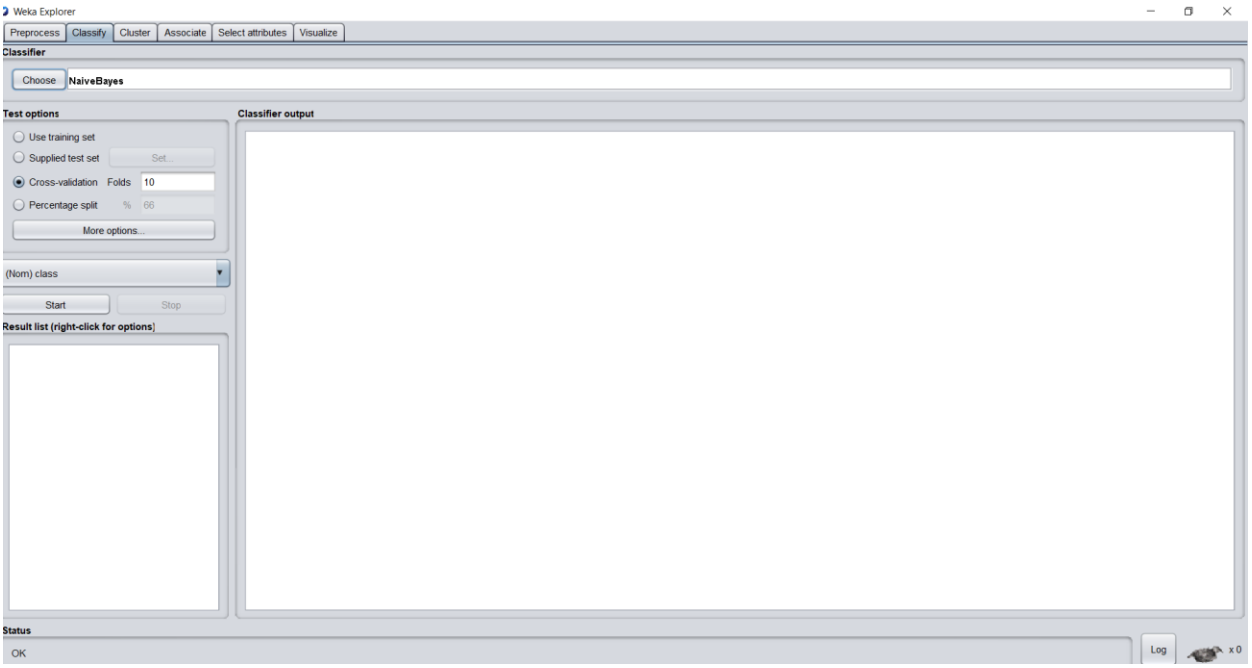
SIT719 Security and Privacy Issues in Analytics

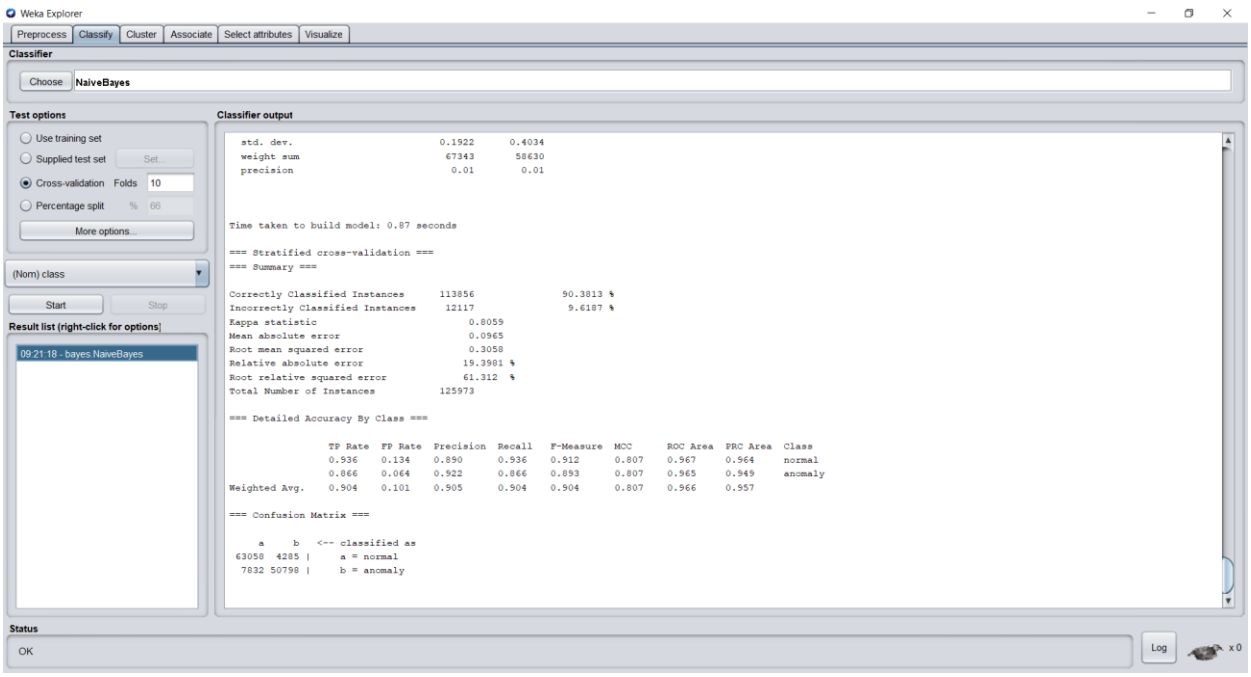
CREDIT TASK 4.2: ATTACK CLASSIFICATION USING NAÏVE BAYES ALGORITHM

Now apply "Naïve Bayes" classification algorithm from the "Classify" tab



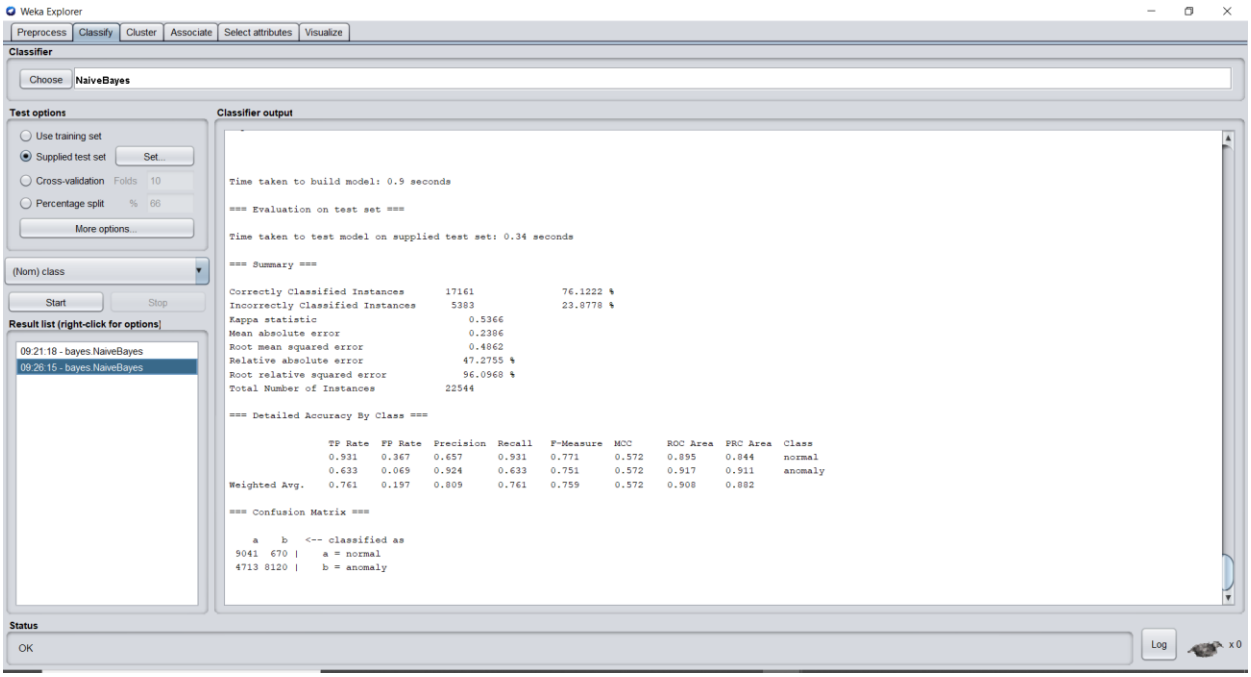
Successfully selected the "Naïve Bayes" classification algorithm from the "Classify" tab.

Check the results with a 10-fold cross validation.



Successfully made the cross-validation with 10 folds and got the above results with correctly classified instances – 90.3813% and incorrectly classified instances – 9.6187%. Time taken to build the model is 0.87 seconds.

Now, upload the test dataset and check the classification results.



Successfully uploaded the test data and got the above results with correctly classified instances – 76.1222% and incorrectly classified instances – 23.8778%. Time taken to build the model is 0.9 seconds.

Compare the results between 10-fold cross validation and the one obtained using the test dataset. Use confusion matrix to explain the results.

```
=== Confusion Matrix ===

     a     b   <-- classified as
 63058  4285 |     a = normal
  7832 50798 |     b = anomaly
```

Confusion matrix for 10 folds

```
=== Confusion Matrix ===

   a    b   <-- classified as
9041  670 |    a = normal
4713 8120 |    b = anomaly
```

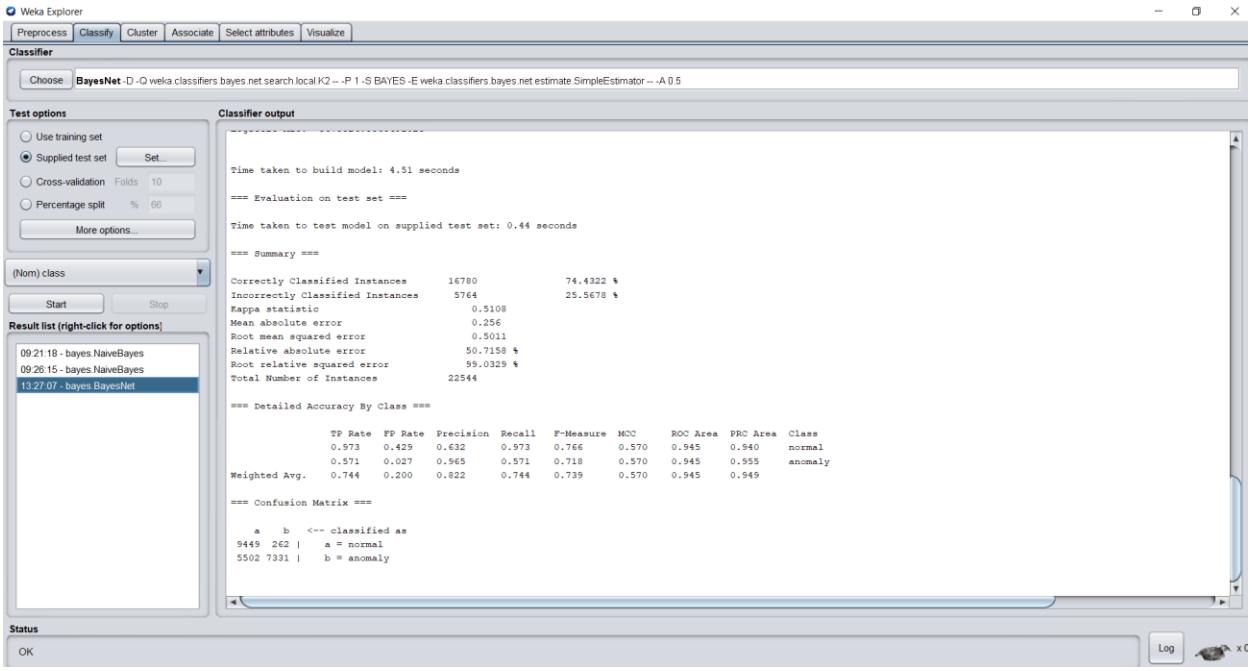
Confusion matrix for test data

On comparing both confusion matrix the matrix resulted using 10 folds has a high true positive value and which rated the correctly classified instances – 90.3813%. On comparing the false positive values, the matrix using 10 folds resulted the value of 4285 whereas the test data resulted 670. And the values of false negative are 7832 for 10 folds and 4713 for test data. The values of true negative are 50798 for 10 folds and 8120 for test data.

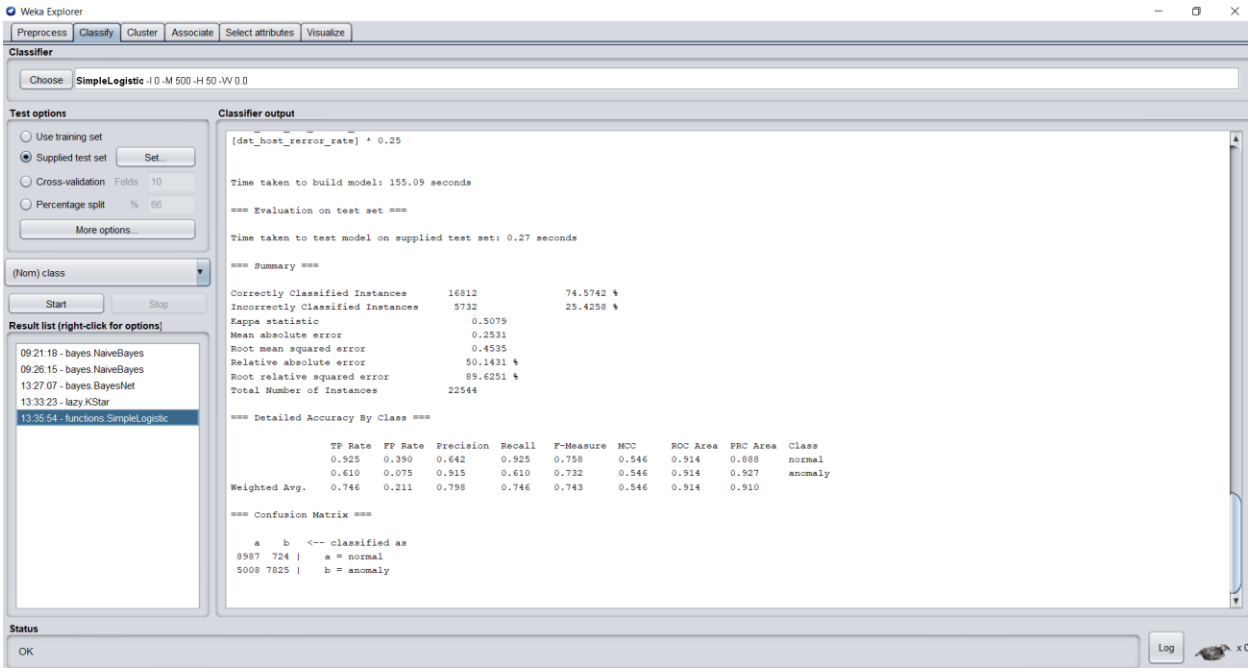
Finally, the output result obtained from the “Naïve Bayes” cross validation test option using 10 folds resulted in the high accuracy whereas the resulted output using test data is bit low accuracy on comparing to the cross validation.

Similar to the “Naïve Bayes”, apply at least 5 other supervised classification techniques and compare their performance. To report the performance create a table and present the following measures. Then compare the outcome of your nominated 5 algorithms. You can choose any 5. However try to consider high performing algorithms.

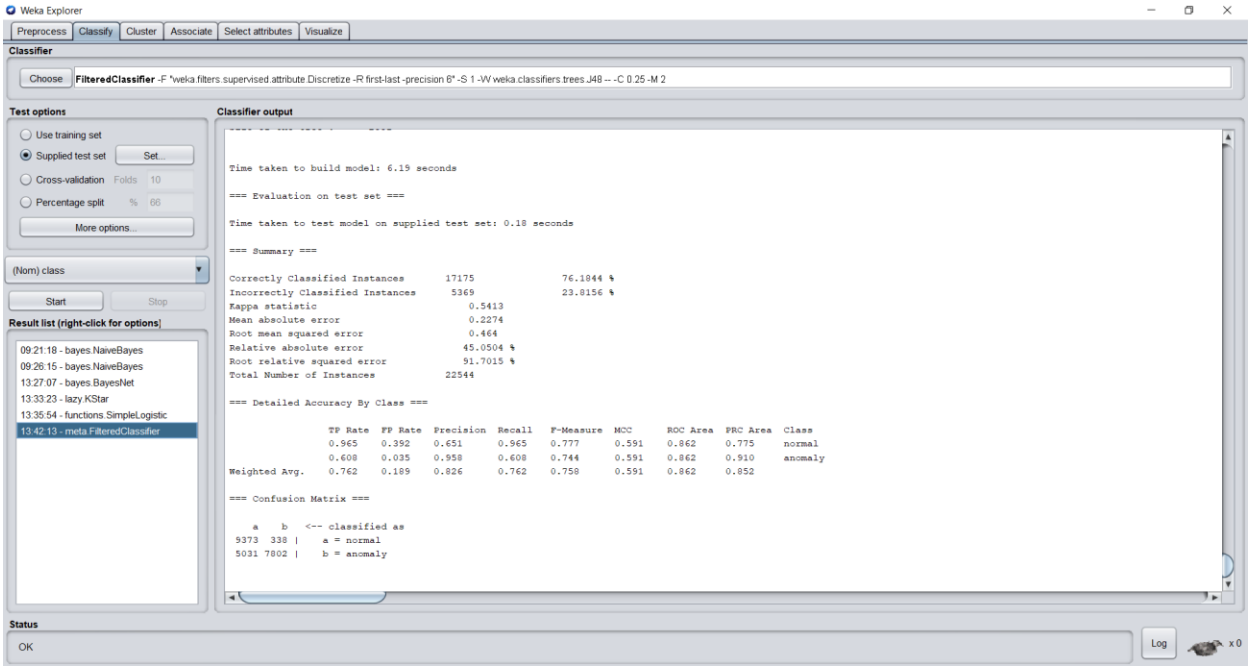
Algorithms	TP Rate	FP Rate	Precision	Re-call	F-Measure	ROC Area
BayesNet	0.744	0.200	0.822	0.744	0.739	0.945
SimpleLogistic	0.746	0.211	0.798	0.746	0.743	0.914
FilteredClassifier	0.762	0.189	0.826	0.762	0.758	0.862
DecisionTable	0.726	0.214	0.814	0.726	0.718	0.949
J48	0.815	0.146	0.858	0.815	0.815	0.840



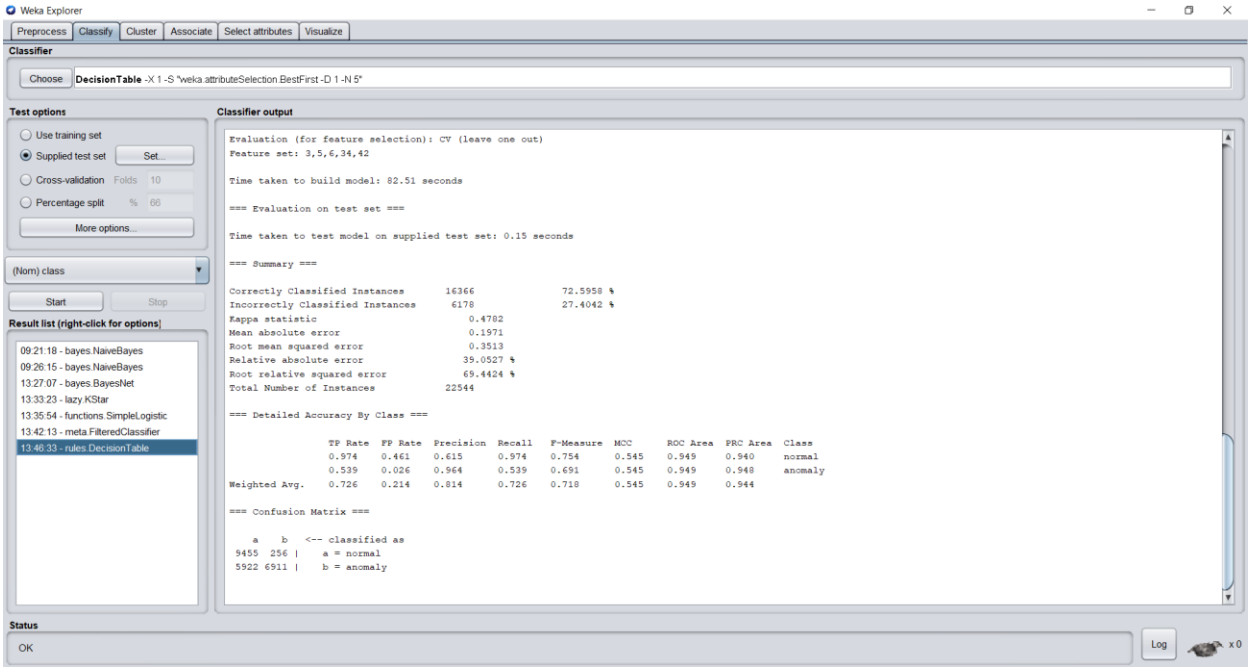
BayesNet Supervised Classification Technique



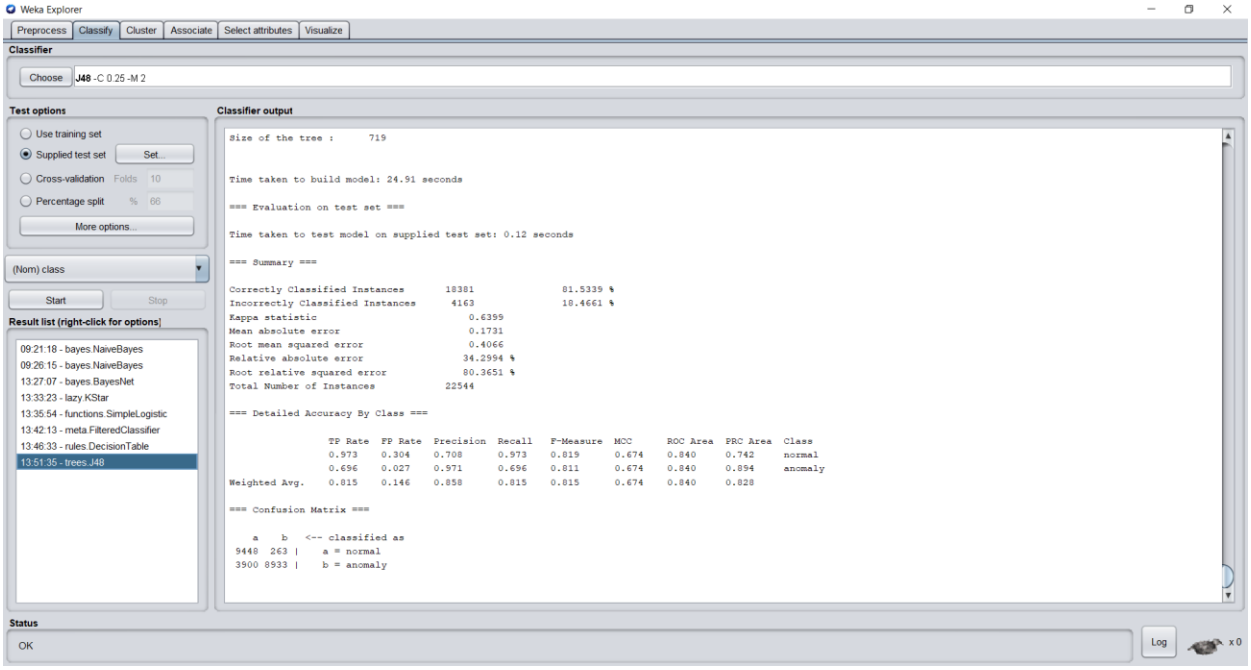
SimpleLogistic Supervised Classification Technique



FilteredClassifier Supervised Classification Technique



DecisionTable Supervised Classification Technique

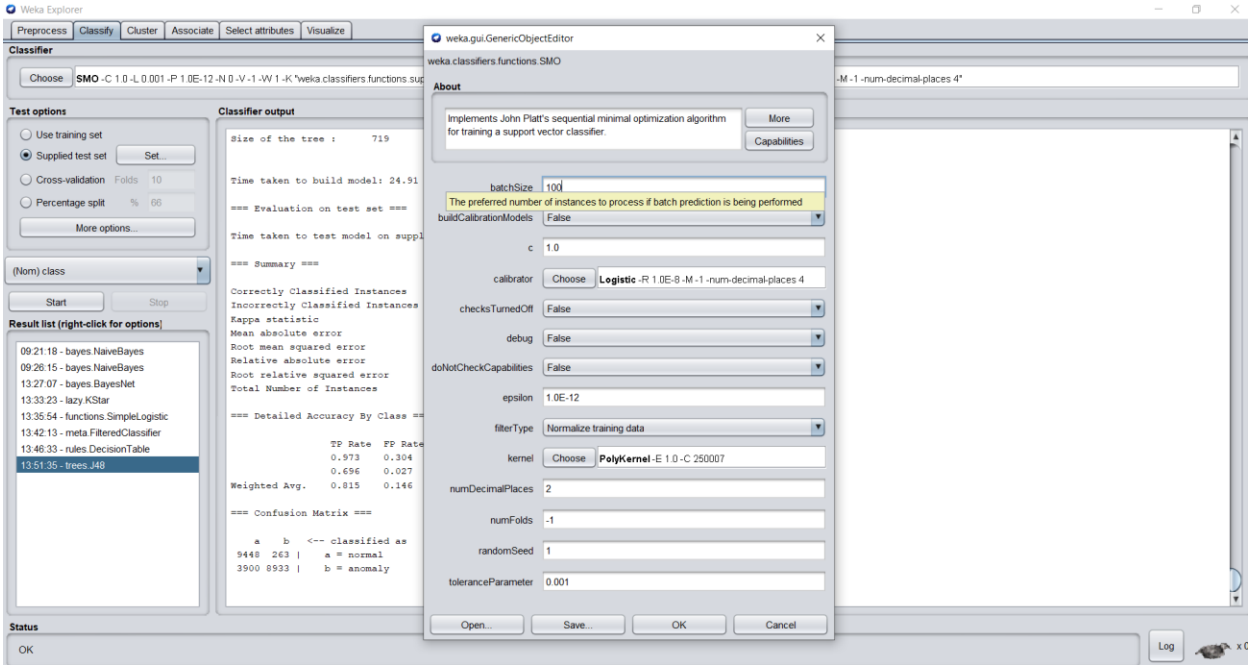


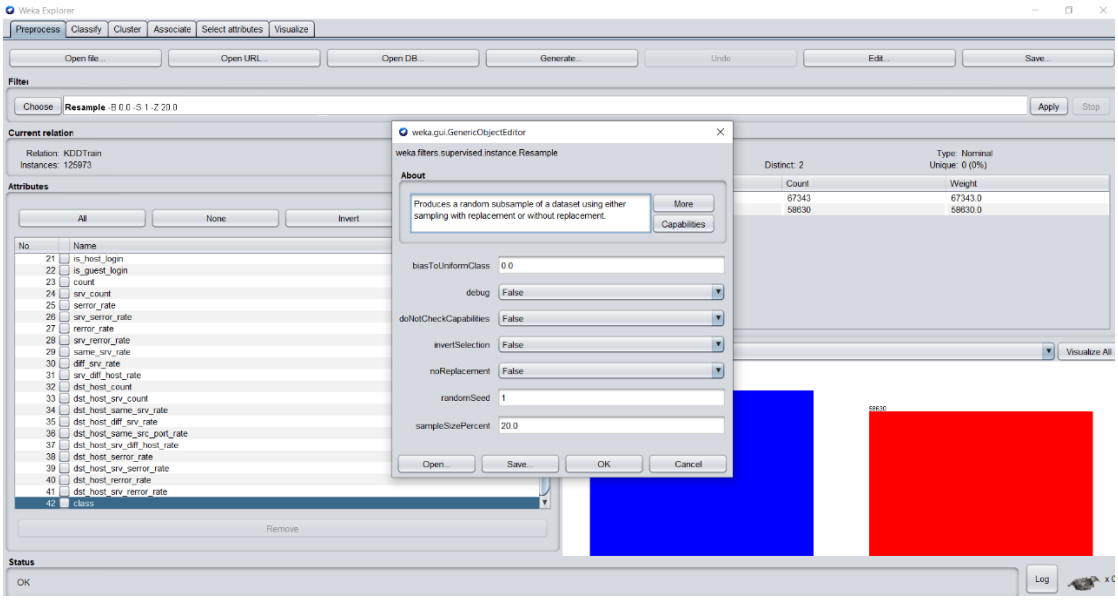
J48 Supervised Classification Technique

I have chosen Bayes Net, Simple Logistic, Filtered Classifier, Decision Table, and J48 supervised classification technique based on the resulted outputs Bayes Net has the lowest build time of 4.15 seconds. Correctly classified instances is more on J48 technique which is

82.5339% which took a build time of 24.91 seconds. On comparing with all the four techniques, J48 is resulted as a high performance because of its highest classified instances.

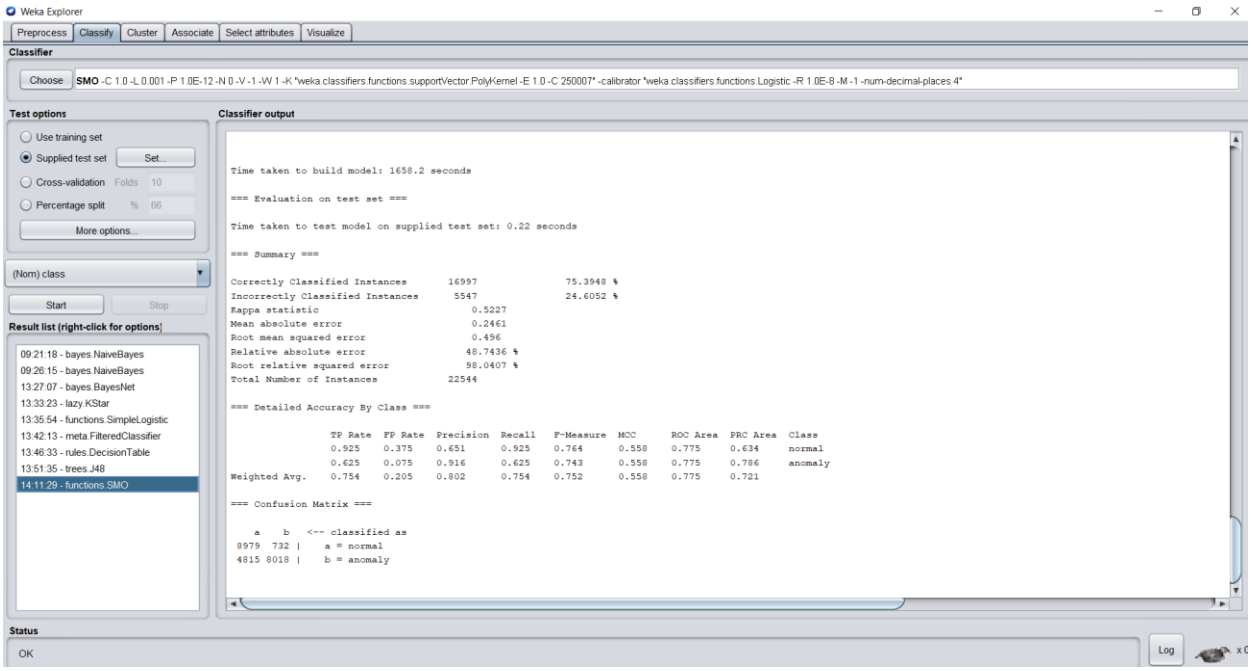
Some algorithms may have tuning parameter. Consider the SMO based SVM algorithm. You can try different kernel trick as shown below. Change the kernels to “PolyKernel” and ensure that the filter normalizes the training data as shown in the figure. If you start the task, it will take too much time on this large dataset. So you need to reduce the sample size of the dataset to make it manageable (note: it may impact on the performance).





Now perform classification task based on SVM classifier (SMO) using POLY and RBF kernels and report the confusion matrices and computation time.

The total computation time is **1658.2 seconds (build time), 0.22 seconds (test time).** Using POLY Kernel.

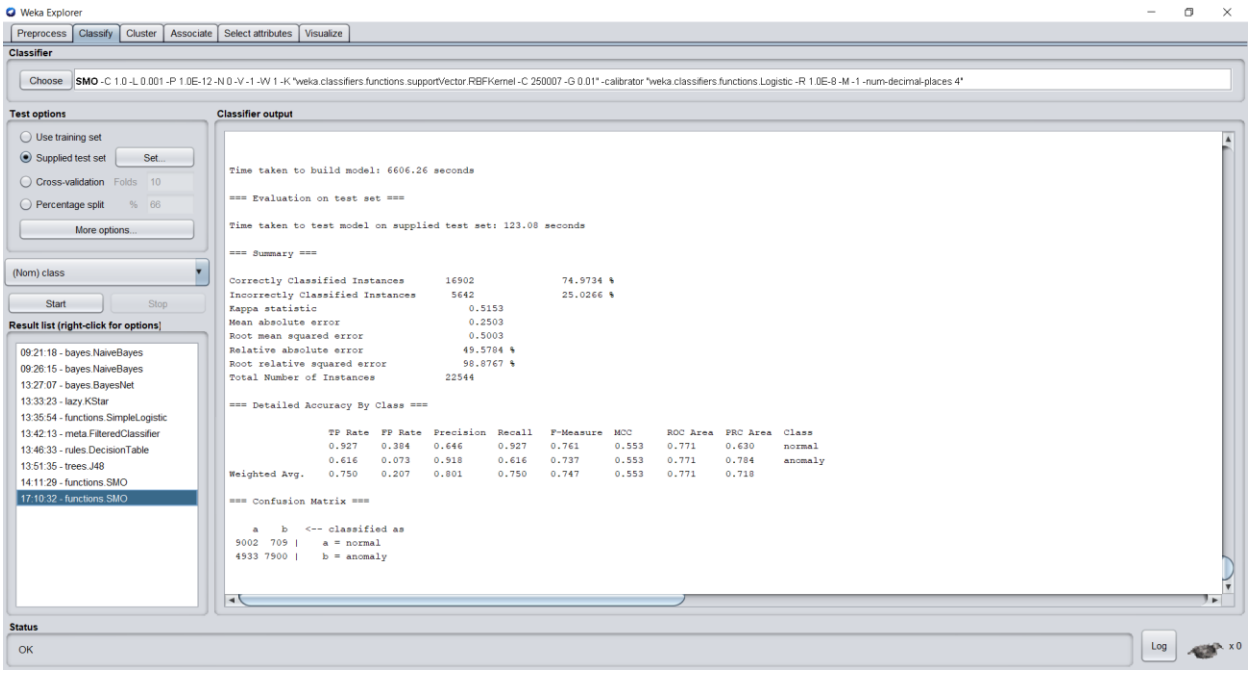


Confusion matrix

```
=== Confusion Matrix ===

    a     b   <-- classified as
 8979   732 |     a = normal
 4815  8018 |     b = anomaly
```

The total computation time is **6606.26 seconds (build time), 123.08 seconds (test time).** Using RBF Kernel.



Confusion matrix

```
=== Confusion Matrix ===

    a     b   <-- classified as
 9002   709 |     a = normal
 4933  7900 |     b = anomaly
```

On analyzing all the supervised classification techniques each one varies in some properties. Some vary in the compilation time, some variation in the correctly classified instances. But there is no statement like the technique which takes more compilation time results in more accuracy. E.g.: J48 took only **24.91 seconds** of build time and resulted in **81.5339%** of correctly classified instances whereas the SMO technique took **6606.26 seconds** and resulted in **74.9734%** of correctly classified instances. But overall, we have options for controlling the algorithm properties we can set the percentage of the dataset to be taken for testing.

Reducing the size of the dataset for testing results in faster outputs than testing and training the complete dataset. This can be made under preprocessing tab -> sample size percent. In my case, I reduced the dataset count to **20%** from **100%** which resulted in faster outputs. I tested

using two options one is the cross-fold validation and the other is using the test dataset. The cross-fold validation is the one if we assume the K value is 2 then the data has been divided into two parts one for the training phase and the other one for the testing phase. After one process has been completed automatically the train and test dataset interchanges as test and train this is the process of cross-validation.

The other one is setting the test dataset which is dataset used for testing purpose instead of using cross validation. In general, the result is a confusion matrix which has the values of TP, FP, TN, FN. From these values' precision, recall is calculated. Precision (P) = TP/(TP+FP). Recall(R) = TP/(TP+FN). Other than these values we also obtain F-Measure and ROC Area. F-Measure is calculated by (2*P*R) / (P+R). These data are used to find the performance of each supervised classification technique.

The values in the resulted confusion matrix are TN on the first cell, FP in the second cell, FN is in the third cell (below TN), TP is in the fourth cell (below FP). As a result of my calculations Bayes Net has the fastest compilation time of **4.28 seconds** and **J48** has the highest correctly classified instances - **82.5339%.** The highest compilation is for the SOM (SVM) classification technique which took **6606.26 seconds** in RBF Kernel and **1658.2 seconds** in the Poly Kernel.

But on comparing the results of both Poly kernel – **75.3948%** and RBF Kernel – **74.9734%** there is no huge difference in the correctly classified instances. But the compilation time between these two kernels varies a lot with a difference of **4948.06 seconds**. So, the Poly Kernel results high in the correctly classified instances and low compilation time. On the whole the J48 supervised classification technique is the best classified one because it resulted high correctly classified instances in low time with the complete (100% dataset).