**COURSE:CS699 DATA MINING**

**FINAL TERM PROJECT**

**NAME-ADIBA NISA**

**BU ID-U69235825**

**Submission Date-1st Dec 2016.**

**STEP 1-BUILDING CLASSIFICATION MODELS ON INITIAL DATA**

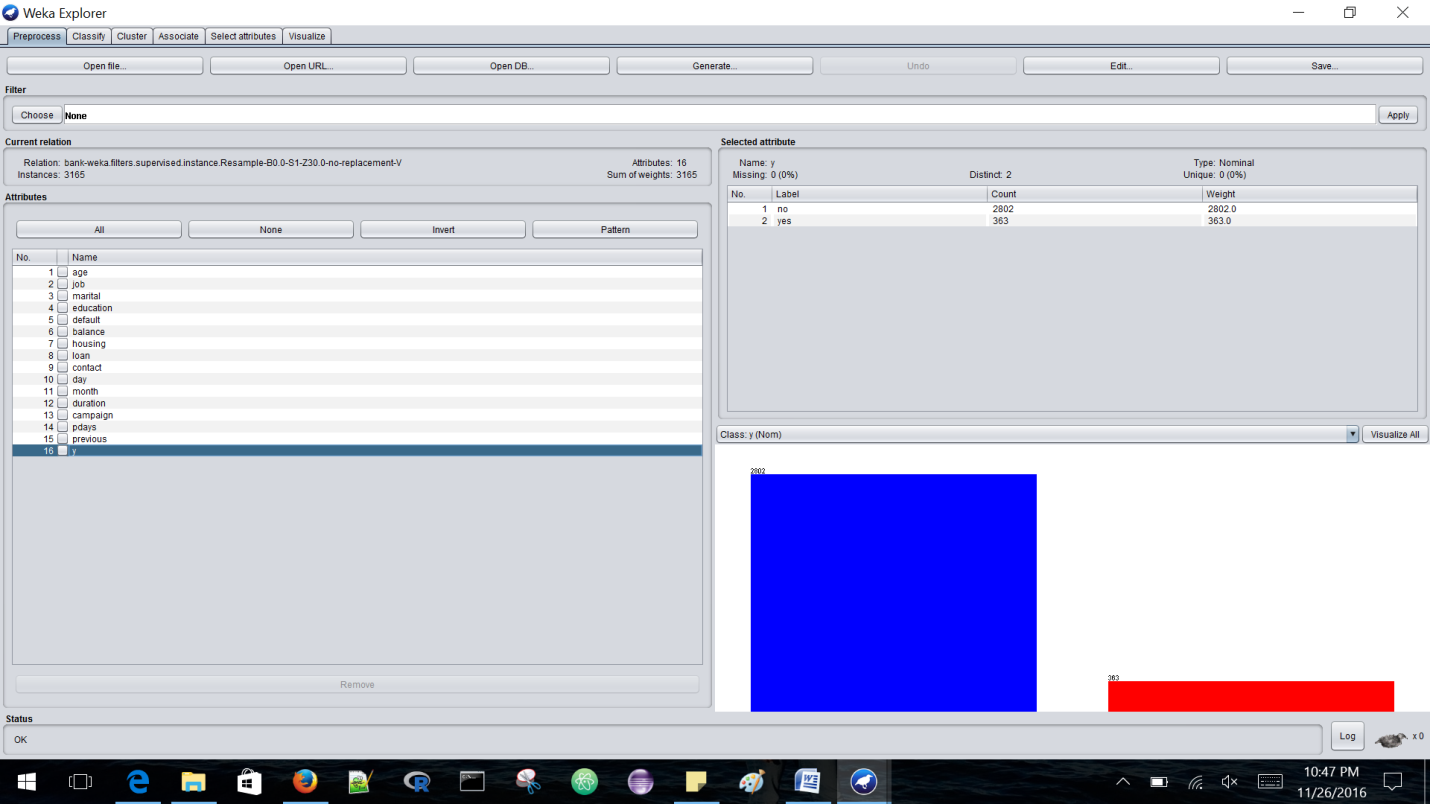
The initial dataset (which is used for this project) has 3165 tuples and 16 attributes. The class attribute is y and it takes on either yes or no. Among 3165 tuples, 2802 tuples are no tuples and 363 tuples are yes tuples. So, this dataset has a very unbalanced class distribution. If you build

a classification model using this dataset, the model will predict no tuples well but it won’t

predict yes tuples well.

A classifier is typically evaluated by a confusion matrix, the columns are the predicted class and the rows are the actual data class. In the confusion matrix, TN is the number of negative values correctly classified(True Negatives), FP is the number of negative values incorrectly classified as positive(False Positives), FN is the number of positive values incorrectly classified as negative(False Negatives) and TP is the number of positive values correctly classified(True Positives).

Let us see the bank\_cs699.arff, as to how it looks when we try to open it using weka tool.



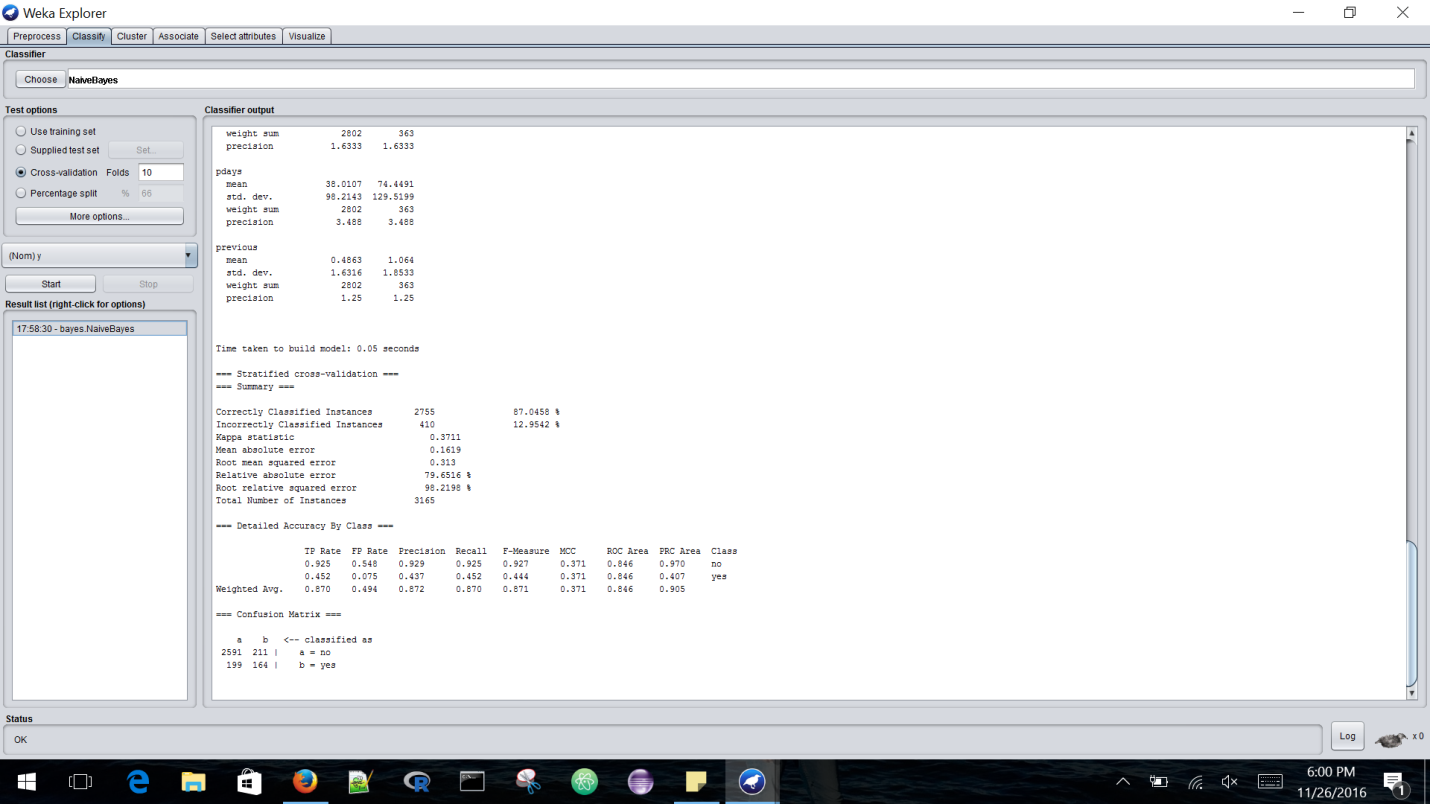
As we can see that the data is very imbalanced, meaning that the number of tuples classified as no are extremely high which is 2802 when compared to the number of tuples classified as yes which is 363. When we try to classify the data using some classifiers the TP rate for tuples classified as “no” would be very high when compared to the ones classified as “yes” which means that the number of tuples correctly classified as “no” are high.

1. **NAÏVE BAYES ALGORITHM-**

Let us try to classify the data using **Naïve Bayes Algorithm** with the default settings:

**Test Options:**

**Cross Validation with 10 folds.**



The accuracy with which this algorithm classified the data is as below:

|  |  |
| --- | --- |
| Correctly classified Instances | 87.058% |
| Incorrectly classified Instances | 12.952% |

Detailed Accuracy by class is provided below:

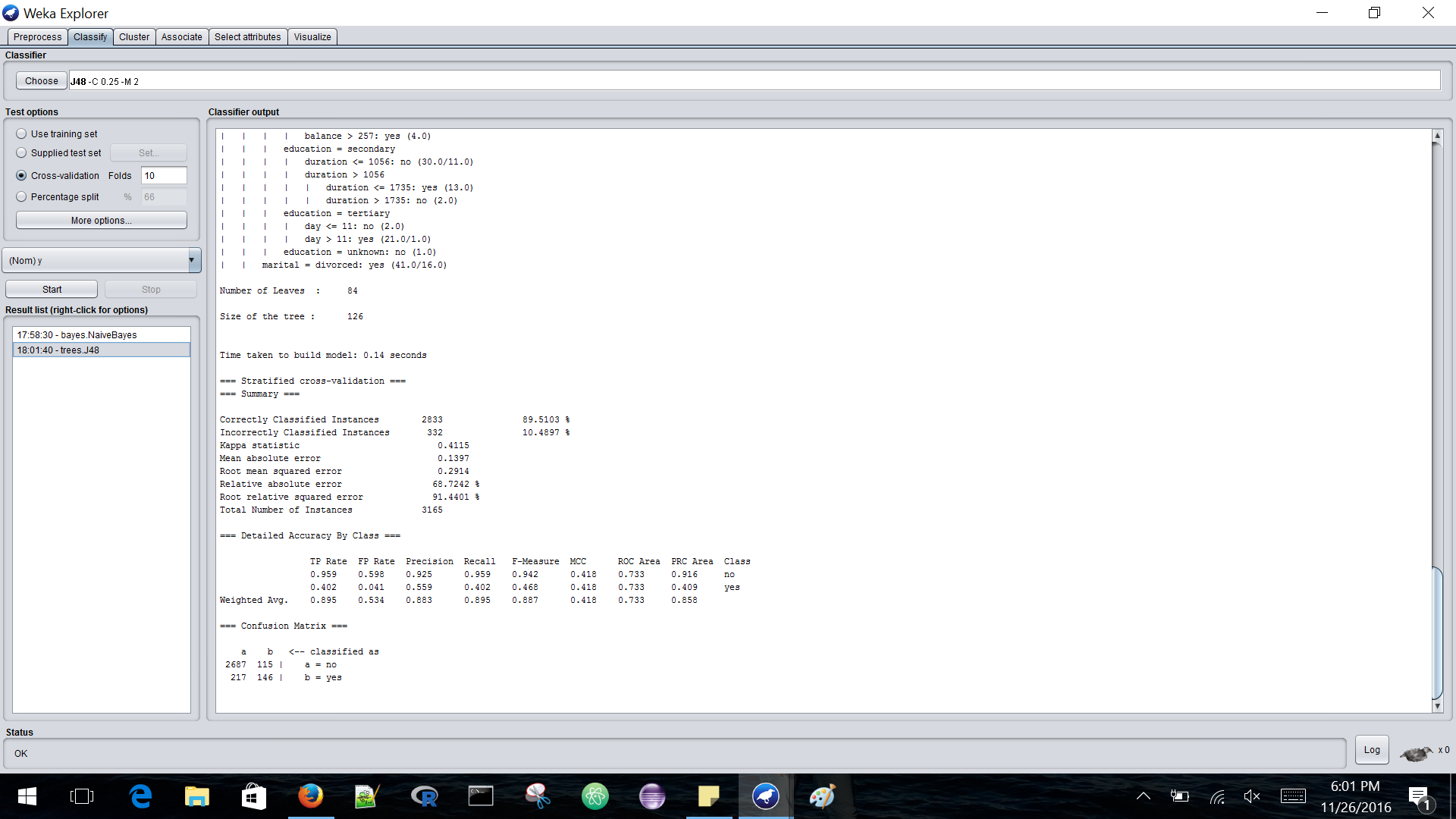
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **TP Rate** | **FP Rate** | **Precision** | **Recall** | **F-measure** | **ROC Area** | **Class** |
|  | 0.925 | 0.548 | 0.929 | 0.925 | 0.927 | 0.846 | No |
|  | 0.452 | 0.075 | 0.437 | 0.452 | 0.444 | 0.846 | yes |
| **Weighted Avg** | 0.870 | 0.494 | 0.872 | 0.870 | 0.871 | 0.846 |  |

1. **J-48 ALGORITHM:**

Now let us try to classify the data using **J-48 tree Algorithm** with the default settings:

**Test Options:**

**Cross Validation with 10 folds.**

****

The accuracy with which this algorithm classified the data is as below:

|  |  |
| --- | --- |
| Correctly classified Instances | 89.5103% |
| Incorrectly classified Instances | 10.4897% |

Detailed Accuracy by class is provided below:

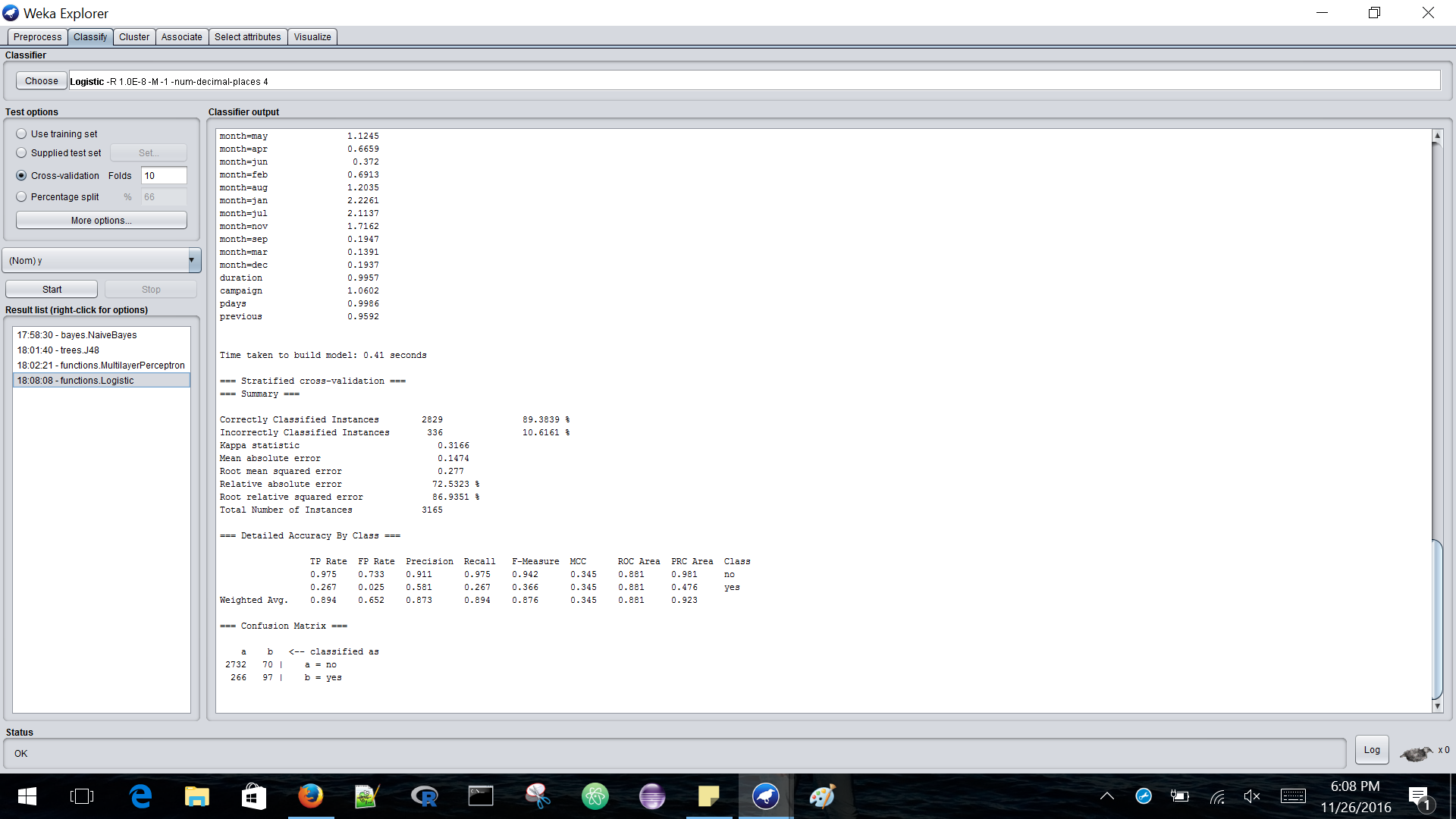
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **TP Rate** | **FP Rate** | **Precision** | **Recall** | **F-measure** | **ROC Area** | **Class** |
|  | 0.959 | 0.598 | 0.925 | 0.959 | 0.942 | 0.733 | No |
|  | 0.402 | 0.041 | 0.559 | 0.402 | 0.468 | 0.733 | yes |
| **Weighted Avg** | 0.895 | 0.534 | 0.883 | 0.895 | 0.887 | 0.733 |  |

1. **LOGISTIC ALGORITHM:**

Now let us try to classify the data using **Logistic Algorithm** with the default settings:

**Test Options:**

**Cross Validation with 10 folds.**



The accuracy with which this algorithm classified the data is as below:

|  |  |
| --- | --- |
| Correctly classified Instances | 89.3839% |
| Incorrectly classified Instances | 10.6161% |

Detailed Accuracy by class is provided below:

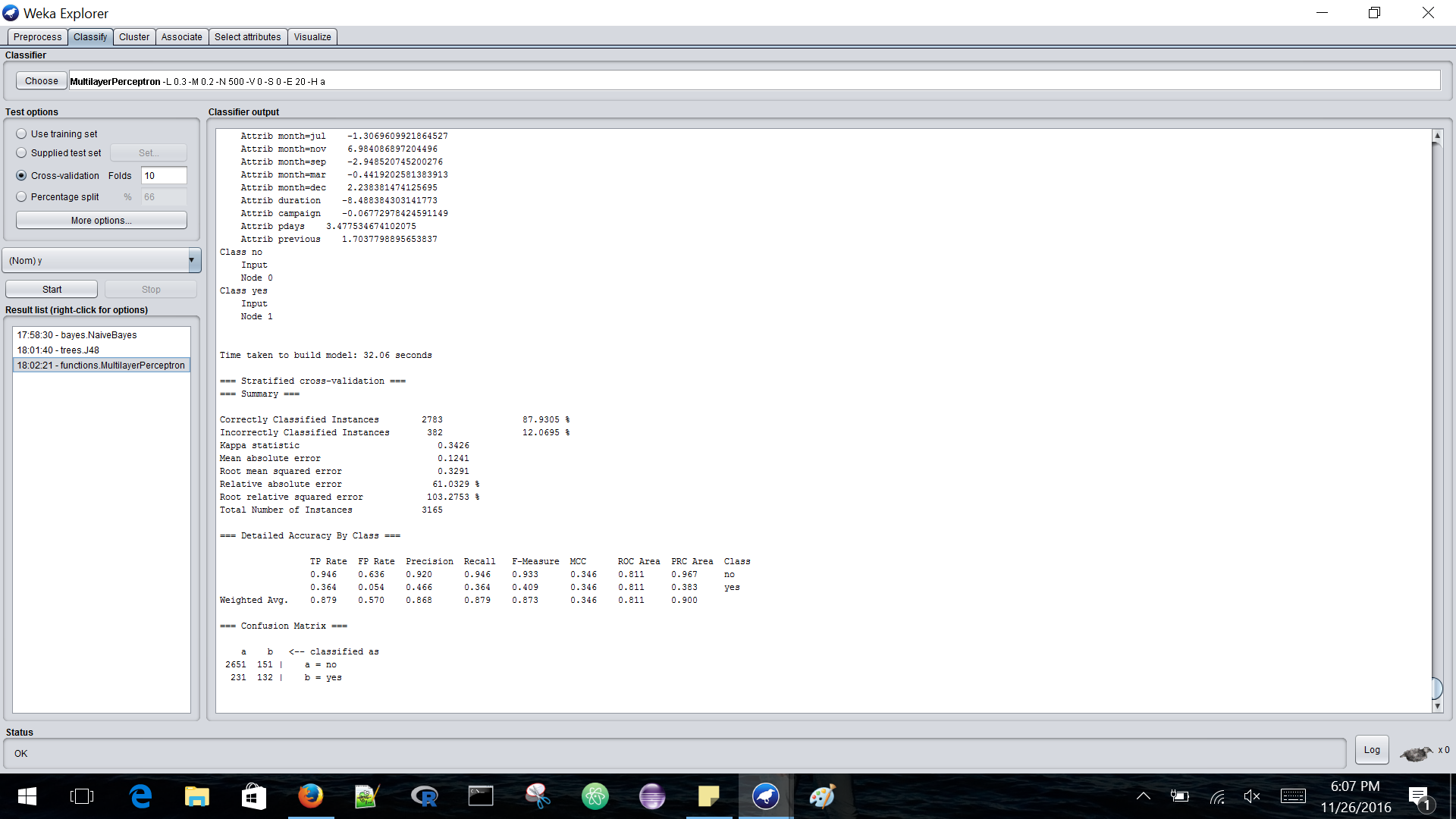
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **TP Rate** | **FP Rate** | **Precision** | **Recall** | **F-measure** | **ROC Area** | **Class** |
|  | 0.975 | 0.733 | 0.911 | 0.975 | 0.942 | 0.881 | No |
|  | 0.267 | 0.025 | 0.581 | 0.267 | 0.366 | 0.881 | yes |
| **Weighted Avg** | 0.894 | 0.652 | 0.873 | 0.894 | 0.876 | 0.881 |  |

1. **MULTILAYER PERCEPTRON ALGORITHM-**

Now let us try to classify the data using **Multilayer Perceptron Algorithm** with the default settings:

**Test Options:**

**Cross Validation with 10 folds.**

****

The accuracy with which this algorithm classified the data is as below:

|  |  |
| --- | --- |
| Correctly classified Instances | 87.9305% |
| Incorrectly classified Instances | 12.0695% |

Detailed Accuracy by class is provided below:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **TP Rate** | **FP Rate** | **Precision** | **Recall** | **F-measure** | **ROC Area** | **Class** |
|  | 0.946 | 0.636 | 0.920 | 0.946 | 0.933 | 0.811 | No |
|  | 0.364 | 0.054 | 0.466 | 0.364 | 0.409 | 0.811 | yes |
| **Weighted Avg** | 0.879 | 0.570 | 0.868 | 0.879 | 0.873 | 0.811 |  |

As we can see from the above classifiers that J-48 tree algorithm has the highest accuracy of 89.51% which means that it has correctly classified 89.51% of tuples but if we closely observe the TP-rate of “no” labeled tuples is 0.959 which is very high. The accuracy might not be appropriate when the data is imbalanced, To improve the accuracy of the data we can follow a general approach of over sampling or undersampling the data.

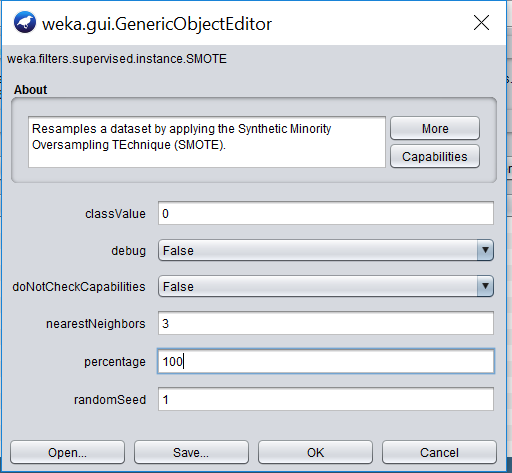
**STEP 2-OVERSAMPLING**

In this method, data is distributed so that the rare(positive) class is well represented, the technique followed here is that it resamples the positive tuples so that resulting training set contains equal number of positive and negative tuples.

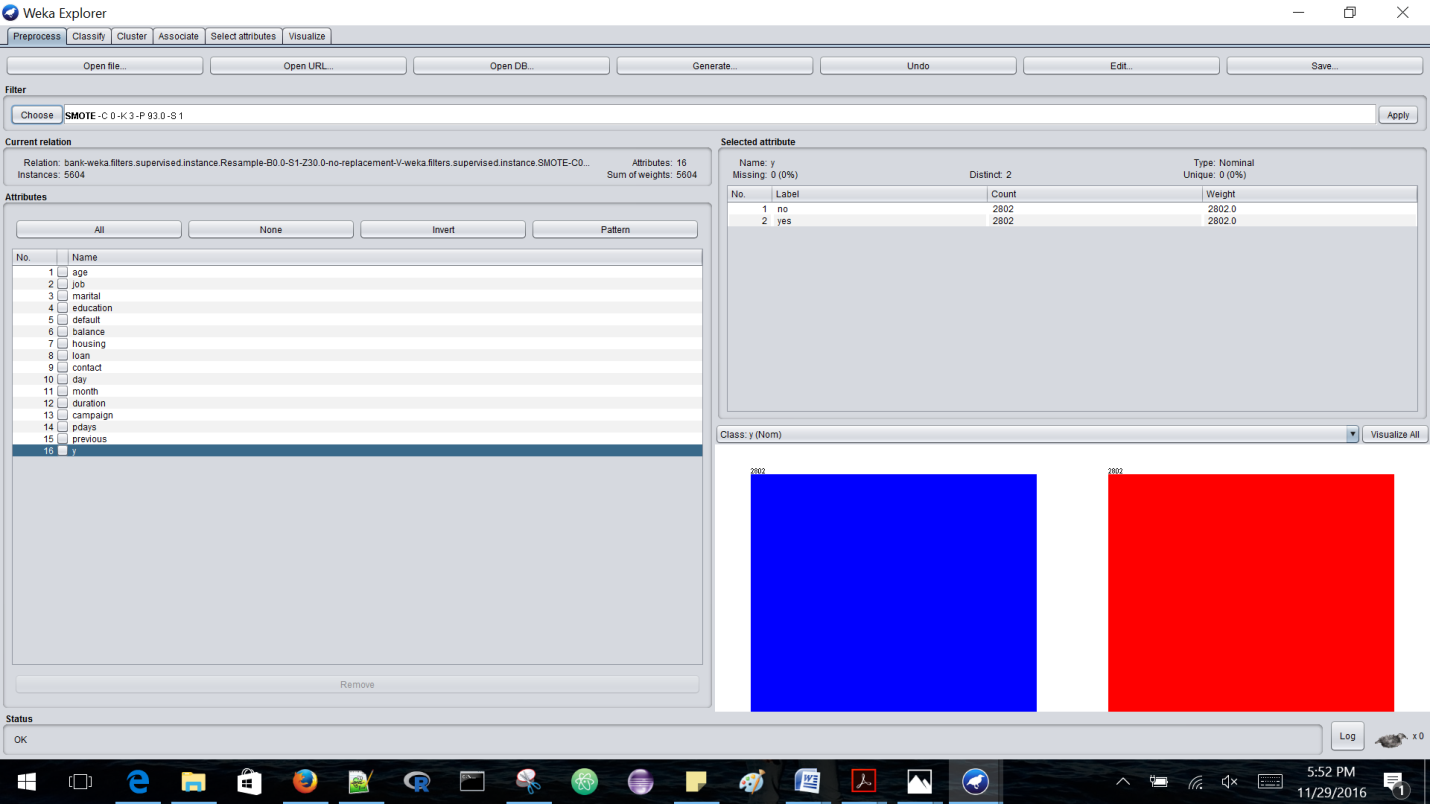
For this data let us use SMOTE Algorithm to resample the data, where the synthetic tuples are added which are close to the given positive tuples in tuple space.

We can download the SMOTE algorithm from the package manager.

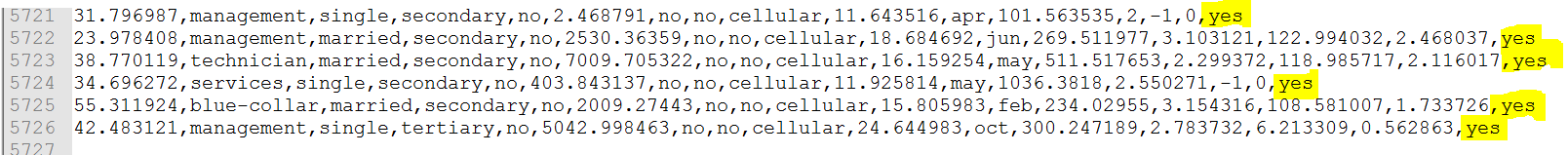
When applying SMOTE Algorithm use the below default options:

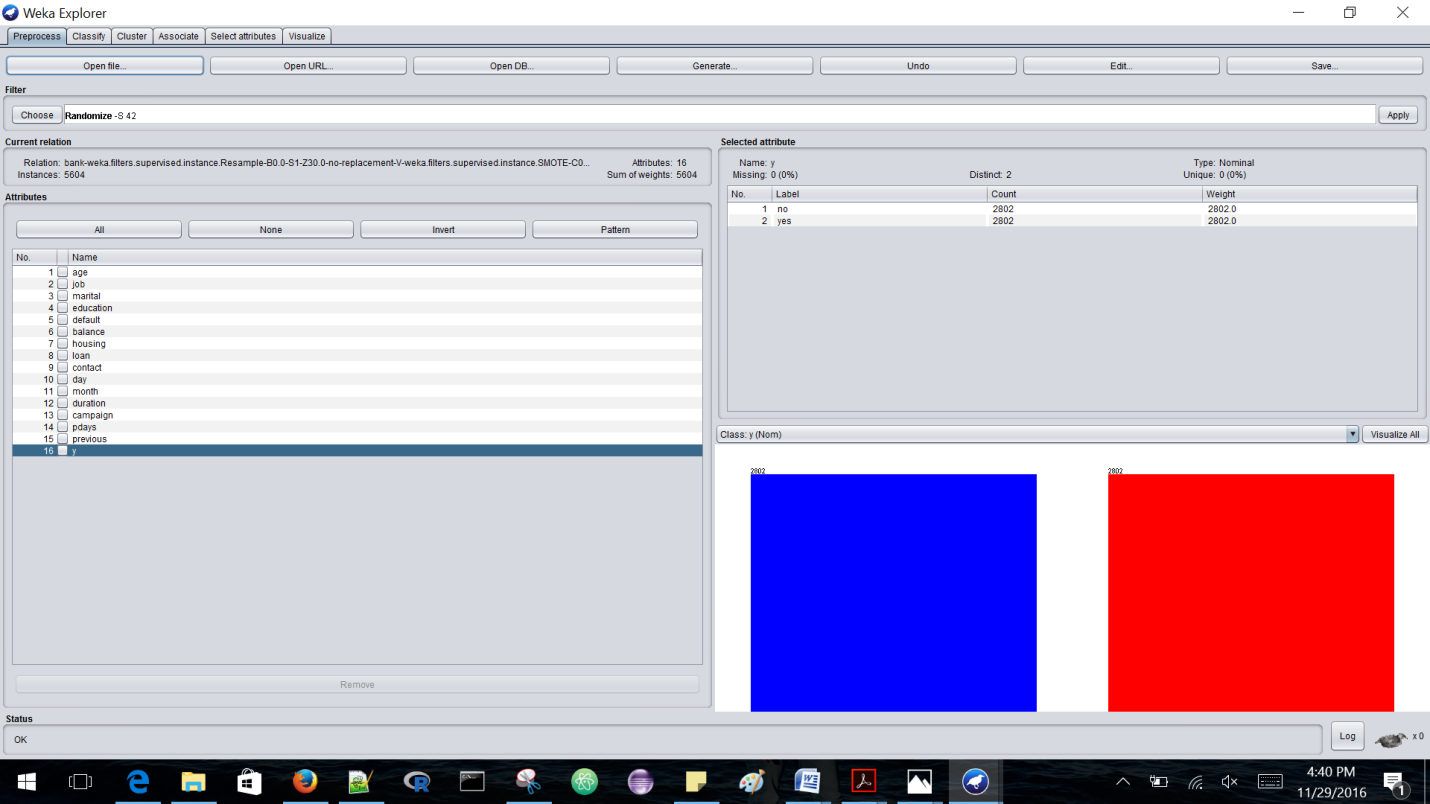


Here perecentage is 100 which means that the positive tuples are duplicated completely. When we apply this algorithm once we get 726 yes tuples and 2802 no tuples. When we apply this algorithm again to the same data set, the positive tuples get duplicated again which means that now we have 1452 yes tuples and 2802 no tuples.Let us try to apply SMOTE again but this time with **93** percentage instead of 100 so we get 2802 yes and 2802 no tuples.  
Also note that SMOTE Algorithm works best on this data when nearest neighbours are-**3.**

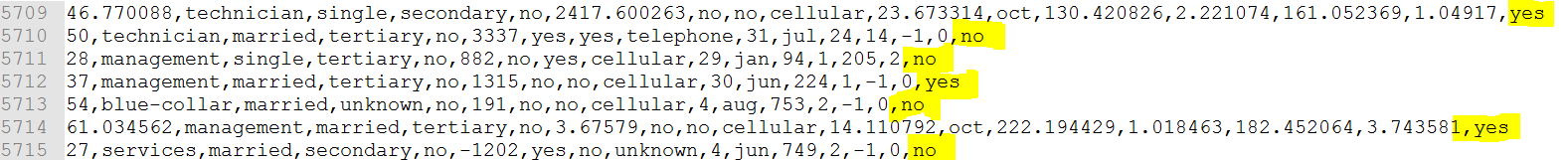


Now save the data using the save button on top, if we open the .arff file from backend we can see that replicated data is piled up in order like, all “yes” tuples that were replicated are all together as shown below:



This will cause overfitting problem, so let us apply the Randomize algorithm which will arrange all the tuples randomly:

Now save the data as another .arff file, we can see that the tuples are arranged randomly as below:

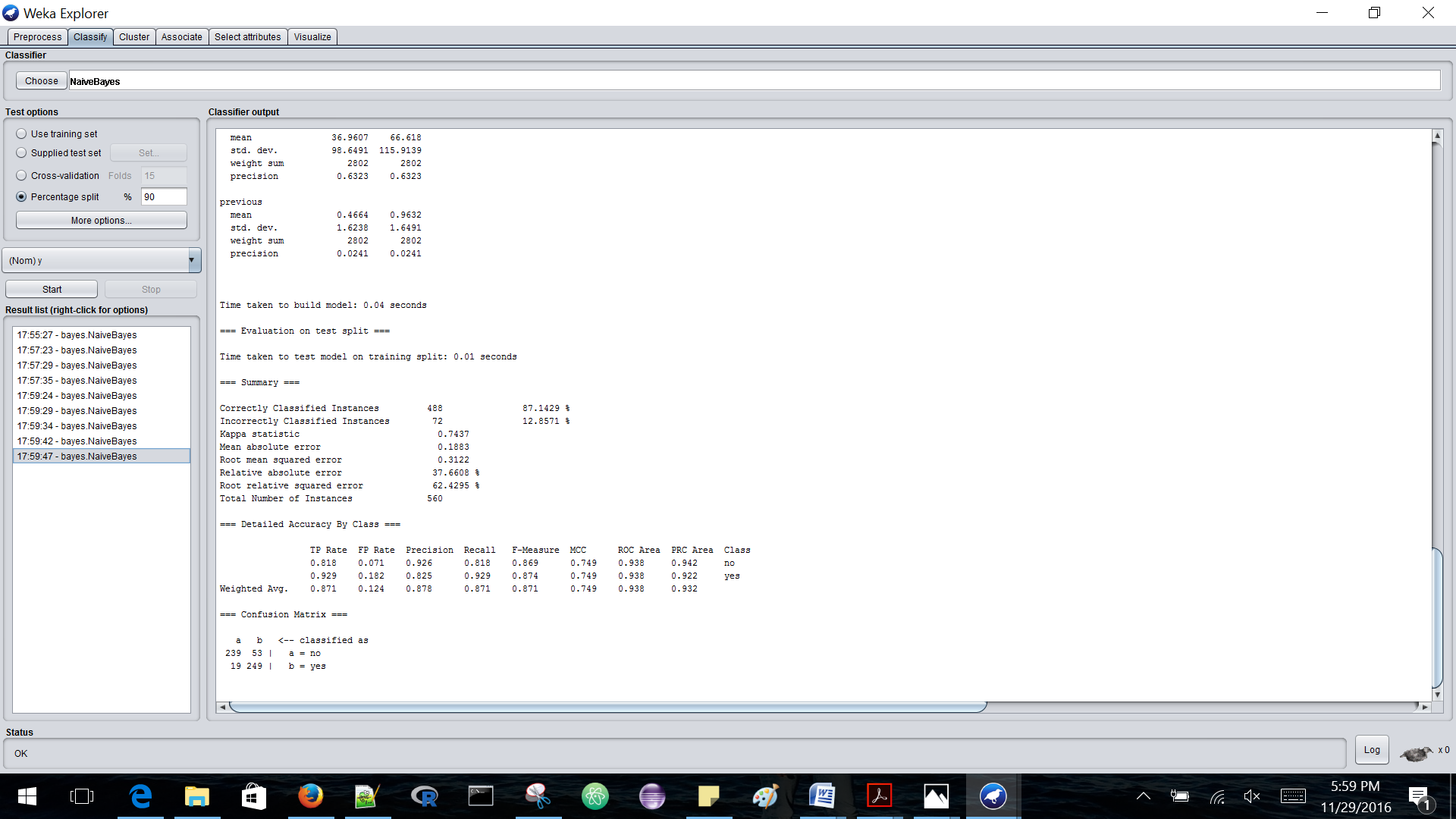


Let us do apply some classifiers on the above saved data.

1. **NAÏVE BAYES ALGORITHM-**

|  |  |
| --- | --- |
| **TESTING OPTIONS** | **ACCURACY OBTAINED** |
| Accuracy of Naïve Bayes using 10 folds cross validation | 86.7595% |
| Accuracy of Naïve Bayes using training set | 86.7773% |
| Accuracy of Naïve Bayes using 90 percentage split | 87.1429% |

Naïve Bayes using Percentage split with 90% gives best values as below:



The accuracy with which this algorithm classified the data is as below:

|  |  |
| --- | --- |
| Correctly classified Instances | 87.1429% |
| Incorrectly classified Instances | 12.8571% |

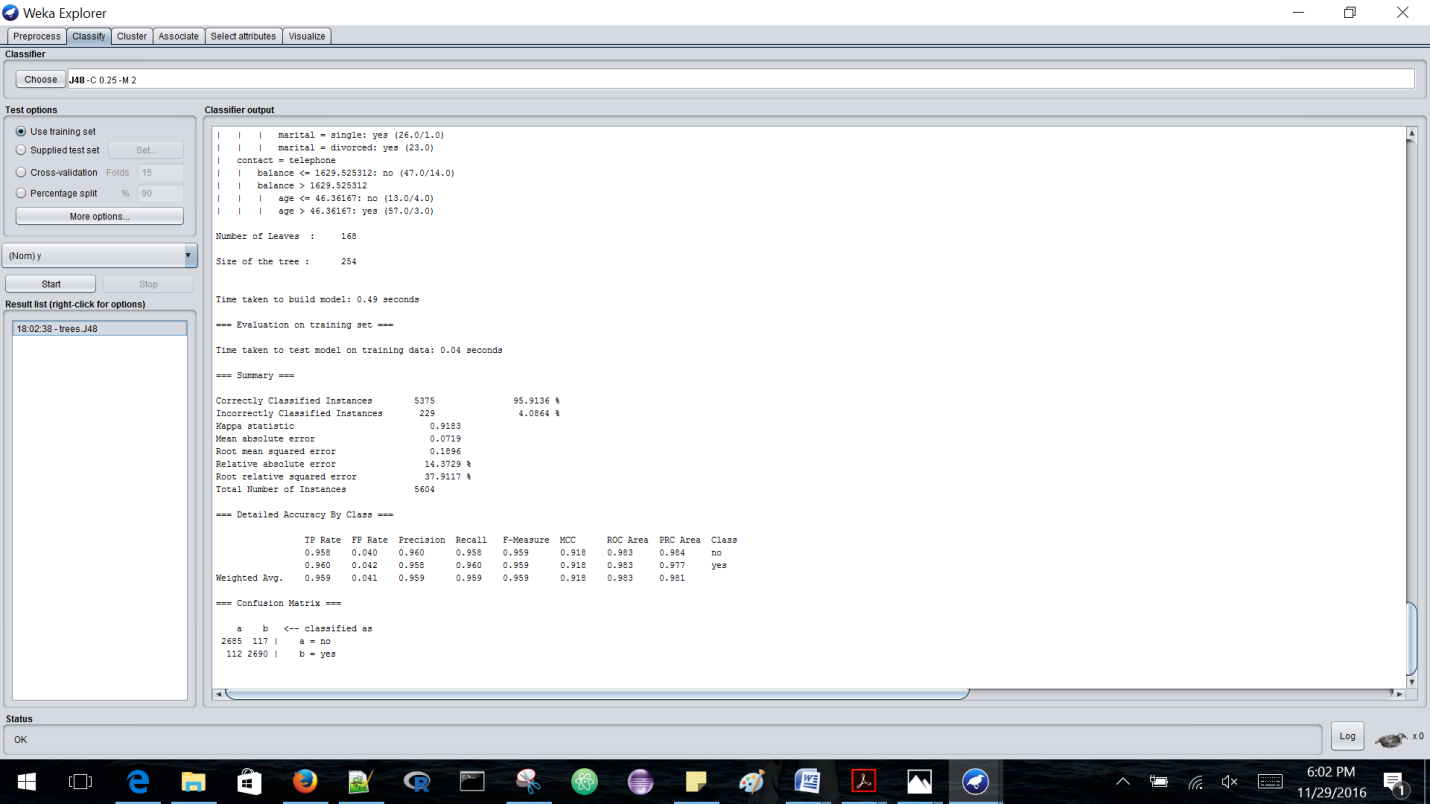
Detailed accuracy by class is provided as below:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **TP Rate** | **FP Rate** | **Precision** | **Recall** | **F-measure** | **ROC Area** | **Class** |
|  | 0.818 | 0.071 | 0.926 | 0.818 | 0.869 | 0.938 | No |
|  | 0.929 | 0.182 | 0.825 | 0.929 | 0.874 | 0.938 | yes |
| **Weighted Avg** | 0.871 | 0.124 | 0.878 | 0.871 | 0.871 | 0.938 |  |

1. **J-48 ALGORITHM -**

|  |  |
| --- | --- |
| **TESTING OPTIONS** | **ACCURACY OBTAINED** |
| Accuracy of J-48 using training set | 95.9136% |
| Accuracy of J-48 using 22 folds cross validation | 93.0407% |
| Accuracy of J-48 using 90 percentage split | 94.1071% |

J48 gives best accuracy with test option as Training set, below are more details:



The accuracy with which this algorithm classified the data is as below:

|  |  |
| --- | --- |
| Correctly classified Instances | 95.9136% |
| Incorrectly classified Instances | 4.0864% |

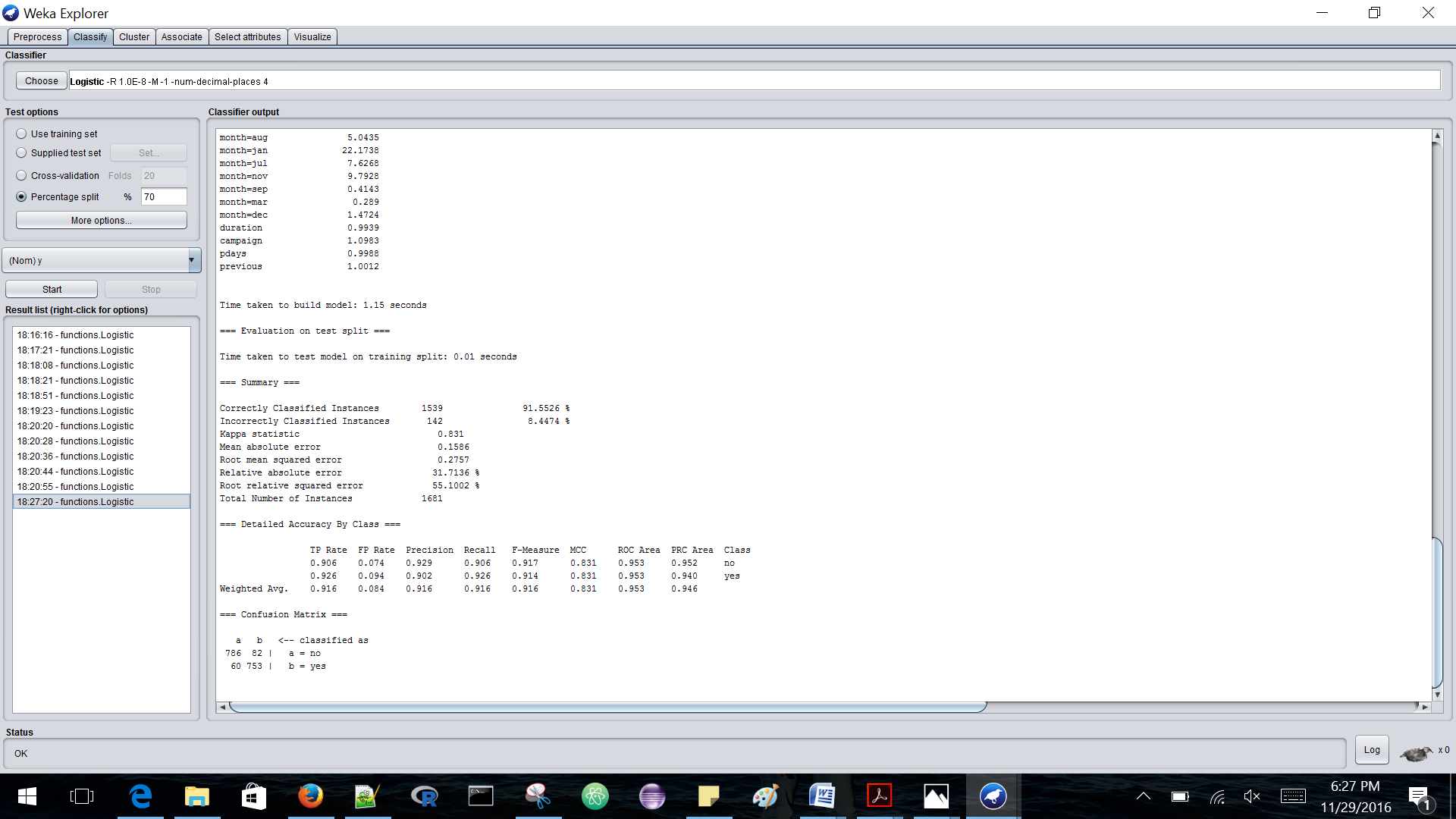
Detailed Accuracy by class is provided below:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **TP Rate** | **FP Rate** | **Precision** | **Recall** | **F-measure** | **ROC Area** | **Class** |
|  | 0.958 | 0.040 | 0.960 | 0.958 | 0.959 | 0.983 | No |
|  | 0.960 | 0.042 | 0.958 | 0.960 | 0.959 | 0.983 | yes |
| **Weighted Avg** | 0.959 | 0.041 | 0.959 | 0.959 | 0.959 | 0.983 |  |

1. **LOGISTIC ALGORITHM-**

|  |  |
| --- | --- |
| **TESTING OPTIONS** | **ACCURACY OBTAINED** |
| Accuracy of Logistic using training set | 91.1849% |
| Accuracy of Logistic using 20 folds cross validation | 90.8815% |
| Accuracy of Logistic Algorithm using 70 percentage split | 91.5526% |

Logistic algorithm works best with 70 percentage split as testing option, let us see more details below-



The accuracy with which this algorithm classified the data is as below:

|  |  |
| --- | --- |
| Correctly classified Instances | 91.5526% |
| Incorrectly classified Instances | 8.4474% |

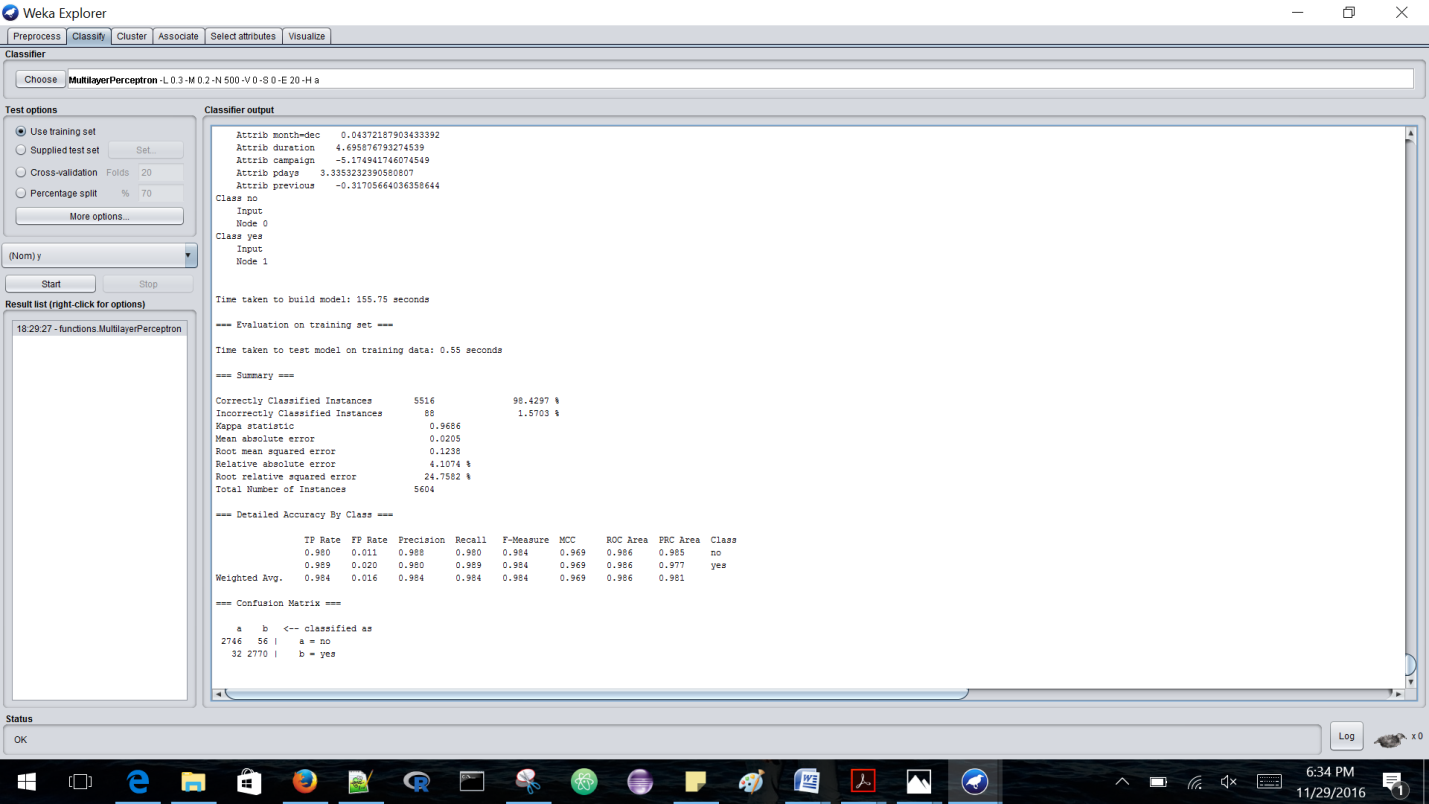
Detailed Accuracy by class is provided below:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **TP Rate** | **FP Rate** | **Precision** | **Recall** | **F-measure** | **ROC Area** | **Class** |
|  | 0.906 | 0.074 | 0.929 | 0.906 | 0.917 | 0.953 | No |
|  | 0.926 | 0.094 | 0.902 | 0.926 | 0.914 | 0.953 | yes |
| **Weighted Avg** | 0.916 | 0.084 | 0.916 | 0.916 | 0.916 | 0.953 |  |

1. **MULTILAYER PERCEPTRON ALGORITHM-**

|  |  |
| --- | --- |
| **TESTING OPTIONS** | **ACCURACY OBTAINED** |
| Accuracy of Multilayer using training set | 98.4297% |
| Accuracy of Multilayer using 20 folds cross validation | 92.434% |
| Accuracy of Multilayer using 70 percentage split | 92.231% |
| Accuracy of Multilayer using 90 percentage split | 92.6786% |

MultiLayer Perceptron algorithm with testing option as training set gives us the best result with 98.4% accuracy, let us look at it in more detail as below:



The accuracy with which this algorithm classified the data is as below:

|  |  |
| --- | --- |
| Correctly classified Instances | 98.4297% |
| Incorrectly classified Instances | 1.5703% |

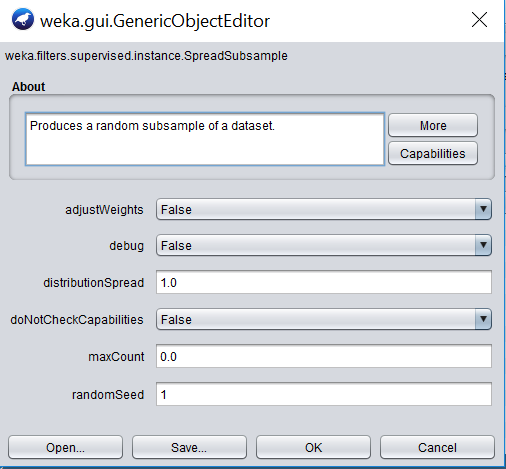
Detailed Accuracy by class is provided below:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **TP Rate** | **FP Rate** | **Precision** | **Recall** | **F-measure** | **ROC Area** | **Class** |
|  | 0.980 | 0.011 | 0.988 | 0.980 | 0.984 | 0.986 | No |
|  | 0.989 | 0.020 | 0.980 | 0.989 | 0.989 | 0.986 | yes |
| **Weighted Avg** | 0.984 | 0.016 | 0.984 | 0.984 | 0.984 | 0.986 |  |

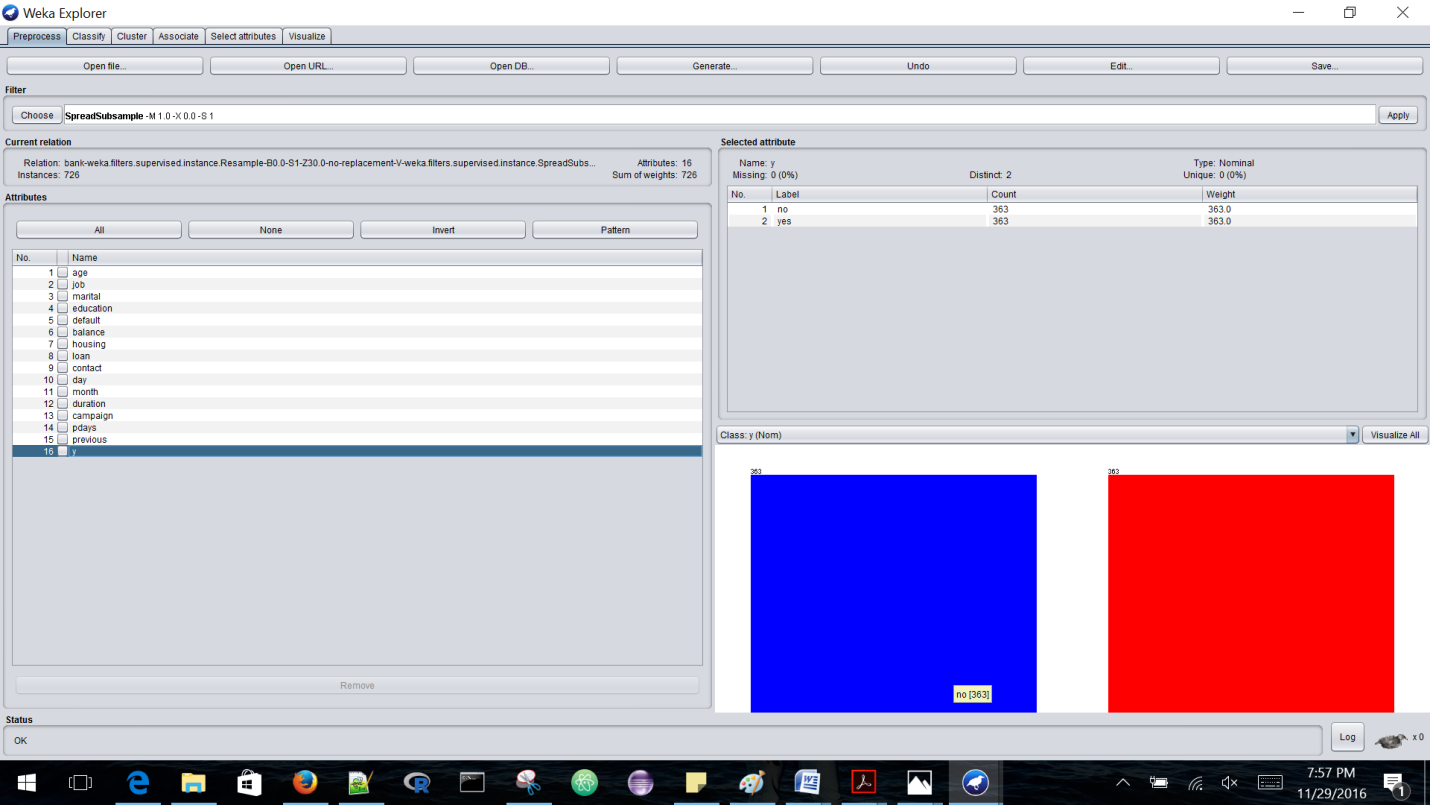
**STEP 3-UNDERSAMPLING**

Undersampling is another technique which we use on imbalanced data, this technique works by decreasing the number of negative tuples from the majority(negative) tuples. It randomly eliminates tuples from the majority(negative) class until there are an equal number of positive and negative tuples.

For undersampling, let us use SpreadSubSample Algorithm and set values as below-(Distribution Speed is 1.0)



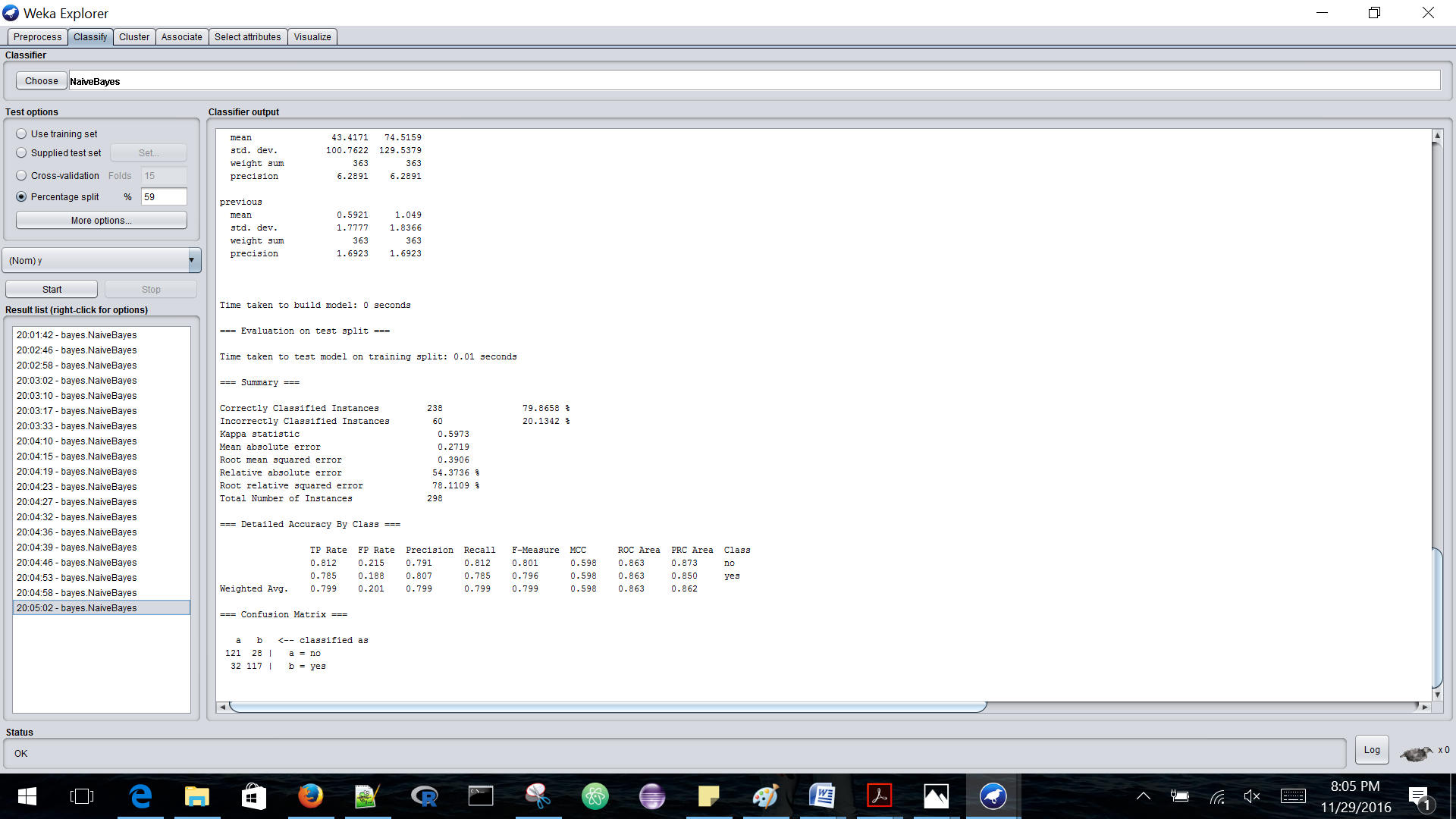
The undersampled data obtained is as below-



1. **NAÏVE BAYES ALGORITHM-**

|  |  |
| --- | --- |
| **TESTING OPTIONS** | **ACCURACY OBTAINED** |
| Accuracy of Naïve Bayes using training set | 79.4776% |
| Accuracy of Naïve Bayes using 15 folds cross validation | 77.4105% |
| Accuracy of Naïve Bayes using 59 percentage split | 79.8658% |

This algorithm works best with testing option as 59 percentage split as shown below-



The accuracy with which this algorithm classified the data is as below:

|  |  |
| --- | --- |
| Correctly classified Instances | 79.8658% |
| Incorrectly classified Instances | 20.1342% |

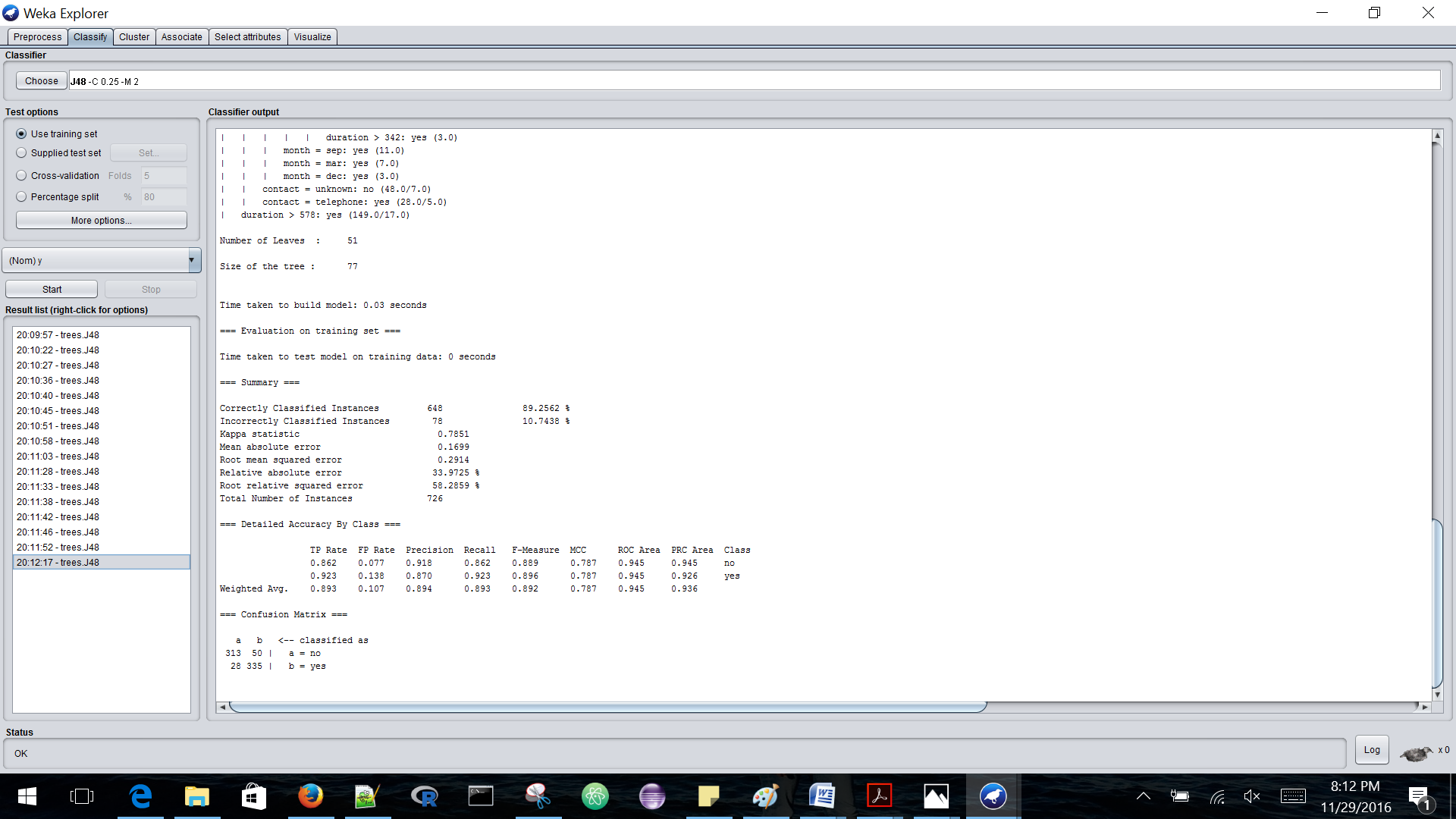
Detailed Accuracy by class is provided below:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **TP Rate** | **FP Rate** | **Precision** | **Recall** | **F-measure** | **ROC Area** | **Class** |
|  | 0.812 | 0.215 | 0.791 | 0.812 | 0.801 | 0.863 | No |
|  | 0.785 | 0.188 | 0.807 | 0.785 | 0.796 | 0.863 | yes |
| **Weighted Avg** | 0.799 | 0.201 | 0.799 | 0.799 | 0.799 | 0.863 |  |

1. **J-48 ALGORITHM-**

|  |  |
| --- | --- |
| **TESTING OPTIONS** | **ACCURACY OBTAINED** |
| Accuracy of J-48 using training set | 89.2562% |
| Accuracy of J-48 using 5 folds cross validation | 81.1295 |
| Accuracy of J-48 using 80 percentage split | 79.3103% |

J48 Algorithm works best with test option as Training Set as shown below-



The accuracy with which this algorithm classified the data is as below:

|  |  |
| --- | --- |
| Correctly classified Instances | 89.2562% |
| Incorrectly classified Instances | 10.7438% |

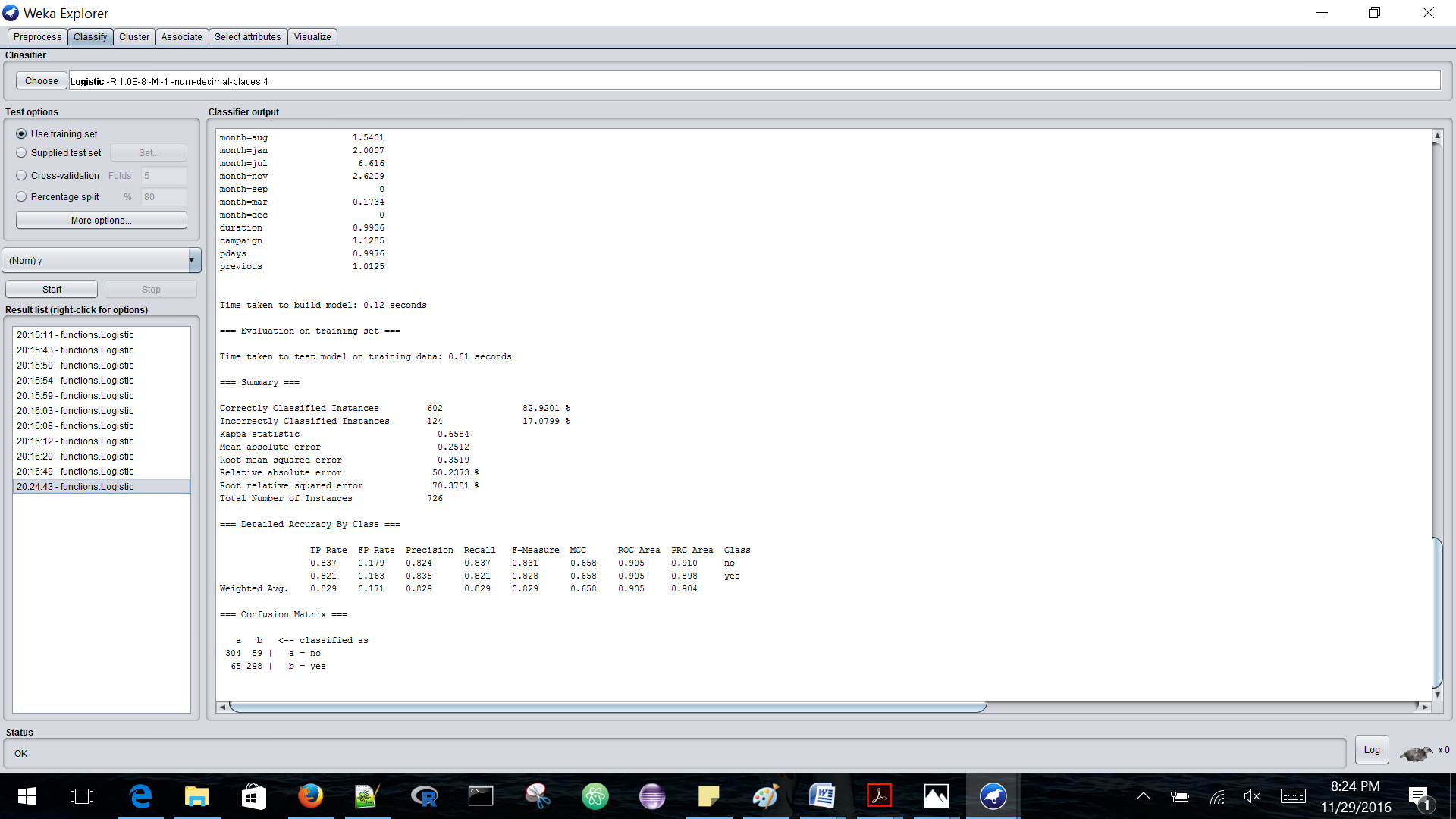
Detailed Accuracy by class is provided below:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **TP Rate** | **FP Rate** | **Precision** | **Recall** | **F-measure** | **ROC Area** | **Class** |
|  | 0.862 | 0.077 | 0.918 | 0.862 | 0.889 | 0.945 | No |
|  | 0.923 | 0.138 | 0.870 | 0.923 | 0.896 | 0.945 | yes |
| **Weighted Avg** | 0.893 | 0.107 | 0.894 | 0.893 | 0.892 | 0.945 |  |

1. **LOGISTIC ALGORITHM-**

|  |  |
| --- | --- |
| **TESTING OPTIONS** | **ACCURACY OBTAINED** |
| Accuracy of Logistic using training set | 82.9201% |
| Accuracy of Logistic using 5 folds cross validation | 78.6501% |
| Accuracy of Logistic using 80 percentage split | 78.6207% |

This algorithm gives best accuracy with test option set to training set-



The accuracy with which this algorithm classified the data is as below:

|  |  |
| --- | --- |
| Correctly classified Instances | 82.9201% |
| Incorrectly classified Instances | 17.0799% |

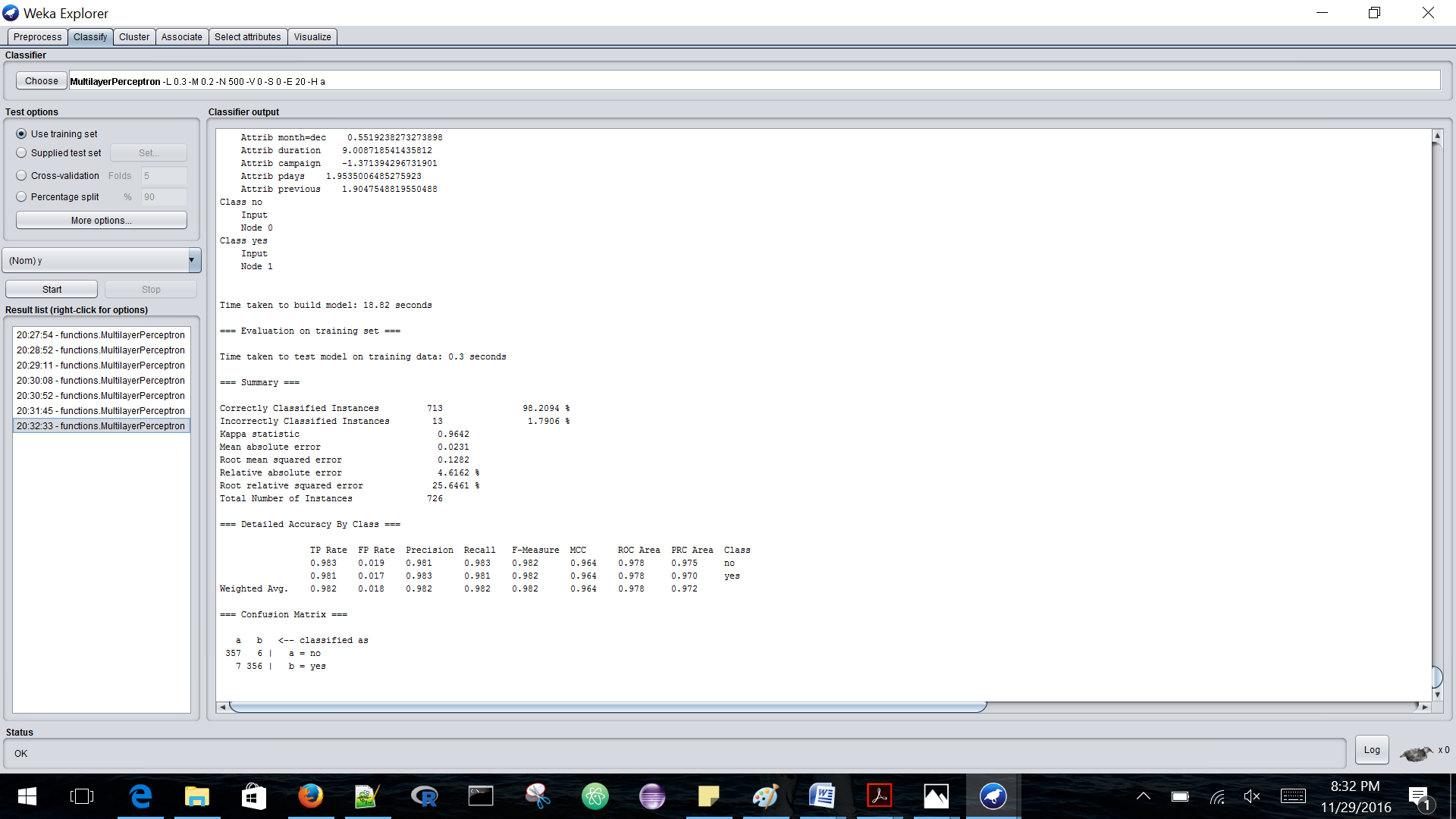
Detailed Accuracy by class is provided below:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **TP Rate** | **FP Rate** | **Precision** | **Recall** | **F-measure** | **ROC Area** | **Class** |
|  | 0.837 | 0.179 | 0.824 | 0.837 | 0.831 | 0.905 | No |
|  | 0.821 | 0.163 | 0.835 | 0.821 | 0.828 | 0.905 | yes |
| **Weighted Avg** | 0.829 | 0.171 | 0.829 | 0.829 | 0.829 | 0.905 |  |

1. **MULTILAYER ALGORITHM-**

|  |  |
| --- | --- |
| **TESTING OPTIONS** | **ACCURACY OBTAINED** |
| Accuracy of Multilayer using training set | 98.2094% |
| Accuracy of Multilayer using 5 folds cross validation | 75.4821% |
| Accuracy of Multilayer using 60 percentage split | 75.2294% |

This algorithm works best with test option set to training set-



The accuracy with which this algorithm classified the data is as below:

|  |  |
| --- | --- |
| Correctly classified Instances | 98.2094% |
| Incorrectly classified Instances | 1.7906% |

Detailed Accuracy by class is provided below:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | **TP Rate** | **FP Rate** | **Precision** | **Recall** | **F-measure** | **ROC Area** | **Class** |
|  | 0.983 | 0.019 | 0.981 | 0.983 | 0.982 | 0.978 | No |
|  | 0.981 | 0.017 | 0.983 | 0.981 | 0.982 | 0.978 | yes |
| **Weighted Avg** | 0.982 | 0.018 | 0.982 | 0.982 | 0.982 | 0.978 |  |

**STEP 4-CONCLUSION**

If we observe closely, then we can notice that multilayer perceptron works well on both oversampled and undersampled data. The accuracy of this algorithm on oversampled data is 98.4297% and accuracy on undersampled data is 98.2094%.

With ROC value of 0.986 for oversampled data which is higher than undersampled data of 0.978. And also the TP-rate for positive tuples is higher for oversampled data which is 0.989 and for undersampled it is 0.981.

Therefore, we can conclude that the **Multilayer Perceptron is best among all with accuracy of 98.4297%.**

**REFERENCES**

* Jiawei Han, M. Kamber, and J. Pei, "Data Mining Concepts and Techniques",Third Ed., 2012, Morgan Kaufmann
* https://www3.nd.edu/~dial/publications/chawla2005data.pdf