distribution for Area and Extent after making discretization so the changed from continuous to discrete(Nominal type).

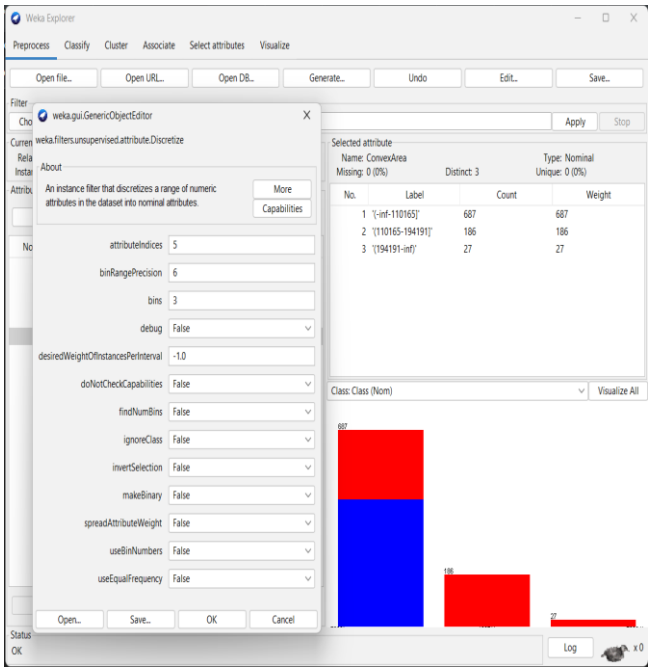

Figure 5-1 new distribution for Area

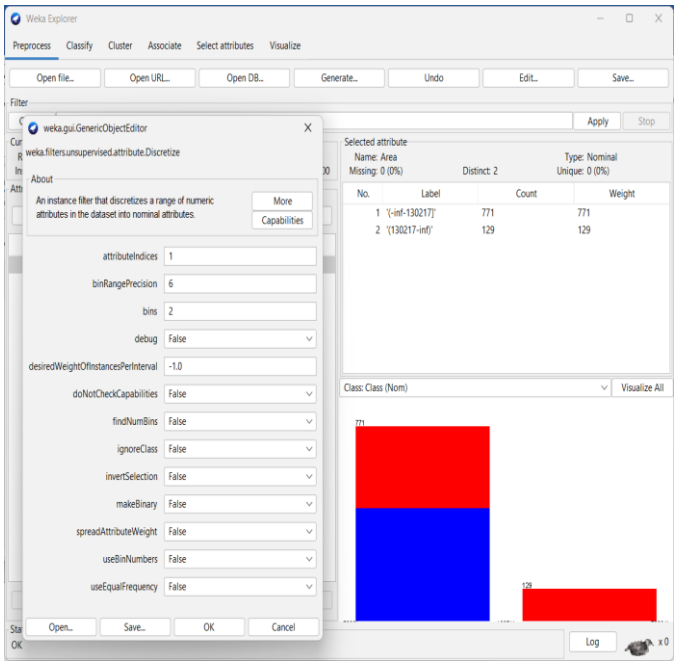

Figure 5-2 new distribution for ConvexArea

```
=== Summary ===

Correctly Classified Instances         775               86.1111 %
Incorrectly Classified Instances       125               13.8889 %
Kappa statistic                          0.7222
Mean absolute error                      0.2055
Root mean squared error                  0.328
Relative absolute error                 41.0954 %
Root relative squared error             65.6034 %
Total Number of Instances              900

=== Detailed Accuracy By Class ===

                 TP Rate   FP Rate   Precision   Recall  F-Measure  MCC      ROC Area  PRC Area  Class
                 0.873     0.151     0.852       0.873   0.863      0.722    0.916     0.887     Kecimen
                 0.849     0.127     0.870       0.849   0.859      0.722    0.916     0.927     Besni
Weighted Avg.    0.861     0.139     0.861       0.861   0.861      0.722    0.916     0.907

=== Confusion Matrix ===

   a   b   <-- classified as
 393  57 |   a = Kecimen
  68 382 |   b = Besni
```

Figure 5-3 Summary of multilayer perceptron

From the figure 5-3 we see that from the test we got 775 instances are correctly classified with a percentage of accuracy of 86.1111% and with 125 incorrectly classified instances with a percentage of accuracy of 13.8889%. Moreover, we see that the Confusion matrix results are initialized as the Class option results, for example Kecimen (a) is true so 393 instances out of 450 is true with rate of 0.873 and this rate is the true positive rate for Keciman and the False Positive is 68 with a rate of 0.151. The precision, recall and F-measure with their weighted averages were shown in the figure. The same thing for Besni with different numbers and rates.

For The precision we can calculate it as:

Precision = TruePositives / (TruePositives + FalsePositives), for example for Kecimen Precision it is equal: (0.873 / (0.873+0.151)) = 0.852

For The Recall we can calculate it as: **Recall = TruePositives /(TruePositives+FalseNegatives)** for example for Kecimen Recall it is equal: (0.873/ (0.873+0.126)) = 0.873

For The F-Measure we can calculate it as: **F-Measure = (2 * Precision * Recall) / (Precision + Recall)** for example for Kecimen F-Measure it is equal:

(2*0.852*0.87 / (0.852+0.87)) = 0.863

The report shows also all this measures and calculations and confusion matrix for Besni with different numbers and rates.

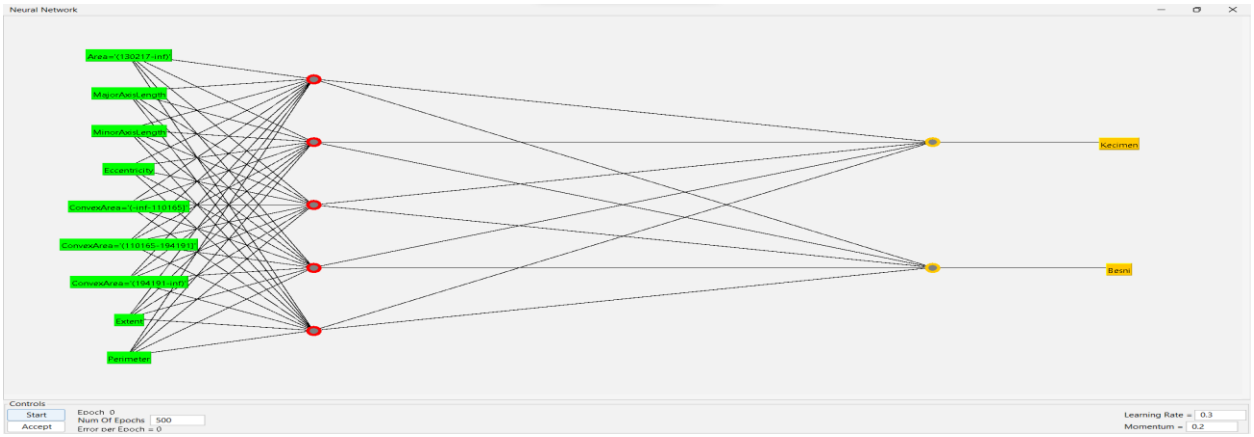

Figure 5-4 multilayer perceptron system

Now we will change the num of hidden layer. So, the precision, recall, F-measure with their weighted averages and percentage of correct/incorrect classification were also affected but the calculation rules will remain the same as shown in the figure below:

```
=== Summary ===

Correctly Classified Instances         774               86      %
Incorrectly Classified Instances       126               14      %
Kappa statistic                          0.72
Mean absolute error                      0.2128
Root mean squared error                  0.3323
Relative absolute error                 42.5588 %
Root relative squared error             66.4573 %
Total Number of Instances              900

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.876    0.156    0.849      0.876    0.862      0.720    0.911     0.883     Kecimen
                 0.844    0.124    0.872      0.844    0.858      0.720    0.911     0.920     Besni
Weighted Avg.    0.860    0.140    0.860      0.860    0.860      0.720    0.911     0.902

=== Confusion Matrix ===

   a   b   <-- classified as
 394  56 |   a = Kecimen
  70 380 |   b = Besni
```

Figure 5-5 Summary of multilayer perceptron

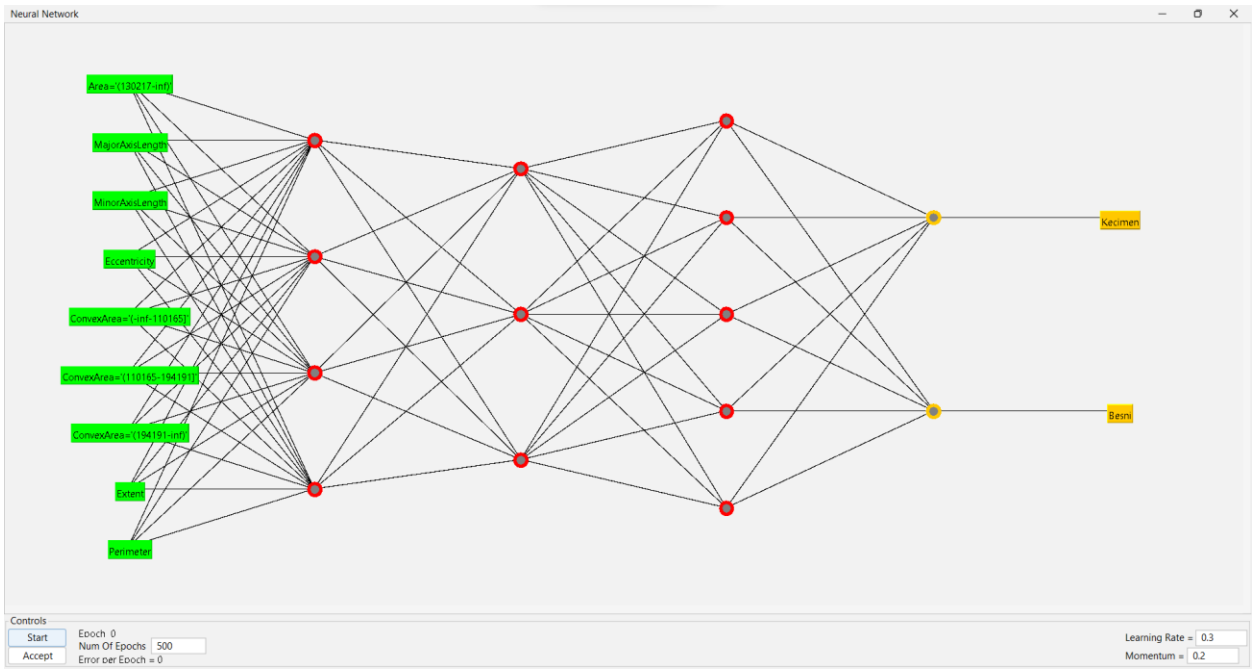

Figure 5-6 multilayer perceptron system

6. Conclusion

In the end, we can say that we learnt how to use a new tool that is WEKA and used different models to test same data and notice the differences between them how do they work and in the results. As we saw in the results and summary, **Decision Tree** is a supervised algorithm It is using a binary tree and the target values are presented in the tree leaves and we knew the size of tree. On the other hand, Naïve Bayes Tree uses decision tree as the general structure and deploys naïve Bayesian classifiers at leaves it classifiers work better than decision trees when the sample data set is small. So since data is big we can see depend on the percentage of the correct instances for each model that decision tree is better than Naïve Bayes in our case. In Multi-Layer Perceptron the connections between neurons are so-called weights. Their values are selected during the training process. Finally, we can notice that the results for the same dataset has changed depend on the type of model we choose, also in the same model the results changed if we change hyper-parameter and we can notice that when 5 cross validation option test it test all the instances and