Portuguese banking institution marketing campaign classification Report

Dayi Fang

June 11, 2017

Introduction:

This project focuses on analyzing a real-world marketing campaign dataset to classify if the campaign will be success or not based on limited information. Within this project, we have performed exploratory data analysis and imbalanced data analysis to understand and resample the raw data. We have also constructed various classification models and evaluated their performances include linear discriminant analysis, quadratic discriminant analysis, logistic regression with lasso regularization, random forest, k nearest neighbor, support vector machine, and neural network algorithms. In conclusion, we have discussed and compared each model to identify the best one.

Data:

The data is collected from UCI Machine Learning Repository under the name Bank Marketing Data Set. Our raw data contains 45211 different observations and 17 variables, the response variable is a binary variable indicating whether the client subscribed a term deposit (yes) or no (no). The original dataset and detailed variable descriptions can be found from the following URL:

http://archive.ics.uci.edu/ml/datasets/Bank+Marketing#

Since the raw data is clean and managed, we do not need to do much data cleaning or regrouping. We simply drop three variables: “contact” is dropped because it is an useless communication type indicator, which does not affect our response variable; “duration” is dropped since we do not know its value until the result of campaign gets out; “pdays” is dropped because it contains incorrect information.

In addition, we have standardized four continuous variables (“age”, “balance”, “campaign”, and “previous”) by their standard deviations to minimize the effects of outliers, and generated dummy variables for each of our categorical predictor.

For missing values, the raw dataset contains certain rows named as “unknown” for all categorical variables indicating their missing values. Most of those “unknown” categories take a small proportion of the population, so that it is ok to leave them in our dataset. The only exception is the “poutcome”, which contains “unknown” for about 80% of its total number of observations. Therefore, we can either drop this variable or keep it to find out whether other levels of “poutcome” have significant influence on the response variable. We chose to keep this variable in our study.

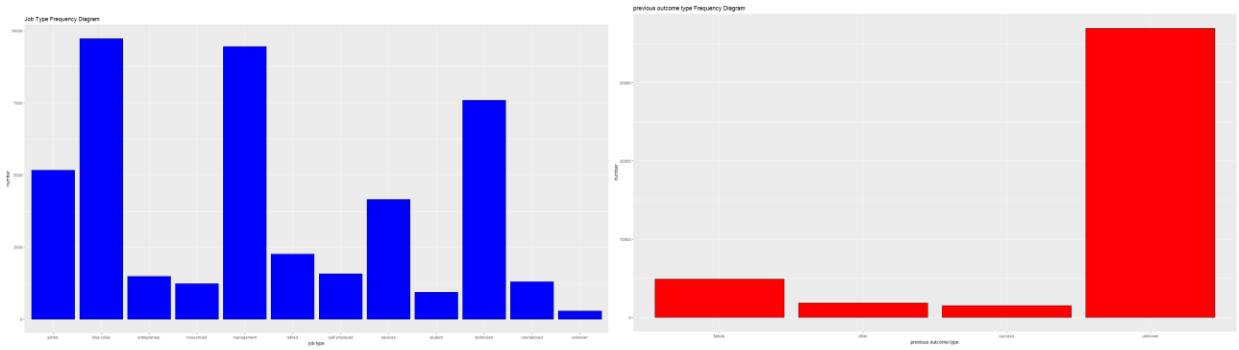


Figure 1

Imbalanced data is also another important issue in our analysis. The raw data has 90% response as “no” and 10% as “yes”, which is not severely imbalanced not still a fact needs to be considered. There are several resampling methods to deal with imbalanced data problem, such as oversampling and undersampling. However, because oversampling may lead to overfitting problem, and undersampling may cause loss of information. We simply choose to use different performance measures instead of the regular overall accuracy rate to evaluate our models. Figure 1 shows the frequency plots of “job” and “poutcome”.

Performance Measures:

1.) Overall Accuracy:

In classification problems, overall accuracy is calculated as: ratio of the number of correct predictions to total number of predictions. This is a good performance measure for balanced data. However, we have an imbalanced dataset with 90% negatives and 10% positives, we can simply

achieve a 90% overall accuracy by making all predictions as negative. Therefore, overall accuracy is not a good evaluation score in our case.

2.) Sensitivity and Recall:

Both sensitivity and recall are defined as: total number of true positive predictions/ total number of actual positive observations. The true positive prediction is defined as the cases that we predict the client will subscribe and they actually subscribe. This measurement can be understood as "when it is actually 'yes', how often does our model predict 'yes'". Sensitivity and recall are good measurements in our case, since we care more about correctly finding out the number of clients that are more likely to subscribe term deposits.

3.) Precision and Positive Predicted value:

Both precision and positive predicted value are defined as: total number of true positive predictions/ total number of predicted "yes". This measurement can be understood as "when our model predicts the client will subscribe, how often is it correct." Precision and positive predicted value are also important measurements in our study, since we want to how accurate is our model when we are predicting "yes". The ideal model should have both high sensitivity/recall rate and high precision/positive predicted value rate.

4.) Specificity and Negative Predicted Value:

Specificity is defined as: total number of true negative predictions/ total number of actual negative observations. It can be understood as "when the clients are not subscribing, how often does our model predict they will not subscribe." Negative predicted value is defined as: total number of true negative predictions/ total number of predicted "no". It can be understood as" when our model predicts the client will not subscribe, how often is it correct." We are introducing these two measurements because sometimes our model does not perform well in correctly predicting the client will subscribe. In that case, we may look at these two measures and consider filtering out those clients that are less likely to subscribe. By this process, we can find a smaller group of clients are more likely to subscribe the deposits to apply future analysis.

5.) ROC and AUC:

ROC stands for Receiver Operating Characteristic, and AUC stands for Area Under Curve. Both are served as general measurements reflecting the overall performance of the model. ROC can also help select the best threshold that achieve the best accuracy rate. However, as we stated in overall accuracy part: since we have imbalanced data, we only use these measures as inferences but not value them as the gold standards of our model evaluations.

Methodology and Model:

1.) Linear discriminant analysis and quadratic discriminant analysis

LDA and QDA are generalizations of the fisher's linear discriminant, they assume that the class conditional distribution of the data $P(X|y=k)$ is modelled as a multivariate Gaussian distribution with density:

$$p(X|y=k) = \frac{1}{(2\pi)^n|\Sigma_k|^{1/2}} \exp\left(-\frac{1}{2}(X-\mu_k)^t\Sigma_k^{-1}(X-\mu_k)\right)$$

Thus, use the conditional distribution to make predictions by applying Bayes' rule.

The only difference between LDA and QDA is that they have different assumptions in setting covariance matrices.

LDA and QDA are simple structured and can be easily computed without tuning hyperparameters, and have proven to work well in practice. However, neither of them are designed for dealing with categorical predictors, which are important independent variables in our question. Since R has robust package that can lead LDA and QDA work with categorical predictors and model them within a short period of time, we use them as illustrative models for our dataset.

```
Confusion Matrix and Statistics

          Reference
Prediction    no   yes
       no  11612  1092
       yes   393   467

               Accuracy : 0.890519
                 95% CI : (0.8851435, 0.8957264)
    No Information Rate : 0.8850634
    P-Value [Acc > NIR] : 0.02335006

                  Kappa : 0.3314753
 Mcnemar's Test P-Value : < 0.000000000000000222

            Sensitivity : 0.29955099
            Specificity : 0.96726364
         Pos Pred Value : 0.54302326
         Neg Pred Value : 0.91404282
             Prevalence : 0.11493660
         Detection Rate : 0.03442937
   Detection Prevalence : 0.06340313
      Balanced Accuracy : 0.63340732

       'Positive' Class : yes
```

```
Confusion Matrix and Statistics

          Reference
Prediction    no   yes
       no  10682   777
       yes  1323   782

               Accuracy : 0.8451784
                 95% CI : (0.83898, 0.8512283)
    No Information Rate : 0.8850634
    P-Value [Acc > NIR] : 1

                  Kappa : 0.3396469
 Mcnemar's Test P-Value : <0.0000000000000002

            Sensitivity : 0.50160359
            Specificity : 0.88979592
         Pos Pred Value : 0.37149644
         Neg Pred Value : 0.93219304
             Prevalence : 0.11493660
         Detection Rate : 0.05765261
   Detection Prevalence : 0.15519021
      Balanced Accuracy : 0.69569976

       'Positive' Class : yes
```

Figure 2: Confusion Matrix for LDA and QDA

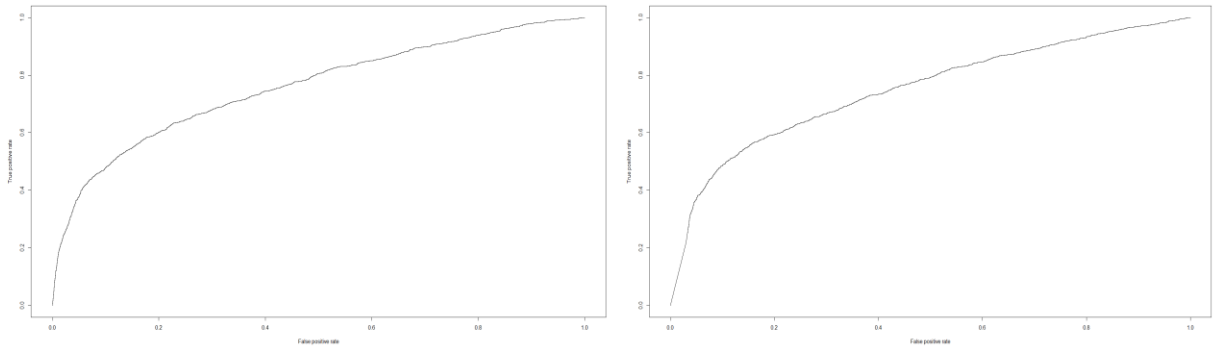


Figure 3: ROC curves for LDA and QDA

From the ROC curves of LDA and QDA, we can find that their overall performances are not very well, since neither of the ROC curve is close to the top left corner. They both have AUC number around 0.75, which also indicate average model performance. To take a closer look at the specificity and sensitivity scores, we can find that QDA does a better job in predicting the actual subscriptions of the deposit (yes). However, higher sensitivity score is in the cost of lower precision rate, which means QDA is less accurate than LDA in making the right prediction regarding clients' subscriptions. We can adjust the threshold rate (change the threshold rate from 0.5 to 0.4 means we believe the client will subscribe the deposit when his subscription probability is greater than 0.4) in our model to get better sensitivity. However, this will cost the overall model accuracy, specificity, and precision to get worse as a result.

Since LDA and QDA are only our illustrative examples, we will discuss more about model improvement, threshold selection, and algorithm advantages/disadvantages for the other algorithms.

2.) Logistic model using lasso regularization

Our second model choice is logistic model with lasso regularization. Logistic model is a powerful linear predicting model that both work for regression and classification. Logistic model uses log odds as the dependent response so that it can generates probability between [0,1] for predicting binary outputs.

$$odds = e\text{^}(\beta_0 + \beta_1 * x_1 + \beta_2 * x_2 + \cdots)$$

Compare to the LDA and QDA models, logistic model can involve categorical predictors and provide a better interpretation about relationship between predictors and response variable.

However, logistic model may suffer from multi-collinearity between its predictors as well as a large number of unnecessary predictors. Therefore, we apply lasso regularization to select important variables form 60+ predictors.

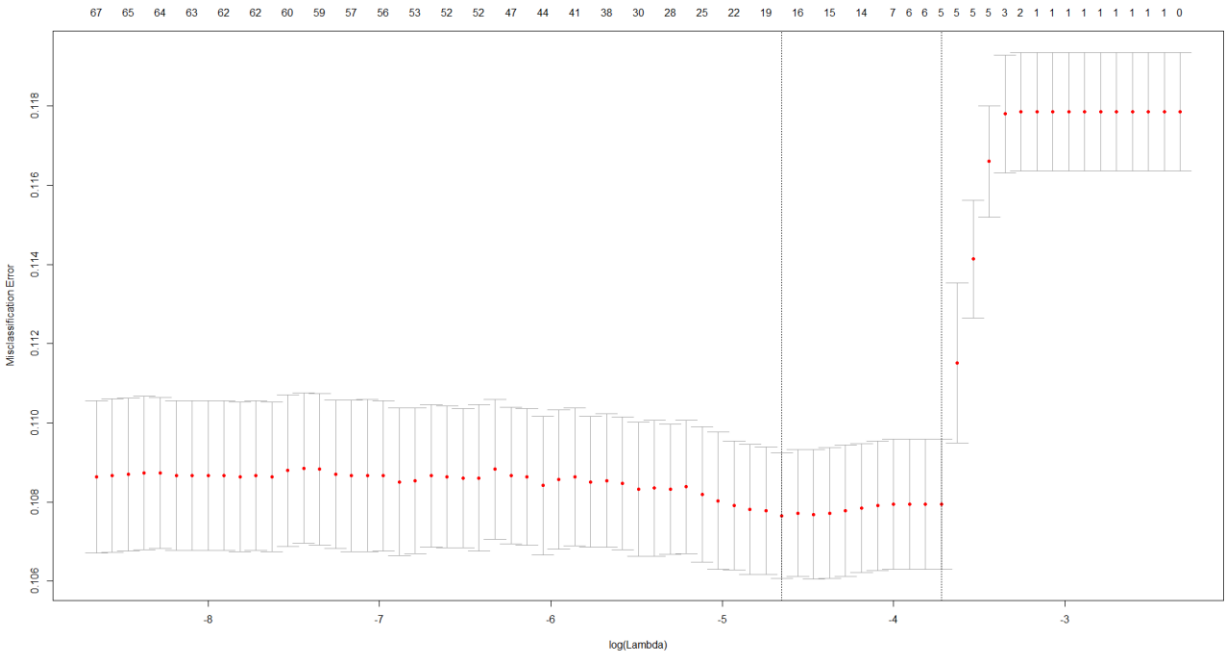


Figure 4: Lasso variable selection

This graph provides two distinguished lambda choices – the lambda with the least misclassification error and the lambda with the most regularized model. In these cases, the lasso variable selection decreases the number of predictors in our model from 67 to 18 and 5.

```
Confusion Matrix and Statistics

          Reference
Prediction    no   yes
       no  11892  1301
       yes   113   258

               Accuracy : 0.8957535
                 95% CI : (0.8904889, 0.9008476)
    No Information Rate : 0.8850634
    P-Value [Acc > NIR] : 0.00004056004

                  Kappa : 0.2334868
 Mcnemar's Test P-Value : < 0.00000000000000022204

            Sensitivity : 0.16549070
            Specificity : 0.99058726
         Pos Pred Value : 0.69541779
         Neg Pred Value : 0.90138710
             Prevalence : 0.11493660
         Detection Rate : 0.01902094
   Detection Prevalence : 0.02735181
      Balanced Accuracy : 0.57803898

       'Positive' Class : yes
```

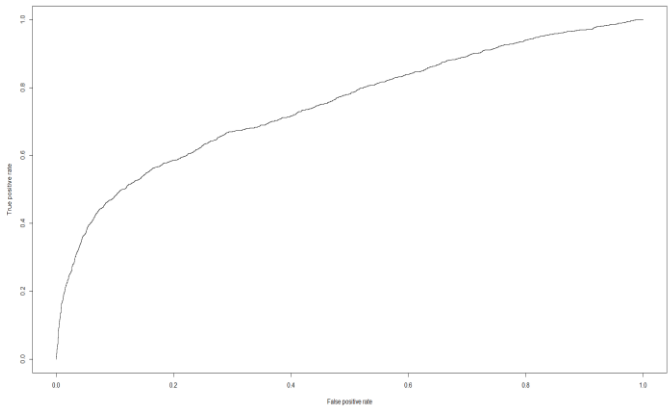


Figure 5: Confusion Matrix and ROC curve for logistic model with 18 predictors

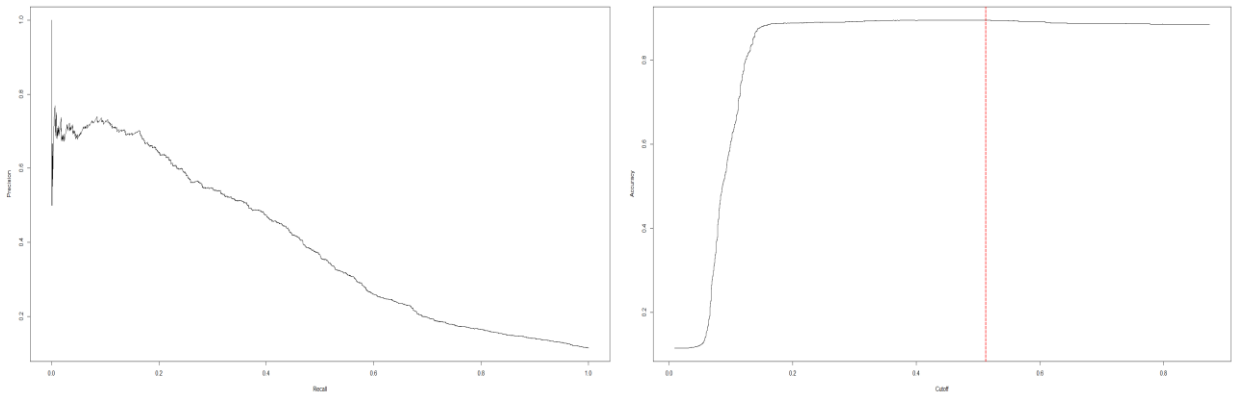


Figure 6: precision vs recall plot and overall accuracy vs threshold plot

From the confusion matrix and the ROC graph, we can find that our logistic model does not perform well in correctly predicting clients' subscriptions. The AUC number is still around 0.74, which also indicates a poor model performance.

Therefore, we consider adjusting threshold level to get better prediction of the actual subscriptions. However, from the precision vs recall (sensitivity) plot, we can find that as we adjust our threshold to make the recall rate get higher, our precision rate goes down very quickly. This fact demonstrates that our model cannot predict better by simply adjusting the threshold level, since our model ends up throwing all the predictions as the client will subscribe and make a huge misclassification error.

What is the best threshold level we can choose to get a higher sensitivity rate while the model remains stable? From the overall accuracy vs threshold plot, we can find that if we adjust our threshold level to around 0.18, we can get a comparatively better model (the red line shows the threshold level 0.513 with most overall accuracy rate).

```
Confusion Matrix and Statistics

          Reference
Prediction    no   yes
       no  11517  1035
       yes   488   524

               Accuracy : 0.8877175
                 95% CI : (0.8822841, 0.8929839)
    No Information Rate : 0.8850634
    P-Value [Acc > NIR] : 0.169686

                  Kappa : 0.3486911
 Mcnemar's Test P-Value : < 0.0000000000000002

            Sensitivity : 0.33611289
            Specificity : 0.95935027
         Pos Pred Value : 0.51778656
         Neg Pred Value : 0.91754302
             Prevalence : 0.11493660
         Detection Rate : 0.03863167
   Detection Prevalence : 0.07460926
      Balanced Accuracy : 0.64773158

       'Positive' Class : yes
```

```
Confusion Matrix and Statistics

          Reference
Prediction    no   yes
       no  11897  1310
       yes   108   249

               Accuracy : 0.8954586
                 95% CI : (0.8901876, 0.9005592)
    No Information Rate : 0.8850634
    P-Value [Acc > NIR] : 0.00006398142

                  Kappa : 0.2267993
 Mcnemar's Test P-Value : < 0.00000000000000022204

            Sensitivity : 0.15971777
            Specificity : 0.99100375
         Pos Pred Value : 0.69747899
         Neg Pred Value : 0.90081018
             Prevalence : 0.11493660
         Detection Rate : 0.01835742
   Detection Prevalence : 0.02631967
      Balanced Accuracy : 0.57536076

       'Positive' Class : yes
```

Figure 7: Confusion Matrix of threshold 0.18 vs 0.513

From figure 7, the confusion matrix for threshold 0.18 clearly provide better results for our purpose compare to the maximum accuracy threshold 0.513. However, the sensitivity rate is still poor, we may consider lack of information in our case.

3.) Random Forest

Random Forest is an improved bootstrap aggregated version of decision tress with randomly selected “p” variables from all predictors in each tree. Therefore, random forest can successfully reduce the correlation between trees in the ensembles. Random forest is a strong machine learning tool that can help us deal with either regression problem or classification problem.

Unlike previous algorithms, random forest requires three tuning parameters before running the model. They are number of trees, number of variables in each tree, and the size of the node at the bottom of the tree. The number of tree affects the overall accuracy of our model, so that we should set a larger number for the accuracy to converge. The number of variables in each tree affects the correlation variance between our predictors, and we use the most popular method – square root of the total number of the predictors to set it. As for the size of the node, we can simply set it to 1, which means we want to have the fully-grown trees instead of the smaller trees.

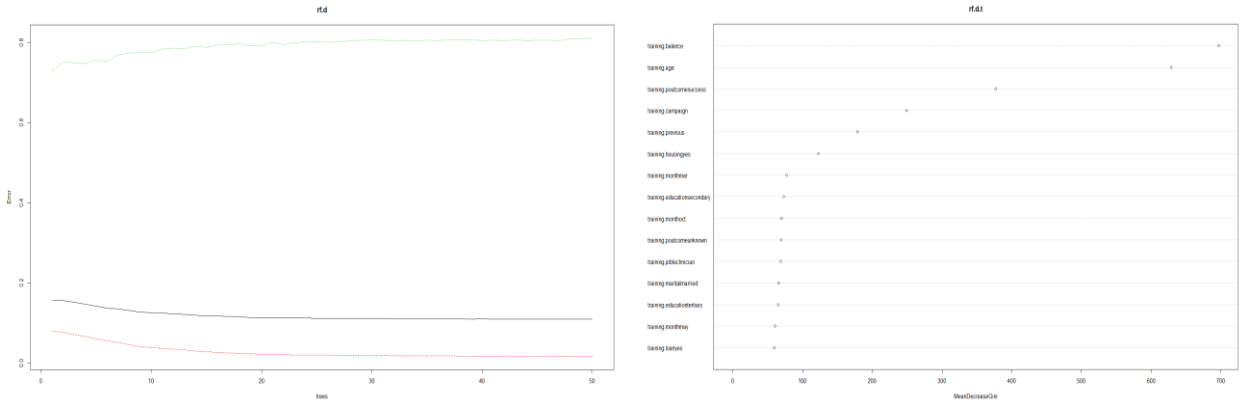


Figure 8: rf error vs number of tree plot and variable importance plot

From figure 8, we can find that our random forest model converges at about 35 trees with an misclassification error around 0.16. The variable imporance graph provides us good information about which variables are determinate for our model. It makes sense that our four continuous variables: balance, age , campaign, and previous are listed as the top5 most important variables.

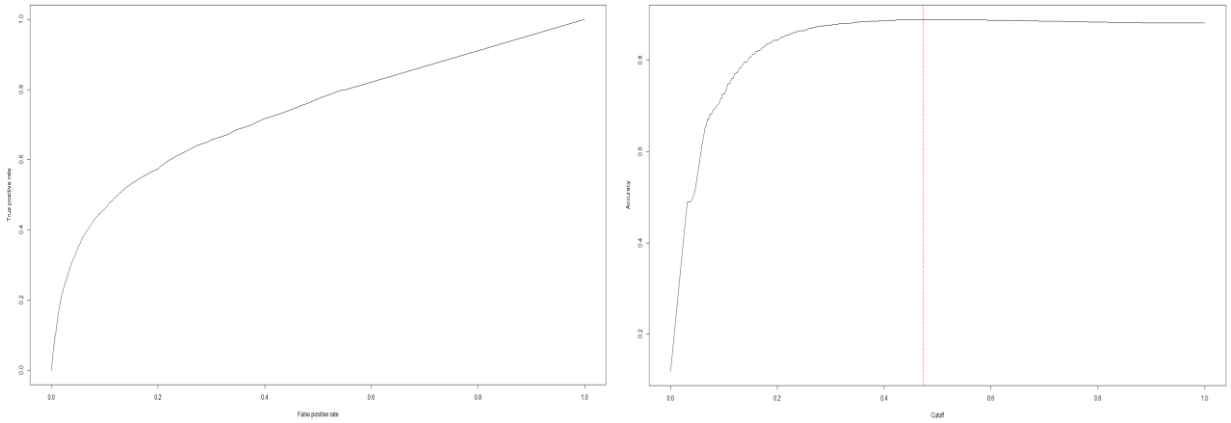


Figure 9: ROC and accuracy vs threshold plots

The ROC plot of random forest does not provide strong evidence that our random forest works well in predicting clients' subscriptions, and the AUC value is around 0.74. However, we can mannully adjust the threshold to about 0.25 to get a comparatively better model from the information provided by accuracy vs threshold plot.

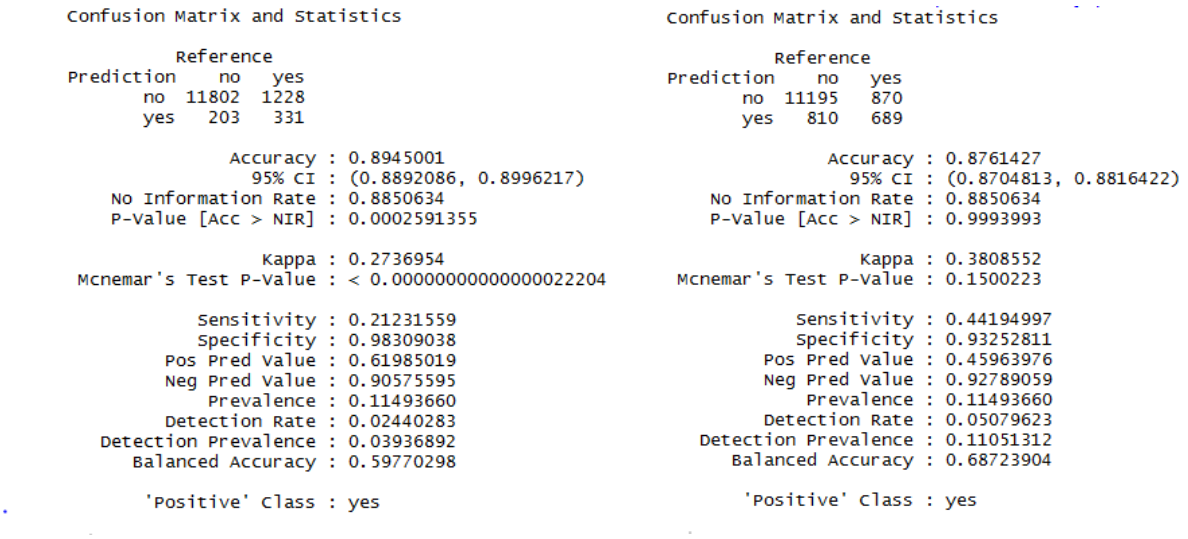


Figure 10: Random Forest Confusion Matrix of threshold 0.5 and threshold 0.25

From figure 10, we can find that by adjusting threshold level from 0.5 to 0.25, we significantly increase the sensitivity rate in a small cost of overall accuracy, specificity, and precision rates.

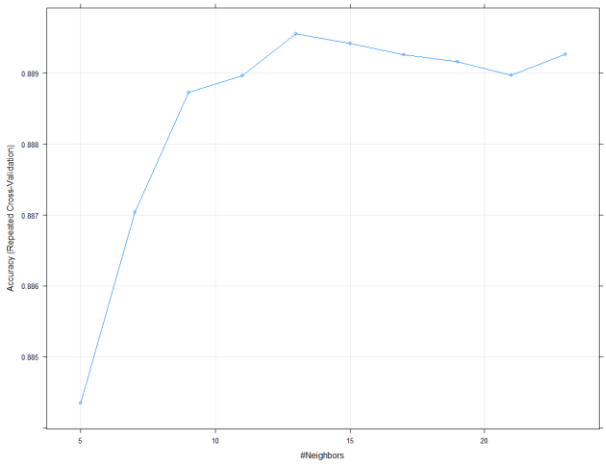
4.) K Nearest Neighbor Algorithm

KNN is a famous nonparametric algorithm to do classification. The principle of KNN classifier algorithm is to find K predefined number of training samples that are closest in the Euclidean distance to a new point and predict the class of the new point using these samples. Compare to regression methods, KNN is more flexible and simpler to set up. However, it requires large computational power and comparatively more time to run. The detailed equation is as below:

$$\Pr(Y = j|X = x_0) = \frac{1}{K}\sum_{i \in N_0} [\![I(y_i]\!] = j)$$

From the equation, we can find that there is only one tuning parameter in KNN algorithm, which is the K – number of nearest samples. In general, as we increase the number of K, the classifier bias will decrease, but the variance will increase. Therefore, it is important for us to choose a good K value. This can be done by cross validation.

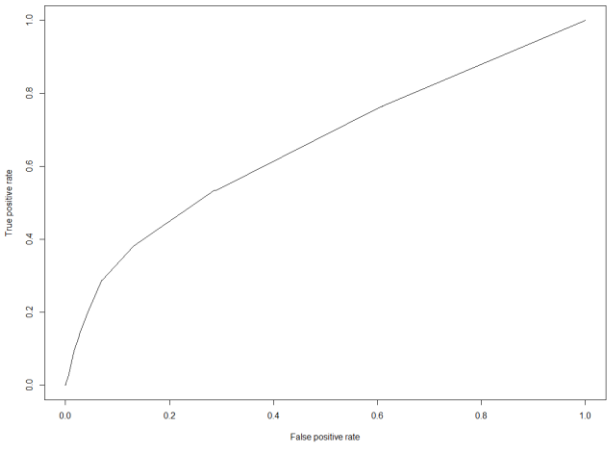
However, we cannot understand the variable importance from our KNN model. In our case, we always want to get more information about both our model and the predictors. Hence KNN method is only considered as a benchmark classifier in our study.




```
k-Nearest Neighbors

31647 samples
   67 predictor
    2 classes: '1', '2'

Pre-processing: centered (67), scaled (67)
Resampling: Cross-Validated (10 fold, repeated 3 times)
Summary of sample sizes: 28482, 28483, 28482, 28482, 28482, 28482, ...
Resampling results across tuning parameters:

  k   Accuracy   Kappa
   5  0.8843493  0.2543350
   7  0.8870351  0.2461415
   9  0.8887309  0.2418704
  11  0.8889627  0.2338521
  13  0.8895525  0.2288254
  15  0.8894156  0.2220871
  17  0.8892576  0.2162974
  19  0.8891628  0.2091017
  21  0.8889732  0.2035790
  23  0.8892681  0.2023811

Accuracy was used to select the optimal model using  the largest value.
The final value used for the model was k = 13.
```

Figure 10: KNN cross validation plot and result

Figure 10 illustrates the result of the best number choice of K performed by the cross validation method. Kappa is another measurement of the model performance. However, in our case, kappa is always greater with smaller value of k. Therefore, we choose the accuracy value as the standard of choosing k. We use 13 as the number of the nearest neighbors. From the plot, we can also find that adjusting the magnitude of k does not affect our model significantly. Sensitivity rate and precision rate are still low, the ROC curve does not look good, and the AUC value is around 0.66.




```
Confusion Matrix and Statistics

          Reference
Prediction    no   yes
       no  11861  1359
       yes   144   200

               Accuracy : 0.8892
                 95% CI : (0.8838, 0.8944)
    No Information Rate : 0.8851
    P-Value [Acc > NIR] : 0.06701

                  Kappa : 0.176
 Mcnemar's Test P-Value : < 2e-16

            Sensitivity : 0.12829
            Specificity : 0.98800
         Pos Pred Value : 0.58140
         Neg Pred Value : 0.89720
             Prevalence : 0.11494
         Detection Rate : 0.01474
   Detection Prevalence : 0.02536
      Balanced Accuracy : 0.55815

       'Positive' Class : yes
```

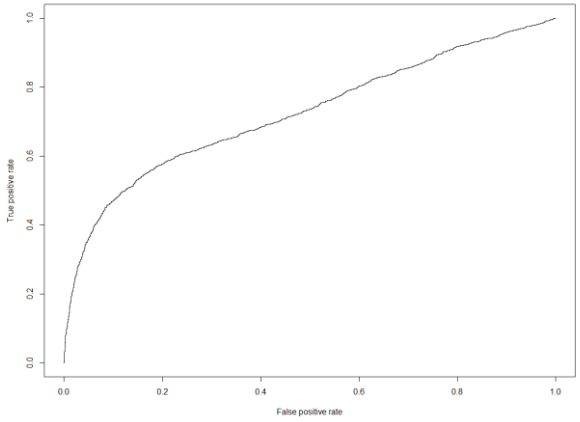
Figure 11: KNN ROC plot and Confusion Matrix Result

5.) Support Vector Machine

Support vector machine is a powerful extension of the support vector classifier, which applies linear hyperplanes to build decision boundary. SVM enlarges the feature space from linear to non-linear by using the kernels, so that we can perform non-linear boundary between the classes.

As for the supervised machine learning algorithm, there are three tuning hyperparameters available to help us improve our model. They are the choice of kernel, C, and gamma. The choice of kernel determines the shape of our boundaries. Since our data is significantly not linear separated, we use the "radial" kernel for the SVM model. C is a parameter controls the cost of misclassifications around the margins. In general, a smaller C leads to a lower cost of misclassification, to a softer margin, to higher bias but lower variance. On the other side, larger C leads to a higher cost of misclassification, to a harder margin, to lower bias but higher variance. Gamma is a tuning parameter of the radial basis function kernel, a larger gamma leads to higher bias but lower variance model, and vice-versa.

Since SVM runs slowly on large cross validation data set, we use 50% of the original data to apply 10-fold cross validation, and find that the best parameters for C and gamma, which are C equals to 0.01 and gamma equals to 0.2.




```
Confusion Matrix and Statistics

          Reference
Prediction    no   yes
       no  11839  1336
       yes   136   253

               Accuracy : 0.8915
                 95% CI : (0.8861, 0.8967)
    No Information Rate : 0.8829
    P-Value [Acc > NIR] : 0.0008418

                  Kappa : 0.2199
 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 0.15922
            Specificity : 0.98864
         Pos Pred Value : 0.65039
         Neg Pred Value : 0.89860
             Prevalence : 0.11715
         Detection Rate : 0.01865
   Detection Prevalence : 0.02868
      Balanced Accuracy : 0.57393

       'Positive' Class : yes
```

Figure 12: ROC graph for SVM model and confusion matrix at 0.5 threshold level

From the ROC graph and the confusion matrix, we can find that our SVM model does not provide better results compare to the others. The AUC is 0.72, which is also lower than the other models. However, the further improvement can be done through adjusting the threshold level to gain higher sensitivity rate

but in cost of lowering model precision or force to model to apply harder margin to decrease level of bias error but in cost of higher variance error.

6.) Neural network

Neural Network is a machine learning framework that mimics the learning process of natural biological neural networks. Basically, it generates several hidden layers to learn from the original inputs, and use these hidden layers to calculate the final output. There could be many perceptrons per layer and multiple layers per neural network model. Besides these perceptrons and layers, neural network also add weights and bias to the final model.

Neural Network is a strong self-learning tool that can usually provide well-performed outputs. However, we must be aware that neural network model does not provide any information about our variables' importance and run as a black box process. We should consider whether we care more about the information of our predictors or simply want good predictions when using neural network.

In our case, we use 10 perceptrons and one hidden layer neural network to build our classifier. The results by cross validation show that increasing the number of perceptrons and hidden layers does not lead to sufficiently better result. Therefore, we just present our model for an illustrative purpose.

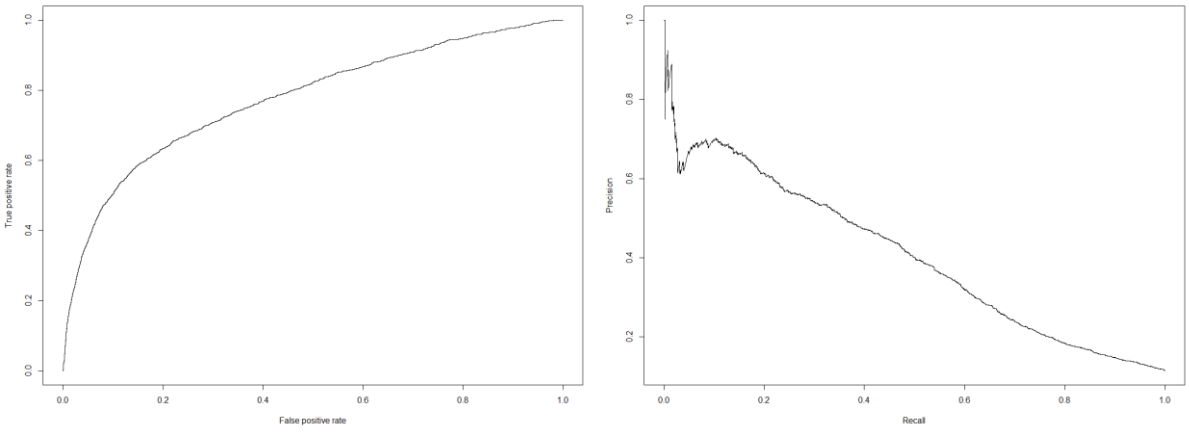
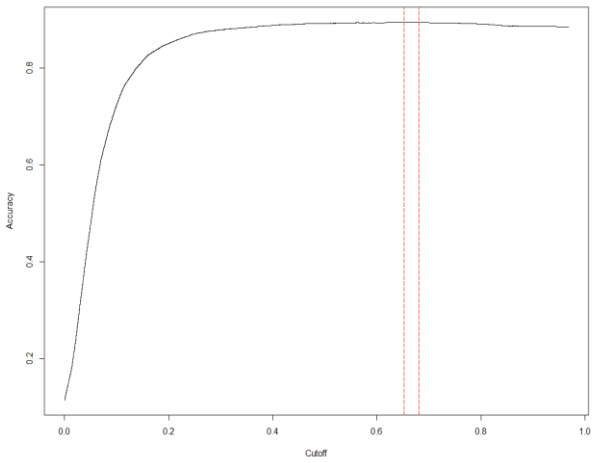


Figure 13: neural network ROC graph and precision vs recall graph

The ROC graph and precision vs recall graph are in the similar shapes as the others we have showed previously. We should look at the accuracy rate graph and try to adjust threshold for better sensitivity rate.




```
Confusion Matrix and Statistics

          Reference
Prediction    no   yes
       no  10730   752
       yes  1275   807

               Accuracy : 0.8506
                 95% CI : (0.8444, 0.8565)
    No Information Rate : 0.8851
    P-Value [Acc > NIR] : 1

                  Kappa : 0.359
 Mcnemar's Test P-Value : <2e-16

            Sensitivity : 0.5176
            Specificity : 0.8938
         Pos Pred Value : 0.3876
         Neg Pred Value : 0.9345
             Prevalence : 0.1149
         Detection Rate : 0.0595
   Detection Prevalence : 0.1535
      Balanced Accuracy : 0.7057

       'Positive' Class : yes
```

Figure 14: neural network accuracy vs cutoff plot and confusion matrix at 0.2 threshold level

From the accuracy vs cutoff graph above, we can find that adjust the threshold level to around 0.2 will not significantly affect the over accuracy rate of our model. The confusion matrix of neural network model at 0.2 threshold level provides higher sensitivity but lower precision rate. Based on that information, we find out that simply adjusting threshold level cannot help our model predict better but only predict more subscription cases. Therefore, the precision rate gets lower.

Conclusion:

After modeling with logistic, knn, random forest classifier, support vector machines, neural network, we have found that neural network and svm can give better results when giving large computational power and properly tuned hyperparameters, but they do not provide much information about the predictors; KNN is the simplest algorithm to run without any pre-requests but it also requires great computational power and provides almost zero information about the predictors; whereas random forest and logistic classifiers are easier to tune, provide fair results, and give rich information about the variable importance. Since all our models' AUC scores are around 75% with the highest close to 80% and the lowest close to 72%, the predictive accuracies of our models are not significantly different. Therefore, we would like to choose the model that provides the best interpretations of our predictors. Random forest

classifier not only provides good AUC score which is around 74%, but also informs us the level of importance of our predictors. Its result indicates that both account balance and customer age are influential factors for subscribing the term deposit. Whereas, job type, marriage status, and date of the campaign are least important predictors. Thus, we can use those information to either filter out the un-subscribe group or predict customers that are more likely to subscribe.

Reference:

James, G., Witten, D., Hastie, T., & Tibshirani, R. (n.d.). An introduction to statistical learning: with applications in R.

Hastie, T., Tibshirani, R., & Friedman, J. H. (2017). The elements of statistical learning: data mining, inference, and prediction. New York, NY: Springer.