SAN DIEGO STATE UNIVERSITY

MIS-749 FINAL PROJECT

Bank Marketing Campaign Analysis for Account Subscription

By,

Shruti Venkatesh Kulkarni
824677549

1. Executive summary

This assessment aims to develop analytical models in order to solve the business problem. The problem is detailed as trying to predict the outcome of customers of a bank subscribing to a deposit account when certain demographic, financial and historical marketing data of the customer is known.

The document is segmented into 3 major sections. The initial section details the exploratory data analysis performed on the dataset that consisted of 45211 observations over 17 variables. This section also includes visualizations that show predictors as a function of another. Visualizations detailed in this section provide insight into variables that have a correlation, demonstrate variation in values for some and contrast the overall rate against one another. It explains the data preparation that was carried out in order to wrangle the initial dataset into one that can be utilized effectively to train the developed models. As part of this step, predictors such as duration and pdays were deemed to be not useful based on domain knowledge and insights from visualizing correlations. As a result, they have been dropped from the dataset. This section also explains the creation of training and test dataset to validate model's results and measure its performance. Methods like SMOTE were applied to correct the data imbalance that was observed.

The second section outlines the process of building predictive models. Decision Trees, Random Forest, Logistic Regression and Boosting methods were selected as good candidates to compare and contrast for this classification problem. A 10-fold cross validation method was incorporated for all models to get a better estimate of test error and avoid over-fitting. Model performance is outlined and key metrics that are relevant to our business case such as sensitivity

(TPR) and accuracy is compared across models. ROC values are also compared to determine the best performing model.

The next section deals with predicting the dependent variable on the test dataset. All of the models that were developed are used to predict the deposit outcome on the same test dataset. Confusion matrices for these predictions are detailed and thus enabled us to select the best performing model using the sensitivity (TPR) and the accuracy metric. Specificity loss due to having the training dataset balanced is described and found to be an acceptable tradeoff that helped in increasing the sensitivity of the predictions. Random forest model was the best performing model with a sensitivity of 0.54 and an accuracy of 0.84.

The final section summarizes the findings of the assessment. Based on the training set, random forest and boosting model fared better. Considering the results from the prediction, logistic regression and random forest model had the top 2 highest sensitivity rates. ROC values were plotted for all four predictions and helps understand the model performance better. Insights that was gathered based on the models point towards a set of important predictors. Recommendations are made towards utilizing a better tune length given better computational capabilities. Overall, the analytical models were able to satisfactorily answer the business question within the limits on computational resources and available predictors.

2. Discovery and Data Preparation

2.1 Business case for selecting data

Banks rely on targeted marketing campaigns to attract and retain customers, sell products and services and grow the overall business. An analytical approach towards their marketing campaigns helps them lower costs, increase their chance of success and will also benefit customers by way of targeting only those that are interested. The business case around selecting this dataset is to help the bank make targeted telemarketing campaigns with a higher chance of success as compared to targeting customers at random.

This dataset is contains real-time customer information related to direct marketing campaigns (phone calls) of a banking institution. The dataset consists of variables around the previous marketing campaigns, customer demographic information, their bank account details and timelines and results around any previous campaigns.

Data link: https://archive.ics.uci.edu/ml/machine-learning-databases/00222/

We form our hypothesis and success criteria as follows:

Null Hypothesis: There is no relation between the predictors and the dependent variable.

Alternative Hypothesis: There is a relationship between the predictors and dependent variable.

Success criteria: Using this dataset, we aim to predict if customers contacted via a telemarketing campaign will subscribe for a term deposit in the future.

2.2 Count and column explanations

The dataset contains 45211 observations in total and 17 variables, of which 16 are predictors and the dependent variable is “y” in the dataset. Following Table 1 outlines the data structure of the dataset:

VARIABLE NAME	EXPLANATION	DATATYPE	COUNT
AGE	Age of the customer	numeric	425211
JOB	Job type of the customer	Categorical, Factor with 12 levels:	Admin: 5171 blue-collar: 9732

		Admin, blue-collar, entrepreneur, housemaid, management, retired, self-employed, services, student, technician, unemployed, unknown	entrepreneur: 1487 housemaid: 1240 management: 9458 retired: 2264 self-employed: 1579 services: 4154 student: 938 technician: 7597 unemployed: 1303 unknown: 288
MARITAL	Marital status of the customer	Categorical Factor with 3 levels: divorced, married, single	Divorced: 5207 Married: 27214 Single: 12790
EDUCATION	The education obtained by the customer	Categorical, Factor with 4 levels: primary, secondary, tertiary, unknown	Primary: 6851 Secondary: 23202 Tertiary: 13301 Unknown: 1857
DEFAULT	Does the customer have a credit in default?	Categorical, Factor with 2 levels: no, yes	No: 44396 Yes: 815
BALANCE	average yearly balance, in euros	numeric	45211
HOUSING	Does the customer have a housing loan?	Categorical, Factor with 2 levels: no, yes	No: 20081 yes: 25130
LOAN	Does the customer have a personal loan?	Categorical, Factor with 2 levels: no, yes	No: 37967 yes: 7244

CONTACT	The mode of communication type used for the customer	Categorical, Factor with 3 levels: Cellular, telephone, unknown	Cellular: 29285 telephone: 2906 unknown: 13020
DAY	Customer contacted day (date) of the month	numeric	45211
MONTH	Customer contacted month of the year	Categorical, Factor with 12 levels: Jan, Feb, March, Apr, May, Jun, July, Aug, Sep, Oct, Nov, Dec	Apr: 2932 Aug: 6247 Dec: 214 Feb: 2649 Jan: 1403 Jul: 6895 Jun: 5341 Mar: 477 May: 13766 Nov: 3970 Oct: 738 Sep: 579
DURATION	last contact duration, in seconds	numeric	45211
CAMPAIGN	number of times the customer was contacted last time during the campaign	numeric	45211
PDAYS	number of days that have passed after the customer was last contacted from a previous campaign (-1 means client was not previously contacted)	numeric	45211
POUTCOME	outcome of the previous marketing campaign w.r.t to the customer	Categorical, Factor with 4 levels: Failure, other, success, unknown	Failure: 4901 Other: 1840 success: 1511

			unknown: 36959
Y	has the customer subscribed a term deposit?	Categorical, Factor with 2 levels: Yes, no	No: 39922 yes: 5289

Table 1 – Variable definitions in the dataset

A lot of the variables in the dataset are categorial, therefore, our insights from descriptive statistics is limited. The data did not have any missing or null values, and as a result, there was no need to perform any imputations or data removal. For the purpose of this analysis and report, we will rename the dependent variable "y" to "deposit" since that accurately describes the intent of the variable.

2.3 Data visualizations and inferences

Visualizing overall customers who have subscribed for deposit accounts, the data is highly imbalanced, there are a large number of people who have not subscribed compared to the number of customers who have.

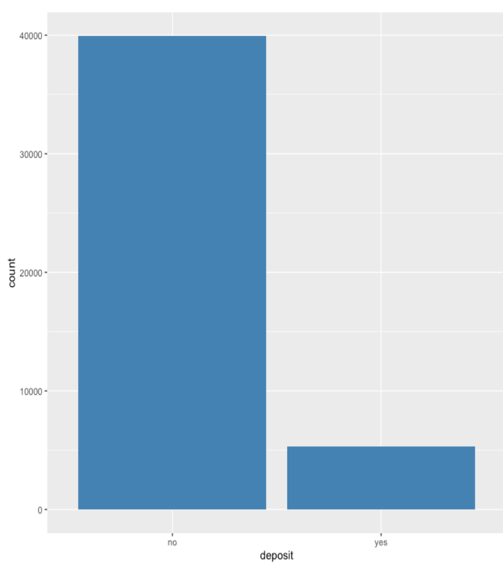

Fig 1 – Count of deposit results

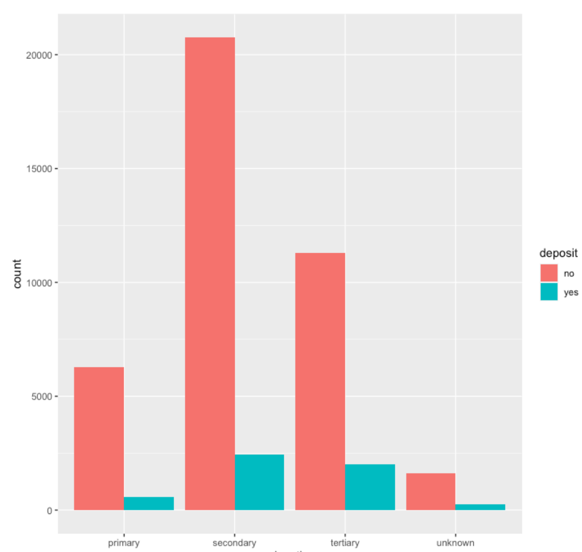

Fig 2 – Count of deposit results by education

Fig 1 shows the imbalance in the dataset around the dependent variable “deposit”. Fig 2 shows the distribution of deposit outcomes based on the customer’s education level. We see that overall, the education level does not have a high impact in subscribing to a deposit account.

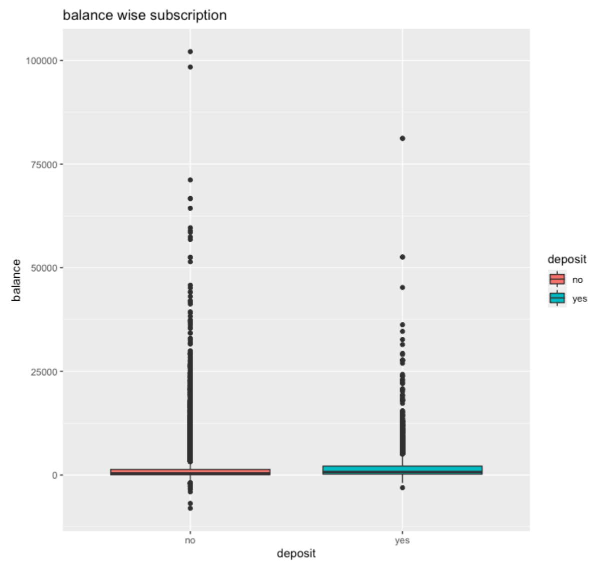

Fig 3 – Deposit decision vs balance distribution

From Fig 3, we see that the data has a high range. Customers with a negative balance have subscribed for the deposit account as have customers with a positive account balance.

Fig 4 shows the deposit decision vs employment categories and their balance distribution. Categories like management are observed to have a high range in terms of account balance. We can also infer that the deposit decision is not entirely dependent on balance alone since in several categories such as blue-collar or entrepreneur, we observe that a positive subscription decision is not in line with the highest balance which could also be attributed to outliers in the category. Highest balance is found in management and retired job types.

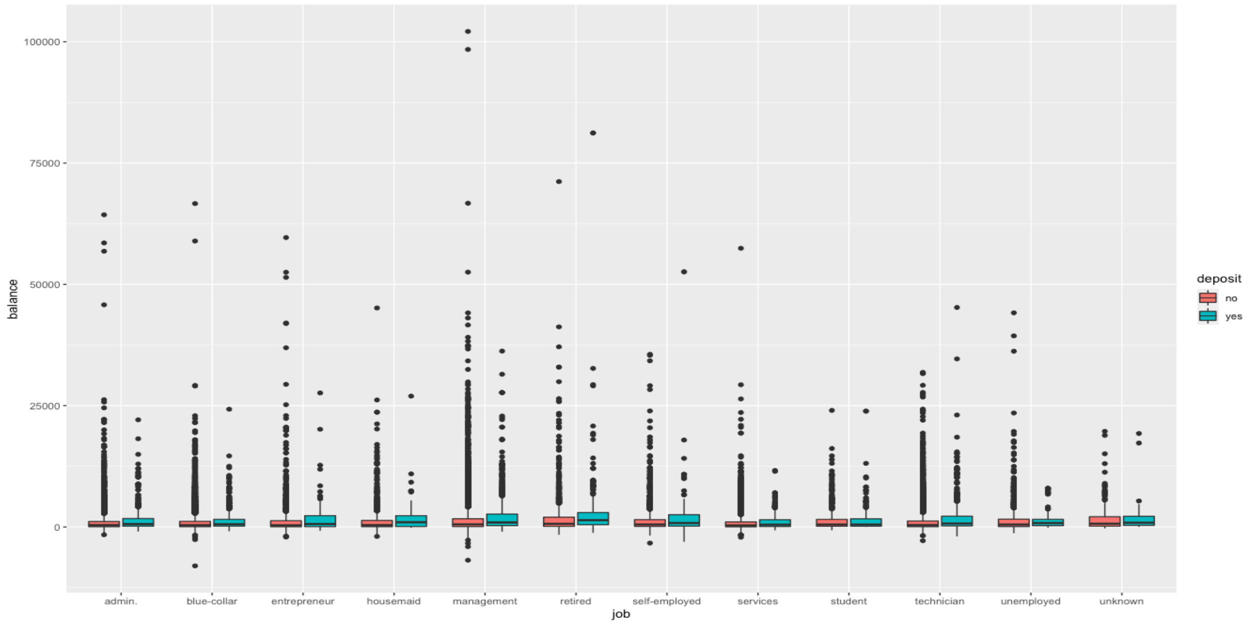

Fig 4 – Deposit decision vs employment category balance

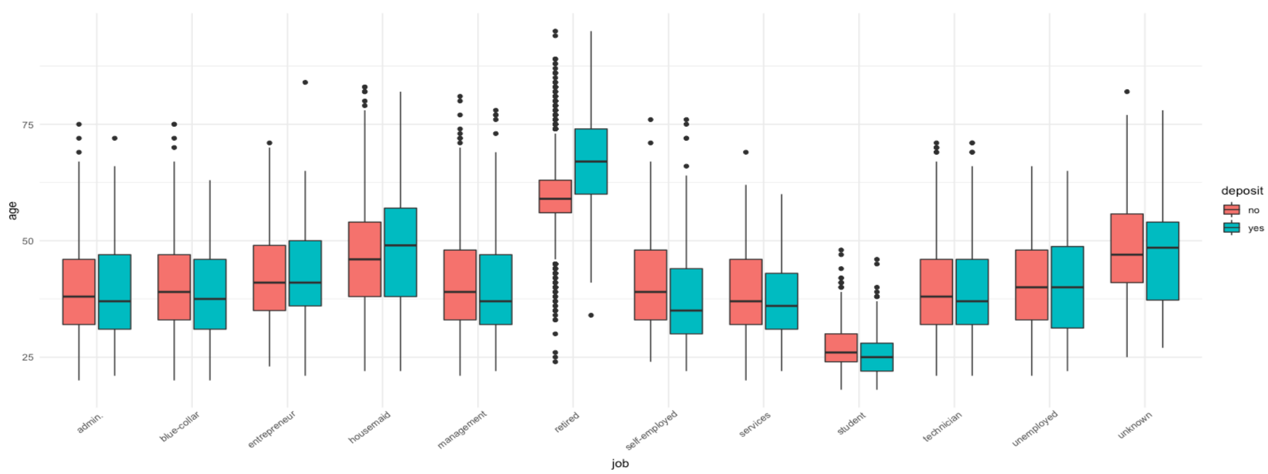

Fig 5 - Deposit decision vs employment category age

Fig 5 shows the deposit decision vs employment categories and their age distribution. A large number of retired customers tend to subscribe, followed by housemaids. Naturally, student category falls in the lowest age range and also the lowest subscription.

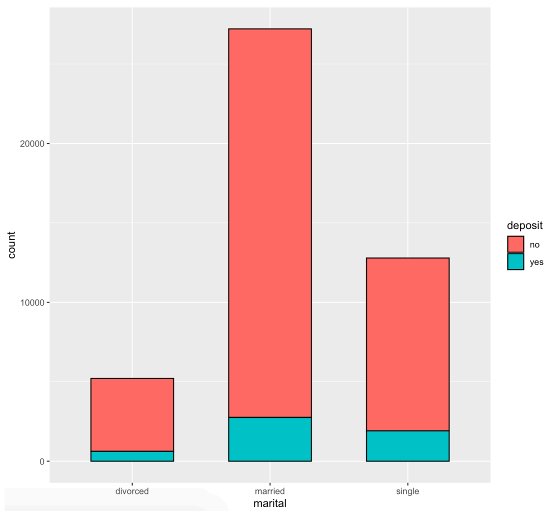

Fig 6 - Deposit decision vs marital status

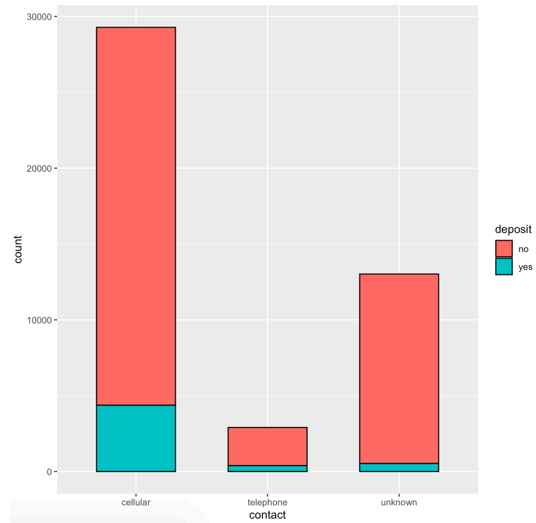

Fig 7 – Deposit decision vs method of contact

Fig 6 shows the distribution of deposit decision based on the customer's marital status. We see that the largest section of customers are married followed by single and divorced being the smallest section. Fig 7 details the distribution of deposit decision based on the bank's method of contact during a campaign. The largest number of customers were contacted via a cellular phone and also have the largest positive deposit decision by count. We also see that the contact variable has an 'unknown' category.

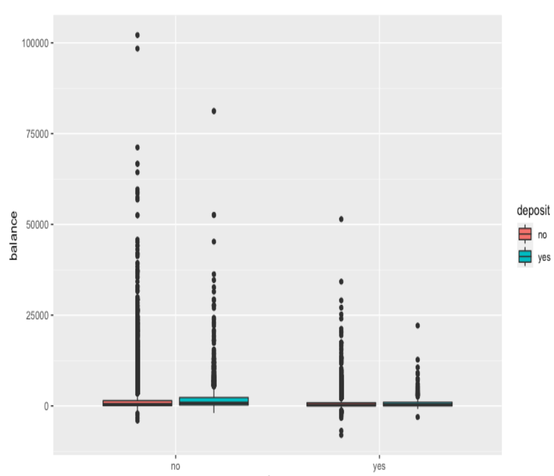

Fig 8 – Deposit decision vs loan and balance.

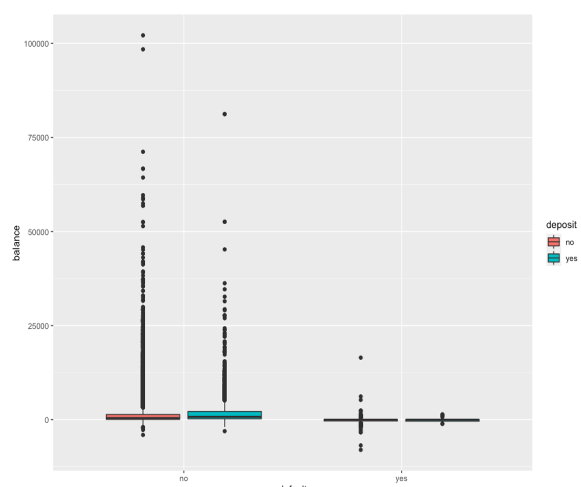

Fig 9 - Deposit decision vs credit default and balance

Fig 8 shows the deposit decision for customers who have loans against those who do not. We see that the customers that do not have loans tend to have a higher account balance and a higher range of balance. The chart also suggests that customers that do have loans are less likely to subscribe to a deposit account. Similar to the previous chart, Fig 9 shows deposit decision for customers who have credit default against those who do not. The dataset consists of a large number of customers who do not have a credit default and have a high balance range.

2.4 Data preparation

2.4.1 Understanding the variables

This dataset does not have any missing or empty values and does not require imputation or removal. In this section, we outline the data preparation processes that was undertaken to be used for modeling. Several categorical variables in the dataset have 'unknown' as a level. We are treating that as a value in itself and not dropping that level.

The variable 'pdays' which represents the number of days passed since the customer was last contacted from a previous campaign has values that range from -1 to 871. The variable 'previous', which represents the number of contacts performed before this campaign for this client has values that range from 0 to 275. 'pdays' having a value of -1 indicates that the customer was not previously contacted. Similarly, the variable 'previous' having a value of 0 also indicates that customer was not previously contacted. Looking at the count of 'pdays' of value -1 and 'previous' of value 0, we see that they have the same count.

Pdays: 1 - 36954   Previous: 0 - 36954

This suggests that they are representing the same thing. Fig 10 shows the correlation (corrplot) for the numerical variables. We observe that 'pdays' and 'previous' are slightly correlated.

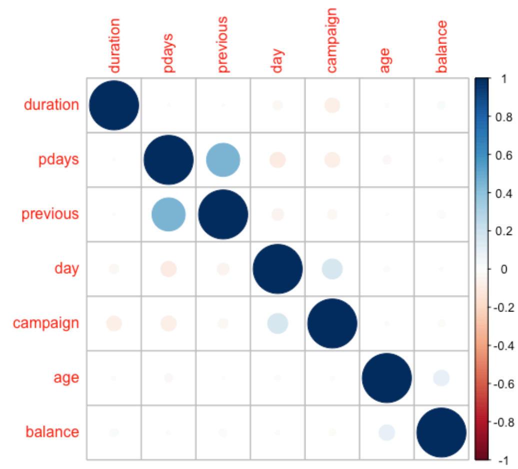

Fig 10 – Corrplot for numerical variables

Based on the correlation between ‘pdays’ and ‘previous’, we can choose to drop ‘pdays’ and retain the ‘previous’ variable.

The variable ‘poutcome’ which represents the outcome of the previous marketing campaign to a customer has 4 levels – ‘failure’, ‘success’, ‘other’ and ‘unknown’. Since ‘other’ and ‘unknown’ both suggest that the value is neither a ‘success’ nor a ‘failure’, we will convert all occurrences of ‘other’ to ‘unknown’ and as a result, the variable is reduced to having 3 levels.

The variable ‘duration’ is of particular interest since it represents the duration of the marketing call that lead to one of the ‘deposit’ decisions. The business case that we are attempting to model is the scenario of having an understanding of which customers are likely to subscribe to a deposit account before the telemarketing call is made. Based on that, we do not have any knowledge about the duration of the call a potential customer will engage in.

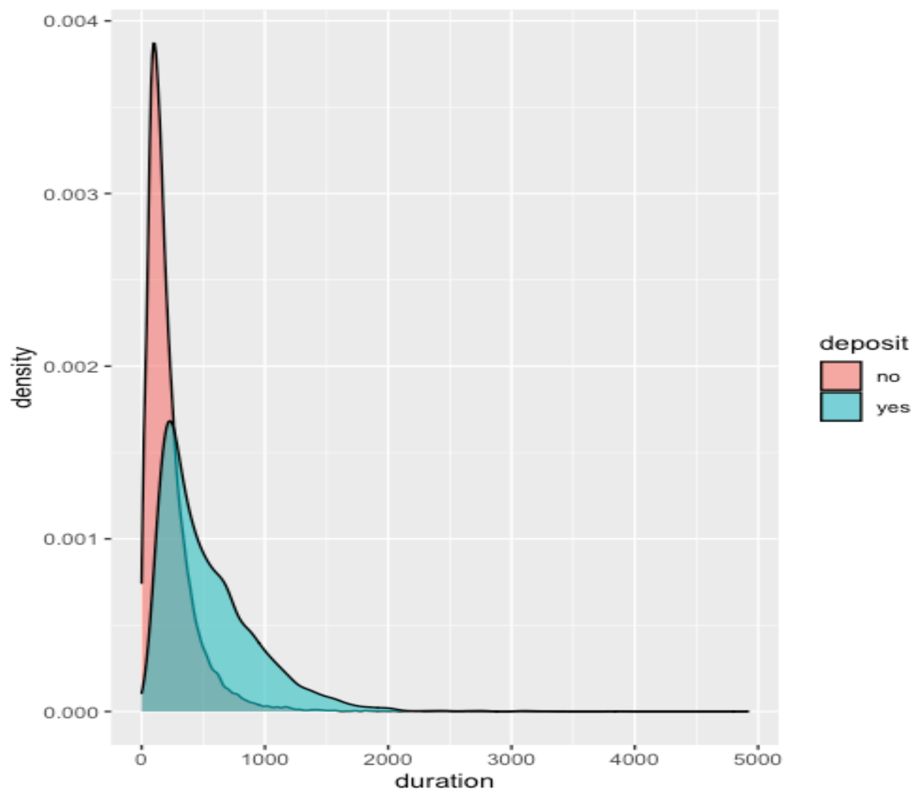

Fig 11 – Deposit decision vs duration of the call

Fig 11 shows the duration density of the call and the deposit decision. We see that the longer the duration, the higher the chance of subscribing to a deposit account. The decision to not subscribe is also suggested by the short duration of the call. For those reasons, we will drop the 'duration' variable from our dataset.

2.4.2 Training and test dataset split

The original dataset was split into a training set and a testing set on an 80 – 20 split. Training data consisted of a random 80% of the original observations and the test set consisted of the other 20%. This split was carried out due to the imbalance in the original dataset with the intent of applying balancing techniques on the training dataset while retaining the original imbalance in the test dataset.

2.4.3 Data imbalance

The dataset contains 45211 observations and 17 variables. From Table 1, we see that the variable ‘y’ (renamed to ‘deposit’) is a factor with 2 levels. The ‘yes’ level is what we are aiming to model. Looking at the distribution we see that there is a high imbalance:

no: 39922 yes: 5289

This suggests that the models will have a lot more observations with the no value compared to the number of observations with the yes value. This imbalance would cause the models to have a higher accuracy in predicting the no value compared to the yes value which the business case is interested in. To solve this imbalance, we will use the hybrid SMOTE technique to generate an even number of yes and no observations. The duration value distribution after SMOTE:

no :12717 yes:12717

The hybrid SMOTE technique was applied only to the training dataset and the test dataset retained the original imbalance.

3. Model planning and building

3.1 Models used

The business case suggests that this is a classification type problem. We will use the relevant classification models to predict the outcome.

Decision Tree We used the decision tree model to identify the top predictors in the dataset. The performance metric that we are interested in this problem is the True Positive Rate (TPR) since that represents the proportion of actual positives that were correctly identified. TPR is obtained via the confusion matrix and is represented by the Sensitivity rate.

Logistic Regression Since the goal is to predict if a customer will subscribe “yes” or “no”, a generalized linear model, where family = “binomial” was used and the model performance was observed on test and train set using 10 - fold cross validation.

Random Forest: Random Forest model was used to add randomness around finding the next set of trees. Random Forest randomly samples columns or predictors at each split to yield a more accurate model. Using cross validation, the appropriate hyperparameter (mtry) values were found to lie in the range of 2 to 8.

Boosting: Smaller sequential trees were fit using cross validation.

For better performance, to avoid overfitting and to also set appropriate tune lengths, a 10-fold cross validation method was used. This approach involves randomly dividing the set of observations into 10 groups, or folds, of approximately equal size. The first fold is treated as a test set, and the method is fit on the remaining $10 - 1$ folds and considered as training set. This allowed us to obtain the true test error.

After the model was fit on the training data, the test set was used to predict, and the metrics used were True Positivity rate (TPR or sensitivity) and Accuracy. To compare the performance of individual models, ROC curve was used, it is a useful tool for a few reasons:

- The curves of different models can be compared directly in general or for different thresholds.
- The area under the curve (AUC) can be used as a summary of the model skill.

Predicting the fitted models on the test dataset led to a low TPR but a higher Accuracy. However, the goal was not to gain high accuracy such that the model predicts how many customers won’t subscribe for an account due to the data imbalance, instead the goal was to predict which of the customers will indeed subscribe for an account. Thus, the Sensitivity metric was low the AUC observed for all these models was below the basic model which predicts on a 50-50 basis (No Information Rate)

3.2 Model Performance

The following section describes the model performance using the 4 models. Fig 12 and Fig 13 below compare the models using the ROC metric. The models trained with the balanced dataset performed better compared to the ones that were trained on the original dataset.

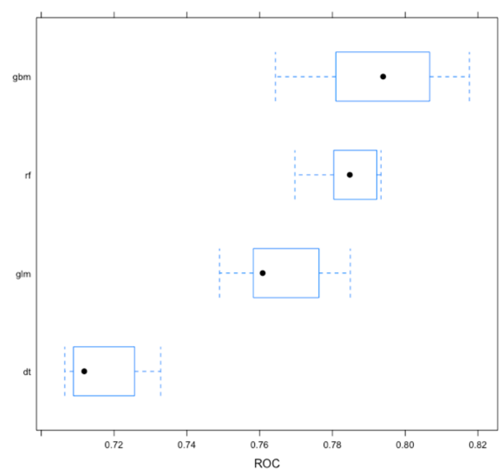

Fig 12 – ROC values for models using original dataset

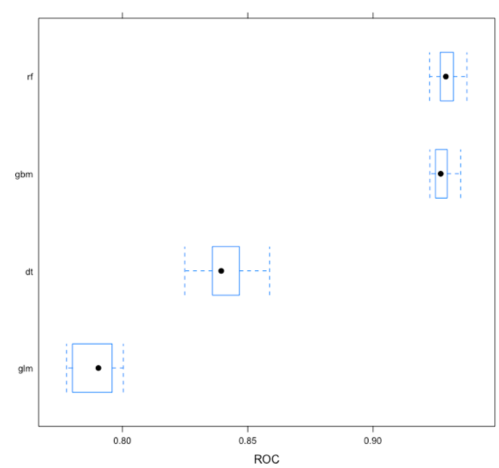

Fig 13 – ROC values for models using SMOTE dataset

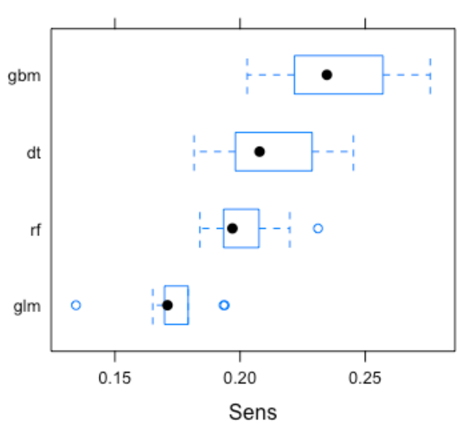

Fig 14 – Sensitivity values using original dataset

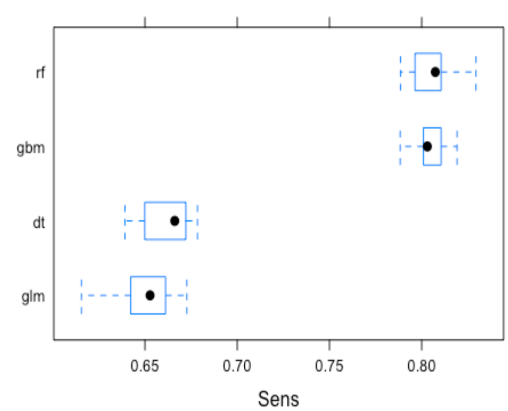

Fig 15 – Sensitivity values using SMOTE dataset

Fig 14 and Fig 15 show the Sensitivity values for all the 4 models used on original and SMOTE dataset. It is evident that the balanced training set was much more effective in training the models to increase sensitivity.

4. Models results

4.1 Confusion Matrices

		Actual	
Prediction		**yes**	**no**
	yes	254	177
	no	796	7815

Table 2 – Decision tree confusion matrix

Sensitivity: 0.24190

Accuracy: 0.8924

		Actual	
Prediction		**yes**	**no**
	yes	391	509
	no	659	7483

Table 3 – Decision Tree confusion matrix (SMOTE)

Sensitivity: 0.37238

Accuracy: 0.8708

Table 2 and Table 3 show the confusion matrices for the predictions using the Decision Tree model with the original imbalanced training dataset and the balanced dataset respectively. We see that the sensitivity metric increased with the balanced dataset, but not by a lot.

		Actual	
		yes	no
Prediction	yes	198	107
	no	852	7885

Table 4 – Logistic regression confusion matrix

Sensitivity: 0.18857

Accuracy: 0.8939

		Actual	
		yes	no
Prediction	yes	597	1689
	no	453	6303

Table 5 – Logistic regression confusion matrix (SMOTE)

Sensitivity: 0.56857

Accuracy: 0.7631

Table 4 and Table 5 show the confusion matrices for the predictions using Logistic Regression model using the original imbalanced training dataset and the balanced dataset respectively. We see that the sensitivity metric increased significantly with the balanced dataset, but also resulted in a minor drop in accuracy. Specificity on the other hand dropped from 0.98699 to 0.78866 which is a perfectly acceptable tradeoff for our problem.

		Actual	
		yes	no
Prediction	yes	228	123
	no	822	7869

Table 6 – Random forest confusion matrix

Sensitivity: 0.21714

Accuracy: 0.8955

		Actual	
		yes	no
Prediction	yes	575	896
	no	475	7096

Table 7 – Random forest confusion matrix (SMOTE)

Sensitivity: 0.54762

Accuracy: 0.8484

Table 6 and Table 7 show the confusion matrices for the predictions using the Random Forest model using the original imbalanced training dataset and the balanced dataset respectively. We see that the sensitivity metric increased significantly with the balanced dataset, but also resulted in a minor drop in accuracy. Specificity on the other hand dropped from 0.98198 to 0.88789 which is a perfectly acceptable tradeoff for our problem.

		Actual	
Prediction		**yes**	**no**
	yes	261	165
	no	789	7827

Table 8 – Boosting model confusion matrix

Sensitivity: 0.24857

Accuracy: 0.8945

		Actual	
Prediction		**yes**	**no**
	yes	519	580
	no	531	7412

Table 9 – Boosting confusion matrix (SMOTE)

Sensitivity: 0.4943

Accuracy: 0.8771

Table 8 and Table 9 show the confusion matrices for the predictions using the Boosting model using the original imbalanced training dataset and the balanced dataset respectively. We see that the sensitivity metric increased significantly with the balanced dataset, but also resulted in a minor drop in accuracy. Specificity on the other hand dropped from 0.97935 to 0.9274 which is a perfectly acceptable tradeoff for our problem.

4.2 ROC Plot

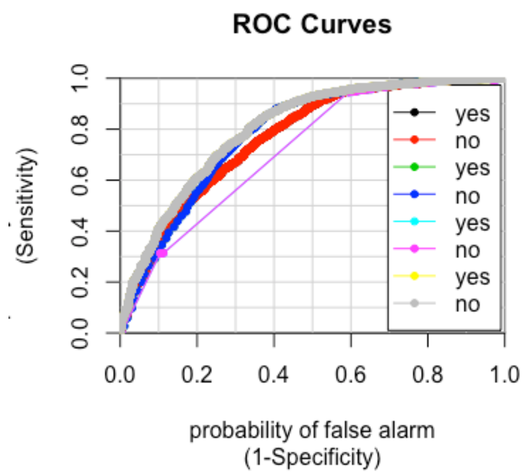

Fig 16 – ROC plot for the predictions.

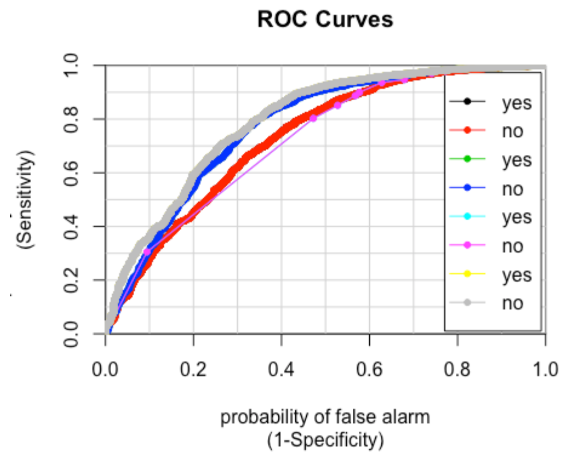

Fig 17 – ROC plot for the predictions

Fig 16 and Fig 17 shows the ROC plots for the predictions carried out using the 4 models on original and on smote dataset. Red line – Logistic regression, Blue line – Random Forest, Pink line – Decision Tree and Grey line – Boosting model.

5. Conclusion

5.1 Discussion and recommendations

Based on the training set, random forest and boosting model fared better. Considering the results from the prediction, logistic regression and random forest model had the top 2 highest sensitivity rates. Recommendations are made towards utilizing a better tune length given better computational capabilities. Overall, the analytical models were able to satisfactorily answer the business question and thus supporting the alternate hypothesis -

Alternative Hypothesis: There is a relationship between the predictors and dependent variable (Deposit).

Across all models, poutcome, balance, age, campaign, previous, day and month appeared to be the most important predictors. It is recommended to collect a larger variety of variables around the marketing campaign. The success criteria was met as we are able to use the developed models to predict if customers contacted via a marketing campaign will subscribe for a term deposit in the future.

Appendix

1. R code

```
library(caret)
library(corrplot)
library(dplyr)
library(tidyverse)
library(ggplot2)
library(rattle)
library(rpart)
library(rpart.plot)
library(caTools)
install.packages("DMwR")
library(DMwR)
install.packages("reshape2")
library(reshape2)

data <- read.csv("bank-full.csv", sep=";")
str(data)
glimpse(data)
attach(data)

sum(is.na(data))
# [1] 0

summary(data)

# Rename variable 'y' to 'deposit'
colnames(data)[17] = "deposit"


#####Visualizations
ggplot(data = data) + geom_bar(aes (x = deposit), fill = "steel blue")

ggplot(data = data) + geom_bar(aes(x = education, fill = deposit),
                        position = position_dodge(width = 0.9))

ggplot(data = data) + geom_boxplot(aes(y = balance, x = deposit, fill = deposit))+
                  ggtitle("balance wise subscription") + xlab("deposit") + ylab("balance")
summary(data$balance)


ggplot(data = data) + geom_boxplot(aes(y = balance, x = job, fill = deposit))+
```

```
                  xlab("job") + ylab("balance")
table(data$job)

ggplot(data = data) + geom_boxplot(aes(y = age, x = job, fill = deposit))+
                  xlab("job") + ylab("age") + theme_minimal() +
                  theme(legend.position = "right", axis.text.x.bottom = element_text(angle=40,
hjust=0.9))


ggplot(data = data) + geom_boxplot(aes(y = balance, x = marital, fill = deposit))+
                  xlab("marital") + ylab("balance") + theme_minimal()

ggplot(data = data) + geom_boxplot(aes(y = balance, x = loan, fill = deposit))+
                  xlab("loan") + ylab("balance")

ggplot(data = data) + geom_boxplot(aes(y = balance, x = housing, fill = deposit))+
                  xlab("housing") + ylab("balance")

ggplot(data = data) + geom_boxplot(aes(y = balance, x = default, fill = deposit))+
                  xlab("default") + ylab("balance")

marital_deposit <- table(data$marital, data$deposit)
mydata <- as.data.frame.matrix(marital_deposit)
mydata$marital <- rownames(mydata)
mydataLong <- melt(mydata, id.vars=c("marital"), value.name = "count")
names(mydataLong)[2] <- paste("deposit")
p <- ggplot(data=mydataLong, aes(x=marital,
                           y=count,
                           fill=deposit))
p + geom_bar(stat = "identity",
          width = 0.6,
          size = 0.5,
          color = "black")

myTable <- table(data$contact, data$deposit)
mydata <- as.data.frame.matrix(myTable)
mydata$contact <- rownames(mydata)
mydataLong <- melt(mydata, id.vars=c("contact"), value.name = "count")
names(mydataLong)[2] <- paste("deposit")
p <- ggplot(data=mydataLong, aes(x=contact,
                           y=count,
                           fill=deposit))
p + geom_bar(stat = "identity",
          width = 0.6,
          size = 0.5,
          color = "black")
```

```
#Visualizing continuous variables
data %>% ggplot(aes(x=deposit, y=campaign, fill=deposit)) + geom_boxplot()
data %>% ggplot(aes(x=deposit, y=previous, fill=deposit)) + geom_boxplot()

#Statistcal tests some categorical predictors
#Our hypothesis is that people who do have loans, housing or default they will not subscribe for
deposit account
table(data$default, data$deposit)
#          deposit
# default     no   yes
#      no   39159  5237
#      yes   763    52
chisq.test(table(data$default, data$deposit),correct=FALSE)
# data:  table(default, deposit)
# X-squared = 22.724, df = 1, p-value = 1.871e-06
#the p-value<0.05, proves that people who do default, they are less likely to subscribe for deposit
account


table(data$loan, data$deposit)
#         deposit
# loan     no     yes
#    no  33162   4805
#    yes  6760   484
chisq.test(table(data$loan, data$deposit))
# data:  table(loan, deposit)
# X-squared = 209.62, df = 1, p-value < 2.2e-16
#the p-value<0.05, proves that people who do have loan, they are less likely to subscribe for
deposit account


table(data$housing, data$deposit)
#           deposit
# housing    no   yes
#      no  16727  3354
#      yes 23195  1935
chisq.test(table(data$housing, data$deposit))
# data:  table(housing, deposit)
# X-squared = 874.82, df = 1, p-value < 2.2e-16
#the p-value<0.05, proves that people who do have housing, they are less likely to subscribe for
deposit account


#Data preprocessing, since we have a lot of factors, we might have to convert them
```

```
levels(data$poutcome)
# [1] "failure" "other"   "success" "unknown"

# Converting the other level to unknown and dropping the level other
other = which(data$poutcome=="other")
data$poutcome[other ] = "unknown"
table(data$poutcome)
data$poutcome <- droplevels(data$poutcome)
str(data)


# Identify correlated predictors
df_cor <- select_if(data, is.numeric) %>% cor()
corrplot(df_cor, method = "circle", order = "hclust")

#Pdays : "-1" means client was not previously contacted, this depicts the number of days sonce the customer was contacted,
#this is in a way explained using the previous variable too, since the previous is the number of conacts
#made with the customer as part of the campaign, thus dropping pdays and instead keeping Previous

#previous: number of contacts performed before this campaign and for this client
data %>% group_by(previous) %>%
    count() %>%
    arrange(desc(n)) %>%
    head()
# # A tibble: 6 x 2
# # Groups:   previous [6]
# previous     n
#       <int> <int>
# 1       0 36954
# 2       1  2772
# 3       2  2106
# 4       3  1142
# 5       4   714
# 6       5   459
# a large number of observations 0:36954 were not contaced ever

#Exploring pdays, pdays: -1 means client was not previously contacted
data %>% group_by(pdays) %>%
    count() %>%
    arrange(desc(n)) %>%
    head()
# # A tibble: 6 x 2
# pdays     n
```

```
#    <int> <int>
# 1    -1  36954
# 2   182   167
# 3    92   147
# 4    91   126
# 5   183   126
# 6   181   117

#the "pdays" value also suggests the same thing as the "previous" variable did, that is,
#a large number of customers were not contacted and thus keeping one of these instead of both
#The similar count of 0 and -1 values in both previous and pdays variables
#respectively suggest they are describing the same thing.

#correlation between the two
cor(data$pdays, data$previous)
# [1] 0.4548196    #slightly correlated in number but going by the pattern, it makes sense to drop
one of these

#dropping pdays
data$pdays <- NULL


data %>% count(data$month, sort = TRUE)
# A tibble: 12 x 2
# `data$month`     n
#   <fct>          <int>
# 1  may          13766
# 2  jul          6895
# 3  aug           6247
# 4  jun          5341
# 5  nov           3970
# 6  apr          2932
# 7  feb          2649
# 8  jan          1403
# 9  oct           738
# 10 sep           579
# 11 mar           477
# 12 dec           214

# Removing duration predictor
ggplot(data= data) + geom_density(aes(x = duration, fill = deposit), alpha = 0.6)
#The plot clearly suggests that the more time or the call duration is
#the more likely the person is going to subscribe for the deposit account, this predictor thus
cannot be used
data$duration <- NULL
```

```
str(data)
summary(data)
# data$deposit <- ifelse(data$deposit=="yes",1, 0)
# data$deposit <- factor(data$deposit)#,levels = c(0,1), labels= c("no", "yes") )
# levels(data$deposit)

#relevel the deposit factor so Yes becomes the Positive class by default (Sensitivity)
data$deposit <- relevel(data$deposit, ref="yes")
levels(data$deposit)

#This data is clean enough for our analyses

#----Fitting models

#dividing the dataset into train and test using 80/20 split
set.seed(123)
rows = sample(nrow(data))
shuffled_deposit = data[rows, ]
str(shuffled_deposit)
split = round(nrow(shuffled_deposit) * 0.80)
train_data = shuffled_deposit [1:split, ]
str(train_data)
test_data = shuffled_deposit [(split +1): nrow(shuffled_deposit), ]
str(test_data)

# Creating customized control for Cross Validation on training data only
set.seed(123)
control <- trainControl(method = "cv",
                number = 10,
                summaryFunction = twoClassSummary,
                classProbs = TRUE,
                verboseIter = TRUE)

# Create grid
rfGrid <-  expand.grid(mtry = c(2, 4, 6, 8))

# Fit the models

#Decision tree
set.seed(123)
dt_train <- train(deposit~. ,
            data = train_data,
            method= "rpart",
            trControl = control,
            preProcess = "range",
```

```
                tuneLength = 10)
plot(dt_train)
plot(dt_train$finalModel)
text(dt_train$finalModel)


prediction_dt <- predict(dt_train, test_data)
confusionMatrix(table(prediction_dt, test_data$deposit))

# Confusion Matrix and Statistics
#
#
# prediction_dt  yes   no
# yes           254  177
# no            796 7815
#
# Accuracy : 0.8924
# 95% CI : (0.8858, 0.8987)
# No Information Rate : 0.8839
# P-Value [Acc > NIR] : 0.005635
#
# Kappa : 0.2954
#
# Mcnemar's Test P-Value : < 2.2e-16
#
#            Sensitivity : 0.24190
#            Specificity : 0.97785
#         Pos Pred Value : 0.58933
#         Neg Pred Value : 0.90756
#             Prevalence : 0.11612
#         Detection Rate : 0.02809
#   Detection Prevalence : 0.04767
#      Balanced Accuracy : 0.60988
#
#       'Positive' Class : yes


#Random forests
set.seed(123)
rf_model <- train(deposit~. ,
                data = train_data,
                method= "rf",
                trControl = control,
                preProcess = "range",
                tuneGrid = rfGrid)
```

```
plot(rf_model)
varImp(rf_model)
#mtry of 6 gave higher Cross validated ROC and balance, age, day, poutcome, campaign,
previous were most important

prediction_rf <- predict(rf_model, test_data)
confusionMatrix(table(prediction_rf, test_data$deposit))

# Confusion Matrix and Statistics
#
#
# prediction_rf  yes   no
# yes  228  123
# no   822 7869
#
# Accuracy : 0.8955
# 95% CI : (0.889, 0.9017)
# No Information Rate : 0.8839
# P-Value [Acc > NIR] : 0.0002529
#
# Kappa : 0.2838
#
# Mcnemar's Test P-Value : < 2.2e-16
#
#             Sensitivity : 0.21714
#             Specificity : 0.98461
#          Pos Pred Value : 0.64957
#          Neg Pred Value : 0.90542
#              Prevalence : 0.11612
#          Detection Rate : 0.02522
#    Detection Prevalence : 0.03882
#       Balanced Accuracy : 0.60088
#
#        'Positive' Class : yes

#fitting a boosting model
set.seed(44)
gbm_model <- train(deposit ~.,
                   data=train_data,
                   method="gbm",
                   tuneLength=8,
                   trControl=control)

summary(gbm_model)
# balance, age, day, poutcome, contact, previous were most important
```

```
prediction_gbm <- predict(gbm_model, test_data)
confusionMatrix(table(prediction_gbm, test_data$deposit))

# Confusion Matrix and Statistics
#
#
# prediction_gbm  yes   no
# yes  261  165
# no   789 7827
#
# Accuracy : 0.8945
# 95% CI : (0.888, 0.9008)
# No Information Rate : 0.8839
# P-Value [Acc > NIR] : 0.000753
#
# Kappa : 0.3072
#
# Mcnemar's Test P-Value : < 2.2e-16
#
#            Sensitivity : 0.24857
#            Specificity : 0.97935
#         Pos Pred Value : 0.61268
#         Neg Pred Value : 0.90843
#             Prevalence : 0.11612
#         Detection Rate : 0.02887
#   Detection Prevalence : 0.04711
#      Balanced Accuracy : 0.61396
#
#       'Positive' Class : yes

#fitting a logistic regression model
set.seed(123)
glm_model <- train(deposit ~ .,
                   data = train_data,
                   method = "glm",
                   family = "binomial",
                   #metric = "ROC",
                   preProcess = "range",
                   trControl = control)

summary(glm_model)
#poutcome, balance, campaign, previous amongst others appear significant age and daya does
not appear significant

prediction_glm <- predict(glm_model, test_data)
confusionMatrix(table(prediction_glm, test_data$deposit))
```

```
# Confusion Matrix and Statistics
#
#
# prediction_glm  yes   no
# yes  198  107
# no   852 7885
#
# Accuracy : 0.8939
# 95% CI : (0.8874, 0.9002)
# No Information Rate : 0.8839
# P-Value [Acc > NIR] : 0.001328
#
# Kappa : 0.2532
#
# Mcnemar's Test P-Value : < 2.2e-16
#
#            Sensitivity : 0.18857
#            Specificity : 0.98661
#         Pos Pred Value : 0.64918
#         Neg Pred Value : 0.90248
#             Prevalence : 0.11612
#         Detection Rate : 0.02190
#   Detection Prevalence : 0.03373
#      Balanced Accuracy : 0.58759
#
#       'Positive' Class : yes


# Comparing model performance
models<- list("glm" = glm_model,
             "rf" = rf_model,
             "dt"=dt_train,
             "gbm"=gbm_model)

model.resamples<- resamples(models)
summary(model.resamples)

# Models: glm, rf, dt, gbm
# Number of resamples: 10
#
# ROC
# Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
# glm 0.7489504 0.7585601 0.7608315 0.7648752 0.7728313 0.7849076    0
# rf  0.7697521 0.7804738 0.7831527 0.7842175 0.7892213 0.7986660    0
# dt  0.7064691 0.7093583 0.7118115 0.7161127 0.7233412 0.7328114    0
```

```
# gbm 0.7669434 0.7864576 0.7945238 0.7941535 0.8068658 0.8107117    0
#
# Sens
#         Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
# glm 0.1344340 0.1698113 0.1709906 0.1722155 0.1780660 0.1938534    0
# rf  0.1839623 0.1939858 0.1969340 0.2017028 0.2069575 0.2311321    0
# dt  0.1816038 0.1981132 0.2077869 0.2094819 0.2246462 0.2452830    0
# gbm 0.2028302 0.2240566 0.2346698 0.2380347 0.2558962 0.2759434    0
#
# Spec
# Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
# glm 0.9840276 0.9861416 0.9876292 0.9874100 0.9880990 0.9915440    0
# rf  0.9824616 0.9849671 0.9857501 0.9855622 0.9865330 0.9871594    0
# dt  0.9771375 0.9790166 0.9797996 0.9799562 0.9810523 0.9827748    0
# gbm 0.9715002 0.9768243 0.9782336 0.9788600 0.9819136 0.9852803    0

#plot performances
bwplot(model.resamples, metric="ROC")
bwplot(model.resamples, metric="Sens")
bwplot(model.resamples, metric="Spec")

# ROC plot
# Predict on test
glm_pred = predict(glm_model, test_data, type="prob")
rf_pred = predict(rf_model, test_data, type="prob")
dt_pred = predict(dt_train, test_data, type="prob")
gbm_pred = predict(gbm_model, test_data, type="prob")

# Make ROC curve
colAUC(cbind(glm_pred, rf_pred, dt_pred, gbm_pred), test_data$deposit, plotROC = TRUE,
alg="ROC")

########################################################################################
###################################################
# Data imbalance - Use SMOTE balance train data
set.seed(9560)
train_data_smote <- SMOTE(deposit ~ ., perc.over=250, perc.under=150,
                    data  = train_data)

summary(train_data_smote)

# Fit the models

#Decision tree on the balanced data
set.seed(123)
dt_train_smote <- train(deposit~. ,
```

```
                data = train_data_smote, # Balanced training data
                method= "rpart",
                trControl = control,
                preProcess = "range",
                tuneLength = 10)

plot(dt_train_smote)
plot(dt_train_smote$finalModel)
text(dt_train_smote$finalModel)

prediction_dt_smote <- predict(dt_train_smote, test_data)
confusionMatrix(table(prediction_dt_smote, test_data$deposit))

# Confusion Matrix and Statistics
#
#
# prediction_dt_smote  yes   no
# yes  391  509
# no   659 7483
#
# Accuracy : 0.8708
# 95% CI : (0.8637, 0.8777)
# No Information Rate : 0.8839
# P-Value [Acc > NIR] : 0.9999
#
# Kappa : 0.3291
#
# Mcnemar's Test P-Value : 1.302e-05
#
#            Sensitivity : 0.37238
#            Specificity : 0.93631
#         Pos Pred Value : 0.43444
#         Neg Pred Value : 0.91906
#             Prevalence : 0.11612
#         Detection Rate : 0.04324
#   Detection Prevalence : 0.09954
#      Balanced Accuracy : 0.65435
#
#       'Positive' Class : yes


#fitting a GLM model on the balanced data
set.seed(123)
glm_model_smote <- train(deposit ~ .,
                 data = train_data_smote, # Balanced training data
                 method = "glm",
```

```
                family = "binomial",
                metric = "ROC",
                preProcess = "range",
                trControl = control)

summary(glm_model_smote)

prediction_glm_smote <- predict(glm_model_smote, test_data)
confusionMatrix(table(prediction_glm_smote, test_data$deposit))

# Confusion Matrix and Statistics
#
#
# prediction_glm_smote  yes   no
# yes  597 1689
# no   453 6303
#
# Accuracy : 0.7631
# 95% CI : (0.7542, 0.7718)
# No Information Rate : 0.8839
# P-Value [Acc > NIR] : 1
#
# Kappa : 0.2364
#
# Mcnemar's Test P-Value : <2e-16
#
#            Sensitivity : 0.56857
#            Specificity : 0.78866
#         Pos Pred Value : 0.26115
#         Neg Pred Value : 0.93295
#             Prevalence : 0.11612
#         Detection Rate : 0.06603
#   Detection Prevalence : 0.25282
#      Balanced Accuracy : 0.67862
#
#       'Positive' Class : yes


#Random forests with balanced data
set.seed(123)
rf_model_smote <- train(deposit~. ,
               data = train_data_smote, # Balanced training data
               method= "rf",
               trControl = control,
               tuneGrid = rfGrid)
```

```
plot(rf_model_smote)

prediction_rf_smote <- predict(rf_model_smote, test_data)
confusionMatrix(table(prediction_rf_smote, test_data$deposit))

# Confusion Matrix and Statistics
#
#
# prediction_rf_smote  yes   no
# yes  575  896
# no   475 7096
#
# Accuracy : 0.8484
# 95% CI : (0.8408, 0.8557)
# No Information Rate : 0.8839
# P-Value [Acc > NIR] : 1
#
# Kappa : 0.3709
#
# Mcnemar's Test P-Value : <2e-16
#
#            Sensitivity : 0.54762
#            Specificity : 0.88789
#         Pos Pred Value : 0.39089
#         Neg Pred Value : 0.93726
#             Prevalence : 0.11612
#         Detection Rate : 0.06359
#   Detection Prevalence : 0.16269
#      Balanced Accuracy : 0.71775
#
#       'Positive' Class : yes


#fitting a boosting model with balanced data
set.seed(44)
gbm_model_smote <- train(deposit ~.,
                 data=train_data_smote, # Balanced training data
                 method="gbm",
                 tuneLength=8,
                 trControl=control)

summary(gbm_model_smote)

prediction_gbm_smote <- predict(gbm_model_smote, test_data)
confusionMatrix(table(prediction_gbm_smote, test_data$deposit))
```

```
# Confusion Matrix and Statistics
#
#
# prediction_gbm_smote  yes   no
# yes  519  580
# no   531 7412
#
# Accuracy : 0.8771
# 95% CI : (0.8702, 0.8838)
# No Information Rate : 0.8839
# P-Value [Acc > NIR] : 0.9776
#
# Kappa : 0.4133
#
# Mcnemar's Test P-Value : 0.1498
#
#            Sensitivity : 0.4943
#            Specificity : 0.9274
#         Pos Pred Value : 0.4722
#         Neg Pred Value : 0.9331
#             Prevalence : 0.1161
#         Detection Rate : 0.0574
#   Detection Prevalence : 0.1215
#      Balanced Accuracy : 0.7109
#
#       'Positive' Class : yes


# Comparing model performance
models_smote<- list("glm" = glm_model_smote,
            "rf" = rf_model_smote,
            "dt" = dt_train_smote,
            "gbm" = gbm_model_smote)

model.resamples_smote<- resamples(models_smote)
summary(model.resamples_smote)

# Models: glm, rf, dt, gbm
# Number of resamples: 10
#
# ROC
# Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
# glm 0.7777203 0.7821073 0.7904101 0.7897152 0.7957770 0.8003374    0
# rf  0.9226117 0.9273752 0.9290719 0.9300127 0.9318221 0.9374784    0
# dt  0.8248598 0.8358744 0.8394433 0.8407863 0.8458831 0.8587343    0
# gbm 0.9227209 0.9249813 0.9270228 0.9273514 0.9294836 0.9350002    0
```

```
#
# Sens
#          Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
# glm 0.6155660 0.6439955 0.6527724 0.6507056 0.6603774 0.6726987    0
# rf  0.7885220 0.7981139 0.8073899 0.8064016 0.8099072 0.8294025    0
# dt  0.6391509 0.6502754 0.6661418 0.6618684 0.6719733 0.6784591    0
# gbm 0.7883556 0.8011408 0.8030660 0.8046691 0.8101415 0.8191824    0
#
# Spec
# Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
# glm 0.7537372 0.7797443 0.7841981 0.7826513 0.7913472 0.7932390    0
# rf  0.9118804 0.9184355 0.9213836 0.9216801 0.9266476 0.9307632    0
# dt  0.9221698 0.9257075 0.9331517 0.9351285 0.9461172 0.9488994    0
# gbm 0.9166667 0.9286134 0.9323630 0.9319811 0.9386435 0.9394654    0

#plot performances
bwplot(model.resamples_smote, metric="ROC")
bwplot(model.resamples_smote, metric="Sens")
bwplot(model.resamples_smote, metric="Spec")

# ROC plot
# Predict on test
glm_pred_smote = predict(glm_model_smote, test_data, type="prob")
rf_pred_smote = predict(rf_model_smote, test_data, type="prob")
dt_pred_smote = predict(dt_train_smote, test_data, type="prob")
gbm_pred_smote = predict(gbm_model_smote, test_data, type="prob")

# Make ROC curve
colAUC(cbind(glm_pred_smote, rf_pred_smote, dt_pred_smote, gbm_pred_smote),
test_data$deposit, plotROC = TRUE)
```