PREDICTING TERM DEPOSIT TAKERS USING CLASSIFICATION

GROUP 7

- **Annapurna Madireddy**
- **Pavan Reddy**
- **Ravi Bhagwat**
- **Saniya Khan**

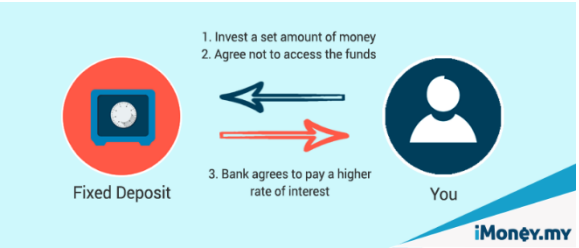
Summary

Majority of the bank offers accounts where the owner can withdraw or deposit the money at any time. This makes it difficult for the banks to plan ahead of time about their lending power. To deal with it, banks introduced term deposit accounts where the money will be locked with the bank for a certain period of time. This gave the bank the flexibility to lend money forward. However, one of the major challenges was to identify customers who would be interested in subscribing for the term deposit.

In this report, we will introduce you with a dataset related to the direct marketing campaigns of a Portuguese banking institute. We cleaned and pre-processed the data to prepare it to build a model. Then, we used Random forest and Support Vector Machine algorithms to build model and compare the results.

Introduction

Term deposit gives a certain lending power to the banks so that they can lend money on higher rates as loans to earn revenue and profits. If we have to define term deposit, then the following image does it nicely.



Banks always wants customers to avail the term deposit service for which they have to market it and promote it which costs money. These marketing campaigns can be successful or a failure depending upon whom you target. The customer may not be interested in term deposit at all and still the campaign targets him/her, is a waste of money. In order to minimize the costs and target customers who have high chances of subscribing to the service, we build a predictive model that will classify customers into high potential and low potential. We have a dataset of direct marketing campaigns run by a Portuguese banking institute. The marketing campaigns were based on phone calls. Often, more than one contact to the same client was required, in order to access if the product would be ('yes') or not ('no') subscribed. There are 41,188 observations of 20 variables, with one more dependent variable *y*.

Data

The observations in the dataset were recorded between May 2008 and November 2010 from multiple marketing campaigns that the institute did. We will be using the whole dataset for model building and no sampling will be done. The variables are described below:

bank client data

- *age* (numeric)
- *job* : type of job (categorical)
- *marital* : marital (categorical)
- *education* (categorical)
- *default*: has credit in default (categorical)
- *housing*: has housing loan? (categorical)
- *loan*: has personal loan? (categorical)

related with the last contact of the current campaign:

- *contact*: contact communication type (categorical)
- *month*: last contact month of year (categorical)
- *day_of_week*: last contact day of the week (categorical)
- *duration*: last contact duration, in seconds (numeric). Important note: this attribute highly affects the output target (e.g., if duration=0 then y='no'). Yet, the duration is not known before a call is

performed. Also, after the end of the call y is obviously known. Thus, this input should only be included for benchmark purposes and should be discarded if the intention is to have a realistic predictive model. (numeric)

other attributes:

- *campaign*: number of contacts performed during this campaign and for this client (numeric)
- *pdays*: number of days that passed by after the client was last contacted from a previous campaign (numeric)
- *previous*: number of contacts performed before this campaign and for this client (numeric)
- *poutcome*: outcome of the previous marketing campaign (categorical)

social and economic context attributes

- *emp.var.rate*: employment variation rate - quarterly indicator (numeric)
- *cons.price.idx*: consumer price index - monthly indicator (numeric)
- *cons.conf.idx*: consumer confidence index - monthly indicator (numeric)
- *euribor3m*: euribor 3 month rate - daily indicator (numeric)
- *nr.employed*: number of employees - quarterly indicator (numeric)

Output variable (desired target):

- *y* - has the client subscribed a term deposit? (binary: 'yes','no')

Data Preprocessing

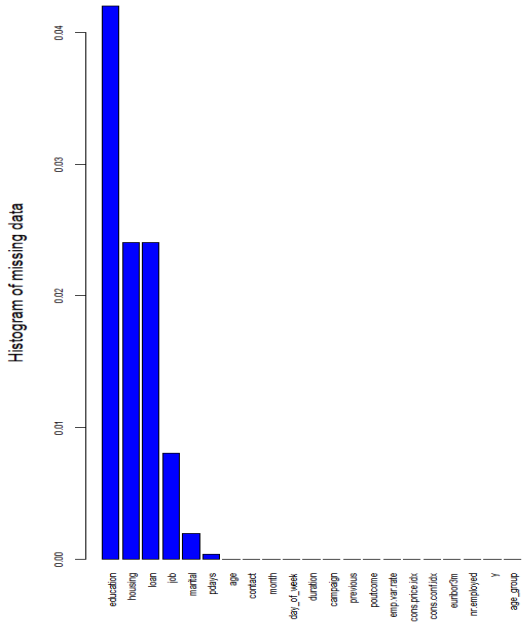
We start by running some descriptive statistics on the data. We ran code in R to find the dimension, structure, and the summary of the data. After running summary, we found out that there are some variables which have 'unknown' values, namely – *marital*, *default*, *housing*, and *loan*. Our group decided to make an assumption that these values should be treated as missing value as we could not find an explanation for it. We converted these 'unknown' values in NA's so that we can later treat them as missing values.

Using the summary data, we removed the *default* variable as it has more than 99% same observations. This bias in the variable could have affect during model building. We also transformed some variables for simplifying the data. Below is one example of how we transformed the *y* variable.

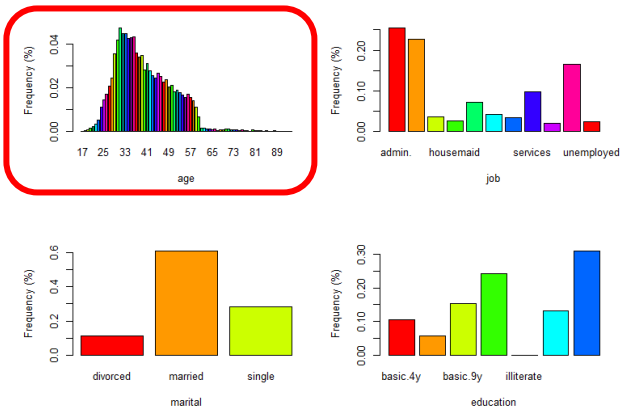
```
> bankdata$y<-ifelse(bankdata$y =='yes', 1,0)
> bankdata$y<-as.factor(bankdata$y)
> table(bankdata$y)

    0     1
36548  4640
```

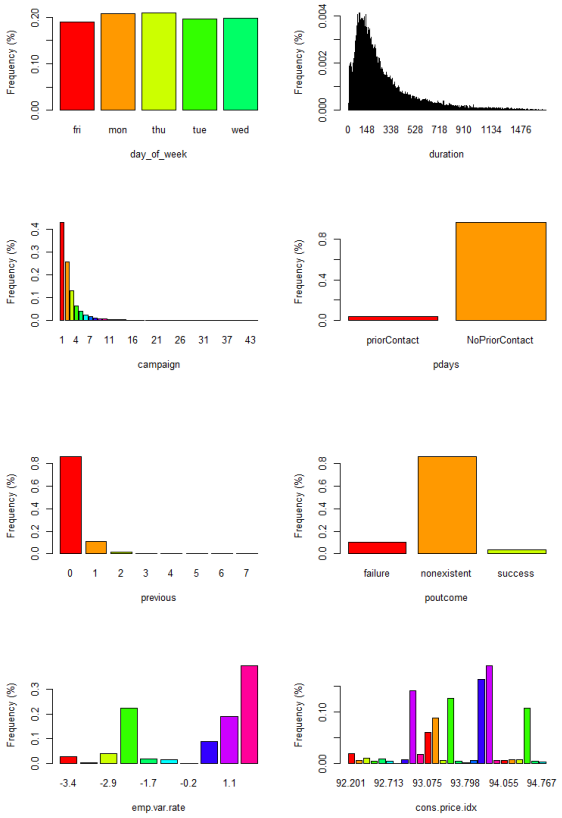
Another small yet essential step is to remove duplicate data. We figured there were 14 duplicate rows to be removed. Next step was to deal with missing data. In our dataset, there were 0 rows which had completely missing values but there were 2957 rows where some data was missing. Below is a histogram showing the percentage of missing values in each variable. Education topped the list with only 6 variables having missing values.

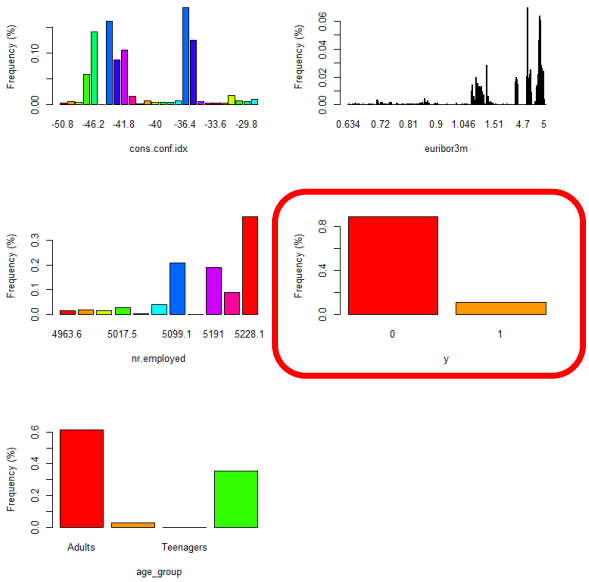


To learn more about the data, we did some exploration by finding the frequency range of the variables. Here, are some of the examples.



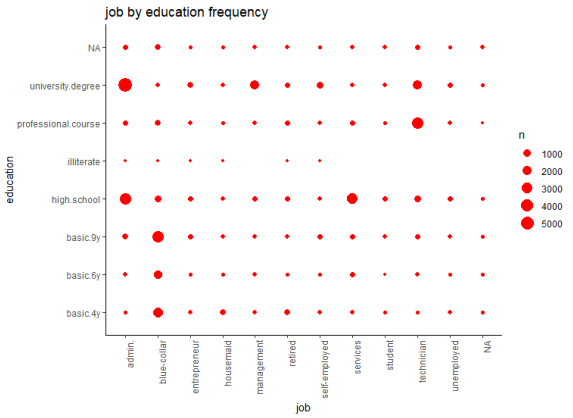
For the variable age, which is highlighted in red we can see that people between the age 30 and 40 have the highest frequency. There are very less people below the age 25 and same goes for age above 60.





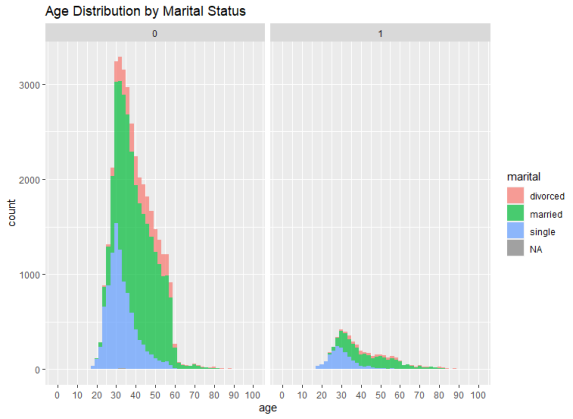
The graph for our dependent variable *y*, again highlighted in red, indicates of the dataset being imbalanced and people not subscribed to term deposit make the majority class.

The frequency plots tell us a lot about individual variables but to understand relationship between variables we plotted the below graphs.



From the plot, we can say that admin jobs are mostly taken by people who have a university degree and next by people who graduated high-school. Similarly, the technician jobs are taken up by people who have done a professional course or have a university degree.

The distribution of age group, marital status that subscribes to term deposit is plotted.



While we have imbalanced data, the age and marital status have similar graphs if scaled with divorced people having the highest numbers in both the classes, followed by married and last singles. Also, most customers who subscribe to term deposit, range between 25 to 45.

The dataset had observations from 6 different campaigns, so we wanted to know how many people subscribed to term deposits in each of them.

Campaign	People contacted	People subscribing to term deposit
1	17632	13%
2	10568	11.5%
3	5340	10.7%
4	2650	9.4%
5	1599	7.5%
6	979	7.66%

The percentage of people subscribing to term deposit has gradually decreased probably because they don't want to be contacted again and again. The number of direct contact can be limited to 3 which gives us more than 10% success in subscription.

This dataset also had missing data, hence, to treat this we used the MICE library to impute the data.

Model Building

For model building, we chose to do two supervised algorithms – **Support Vector Machines** (SVM) and **Decision Trees**. We selected SVM because it handles non-linear data really well using its Kernel trick and generally considered to be stable if there is change in data. Decision trees are considered to be intuitive and easy to explain and considered to be one of the best predictive models.

We split our data in 80/20 split but before starting the modelling process, we knew we had imbalanced data, so, to make it balanced we used the Synthetic minority oversampling technique (SMOTE) method, which is an over-sampling technique that creates artificial data for the minority class based on nearest neighbors.

Support Vector Machine Model

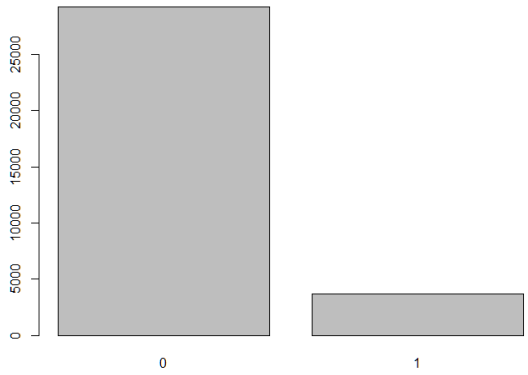
Support vector machines offer a way for binary classification by trying to separate classes using a hyperplane in feature space. Although it is difficult to find such a hyperplane for most datasets, SVM achieves this by using 2 methods:

- Using a soft margin : In this we allow some points (which maybe outliers for the class) to be included on wrong side of the margin and tuning the hyperparameter C which is the allowable cost for misclassification. Tuning is done via k-fold cross validation.
- Employing the kernel trick : Data sometimes might be no-separable in the original feature space but when SVM uses kernel trick to enlarge the feature space, a hyperplane maybe found to separate the classes.

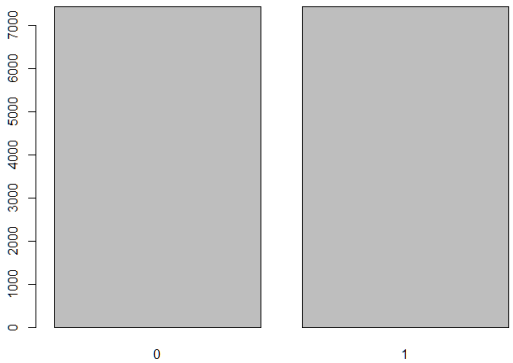
Prerequisites

After the general EDA and data cleaning tasks, SVM requires some preprocessing to be done to the data to construct a model.

- Irrelevant and Redundant Variables are handled by SVM hence we have not done any feature selection
- Missing rows are imputed in previous steps.
- Scaling is done by the SVM algorithm itself.
- One hot encoding is done for categorical variables.
- Data is divided -- 80% training set, 20% to testing set
- Data is balanced using SMOTE, this is done to improve performance i.e. detecting the minority class 1 (which corresponds to "Yes"). The parameters perc.over and perc.under drives the oversampling of minority class & under sampling of the majority class. Perc.under argument controls the fraction of cases of the majority class that will be randomly selected for the final "balanced" data set.



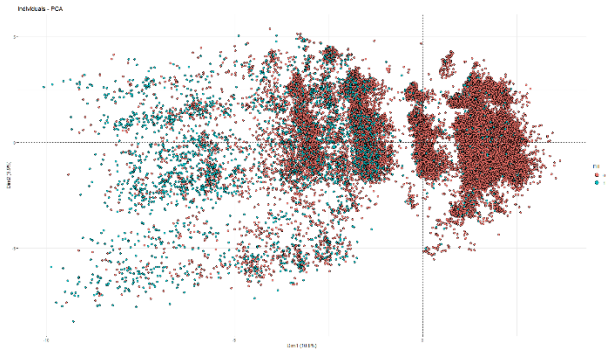
Distribution of class variable before Balancing



1Distribution of class variable after SMOTE Balancing

SVM Process:

- As we have about 49 columns after creation of dummy values, to visualize the data in 2 d space we have used PCA. As we can see from the plot below a lot of observations corresponding to "No" (0) is clustered towards the right side of the plot and "Yes" on the left but there are a huge number of observations which are not separable in 2-D space. Selecting an appropriate hyperplane seems difficult in 2-D space.



- Kernel selection : We use SVM function from the package e1071 and fit the balanced training data to different types of kernels – Linear, Radial, Polynomial and Sigmoid and evaluate performance. Accuracy measures for the different kernels are :

Kernel	Accuracy
Linear	0.887325
Radial	0.917565
Polynomial	0.928408
Sigmoid	0.817147

- As the best performance is on polynomial kernel, we perform hyperparameter tuning of the C and degree parameters using 10 fold cross validation and grid search. Results are shown below:

```
Parameter tuning of 'svm':

- sampling method: 10-fold cross validation

- best parameters:
 degree cost
      5    8

- best performance: 0.09240341

- Detailed performance results:
   degree cost      error  dispersion
1       3  0.5 0.10890395 0.008309430
2       4  0.5 0.13968277 0.009251837
3       5  0.5 0.19120495 0.013218399
4       3  1.0 0.10459319 0.008218210
5       4  1.0 0.11510060 0.010291433
6       5  1.0 0.12803171 0.010516710
7       3  2.0 0.10109065 0.009783082
8       4  2.0 0.10499796 0.009449785
9       5  2.0 0.10722072 0.010534347
10      3  4.0 0.09846411 0.008923936
11      4  4.0 0.09785891 0.009274553
12      5  4.0 0.09785869 0.011046881
13      3  8.0 0.09711704 0.011376054
14      4  8.0 0.09314366 0.009553952
15      5  8.0 0.09240341 0.011821568
```

As we can see the best model is the one with degree 5 and cost 8.

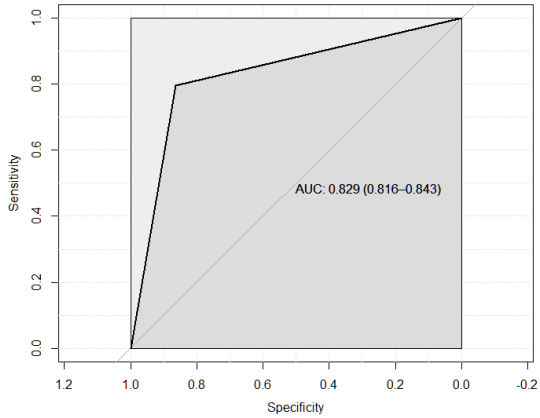
- We use this model to evaluate the performance on complete train data set and we get very high values for accuracy, kappa, Recall and Precision. It almost looks too good to be true with accuracy of 97%, Kappa of 94%, F1 of 97% and precision at 96%.

- We use this model to check its performance on test dataset, as we can see accuracy and recall reduce to some extent but the value for kappa , precision and F1 have drastically fallen down, showing that the model has some degree of overfitting. Accuracy stands at 85.6% and Kappa at low 0.48. Other measures we also pretty low.

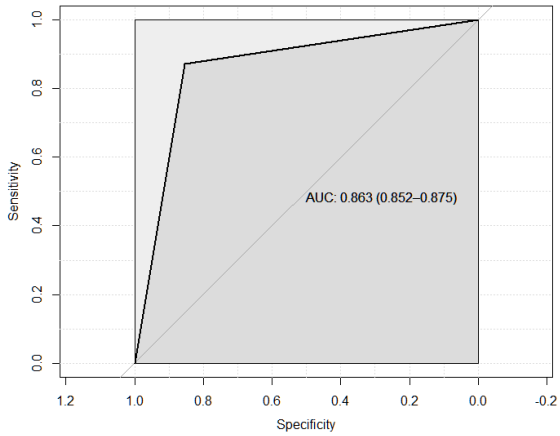
- We evaluate the test performance on the non-cross validated polynomial SVM , which was giving the best performance earlier. Recall has increased to 87.16% and the kappa value has increased indicating moderate agreement.

Model Performance :

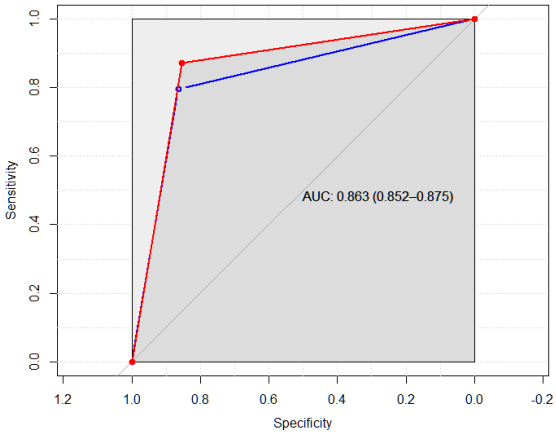
Performance on cross validated model on test data (degree=5,c=8) :



Performance on original Polynomial SVM model on test data (degree=3,c=1) :



The latter has a higher AUC on test data; hence it is the final model:



KERNEL	POLYNOMIAL
DEGREE	3
COST	1
TRAIN ACCURACY	92.84%
TRAIN KAPPA	85.68%
TRAIN PRECISION	90.84%
TRAIN RECALL	93.01%
TEST ACCURACY	85.71%
TEST KAPPA	50.40%
TEST PRECISION	43.33%
TEST RECALL	87.10%
TEST F1	57.85%
TEST AUC	86.30%

```
Confusion Matrix and Statistics

          Reference
Prediction     0     1
         0 31502   346
         1  5033  4293

               Accuracy : 0.8694
                 95% CI : (0.8661, 0.8726)
    No Information Rate : 0.8873
    P-Value [Acc > NIR] : 1

                  Kappa : 0.5466

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

            Sensitivity : 0.9254
            Specificity : 0.8622
         Pos Pred Value : 0.4603
         Neg Pred Value : 0.9891
              Precision : 0.4603
                 Recall : 0.9254
                     F1 : 0.6148
             Prevalence : 0.1127
         Detection Rate : 0.1043
   Detection Prevalence : 0.2265
      Balanced Accuracy : 0.8938

       'Positive' Class : 1
```

Our goal is to identify people who will subscribe to the term deposit i.e. we are aiming for a high recall. The final model has a high True positive rate (TPR) of 87.16% and low false positive rate (FPR) of 14.5%. The value for precision is low due to the fact the positive class(1 or "Yes") is in minority and even a small number of false positives overwhelm the true positive and skew the precision calculation. Hence as evidenced by the AUC values, we can say that this model does a good job at identifying potential people who would subscribe to the term deposit.

Decision Tree Model

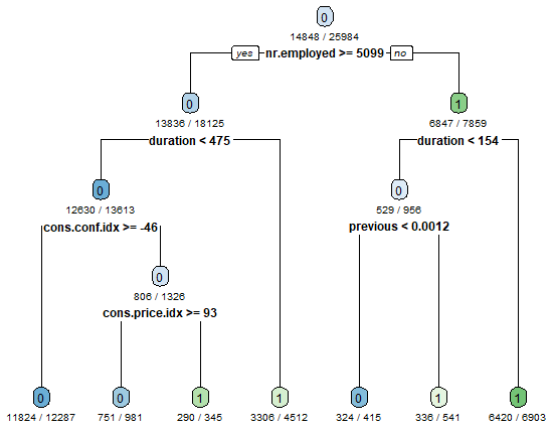
Everyone is familiar with a tree which makes is easier to explain. They are like flow charts where each node has branch node connecting leaf node which is the decision taken after computation.

A decision tree has the following components:

- Root Node: represents the entire population
- Splitting: dividing a node into two or more sub-nodes
- Decision Node: when sub-nodes splits into two or more sub-nodes
- Leaf/Terminal Node: when a node does not split further

Decision tree uses two hyperparameter tuning techniques – grid search and random search. These methods are used to select the complexity parameter associated with the smallest cross-validated error.

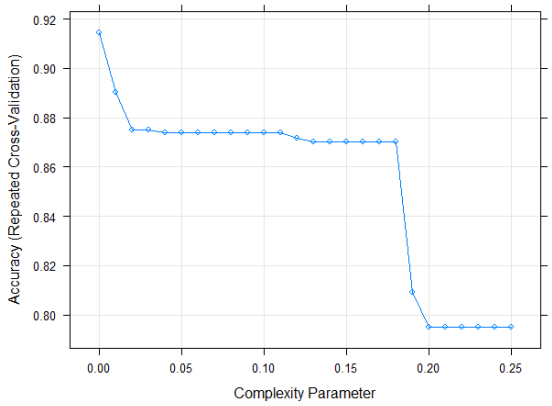
First, we build a basic classification tree without doing hyperparameter tuning which gave is 89.5% accuracy.



The tree says that if *nr.employed* is greater than 5099, then it is one of the factors of customers to subscribe to term deposit. Also, if the duration of contact is less than 154 seconds then they classify as 1, however, if it is more than 154 seconds then there is one more decision node for variable *previous*, which says if its less than 0.0012 then user will subscribe or won't.

Grid search

In grid search, hyperparameters are set in advance of the training and control aspects of the model implementation. Using these hyperparameter values, model is trained, and scoring is done on the test data for each combination. Every combination of hyperparameter values are tried in this method which can be very inefficient. We use 10-fold cross validation on the train dataset.



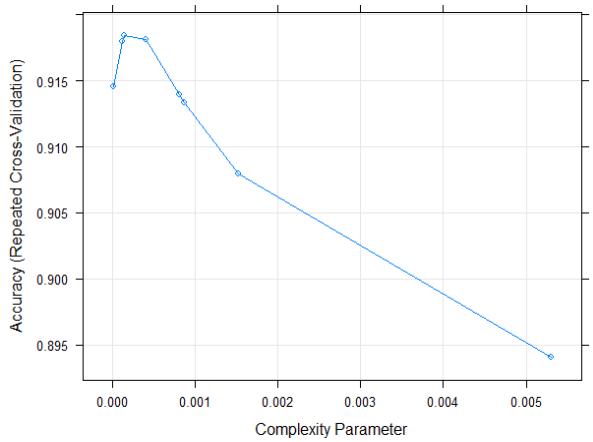
The above graph shows that all the values of the hyperparameter has been used from 0 to 0.25. We also found out that variables *duration, previous, nr.employed, euribor3m, emp.var.rate* are the top important variables. This lets the company acquire key insights about where to focus on.

Using the hyperparameter tuning grid search method we achieved better accuracy of 95%, with Kappa value at 89.7%, sensitivity and specificity at 95.6% and 94% respectively.

Random search

Random Search sets up a grid of values for hyperparameters and chooses random combinations to train the model and score. It helps you to control directly how many combinations of parameters are being attempted.

We will be again using 10-fold cross validation but specify the search to be random.



The above graph shows that unlike grid search, random search is not using all the hyperparameter values in the range. Hence, it is considered one of the best in parameter search techniques.

Using random search also we got similar results for important variables and there is no significant difference. This strengthens the fact that those variables are important. The confusion matrix for random search model gave us an accuracy of 94.4%, sensitivity of 94.5% and specificity of 94.3%.

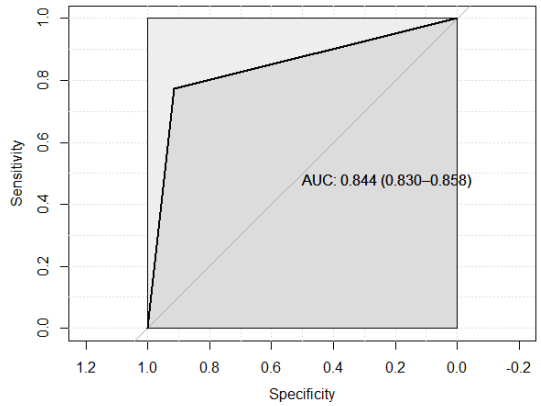
Although, both the models gave comparable results, we leaned towards random search because it has a bit better sensitivity which is our goal. Sensitivity is basically the number of people who have high chance of subscribing to the term deposit service.

Since, we decided that we would be going with grid search, we make predictions for test data using Grid search model. We can say that our model has achieved an accuracy of 89.86 % with 77.35% sensitivity. This tells us that though our model has performed well without overfitting or underfitting, we haven't achieved what is desired i.e, sensitivity.

From the above parameters, since there is a drastic decrease in sensitivity from train to test, it indicates that the model has performed poorly.

	train	test
Accuracy	9.441195e-01	8.985912e-01
Kappa	8.861457e-01	5.754590e-01
AccuracyLower	9.412574e-01	8.918686e-01
AccuracyUpper	9.468817e-01	9.050303e-01
AccuracyNull	5.714286e-01	8.874180e-01
AccuracyPValue	0.000000e+00	6.114115e-04
McnemarPValue	1.566933e-06	1.482639e-46

	train	test
Sensitivity	0.9430675	0.77346278
Specificity	0.9449084	0.91446558
Pos Pred Value	0.9277385	0.53427720
Neg Pred Value	0.9567649	0.96952989
Precision	0.9277385	0.53427720
Recall	0.9430675	0.77346278
F1	0.9353402	0.63199647
Prevalence	0.4285714	0.11258198
Detection Rate	0.4041718	0.08707797
Detection Prevalence	0.4356527	0.16298275
Balanced Accuracy	0.9439880	0.84396418



From the above parameters, while there is a decrease in specificity from train to study, because there is not much shift in sensitivity, we can conclude that we can go ahead with this model.

Model Interpretation

Based on the important features derived from decision tree, we can say that:

Duration: It is the length the client talked on the call for. So, if a person is talking for longer, he / she might be asking questions about the subscription that indicates they might be interested in it. Therefore, we can say it's a very important variable.

Previous: It is number of contacts performed before this campaign and for this client (numeric) and this is the second most important variable. If marketing individuals have already approached a certain person several times, it indicates that they have received positive response from that person and should also be able to approach that person again.

nr.employed: number of employees - quarterly indicator (numeric). This indicates that higher the number of employees, higher the chances of client subscribing to a term deposit. Hence, if the marketing department is more focused on companies with higher number of employees, their chances of having the client subscribed would increase.

In this way, we will be able to make inferences in the real world, using the machine learning techniques as we have done in the above. It would also sometimes help us notice few observations that would otherwise be undetected.

Comparing SVM and Decision tree

Although, decision tree models are more interpretable, from our analysis we found SVM to be performing better as the true positive rate is better with SVM which is the sensitivity.

Conclusion

Predictive modeling using SVM or decision trees can help for classification problem. Banks can utilize the information to cut down on their costs of marketing costs and target customers who are more willing to subscribe to the term deposit. This type of modeling can help other institutions as well.

Citations

Brid, Rajesh S. “Decision Trees - A Simple Way to Visualize a Decision.” Medium, GreyAtom, 26 Oct. 2018, medium.com/greyatom/decision-trees-a-simple-way-to-visualize-a-decision-dc506a403aeb.

Abdulazeez, Tajudeen. Bank Marketing Association Rule Mining. 5 Feb. 2019, www.kaggle.com/toraaglobal/bank-marketing-association-rule-mining/notebook.

Keh Soon, Yong. “Portuguese Bank Marketing Data.” RPubs, 2018, rpubs.com/shienlong/wqd7004_RRookie.

Sawant, Punit. “Sign In.” RPubs, 2018, rpubs.com/Punit_Sawant/275808.