Title: Finding best Classifier and evaluating performance using test and training set for Glass Identification dataset

Objective:

In this Project, I will Choose five classifier algorithms and apply them into a dataset referred with http://archive.ics.uci.edu/ml/datasets/Glass+Identification . And also, I will find the ROC curve to make a precise decision that which is the best among the selected classifiers. Additionally, we will split our whole dataset into two segments. One is test set and another is training set. I will run this test data set with the training data set and trying to evaluate the performance of the test data. All the task will be done by Weka (a machine learning tool).

Dataset description:

In the dataset there are 214 instances and 11 attributes

Attribute Information:

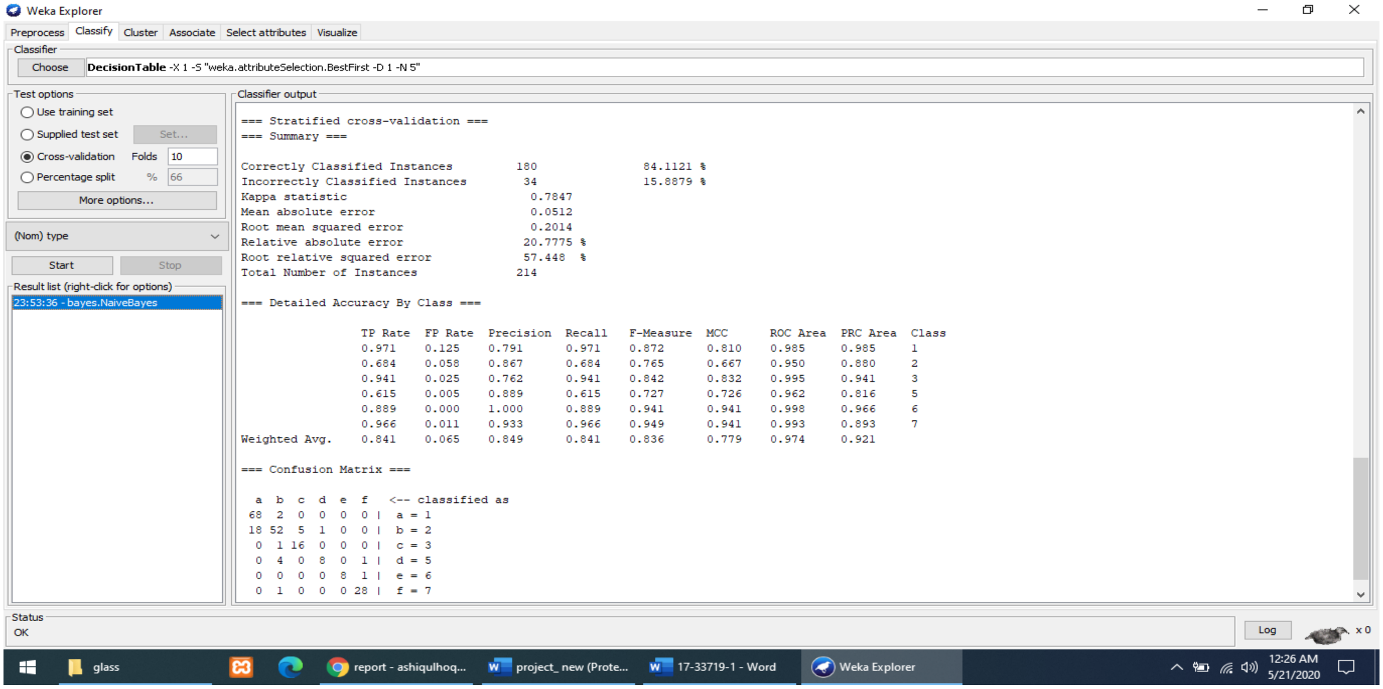
Attributes	values
Id: instances number	1-214
RI: refractive index	Numeric
Na: Sodium	Numeric
Mg: Magnesium	Numeric
Al: Aluminum	Numeric
Si: Silicon	Numeric
K: Potassium	Numeric
Ca: Calcium	Numeric
Ba: Barium	Numeric
Fe: Iron	Numeric
Type of glass: (class attribute)	1: building_windows_float_processed (70 out of 214)
	2: building_windows_non_float_processed (76 out of 214)
	3: vehicle_windows_float_processed (17 out of 214)
	4: vehicle_windows_non_float_processed (0 out of 214)
	5: containers (13 out of 214)
	6: tableware (9 out of 214)
	7: headlamps (29 out of 214)

Note: There is no missing value of attributes

Output: We select five Classification Algorithms and apply them into the mentioned training set. All the results are as follows:-

1. NaiveBayes

NaiveBayes Classifier works on Bayes theorem of probability to predict the class of unknown data sets. Here is output of it.



DecisionTable -X 1 -S "weka.attributeSelection.BestFirst -D 1 -N 5"

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

(Nom) type

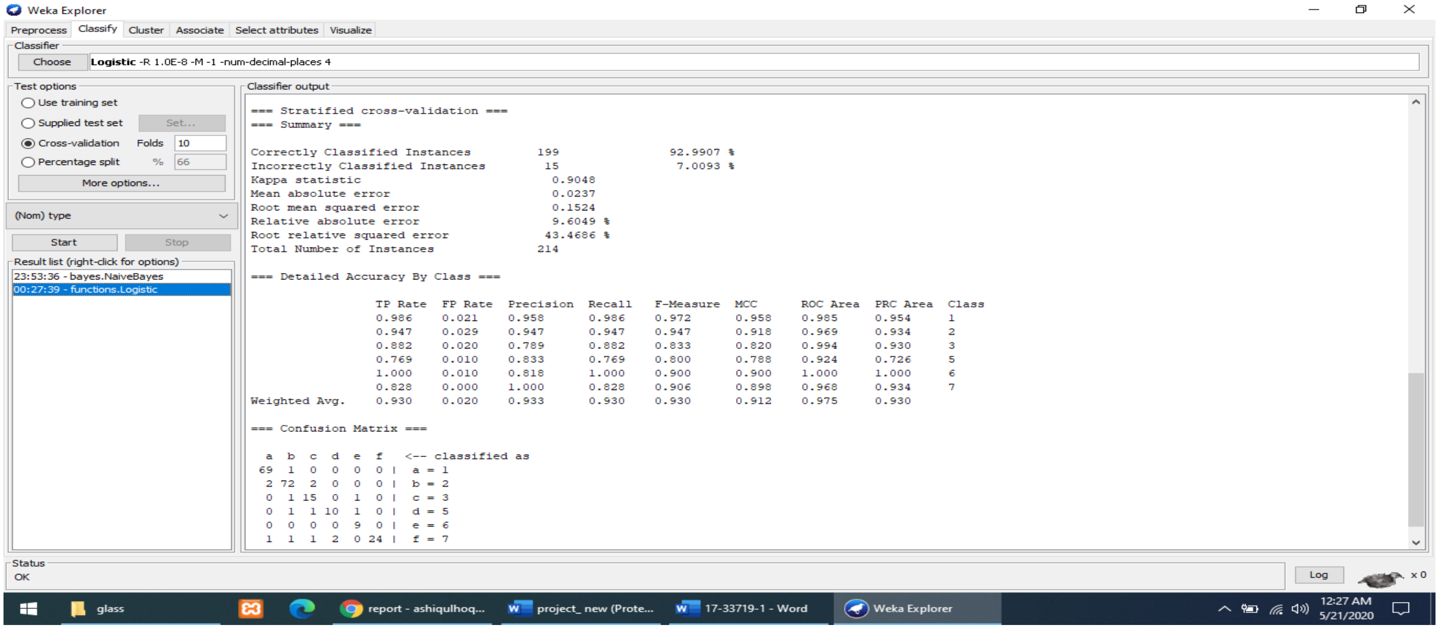
23:53:36 - bayes.NaiveBayes

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

Correctly Classified Instances         180               84.1121 %
Incorrectly Classified Instances        34               15.8879 %
Kappa statistic                          0.7847
Mean absolute error                      0.0512
Root mean squared error                  0.2014
Relative absolute error                 20.7775 %
Root relative squared error             57.448  %
Total Number of Instances              214

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.971    0.125    0.791      0.971    0.872      0.810    0.985     0.985     1
                 0.684    0.058    0.867      0.684    0.765      0.667    0.950     0.880     2
                 0.941    0.025    0.762      0.941    0.842      0.832    0.995     0.941     3
                 0.615    0.005    0.889      0.615    0.727      0.726    0.962     0.816     5
                 0.889    0.000    1.000      0.889    0.941      0.941    0.998     0.966     6
                 0.966    0.011    0.933      0.966    0.949      0.941    0.993     0.893     7
Weighted Avg.    0.841    0.065    0.849      0.841    0.836      0.779    0.974     0.921

=== Confusion Matrix ===

  a  b  c  d  e  f   <-- classified as
 68  2  0  0  0  0 |  a = 1
 18 52  5  1  0  0 |  b = 2
  0  1 16  0  0  0 |  c = 3
  0  4  0  8  0  1 |  d = 5
  0  0  0  0  8  1 |  e = 6
  0  1  0  0  0 28 |  f = 7
```

Status
OK

2.Logistic

Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. Here is output of it.



Logistic -R 1.0E-8 -M -1 -num-decimal-places 4

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

(Nom) type

Result list (right-click for options)
23:53:36 - bayes.NaiveBayes
00:27:39 - functions.Logistic

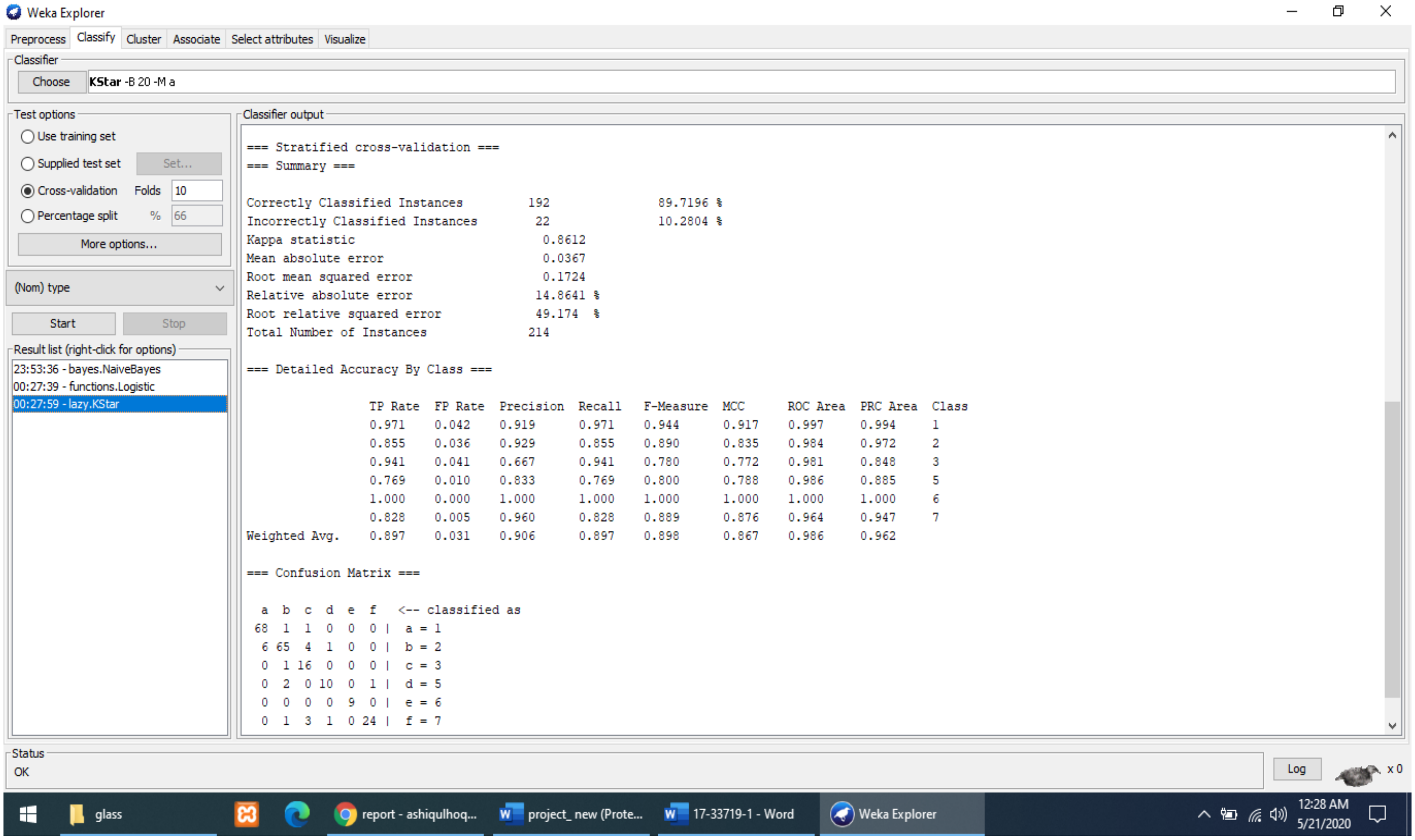
Status
OK

Classifier output

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

Correctly Classified Instances         199               92.9907 %
Incorrectly Classified Instances        15                7.0093 %
Kappa statistic                          0.9048
Mean absolute error                      0.0237
Root mean squared error                  0.1524
Relative absolute error                  9.6049 %
Root relative squared error             43.4686 %
Total Number of Instances              214

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.986    0.021    0.958      0.986    0.972      0.958    0.985     0.954     1
                 0.947    0.029    0.947      0.947    0.947      0.918    0.969     0.934     2
                 0.882    0.020    0.789      0.882    0.833      0.820    0.994     0.930     3
                 0.769    0.010    0.833      0.769    0.800      0.788    0.924     0.726     5
                 1.000    0.010    0.818      1.000    0.900      0.900    1.000     1.000     6
                 0.828    0.000    1.000      0.828    0.906      0.898    0.968     0.934     7
Weighted Avg.    0.930    0.020    0.933      0.930    0.930      0.912    0.975     0.930

=== Confusion Matrix ===

  a  b  c  d  e  f   <-- classified as
 69  1  0  0  0  0 |  a = 1
  2 72  2  0  0  0 |  b = 2
  0  1 15  0  1  0 |  c = 3
  0  1  1 10  1  0 |  d = 5
  0  0  0  0  9  0 |  e = 6
  1  1  1  2  0 24 |  f = 7
```

3.Kstar

K* is an instance-based classifier, that is the class of a test instance is based upon the class of those training instances similar to it, as determined by some similarity function. It differs from other instance-based learners in that it uses an entropy-based distance function. Here is output of it.



KStar -B 20 -M a

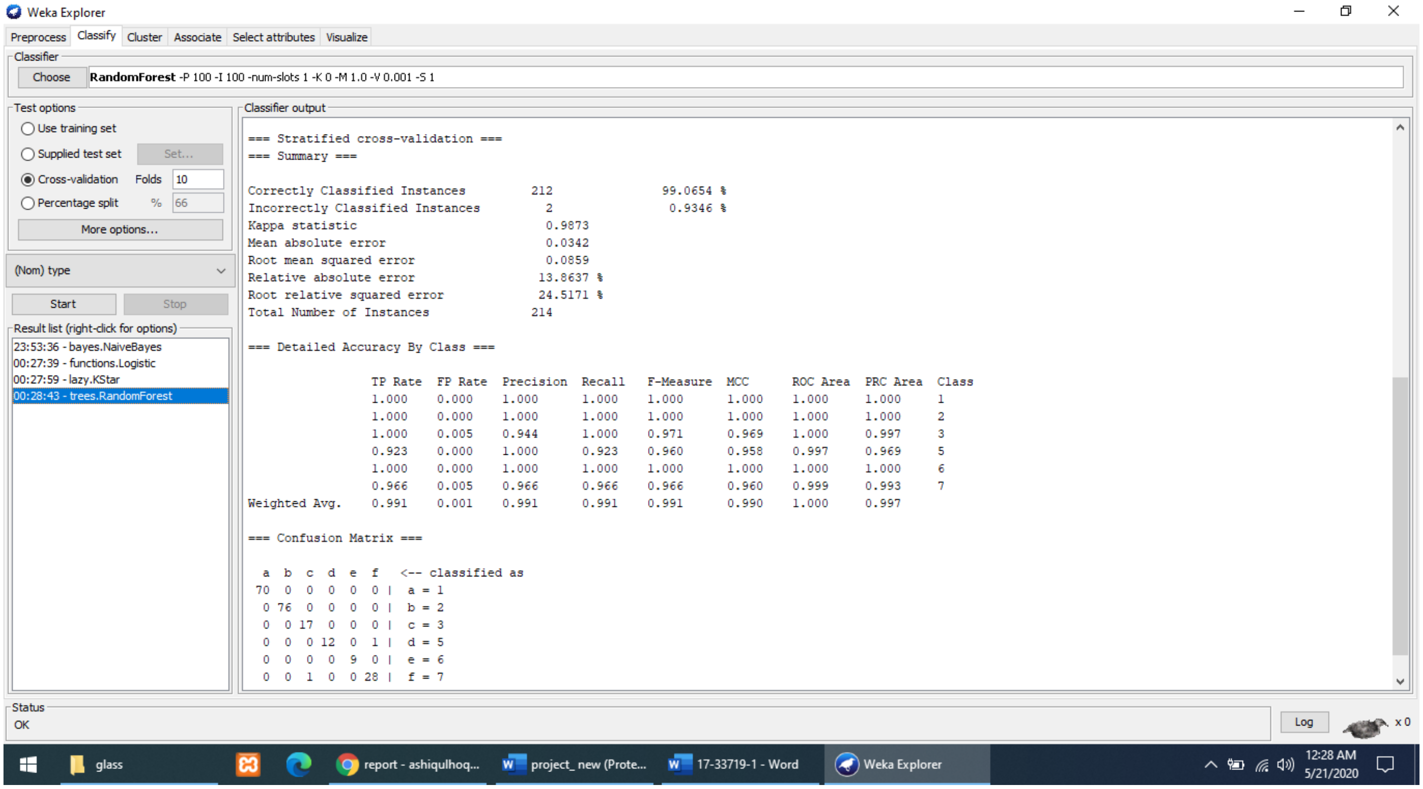
Folds 10

% 66

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

Correctly Classified Instances         192               89.7196 %
Incorrectly Classified Instances        22               10.2804 %
Kappa statistic                          0.8612
Mean absolute error                      0.0367
Root mean squared error                  0.1724
Relative absolute error                 14.8641 %
Root relative squared error             49.174  %
Total Number of Instances              214

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.971    0.042    0.919      0.971    0.944      0.917    0.997     0.994     1
                 0.855    0.036    0.929      0.855    0.890      0.835    0.984     0.972     2
                 0.941    0.041    0.667      0.941    0.780      0.772    0.981     0.848     3
                 0.769    0.010    0.833      0.769    0.800      0.788    0.986     0.885     5
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     6
                 0.828    0.005    0.960      0.828    0.889      0.876    0.964     0.947     7
Weighted Avg.    0.897    0.031    0.906      0.897    0.898      0.867    0.986     0.962

=== Confusion Matrix ===

  a  b  c  d  e  f   <-- classified as
 68  1  1  0  0  0 |  a = 1
  6 65  4  1  0  0 |  b = 2
  0  1 16  0  0  0 |  c = 3
  0  2  0 10  0  1 |  d = 5
  0  0  0  0  9  0 |  e = 6
  0  1  3  1  0 24 |  f = 7
```

4.RandomForest

RandomForest Classifier Works like ‘Decision tree’ but the difference is it split tree randomly rather than traditional split of tree. Here is output of it.



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

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

(Nom) type

Result list (right-click for options)

```
23:53:36 - bayes.NaiveBayes
00:27:39 - functions.Logistic
00:27:59 - lazy.KStar
00:28:43 - trees.RandomForest
```

Classifier output

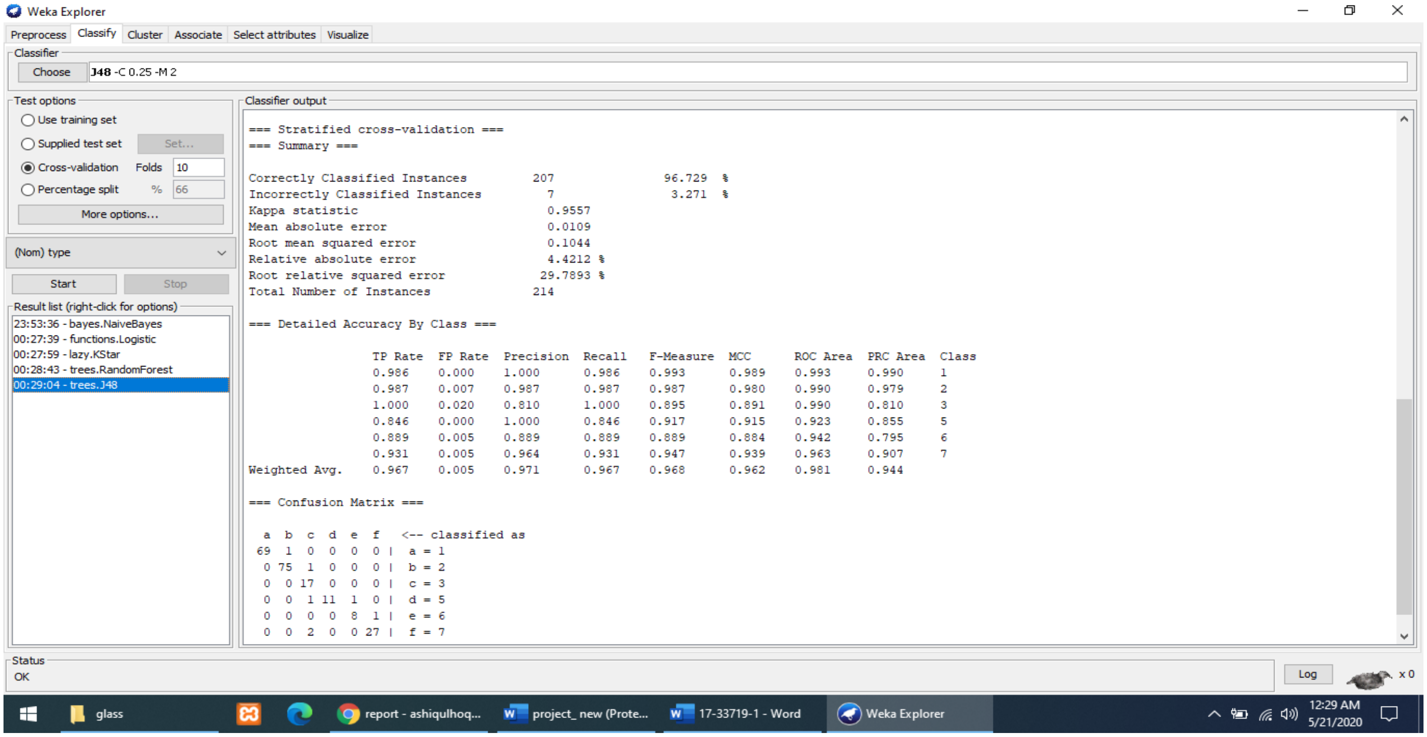
```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         212               99.0654 %
Incorrectly Classified Instances         2                0.9346 %
Kappa statistic                          0.9873
Mean absolute error                      0.0342
Root mean squared error                  0.0859
Relative absolute error                 13.8637 %
Root relative squared error             24.5171 %
Total Number of Instances              214

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     1
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     2
                 1.000    0.005    0.944      1.000    0.971      0.969    1.000     0.997     3
                 0.923    0.000    1.000      0.923    0.960      0.958    0.997     0.969     5
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     6
                 0.966    0.005    0.966      0.966    0.966      0.960    0.999     0.993     7
Weighted Avg.    0.991    0.001    0.991      0.991    0.991      0.990    1.000     0.997

=== Confusion Matrix ===

  a  b  c  d  e  f   <-- classified as
 70  0  0  0  0  0 |  a = 1
  0 76  0  0  0  0 |  b = 2
  0  0 17  0  0  0 |  c = 3
  0  0  0 12  0  1 |  d = 5
  0  0  0  0  9  0 |  e = 6
  0  0  1  0  0 28 |  f = 7
```

Status

OK

5.J48

J48 algorithm is one of the best machine learning algorithms to examine the data categorically and continuously. Here is output of it.



Classifier: **J48** -C 0.25 -M 2

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

(Nom) type

Result list (right-click for options)

```
23:53:36 - bayes.NaiveBayes
00:27:39 - functions.Logistic
00:27:59 - lazy.KStar
00:28:43 - trees.RandomForest
00:29:04 - trees.J48
```

Classifier output

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

Correctly Classified Instances         207               96.729  %
Incorrectly Classified Instances         7                3.271  %
Kappa statistic                          0.9557
Mean absolute error                      0.0109
Root mean squared error                  0.1044
Relative absolute error                  4.4212 %
Root relative squared error             29.7893 %
Total Number of Instances              214

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.986    0.000    1.000      0.986    0.993      0.989    0.993     0.990     1
                 0.987    0.007    0.987      0.987    0.987      0.980    0.990     0.979     2
                 1.000    0.020    0.810      1.000    0.895      0.891    0.990     0.810     3
                 0.846    0.000    1.000      0.846    0.917      0.915    0.923     0.855     5
                 0.889    0.005    0.889      0.889    0.889      0.884    0.942     0.795     6
                 0.931    0.005    0.964      0.931    0.947      0.939    0.963     0.907     7
Weighted Avg.    0.967    0.005    0.971      0.967    0.968      0.962    0.981     0.944

=== Confusion Matrix ===

  a  b  c  d  e  f   <-- classified as
 69  1  0  0  0  0 |  a = 1
  0 75  1  0  0  0 |  b = 2
  0  0 17  0  0  0 |  c = 3
  0  0  1 11  1  0 |  d = 5
  0  0  0  0  8  1 |  e = 6
  0  0  2  0  0 27 |  f = 7
```

Status

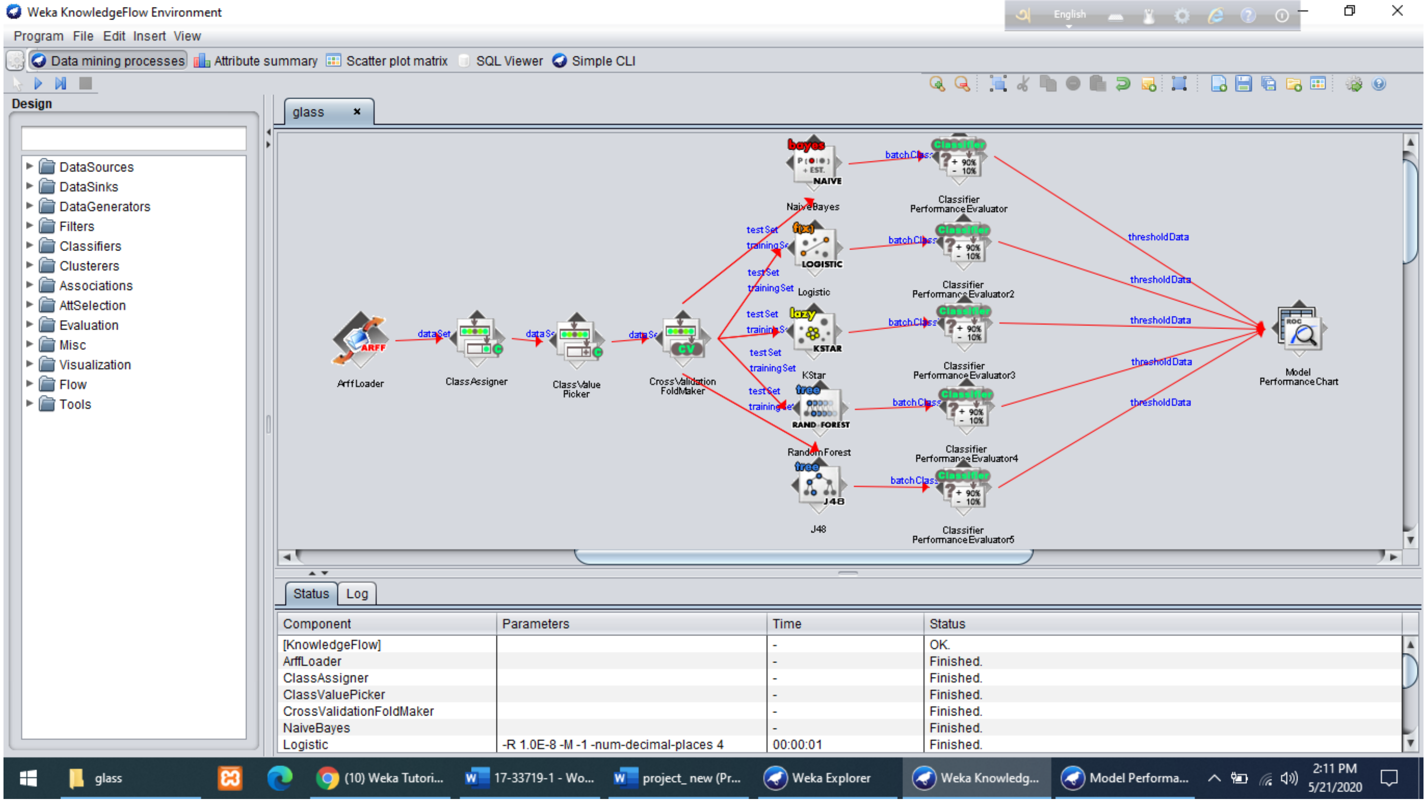
OK

Performance Analysis:

Classifiers	Correctly Classified instance	Total number of instances
NaiveBayes	180	214
Logistic	199	
Kstar	192	
RandomForest	**212**	
J48	207	

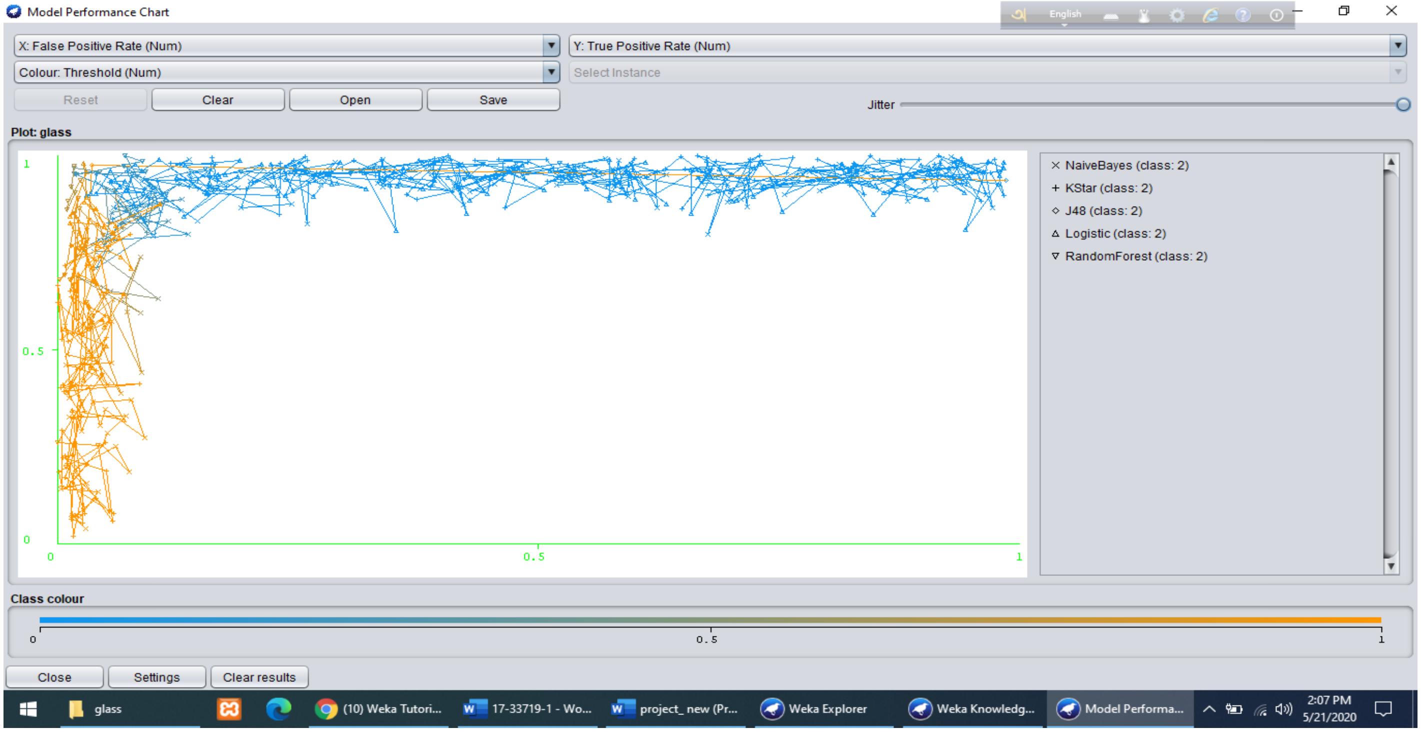
Here we see, among 214 instances Random Forest Correctly Classified 212 instance which is 99.0654 % and it's the highest percentage among the other Classifiers. So 'RandomForest' is the best classifier for the dataset.

Knowledge Flow Diagram



Component	Parameters	Time	Status
[KnowledgeFlow]		-	OK.
ArffLoader		-	Finished.
ClassAssigner		-	Finished.
ClassValuePicker		-	Finished.
CrossValidationFoldMaker		-	Finished.
NaiveBayes		-	Finished.
Logistic	-R 1.0E-8 -M -1 -num-decimal-places 4	00:00:01	Finished.

ROC Curve



Model Performance Chart

X: False Positive Rate (Num)

Y: True Positive Rate (Num)

Colour: Threshold (Num)

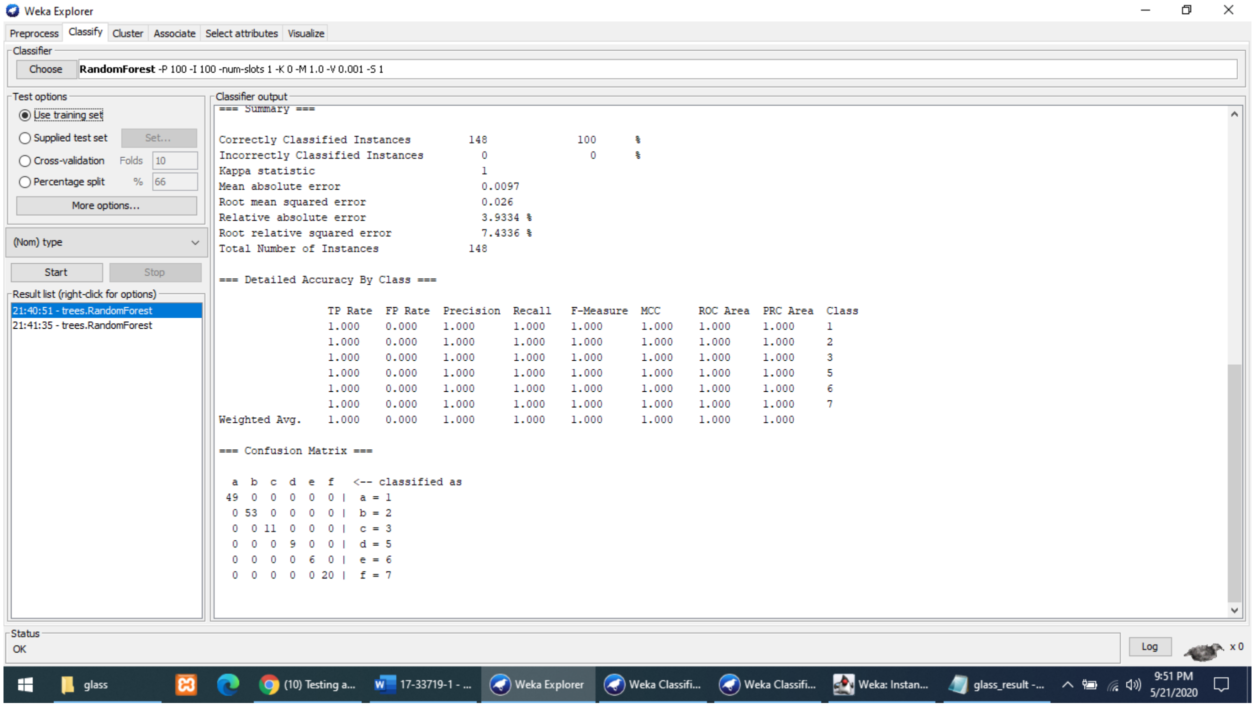
Plot: glass

This ROC curve is merged of all algorithm that we used, considering class value 2. We know that the curve goes nearer to the y-axis claimed as the better classifier. From the ROC curve we see that curve of RandomForest is nearer than the other curves. So, we can say that ‘RandomForest’ is best for the dataset.

Additional task

In additionally, we split data into two segments. I take 70% of original data as training set(148 out of 214 instances) and 30% of data as test set(61 out of 214 instances). This task is done by weka, and run this test data set with the training data set. Here we evaluate the performance of test data by visualizing the classifiers error.

Training set



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

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

(Nom) type

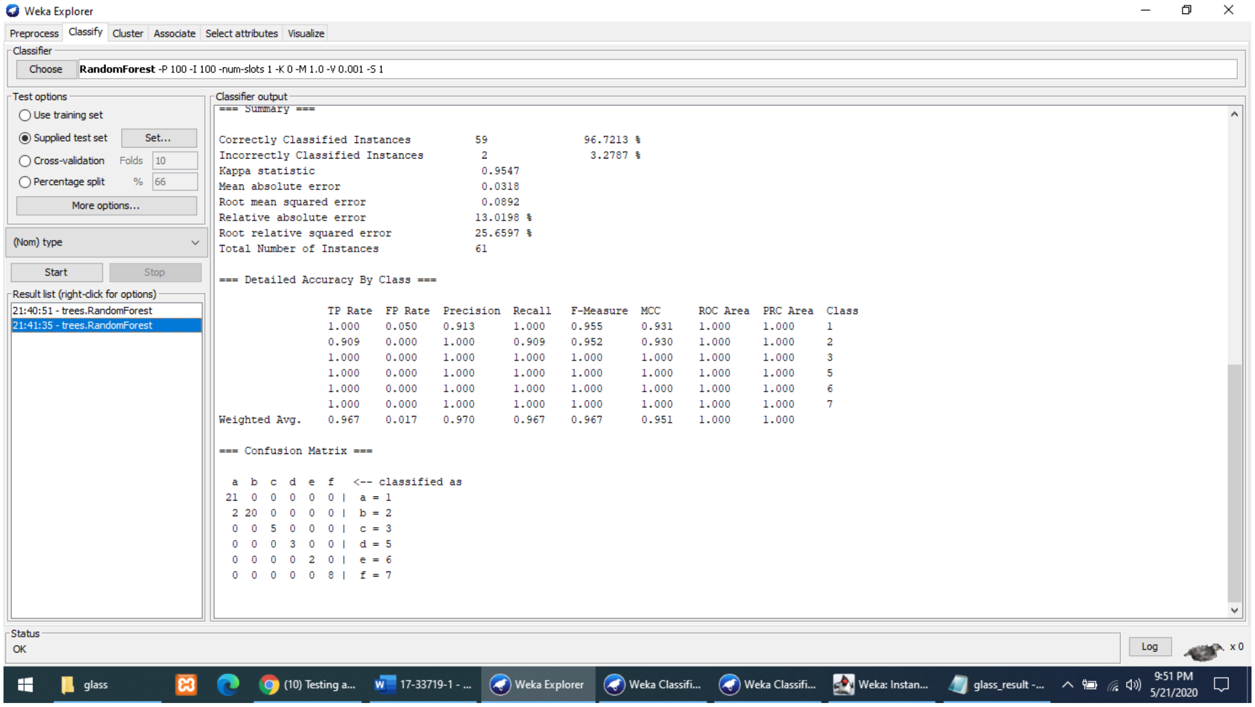
Result list (right-click for options)
21:40:51 - trees.RandomForest
21:41:35 - trees.RandomForest

Classifier output

```
=== Summary ===

Correctly Classified Instances         148              100      %
Incorrectly Classified Instances         0                0      %
Kappa statistic                          1
Mean absolute error                      0.0097
Root mean squared error                  0.026
Relative absolute error                  3.9334 %
Root relative squared error              7.4336 %
Total Number of Instances              148

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     1
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     2
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     3
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     5
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     6
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     7
Weighted Avg.    1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000

=== Confusion Matrix ===

  a  b  c  d  e  f   <-- classified as
 49  0  0  0  0  0 |  a = 1
  0 53  0  0  0  0 |  b = 2
  0  0 11  0  0  0 |  c = 3
  0  0  0  9  0  0 |  d = 5
  0  0  0  0  6  0 |  e = 6
  0  0  0  0  0 20 |  f = 7
```

Status
OK

Run with test data



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

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

(Nom) type

Result list (right-click for options)
21:40:51 - trees.RandomForest
21:41:35 - trees.RandomForest

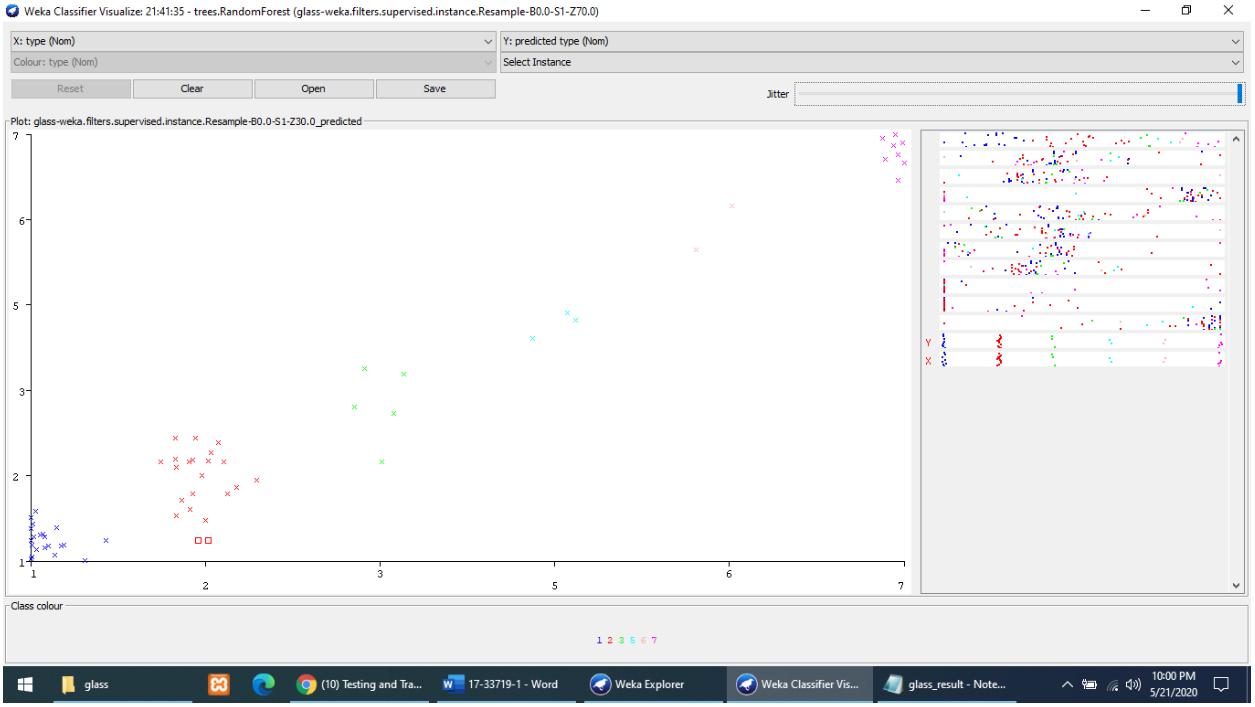
Classifier output

```
=== Summary ===

Correctly Classified Instances          59               96.7213 %
Incorrectly Classified Instances         2                3.2787 %
Kappa statistic                          0.9547
Mean absolute error                      0.0318
Root mean squared error                  0.0892
Relative absolute error                 13.0198 %
Root relative squared error             25.6597 %
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    0.050    0.913      1.000    0.955      0.931    1.000     1.000     1
                 0.909    0.000    1.000      0.909    0.952      0.930    1.000     1.000     2
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     3
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     5
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     6
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     7
Weighted Avg.    0.967    0.017    0.970      0.967    0.967      0.951    1.000     1.000

=== Confusion Matrix ===

  a  b  c  d  e  f   <-- classified as
 21  0  0  0  0  0 |  a = 1
  2 20  0  0  0  0 |  b = 2
  0  0  5  0  0  0 |  c = 3
  0  0  0  3  0  0 |  d = 5
  0  0  0  0  2  0 |  e = 6
  0  0  0  0  0  8 |  f = 7
```

Status
OK

From the Classifiers output, I determined that the performance is 96.7213 %. It Classified 59 instances correctly out of 61 test data set.

Classifiers error visualization



Weka Classifier Visualize: 21:41:35 - trees.RandomForest (glass-weka.filters.supervised.instance.Resample-B0.0-S1-Z70.0)

X: type (Nom)

Y: predicted type (Nom)

Colour: type (Nom)

Select Instance

Plot: glass-weka.filters.supervised.instance.Resample-B0.0-S1-Z30.0_predicted