DMAT – Assignment 2

Name : NITHIN UMAMAHESH Student Number: ­2996085

|  |  |
| --- | --- |
| **Course** | MSCBD-DMAT |
| **Stage / Year** | 1 |
| **Module** | Data Mining Algorithms & Techniques |
| **Semester** | 2 |
| **Assignment** | Assignment 2 |
| **Date of Title Issue** | 18thNov |
| **Assignment Deadline** | 13th Dec |
| **Assignment Submission** | Upload to Moodle |
| **Assignment Weighting** | 30% of module |

# Objective

1. To successfully apply a set of data mining skills imparted in this module to a previously unseen datasets to achieve knowledge discovery.
2. Evaluate a well-regarded peer reviewed paper or journal article which concerns the application of one of the techniques covered in this module and comment on its relevance to your dataset.

# Deliverables

A single zip called firstName1\_lastName1\_studentNumber1\_firstName2\_lastName2\_studentNumber2\_assignment2.zip to be uploaded to Moodle containing the following files:

* This file edited to contain the results of your investigation. Each of the **NUMBERED HEADINGS IN RED** should be expanded to satisfy the requirements of the section.
* A set of supporting files including but not limited to the following, which should be clearly referenced from your documentation. You only need to submit the files relevant the techniques you have explored.
  + The original dataset file
  + dataset.arff
  + trainigSet.arff
  + testingSet.arff
  + j48tree.arff
  + associationrules.arff
  + kmeans.arff
  + dbscan.arff
  + mlp.arff
  + The research paper.

# Choosing Your Dataset

1. Your dataset should concern a real-world problem that lends itself to easy understanding by your classmates.
2. It should not be identical to the dataset you used in assignment1.
3. It should have >1000 tuples/rows/instances.
4. It should have >=10 attributes
5. It should have attributes which can serve as labels so that the accuracy of your data analysis can be determined.
6. If you cannot find one dataset which is suitable for use with all techniques then you may choose 2. Please clearly indicate which dataset was used in which case.

The list below should help you on your search, student please share additional sources on Moodle discussion form.

* [**UCI Machine Learning Repository**](http://archive.ics.uci.edu/ml/)- A repository of more than 200 data sets for machine learning and data mining
* [**Movie Ratings Data**](http://facweb.cs.depaul.edu/mobasher/classes/ect584/data/movielens.zip) - Real movie ratings data from [**www.movielens.org**](http://www.movielens.org/) Web site. Contains ratings on 1600+ movies by 1000 users
* [**Kaggle.com Competition Data Sets**](http://www.kaggle.com/competitions) - Data sets from a variety of competitions. Also a good source for assignment ideas.
* [**Stanford Large Network Dataset Collection**](http://snap.stanford.edu/data/)- A variety of network data sets, including data from social networks, product reviews, online communities, etc.
* [**Yelp Data Set Challenge**](http://www.yelp.com/dataset_challenge/) - Reviews and check-in data on thousands of businesses.
* [**Million Song Dataset**](http://labrosa.ee.columbia.edu/millionsong/) - Freely-available collection of audio features and metadata for a million contemporary popular music tracks.
* [**Public Data sets on Amazon Web Services**](http://aws.amazon.com/publicdatasets/) - Large public data sets (including data sets for US Census, Wikipedia, Freebase, human genome project), ready for big data analytics on the cloud.
* [**Data.gov**](http://catalog.data.gov/dataset)- Publically available data sets from Federal, State, and local government, including economic, geological, demographic and many other types of data sources. This site also includes a list of other [**Open Data Sites**](http://www.data.gov/opendatasites) with similar publicly available data sources from various cities, states, and countries.
* [**KDnugget's list of data sets for data mining**](http://www.kdnuggets.com/datasets/)
* [**Infochimps Data Market**](http://www.infochimps.com/datasets) - Thousands of data sets, including data from various social networks and collaborative tagging sites such as Twitter, Delicious, Last.fm, MusicBrainz, as well as data sets from many other domains.

#### Initial Tasks

## 1. Description of your dataset and findings – 20%

* **Title**: **Bank Marketing** **by Portuguese banking institution**
* **Data description:** A description of the data in detail under the following subheadings:
  + The problem domain

**Marketing**

* + The source of the data

**This dataset is taken from UCI machine learning repository.**

<https://archive.ics.uci.edu/ml/datasets/bank+marketing>

* + The agencies working with the data

**Portuguese banking institution.**

* + The intended use of the data

**The classification goal is to predict if the client will subscribe a term deposit.**

* + The attribute types of the data

**This dataset contains 17 attributes (including the target variable) and 45211 instances.**

|  |  |  |
| --- | --- | --- |
| Attribute | Attribute Type | Description |
| Age | Numeric | Age of the customer. |
| Job | Categorical | Type of the job. |
| Marital | Categorical | Marital status of the customer. |
| education | Categorical | Educatio level of the customer. |
| default | Categorical | Shows if the customer has credit by default. |
| balance | Numeric | Balance of the customer. |
| housing | Categorical | Tells if the customer has a housing loan. |
| loan | Categorical | Tells if the customer has a personal loan. |
| contact | Categorical | Communication type. |
| month | Categorical | Last contact month of the year. |
| day\_of\_week | Categorical | Last contact day of the year. |
| duration | Numeric | Duration of the last call. Zero if not contacted. |
| campaign | Numeric | Number of contacts performed to this client during campaign. |
| pdays | Numeric | number of days that passed by after the client was last contacted from a previous campaign |
| previous | Numeric | number of contacts performed before this campaign and for this client |
| poutcome | Categorical | outcome of the previous marketing campaign |
| y | Target variable | has the client subscribed a term deposit? |

Please include screen shots (with one or two sentences of summary) of the dataset and also of the data summaries that are available though Weka.

* **Objective**: Your objective. You can update this as you progress through your assignment revising it and making it more specific.

**The objective is to design a model to predict if the customer will subscribe a term deposit based on certain features such as job, marital status, education, balance etc.**

* **Summary of Findings**: This should be written following the application of your data mining techniques.

**After performing the J48 experiments on our bank dataset, we see that decision tree has been successful in classifying instances where the target class is “no”. However, we see some variations in classifying class “yes” variables. The below table shows the variation in results for class “yes”**

|  |  |  |  |
| --- | --- | --- | --- |
| **Experiment** | **Parameter setting** | **TP Rate** | **Correctly classified instances(class “yes”)** |
| 1 | No pruning, min obj=30 | 0.225 | 115/512 |
| 2 | pruning, confidence factor= 0.15 and min obj=20 | 0.135 | 69/512 |
| 3 | pruning, confidence factor= 0.6 and min obj=10 | 0.229 | 117/512 |
| 4 | Reduced error pruning, min obj=30 and num folds =3 | 0.221 | 113/512 |

**From the above we can see that our model predicts class “yes” instances when there is very less pruning or no pruning at all. In our experiment 1 where no pruning was performed 115 instances were correctly classified. However, in our second experiment were more pruning (low confidence factor means more pruning) we see only 69 instances were correctly classified. Again in our third experiment less pruning was performed (high confidence factor) and we see that 117 instances were classified correctly. This could also be due to the less number of objects required at each node. In our last experiment we used reduced error pruning instead of C48 and we see 113 instances were classified correctly. After looking at the above table we can say that our model performs best in predicting class “yes” instances when there is very less pruning. This could be because of the less number of instances available for class “yes” which does not support the model in training for class “yes” instances. One way to solve this could be to add synthetic instances for class “yes” and then perform the training and testing again. We have not done this as Weka does not support SMOTE for balancing the dataset.**

## 2. Preprocessing – 10%

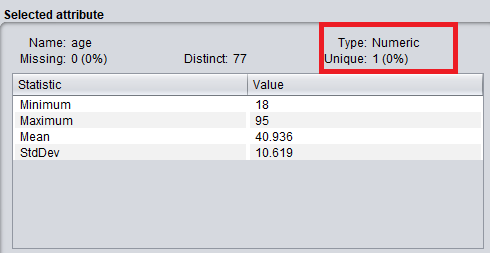
In this section you should

1. Identify the set of preprocessing techniques that can be applied to your data and clearly indicate which techniques are appropriate and which ones are not.
2. Provide evidence through screenshot of the effects of preprocessing the data along with a short explanation.
3. Generate a file called dataset.arff which is the outcome of the preprocessing.

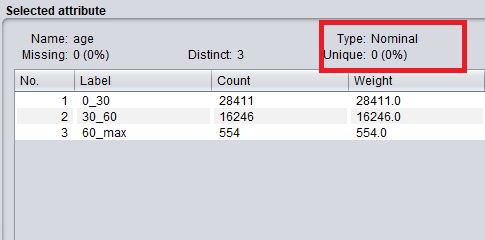
**Preprocessing 2.1: Descretizing age**

**We descretize age into three groups using the descretize filter of Weka.**

**Classification algorithms work well when the attributes are nominal. Hence, we descritize age as shown in the below figures 2.1(a) and 2.1(b). In the below figure 2.1(b), age has been descretized and manually changed in the .arff file to integer values.**



**Figure 2.1(a) – Before descretization**

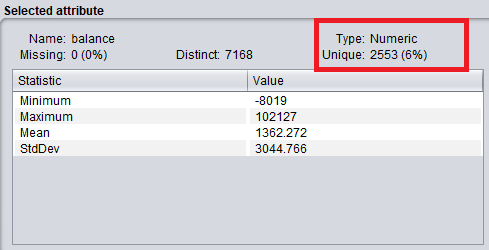


**Figure 2.1(b) – After descretization**

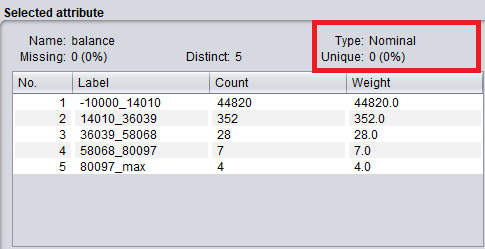
**Preprocessing 2.2: Descretizing balance**

**We descretize balance into five groups using the descretize filter of Weka.**

**Classification algorithms work well when the attributes are nominal. Hence, we descritize balance as shown in the below figures 2.2(a) and 2.2(b). In the below figure 2.2(b), age has been descretized and manually changed in the .arff file to integer values.**



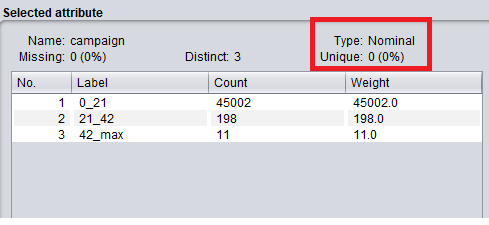
**Figure 2.2(a) – Before descretizing balance.**



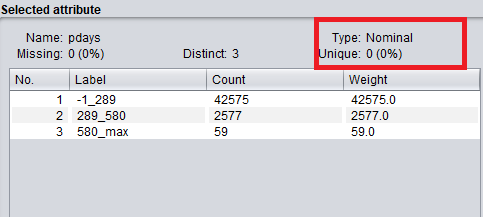
**Figure 2.2(b) – After descretizing balance.**

**Preprocessing 2.3: Descretizing campaign, pdays and previous**

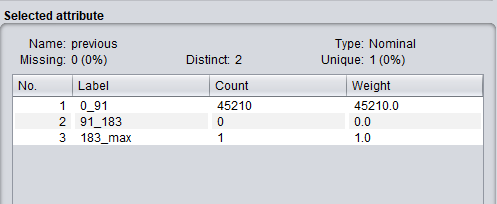
**Similar to the above preprocessing steps we also descretize campaign, pdays and previous into three intervals as shown in the below figures 2.3(a), 2.3(b) and 2.3(c).**

****

**Figure 2.3(a) – After descretizing campaign**

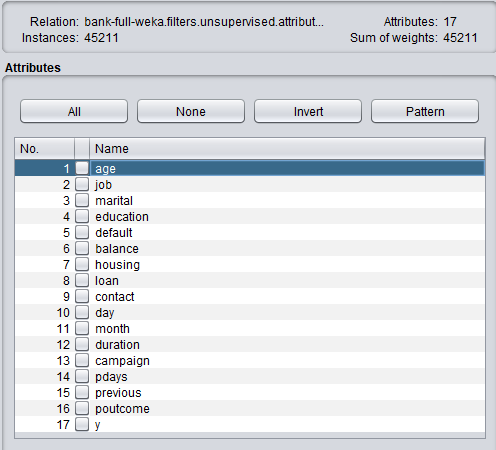
****

**Figure 2.3(b)- After descretizing pdays**

****

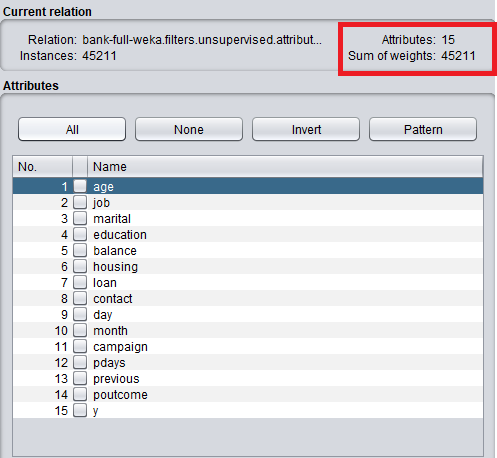
**Figure 2.3 c- After descretizing previous**

**Preprocessing 2.4: Removing attributes that does not add importance to our prediction.**

****

**Figure 2.4(a) – Before removing the attributes**

**From the above figure 2.4(a), we remove default and duration as they are not required for our classification as they do not add much importance in our prediction process. We mainly aim in predicting the outcome based on customer age, marital status, education, balance, etc.**

****

**Figure 2.4(b) – After removing the attributes**

**After performing the above preprocessing step we save our file as *bank.arff*.**

## 3. Divide your dataset into training and test set – 0%

Follow the instructions presented in the link below divide the test into a training and testing set in the ration of (9:1).

<https://www.youtube.com/watch?v=uiDFa7iY9yo>

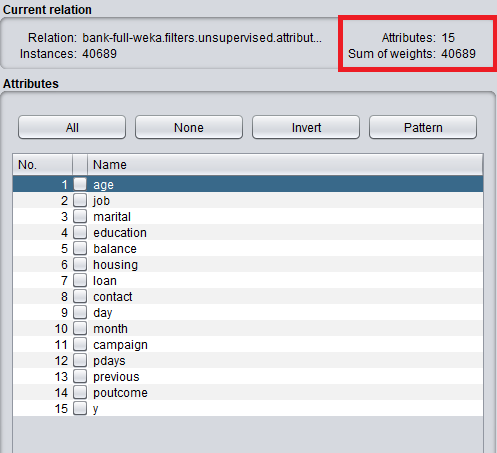
The files generated as part of this process should be saved and submitted as the following

* trainingSet.arff and
* testingSet.arff

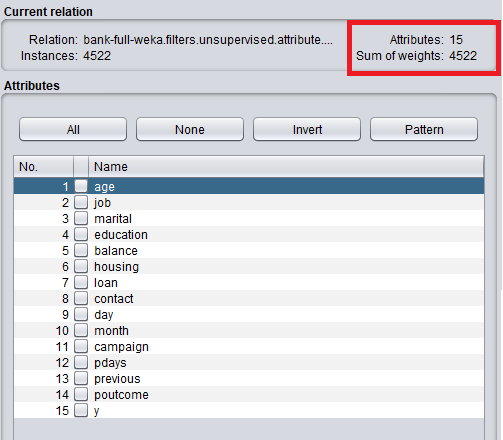
Screen shorts of these files should be included.



**After performing the necessary preprocessing and saving the changes into .arff file, we split the file into training set and testing set in the ratio 9:1. The training set and testing set files are created and saved as shown in the below figure 3(a) and 3(b).**

****

**Figure 3(a) – Training set**

****

**Figure 3(b) – Testing set**

#### Data Mining Techniques

# Classification / Association

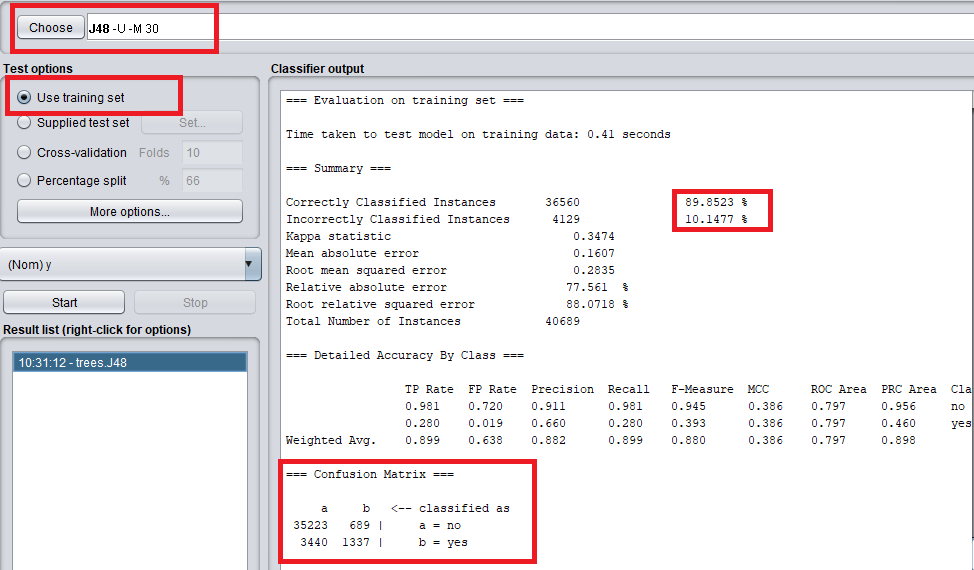
For each of the following classification techniques

1. Train your model using trainingSet.arff
2. Test your model using testingSet.arff
3. Write a few paragraphs analyzing the results. Be sure to vary parameters at least 3 times in each case. Support this analysis with screenshots of the following
   1. The model or a visualization of the model
   2. The results of the model
   3. Any additional output of the model including but not limited to
      1. Rules
      2. Confidence Values
      3. Confusion Matrixes
      4. etc
   4. Simple references to the notes or URL links to online resources complete with a sentence or two of explanation.

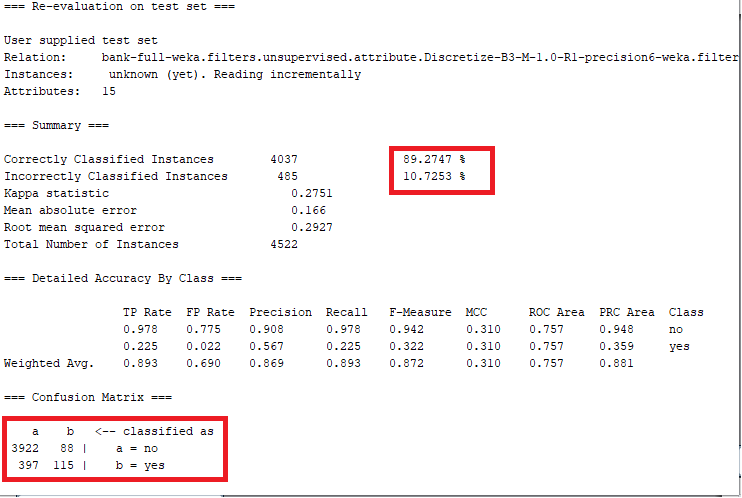
## 4. Classification/ Association: J48 Tree or Association Rules – 10%

**Experiment 1: Without pruning (i.e., unpruned=True) and 30 minimum objects**

**We train our model using trainingSet.arff and “use training set” test option. As seen in the below figure, evaluating on the training set (figure 1(a)) provide us an accuracy of approximately 90%. Also, the weighted average of the accuracy measures of the model are pretty reasonable. Since training and testing on the same dataset is not a good approach in machine learning, we next reevaluate the model providing the testingSet.arff to check the accuracy of the model.**



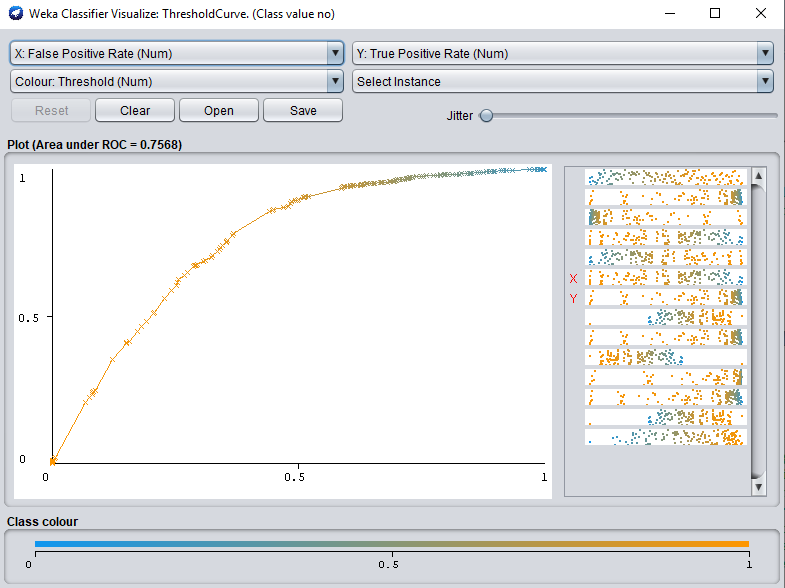
**Figure 1(a) – Evaluating on training set**



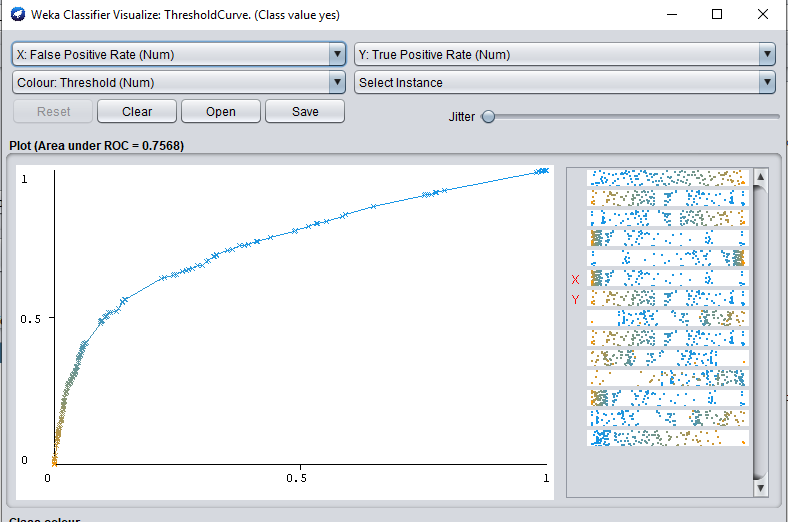
**Figure 1(b) – Evaluating on testing set**

**From the above figure 1(b), we can observe that our model is performing better on test set by correctly classifying 89% of the instances of the test set. Also from the accuracy measures we can see that the weighted average precision model is 0.89 which is reasonably good. True positive rate for classifying if the customer will not subscribe for term deposit is 0.978 which means our model has been successful in predicting class ‘no’ in most of the cases. However, true positive rate for prediction of class “Yes” is 0.225 which is not that great, but acceptable as our dataset contains less number of records for customer who subscribe for term deposit and this might have affected our prediction. Also, average f-measure is 0.872 which says that our model is accurate.**

**The below figures 1© and 1(d), ROC curves further conclude that our model performs reasonably better but not very good as the curves are very far from 1.**

****

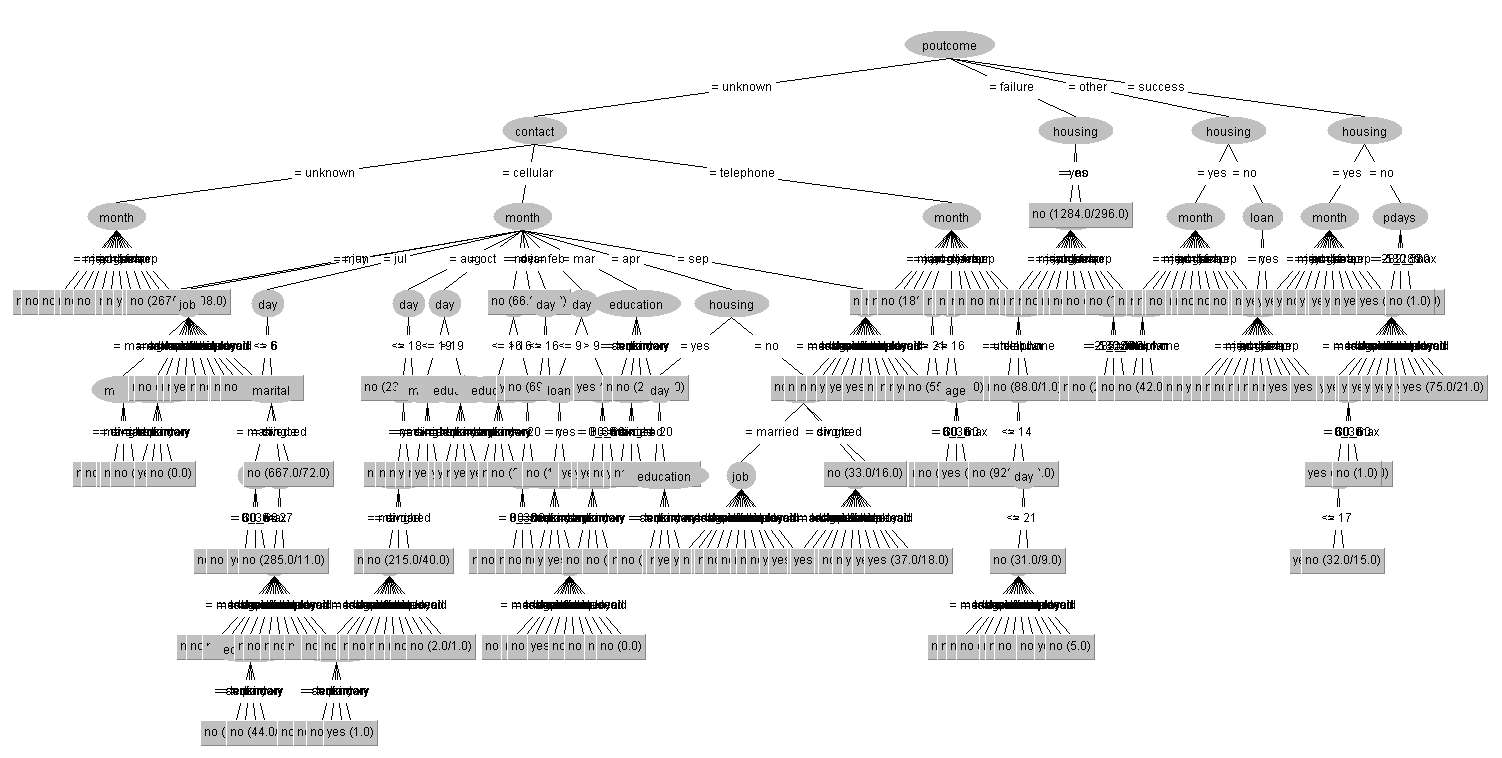
**Figure 1© - ROC curve for target class ‘no’.**

****

**Figure 1(d) – ROC curve for target class ‘yes’.**

**From the above graph we can see that our model for predicting class ‘yes’ is poor.**

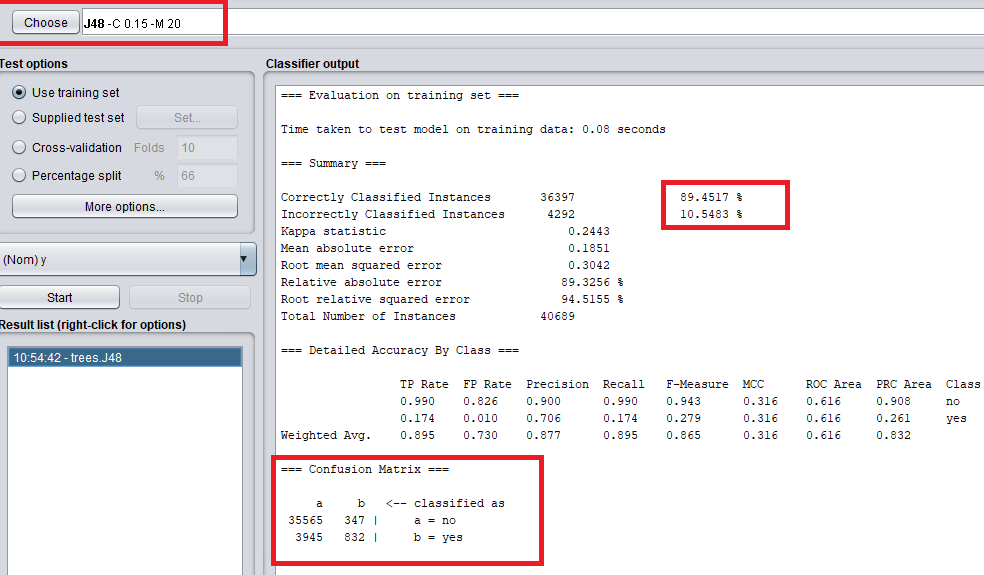
**Visualization of tree**

****

**From the above figure we see that our tree is massive without pruning and is dense even though the minimum number of required objects at each node is 30.**

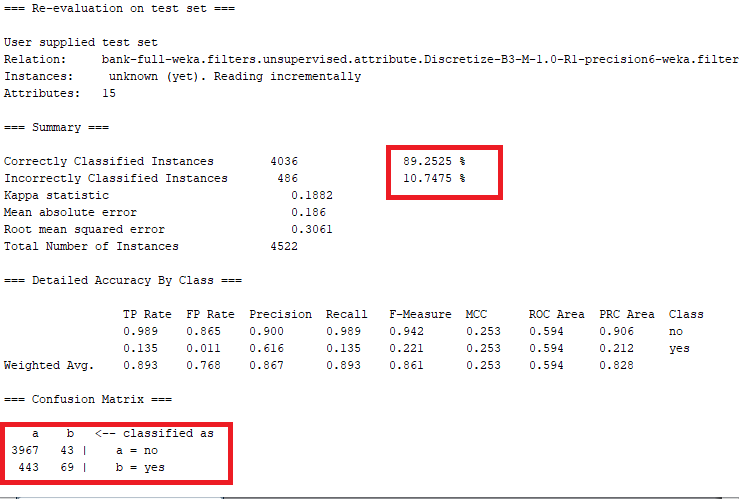
**Experiment 2: With pruning (i.e., unpruned=False) at confidence factor of 0.15 and 20 minimum objects.**

**We perform our second experiment by pruning the decision tree with the confidence factor of 0.15. Lesser confidence factor means more pruning. We also set a constraint of 20 minimum objects at each node.**



**Figure 2(a) – Evaluating using training set.**

**From the above figure we infer that this model performs similar to our previous model where pruning was not performed. However, testing on training data does not provide much insight hence we test this model on our testing data as shown below in figure 2(b).**



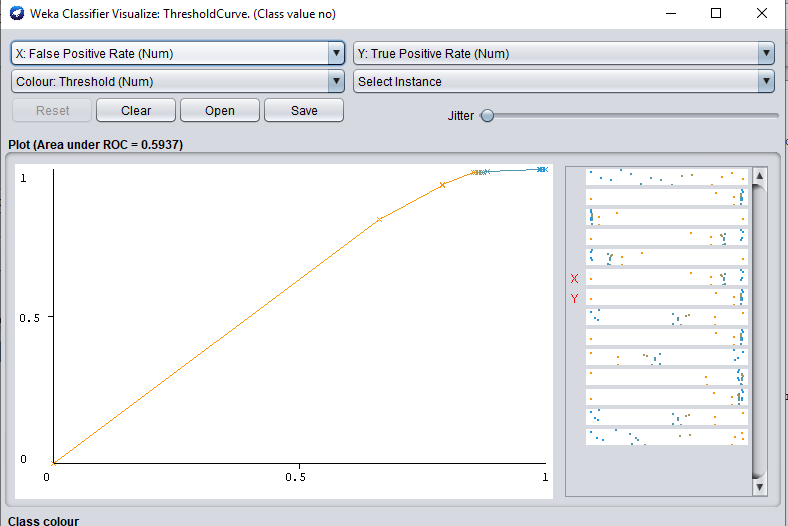
**Figure 2(b) – Evaluating using testing set.**

**From the above figure 2(b), we can see that testing our model on testing set provides same results as our evaluation on training set. Our model has an overall performance of approximately 89%. Also we can see that average true positive rate is good with 0.893. However, like our previous model, this model also performs poor in predicting whether the customer subscribes for term deposit. This can be inferred as the TP rate for class ‘yes’ is very poor with 0.135. On the other hand average precision is comparatively good in predicting both the classes. Average f-measure is 0.861 which shows our model is accurate. Further, from our confidence matrix, we can see that 443 instances has been classified wrongly to class ‘no’ where they should’ve been classified to class ‘yes’. This could be due to loss information with high pruning and min obj constraint.**

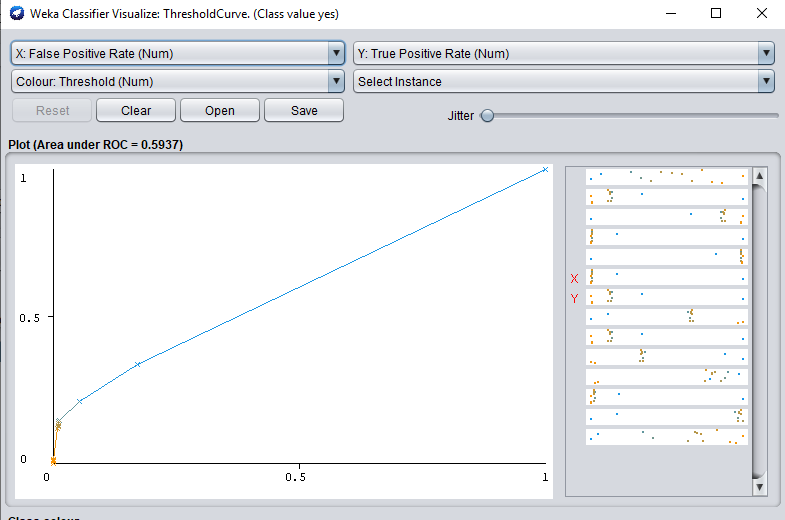
**The below ROC curves in figures 2© and 2(d) show how far the curves are in predicting class ‘no’ and ‘yes’ from 1.**

**We can see that the curves are very far from 1 which says that the model is not very accurate in predicting whether the customer will subscribe term deposit or not.**

**This variation in the accuracy could be due to the minimum object constraint and high pruning which could’ve removed some important factors that has affected our model.**

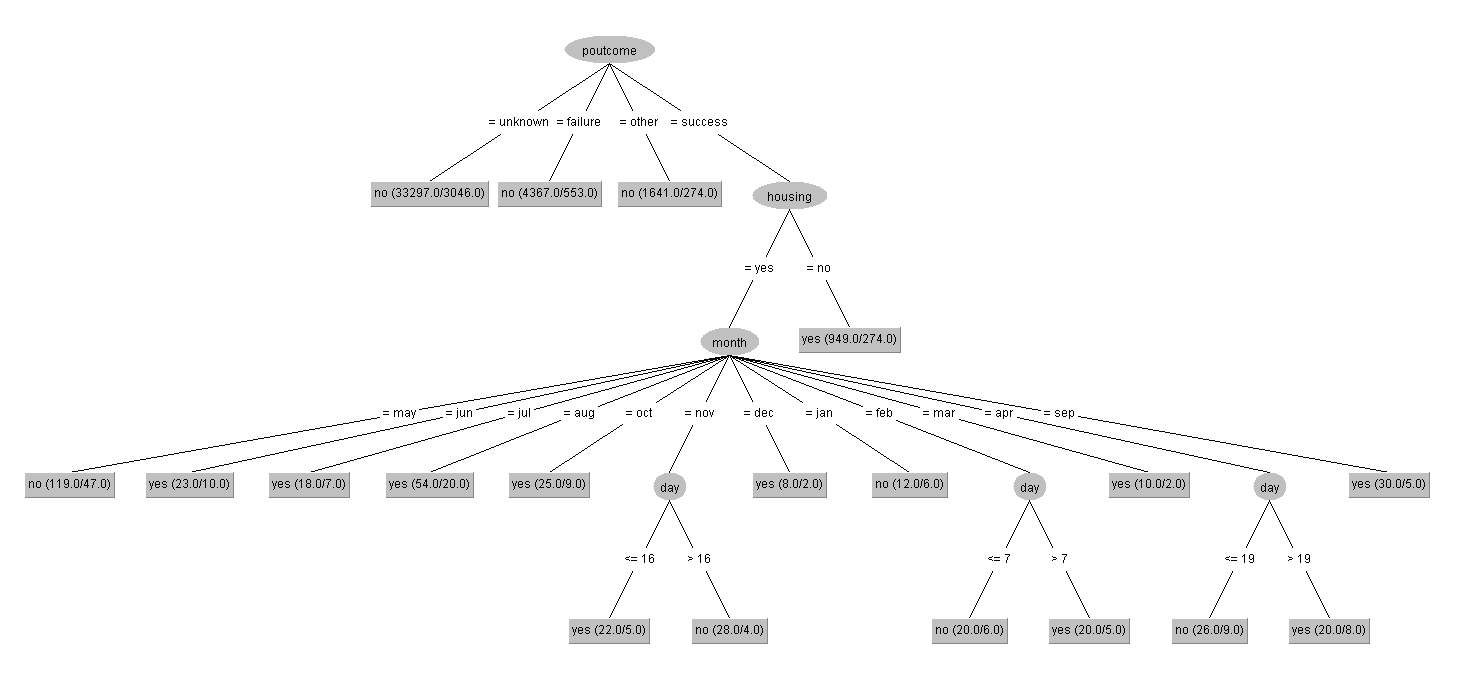


**Figure 2© - ROC curve for class ‘no’**



**Figure 2(d) – ROC curve for class ‘yes’.**

**Visualization of tree**

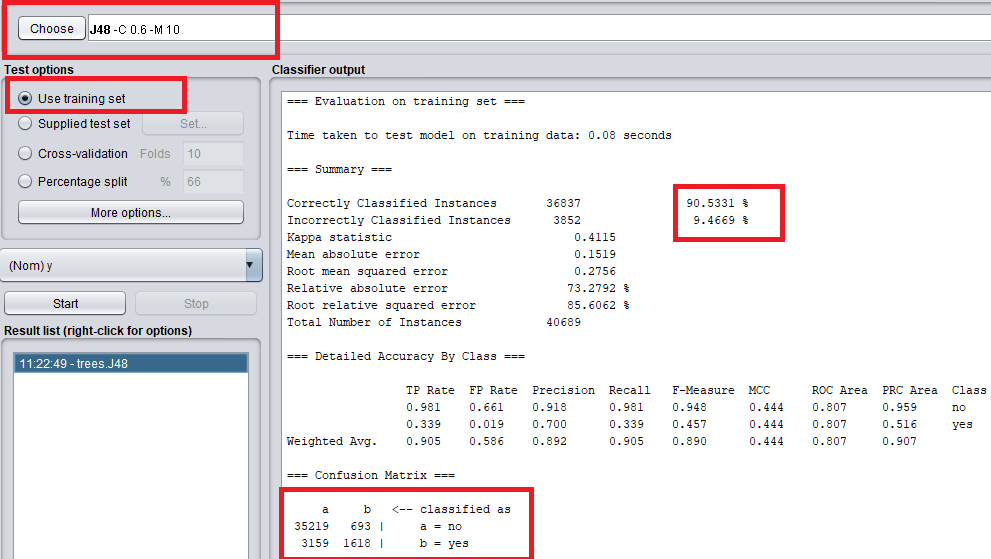
****

**The above provide a good understanding of how the decisions are made. However, this tree would’ve compromised some information which has affected the model in classifying our instances.**

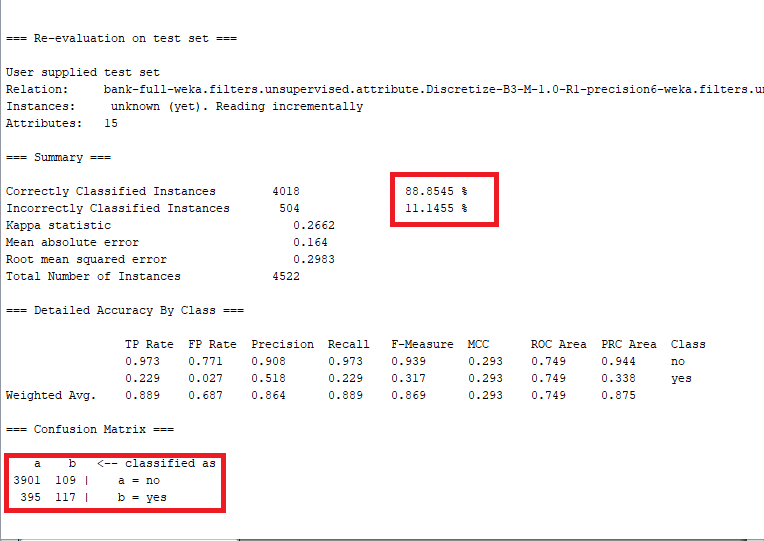
**Experiment 3: With pruning at confidence factor of 0.6 and 10 min objects.**

**In our last experiment we observed that our model does not perform well with more pruning and high minimum objects at each node. Hence, in this experiment we increase the confidence factor which reduces pruning. Also, we decrease the minimum objects required to 10.**

**Below figure 3(a) shows how our model performance with the above parameter setting. Like our previous two models, this model also performs good in predicting class ‘no’. However, we also see some improvements in predicting class ‘yes’ which we will discuss upon testing our model on testing set.**



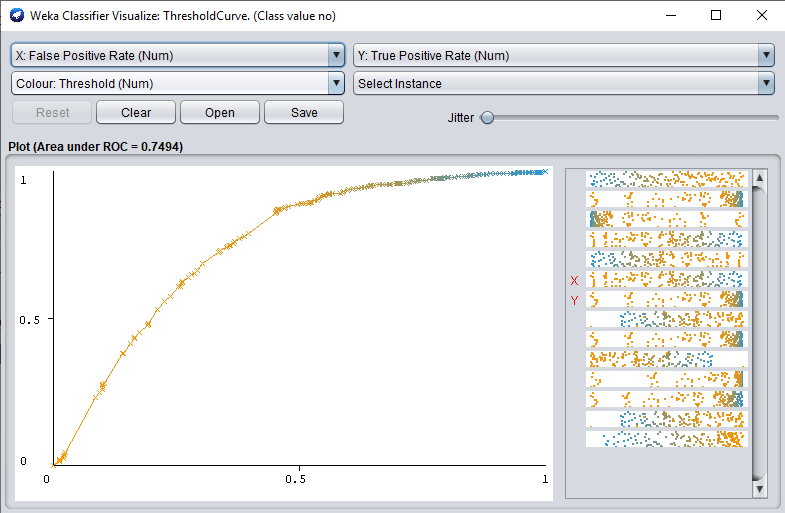
**Figure 3(a) – Evaluating on training set.**



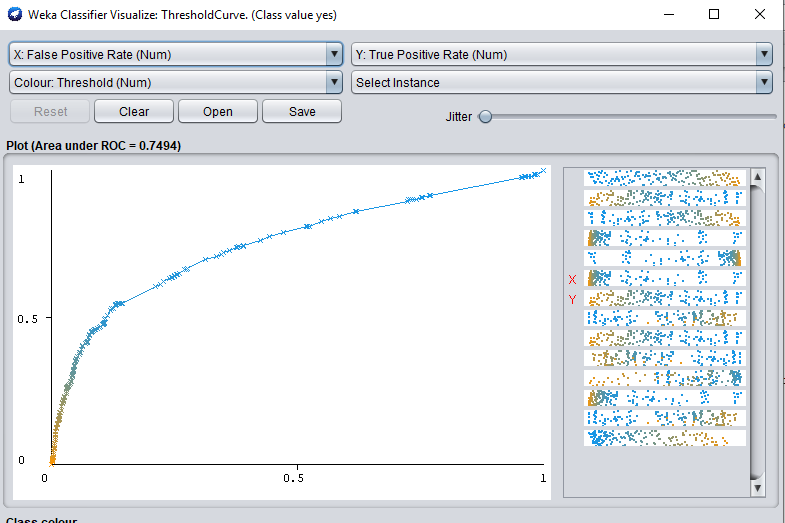
**Figure 3(b) – Evaluating on testing set.**

**From the above figure we can see that the overall performance of our model is good and acceptable at 88.85%. Also, we can see that accuracy measures such as TP rate, f-measure, ROC area are reasonably good and the model performs better on test set. The above model performs better in classifying class ‘yes’ instances compared to our previous models. But, this comes with some instances of class ‘no’ being predicted as class ‘yes’. From the confusion matrix we can see that 395 instances of class ‘yes’ has been wrongly classified as class ‘no which is comparatively better than our previous models. Hence we can say that for this dataset, our model performs better with less pruning and less number of minimum objects required at each node.**

**The below ROC curves figure 3(a) and 3(b) shows that our classification is reasonably good as the curves are close to 1.**



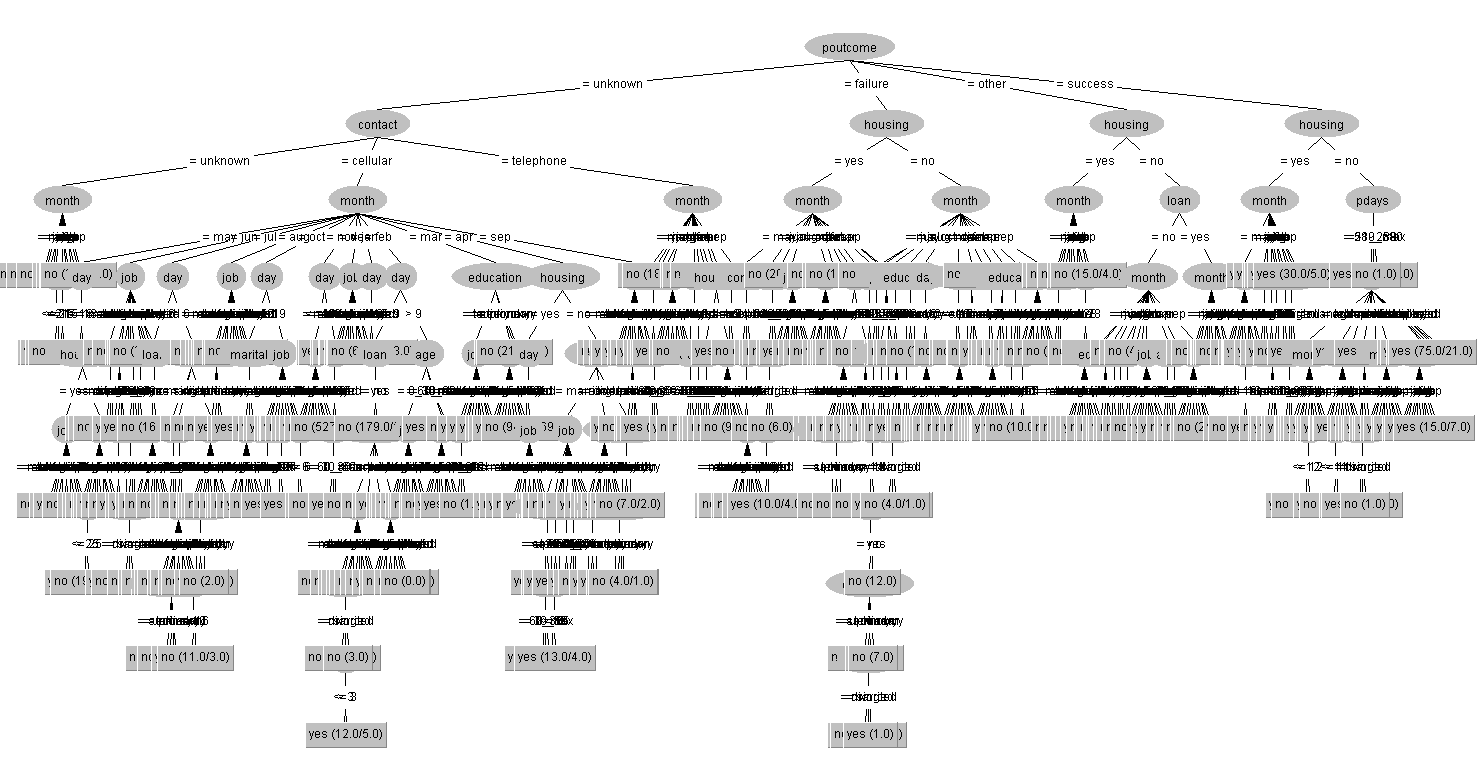
**Figure 3© - ROC curve for class ‘no’.**



**Figure 3(d) – ROC curve for class ‘yes’.**

**From the above figure we can see that our model is better than our previous model in classifying class ‘yes’.**

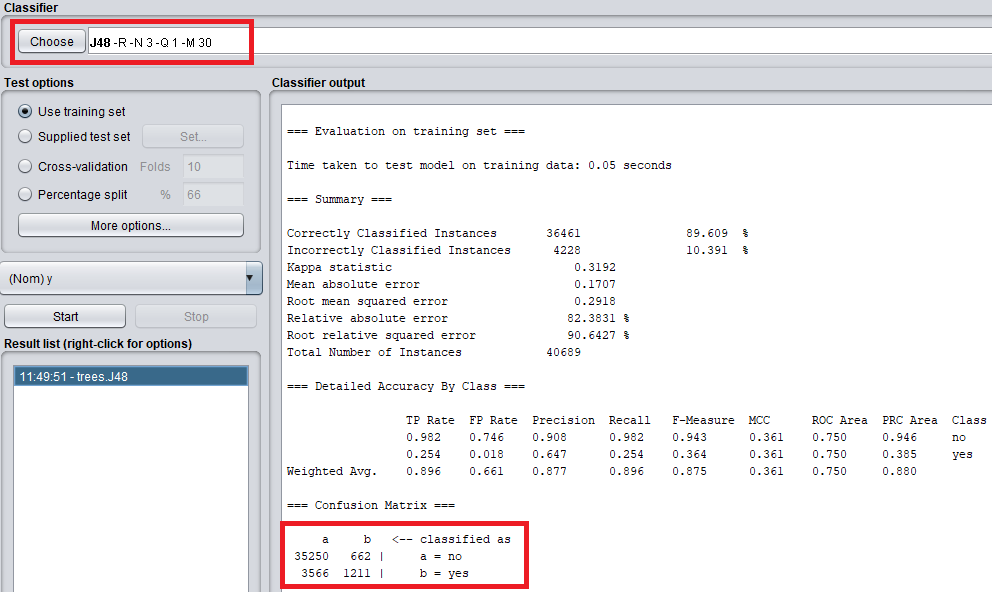
**Visualization of tree**

****

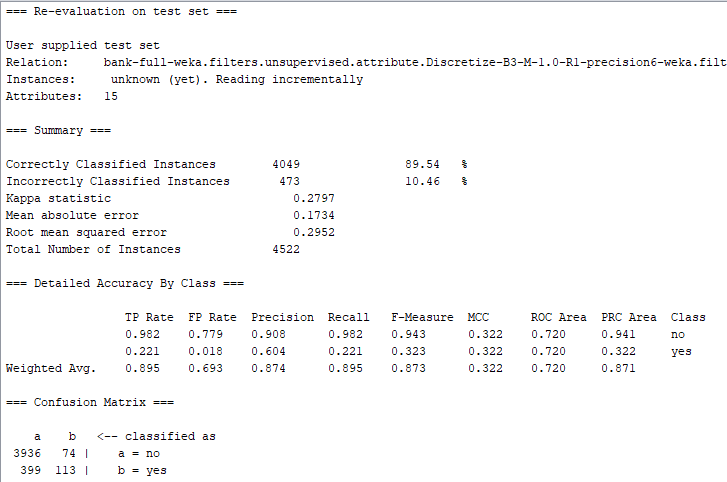
**From the above figure we can see that the tree is dense. This is because of the less pruning and less minimum number of objects at each node constraints.**

**Experiment 4: Using reduced error pruning , 30 min objects and 3 number of folds.**

**For our third experiment, we set the minimum number of instances at each node to 30 and we use reduced error pruning instead of C 4.5 pruning. In this pruning process, each node is considered for pruning by removing the subtree and making it the leaf node and assigning most common class at that node. A node is removed if the resulting tree does not perform worse the accuracy of the decision tree. Pruning is continued until further pruning is harmful. Below figure a shows the changes made.**

****

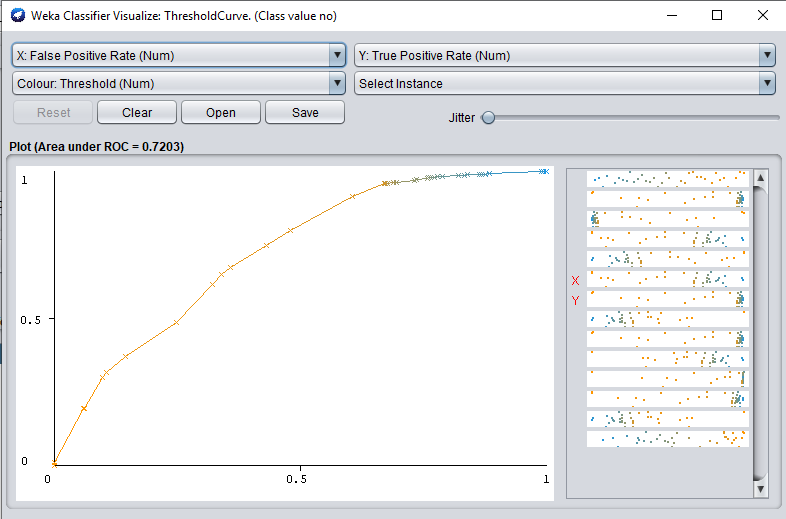
**Figure 4(a) – Evaluating on training set.**



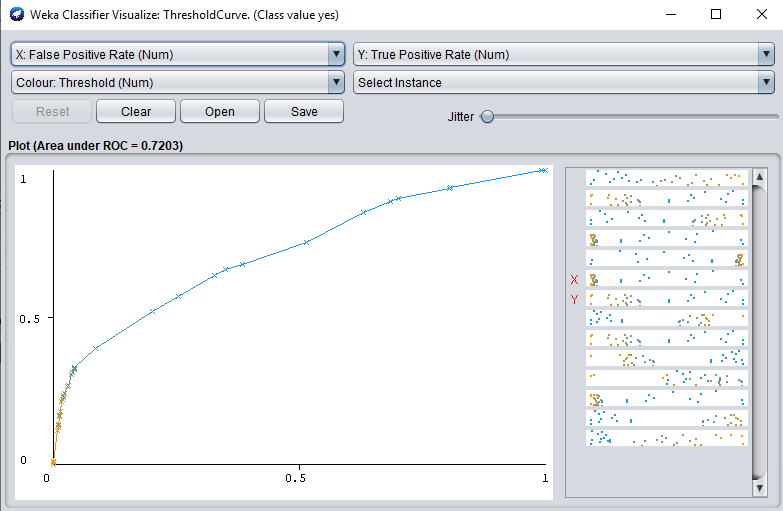
**Figure 4(b) – Evaluating on testing set.**

**From the above we figure we see that the model performs good at the rate of 89.54%. Also we can see that the average TP rate, precision, f-measure and Roc area are quite good. We can see that average true positive rate is 0.895. Also, f-measure is 0.873 which shows that our model is accurate and acceptable.**

**Below figures 4(c) and 4(d) shows the ROC curves.**

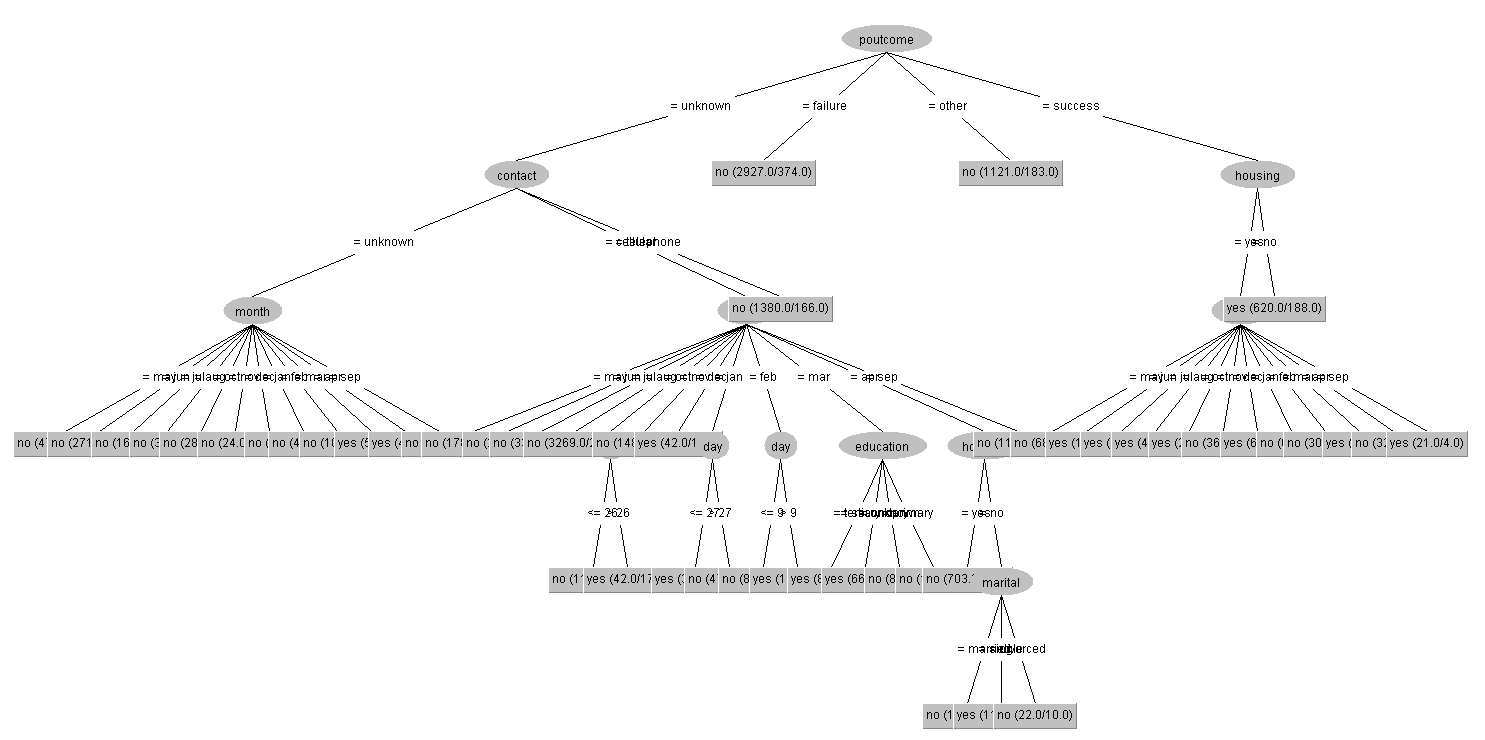


**Figure 4(c) – ROC curve for class ‘no’.**



**Figure 4(d) – ROC curve for class ‘yes.**

**Visualization of tree**



## 5. Classification: MLP or a similar advanced technique from Weka – 15%

If you are using a similarly advanced technique please clearly identify the technique and the steps you are taking. You may reference online tutorials and videos.

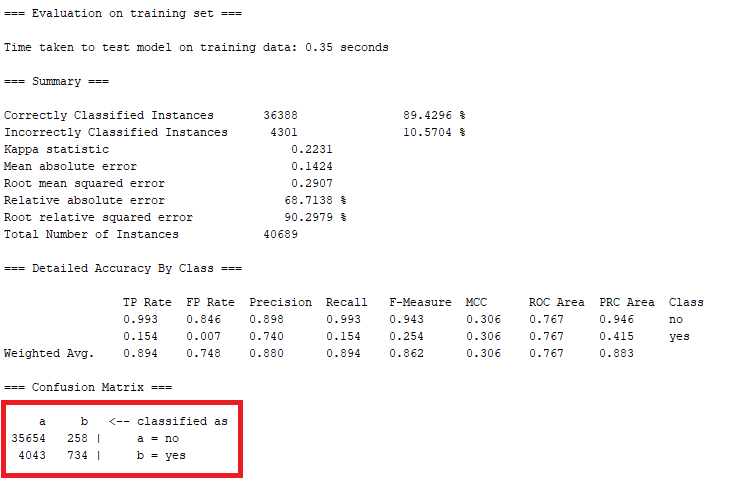
**Summary of findings:**

**Just like our J48, MLP performs better in predicting class “no” instances. However, MLP performs worst in predicting class “yes” instances as shown in the below table**

|  |  |  |  |
| --- | --- | --- | --- |
| **Experiment** | **Parameter setting** | **TP Rate** | **Correctly classified instances** |
| 1 | hidden layers = 3, training time = 5000 and learning rate =0.3 | 0.109 | 56/512 |
| 2 | hidden layers = 5, training time = 500 and learning rate =0.03 | 0.197 | 101/512 |
| 3 | hidden layers = 4, training time = 10000 and learning rate =0.4 | 0.131 | 67/512 |
| 4(balanced dataset) | hidden layers =3, training time = 1000 and learning rate=0.3 | 0.55 | 2216/4032 |

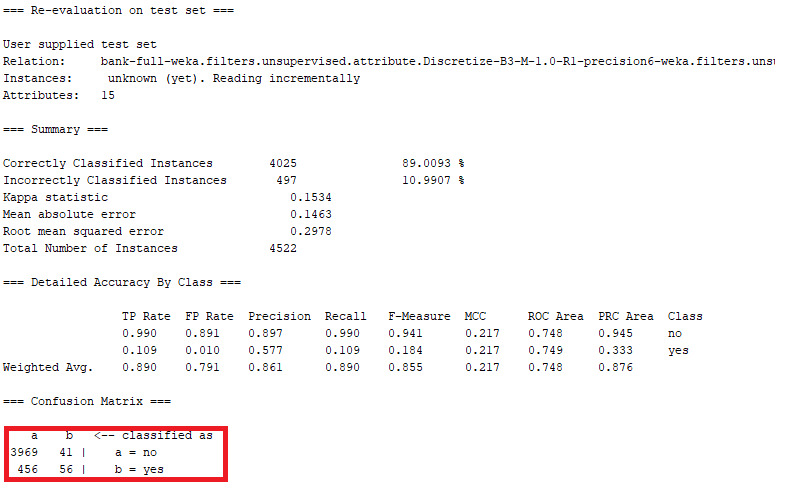
**As we can see from the above table, MLP predicts better when the learning rate is and training time is less with 5 hidden layers. However, it performs worst when the training time is increased to 5000. Another interesting factor is that with high training time (10000) MLP seems to start classifying the class “yes” instances better by predicting 67 out of 512 instances correctly. Hence in our future experiments we can try increasing the training time more than 10000 and check the results. In neural networks, high learning time require low learning rate so that the model is trained properly. However, low learning rate could make the model to be stuck in some state. Hence choosing the learning rate and training time plays an important role in neural networks. In our fourth experiment with balanced data set we see that model performs good in predicting lass “yes” instances classifying 2216 of 4032 instances correctly. Hence, in our case, MLP works better on balanced dataset.**

**Experiment 1: Number of hidden layers = 3, training time = 5000 and learning rate = 0.3**



**Figure 1(a) – Evaluating on training set.**

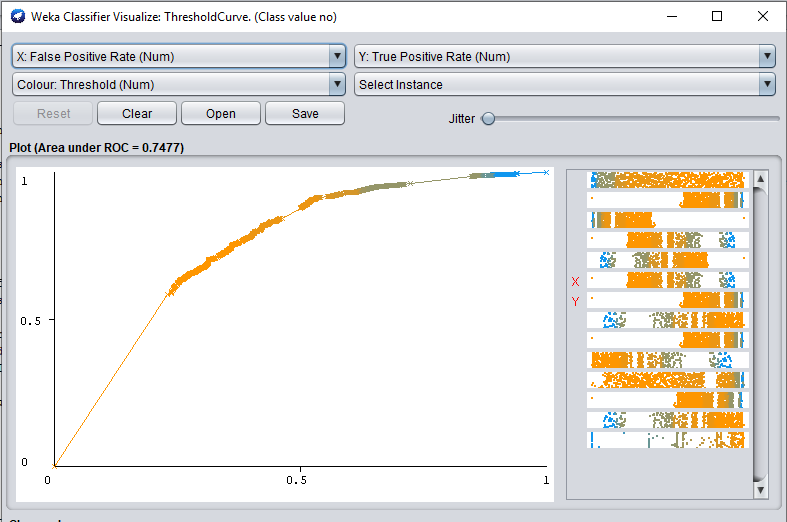
**From the above figure 1(a) we can see that our model has achieved a performance of 89.42% upon evaluating on the training data. Also we can see that the model works better for classifying class ‘no’ with true positive rate of 0.993 over classifying class ‘no’ with TP rate of 0.154.**



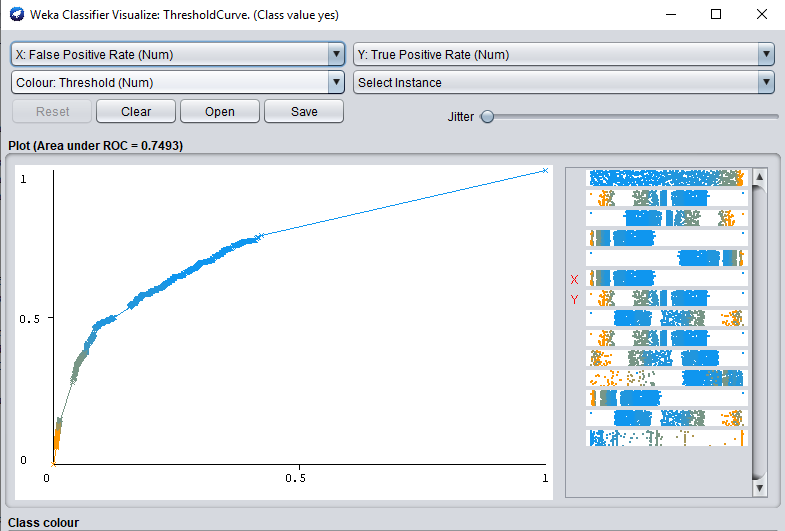
**Figure 1(b) – Evaluating on testing set**

**From the above figure 1(b) we can see that our model’s performance has been decreased when compare to the evaluation on training set. This could be because of the less number of instances for classifying class ‘yes’. Also, we see that the TP rate for classifying class ‘yes’ has been reduced to 0.109 from 0.154 in our evaluation on training set. From the confusion matrix we can infer that 456 class ‘yes’ instances has been wrongly classified as class ‘no’. Hence this model works better in classifying class ‘no’ instances over class ‘yes’ due to the less number of instances for class ‘yes’ which does not allow our model to train properly for class ‘yes’ classification. All the accuracy measures such as F-measure, ROC area are reasonably good.**

**The below figure 1© and 1(d) shows the ROC curve for class ‘no ‘ and class ‘yes’ classification.**

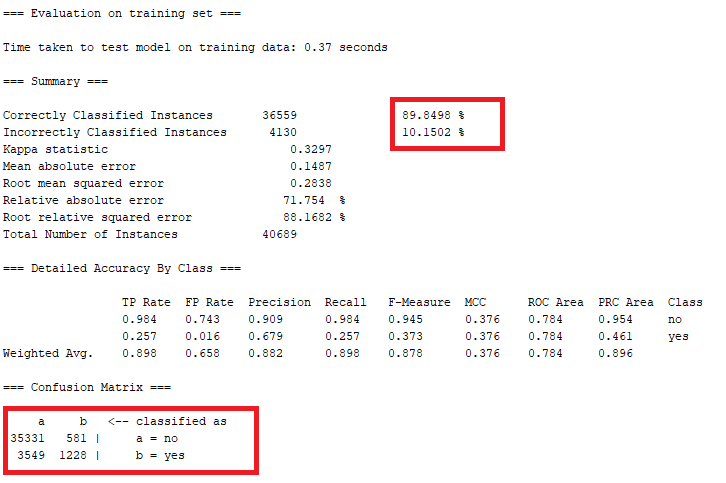


**Figure 1© - ROC curve for class ‘no’.**



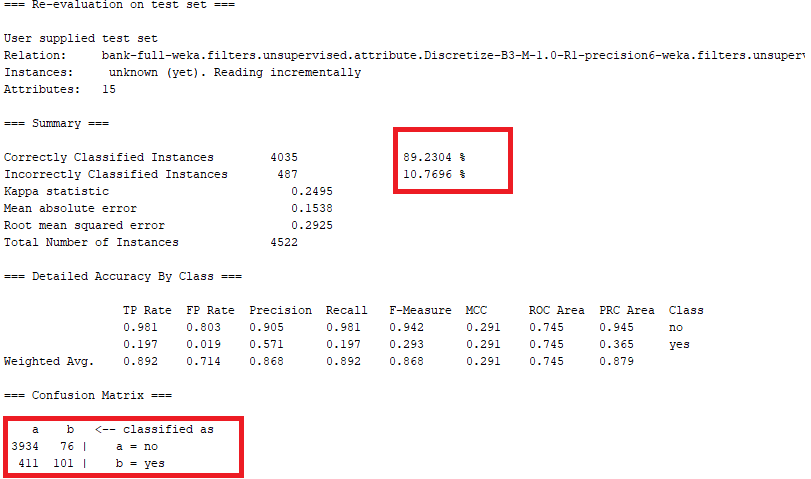
**Figure 1(d) – ROC curve for class ‘yes’.**

**Experiment 2: number of hidden layers = 5, training time = 500 and learning rate = 0.03.**



**Figure 2(a) – Evaluating on training set.**

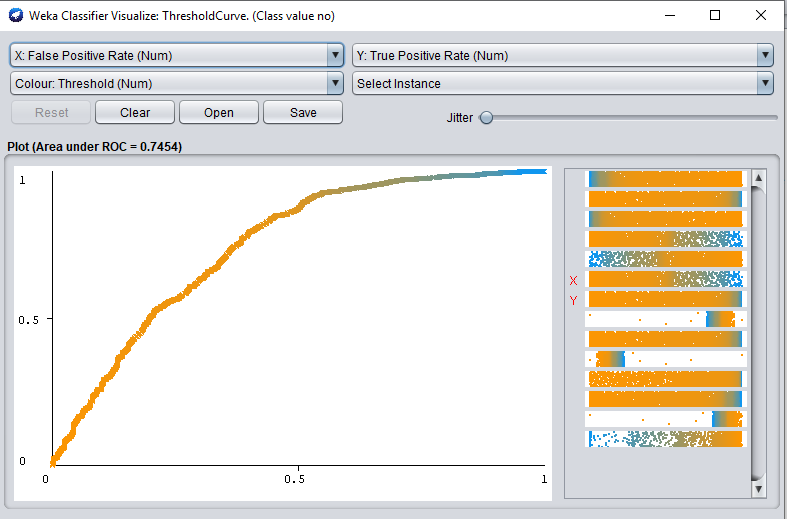
**From the above figure 2(a) we can see that our model performance over training set is approximately 90% which is better than our previous model. This model performs better than previous which will be explained in comparison with the evaluation on testing set.**



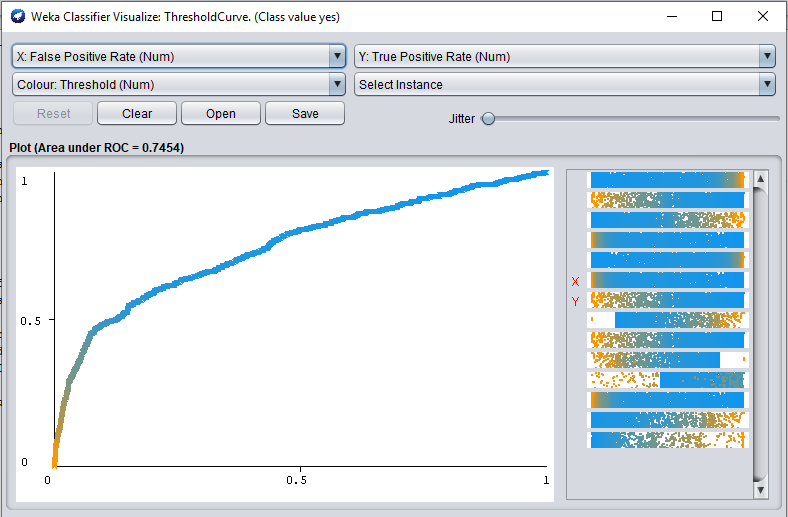
**Figure 2(b) – Evaluation on test set.**

**In above figure 2(b) we can infer that the model performs better in classifying class ‘yes’ instances when compared to our previous model with 3 hidden layers and 0.3 learning rate. This improvement can be seen in TP rate which has increased to 0.197 which is approximately 0.09 higher than our previous model (0.109). Also, we can see that the average of all the accuracy measures such as f-measure, precision and ROC area is reasonably good.**

**The below figures 2© and 2(d) show the ROC curves of our model.**

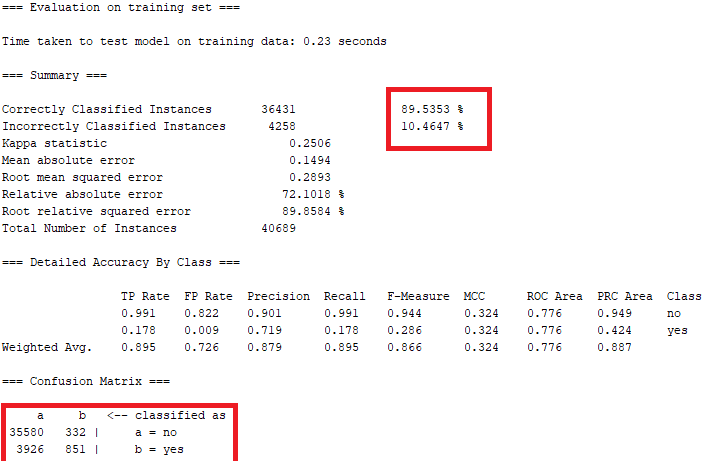


**Figure 2© - ROC curve for class ‘no’.**



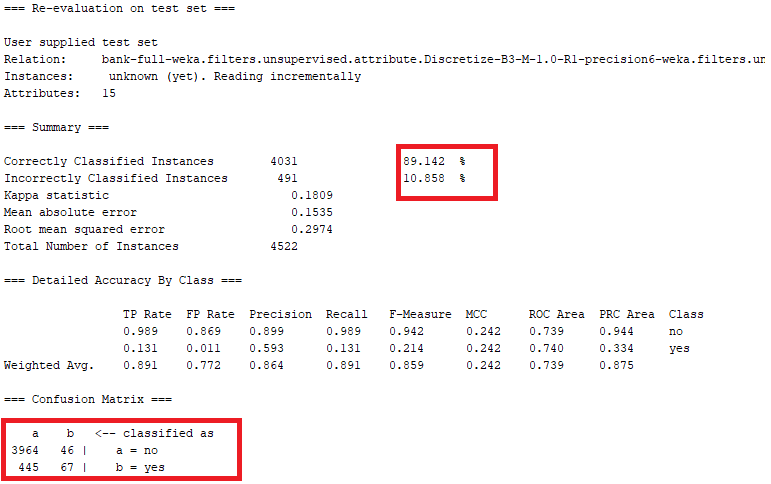
**Figure 2(d) – ROC curve for class ‘yes’.**

**Experiment 3: learning time = 10000, number of hidden layers = 4 and learning rate = 0.4.**

****

**Figure 3(a) – Evaluating on training set.**

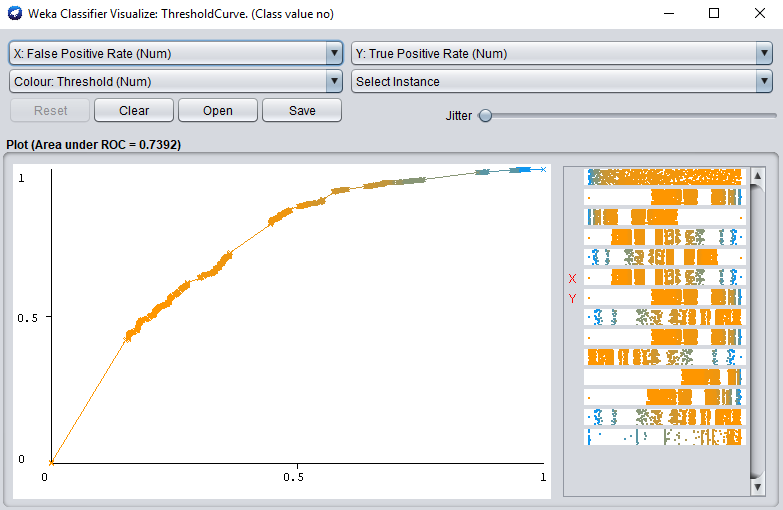
**From the above figure we can see that the model performs better than our first model and little less performance than previous model (89.84%). Though overall performance is good there are certain noticeable factors in the result. We will evaluate these factors after testing our model upon test set as shown in the below figure.**

****

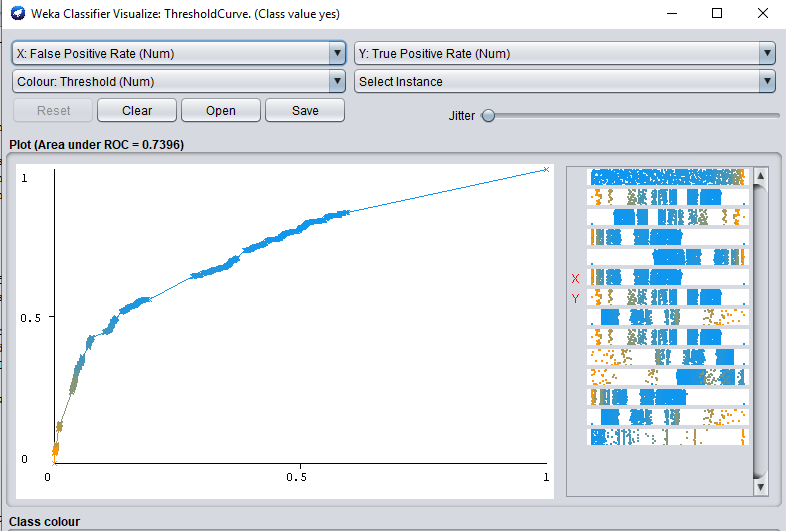
**Figure 3(b) – Evaluation on testing set.**

**In the above figure we can see that there is a performance when tested upon testing set. TP rate for this model is less compared to our previous model. This could be because of the large training time where the neural network would have been overfitted. Hence the model is predicting class ‘no’ while there is a performance drop in predicting class ‘yes’. Also we can see that the average accuracy measures are reasonably good. The model is affected by the less number of instances for class ‘yes’.**

**The below figures 3© and 3(d) show the ROC curve for the model.**

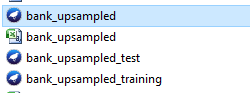
****

**Figure 3© - ROC curve for class ‘no’.**

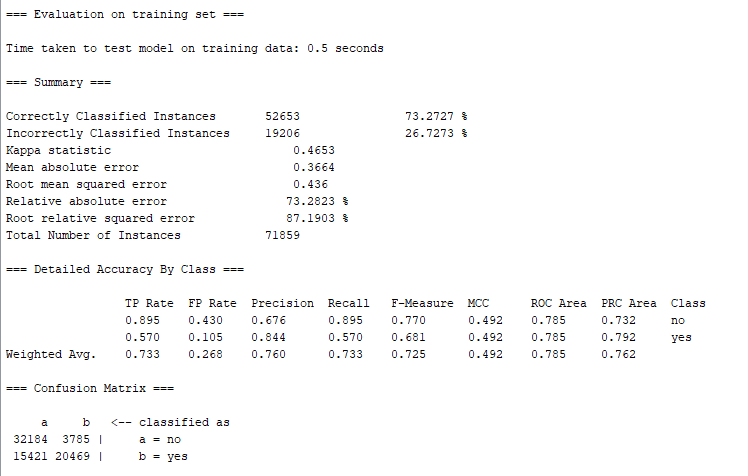
****

**Figure 3(d) – ROC curve for class ‘yes’.**

**Note:** *Since our dataset is not balanced, for the below experiment we balance our dataset by oversampling the class “yes” instances to match the number of class “no” instances. Balancing the dataset was performed using python and scikit learn library. We perform all the pre-processing steps performed previously on the balanced dataset and divide into training set and testing set as shown in the below figure*

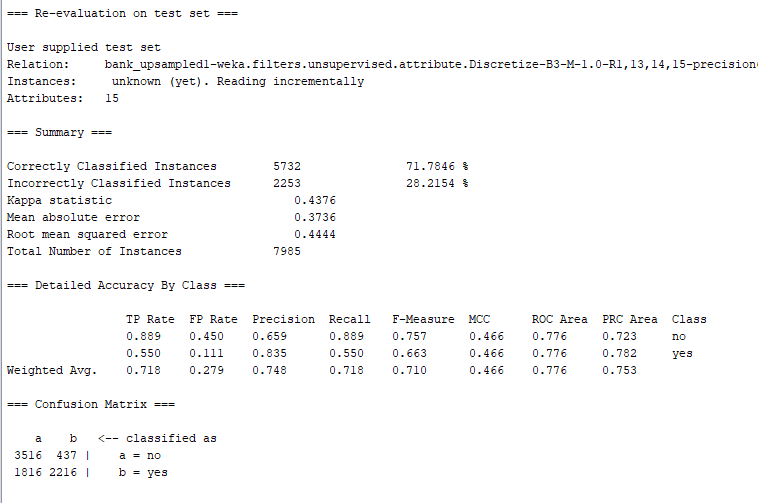


**Experiment 4: number of hidden layers =3, training time =1000 and learning rate =0.3.**



**Figure 4(a) – Evaluating on training set**

**From the above figure 4(a), we can see that our model has a performance of 73.27% after balancing the dataset. Also, we can infer that predicting the class “yes” instances has increased with TP Rate of 0.570. Average of accuracy measures such as precision, recall, f-measure and ROC area are reasonably good on evaluating on the training data. Next, we check the model performance on the test dataset.**



**Figure 4(b) – evaluating on test data**

**Upon testing our model on the test data we see there is a little reduction in the performance of the model. However, we see that the true positive rate for classifying the classes are good with an average TP rate of 0.718. From the confusion matrix we can see that 437 instances have been wrongly classified as class “yes” and 1816 have been wrongly classified as class “no”.**

**Hence, our model performs better on balanced dataset for classifying the minority class “yes” instances.**

# Clustering

For one of the following 2 clustering techniques

1. Use dataset.aff as input. If adaptions are necessary clearly indicate them.
2. Write one or two paragraph analyzing the results of the clustering. Be sure to vary parameters at least 3 times in each case. Support this analysis with screenshots of the following
   1. The clusters and/or a visualization of the clusters
   2. The results of the clusters
   3. Any additional output of the clustering process
   4. Simple references to the notes or URL links to online resources complete with a sentence or two of explanation.
   5. Evaluate the clusters using the “classes to clusters evaluation”. A worked example may be found here <http://www.cs.ccsu.edu/~markov/ccsu_courses/datamining-ex3.html>

## 6. Clustering: K-Means or DBSCAN – 10%

* **Title**: Breast Cancer wisoncsin
* **Data description:** A description of the data in detail under the following subheadings:
  + The problem domain

**Healthcare**

* + The source of the data

**This dataset was taken from Kaggle.**

**Originally the dataset was created by  
  
1. Dr. William H. Wolberg, General Surgery Dept.  
University of Wisconsin, Clinical Sciences Center  
Madison, WI 53792  
wolberg '@' eagle.surgery.wisc.edu  
  
2. W. Nick Street, Computer Sciences Dept.  
University of Wisconsin, 1210 West Dayton St., Madison, WI 53706  
street '@' cs.wisc.edu 608-262-6619  
  
3. Olvi L. Mangasarian, Computer Sciences Dept.  
University of Wisconsin, 1210 West Dayton St., Madison, WI 53706  
olvi '@' cs.wisc.edu**

* + The agencies working with the data

**This data can be used to determine whether the cancer is malign or benign in healthcare.**

* + The intended use of the data

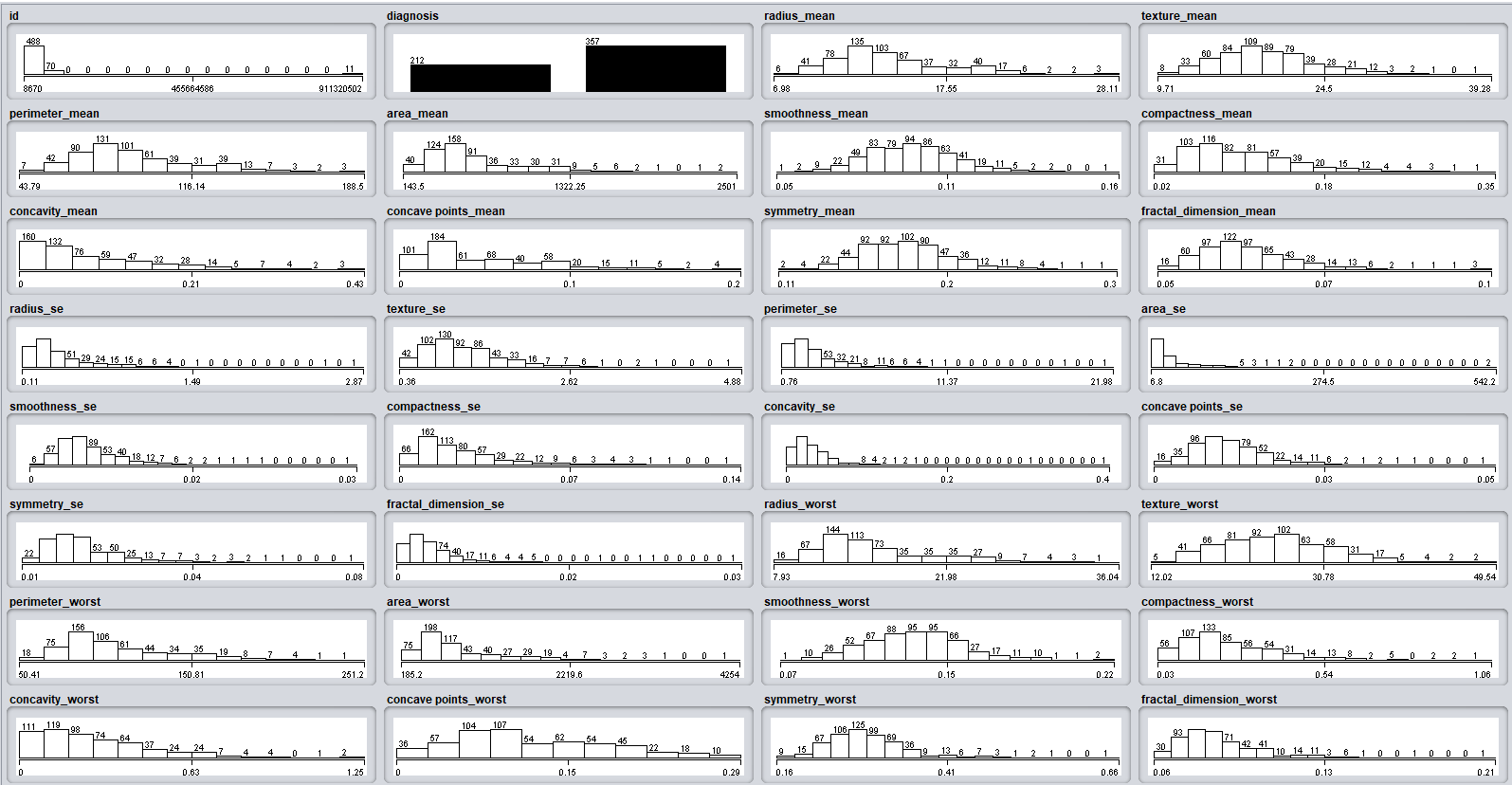
**The data will be used to cluster the data into two clusters, one for malign and another for benign.**

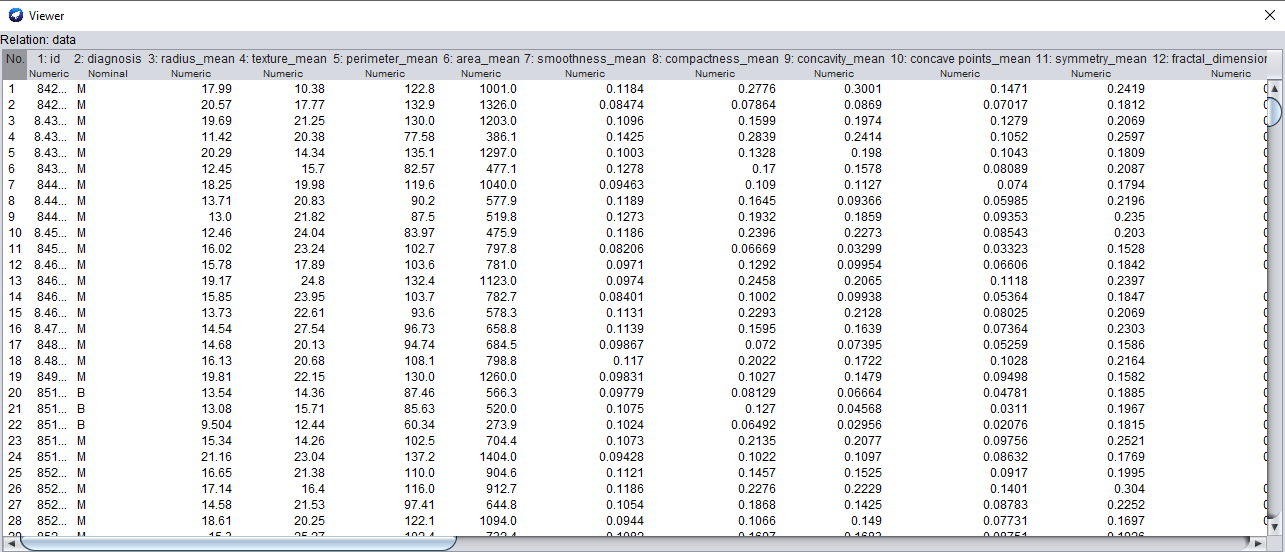
* + The attribute types of the data

**This dataset contains 32 attributes and 569 instances. The list of attributes are as shown below. All the except diagnosis are numeric. Diagnosis is our target class.**

*id, diagnosis, radius\_mean, texture\_mean, perimeter\_mean, area\_mean, smoothness\_mean, compactness\_mean, concavity\_mean, concave points\_mean, symmetry\_mean, fractal\_dimension\_mean, radius\_se, texture\_se, perimeter\_se, area\_se, smoothness\_se, compactness\_se, concavity\_se, concave points\_se, symmetry\_se, fractal\_dimension\_se, radius\_worst, texture\_worst, perimeter\_worst, area\_worst, smoothness\_worst, compactness\_worst, concavity\_worst, concave points\_worst, symmetry\_worst, fractal\_dimension\_worst*

Please include screen shots (with one or two sentences of summary) of the dataset and also of the data summaries that are available though Weka.

****

****

**Summary of Findings**

**The table below gives the summary of our findings using k-means clustering algorithm on breast cancer dataset.**

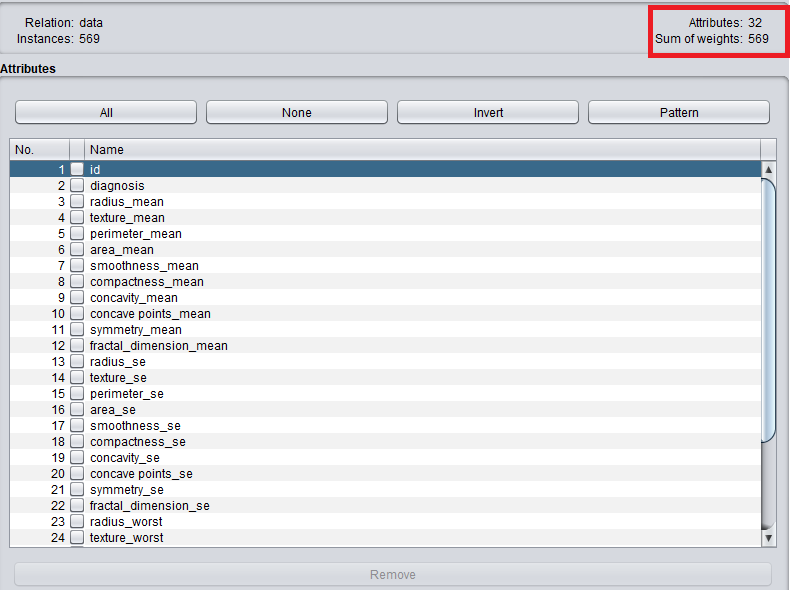
|  |  |  |
| --- | --- | --- |
| **Experiment** | **Parameter setting** | **Incorrectly clustered instances** |
| 1 | number of cluster = 2 and Euclidean distance | 41 |
| 2 | number of cluster = 2 and Manhattan distance | 36 |
| 3 | number of clusters = 3 and Euclidean distance. | 123 |

**From the above figure we can see the results of evaluation using classes to cluster. In our first experiment with 2 clusters and Euclidean distance we see there are 41 data objects that are incorrectly classified into different cluster. From our second experiment with 2 clusters again and Manhattan distance we can see there are only 36 instances that are incorrectly classified. As our third experiment, we chose 3 clusters and Euclidean distance and we see there are 123 instances incorrectly classified. This is expected as there are only two target class labels and we are checking for 3 clusters. Hence increase in incorrectly clustered instances.**

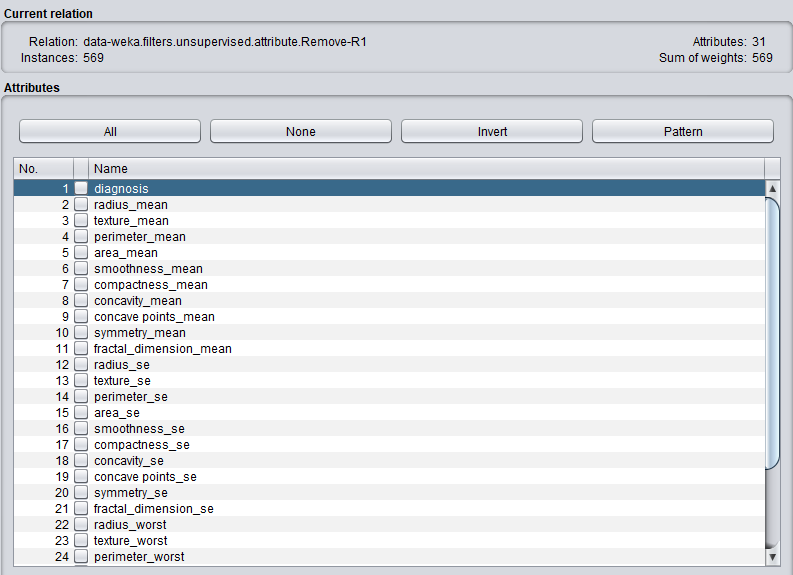
**After looking at the above table we can conclude that k-means works better with Manhattan distance in clustering Malign and Benign cancer. Euclidean distance also works well but Manhattan distance is preferred for this dataset.**

**Preprocessing 1: Removing identity attribute**

**In this step we remove the id column as it does not help in our clustering algorithm.**

****

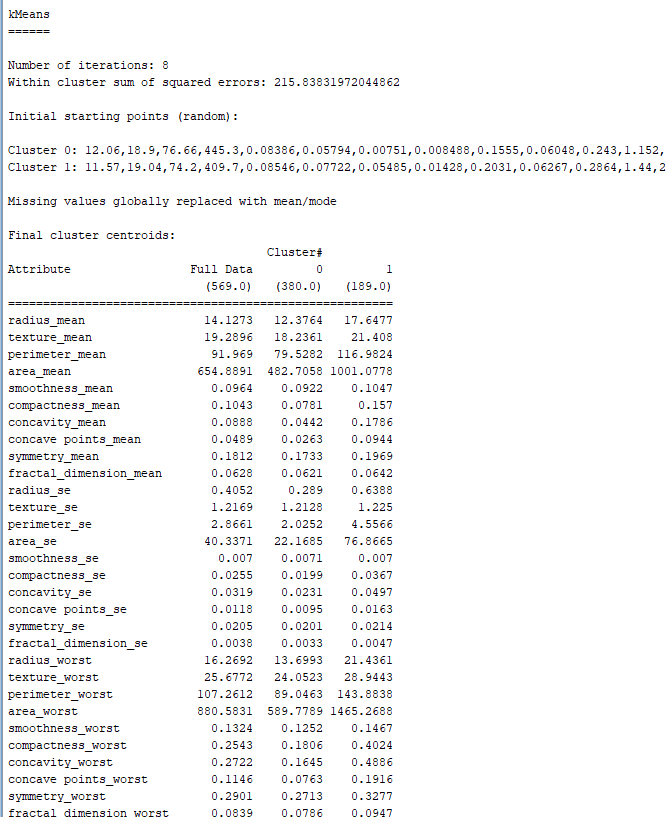
**Figure 1(a) – Before removing the id attributes.**

****

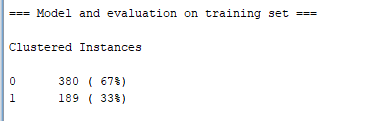
**Figure 1(b) – After removing the id attribute.**

**Experiment 1: Number of clusters = 2, number of iterations = 500 and Euclidean distance.**

**We perform this experiment for two clusters as we know there are only two class labels i.e., Malign (M) and Benign (B). For this experiment we use Euclidean distance for clustering.**

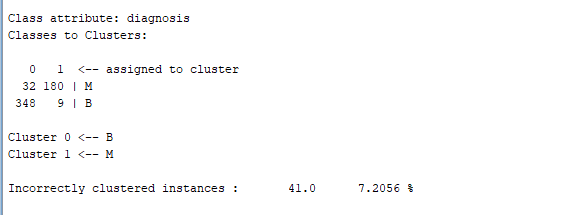
****

**Figure 1(a) – cluster centroids**

****

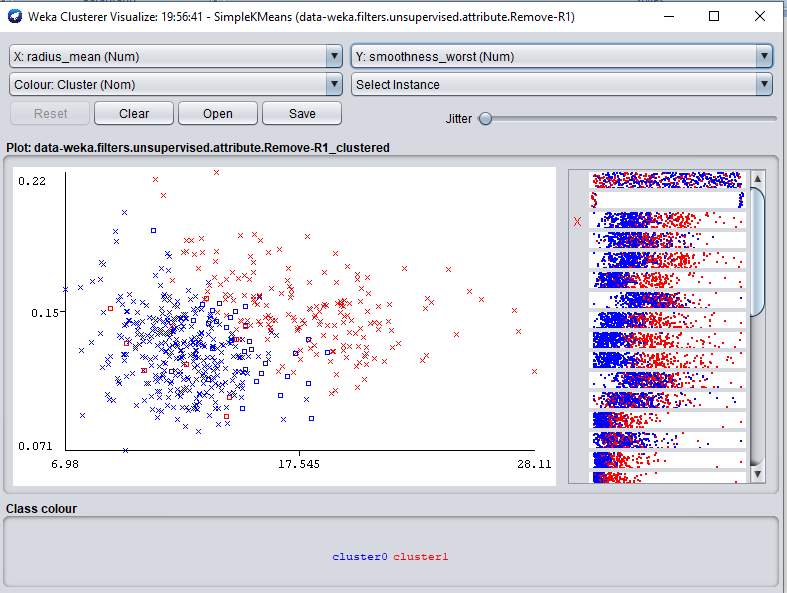
**Figure 1(b) – Result of k-means clustering.**

**The above figure 1(a) and 1(b) shows the result of clustering using Euclidean distance. As we can see 380 instances have been clustered into cluster 1 and 189 have been clustered into cluster 2 that form 33% of the data points. Also from figure 1(a) we can see that the sum of squared errors is 215 for this experiment.**

****

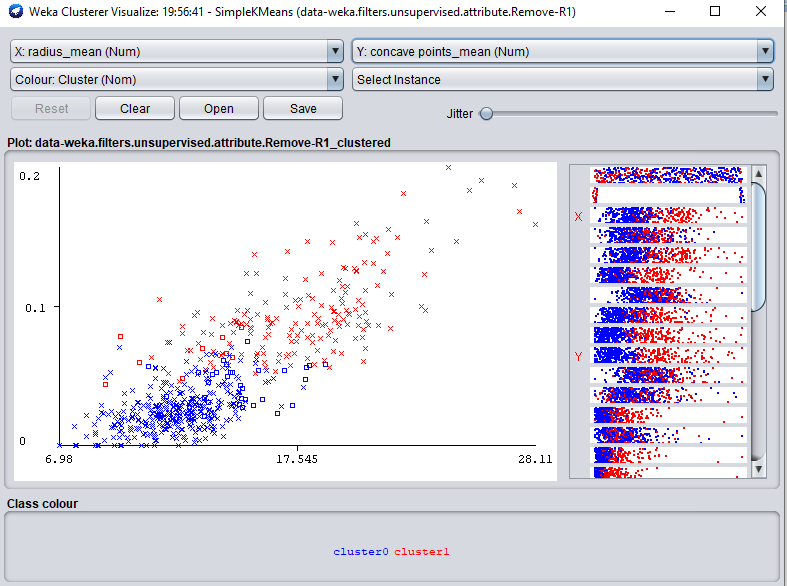
**Figure 1© - Evaluation using classes to cluster evaluation.**

**The above figure gives the result of evaluating clusters based on classes. As we can see from the confusion matrix, 32 instances have been incorrectly classified into class M (cluster 1) and 9 have been wrongly clustered into class B (cluster 0). Totally 7.2% of the instances have been incorrectly classified in this experiment using Euclidean distance.**

****

**Figure 1(d) – *radius\_mean vs smoothness\_worst***

**From the above figure 1(d) we can see that there is certain distinction between the clusters when plotting *radius\_mean vs smoothness\_worst.* Also, we can see the wrongly clustered instances with square dot representation.**

****

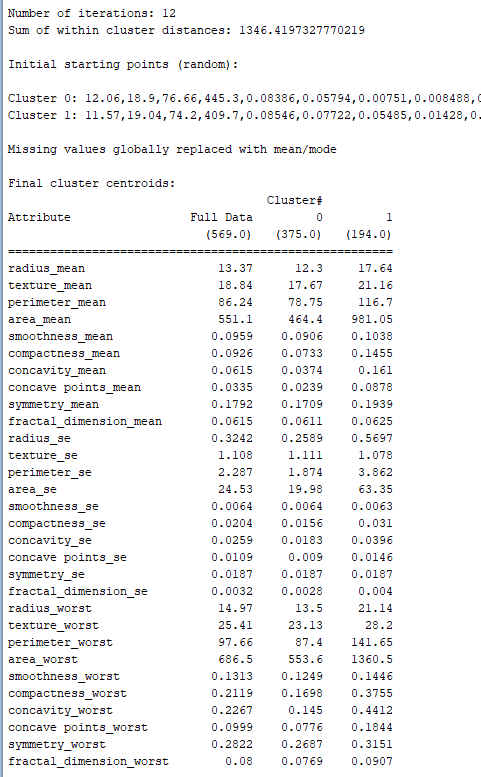
**Figure 1(e) - *radius\_mean vs concave points\_mean***

**The above figure 1(e) gives the cluster representation by plotting *radius\_mean vs concave points\_mean.* This has certain degree of distinction among clusters.**

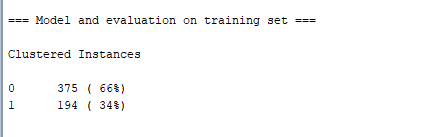
**Experiment 2: Number of clusters = 2, max number of iterations = 1000 and Manhattan distance.**

**We perform this experiment for two clusters as we know there are only two class labels i.e., Malign and Benign. For this experiment we use Manhattan distance for clustering.**

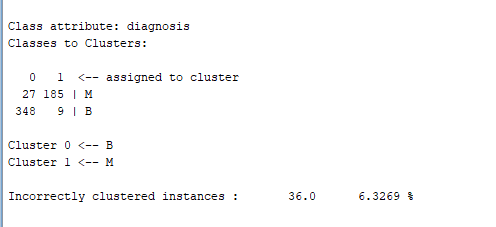
**The above figure 2(a) and 2(b) shows the result of clustering using Euclidean distance. As we can see 375 instances have been clustered into cluster 0 and 194 have been clustered into cluster 1 that form 34% of the data points. Also from figure 2(a) we can see that the sum of squared errors is 1346 for this experiment which very high when compared to our previous experiment with Euclidean distance where the SSE was 215.**

****

**Figure 2(a) – cluster centroids**

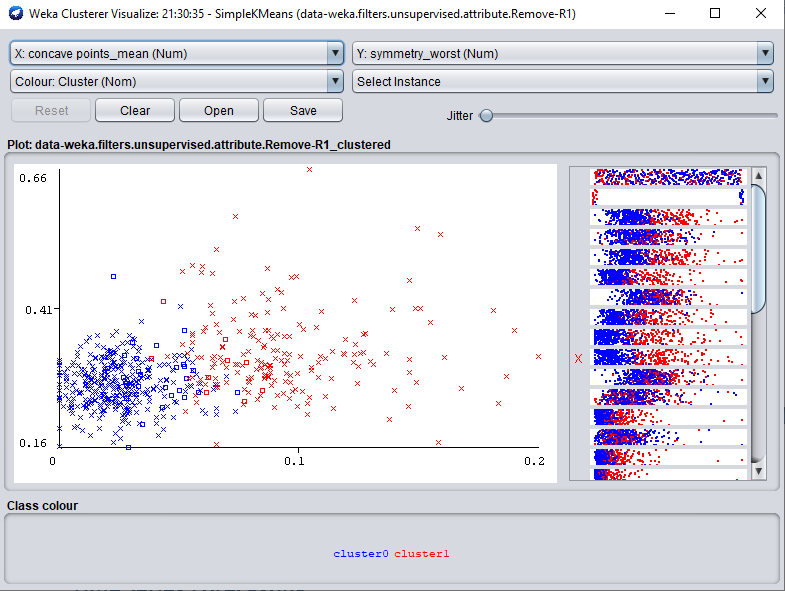
****

**Figure 2(b) - Result of k-means clustering.**

****

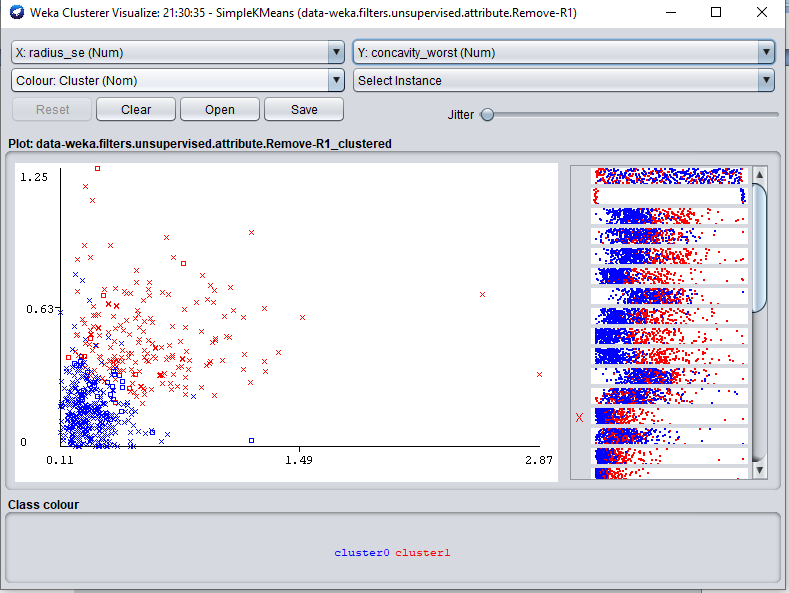
**Figure 2© - Evaluation using classes to cluster evaluation.**

**In the above figure 2© we evaluate our clusters based on classes. As we can see from the resultant confusion matrix, 27 instances have been incorrectly classified into class M (cluster 1) and 9 have been wrongly clustered into class B (cluster 0). We see some improvement in clustering where the incorrectly clustered instances for class M has reduced to 27 from 32 in our previous experiment. Totally 6.3% of the instances have been incorrectly classified in this experiment using Manhattan distance.**

****

**Figure 2(d) – *concave points\_mean vs symmetry\_worst***

**From the above figure 2(d) we can see that there is certain distinction between the clusters when plotting *concave points\_mean vs symmetry\_worst.* Also, we can see the wrongly clustered instances with square dot representation. In the above plot we can see that data points in cluster 0 are more closer than data points in cluster 1.**

****

**Figure 2(e) – *radius\_se vs concavity\_worst***

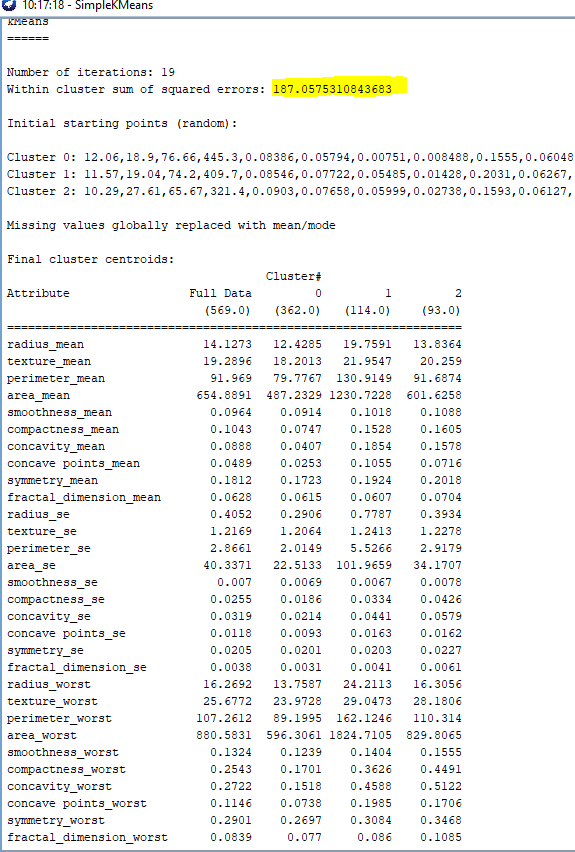
**The above figure 2(e) gives the cluster representation by plotting *radius\_se vs concavity\_worst.* This has certain degree of distinction among clusters with data points in cluster 0 being more close to each other than data points in cluster 1.**

**Experiment 3: number of cluster = 3, max iterations = 1000 and Euclidean distance.**

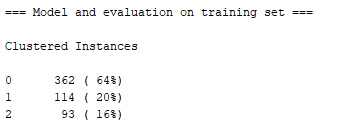
**We try performing this experiment using 3 clusters to check if the cluster classifications will be distinct. Since we have only two class labels and we are checking for 3 clusters we get an additional class saying “No class”.**

**The below figures 3(a) and 3(b) show the result of clustering using Euclidean distance and for three clusters. As we can see 362 instances have been clustered into cluster 0, 114 have been clustered into cluster and 92 into cluster 2. Also from figure 3(a) we can see that the sum of squared errors is 187 which is comparatively less when compared to our previous two experiments.**

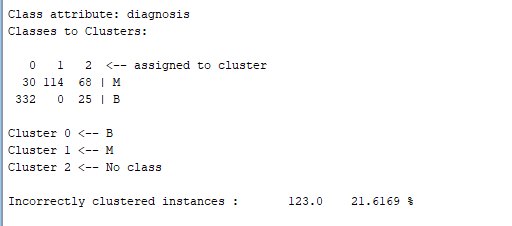
**This change in SSE is due to the increase in number of clusters which pulls the data points close to the cluster cetroids.**

****

**Figure 3(a) – cluster centroids**

****

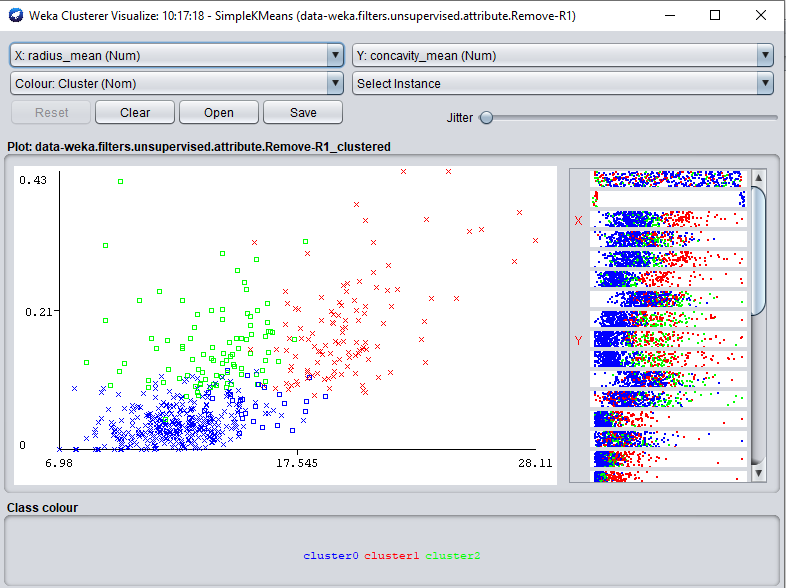
**Figure 3(b) – Result of k-means clustering using Euclidean distance.**

****

**Figure 3© - Evaluation using classes to cluster evaluation.**

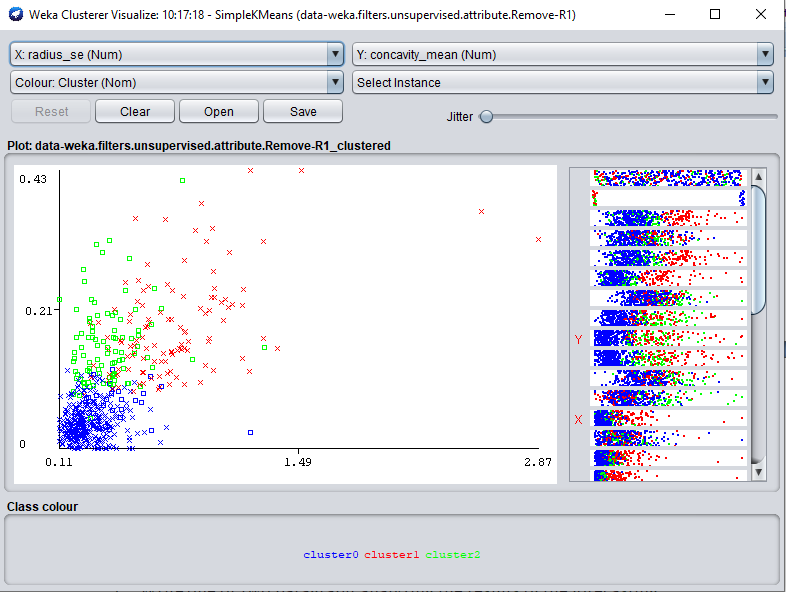
**The above figure 3© gives the evaluation of our clusters using class labels i.e., Malign and Benign. Because we have taken more number of clusters than our labels, an additional class has been introduced as ‘No class’. From the confusion matrix we can see that 68 instances of class M and 25 instances of class B have been wrongly clustered into class 2. There is total 123 incorrectly classified instances due to the introduction of cluster 2.**

**We will now see if our cluster 0 and cluster 1 are distinct after introducing the third cluster.**

****

**Figure 3(d) – *radius\_mean vs concavity\_mean***

**From the above figure we can see that the two clusters 0 and 1 are distinct after the introduction of cluster 2. We can consider the data points in clusters as noise as they are not classified into any class in this experiment.**

****

**Figure 3(e) – *radius\_se vs concavity\_mean***

**From the above figure 3(e) we can see that data points in cluster 0 are close. However, data points in cluster 1 and 2 are far when plotting *radius\_se vs concavity\_mean*.**

# Time Series Forecasting

For the following task

1. Use dataset.aff as input. If adaptions are necessary clearly indicate them.
2. Write one or two paragraph analyzing the results of the forecasting. Support this analysis with screenshots of
   1. The regression equation
   2. Diagram of the historical values
   3. Diagram of the predictions

## 7. Time Series Forecasting – 15%

* **Title**: **House property Sales for Time Series**
* **Data description:** A description of the data in detail under the following subheadings:
  + The problem domain

**Housing sales**

* + The source of the data

**This dataset was taken from Kaggle.**

<https://www.kaggle.com/deltacrot/property-sales>

* + The agencies working with the data

**This data can be used in real estate and banking to provide loan to own a house.**

* + The intended use of the data

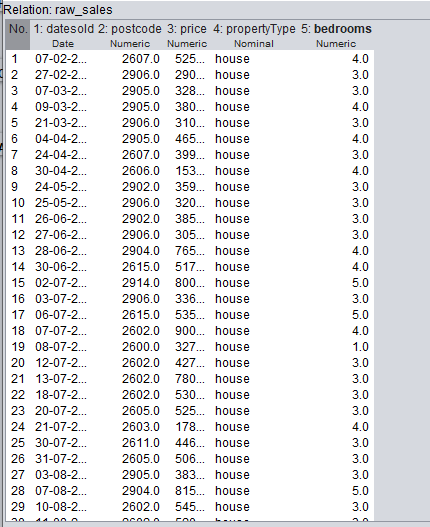
**The data will be used to forecast the prices of the house.**

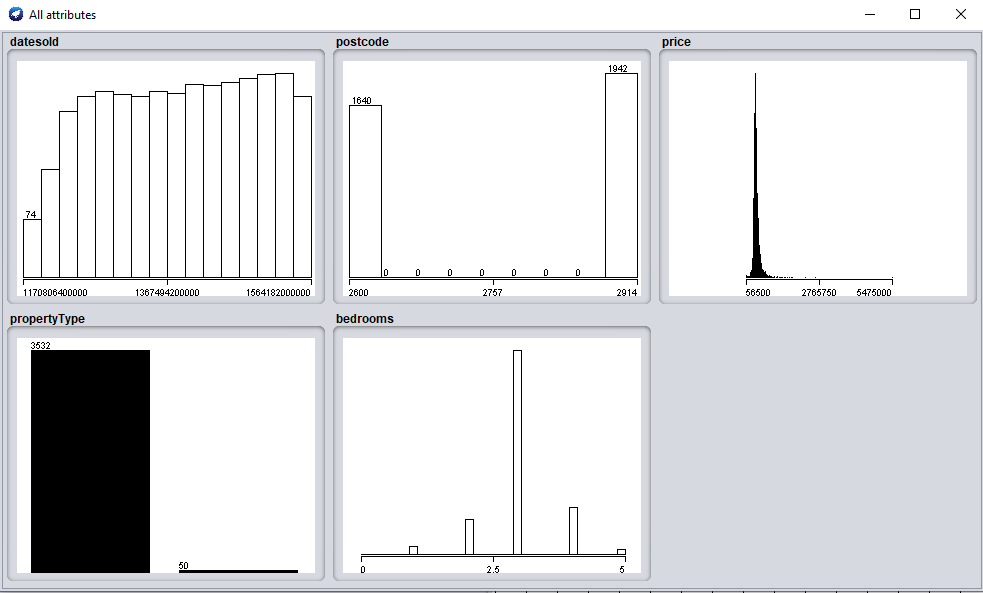
* + The attribute types of the data

**This dataset contains 5 attributes and 3582 instances. Also this data set contains houses sold from 2007-2019.**

|  |  |  |
| --- | --- | --- |
| Attribute name | Type | Description |
| datesold | Date | Gives the date on which the property was sold. |
| postcode | numeric | gives the zip code of the property. |
| price | numeric | price of the house. |
| bedrooms | numeric | number of bedrooms in the house. |

Please include screen shots (with one or two sentences of summary) of the dataset and also of the data summaries that are available though Weka.





* **Objective**: Your objective. You can update this as you progress through your assignment revising it and making it more specific.

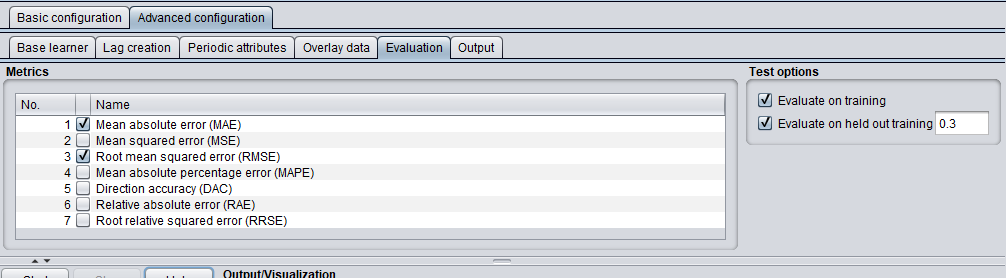
**The objective is to design a model to predict the price of the house over the upcoming days.**

* **Summary of Findings**: This should be written following the application of your data mining techniques.

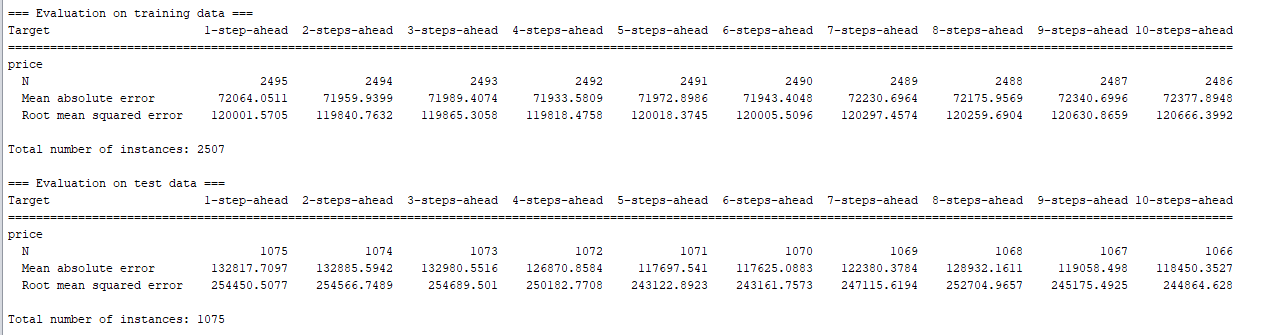
**After performing the below two experiments, we see that the model selected does not forecast the future accurately. In our first experiment with 12 lag with see that there is less smoothing performed. However, in the second experiment with 6 lags the values has been smoothed by calculating the average. We see almost horizontal straight line in our second experiment at each step.**

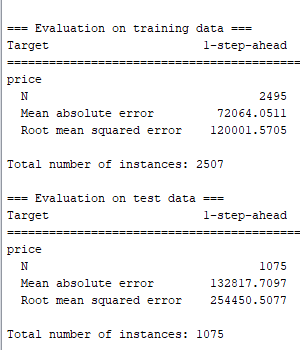
**Experiment 1:**

**To evaluate our forecasting model we take 0.3 of the instances to train and test our prediction as shown below. We would check for Mean absolute error(MAE) and Mean Squared Error(MSE)as shown in the below figure.**



**Figure 1(a) – Advanced configuration for evaluation**

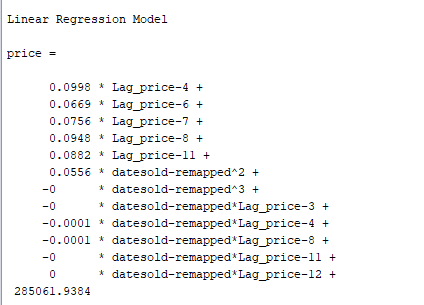




**Figure 1(b) – Evaluation on training and test data**

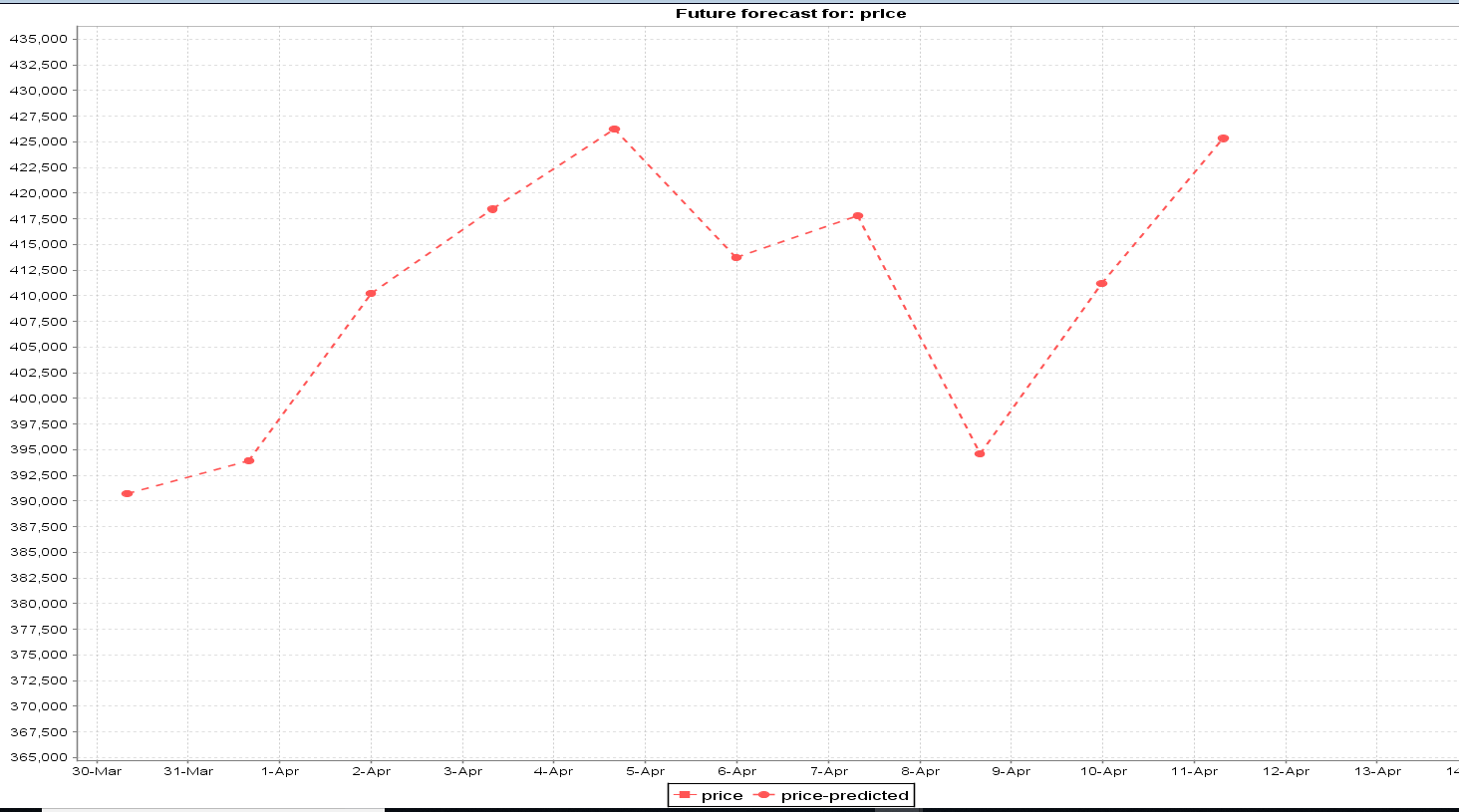
**From the above figure we can see that the MAE and MSE are very high for our regression equation and hence needs some tuning to check the performance.**

**The below figure gives the linear regression equation of the model used to forecast the house prices.**

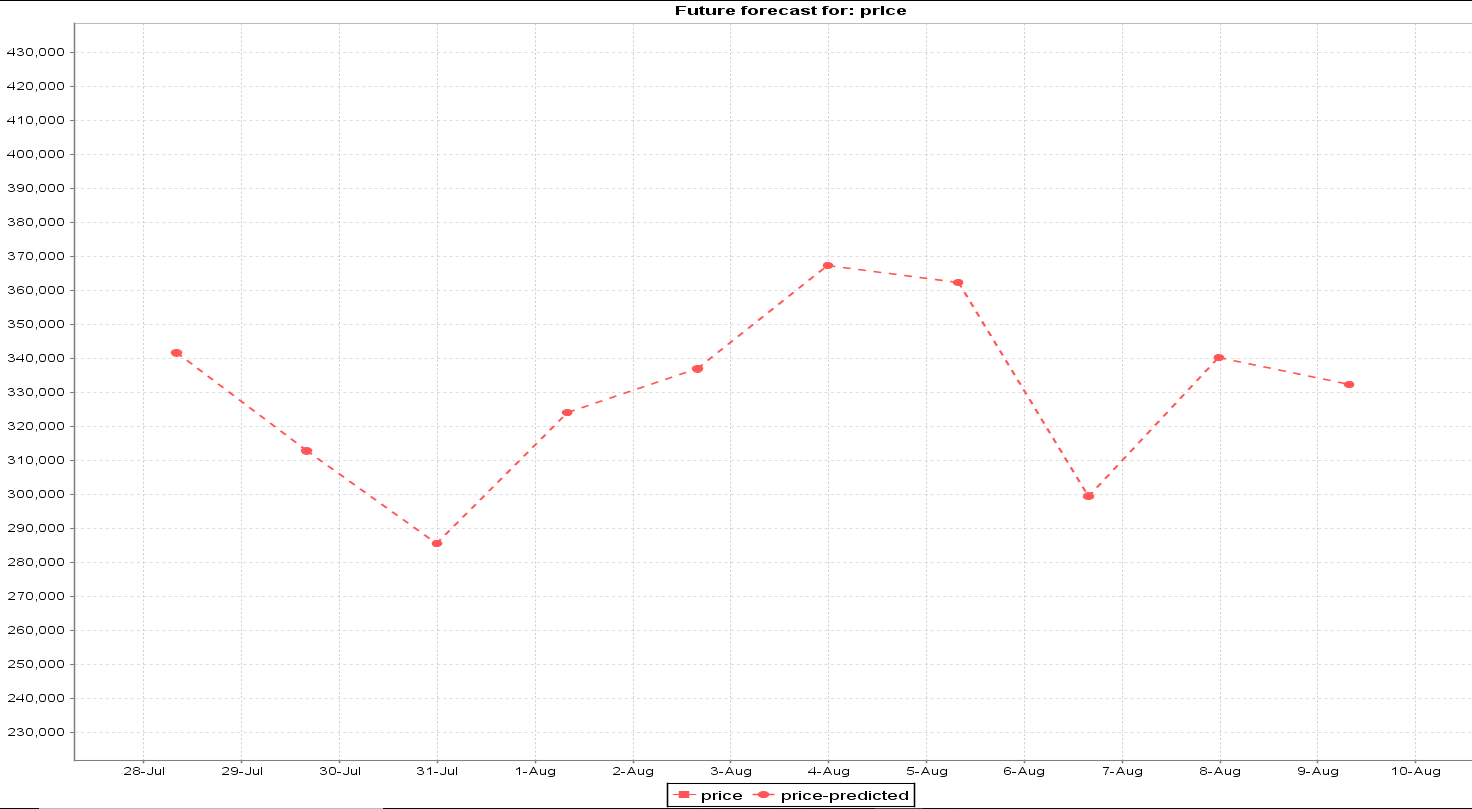


**Figure 1(c) – Regression equation.**

**The below figures shows the prices of the house for train predicted and test predicted data.**

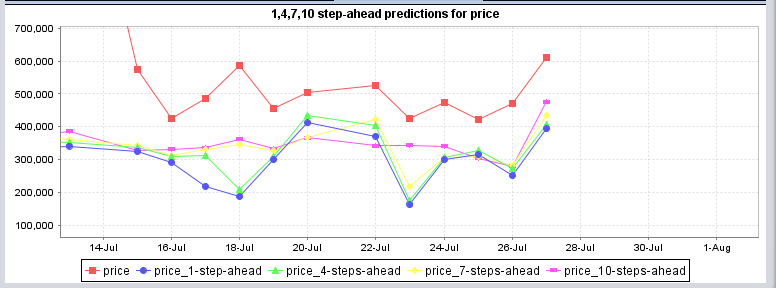


**Figure 1(d) – Train future predicted**



**Figure 1(e) - Test future predicted.**

**From the above figures 1(d) and 1(e), we can see that there is huge difference between predicted and actual price.**

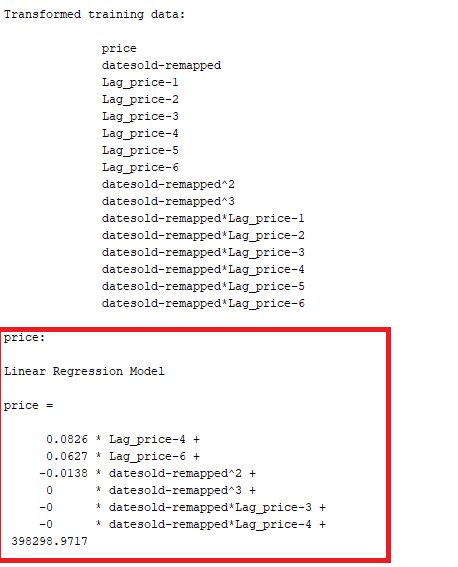


**Figure 1(f) – Test predictions at steps**

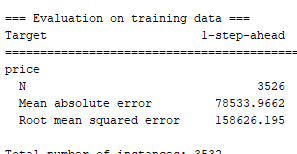
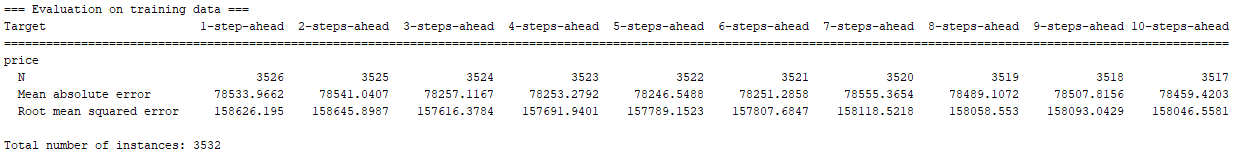
**From the above figure we can see that after step 7 there is some smoothing at step 10.**

**Experiment 2:**

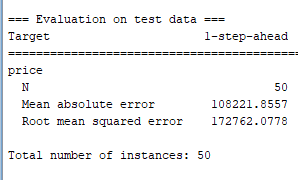
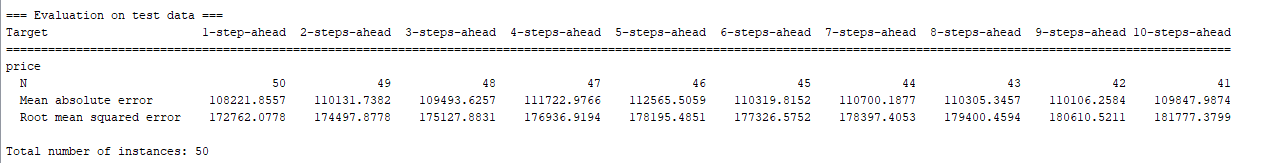
**In this experiment, we use linear regression model and custom lag between 1 and 6. We use 50 instances for evaluation on test data.**



**Figure 2(b) – Transformed training data and linear regression equation.**



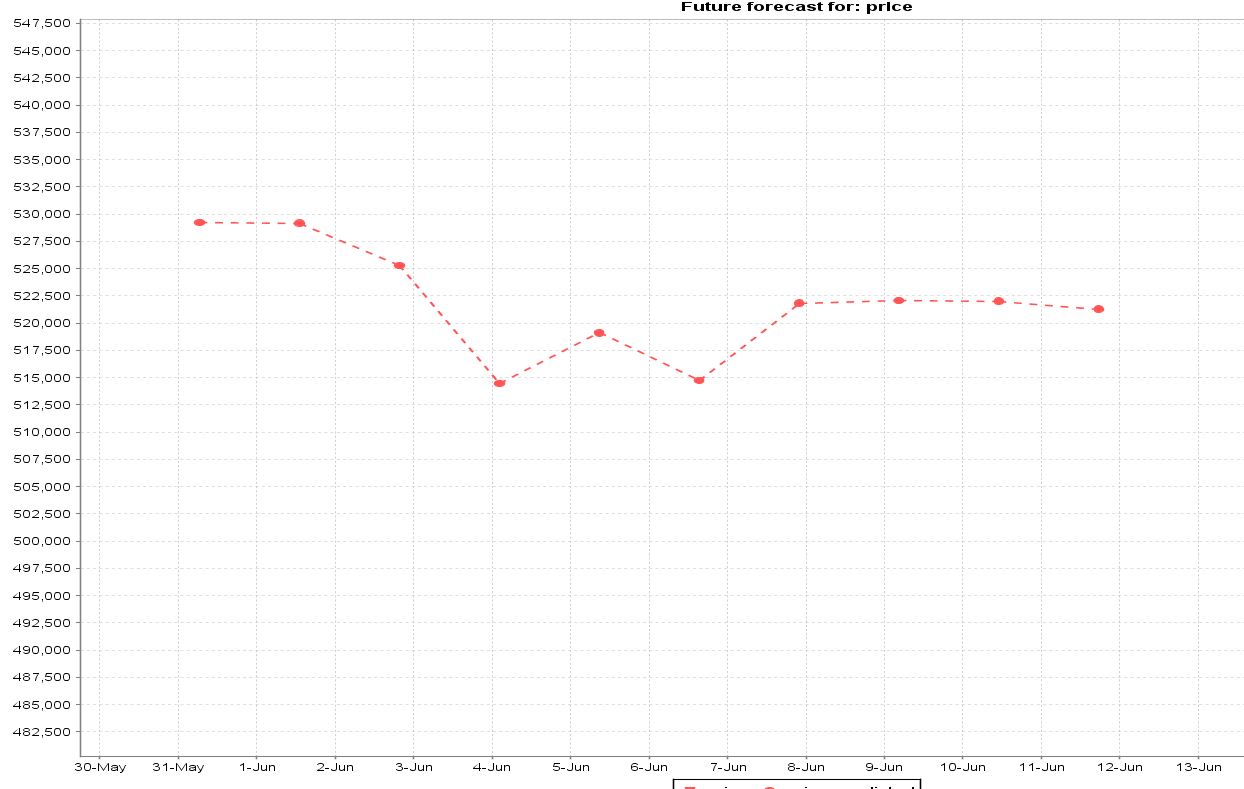
**Figure 2© - Evaluation on train data**



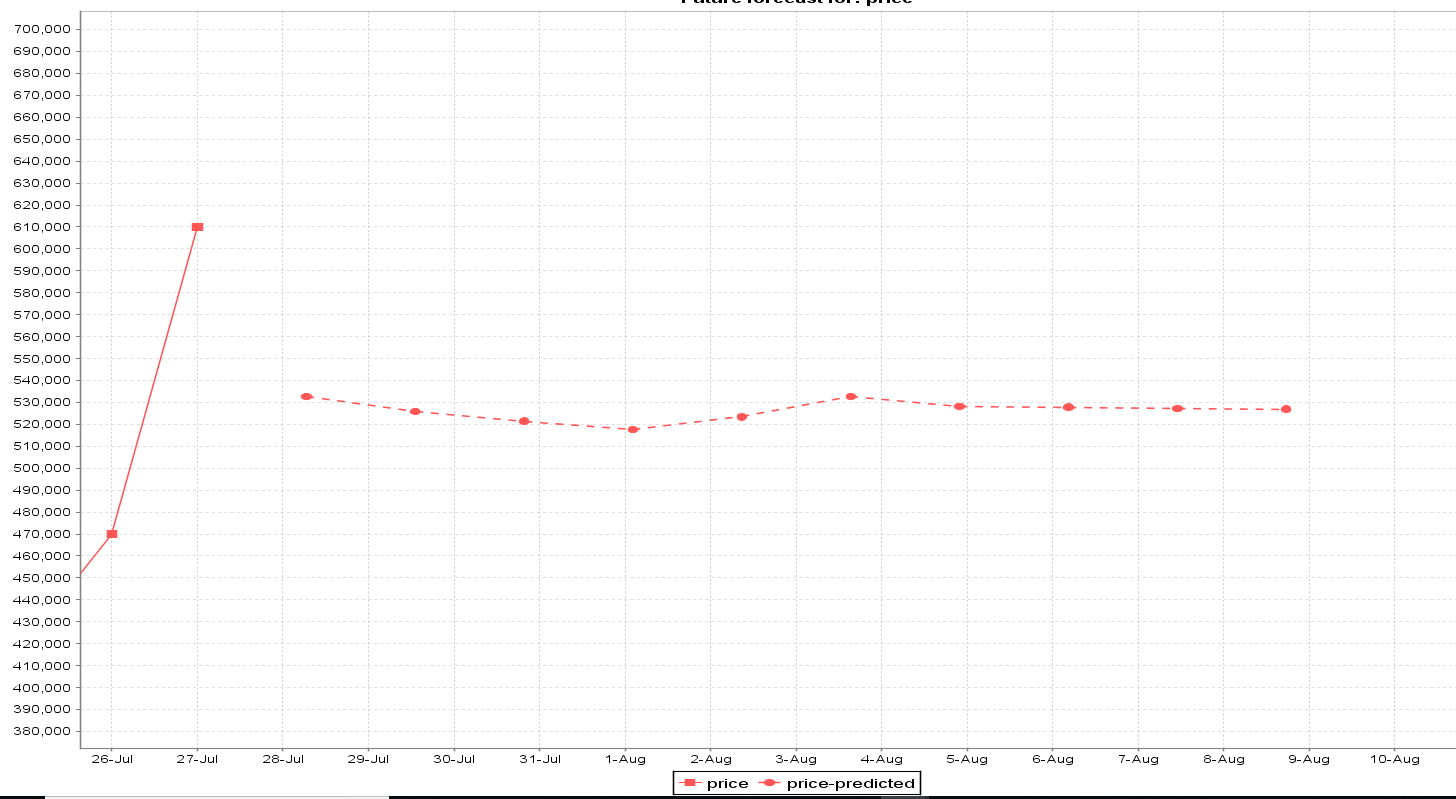
**Figure 2(d) – Evaluation on test data result**

**From the above figure 2(c) and 2(d) we can see that the MSE and MAE for evaluation on test data have been reduced when compared to our previous model.**

**The below figures 2(e) and 2(f) shows the prices predicted for train and test data.**



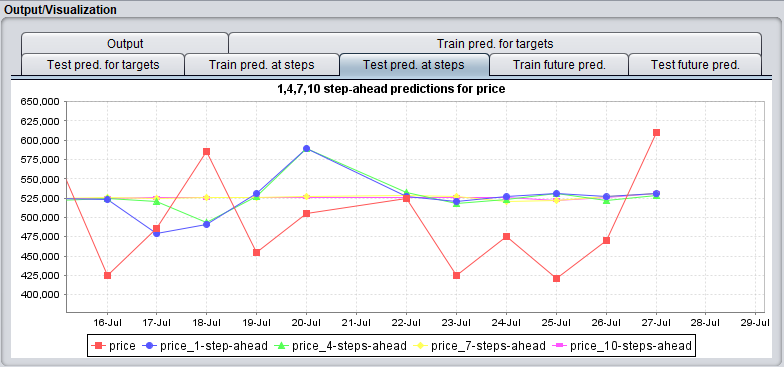
**Figure 2(e) - Train future predicted.**



**Figure 2(f) – Test future predicted**

**From the above figures 2(e) and 2(f) we can see that the data is smoothed due to the moving average. This could lead to some loss of information. This change is being seen due to the reduction in the lag from 12 to 6.**

**The below figure 2(g) shows the predictions at step 1, 4, 7 and 10. It can be observed from the graph that how the data is smoothed at each step of the prediction. This makes the price to be very little varying as smoothing is done by averaging.**



**Figure 2(g) – Test prediction at steps**