Project Report

Banking-AMZ Bank Marketing Analysis

Prepared by: Group 5

Yuanying Li,

Ketaki Joshi

Na Qian

Chen Liang

Course: ALY 6020

Under the Guidance Of:

Prof. Alakh Verma

Northeastern University

Oct 25, 2020

Contents

1. Introduction

AMZ Bank is recognized for its high standards and advanced service philosophy and is the 7th largest lender in terms of assets Network of over 400 branches in Asia Pacific and Relationships with 1000 banks in 70 countries around the world. They are already a large institution with large data so the ability to scale is mandatory, but they wanted to place themselves at the leading edge. Their vision is to become ultra-modern and data-driven—an organization enabled to use all their data to drive business excellence.

Recently they decided to maximize the number of active credit card customers by better targeting marketing incentives to those most likely to activate and use them for their business transactions. They also want to isolate the cards that would likely never be activated to reduce wasted marketing spend. They wish to reward their active and paying customers and reduce overheads on maintaining not so paying customers.

For the above purposes, AMZ bank would like to understand the demographics and other characteristics associated with whether a customer accepts a credit card offer (activate a new credit card from AMZ Bank). To get around this, we designed a focused marketing study, with 18,000 current bank customers. This focused approach allows the bank to know who does and does not respond to the offer, and to use existing demographic data that is already available on each customer.

Besides, the AMZ bank wants to reduce the customer churn with the data-driven strategy. Customer churn also known as customer turnover is the loss of customers. It is an important metric for AMZ Bank to evaluate whether customers stop using the bank's products or services.

As an international banking giant, understanding the needs, preferences, sentiments of existing customers are keys to ensuring business expansion and long-term profitability. In order to identify potential customer churn, it is essential to analyze our customers' attributes and behaviors.

In this project, we developed statistical models that will provide insight into why some customers accept credit card offers (activate a new credit card or not). This designed approach will allow the bank to understand potential and important factors that affects customers' decisions in accepting or rejecting an offer. This analysis will help bank to reward their valuable customers and reduce overheads on maintaining not so paying customers. With the help of our analysis, AMZ bank will be able to focus and target customers for a marketing campaign to increase active credit card rate and reduce unnecessary marketing costs.

Besides that, we found out what features (i.e. age, gender, balance, etc.) are highly related to customer churn through Exploratory Data Analysis. Then, we used predictive analytics techniques in R to build machine learning models and selected the best model by evaluating and comparing the prediction accuracy of each model. We used the best model to predict who is most likely to churn. We also came up with some marketing strategies to reduce customer churn.

2. Dataset(s) information

Part I: We used the credit card marketing data from the data world. The features in the data set represent the characteristics of each customer in the bank.

Goal: To identify and predict which customers will accept the offer and activate a credit card and who will decline the offer.

Link: https://data.world/gautam2510/credit-card-dataset/workspace/data-dictionary

The explanation of the features is as follows:

- Customer Number: A sequential number assigned to the customers (this column is hidden and excluded – this unique identifier will not be used directly).
- ActiveCreditCard: Did the customer active (Yes) or not active (No) the credit card.
- Reward: The type of reward program offered for the card.
- Mailer Type: Letter or postcard.
- Income Level: Low, Medium, or High.
- Age: customers' age
- Tenure: The number of years that the customer has been a client of the bank.
- Job: Job type of customer (admin, blue-collar, entrepreneur, housemaid, management, retired, self-employed, services, student, technician, unemployed, unknown)
- Bank Accounts Open: How many non-credit-card accounts are held by the customer
- Overdraft Protection: Does the customer have overdraft protection on their checking account(s) (Yes or No).
- Credit Rating: Low, Medium, or High.
- Credit Cards Held: The number of credit cards held at the bank.
- Homes Owned: The number of homes owned by the customer.
- Household Size: Number of individuals in the family.
- Own Your Home: Does the customer own their home? (Yes or No).
- Average Balance: Average account balance (across all accounts over time).
- Q1, Q2, Q3, and Q4 Balance: Average balance for each quarter in the last year.

Based on these column names, some discussion could be generated before further step:

1) Segment of the Population: Which segment of the population is going to activate the credit card and why? What kind of people would accept a new credit card and activate it? Is it related to a customer's age, job, tenure, etc.? This aspect is extremely important since it will tell which part of the population should most likely activate their credit card.

2) Distribution channel to reach the customer's place: Implementing the most effective strategy in order to get the most out of a marketing campaign. What segment of the population should

we address? Which instrument should we use to get our message out? (Ex: in this data set, for mailer type, which way is more efficient? letter or postcard.)

Part II: We used the churn data from kaggle. The features in the data set represent the characteristics of each customer in the bank.

Goal: Classify the customers who will and will not churn

Link: https://www.kaggle.com/mathchi/churn-for-bank-customers

The explanation of the features is as follows:

- Surname: The surname of a bank customer
- CreditScore: Credit score for a customer
- Gender: Gender (Female / Male)
- Age: Age
- Tenure: The number of years that the customer has been a client of the bank.
- Balance: Balance in the bank account
- NumOfProducts: The number of products that a customer has in the bank
- HasCrCard: Credit card status (0 = No, 1 = Yes)
- UseFrequency: Active membership status (0 = No, 1 = Yes)
- EstimatedSalary: Customer's self-reported annual salary
- Churn: Whether the customer left the bank or not (0 = No, 1 = Yes)

3. Method(s) chosen

Part I: Exploratory data analysis

Statistical models:

- Decision Tree
- Naïve Bayes, and
- Random Forest

Part II: Exploratory data analysis

Statistical models:

- Logistic Regression
- Decision Tree
- Random Forest
- KNN
- Naïve Bayes

Evaluation method

We used confusion matrix to evaluate the performance of the models and tested accuracy.

4. Part I

4.1 Data Cleaning and preprocessing

After installing the required libraries for analysis, model building, and visualizations, we imported the dataset into the R environment.

The dataset contains 18000 observations and 20 variables.

```
Classes ‘spec_tbl_df’, ‘tbl_df’, ‘tbl’ and 'data.frame':        18000 obs. of  20 varia
bles:
 $ CustomerNumber     : num  1 2 3 4 5 6 7 8 9 10 ...
 $ ActiveCreditCard   : chr  "No" "No" "No" "No" ...
 $ Reward             : chr  "Air Miles" "Air Miles" "Air Miles" "Air Miles" ...
 $ MailerType         : chr  "Letter" "Letter" "Postcard" "Letter" ...
 $ IncomeLevel        : chr  "High" "Medium" "High" "Medium" ...
 $ Age                : num  40 91 48 58 77 45 37 74 69 58 ...
 $ Tenure             : num  2 10 6 8 6 10 9 6 3 5 ...
 $ Job                : chr  "management" "retired" "management" "management" ...
 $ BankAccountsOpen   : num  2 1 1 1 1 1 2 1 1 2 ...
 $ OverdraftProtection: chr  "No" "No" "No" "No" ...
 $ CreditRating       : chr  "Medium" "Low" "Medium" "High" ...
 $ CreditCardsHeld    : num  2 2 1 4 2 1 1 1 1 2 ...
 $ HomesOwned         : num  1 2 1 2 1 2 2 1 1 1 ...
 $ HouseholdSize      : num  4 5 3 2 2 2 4 2 3 5 ...
 $ OwnYourHome        : chr  "Yes" "Yes" "Yes" "No" ...
 $ AverageBalance     : num  664 982 178 1120 1194 ...
 $ Q1Balance          : num  695 2041 249 771 1773 ...
 $ Q2Balance          : num  534 885 172 761 1302 ...
 $ Q3Balance          : num  735 950 289 1525 1360 ...
 $ Q4Balance          : num  693 53 4 1421 341 ...
```

The summary of the dataset is as follows:

```
 IncomeLevel             Age              Tenure            Job
 Length:18000       Min.   :18.00   Min.   : 1.000   Length:18000
 Class :character   1st Qu.:38.00   1st Qu.: 3.000   Class :character
 Mode  :character   Median :57.00   Median : 5.000   Mode  :character
                    Mean   :56.81   Mean   : 5.475
                    3rd Qu.:76.00   3rd Qu.: 8.000
                    Max.   :95.00   Max.   :10.000

 BankAccountsOpen OverdraftProtection CreditRating       CreditCardsHeld
 Min.   :1.000    Length:18000        Length:18000       Min.   :1.000
 1st Qu.:1.000    Class :character    Class :character   1st Qu.:1.000
 Median :1.000    Mode  :character    Mode  :character   Median :2.000
 Mean   :1.256                                           Mean   :1.903
 3rd Qu.:1.000                                           3rd Qu.:2.000
 Max.   :3.000                                           Max.   :4.000

   HomesOwned     HouseholdSize   OwnYourHome        AverageBalance
 Min.   :1.000   Min.   :1.000   Length:18000       Min.   :  48.25
 1st Qu.:1.000   1st Qu.:3.000   Class :character   1st Qu.: 787.50
 Median :1.000   Median :3.000   Mode  :character   Median :1007.00
 Mean   :1.203   Mean   :3.499                      Mean   : 940.52
 3rd Qu.:1.000   3rd Qu.:4.000                      3rd Qu.:1153.25
 Max.   :3.000   Max.   :9.000                      Max.   :3366.25
                                                    NA's   :24
   Q1Balance        Q2Balance        Q3Balance        Q4Balance
 Min.   :   0.0   Min.   :   0.0   Min.   :   0.0   Min.   :   0.0
 1st Qu.: 392.8   1st Qu.: 663.0   1st Qu.: 633.0   1st Qu.: 363.0
 Median : 772.0   Median :1032.0   Median : 945.5   Median : 703.0
 Mean   : 910.5   Mean   : 999.4   Mean   :1042.0   Mean   : 810.2
 3rd Qu.:1521.0   3rd Qu.:1342.0   3rd Qu.:1463.0   3rd Qu.:1212.0
 Max.   :3450.0   Max.   :3421.0   Max.   :3823.0   Max.   :4215.0
 NA's   :24       NA's   :24       NA's   :24       NA's   :24
```

Observations:

1. Sample size is large (within 18000), with 20 variables, the dependent variable is called ActiveCreditCard. There are some missing values in AverageBalance, Q1Balance, Q2Balance, Q3Balance, Q4Balance, which would be converted to another value later.

2. The first variable is an integer variable named id. As this is simply a unique identifier (ID) for each patient in the data, it does not provide useful information, and we will need to exclude it from the model.

When we checked for the missing values, we found that 120 values are missing in the dataset. To get the specific details about the missing values, we used **colSums(is.na())** function. This function returns the name of the column and the number of missing values in that column. The result is as follows:

```
> colSums(is.na(df))
     CustomerNumber    ActiveCreditCard             Reward         MailerType
                  0                   0                  0                  0
        IncomeLevel                 Age             Tenure                Job
                  0                   0                  0                  0
   BankAccountsOpen OverdraftProtection       CreditRating     CreditCardsHeld
                  0                   0                  0                  0
        HomesOwned       HouseholdSize       OwnYourHome      AverageBalance
                  0                   0                  0                 24
         Q1Balance           Q2Balance         Q3Balance          Q4Balance
                 24                  24                 24                 24
```

We can see that in total 120 values are missing from AverageBalance, Q1Balance, Q2Balance, Q3Balance, and Q4Balance.

The missing values can be dealt with by taking the mean /median of the values. We took the mean of the values.

In this study, we have 16 factors that possibly influence the decision of accepting or rejecting a credit card offer. There are some unnecessary factors as well. ID is not necessary to keep for predicting the final decision. We could just use AverageBalance to show information about balance, the rest of the quarter's balance could just drop it. So, we dropped unnecessary columns.

```
#Replace null values in titalcharges into mean of averagebalance
df[is.na(df)] <- 940.52

colnames(df)

#We have average balance and all four quarter balance(Q1, Q2, Q3, Q4) as variab
#So, we dropped these four variables and used only Average Balance variable.

# Dropping unwanted columns
Drop <- names(df) %in% c("CustomerNumber", "Q1Balance","Q2Balance","Q3Balance",
df <- df[!Drop]
View(df)
```

Following is the summary of the dataset after dropping the unnecessary variables:

```
> summary(df)
 ActiveCreditCard      Reward           MailerType        IncomeLevel
 Length:18000       Length:18000       Length:18000       Length:18000
 Class :character   Class :character   Class :character   Class :character
 Mode  :character   Mode  :character   Mode  :character   Mode  :character


      Age             Tenure           Job             BankAccountsOpen
 Min.   :18.00   Min.   : 1.000   Length:18000       Min.   :1.000
 1st Qu.:38.00   1st Qu.: 3.000   Class :character   1st Qu.:1.000
 Median :57.00   Median : 5.000   Mode  :character   Median :1.000
 Mean   :56.81   Mean   : 5.475                      Mean   :1.256
 3rd Qu.:76.00   3rd Qu.: 8.000                      3rd Qu.:1.000
 Max.   :95.00   Max.   :10.000                      Max.   :3.000
 OverdraftProtection CreditRating       CreditCardsHeld   HomesOwned
 Length:18000        Length:18000       Min.   :1.000   Min.   :1.000
 Class :character    Class :character   1st Qu.:1.000   1st Qu.:1.000
 Mode  :character    Mode  :character   Median :2.000   Median :1.000
                                        Mean   :1.903   Mean   :1.203
                                        3rd Qu.:2.000   3rd Qu.:1.000
                                        Max.   :4.000   Max.   :3.000
 HouseholdSize   OwnYourHome        AverageBalance
 Min.   :1.000   Length:18000       Min.   :  48.25
 1st Qu.:3.000   Class :character   1st Qu.: 787.94
 Median :3.000   Mode  :character   Median :1006.50
 Mean   :3.499                      Mean   : 940.52
 3rd Qu.:4.000                      3rd Qu.:1152.56
 Max.   :9.000                      Max.   :3366.25
```

4.2 Imbalanced classes conversion

The dataset contains **16977** negative cases and only **1023** positive cases. That means, only 1023 customers accepted the offer and activated the credit card. We checked the percentage, and it is as follows:

```
> table(df$ActiveCreditCard)

   No   Yes
16977  1023
```

```
> prop.table(table(df$ActiveCreditCard))

        No        Yes
0.94316667 0.05683333
```

We can see this data set contains only **5.68%**(rounded) positive cases. This means only 5.68% accepted the offer and activated the credit card. Rest **94.31%** of cases are negative. This is a severely imbalanced data set and will affect the model's prediction accuracy. Therefore, it is necessary to balance data before applying a machine learning algorithm.

We can use the sampling method-under sampling and oversampling.

If we only use the undersampling method, we can lose significant information from the sample. Hence, we used ***'both'*** methods to balance the data set. In this case, the minority class is oversampled with replacement and the majority class is under sampled without replacement.

We use a library(ROSE) to balance the classes. This library provides ovum.sample() function which makes it easy to handle the imbalanced class.

```
#Balancing the classes
df_new<- ovun.sample(ActiveCreditCard ~ ., data = df, method = "both", p=0.5, seed = 2020)
```

Let us check whether the dataset is balanced or not. To check this, we again used the table() function and the result is as follows:

```
> table(df$ActiveCreditCard)

  No  Yes
9104 8896
```

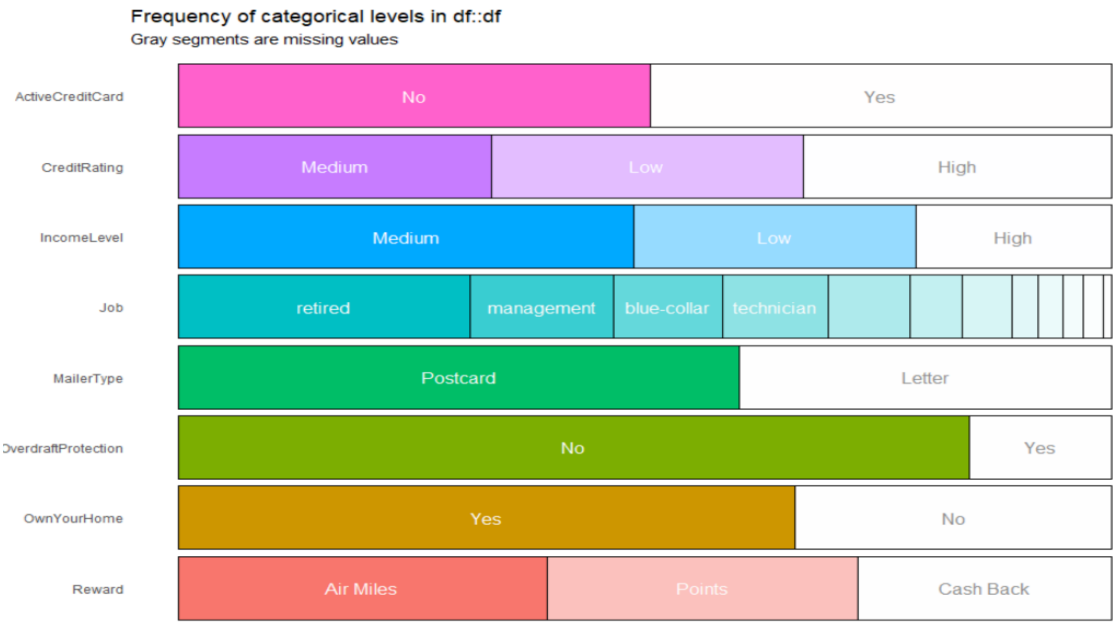
We can see that the dataset is balanced.

After balancing the dataset, we moved to the exploratory data analysis part.

4.3 Exploratory Data Analysis

In this dataset, we have both categorical and numerical columns. Let us look at the values of categorical columns first.

We used library(funmodelling) which is very handy in creating appealing graphs.



We used the following code to create the graphs:

```
#Explore categorical variables
x <- inspect_cat(df)
show_plot(x)

#Get more detail for frequency/count of each level of each categorical variables
freq(df)
```

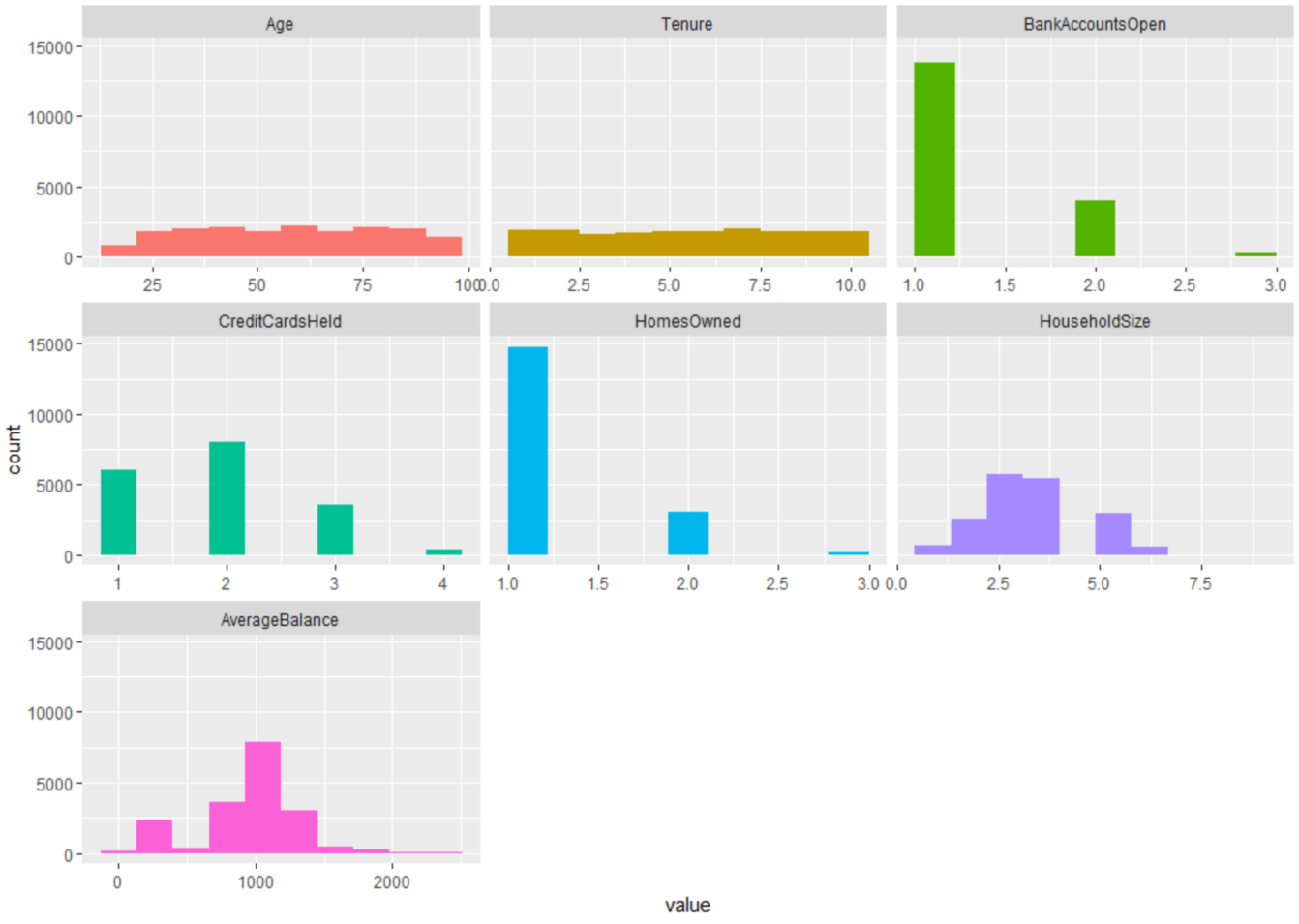
Some observations which we made are as follows:

1. Number of Occupations: Retired and Management occupations are more prevalent in this dataset.

2. Reward: Air Miles reward is preferred.

3. Mailer Type: In this dataset, postcards mailer type is more prevalent.

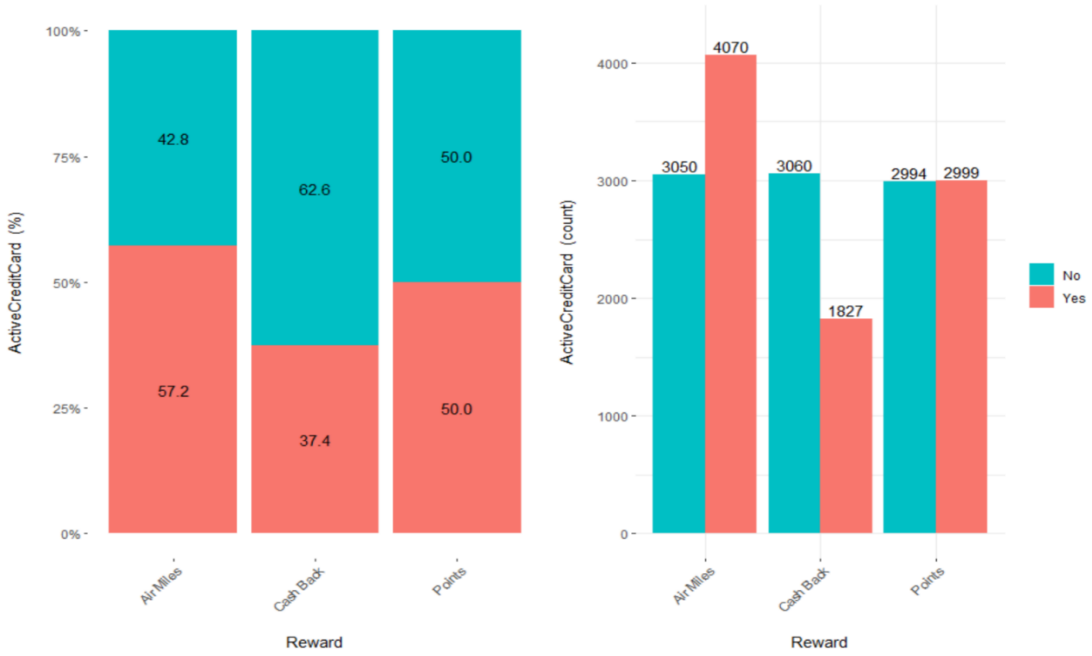
We plot histogram to understand the distribution of numerical variables. We have Age, Tenure, BankAccountsOpen, CreditCardsHeld, HomesOwned, HouseholdSize, and AverageBalance as numerical variables.



4.4 Bivariate Analysis

We explored relationships between our response and potential predictor variables using a cross-plot. This analysis shows potential relationships between activating credit cards and several variables, including Reward, Mailer Type, Income Level, and Credit Rating, etc.

1) Reward

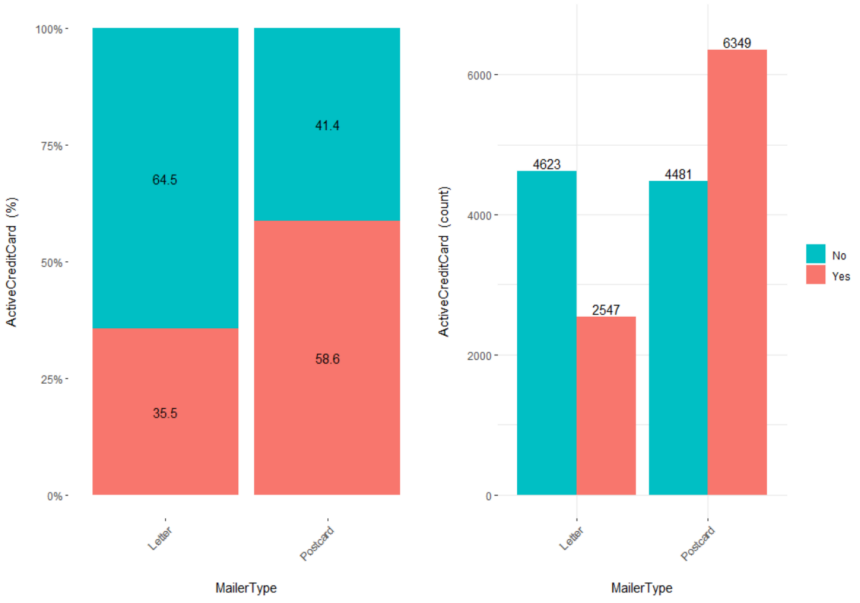


The main reason why we are analyzing rewards is to see if there is any favored reward type and because of that people tend to accept the offer and activate credit cards. There could be external factors that influence the individual to accept or reject the offer. For instance, by looking at the plot above, we can see that the air miles attract customers. However, there are some points we should consider in order to decipher patterns as what is the best reward for the customers, is it the only favored reward type, and so on. By understanding these results, it will help the bank to increase the marketing activity using that campaign to attract potential clients.

Observation:

- The **Air miles** were the most attractive reward, while the CashBack reward was less preferred.
- **Cash Back** may not be a very effective reward to attract customers, only 1827 people would accept a cashback offer, which is only 37.4%.

2) Mailer Type

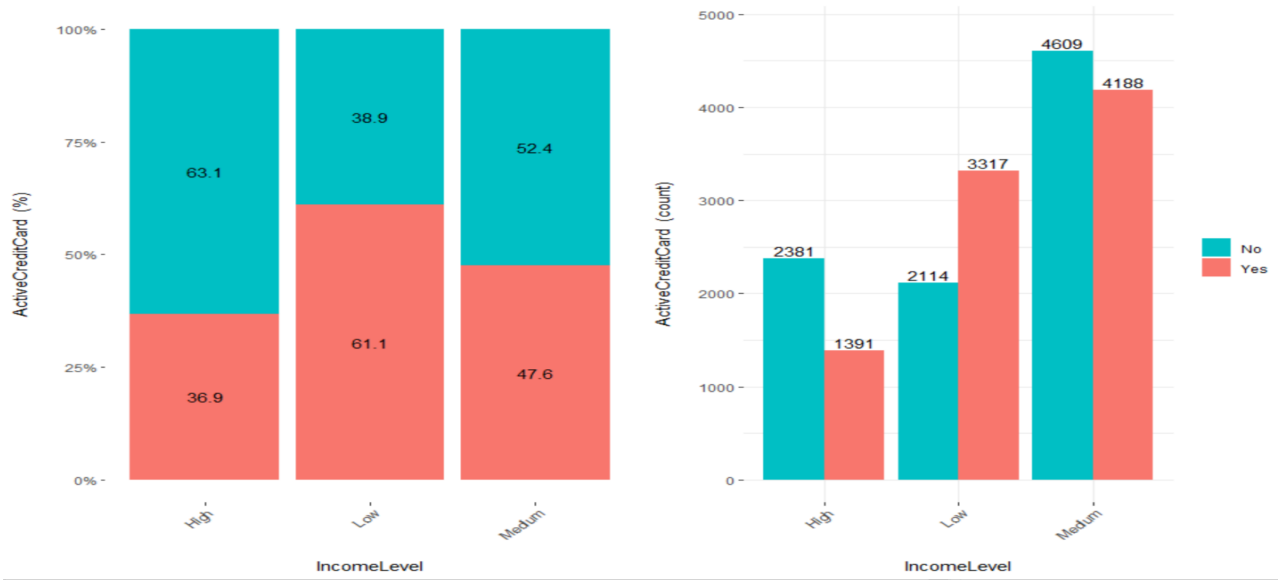


In this section, we visualized which mailer type is effective and how effective were the efforts of marketing people to make these campaigns successful.

Observation:

- Comparing those two ways, individuals opted to accept an offer through the **postcard**.
- It looks like the marketing team of the bank focused its efforts on a postcard campaign to make people accept a credit card offer.

3) Income level

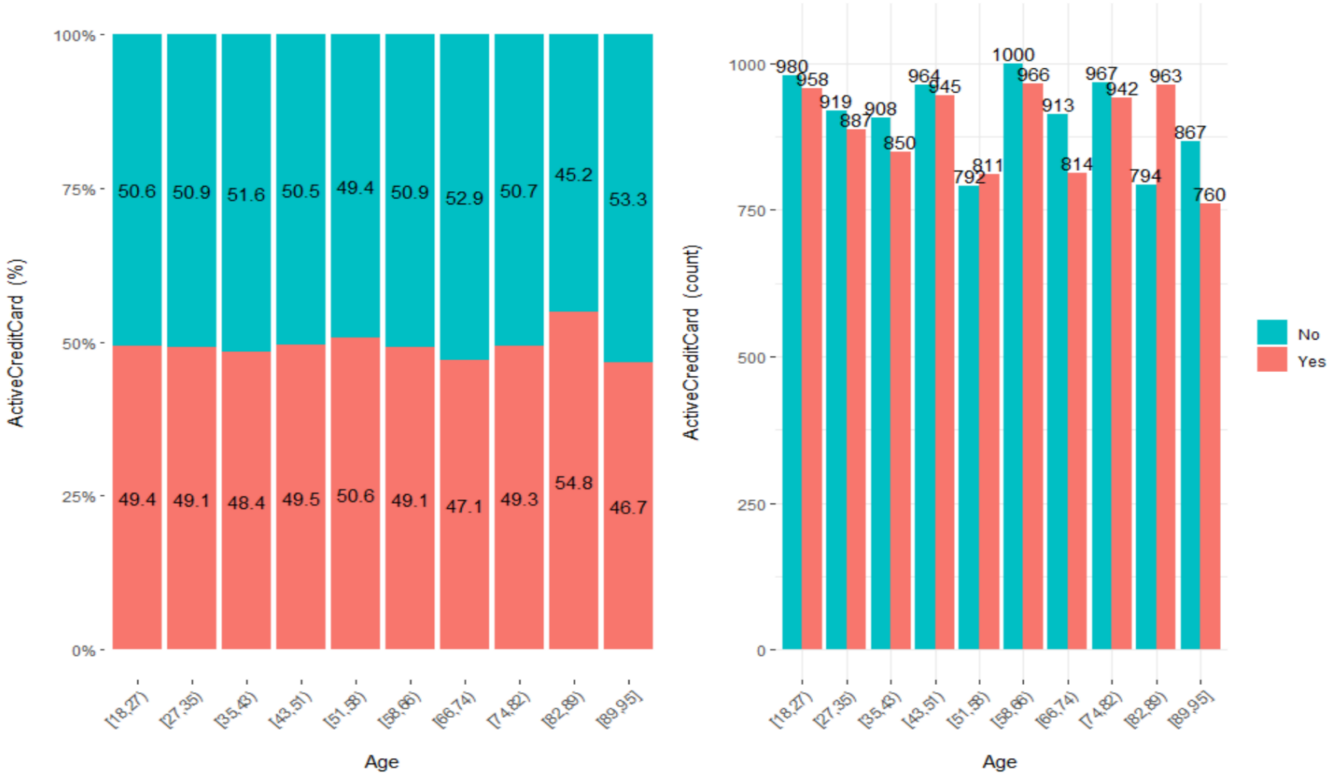


Income also plays a major role in targeting people. This helped to understand who may accept a credit card offer.

Observation:

- People from a low-income level would most likely accept an offer since the credit card could offer them a lot of conveniences and low-income level people may not easily apply for a new credit card.
- Also, medium-level income groups tend to accept the offer. When we compared all three income-level groups, we understand that the maximum number of customers who accepted the offer is from the medium and low-income level group.

4) Age



We created categories that include the range of ages:

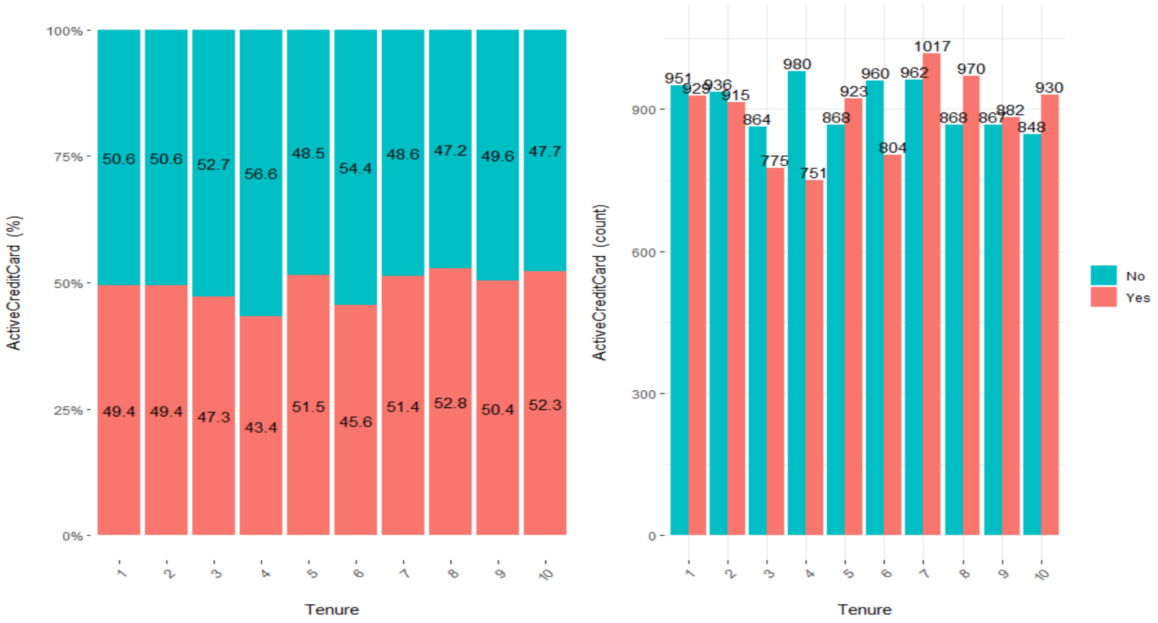
- **20:** This category will include all the ages ranging from 18-27.
- **30:** This category will include all the ages ranging from 27-35.
- **40:** This category will include all the ages ranging from 35-51.
- **50:** This category will include all the ages ranging from 51-58.
- **60:** This category will include all the ages ranging from 59-95. (95 is our maximum age.)

Observation:

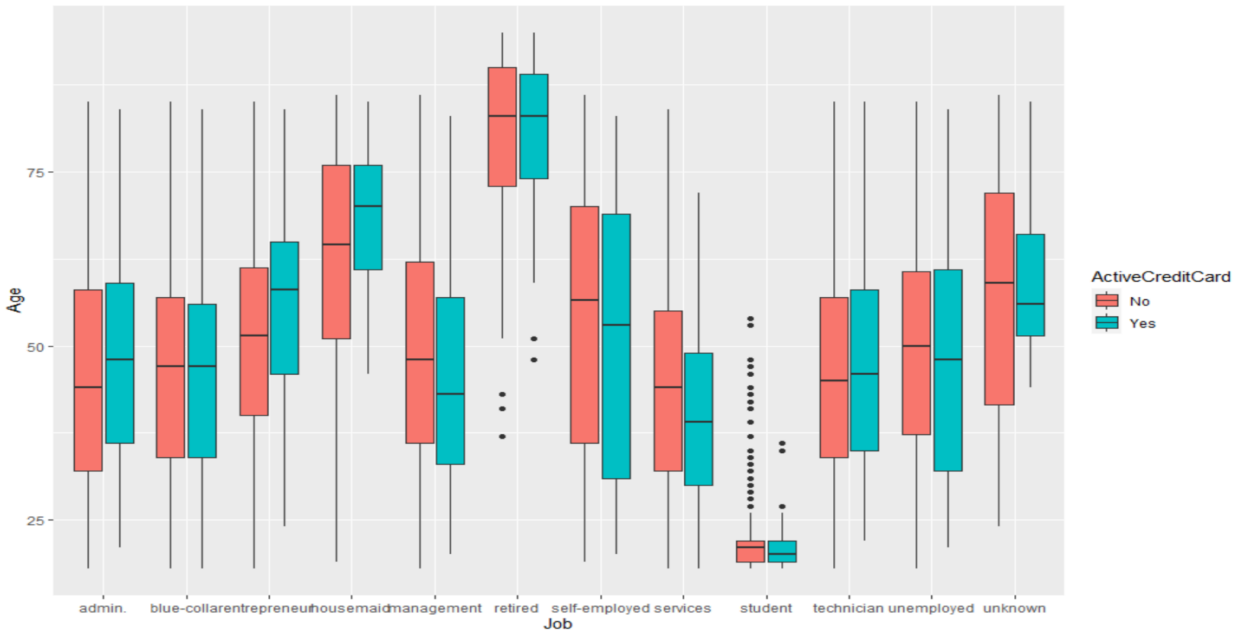
- Most of the potential clients the bank targeted are **18-35,58-66 years old.**
- **60s:** Around 60% of potential clients in this category would accept a credit card during this age.
- **20s,30s:** Around 25% of the potential clients in this category accept a credit card offer
- **40s and 50s:** Around 15% activate their credit cards.

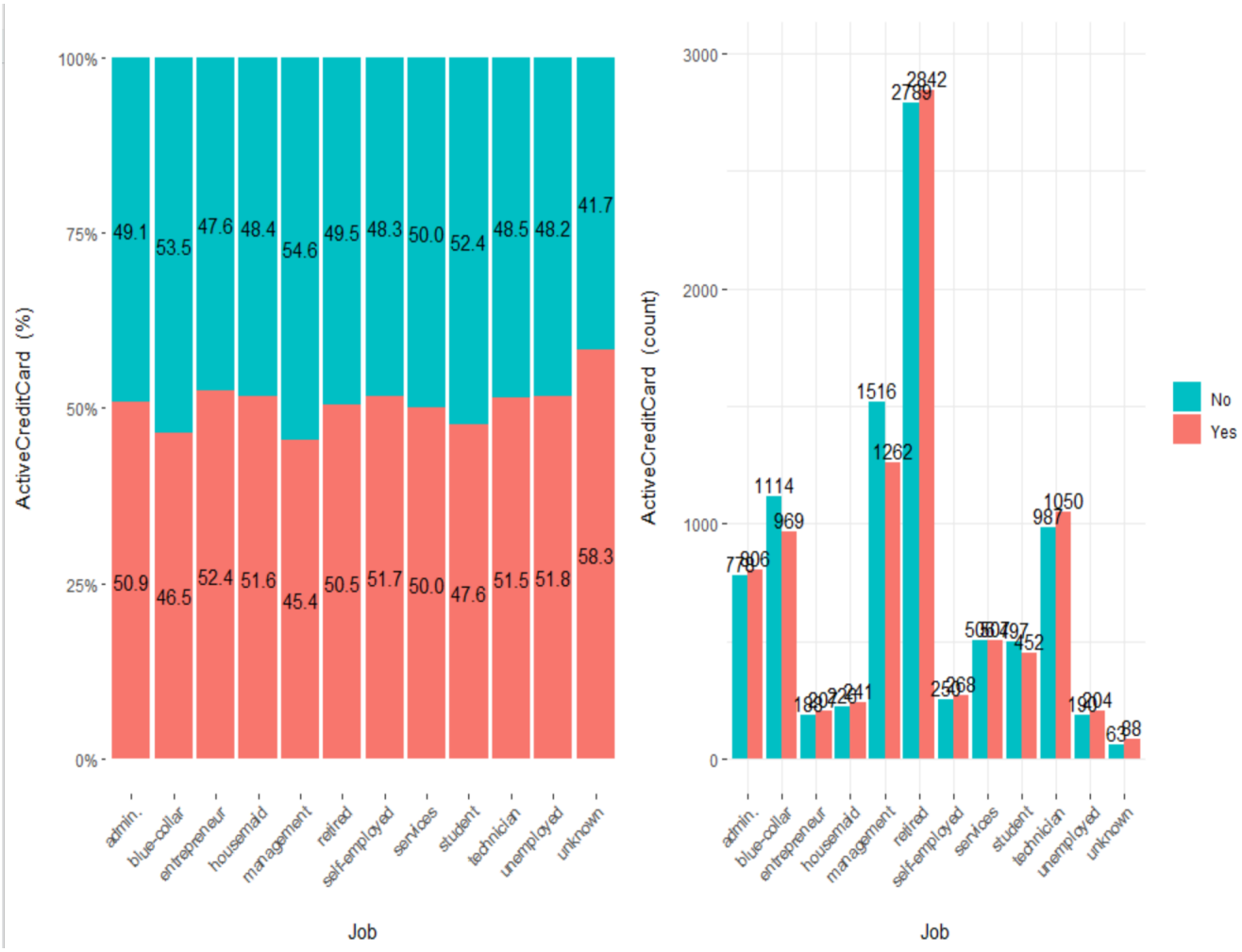
5) Tenure

The customers who are associated with the bank for five years and more, their acceptance rate is higher than the rejection rate.



6) Job (What type of occupation leads to more acceptance of a credit card offer?)

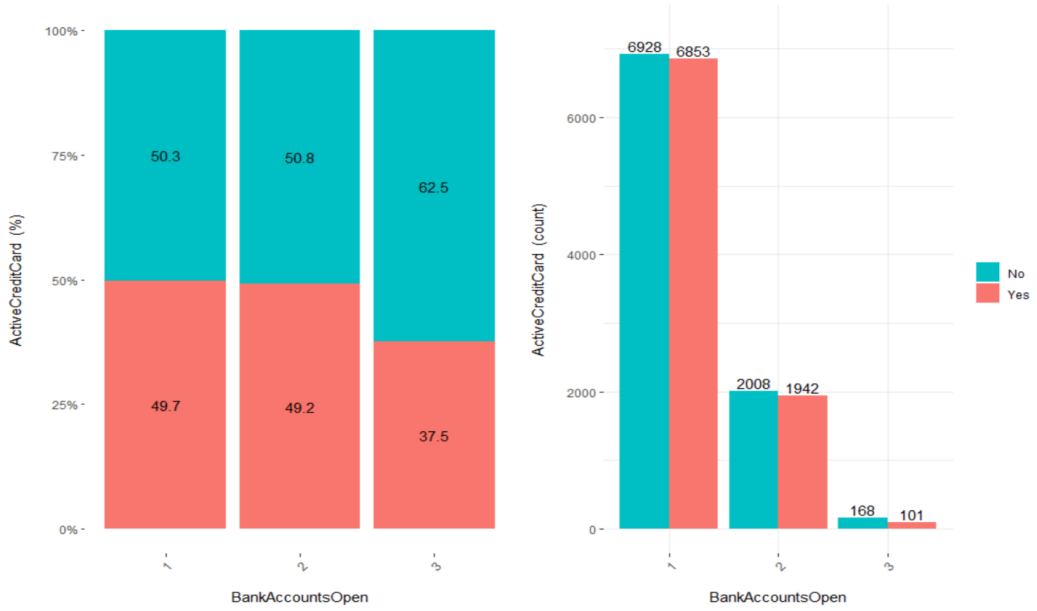




Observations:

- Except for an unknown job, people who are **entrepreneurs, unemployed,** and **self-employed** are most likely to accept the offer and activate a credit card from the marketing campaigns.
- The acceptance rate of **Blue-collar, Students**, and **management** people are less.
- This is a real-life situation. Since in the real world, we do not get detailed information about all the customers. For the customers, whose job profile is unknown, their acceptance rate is **58.3%**.
- **52.4% of people who are entrepreneurs** were willing to activate a credit card (This was also expected since those people's consumption levels may be higher than other jobs).
- **51.8% of the unemployed** were willing to activate credit cards. (People tend to need more money when they are not able to find jobs. Since credit cards are a kind of loan and they can use it for their needs.).
- In the box plot, the potential clients who belong to the entrepreneur category and refused to activate a credit card **were much younger** (median age: Around 54) than the potential clients who accepted to accept a credit card offer (median age: Around 66).

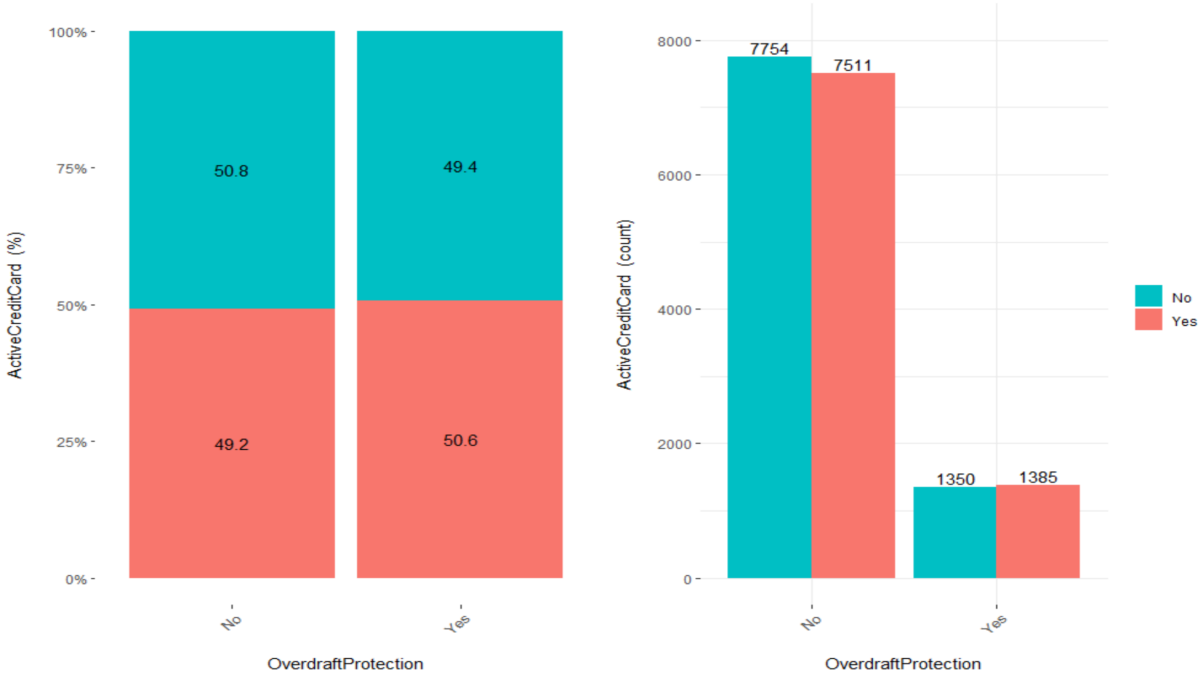
7) Bank Account Open



Observation:

People who have 1 or 2 bank accounts are more likely to accept an offer. Also, in this data set, we only have 269 customers who have 3 accounts, and the conversion rate is only 37.5%.

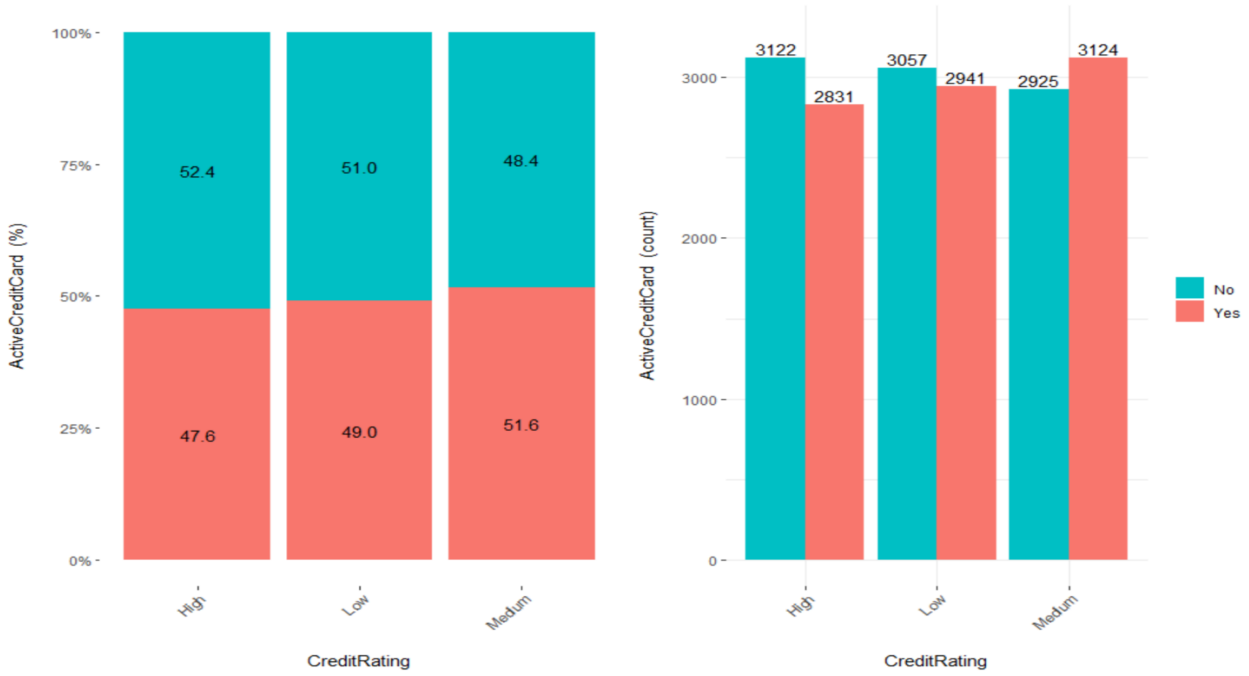
8) Overdraft Protection



Observation:

- The customers who use overdraft protection would likely to activate their credit cards. If we compare the conversion rate, we can see that the conversion rate is higher (50.6%) than those who do not have the facility.
- We have 15,265 customers who have not opted for an overdraft protection facility. So, for such customers, we anticipated that they would reject the offer. But surprisingly, nearly half of the customers accepted the offer (49.2%).

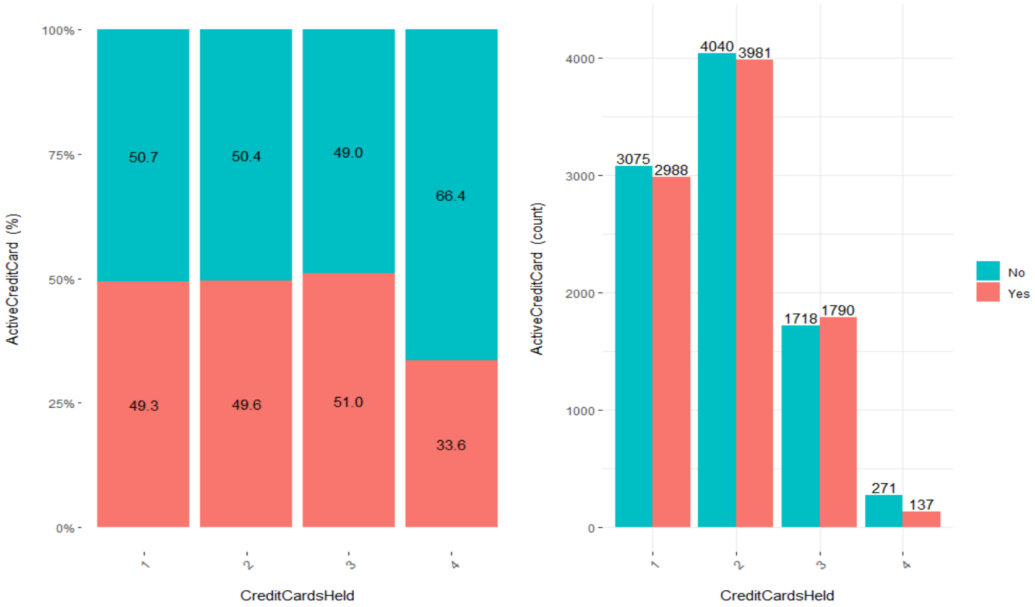
9) Credit Rating



One of the major factors in credit cards is a credit rating. Making a purchase via a credit card is like taking a loan. Creditors review credit scores to gauge how risky a borrower is. The lower the score, the higher the chance that the owner will miss payments. The higher the score, the more likely it is that the owner could manage the card responsibly. We noticed that people who have medium credit ratings would most likely accept an offer.

10) Credit card held

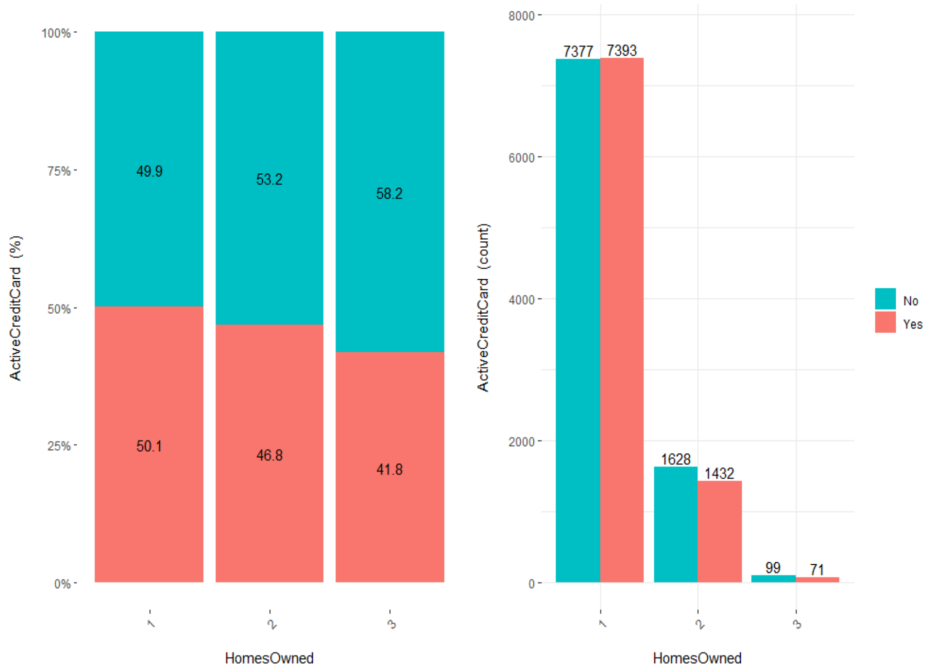
We found that the people who are holding 1 to 3 credit cards are most likely to accept a new credit card. However, people who have 4 may not activate a new credit card, which is reasonable because usually, a person would not have a lot of credit cards.

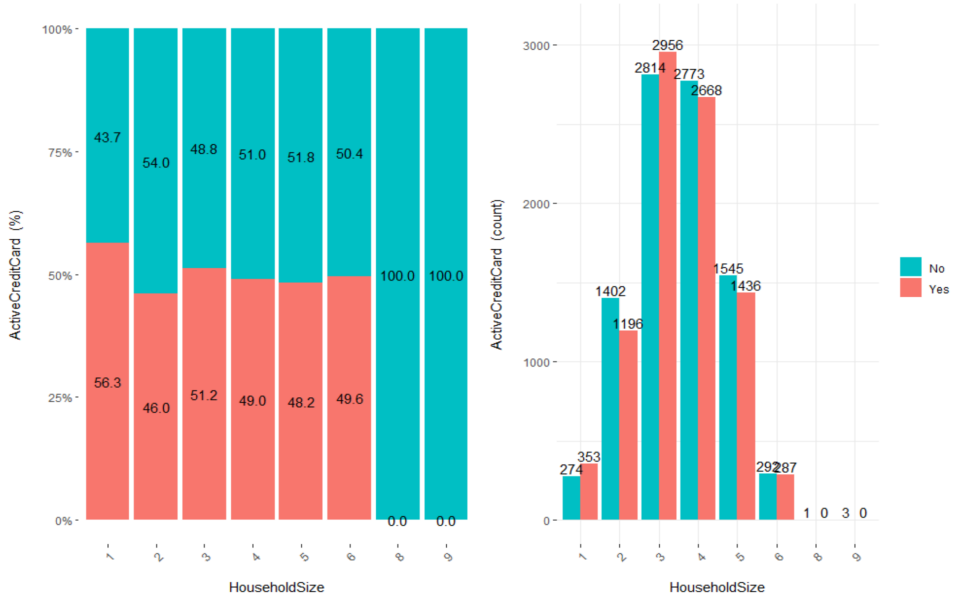


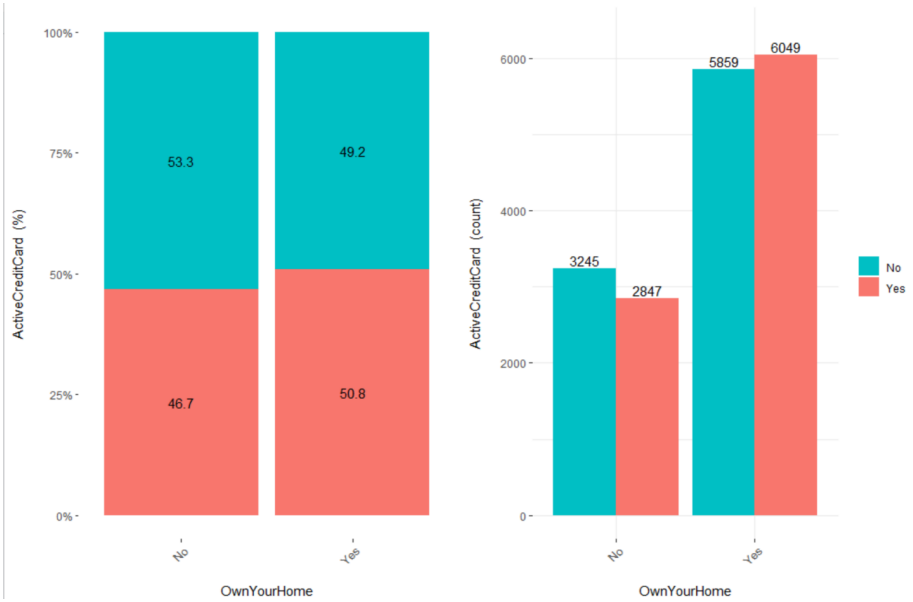
11) House Features

For house features, some observations we made are as follows:

- Homeowned: People who own 1 or 2 houses would likely accept the credit card.
- Householdsize: The number of individuals in the family is 1 or 3 could be the target people to accept an offer.
- Ownyouhome: a person who has his/her own house may accept the offer.

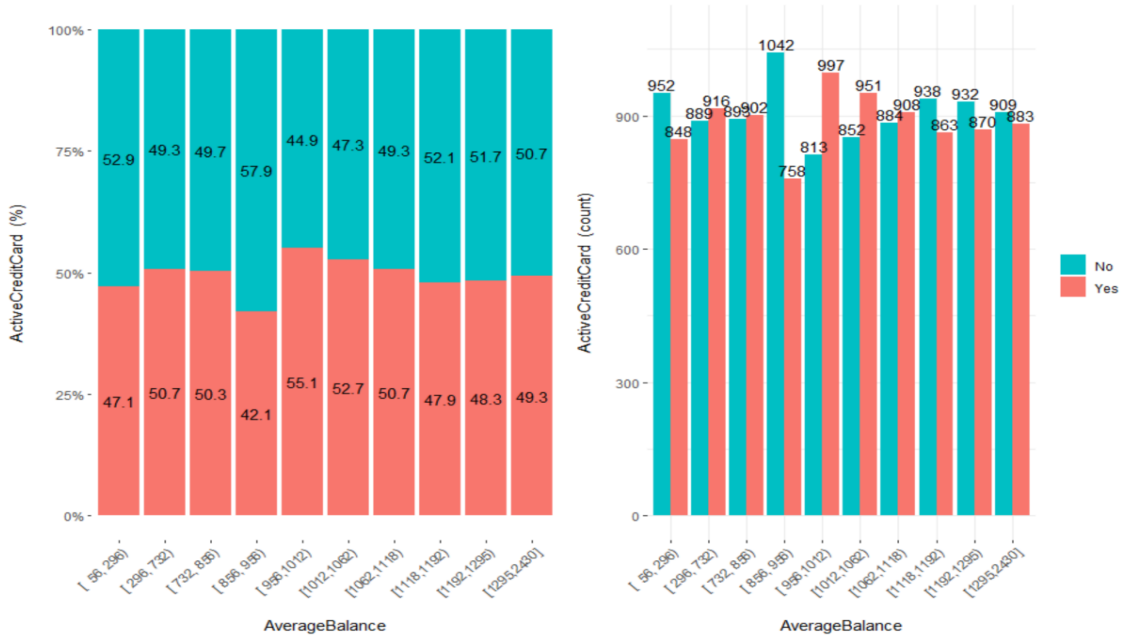






12) Balance

The customers whose balance ranges from 856 – 956 may not likely accept a credit card offer. However, except for this range, there is not a strong relationship between to activate a credit card and balance.



4.5 Statistical Modeling and Data Analysis

After cleaning, data preparation, and balancing the data set, the dataset contains **18000 observations** and **15 variables**. ***ActiveCreditCard*** is a response variable.

```
> str(df)
'data.frame':   18000 obs. of  15 variables:
 $ ActiveCreditCard   : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
 $ Reward             : chr  "Air Miles" "Points" "Points" "Points" ...
 $ MailerType         : chr  "Letter" "Postcard" "Letter" "Postcard" ...
 $ IncomeLevel        : chr  "Medium" "High" "Low" "Medium" ...
 $ Age                : num  84 25 50 83 50 24 87 56 71 84 ...
 $ Tenure             : num  2 8 7 1 4 10 5 3 8 6 ...
 $ Job                : chr  "technician" "services" "technician" "retired" ...
 $ BankAccountsOpen   : num  1 1 1 1 1 1 1 1 2 2 ...
 $ OverdraftProtection: chr  "Yes" "No" "No" "No" ...
 $ CreditRating       : chr  "High" "Medium" "Low" "High" ...
 $ CreditCardsHeld    : num  2 2 2 3 2 2 1 1 4 2 ...
 $ HomesOwned         : num  1 1 1 1 1 1 1 1 1 1 ...
 $ HouseholdSize      : num  3 1 3 3 5 3 2 5 4 6 ...
 $ OwnYourHome        : chr  "No" "Yes" "No" "Yes" ...
 $ AverageBalance     : num  705 136 1230 1262 1080 ...
```

We divided the dataset into a *75:25* ratio.

```
set.seed(2020)
train_indices <- sample(1:nrow(df), 0.75*nrow(df))
df_train <- df[train_indices, ]
df_test <- df[-train_indices, ]
```

After creating training and test sets, the training set contains *13500 observations*, and the test set contains *4500 observations*.

```
> NROW(df_train)
[1] 13500
> NROW(df_test)
[1] 4500
```

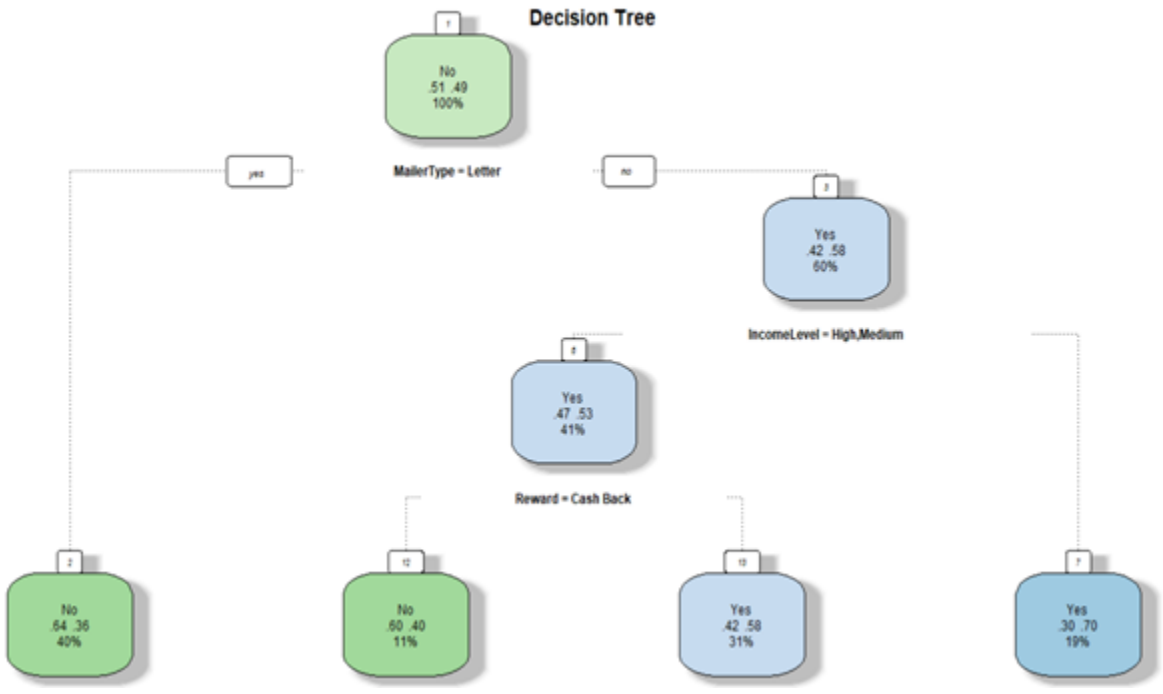
To predict which customers will accept the offer and activate the credit cards and who will decline the offer, we build three machine learning models, **Decision Tree, Naïve Bayes,** and **Random Forest**. We evaluated the performance of the models and used the model with the highest accuracy to predict the results.

We chose these models because of the structure of our dataset. Our dataset contains both, categorical and numerical variables. We tried to analyze which factors are influencing the customer's decisions in accepting or rejecting the offer. With the help of this analysis, we offered strategies and solutions.

We build the model on the training set and used the test set to make the predictions.

Decision Tree

We used rpart() function to build a model and fancyRpartPlot() function to build a graph.



We can start with the root node.

1. At the top, it is the overall probability of users who will not accept the offer. It shows that 51% will reject the offer while 49% will accept the offer and activate a credit card.

2. This node asks whether the MailerType is Letter. If yes, then it goes to the second node, where there is a probability of 40% that 64% of customers reject the offer.

3. If MailerType is not Letter, it goes to the third node where the predicted class is Yes. It says there is a probability of 47 percent where customers will accept the offer while a 53% chance that customer will reject the offer.

4. In this node, it asks about the Income Level. It creates two Yes nodes. At the first node, it says there is a 19% chance that 30% of customers will accept the offer. We can see the difference between acceptance and rejection percentage.

5. For income level, it creates another Yes node of Reward = Cash Back. If the reward type is cashback, then it suggests 42% of customers will accept the offer with a probability of 31%. If the reward type is other than Cash Back, it suggests that 60% of users will decline the offer.

6. The total nodes created by the model is 13. If we keep on going like this, we will understand which features impact the likelihood of acceptance or rejection of the offer.

The following are the snippets of the nodes generated by the model:

```
Node number 1: 13500 observations,    complexity param=0.2061732
  predicted class=No   expected loss=0.4943704  P(node) =1
    class counts:  6826  6674
   probabilities: 0.506 0.494
  left son=2 (5376 obs) right son=3 (8124 obs)
  Primary splits:
      MailerType      splits as  LR,           improve=332.82290, (0 missing)
      IncomeLevel     splits as  LRL,          improve=163.56890, (0 missing)
      Reward          splits as  RLR,          improve=151.04030, (0 missing)
      CreditCardsHeld < 3.5     to the right,  improve= 16.62708, (0 missing)
      Job             splits as  RLRRLRRRLRRR, improve= 12.73142, (0 missing)
  Surrogate splits:
      AverageBalance < 106.5   to the left,  agree=0.602, adj=0.002, (0 split)
      HouseholdSize  < 8.5     to the right, agree=0.602, adj=0.000, (0 split)
```

At node =1, the number of observations is 13500. Primary splits are at MailerType, IncomeLevel, Reward, CreditCardsHeld, and Job. The predicted class is No with the probability of rejecting and accepting an offer is 50.6% and 49.4%, respectively.

Similarly, at node = 2 number of observations is 5376 and the predicted class is No. If the decision at the first node is yes, that means there is a 40% chance that 64% of customers reject the offer.

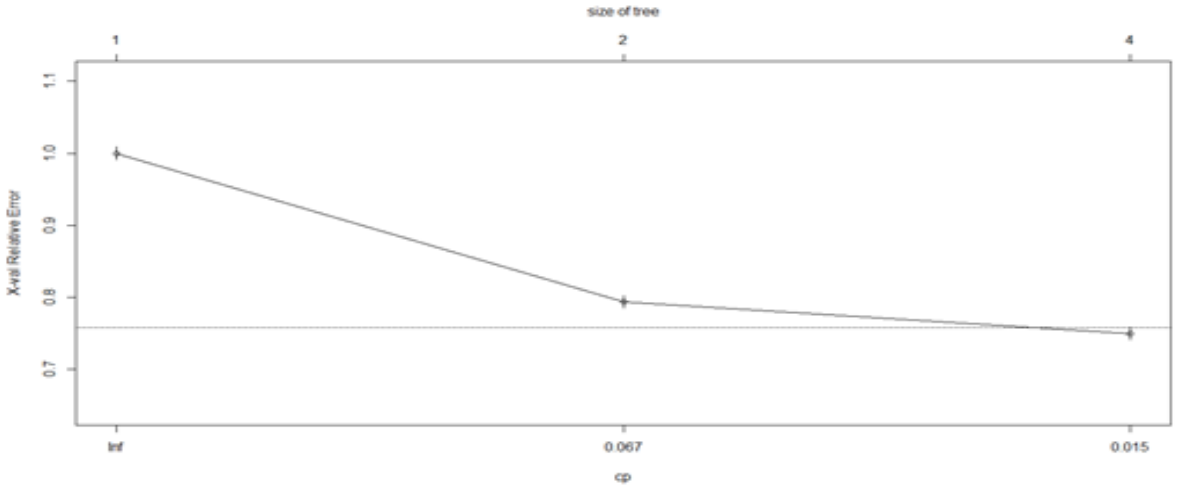
```
Node number 2: 5376 observations
  predicted class=No   expected loss=0.3578869  P(node) =0.3982222
    class counts:  3452  1924
   probabilities: 0.642 0.358
```

At the 13th node, there are 4157 observations, and the predicted class is Yes. If we track this node back, we can understand that the primary split is at node=6 where it says if Reward= CashBack, it will go to node=13 where the probability of 42% customers will accept the offer 31%.

```
Node number 13: 4157 observations
  predicted class=Yes  expected loss=0.4204955  P(node) =0.3079259
    class counts:  1748  2409
   probabilities: 0.420 0.580
```

We also checked the Complexity Parameter of the tree. To find the CP value we used printcp() function and the result are as follows:

```
> printcp(model_tree)

Classification tree:
rpart(formula = ActiveCreditCard ~ ., data = df_train, method = "class")

Variables actually used in tree construction:
[1] IncomeLevel MailerType  Reward

Root node error: 6674/13500 = 0.49437

n= 13500

        CP nsplit rel error  xerror      xstd
1 0.206173      0   1.00000 1.00000 0.0087041
2 0.021801      1   0.79383 0.79383 0.0085009
3 0.010000      3   0.75022 0.75022 0.0084094
```



The one with the least cross-validated error (xerror) is the optimal value of CP given by the printcp() function. Here we can see that the least xerror is 0.75022 corresponding to 0.015 CP value approximately equal to 0.01. To confirm the value, we used the following code:

```
> #Confirming optimal CP value
> cp= model_tree$cptable[which.min(model_tree$cptable[,"xerror"]),"CP"]
> cp
[1] 0.01
```

We can observe that CP = 0.01 which is a default value. So, pruning is not required.

Naïve Bayes

Naïve Bayes called Naïve because it is (almost) never true. In other words, the Naive Bayes classifier assumes that the predictors are independent of each other. This assumption is called class conditional independence.

The Naive Bayes model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

We build the model using **Kernel** to get a better performing model.

```
> #Naive Bayes
> model_naive <- naive_bayes(ActiveCreditCard ~., data = df_train, usekernel = T)
> model_naive
```

Since we have 15 variables in the dataset, the following is not a complete output. Just a snippet.

```
================================== Naive Bayes ==================================

 Call:
naive_bayes.formula(formula = ActiveCreditCard ~ ., data = df_train,
    usekernel = T)

---------------------------------------------------------------------------------

Laplace smoothing: 0

---------------------------------------------------------------------------------

 A priori probabilities:

       No       Yes
0.5056296 0.4943704
```

In the training set, we have **50.56%** points belonging to **No**, meaning customers ***will reject the offer*** while **49.43%** of customers ***will accept the offer***.

We dig deeper to understand which customers the bank should target, we found that customers with a medium-income level have a high probability (47.19%) of accepting an offer than the other two income groups. Interestingly, customers with a high-income level have the lowest acceptance possibility (15.50%).

Also, we found that customers prefer Postcards over letters. Furthermore, if an Air Miles reward is offered, there are high chances that the customer will accept the offer.

```
-------------------------------------------------------------------------------
 ::: Reward (Categorical)
-------------------------------------------------------------------------------

Reward             No       Yes
  Air Miles 0.3348960 0.4554990
  Cash Back 0.3369470 0.2039257
  Points    0.3281570 0.3405754

-------------------------------------------------------------------------------
 ::: MailerType (Bernoulli)
-------------------------------------------------------------------------------

MailerType        No       Yes
  Letter   0.5057134 0.2882829
  Postcard 0.4942866 0.7117171

-------------------------------------------------------------------------------
 ::: IncomeLevel (Categorical)
-------------------------------------------------------------------------------

IncomeLevel        No       Yes
     High   0.2609142 0.1550794
     Low    0.2301494 0.3729398
     Medium 0.5089364 0.4719808
```

Random Forest

Random Forest is a supervised learning algorithm that can be used for both classifications and regression tasks. However, it is mainly used for classification problems. As the name suggests, it consists of many individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction.

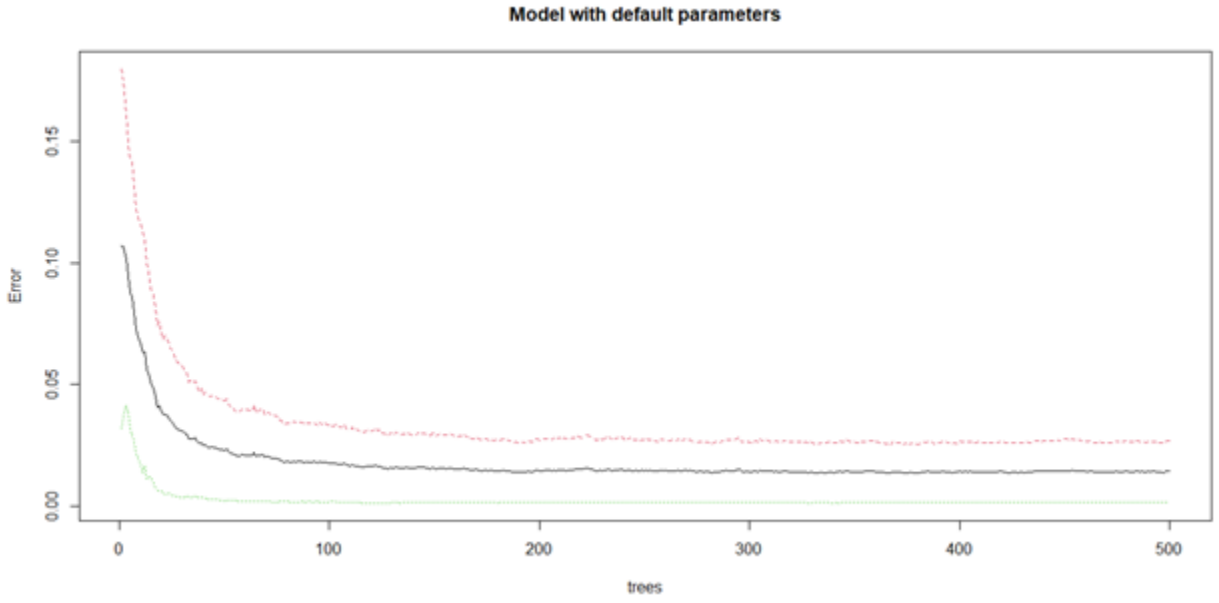
We build a Random Forest model with default parameters.

```
#Random Forest
model_rf1 <- randomForest(ActiveCreditCard~., data = df_train, importance = TRUE)
```

The results are as follows:

```
> model_rf1

Call:
 randomForest(formula = ActiveCreditCard ~ ., data = df_train,      importance = TRUE)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 3

        OOB estimate of  error rate: 1.44%
Confusion matrix:
      No  Yes class.error
No  6642  184 0.026955757
Yes   10 6664 0.001498352
```

By default, the ***number of trees*** is ***500*** and the number of variables tried at each split is ***3*** in this case. The ***error rate*** is ***1.44%***.



We can tune the random forest model by changing the number of trees (ntree) and the number of variables randomly sampled at each stage (mtry). According to the Random Forest package description:

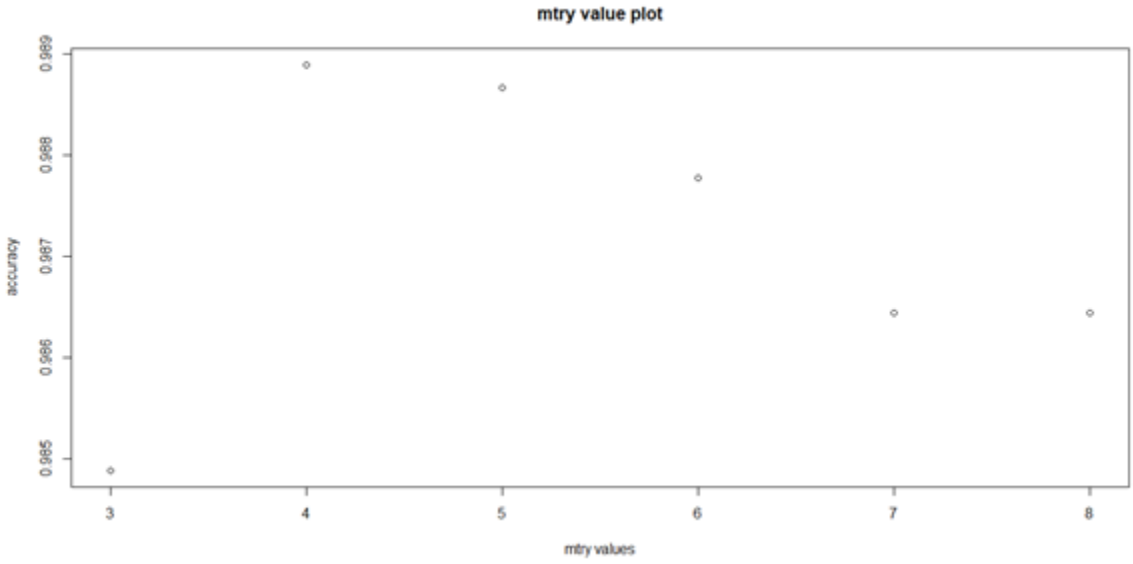
Ntree: Number of trees to grow. This should not be set to too small a number, to ensure that every input row gets predicted at least a few times.

Mtry: Number of variables randomly sampled as candidates at each split. Note that the default values are different for classification (sqrt(p) where p is the number of variables in x) and regression (p/3).

We used a loop to identify the optimal mtry value for the model keeping ntree = 500.

```
#Using loop to identify the optimal mtry value for model
set.seed(123)
acc_check = c()
i= 5
for (i in 3:8) {
  ran_model<- randomForest(ActiveCreditCard ~ ., data = df_train, ntree = 500, mtry = i, importance = TRUE)
  ran_pred <- predict(ran_model, df_test, type = "class")
  acc_check[i-2] <- mean(ran_pred == df_test$ActiveCreditCard)
}
```

We found that at mtry = 4, the model achieved the highest accuracy of 98.88%(rounded).



From the above graph, we can observe that the accuracy increased when mtry was increased from 3 to 4. It keeps on decreasing when mtry was changed from 4. The model achieves the highest accuracy at mtry = 4.

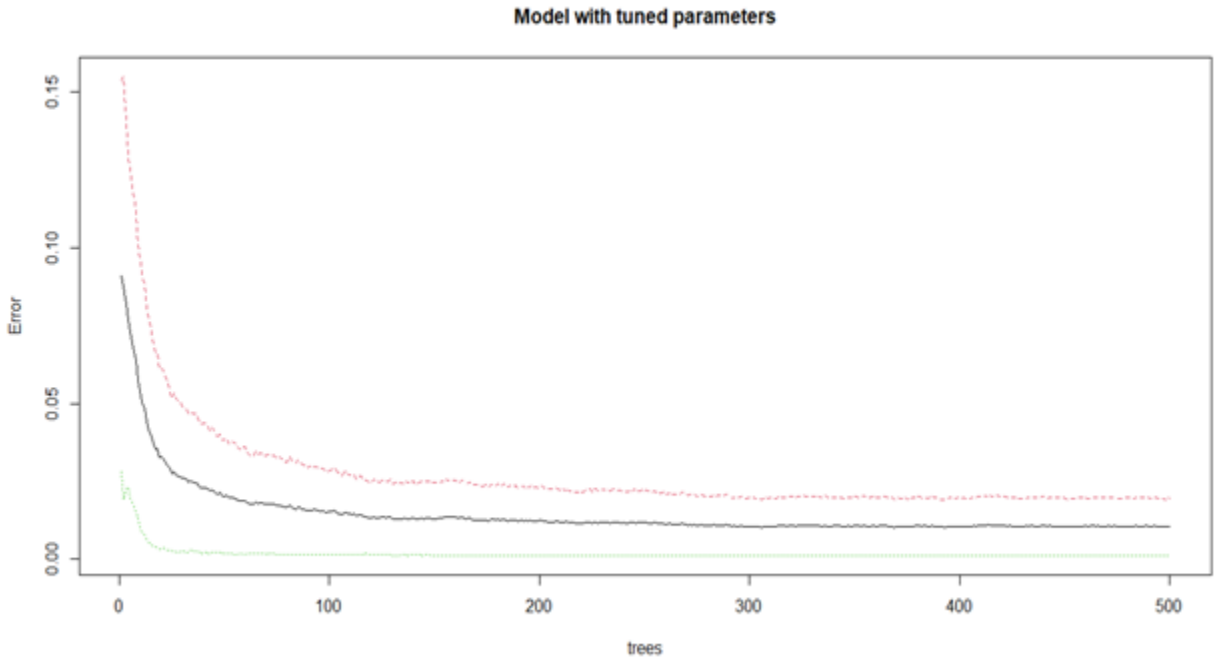
We build a model using mtry = 4 and ntree = 500.

```
#Tuning model with parameters
model_rf2 <- randomForest(ActiveCreditCard~., data = df_train, ntree = 500, mtry = 4,importance = TRUE)
```

The results are as follows:

```
> model_rf2

Call:
 randomForest(formula = ActiveCreditCard ~ ., data = df_train,      ntree = 500, mtry = 4, importance = TRUE)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 4

        OOB estimate of  error rate: 1.04%
Confusion matrix:
      No  Yes class.error
No  6693  133 0.019484325
Yes    8 6666 0.001198681
```

When we tune the mode with mtry = 4, we observed that the OOB error rate is reduced from 1.44% to 1.04%. Model 2 gave us a 1. 04% OOB error estimate rate. That means 98.96% of the OOB samples were correctly classified by the random forest. So, we decided to use this model for further evaluations.

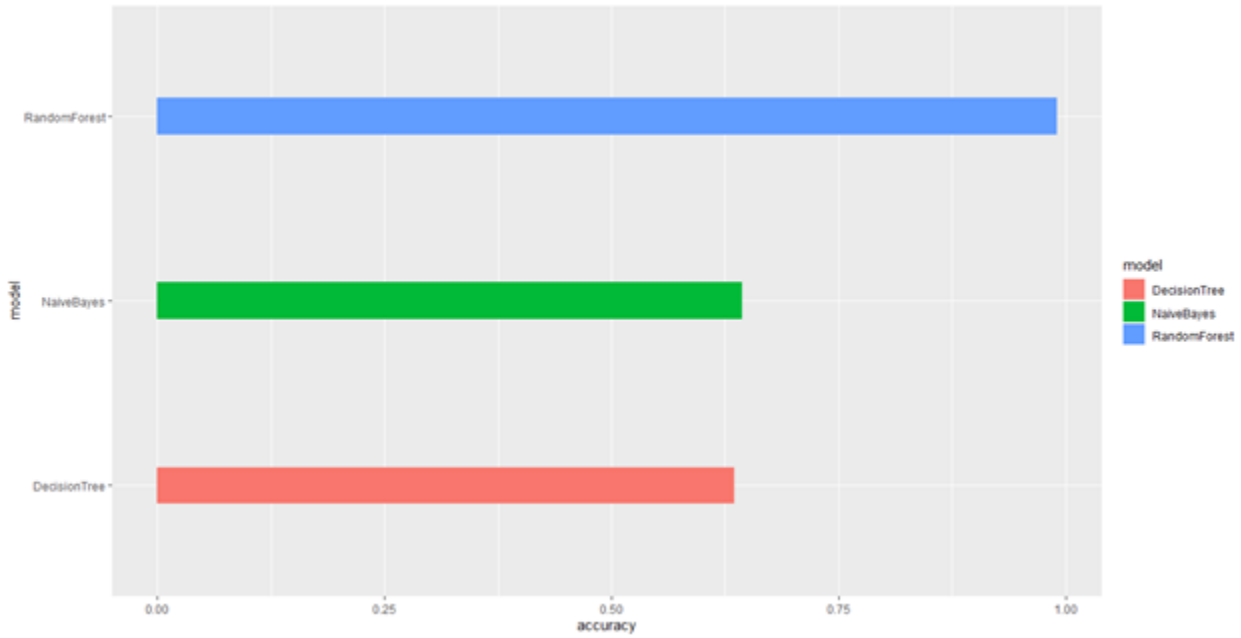


We compared the accuracy and performance of each model by creating a data frame. To evaluate the model's performance, we used a confusion matrix.

```
#creating a data frame for putting every model's accuracy together
model<-c()
accuracy<-c()
```

We used a test set to evaluate the model's performance. We have ***4500 observations*** in the test set. The results are as follows:

Model	Correct Classifications	Misclassifications	Accuracy (in %)
Decision tree	1467 + 1394 = 2861	828+811 = 1639	63.58
Naïve Bayes	1354 + 1544 = 2898	678 + 924 = 1602	64.4
Random Forest	2233 + 2222 = 4455	45 + 0 = 45	98.88



4.6 Predict who will accept or reject the offer

According to the machine learning models, we have studied; the Random Forest gave us the lowest error rate when predicting acceptance or rejection of an offer. So, we used the Random Forest model for predictions. The results are as follows:

	pred_rf	ActiveCreditCard	Reward	MailerType	IncomeLevel	Age	Tenure
2	No	No	Points	Postcard	High	25	8
3	No	No	Points	Letter	Low	50	7
7	No	No	Points	Letter	Medium	87	5
16	Yes	No	Points	Postcard	Low	73	5
19	No	No	Points	Postcard	Low	74	8
22	No	No	Air Miles	Postcard	Low	40	5

	Job	BankAccountsOpen	OverdraftProtection	CreditRating	CreditCardsHeld
2	services	1	No	Medium	2
3	technician	1	No	Low	2
7	retired	1	No	Medium	1
16	retired	1	No	Medium	2
19	technician	1	No	Medium	1
22	entrepreneur	1	No	High	1

	HomesOwned	HouseholdSize	OwnYourHome	AverageBalance
2	1	1	Yes	136.50
3	1	3	No	1230.00
7	1	2	Yes	1041.50
16	1	3	Yes	1060.25
19	1	4	Yes	1328.00
22	1	2	Yes	384.00

4.7 Strategies (What Actions should the Bank Consider?)

Solutions for the Next Marketing Campaign:

1) **Reward:** We observe that the most attractive reward was **Air Miles**. On the other hand, cashback has the lowest effective rate: 20.39%. For the next marketing campaign, it will be wise for the bank to focus on **Air Miles** and **Points** as a reward to increase the active credit card rate.

2) **Mailer Type:** Potential clients chose to accept an offer through a **postcard**. So, for the next marketing campaign, the bank should focus on using this channel.

3) **Income Level:** The bank can put more effort into targeting **low and medium level income group customers** to get more credit card customers. At the same time, to save time and effort, employees in banks can avoid pitching a credit card to high-income level groups. But we also noticed some exceptions for the high-income level groups.

4) **Age Category:** The next marketing campaign of the bank should target potential clients from **18-35 years old**, and **over 58 years old**. The elderly category has a 60% chance of activating a credit card while the young category has a 24% chance of accepting an offer. It will be great if the next campaign of the bank addressed these two categories and therefore, increased the likelihood of new clients.

5) Tenure: The customers who are associated with the bank for five years and more, their acceptance rate is higher than the rejection rate. The bank can pay more focus on customers who have been with the bank for more than 5 years to increase the activation rate.

6) Occupation: From the analysis, we understand that potential clients are entrepreneurs, and self-employed. These two categories were the most likely to accept a new credit card. For example, entrepreneurs tend to have more opportunities to use money and a credit card is a short-term 0% interest loan in which the individual agrees to use this card to purchase things and pay the bill on time. Entrepreneurs and self-employed may tend to spend a large amount of money so they will be interested in having a free interest credit card.

7) Bank services: People who have **1 or 2 bank accounts** would most likely accept an offer. People who **use overdraft protection** would also likely activate their credit cards. The people who are holding **1 to 3 credit cards** are more likely to accept a new credit card. However, people who have 4 cards may not activate a credit card, which is reasonable because usually, a person does not want a lot of credit cards. So, it is a good idea to pay more attention to those customers who have above characteristics and pitch them a new offer.

8) Credit rating: People who have medium credit ratings would most likely accept an offer. It will save some marketing costs if the next campaign of the bank focuses more on medium credit rating holders.

9) House ownership and Balances: There is not a strong relationship between house ownership, balance and accepting offers by the bank. We might need more data to analyze this relationship.

10) Increase the frequency of contact with potential customers: Usually, marketing campaigns would not happen just once. The more we reach out to customers, the more likely for a customer to activate a credit card. When contacting potential clients, a bank can provide an interesting questionnaire for them during the marketing campaign and give them presents to appreciate them. This way might help the bank to better understand what customers desire.

By combining all these strategies and streamlining the target audience the bank should plan the next campaign which will prove to be more effective.

5. Part II

5.1 Data preprocessing

As shown in the first 6 rows of the dataset, this data set has 12 columns.

```
  RowNumber  Surname CreditScore Gender Age Tenure   Balance NumOfProducts HasCrCard UseFrequency EstimatedSalary Churn
1         1 Hargrave         619 Female  42      2      0.00             1         1            1       101348.88     1
2         2     Hill         608 Female  41      1  83807.86             1         0            1       112542.58     0
3         3     Onio         502 Female  42      8 159660.80             3         1            0       113931.57     1
4         4     Boni         699 Female  39      1      0.00             2         0            0        93826.63     0
5         5 Mitchell         850 Female  43      2 125510.82             1         1            1        79084.10     0
6         6      Chu         645   Male  44      8 113755.78             2         1            0       149756.71     1
```

any(is.na()) function was used to check the null values in the dataset. As the result showed, there are no null values in this data.

```
> any(is.na(df)) # there is no null in this dataset
[1] FALSE
```

```
'data.frame':   10000 obs. of  12 variables:
 $ RowNumber      : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Surname        : Factor w/ 2932 levels "Abazu","Abbie",..: 1116 1178 2041 290 1823 538 178 2001 1147 1082 ...
 $ CreditScore    : int  619 608 502 699 850 645 822 376 501 684 ...
 $ Gender         : Factor w/ 2 levels "Female","Male": 1 1 1 1 1 2 2 1 2 2 ...
 $ Age            : int  42 41 42 39 43 44 50 29 44 27 ...
 $ Tenure         : int  2 1 8 1 2 8 7 4 4 2 ...
 $ Balance        : num  0 83808 159661 0 125511 ...
 $ NumOfProducts  : int  1 1 3 2 1 2 2 4 2 1 ...
 $ HasCrCard      : int  1 0 1 0 1 1 1 1 0 1 ...
 $ UseFrequency   : int  1 1 0 0 1 0 1 0 1 1 ...
 $ EstimatedSalary: num  101349 112543 113932 93827 79084 ...
 $ Churn          : int  1 0 1 0 0 1 0 1 0 0 ...
```

The data type of HasCrCard, UseFrequency, and Churn is an integer. But they are categorical data. For example, when UseFrequency equals 0 means the user is not actively using the bank products, while UseFrequency equals 1 means the user is actively using the bank products. In this case, they will be changed to factor after visualizing the heatmap by the following code:

```
> # Convert churn,HasCrCard,UseFrequency to factor and  0,1 to No,Yes
> df[df$Churn==1,]$Churn<-'Yes'
> df[df$Churn==0,]$Churn<-'No'
> df$Churn<-as.factor(df$Churn)
>
> df[df$HasCrCard==1,]$HasCrCard<-'Yes'
> df[df$HasCrCard==0,]$HasCrCard<-'No'
> df$HasCrCard<-as.factor(df$HasCrCard)
>
> df[df$UseFrequency==1,]$UseFrequency<-'High'
> df[df$UseFrequency==0,]$UseFrequency<-'Low'
> df$UseFrequency<-as.factor(df$UseFrequency)
```

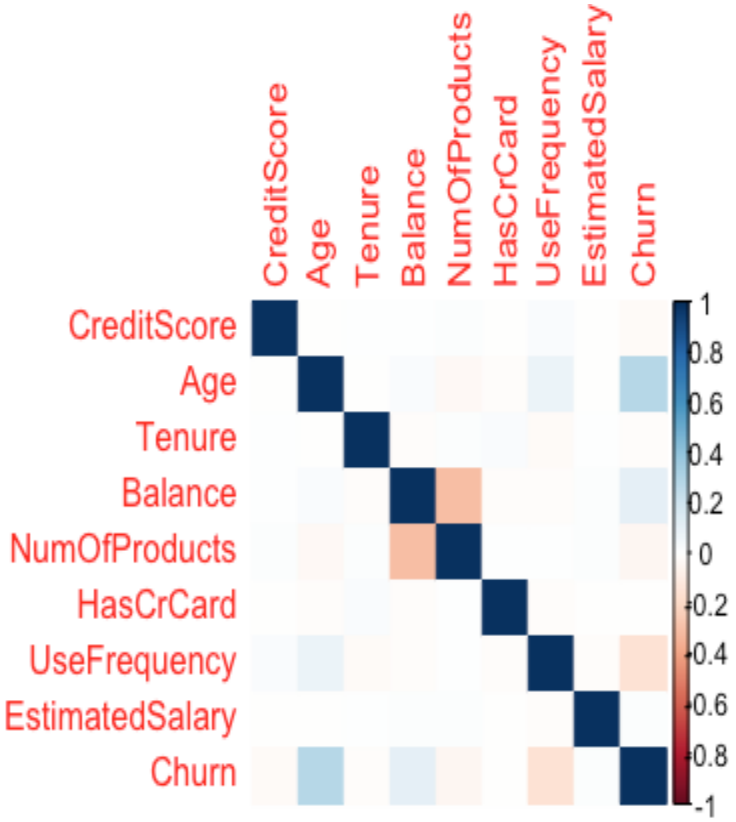
The RowNumber column and Surname column are irrelevant with this churn analysis, so these two columns were dropped.

```
> # drop the inrelevant column RowNumber and Surname
> df<-df[-c(1,2)]
> head(df)
  CreditScore Gender Age Tenure   Balance NumOfProducts HasCrCard UseFrequency EstimatedSalary Churn
1         619 Female  42      2      0.00             1       Yes          High       101348.88   Yes
2         608 Female  41      1  83807.86             1        No          High       112542.58    No
3         502 Female  42      8 159660.80             3       Yes           Low       113931.57   Yes
4         699 Female  39      1      0.00             2        No           Low        93826.63    No
5         850 Female  43      2 125510.82             1       Yes          High        79084.10    No
6         645   Male  44      8 113755.78             2       Yes           Low       149756.71   Yes
```

After cleaning the data, the library 'ggplot2' was used to visualize the relationship between the churn column and other columns.

5.2 Exploratory Data Analysis

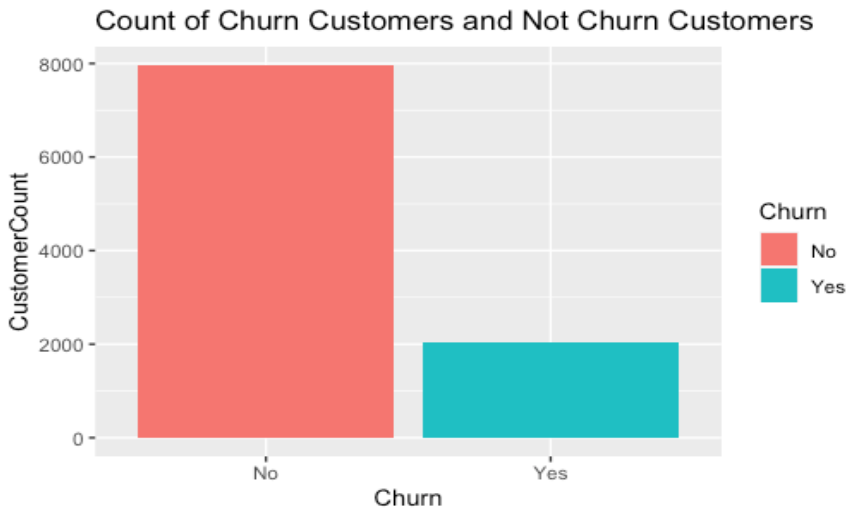
1. Heatmap to analyze the relationships between numeric variables



Observation:

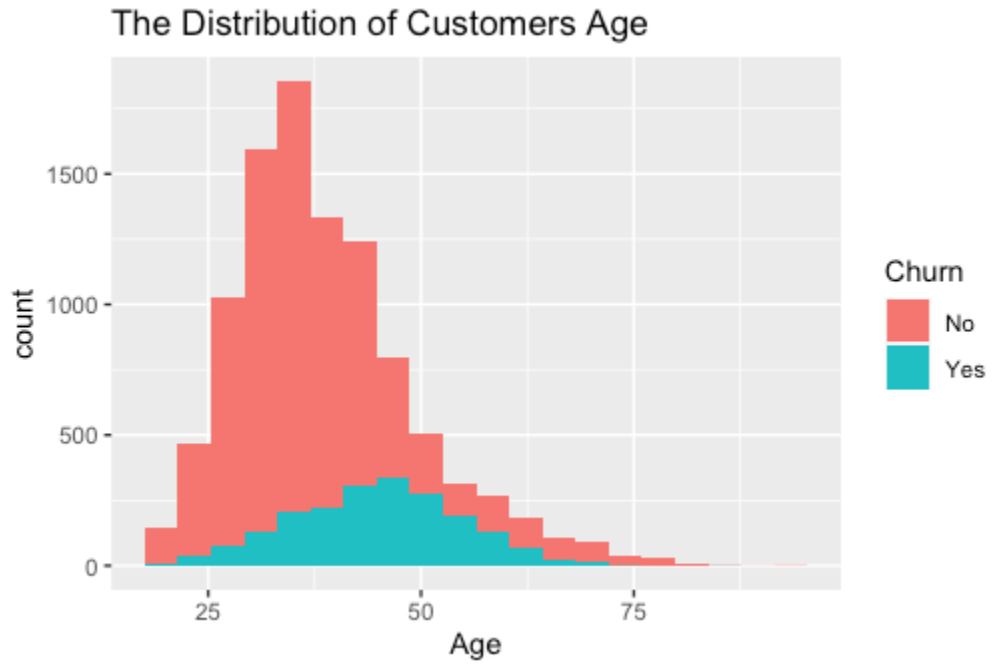
churn is closely related to Age, UseFrequency. Churn is also affected by the variable balance, NumofProfucts. But we cannot see other relationships from the heatmap.

2. Bar plot to show the ratio of Churn Customers



Observation: The ratio of customer churn is 1:4, which means one out of five customers subjected to churn.

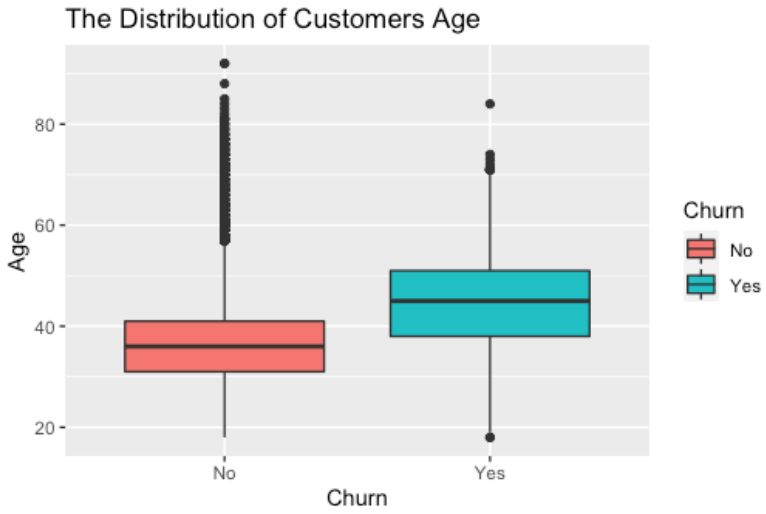
3. Explore the relationship between age and churn



Observation:

The distribution of age is right-skewed, which indicates that we have more younger customers. Most of our customers' age is between 30 and 45 years old. We have fewer customers who are older than 75.

The age distribution of not churn customers is also right skewed, which means that younger customers are less likely subject to churn. The age distribution of churn customers is normally distributed, the mean is about 45-year-old.

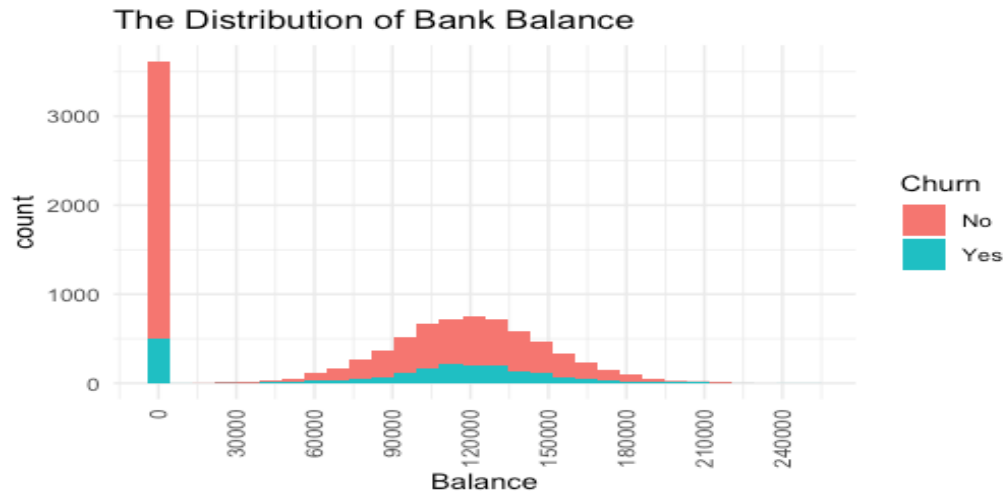


Observation:

The boxplot makes the age distribution of churn and not churn customers clearer.

Not churn customers' average age is about 35-year-old, while the churn customers' average age is about 45-year-old.

4. Explore the relationship between balance and churn



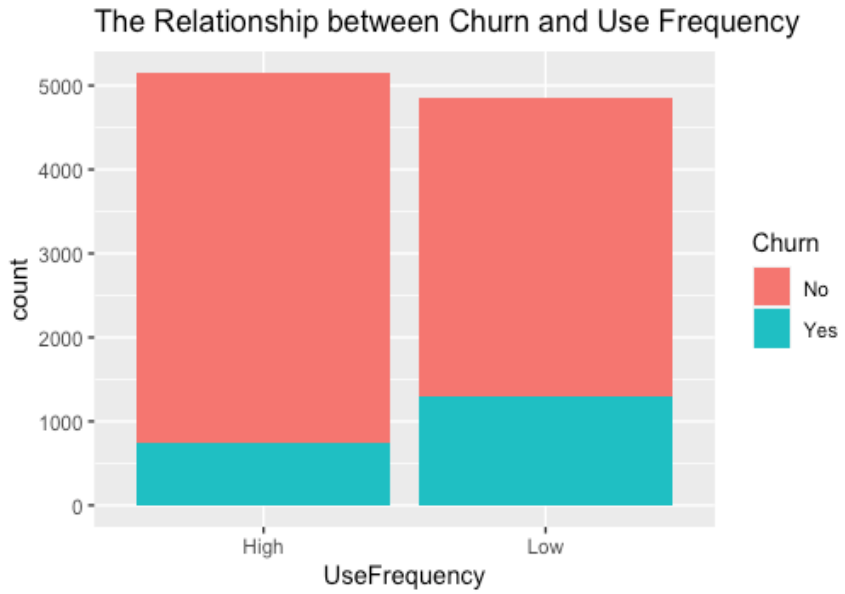
Observation:

The bank balance distribution of churn customers and not churn customers are quite similar.

But notice, for our current customers, there is a large number of them who have no balance in

their bank account which means if a customer does not have money in his bank account, the customer tends to keep his bank account open.

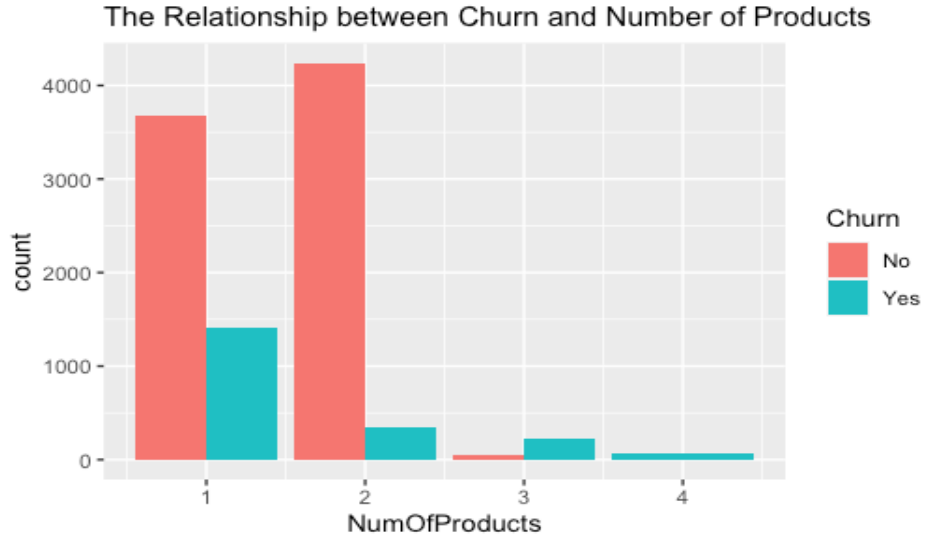
5. Explore the relationship between UseFrequency and Churn



Observation:

Customers who use the bank's products less often are more likely subject to churn. This makes sense because customers who do not use the bank products such as the credit card so often, probably because the card is not convenient to them or they do not like the customer service. In this case, those customers are more likely subject to churn.

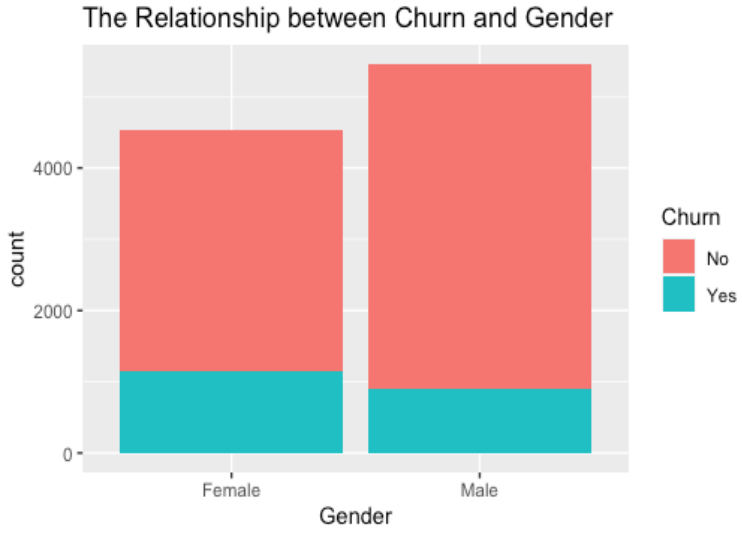
6. Explore the relationship between NumOfProducts and Churn



Observation:

Most of our customers have one or two products from our bank. Only a few amounts of our customers have three or four products from us. However, for those customers who have three or four products from us, they have a high probability of being subject to churn.

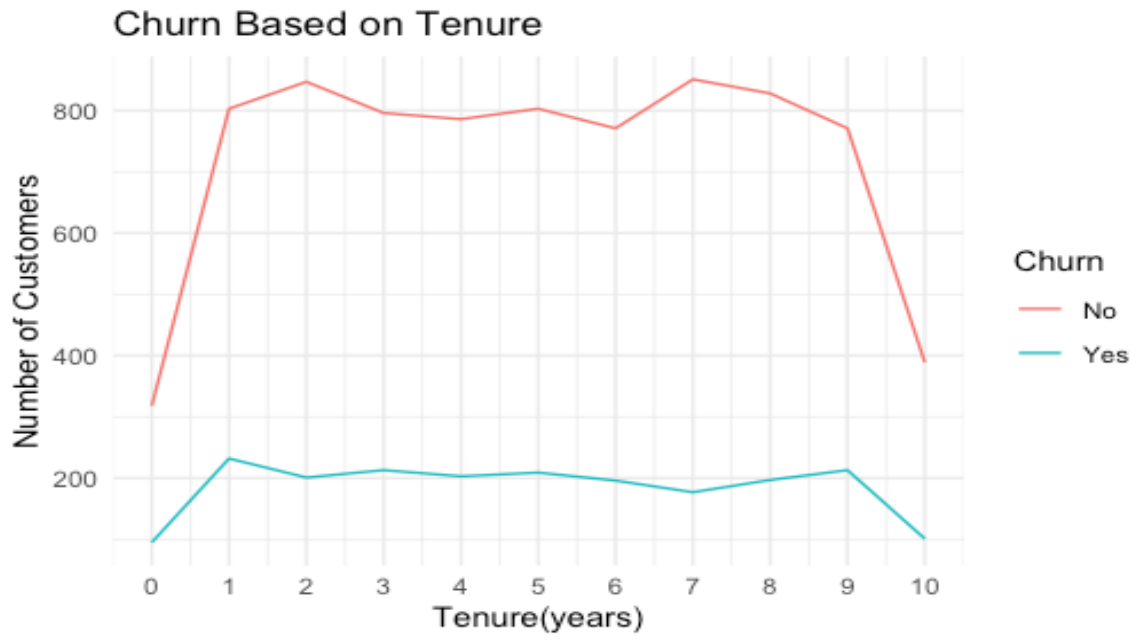
7. Explore the relationship between Gender and Churn



Observation:

The bank has more male customers than female customers. Plus, female customers are more likely subject to churn.

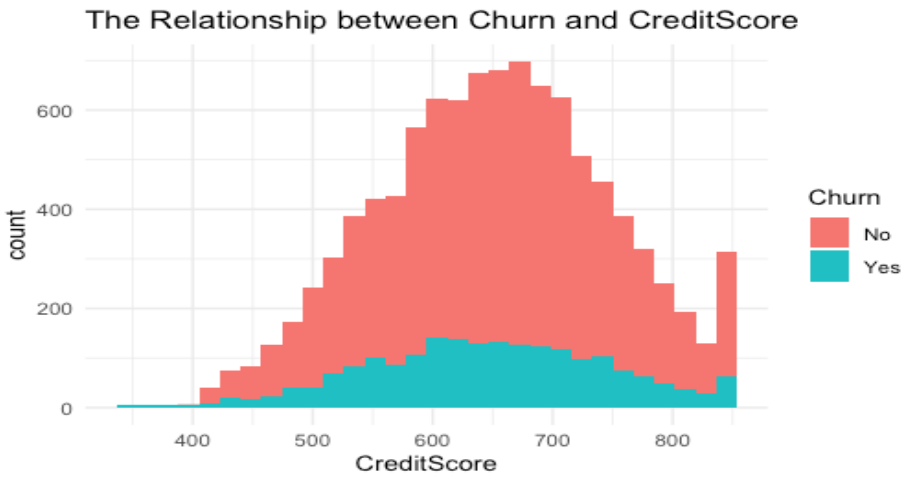
8. Explore the relationship between Tenure and Churn



Observation:

Customers in different tenure groups do not have an apparent tendency to churn or stay. For customers who have left, the tenure among 1 to 9 years is all-around 200 people. For customers who stay, the tenure among 1 to 9 years is all-around 800 people.

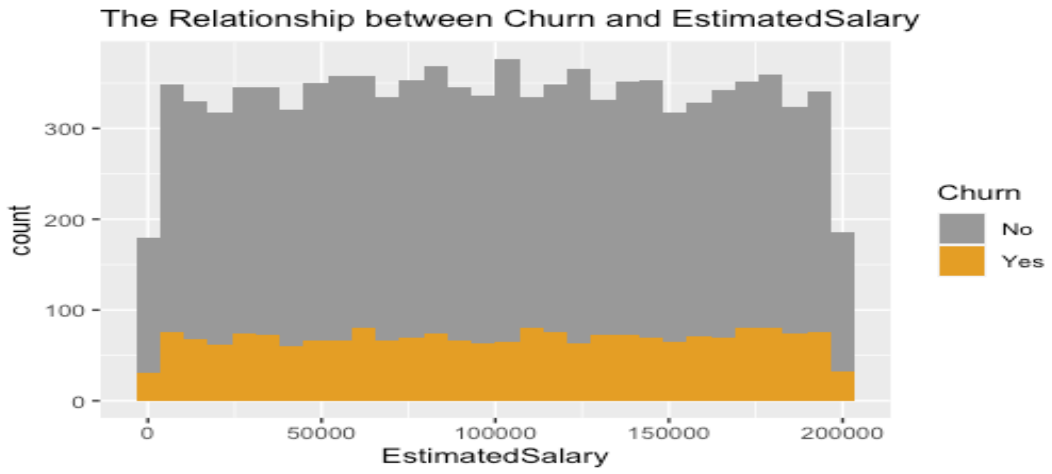
9. Explore the relationship between CreditScore and Churn



Observation:

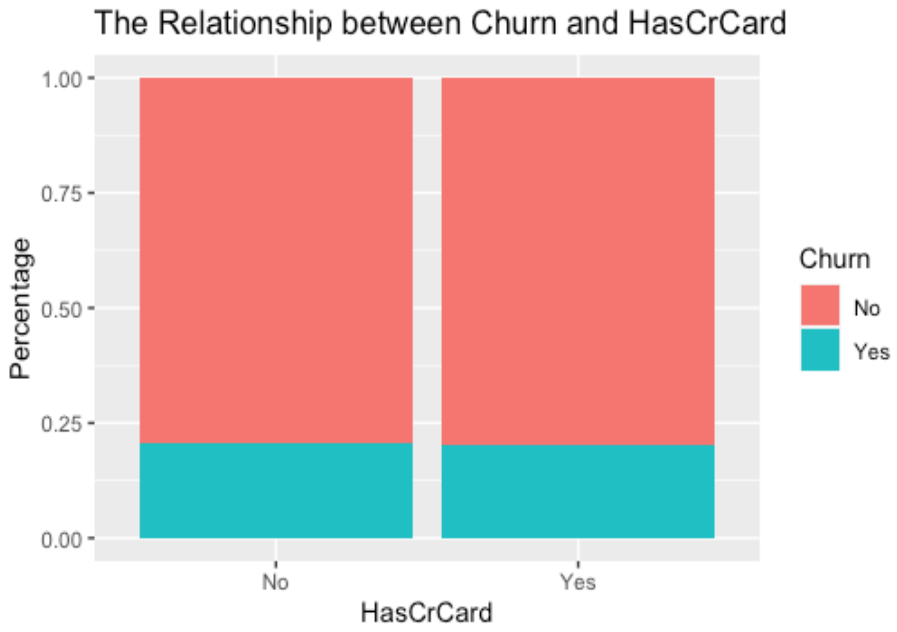
From this graph, the number of customers with credit scores between 600 and 700 are the highest. However, there is a similar distribution of credit score for customers no matter whether they churn or stay, which means there is no clear indication whether customers will churn or not based on the credit scores.

10. Explore the relationship between EstimatedSalary and Churn



Observation:
There is a similar distribution of estimated salary for customers no matter they are churn or stay. The salary does not obviously determine whether a customer will churn or not.

11. Explore the relationship between HasCrCard and Churn



Observation:
We cannot really tell if a customer has a credit card or not since two groups have the same proportion. So, this column will be dropped when building the machine learning model (except for Random Forest Model).

5.3 Statistical Modeling and Data Analysis

Logistic Regression

Logistic Regression is one of the most widely used machine learning algorithms for solving the classification problem. We use this method to predict a dependent variable (Y), given independent variables (X). In this case, the dependent variable 'Churn' is categorical holding values 1 or 0, meaning churn or not.

We split the dataset into train and test data. The training dataset has 7778 observations, and the test dataset has 2222 observations based on an 80-20 split.

```
library(caTools)
set.seed(123)
sample = sample.split(df_new,SplitRatio = 0.80)
train =subset(df_new,sample ==TRUE)
test=subset(df_new, sample==FALSE)
dim(train)
dim(test)
```

We used all features except for 'RowNumber', 'Surname', and 'HasCrCard' to build the logistic regression model.

```
> # Build logistic regression model
> logistic_all<- glm(Churn ~., family = binomial(link = 'logit'), data = train)
>
> # Check the result
> summary(logistic_all)

Call:
glm(formula = Churn ~ ., family = binomial(link = "logit"), data = train)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-2.1304  -0.6715  -0.4624  -0.2759   2.9765

Coefficients:
                  Estimate Std. Error z value Pr(>|z|)
(Intercept)     -4.658e+00  2.833e-01 -16.443   <2e-16 ***
CreditScore     -5.767e-04  3.165e-04  -1.822   0.0684 .
GenderMale      -5.165e-01  6.129e-02  -8.427   <2e-16 ***
Age              7.150e-02  2.907e-03  24.600   <2e-16 ***
Tenure          -1.519e-02  1.054e-02  -1.442   0.1494
Balance          5.467e-06  5.248e-07  10.417   <2e-16 ***
NumOfProducts   -3.179e-02  5.313e-02  -0.598   0.5496
UseFrequencyLow  1.125e+00  6.535e-02  17.208   <2e-16 ***
EstimatedSalary  5.118e-07  5.343e-07   0.958   0.3382
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)
```

From the output above, we got to know a summary of our model. The deviance residuals show the distribution of the deviance residuals for individual cases used in the model. The next part of the output shows each independent variable's coefficient, standard error, the z-statistic, and the associated p-value.

We found that gender, age, balance, and use frequency are statistically significant. For a one-unit increase in age, the log odds of churn increase by 0.0715. For a one-unit increase in the gender of males, the log odds of churn decrease by 0.5165. For a one-unit increase in UseFrequency of low, the log odds of churn increase by 0.5165.

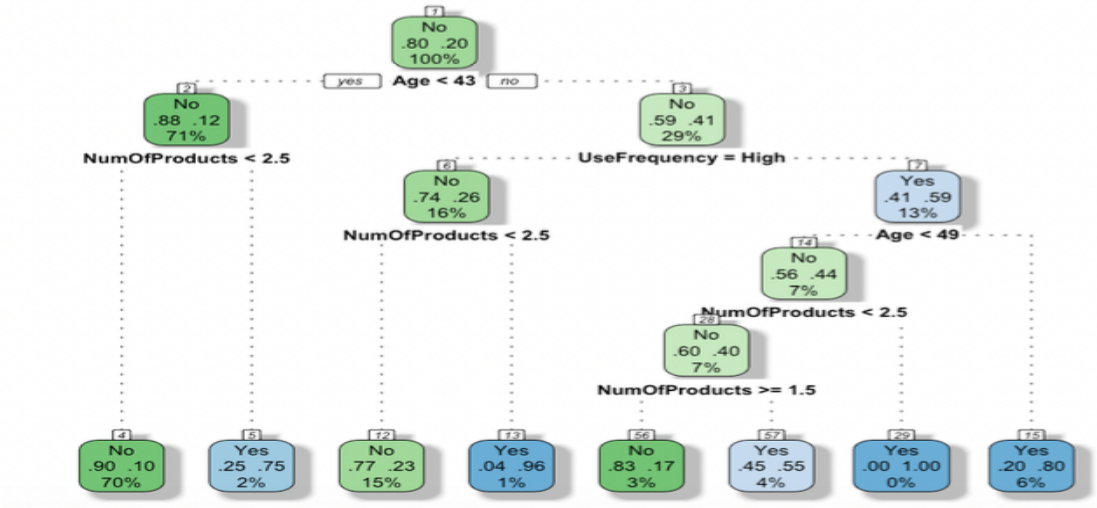
```
confus.matrix <- table(real=test$Churn, predict=pred)
sum(diag(confus.matrix))/sum(confus.matrix)
```

The accuracy of this model is 0.8065.

Decision Tree

A decision tree is a supervised machine learning algorithm that can be also used for classification problems. The composition of a decision tree is the root node, internal nodes, leaf node, and branch. We still used the previous 80-20 split dataset, and also dropped 'RowNumber', 'Surname', and 'HasCrCard' columns. We use repart() in R to run a decision tree model.

```
library(rpart)
library(rattle)
churn_tree <- rpart(Churn ~., data = train, method = 'class')
fancyRpartPlot(churn_tree)
```



The tree shows that age is the root node which is the most important variable to predict customer churn or not churn. If the customer's age is smaller than 43 (71% possibility), then it goes to the left side of the tree and checks the NumOfProducts variable. If the number of products is smaller than 2.5 then the customer does not churn, otherwise, the customer churns.

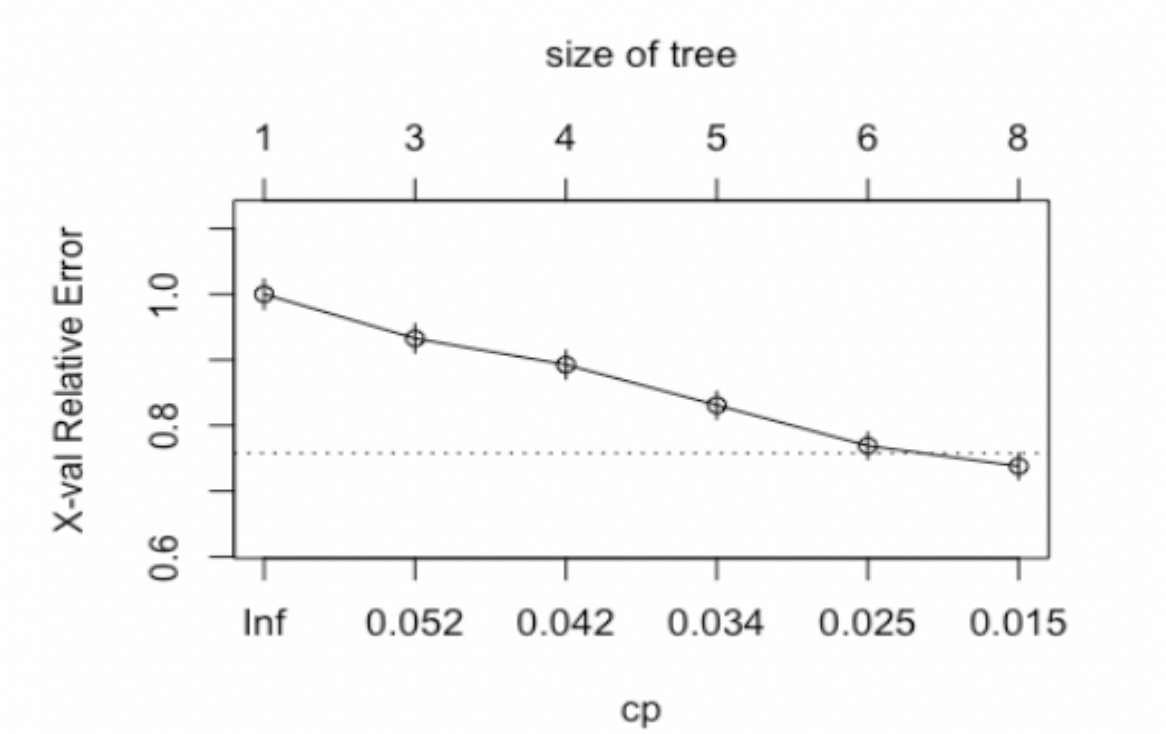
Call summary(churn_tree). The order of importance of the variable is age, NumOfProducts, UseFrequency, Balance, and CreditScore.

```
Variable importance
          Age NumOfProducts  UseFrequency       Balance   CreditScore
           49            31            17             2             1

  Primary splits:
      Age           < 42.5     to the left,  improve=271.53450, (0 missing)
      NumOfProducts < 2.5      to the left,  improve=225.14140, (0 missing)
      UseFrequency  splits as  LR,           improve= 67.08858, (0 missing)
      Balance       < 86022.26 to the left,  improve= 42.34722, (0 missing)
      Gender        splits as  RL,           improve= 27.28764, (0 missing)
  Surrogate splits:
      NumOfProducts < 3.5      to the left,  agree=0.714, adj=0.005, (0 split)
      CreditScore   < 361      to the right, agree=0.713, adj=0.001, (0 split)
```

```
confus.matrix <- table(real=test$Churn, predict=pred)
sum(diag(confus.matrix))/sum(confus.matrix)
```

The results show that the accuracy of this model is 0.8483, but is it possible to improve the prediction accuracy of the model? We pruned the tree.

To validate the model, we used the plot cp functions. 'CP' stands for Complexity Parameter of the tree. The cp values are plotted against the geometric mean to depict the deviation until the minimum value is reached.



We used cp with a minimum error (0.015) to prune the tree and then rebuilt the model.

```
prunetree_cart.model <- prune(churn_tree, cp = churn_tree$cptable[which.min(churn_tree$cptable[,"xerror"]),"CP"])

prunetree_pred <- predict(prunetree_cart.model, newdata=test, type="class")
```

```
confus.matrix.prune <- table(real=test$Churn, predict=prunetree_pred)
sum(diag(confus.matrix.prune))/sum(confus.matrix.prune)
```

The accuracy is still 0.8483. It shows that there is no need to prune the tree.

We also need to consider if this model has an overfitting issue. We used K-fold Cross-Validation to avoid it.

```
library(lattice)
library(caret)
library(e1071)
train_control <- trainControl(method="cv", number=10)
train_control.model <- train(Churn~., data=train, method="rpart", trControl=train_control)
train_control.model
```

```
7778 samples
   8 predictor
   2 classes: 'No', 'Yes'

No pre-processing
Resampling: Cross-Validated (10 fold)
Summary of sample sizes: 7000, 7000, 7000, 7000, 7000, 7001, ...
Resampling results across tuning parameters:

  cp          Accuracy   Kappa
  0.03944020  0.8284877  0.3058284
  0.04516539  0.8234735  0.3197233
  0.05947837  0.8035475  0.1315020

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was cp = 0.0394402.
```

The accuracy is 0.8285. We solved the overfitting issue.

Random Forest

Random forest is an integrated algorithm composed of decision trees, which can perform well in many cases. It builds multiple decision trees and glues them together to get a more accurate and stable prediction.

In this case, we used all features except for the 'RowNumber' and 'Surname' columns.

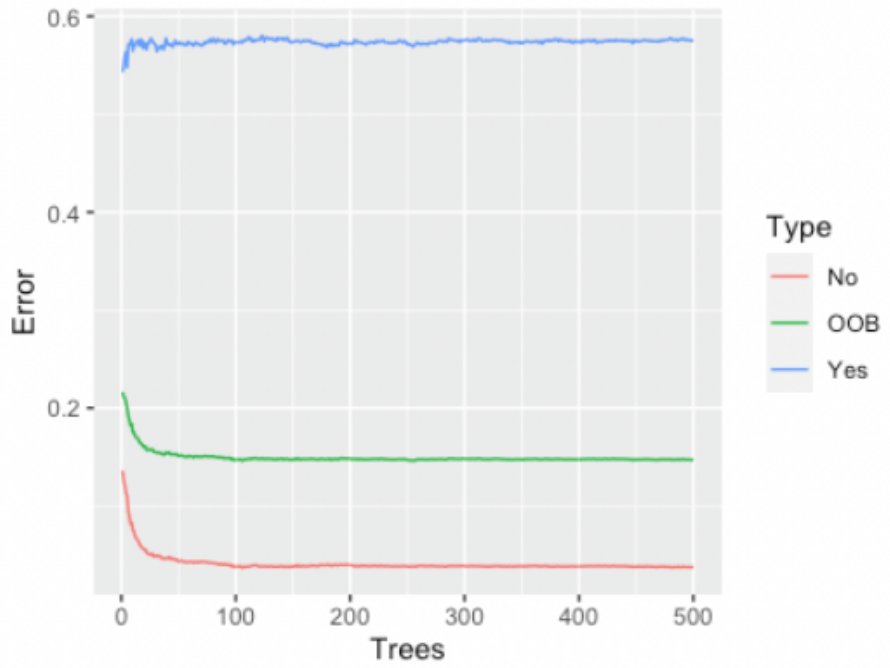
Firstly, we used the default setting ntree=500 to build a random forest model. There are 3 variables that are considered at each internal node. The OOB error estimate is 14.72% It means that 85.28% of the OOB samples were correctly classified by the random forest.

```
> model <- randomForest(Churn ~ ., data=df, proximity=TRUE)
> model

Call:
 randomForest(formula = Churn ~ ., data = df, proximity = TRUE)
               Type of random forest: classification
                     Number of trees: 500
No. of variables tried at each split: 3

        OOB estimate of  error rate: 14.72%
Confusion matrix:
      No Yes class.error
No  7664 299  0.03754866
Yes 1173 864  0.57584683
>
```

The error rate is relatively low when predicting “No”, while the error rate is much higher when predicting “Yes”.

Secondly, we plot the error rate for different numbers of trees. We found that the error rate does not change so much after ntree=100.

```
oob.error.data <- data.frame(
  Trees=rep(1:nrow(model$err.rate), times=3),
  Type=rep(c("OOB", "Yes", "No"), each=nrow(model$err.rate)),
  Error=c(model$err.rate[,"OOB"],
          model$err.rate[,"Yes"],
          model$err.rate[,"No"]))
ggplot(data=oob.error.data, aes(x=Trees, y=Error)) +
  geom_line(aes(color=Type))
```



We rebuilt the model and got an error rate of 14.91%.

```
Call:
 randomForest(formula = Churn ~ ., data = df, proximity = TRUE,      ntree = 100)
               Type of random forest: classification
                     Number of trees: 100
No. of variables tried at each split: 3

        OOB estimate of  error rate: 14.91%
Confusion matrix:
      No Yes class.error
No  7643 320  0.04018586
Yes 1171 866  0.57486500
```

Thirdly, we found the best mtry. The mtry refers to how many variables we should select at a node split. We created a for loop and built a random forest using 'i' to determine the number of variables. Then we printed all the OOB error rates and found that when mtry = 2, we got the lowest OOB error rate.

```
> oob.values
[1] 0.1806 0.1452 0.1482 0.1516 0.1500 0.1523 0.1527 0.1548 0.1558
> #find the minimum error
> min(oob.values)
[1] 0.1452
> #find the optimal value for mtry
> which(oob.values == min(oob.values))
[1] 2
```

Lastly, we rebuilt the model with ntree=100, mtry=2.

```
> model <- randomForest(Churn ~ ., data=df, proximity=TRUE,ntree=100,mtry=2)
> model

Call:
 randomForest(formula = Churn ~ ., data = df, proximity = TRUE,      ntree = 100, mtry = 2)
               Type of random forest: classification
                     Number of trees: 100
No. of variables tried at each split: 2

        OOB estimate of  error rate: 14.65%
Confusion matrix:
      No Yes class.error
No  7766 197  0.02473942
Yes 1268 769  0.62248405
```

The error rate decreased to14.65% from 14.91%, so the accuracy of this model is 0.8535.

KNN

KNN is one kind of supervised machine learning algorithm that can be used for both classification and regression. The KNN algorithm assumes that similar things are near to each other. As a candidate model to classify churn and not churn customers, the KNN model has advantages such as Works with any number of classes, only has two parameters: K and distance Metric. However, there are also disadvantages of this model, for example, the KNN model is not suitable for categorical features. So, before implementing the KNN model, the categorical variables should be converted to numeric data, for those that cannot convert to numeric data type, we need to drop them before applying the KNN model.

The Churn dataset was reloaded to Rstudio, and the RowNumber, Surname, and HasCrCard columns were dropped since they are not relevant to the analysis.

Then the Gender column was converted to an integer, the code as shown below.

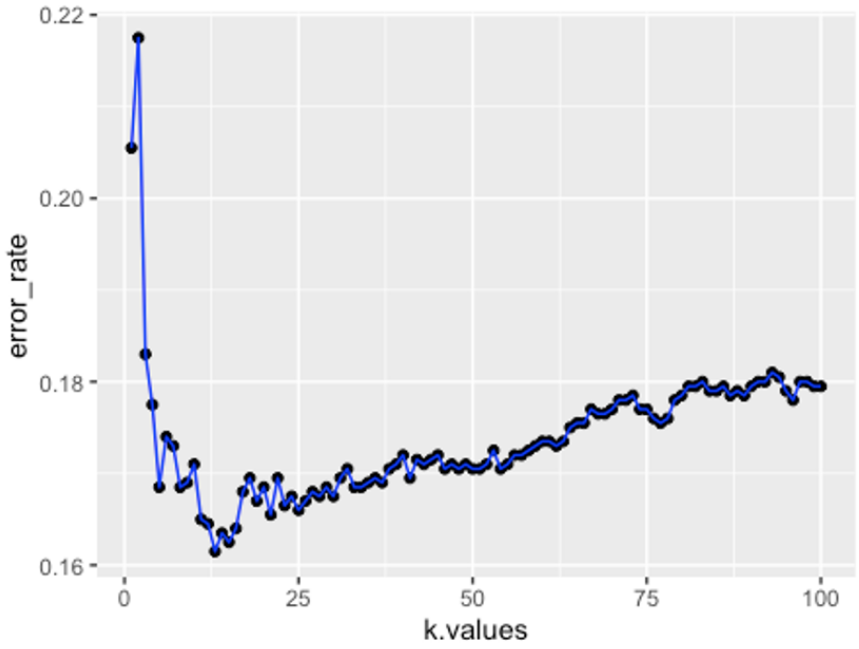
```
> #Convert Gender to int
> df_knn$Gender<- ifelse(df_knn$Gender=="Female", 1, 0)
```

Check the structure of the data. Now, only numeric variables are in the data.

```
> str(df_knn)
'data.frame':   10000 obs. of  9 variables:
 $ CreditScore    : int  619 608 502 699 850 645 822 376 501 684 ...
 $ Gender         : num  1 1 1 1 1 0 0 1 0 0 ...
 $ Age            : int  42 41 42 39 43 44 50 29 44 27 ...
 $ Tenure         : int  2 1 8 1 2 8 7 4 4 2 ...
 $ Balance        : num  0 83808 159661 0 125511 ...
 $ NumOfProducts  : int  1 1 3 2 1 2 2 4 2 1 ...
 $ UseFrequency   : int  1 1 0 0 1 0 1 0 1 1 ...
 $ EstimatedSalary: num  101349 112543 113932 93827 79084 ...
 $ Churn          : int  1 0 1 0 0 1 0 1 0 0 ...
```

Before implementing the KNN model, there is one more important thing to do that is normalize the data. The scale() function was used to normalize the data.

When building the KNN model, the class library was called. As the KNN model stands for K Nearest Neighbor, finding the optimal value of K is an important step. In our analysis, the elbow method was used to find the optimal K value.



The graph showed the error rate of different k values. The lowest error rate appeared at k equal to 13. So, k equal to 13 will be implemented to the KNN model and the accuracy was 0.8385.

```
> error_rate.13<-mean(test.Churn != predicted.Churn.13)
> print(error_rate.13)
[1] 0.1615
```

Naive Bayes

The Naive Bayes algorithm was built based on the Bayes' Theorem. A Naive Bayes classifier is a probabilistic machine learning model that can be used to classify customer churn. The advantages of working with the Naïve Bayes algorithm are 1. Requires a small amount of training data to learn the parameters. 2. Can be trained relatively fast compared to sophisticated models. The disadvantages of the Naïve Bayes algorithm are 1. It is a decent classifier but a bad estimator. 2. It works well with discrete values but will not work with continuous values.
The library 'e1071' should be called when implementing the Naive Bayes model. We used the same train and test data set as when we built the logistic regression. As shown in the results, the error rate was 0.1715. Correspondingly, the accuracy of the Naive Bayes model is 0.8285.

```
pre_Churn   No  Yes
      No  1730  354
      Yes   27  111
> # misclassification rate
> 1-sum(diag(con_mat))/sum(con_mat)
[1] 0.1714671
```

5.4 Predict Customer Churn

According to the machine learning models we have studied above; the Random Forest gave us the lowest error rate when predicting churn. So, the optimal Random Forest model with ntree = 100, mtry=2 will be used to predict customer churn. The information about the new customers is listed below.

	CreditScore	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	UseFrequency	EstimatedSalary
1	650	Female	40	3	4000.84	1	Yes	Low	110000
2	720	Female	45	8	7934.00	1	Yes	Low	120000
3	400	Female	28	5	0.00	3	Yes	High	90000
4	500	Male	65	4	2343.98	2	No	High	95000
5	680	Male	50	2	9000.87	3	No	High	80000

After we run the Random Forest model, the predicted results are shown as below.

	CreditScore	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	UseFrequency	EstimatedSalary	predict.CustomerChurn
1	650	Female	40	3	4000.84	1	Yes	Low	110000	No
2	720	Female	45	8	7934.00	1	Yes	Low	120000	Yes
3	400	Female	28	5	0.00	3	Yes	High	90000	Yes
4	500	Male	65	4	2343.98	2	No	High	95000	No
5	680	Male	50	2	9000.87	3	No	High	80000	Yes

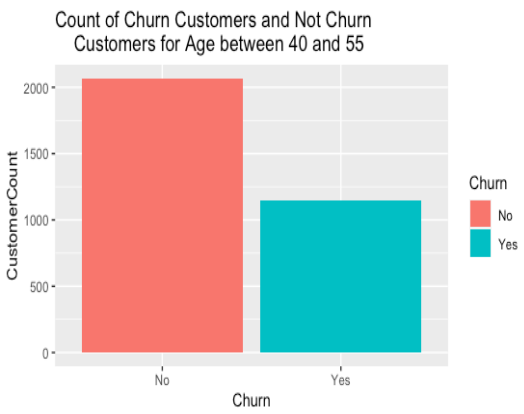
Throughout the above analysis, we have learned several important things:

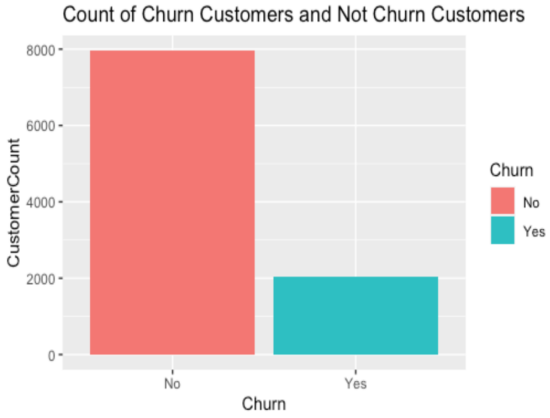
- Age, UseFrequency, balance, and NumofProfucts have higher correlations with Churn columns.
- Most of our customers are between 30 and 45 years old. Customers whose age between 40 and 50 are more likely to leave the bank than younger ones.
- Customers who use products of the bank more frequently are less likely to leave the bank.
- The bank balance distribution of churn customers and not churn customers are quite similar. However, there are many current customers who have no balance in their bank account.

- Most customers have one or two products from this bank. Compare two groups of customers, people who use more than two products are more likely to churn.
- Predictions are made with a total of 5 classification models. Random Forest is the best model to predict whether the customer will stay or not.

5.5 Strategies to reduce Customer Churn

Successfully predicting customer churn is not the end of our analysis. We want to explore why customers are subject to churn and come up with solutions to reduce customer churn. One of the most important findings in our analysis is that customers whose age between 40 to 55 are more likely to leave the bank. They might need to switch to other banking services for retirement purposes or investment purposes.





As shown in the two graphs, the churn rate is obviously higher for customers age between 40 and 55.

In order to reduce customer churn especially keep customers whose age between 40 and 55, some suggestions are given below:

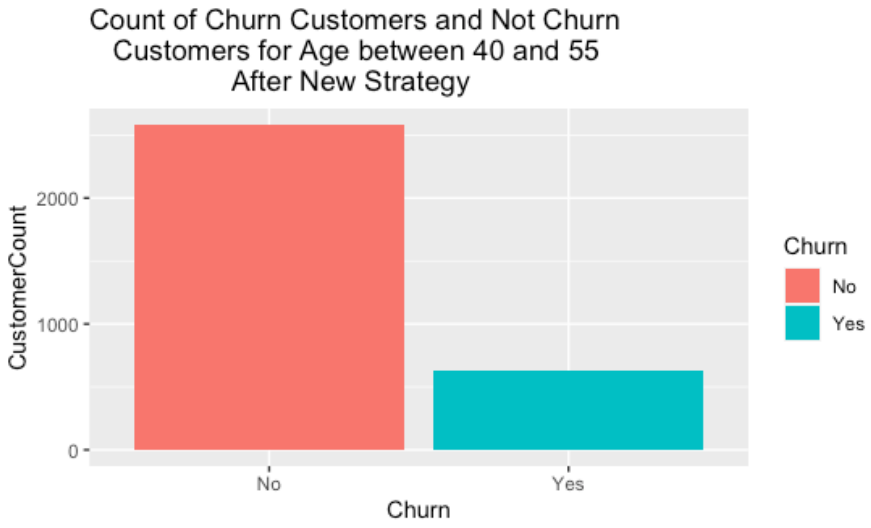
1. Marketing strategies targeted at middle age and elderly customers

- Middle age customers tend to have more savings, so the bank can offer them suitable investment products, such as early-intervention retirement planning.
- For lower-income older adults, our bank can offer low-cost, low-fee checking accounts, low-interest lending, and credit products.
- Develop or keep a user-friendly platform for older customers. Young generations prefer using mobile/online platforms to do transactions, but elderly people may not.

2. Improve services

- Actively engage customers with products from emails, phone calls, and mobile news. Clarify the benefits of products and offer news updates, such as announcements of deals and special offers.
- Examining current products. The data shows that people who use more than two products are very likely to churn. Our Bank can do some surveys to find out the reasons.
- Pay attention to customer complaints.
- Offer door-to-door service for people with mobility issues.

We estimate that our strategies can effectively prevent about 50% of 40 to 55 years old customers from leaving the bank. The bar plot showed the churn rate of 40 to 55 years old customers after we implemented the new strategies.



6. References

1. Credit Card Marketing Classification Trees, Grayson, Gardner,2015

https://www.jmp.com/content/dam/jmp/documents/en/academic/case-study-library/case-study-library-12/analytics-cases/ct-creditcardmarketing.pdf

2. Churn for Bank Customers. Retrieved from Kaggle

https://www.kaggle.com/mathchi/churn-for-bank-customers

3. Maya Abood (January 2015). Adopting Age-Friendly Banking to Improve Financial Well-Being for Older Adults
https://www.frbsf.org/community-development/files/Age_Friendly_Banking_Jan2015.pdf

4. Credit card dataset Retrieved from DataWorld
https://data.world/gautam2510/credit-card-dataset/workspace/data-dictionary

5. Analytics Vidhya (March 28, 2016): Practical Guide to deal with Imbalanced Classification Problems in R Retrieved from Analytics Vidhya
https://www.analyticsvidhya.com/blog/2016/03/practical-guide-deal-imbalanced-classification-problems/

6. Rathnadevi Manivannan (October 09, 2017): Handling Imbalanced Data with R Retrieved from DZone
https://dzone.com/articles/handle-class-imbalance-data-with-r

7. No name (No date): R - Decision Tree retrieved from tutorialspoint
https://www.tutorialspoint.com/r/r_decision_tree.htm