TELECOM CHURN PREDICTION REPORT

Andrii Platonov

17 06 2020

INTRODUCTION

Project goal

The goal of the project is to predict churn of a telecom company and compare some advanced machine learning algorithms by using one of telecom dataset. The telecom dataset was downloaded from www.kaggle.com. It has over 7,000 records and 21 variables.

Models

In the project, the following models will be explored: Decision tree, Random forest, and Support Vector Machine.

Steps

The key steps of the project will be performed:

1. Data Cleaning (downloading and preparation data for analysis)
2. Data Exploration and Visualization (analysis of data and variables).
3. Data Wrangling (identifying/adding necessary variables for data modeling).
4. Data Modeling (covers modeling approach).
5. Results (summarizes the results of data modeling and identifies the best machine learning model for our dataset).
6. Conclusion (provides a brief summary of the report, its potential impact, its limitations, and future work).

Installing Packages

The following packages will be loaded and installed for analysis and modeling within the project.

```
#installing required packages
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
if(!require(dplyr)) install.packages("dplyr", repos = "http://cran.us.r-project.org")
if(!require(plyr)) install.packages("plyr", repos = "http://cran.us.r-project.org")
if(!require(DataExplorer)) install.packages("DataExplorer", repos = "http://cran.us.r-project.org")
if(!require(ggplot2)) install.packages("ggplot2", repos = "http://cran.us.r-project.org")
if(!require(cowplot)) install.packages("cowplot", repos = "http://cran.us.r-project.org")
if(!require(ggpubr)) install.packages("ggpubr", repos = "http://cran.us.r-project.org")
if(!require(scales)) install.packages("scales", repos = "http://cran.us.r-project.org")
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
if(!require(rpart)) install.packages("rpart", repos = "http://cran.us.r-project.org")
```

```r
if(!require(rpart.plot)) install.packages("rpart.plot", repos = "http://cran.us.r-project.org")
if(!require(ROCR)) install.packages("ROCR", repos = "http://cran.us.r-project.org")
if(!require(randomForest)) install.packages("randomForest", repos = "http://cran.us.r-project.org")
if(!require(pROC)) install.packages("pROC", repos = "http://cran.us.r-project.org")
if(!require(e1071)) install.packages("e1071", repos = "http://cran.us.r-project.org")

# opening the libraries
library(tidyverse)
library(dplyr)
library(plyr)
library(DataExplorer)
library(ggplot2)
library(cowplot)
library(ggpubr)
library(scales)
library(caret)
library(rpart)
library(rpart.plot)
library(ROCR)
library(randomForest)
library(pROC)
library(e1071)
```

Downloading the file

The dataset will be downloaded with the Github link.

```r
# Use the Github link to download data set
churn_set <- read.csv("https://raw.githubusercontent.com/aplatonow/Churn-project/master/Telco-Customer-C
```

Dataset and variables

The names of the columns can be represented by `colnames function`.

```r
# explore column names
colnames(churn_set)
```

```
##  [1] "customerID"       "gender"           "SeniorCitizen"    "Partner"
##  [5] "Dependents"       "tenure"           "PhoneService"     "MultipleLines"
##  [9] "InternetService"  "OnlineSecurity"   "OnlineBackup"     "DeviceProtection"
## [13] "TechSupport"      "StreamingTV"      "StreamingMovies"  "Contract"
## [17] "PaperlessBilling" "PaymentMethod"    "MonthlyCharges"   "TotalCharges"
## [21] "Churn"
```

All variables in the dataset can be combined into several data groups:

1. Churn (identifies customers who left a company within the last month);
2. Type of services for customers (phone, internet, different online services and etc.);
3. Customer account information (how long they stay with a company, type of contract, payment methods, monthly charges, and etc.);
4. Demographic information of customers (gender, age, marriage status, and etc.).

ANALYSIS

In this section, the following steps will be covered: 1) Data Cleaning; 2) Data Exploration and Visualization; 3) Data Wrangling and Structuring; 4) Data Modeling.

Data Cleaning

The `glimpse function` can be used to find the number of variables in the dataset and identify their type.

```r
# show variables and their type
glimpse(churn_set)
```

```
## Rows: 7,043
## Columns: 21
## $ customerID       <chr> "7590-VHVEG", "5575-GNVDE", "3668-QPYBK", "7795-CF...
## $ gender           <chr> "Female", "Male", "Male", "Male", "Female", "Femal...
## $ SeniorCitizen    <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,...
## $ Partner          <chr> "Yes", "No", "No", "No", "No", "No", "No", "No", "...
## $ Dependents       <chr> "No", "No", "No", "No", "No", "No", "Yes", "No", "...
## $ tenure           <int> 1, 34, 2, 45, 2, 8, 22, 10, 28, 62, 13, 16, 58, 49...
## $ PhoneService     <chr> "No", "Yes", "Yes", "No", "Yes", "Yes", "Yes", "No...
## $ MultipleLines    <chr> "No phone service", "No", "No", "No phone service"...
## $ InternetService  <chr> "DSL", "DSL", "DSL", "DSL", "Fiber optic", "Fiber ...
## $ OnlineSecurity   <chr> "No", "Yes", "Yes", "Yes", "No", "No", "No", "Yes"...
## $ OnlineBackup     <chr> "Yes", "No", "Yes", "No", "No", "No", "Yes", "No",...
## $ DeviceProtection <chr> "No", "Yes", "No", "Yes", "No", "Yes", "No", "No",...
## $ TechSupport      <chr> "No", "No", "No", "Yes", "No", "No", "No", "No", "...
## $ StreamingTV      <chr> "No", "No", "No", "No", "No", "Yes", "Yes", "No", ...
## $ StreamingMovies  <chr> "No", "No", "No", "No", "No", "Yes", "No", "No", "...
## $ Contract         <chr> "Month-to-month", "One year", "Month-to-month", "O...
## $ PaperlessBilling <chr> "Yes", "No", "Yes", "No", "Yes", "Yes", "Yes", "No...
## $ PaymentMethod    <chr> "Electronic check", "Mailed check", "Mailed check"...
## $ MonthlyCharges   <dbl> 29.85, 56.95, 53.85, 42.30, 70.70, 99.65, 89.10, 2...
## $ TotalCharges     <dbl> 29.85, 1889.50, 108.15, 1840.75, 151.65, 820.50, 1...
## $ Churn            <chr> "No", "No", "Yes", "No", "Yes", "Yes", "No", "No",...
```

The `summary function` can be used to verify the data and understand the attributes.

```r
# summary shows the data summary
summary(churn_set)
```

```
##   customerID           gender          SeniorCitizen      Partner
##  Length:7043        Length:7043        Min.   :0.0000   Length:7043
##  Class :character   Class :character   1st Qu.:0.0000   Class :character
##  Mode  :character   Mode  :character   Median :0.0000   Mode  :character
##                                        Mean   :0.1621
##                                        3rd Qu.:0.0000
##                                        Max.   :1.0000
##
##   Dependents            tenure      PhoneService       MultipleLines
##  Length:7043        Min.   : 0.00   Length:7043        Length:7043
##  Class :character   1st Qu.: 9.00   Class :character   Class :character
##  Mode  :character   Median :29.00   Mode  :character   Mode  :character
##                     Mean   :32.37
##                     3rd Qu.:55.00
##                     Max.   :72.00
```

```
##
##   InternetService    OnlineSecurity     OnlineBackup       DeviceProtection
##   Length:7043        Length:7043        Length:7043        Length:7043
##   Class :character   Class :character   Class :character   Class :character
##   Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##   TechSupport        StreamingTV        StreamingMovies      Contract
##   Length:7043        Length:7043        Length:7043        Length:7043
##   Class :character   Class :character   Class :character   Class :character
##   Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##
##   PaperlessBilling   PaymentMethod      MonthlyCharges    TotalCharges
##   Length:7043        Length:7043        Min.   : 18.25   Min.   :  18.8
##   Class :character   Class :character   1st Qu.: 35.50   1st Qu.: 401.4
##   Mode  :character   Mode  :character   Median : 70.35   Median :1397.5
##                                         Mean   : 64.76   Mean   :2283.3
##                                         3rd Qu.: 89.85   3rd Qu.:3794.7
##                                         Max.   :118.75   Max.   :8684.8
##                                                          NA's   :11
##      Churn
##   Length:7043
##   Class :character
##   Mode  :character
##
##
##
##
```

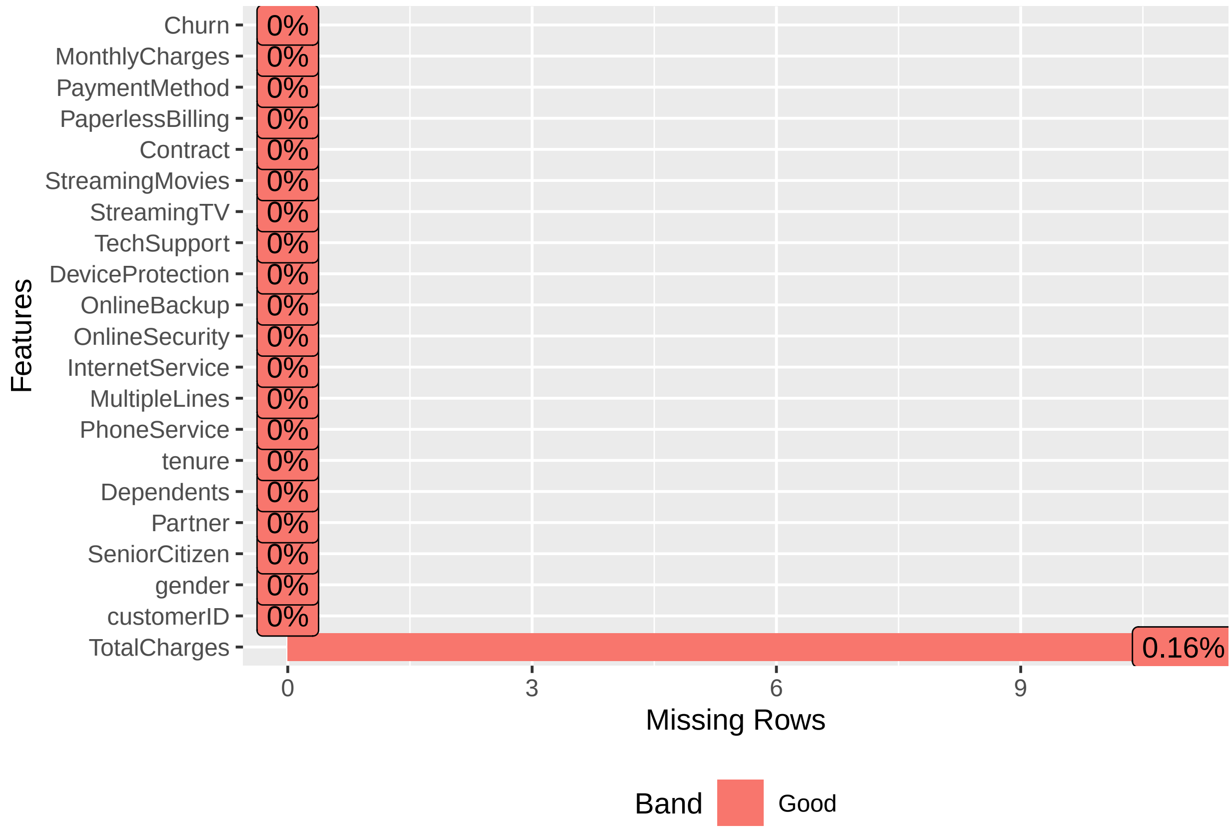
Observation for missing values in data set.

```r
# checking for NA values
apply(is.na(churn_set), 2, sum)
```

```
##       customerID           gender    SeniorCitizen          Partner
##                0                0                0                0
##       Dependents           tenure     PhoneService    MultipleLines
##                0                0                0                0
##  InternetService   OnlineSecurity     OnlineBackup DeviceProtection
##                0                0                0                0
##      TechSupport      StreamingTV  StreamingMovies         Contract
##                0                0                0                0
## PaperlessBilling    PaymentMethod   MonthlyCharges     TotalCharges
##                0                0                0               11
##            Churn
##                0
```

11 records of NA were identified in "Total Charges". The `plot_missing function` will be used to plot NA in order to recognize how many percentages of data are missing.

```r
# Identifying how many percentage of NA in data set
plot_missing(churn_set)
```



Actually, there is small percentage (0.16%) of missing data in Total Charges. However, let's identify what kind of customers they are. To understand how long these customers are staying with a company, let's check their tenure.

```r
# Identifying customer's tenure with NA in Total Charges.
churn_set %>% filter(is.na(TotalCharges)) %>% summarize(customerID, TotalCharges, tenure)
```

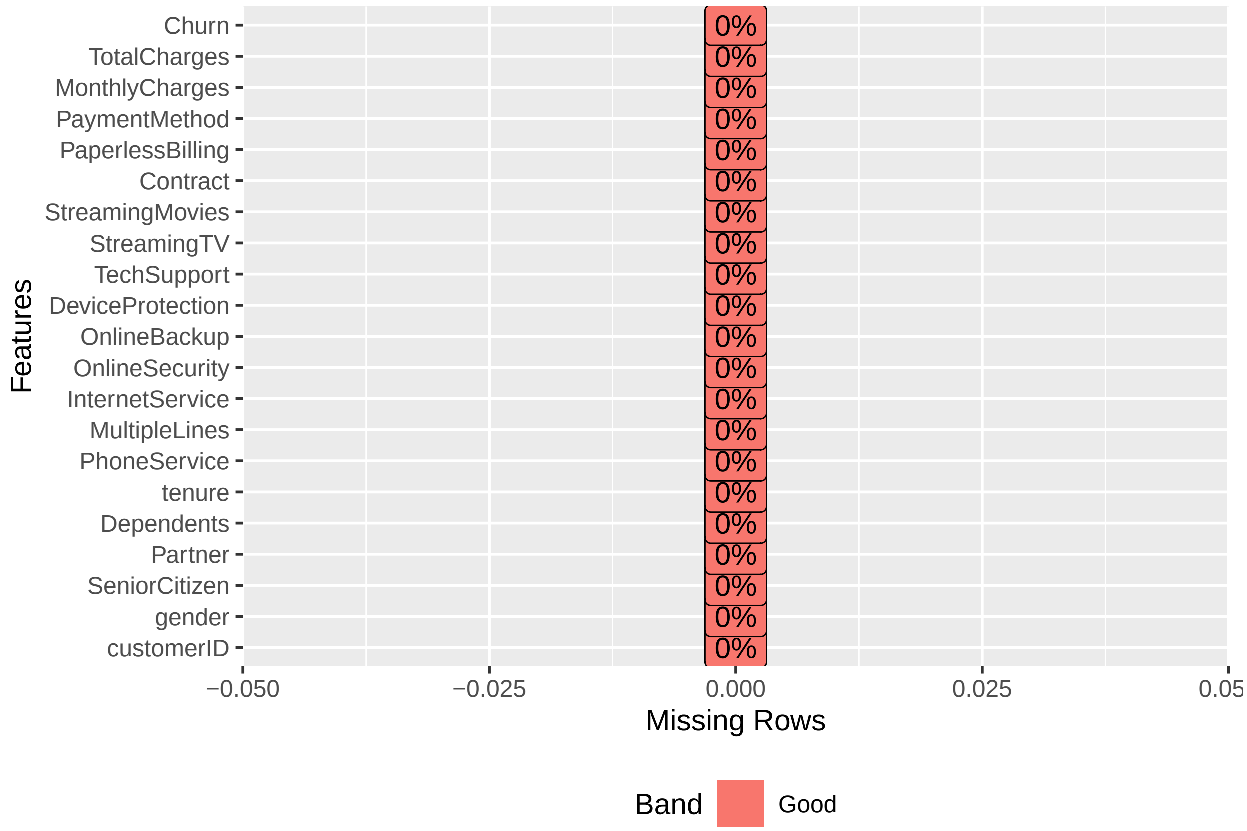
```
##    customerID TotalCharges tenure
## 1  4472-LVYGI           NA      0
## 2  3115-CZMZD           NA      0
## 3  5709-LVOEQ           NA      0
## 4  4367-NUYAO           NA      0
## 5  1371-DWPAZ           NA      0
## 6  7644-OMVMY           NA      0
## 7  3213-VVOLG           NA      0
## 8  2520-SGTTA           NA      0
## 9  2923-ARZLG           NA      0
## 10 4075-WKNIU           NA      0
## 11 2775-SEFEE           NA      0
```

All records with NA support the idea that these are new customers with zero tenure. It can be assumed, that they have just signed up and have no bill to pay yet. In this case, we can change all NA to zero.

```r
# Changing NA values to zero
churn_set[is.na(churn_set)] <- 0
```

To make sure there is no more NA in dataset, let's double-check and plot NA values again.

```
# double checking for NA values again
plot_missing(churn_set)
```



These are no more missing values in data set.

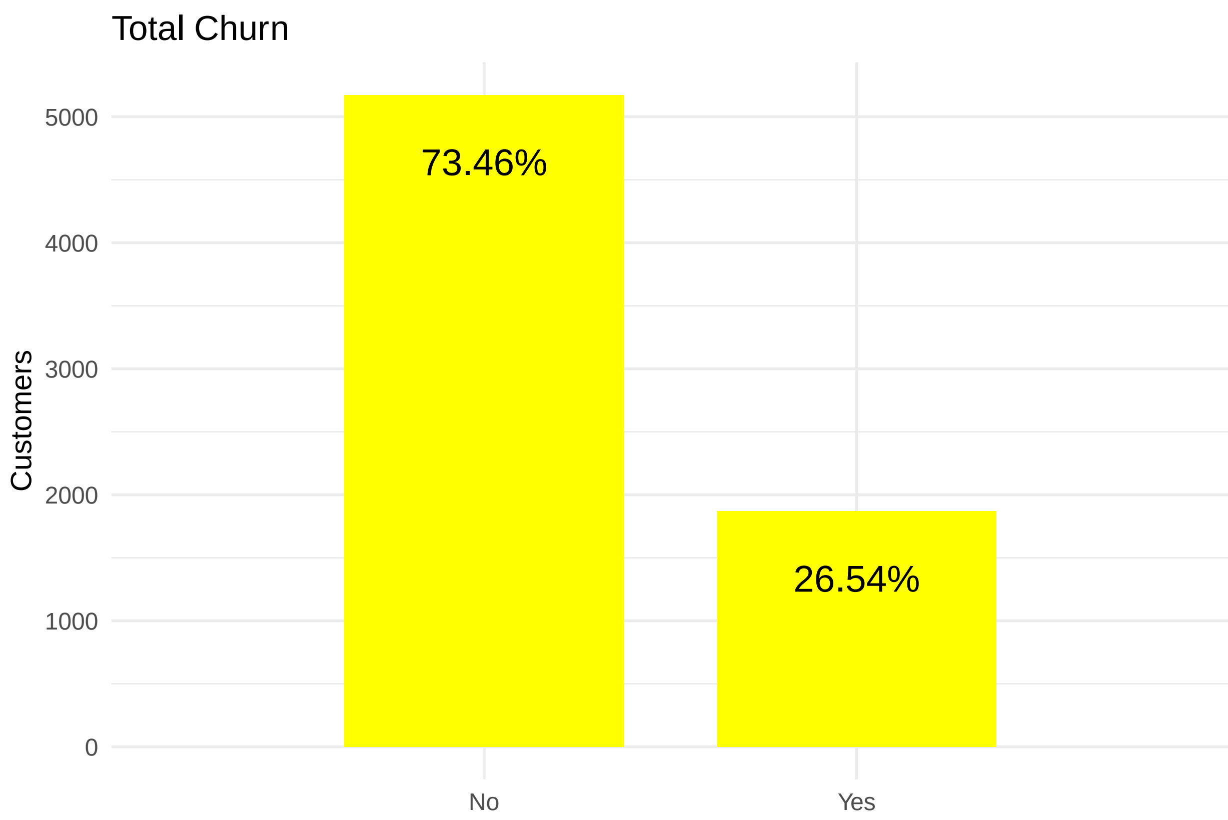
Data Exploration and Visualization

Total churn rate

Total churn rate in data set is 26.54%, which considers to be sufficiently high for telecom industry. Let's do breakdown of churn by variables to identify most critical drivers of churn for a company.

```
# plot total churn in data set
churn <- filter(churn_set, Churn == "Yes") #filter churn
non_churn <- filter(churn_set, Churn == "No") #filter non-churn
churn_plot <- ggplot(churn_set, aes(x=factor(Churn))) +
  geom_bar(fill="yellow", width = .75) +
  geom_text(aes(label = paste0(round(prop.table(..count..) * 100, 2), '%')),
            stat = 'count',
            position = position_dodge(2),
            size = 5,
            vjust = 3) +
  theme_minimal() +
  ggtitle('Total Churn') +
  xlab('') +
  ylab('Customers')
```

```r
churn_plot #churn plot
```



Data structure

There is the data structure in the data set (sample of 6 rows).

```r
# structure of data
head(churn_set)
```

```
##   customerID gender SeniorCitizen Partner Dependents tenure PhoneService
## 1 7590-VHVEG Female             0     Yes         No      1           No
## 2 5575-GNVDE   Male             0      No         No     34          Yes
## 3 3668-QPYBK   Male             0      No         No      2          Yes
## 4 7795-CFOCW   Male             0      No         No     45           No
## 5 9237-HQITU Female             0      No         No      2          Yes
## 6 9305-CDSKC Female             0      No         No      8          Yes
##      MultipleLines InternetService OnlineSecurity OnlineBackup DeviceProtection
## 1 No phone service             DSL             No          Yes               No
## 2               No             DSL            Yes           No              Yes
## 3               No             DSL            Yes          Yes               No
## 4 No phone service             DSL            Yes           No              Yes
## 5               No     Fiber optic             No           No               No
## 6              Yes     Fiber optic             No           No              Yes
##   TechSupport StreamingTV StreamingMovies       Contract PaperlessBilling
## 1          No          No              No Month-to-month              Yes
## 2          No          No              No       One year               No
## 3          No          No              No Month-to-month              Yes
```

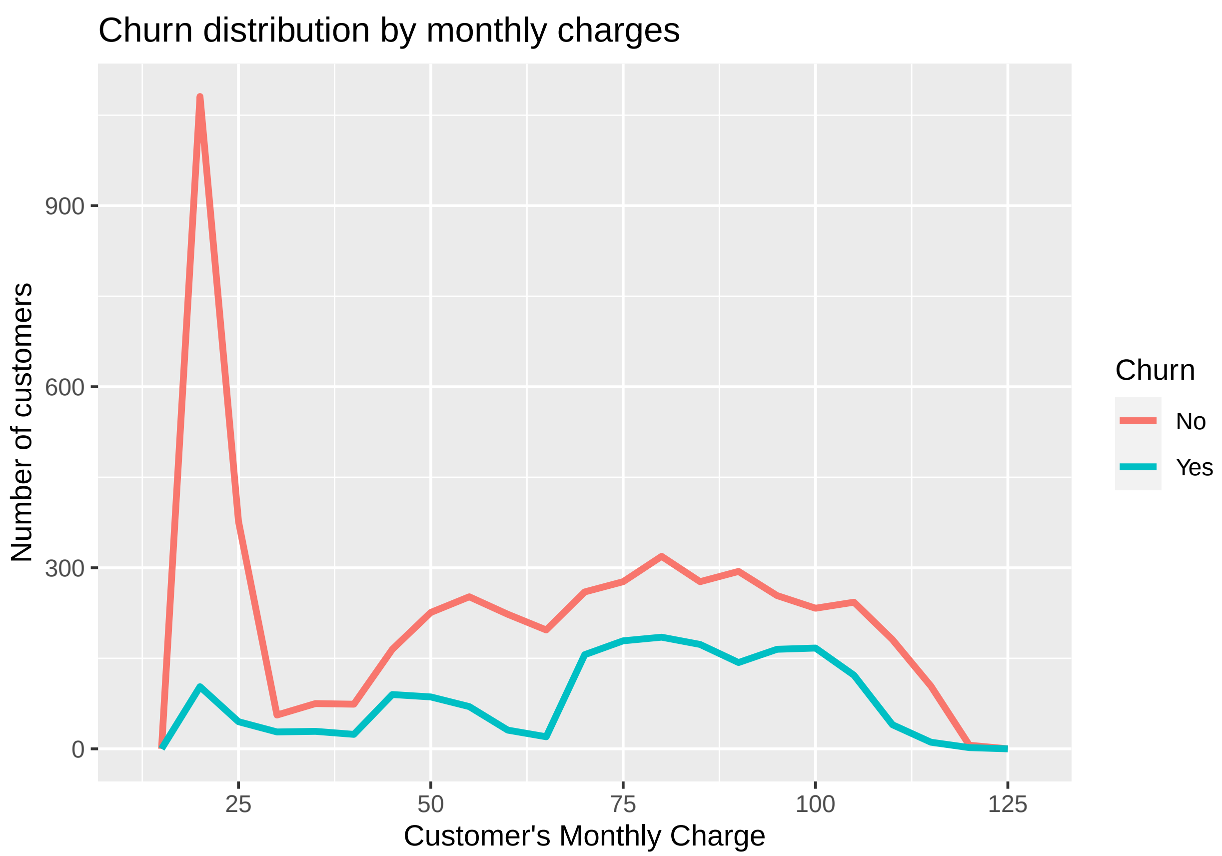
```
## 4         Yes          No                No        One year               No
## 5          No          No                No Month-to-month              Yes
## 6          No         Yes               Yes Month-to-month              Yes
##              PaymentMethod MonthlyCharges TotalCharges Churn
## 1          Electronic check          29.85        29.85    No
## 2              Mailed check          56.95      1889.50    No
## 3              Mailed check          53.85       108.15   Yes
## 4 Bank transfer (automatic)          42.30      1840.75    No
## 5          Electronic check          70.70       151.65   Yes
## 6          Electronic check          99.65       820.50   Yes
```

Continuous Variables analysis

Monthly charges distribution

First of all, churn distribution by monthly charges will be analyzed.

```
# Churn distribution by monthly charges
ggplot(churn_set, aes(MonthlyCharges, color = Churn)) +
  geom_freqpoly(binwidth = 5, size = 1.2) +
  labs(title = "Churn distribution by monthly charges",
       x = "Customer's Monthly Charge",
       y = "Number of customers")
```

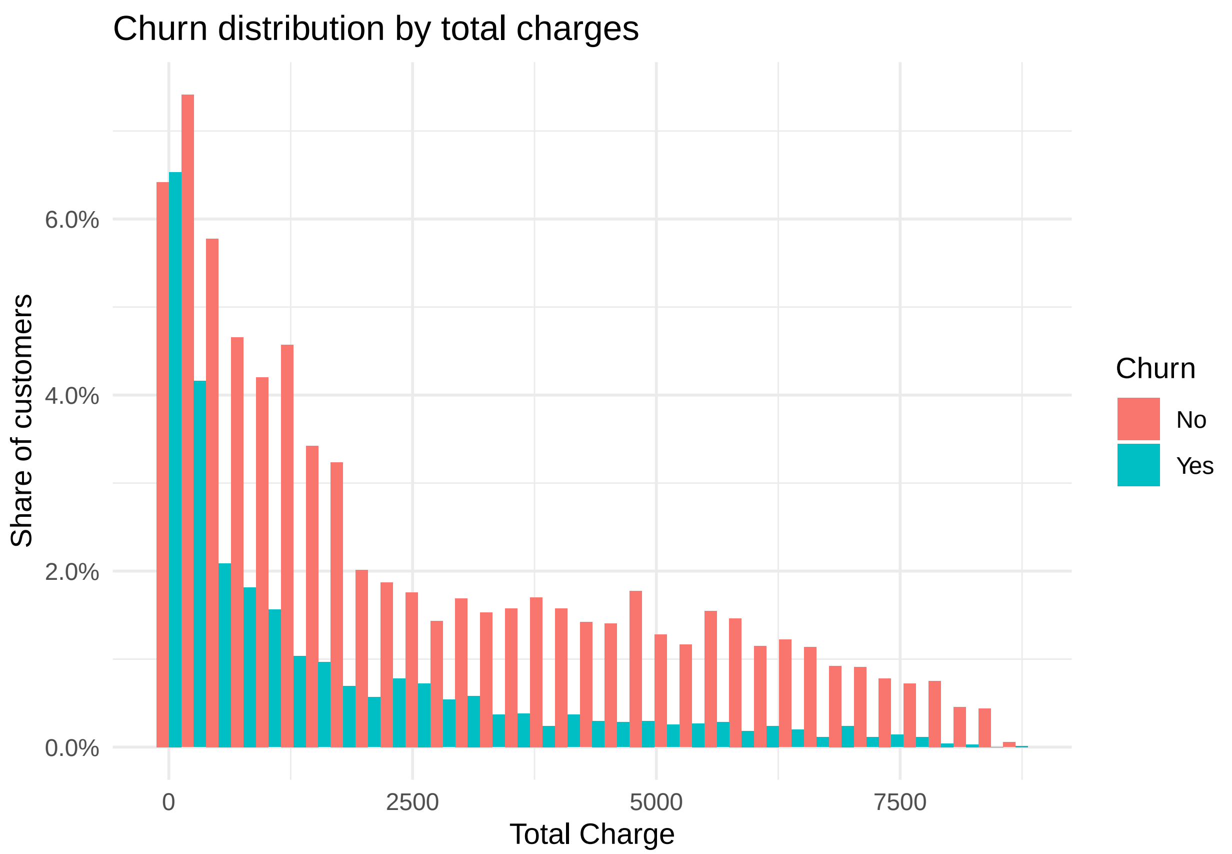


Key findings: Customers with monthly charged less than 25 have significantly lower churn than in other price range. The highest churn in the monthly charges rage between 70 and 100.

Total charges distribution

There is the churn distribution by total charges.

```
# Churn distribution by total charges (histogram)
ggplot(churn_set, aes(x=round(TotalCharges, digits=0),
                     y = (..count..)/sum(..count..),
                     fill=Churn))+
  geom_histogram(stat = 'bin',
                 bins = 35,
                 position=position_dodge()) +
  scale_y_continuous(labels=scales::percent) +
  ggtitle('Churn distribution by total charges') +
  xlab('Total Charge') +
  ylab('Share of customers') +
  theme_minimal()
```



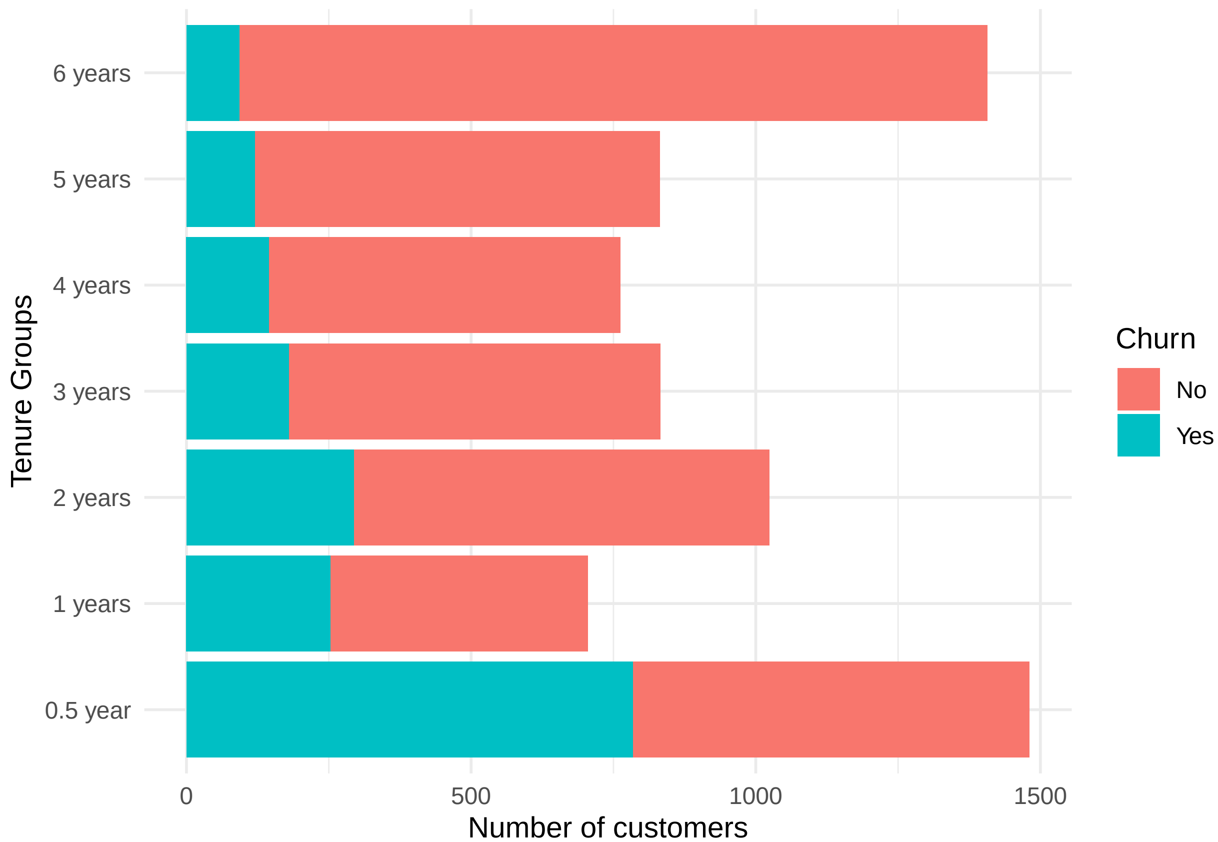
Key findings: Churn customers have pretty similar distribution in comparison with non-churn customers among all total charge range. However, shares of churn and non-churn customers are almost equal (around 6.5%) with zero total charge.

Customer's tenure distribution

Customers will be divided by tenure groups (half-year, 1 year, 2 years and so on) to reveal tenure effect on churn.

```
# creating tenure groups
   churn_gr <- churn_set %>%
     mutate(tenure_gr = case_when(tenure <= 6 ~ "0.5 year",
                                  tenure > 6 & tenure <= 12 ~ "1 years",
                                  tenure > 12 & tenure <= 24 ~ "2 years",
```

```
                                                tenure > 24 & tenure <= 36 ~ "3 years",
                                                tenure > 36 & tenure <= 48 ~ "4 years",
                                                tenure > 48 & tenure <= 60 ~ "5 years",
                                                tenure > 60 & tenure <= 72 ~ "6 years"))


# churn distribution by  tenure groups
ggplot(churn_gr, aes(tenure_gr, fill = Churn))+
    geom_bar()+
    coord_flip()+   #rotate the graph horizontally
    labs(y = "Number of customers", x = "Tenure Groups")+
    theme_minimal()
```



Key findings: Churn is mainly driven by new customers (who are using company services less than half of year). At the same time, churn is much lower among loyal customers who are staying with a company for longer period of time.

Service types analysis

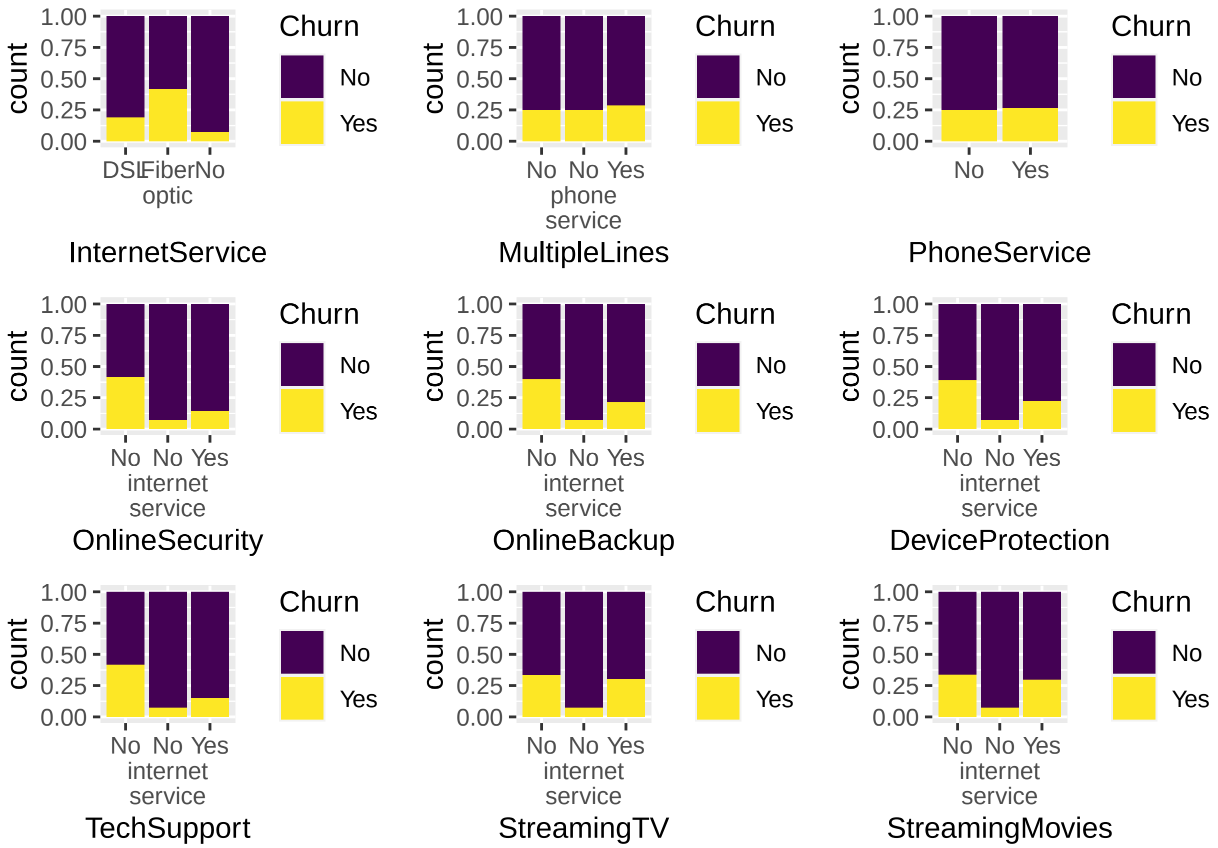
Churn by Type of services

The analysis of churn among type of services is below.

```
# plot churn by type of services
options(repr.plot.width = 10, repr.plot.height = 10)
plot_grid(
```

```r
#plot InternetService
ggplot(churn_set, aes(x=InternetService, fill=Churn))+
geom_bar(position = 'fill')+
    scale_fill_ordinal()+
scale_x_discrete(labels = function(x) str_wrap(x, width = 5)),

#plot MultipleLines
ggplot(churn_set, aes(x=MultipleLines, fill=Churn))+
geom_bar(position = 'fill')+
    scale_fill_ordinal()+
scale_x_discrete(labels = function(x) str_wrap(x, width = 5)),

#plot PhoneService
ggplot(churn_set, aes(x=PhoneService, fill=Churn))+
geom_bar(position = 'fill')+
    scale_fill_ordinal()+
scale_x_discrete(labels = function(x) str_wrap(x, width = 10)),

#plot OnlineSecurity
ggplot(churn_set, aes(x=OnlineSecurity, fill=Churn))+
geom_bar(position = 'fill')+
    scale_fill_ordinal()+
scale_x_discrete(labels = function(x) str_wrap(x, width = 10)),

#plot OnlineBackup
ggplot(churn_set, aes(x=OnlineBackup, fill=Churn))+
geom_bar(position = 'fill')+
    scale_fill_ordinal()+
scale_x_discrete(labels = function(x) str_wrap(x, width = 10)),

#plot DeviceProtection
ggplot(churn_set, aes(x=DeviceProtection, fill=Churn))+
geom_bar(position = 'fill')+
   scale_fill_ordinal()+
scale_x_discrete(labels = function(x) str_wrap(x, width = 10)),

#plot TechSupport
ggplot(churn_set, aes(x=TechSupport, fill=Churn))+
geom_bar(position = 'fill')+
   scale_fill_ordinal()+
scale_x_discrete(labels = function(x) str_wrap(x, width = 10)),

#plot StreamingTV
ggplot(churn_set, aes(x=StreamingTV, fill=Churn))+
geom_bar(position = 'fill')+
    scale_fill_ordinal()+
scale_x_discrete(labels = function(x) str_wrap(x, width = 10)),

#plot StreamingMovies
ggplot(churn_set, aes(x=StreamingMovies, fill=Churn))+
geom_bar(position = 'fill')+
    scale_fill_ordinal()+
scale_x_discrete(labels = function(x) str_wrap(x, width = 10)),
```

```
    #align horizontal reference lines
    align = "h")
```



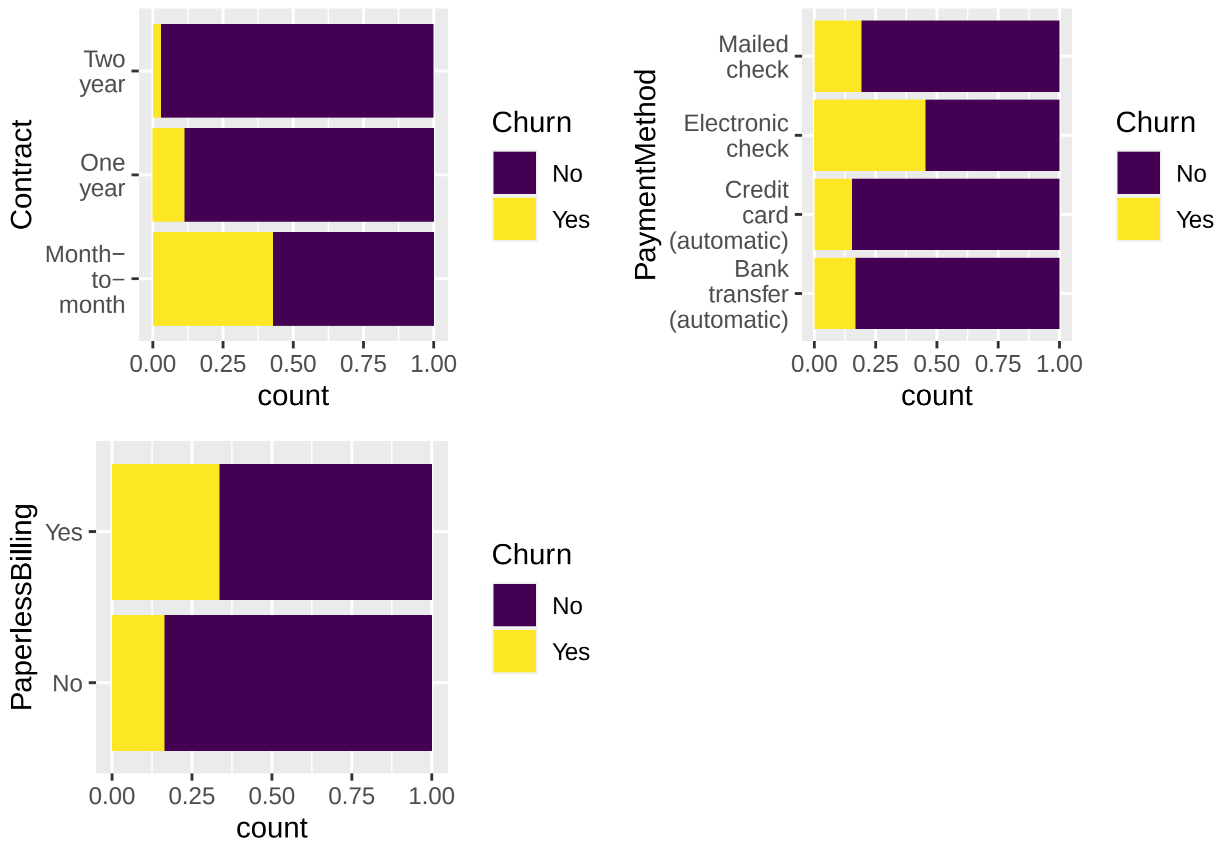
Key findings: Availability of Phone Services and Multiple lines have no significant impact on churn rate. Among Internet Services the highest churn is in fiber optic. Customers with subscriptions on such services as Device Protection, Online Backup, Online Security and Tech Support demonstrate lower churn rate vs. customers who have no subscription on it. At the same time, such services as Phone Service, Multiple Lines, Streaming Movies and Streaming TV have no significant difference in churn rates in comparison between customers who are using this service and who are not using it.

Account data analysis

Customer account data covers information about type of contract (month-to-month or longer), selected method of payment and billing options (online or paperless).

```
# plot churn rate in Customer account data (Contract, PaymentMethod, PaperlessBilling)
plot_grid(
    ggplot(churn_set, aes(x=Contract, fill=Churn))+
    geom_bar(position = 'fill')+
      coord_flip()+    #rotate the graph horizontally
        scale_fill_ordinal()+
    scale_x_discrete(labels = function(x) str_wrap(x, width = 5)),

    ggplot(churn_set, aes(x=PaymentMethod, fill=Churn))+
    geom_bar(position = 'fill')+
      coord_flip()+    #rotate the graph horizontally
```

```
      scale_fill_ordinal()+
    scale_x_discrete(labels = function(x) str_wrap(x, width = 5)),

    ggplot(churn_set, aes(x=PaperlessBilling, fill=Churn))+
    geom_bar(position = 'fill')+
      coord_flip()+   #rotate the graph horizontally
        scale_fill_ordinal()+
    scale_x_discrete(labels = function(x) str_wrap(x, width = 20)))
```



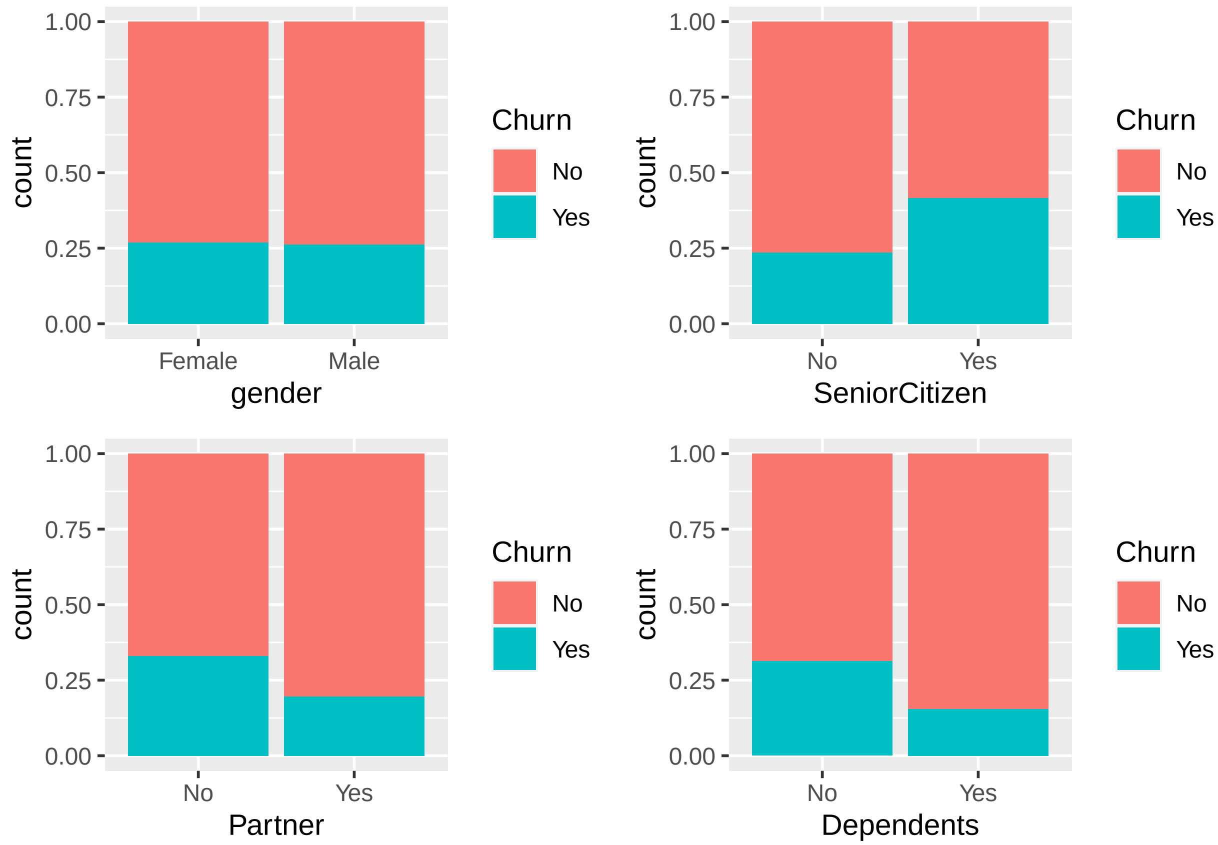
Key findings: Month-to-month contract customers have very high churn rate. At the same time, there is very low churn rate among loyal customers who stay with a company one year and more. Higher churn is among customers who selected paperless billing. Customers who pay with Electronic check have higher churn rate than all others payment method options.

Demographic data analysis

Demographic data can be also useful in terms of revealing impact on customers churn rate. There is below a breakdown of churn by gender, among senior citizens, and availability of partner and / or dependents.

```
# Rename Senior citizen factors from 1/0 to Yes/No
churn_set$SeniorCitizen = mapvalues(churn_set$SeniorCitizen,
                                    from = c("0", "1"), to = c("No", "Yes"))

# plot Churn by Gender, Senior Citizen, Availability of Partner and Dependents
plot_grid(
gender <- ggplot(churn_set) +
  geom_bar(aes(x = gender, fill = Churn), position = "fill", stat = "count"),
```

```r
senior <- ggplot(churn_set) +
  geom_bar(aes(x = SeniorCitizen, fill = Churn), position = "fill", stat = "count"),
partners <- ggplot(churn_set) +
  geom_bar(aes(x = Partner, fill = Churn), position = "fill", stat = "count"),
dependents <- ggplot(churn_set) +
  geom_bar(aes(x = Dependents, fill = Churn), position = "fill", stat = "count"))
```



Key findings: Gender has no impact on churn. Senior customers are prone to higher churn. Moreover, customers without family and / or dependents have high churn rate as well.

Data Wrangling and Structuring

Structuring data

Based on data analysis, such variables as *Gender*, *PhoneService*, *MultipleLines* have no significant impact on company churn. *CustomerID* column is not related for churn prediction modeling. Therefore, these columns will be eliminated from the data set for modeling.

```r
# Remove unnecessary columns
model_set <- churn_set %>%
    select( -customerID, -gender,-PhoneService, -MultipleLines)

# change the charactor variables to factors
model_set <- model_set  %>%
  mutate_if(is.character, as.factor)

# check changes
```

```
str(model_set)
```

```
## 'data.frame':    7043 obs. of  17 variables:
##  $ SeniorCitizen   : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Partner         : Factor w/ 2 levels "No","Yes": 2 1 1 1 1 1 1 1 2 1 ...
##  $ Dependents      : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 2 1 1 2 ...
##  $ tenure          : int  1 34 2 45 2 8 22 10 28 62 ...
##  $ InternetService : Factor w/ 3 levels "DSL","Fiber optic",..: 1 1 1 1 2 2 2 1 2 1 ...
##  $ OnlineSecurity  : Factor w/ 3 levels "No","No internet service",..: 1 3 3 3 1 1 1 3 1 3 ...
##  $ OnlineBackup    : Factor w/ 3 levels "No","No internet service",..: 3 1 3 1 1 1 3 1 1 3 ...
##  $ DeviceProtection: Factor w/ 3 levels "No","No internet service",..: 1 3 1 3 1 3 1 1 3 1 ...
##  $ TechSupport     : Factor w/ 3 levels "No","No internet service",..: 1 1 1 3 1 1 1 1 3 1 ...
##  $ StreamingTV     : Factor w/ 3 levels "No","No internet service",..: 1 1 1 1 1 3 3 1 3 1 ...
##  $ StreamingMovies : Factor w/ 3 levels "No","No internet service",..: 1 1 1 1 1 3 1 1 3 1 ...
##  $ Contract        : Factor w/ 3 levels "Month-to-month",..: 1 2 1 2 1 1 1 1 1 2 ...
##  $ PaperlessBilling: Factor w/ 2 levels "No","Yes": 2 1 2 1 2 2 2 1 2 1 ...
##  $ PaymentMethod   : Factor w/ 4 levels "Bank transfer (automatic)",..: 3 4 4 1 3 3 2 4 3 1 ...
##  $ MonthlyCharges  : num  29.9 57 53.9 42.3 70.7 ...
##  $ TotalCharges    : num  29.9 1889.5 108.2 1840.8 151.7 ...
##  $ Churn           : Factor w/ 2 levels "No","Yes": 1 1 2 1 2 2 1 1 2 1 ...
```

Split train/test sets

Training dataset will have 80% of the original data, and test set - 20%.

```
# Set seed
set.seed(333)

# Split data: 80% for train set, 20% for test set
index <- createDataPartition(y = model_set$Churn, p = 0.8, list = FALSE)

churn_train <- model_set[index,]
churn_test <- model_set[ -index,]
```

Data Modeling

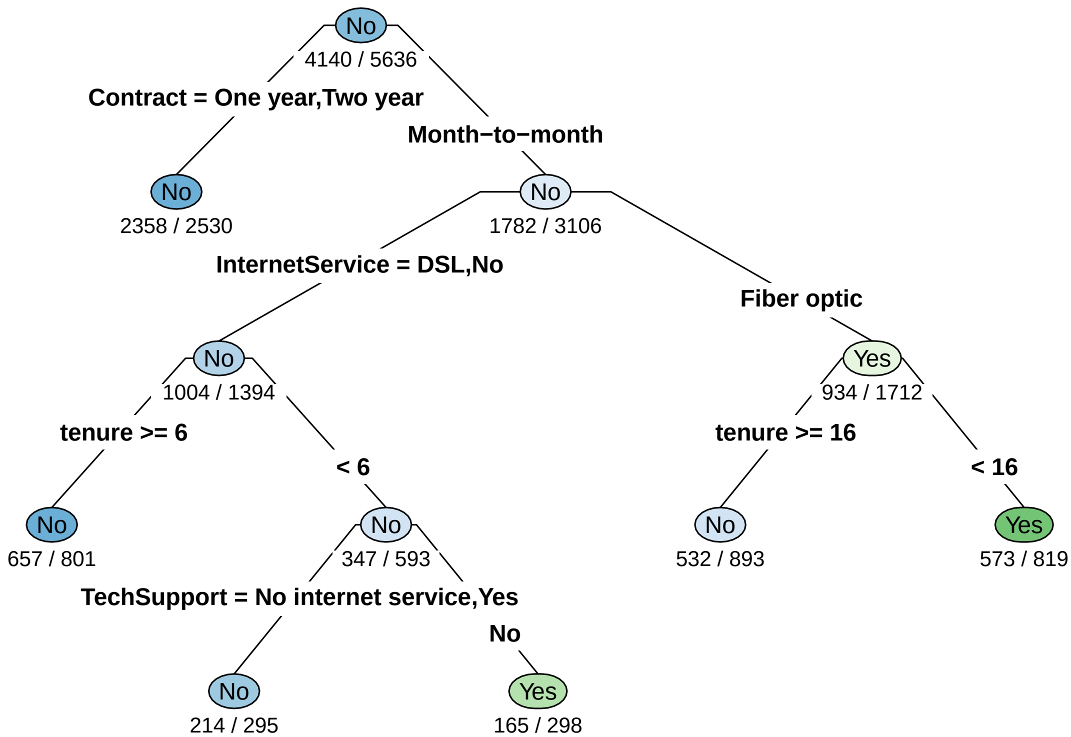
Decision Tree Model

Classification (Decision) Tree is a type of supervised learning algorithm that can be used in both regression and classification problems. It works for both categorical and continuous input and output variables.

Train model

First, the Decision tree model will be trained on train set of data and the results will be visualized with the help of model tree plot.

```
# train Decision Tree model on train set
tree_fit <- rpart(Churn ~ ., data = churn_train,
  method = "class")

# plot Decision tree
rpart.plot(
  tree_fit,
  type = 4,
  extra = 2,
  under = TRUE,
  fallen.leaves = F)
```



Based on the visualization of decision tree, there are two the most vulnerable categories of customers which are tending to churn:

1. Customers on Month-to-Month contract who are using Fiber Optic Service for the period less than 16 months.
2. Customers on Month-to-Month contract who are using DSL Internet Service without Tech Support service for the period less than 6 months.

Predicting with model

Test set of data will be used to predict churn. The Accuracy and other parameters of Decision tree model will be represented in Confusion Matrix and Statistics.

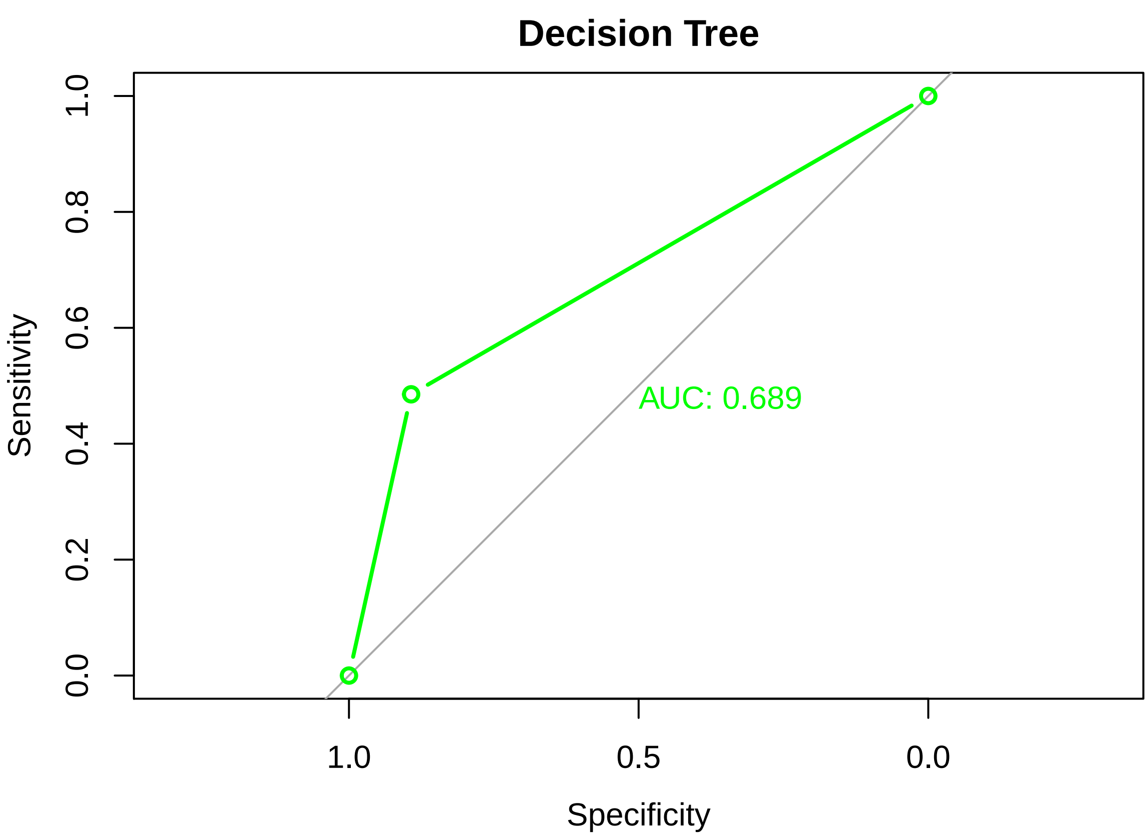
```r
# predict churn on test set
tree_pred <- predict(tree_fit, churn_test,
                     type = "class")

# accuracy of model
confusionMatrix(tree_pred, churn_test$Churn)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  923 192
##        Yes 111 181
##
```

```
##                 Accuracy : 0.7846
##                   95% CI : (0.7622, 0.8059)
##      No Information Rate : 0.7349
##      P-Value [Acc > NIR] : 9.119e-06
##
##                    Kappa : 0.4061
##
##   Mcnemar's Test P-Value : 4.309e-06
##
##              Sensitivity : 0.8926
##              Specificity : 0.4853
##           Pos Pred Value : 0.8278
##           Neg Pred Value : 0.6199
##               Prevalence : 0.7349
##           Detection Rate : 0.6560
##     Detection Prevalence : 0.7925
##        Balanced Accuracy : 0.6890
##
##         'Positive' Class : No
##
```

ROC plot of Decision Tree Model

```
# plot ROC and find AUC for Decision Tree Model
plot.roc(as.numeric(churn_test$Churn), as.numeric(tree_pred),
         main="Decision Tree", lwd=2, type="b",print.auc=TRUE,col ="green")
```



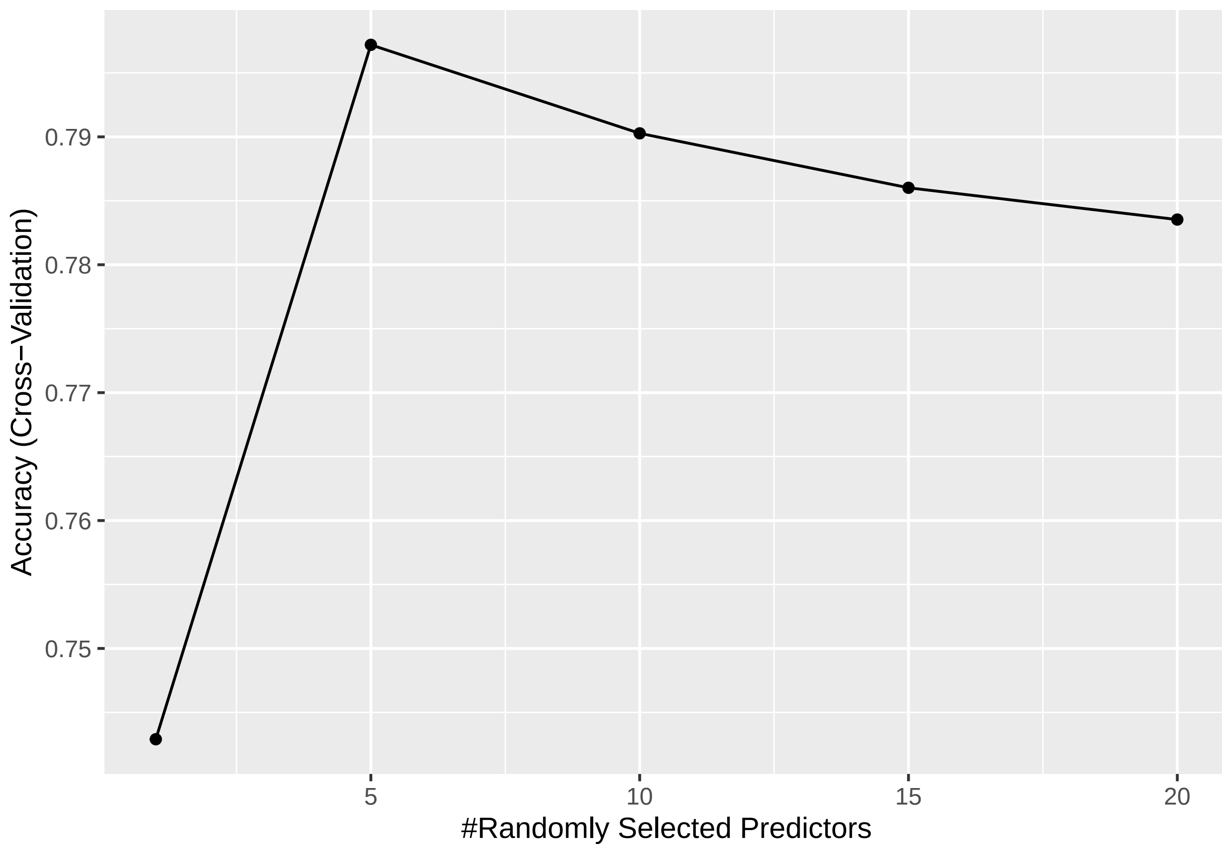
Random Forest Model

Random forests are a very popular machine learning approach that comprises a random collection of a forest tree (decision trees). The random forest algorithm creates multiple decision trees and merges them together to obtain a more stable and accurate prediction. Generally speaking, the more trees in the forest, the more robust would be the prediction and thus higher accuracy.

Cross validation

The fitting of Random forest model is slower procedure rather than the predicting. To make the process faster, 5-fold cross validation will be used only. A random sample of the observations will be taken when building each tree. In the random forest, number of variables available for splitting at each tree node is referred to as the **mtry** parameter, which is tune parameter.

```r
# assign train and test sets for Random Forest model
RF_train <- churn_train
RF_test <- churn_test

# set 5-fold cross validation to make the process faster
control <- trainControl(method="cv", number = 5)

# create list of mtry values (as tune parameter)
grid <- data.frame(mtry = c(1, 5, 10, 15, 20))

# cross validation of accuracy with ntree=150 for faster computing
train_rf <-  train(Churn~., RF_train,
                   method = "rf",
                   ntree = 150,
                   trControl = control,
                   tuneGrid = grid)
# plot results
ggplot(train_rf)
```



Display the best mtry value for the model.

```
# display the best mtry value for the model
train_rf$bestTune
```

```
##   mtry
## 2    5
```

The best mtry is 5.

Fiting model

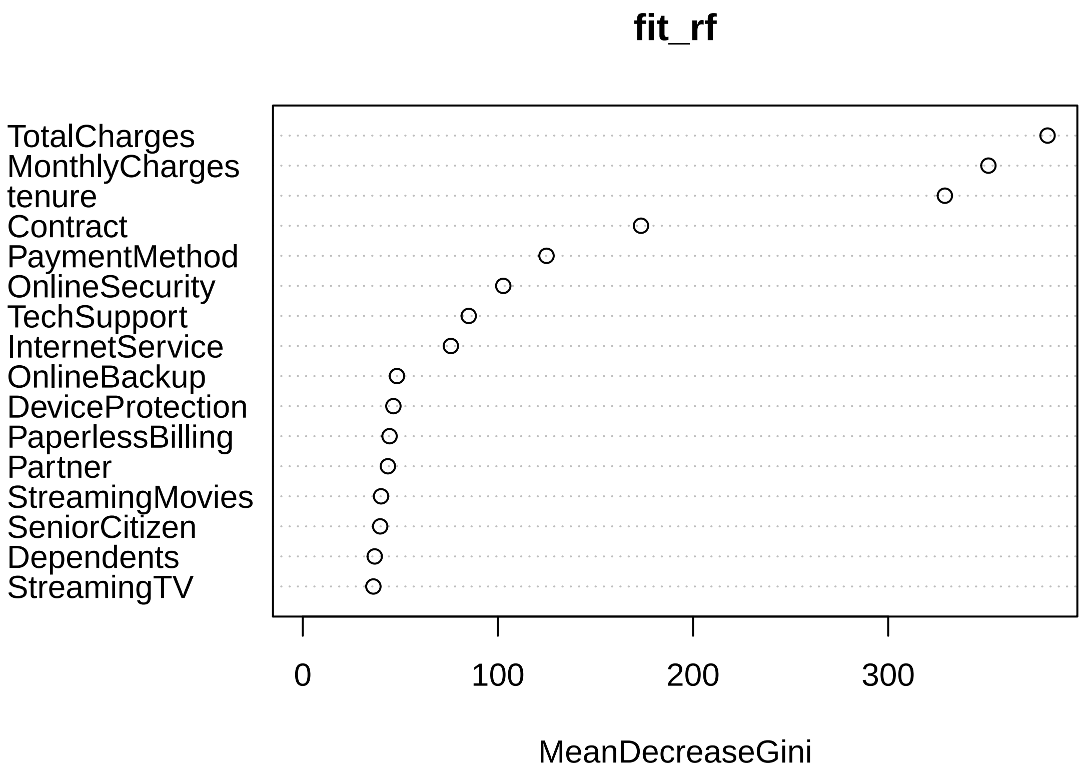
Fiting the Random Forest Model on train set using the best tune mtry value.

```
# fiting RF model with using the best tune mtry
fit_rf <- randomForest(Churn~., RF_train,
                       minNode = train_rf$bestTune$mtry)
```

Parameter ranking

'VarImpPlot function' will plot all variables which were used for modeling and provide their ranking of importance for modeling. In our case, the most important variables in the Random Forest model are total and monthly charges, tenure and contract.

```
# Varplot of different parameters
varImpPlot(fit_rf)
```



Predicting with Random Forest model

The prediction of churn with the Random Forest Model will be executed on test dataset. The confusion matrix will present the Accuracy and other parameters of this model.

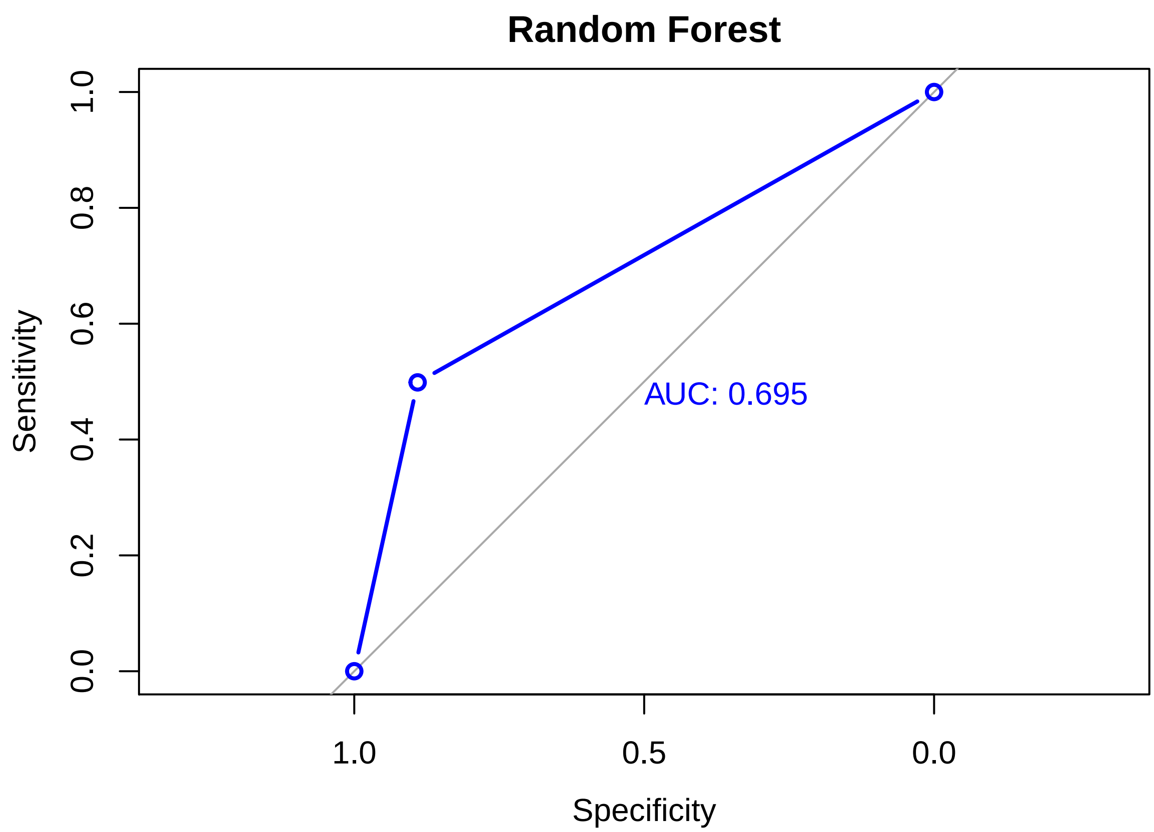
```
# predict with RF model
rf_pred <- predict(fit_rf, RF_test)

# check accuracy with confusion matrix
confusionMatrix(RF_test$Churn, rf_pred)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  921 113
##        Yes 187 186
##
##                Accuracy : 0.7868
##                  95% CI : (0.7644, 0.8079)
##     No Information Rate : 0.7875
##     P-Value [Acc > NIR] : 0.5414
##
##                   Kappa : 0.4157
##
##  Mcnemar's Test P-Value : 2.502e-05
##
```

```
##          Sensitivity : 0.8312
##          Specificity : 0.6221
##       Pos Pred Value : 0.8907
##       Neg Pred Value : 0.4987
##           Prevalence : 0.7875
##       Detection Rate : 0.6546
## Detection Prevalence : 0.7349
##    Balanced Accuracy : 0.7267
##
##     'Positive' Class : No
##
```

ROC plot of Random Forest Model

```r
# plot ROC and find AUC for Random Forest Model
plot.roc(as.numeric(RF_test$Churn), as.numeric(rf_pred),
        main="Random Forest", lwd=2, type="b",print.auc=TRUE,col ="blue")
```



Support Vector Machine (SVM)

Support Vector Machine (SVM) is one of the most prevailing supervised learning models with associated learning algorithms that analyze data and recognize patterns. It is powerful for solving both regression and classification problems.

Tuning parameters

Firstly, the SVM model parameters will be tuned on train set.

```r
# assign train and test sets for SVM model
SVM_train <- churn_train
SVM_test <- churn_test

# tuning parameters
tune_prm <- tune(svm,factor(Churn)~.,data = SVM_train)
```

Training SVM

Training the SVM model by using the tuned parameters from the training data set.

```r
# train SVM
SVM_model <- svm(SVM_train$Churn~., data=SVM_train,
                 type="C-classification", gamma=tune_prm$best.model$gamma,
                 cost=tune_prm$best.model$cost,
                 kernel="radial")
```

Predicting with SVM

Predicting the SVM Model on test set. The confusion matrix will present the accuracy and other parameters of SVM model.

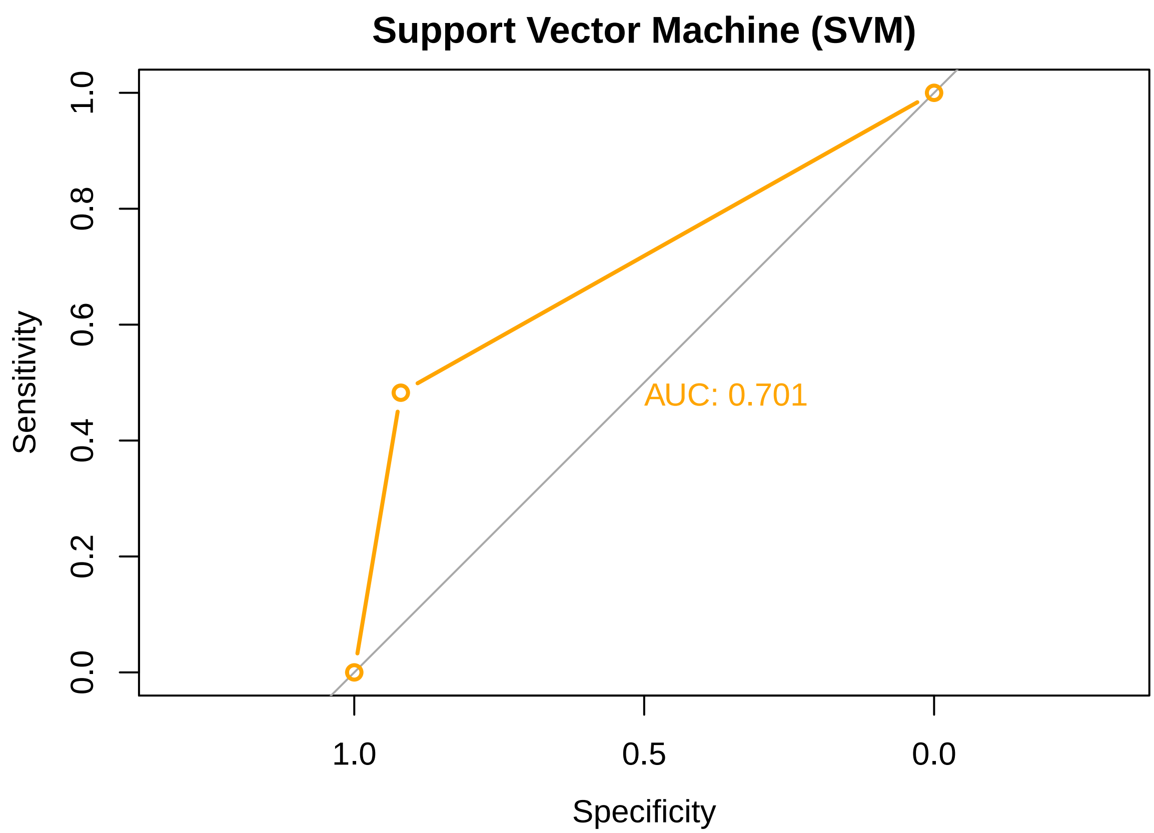
```r
# predict with SMV model
SVM_prd <- predict(SVM_model,newdata=SVM_test)

# check accuracy with confusion matrix
confusionMatrix(SVM_prd,SVM_test$Churn)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  951 193
##        Yes  83 180
##
##                Accuracy : 0.8038
##                  95% CI : (0.7821, 0.8243)
##     No Information Rate : 0.7349
##     P-Value [Acc > NIR] : 9.254e-10
##
##                   Kappa : 0.4442
##
##  Mcnemar's Test P-Value : 5.344e-11
##
##             Sensitivity : 0.9197
##             Specificity : 0.4826
##          Pos Pred Value : 0.8313
##          Neg Pred Value : 0.6844
##              Prevalence : 0.7349
##          Detection Rate : 0.6759
##    Detection Prevalence : 0.8131
##       Balanced Accuracy : 0.7012
##
##        'Positive' Class : No
##
```

ROC plot of SVM model

The plot below is showing the ROC for SVM model and represents AUC.

```r
# plot ROC and find AUC for SVM model
plot.roc (as.numeric(SVM_test$Churn), as.numeric(SVM_prd),
            main="Support Vector Machine (SVM)", lwd=2, type="b",
            print.auc=TRUE,col ="orange")
```



RESULTS

This section will discuss the results of modeling in terms of churn prediction on available data. For this purpose, confusion Matrix and ROC plot of each model will be compared. Based on the comparison of models results, the best approach for churn prediction will be selected.

Confusion Matrix Comparison

Decision Tree model

There are results of the confusion matrix for Decision Tree model.

```r
# confusion matrix for Decision Tree model
confusionMatrix(tree_pred, churn_test$Churn)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  923 192
##        Yes 111 181
```

```
##
##                Accuracy : 0.7846
##                  95% CI : (0.7622, 0.8059)
##     No Information Rate : 0.7349
##     P-Value [Acc > NIR] : 9.119e-06
##
##                   Kappa : 0.4061
##
##  Mcnemar's Test P-Value : 4.309e-06
##
##             Sensitivity : 0.8926
##             Specificity : 0.4853
##          Pos Pred Value : 0.8278
##          Neg Pred Value : 0.6199
##              Prevalence : 0.7349
##          Detection Rate : 0.6560
##    Detection Prevalence : 0.7925
##       Balanced Accuracy : 0.6890
##
##        'Positive' Class : No
##
```

Random Forest model

There are results of the confusion matrix for Random Forest model.

```r
# confusion matrix for Random Forest model
confusionMatrix(RF_test$Churn, rf_pred)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  921 113
##        Yes 187 186
##
##                Accuracy : 0.7868
##                  95% CI : (0.7644, 0.8079)
##     No Information Rate : 0.7875
##     P-Value [Acc > NIR] : 0.5414
##
##                   Kappa : 0.4157
##
##  Mcnemar's Test P-Value : 2.502e-05
##
##             Sensitivity : 0.8312
##             Specificity : 0.6221
##          Pos Pred Value : 0.8907
##          Neg Pred Value : 0.4987
##              Prevalence : 0.7875
##          Detection Rate : 0.6546
##    Detection Prevalence : 0.7349
##       Balanced Accuracy : 0.7267
##
##        'Positive' Class : No
##
```

SVM model

There are results of the confusion matrix for SVM model.

```
# confusion matrix for SVM model
confusionMatrix(SVM_prd,SVM_test$Churn)
```

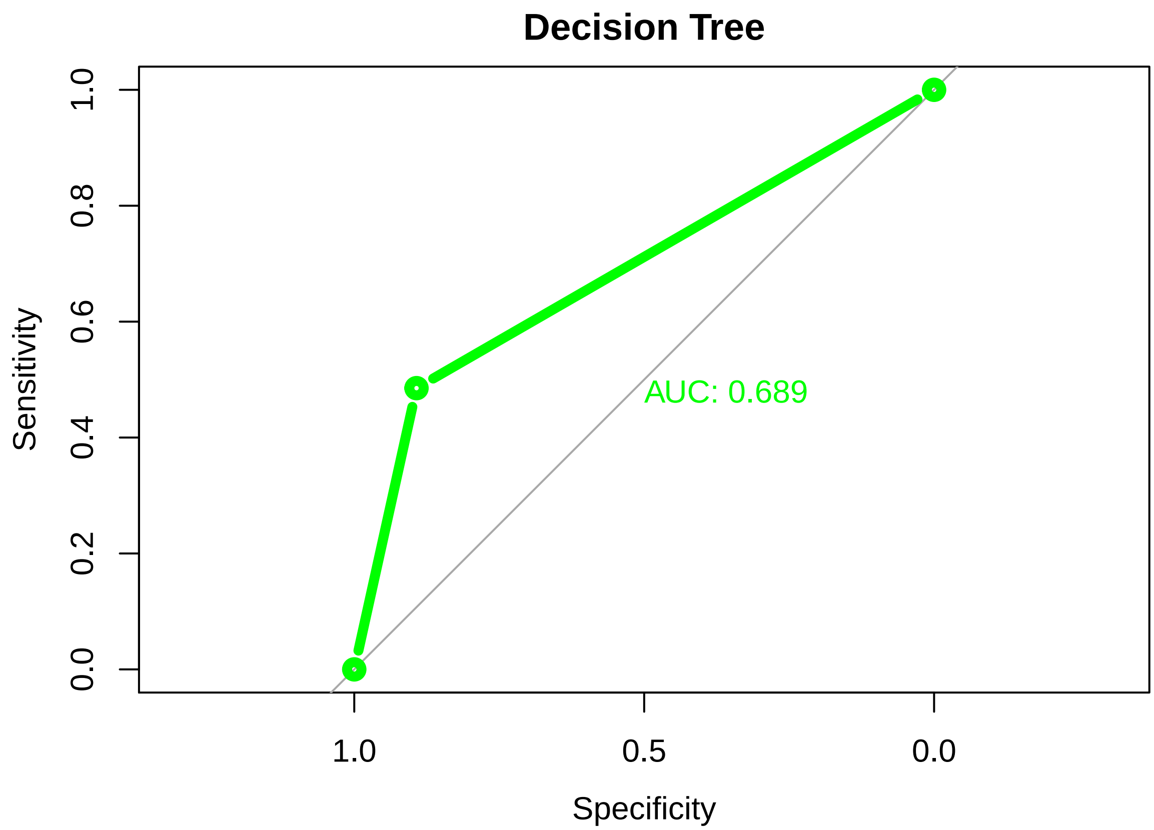
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  951 193
##        Yes  83 180
##
##                Accuracy : 0.8038
##                  95% CI : (0.7821, 0.8243)
##     No Information Rate : 0.7349
##     P-Value [Acc > NIR] : 9.254e-10
##
##                   Kappa : 0.4442
##
##  Mcnemar's Test P-Value : 5.344e-11
##
##             Sensitivity : 0.9197
##             Specificity : 0.4826
##          Pos Pred Value : 0.8313
##          Neg Pred Value : 0.6844
##              Prevalence : 0.7349
##          Detection Rate : 0.6759
##    Detection Prevalence : 0.8131
##       Balanced Accuracy : 0.7012
##
##        'Positive' Class : No
##
```

Based on the comparison of confusion matrix for each model, all three models display almost the same level of Accuracy (around 0.8). If the goal of project is to provide high Accuracy and Sensitivity then SVM model can be preferred. At the same time, the Random Forest model has the best combination of balanced values for other parameters of the confusion matrix (Sensitivity, Specificity and Prevalence).

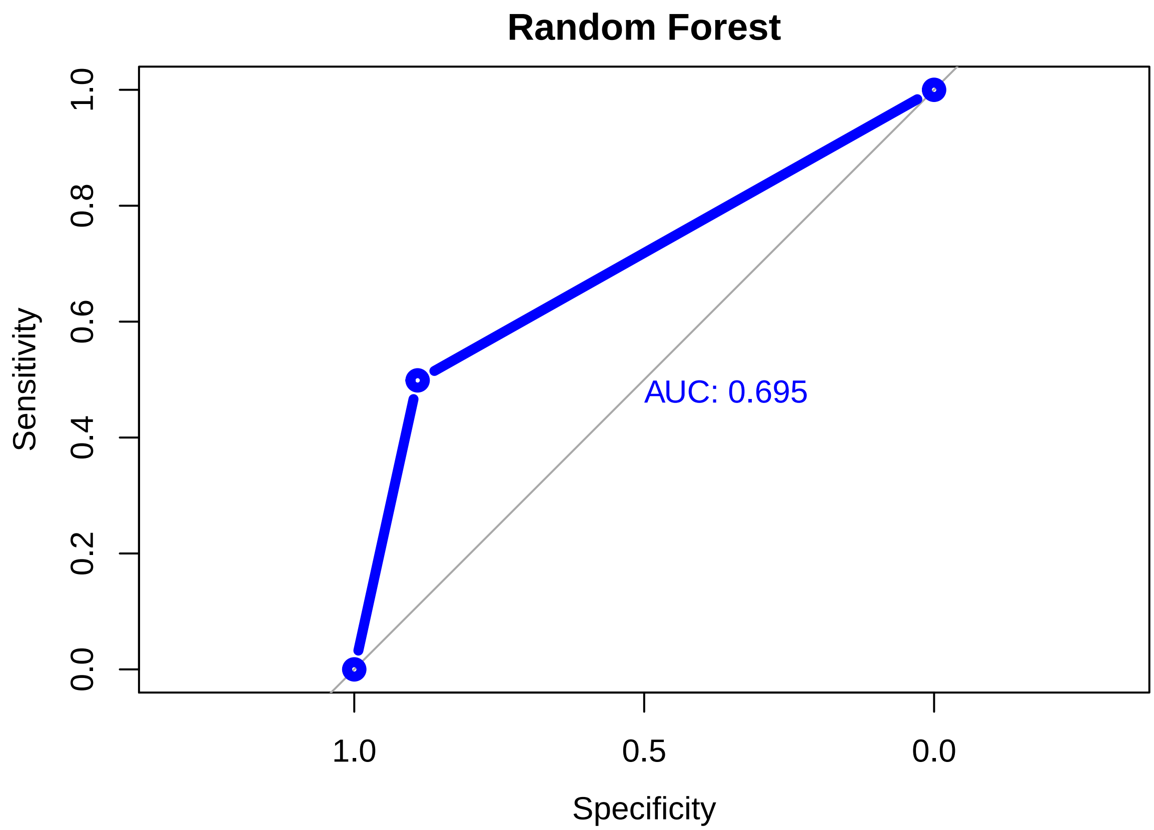
ROC plots comparison

ROC (Receiver Operator Characteristic) curve is a graphical tool for diagnostic test evaluation. In a ROC curve the true positive rate (Sensitivity) is plotted in function of the false positive rate (1-Specificity) for different cut-off points of a parameter. Therefore, the closer the ROC curve is to the upper left corner, the higher the overall accuracy of the model. The area under the ROC curve (AUC) is a measure of how well a parameter can discern between two churn and non-churn. Higher the AUC, better the model is at predicting.

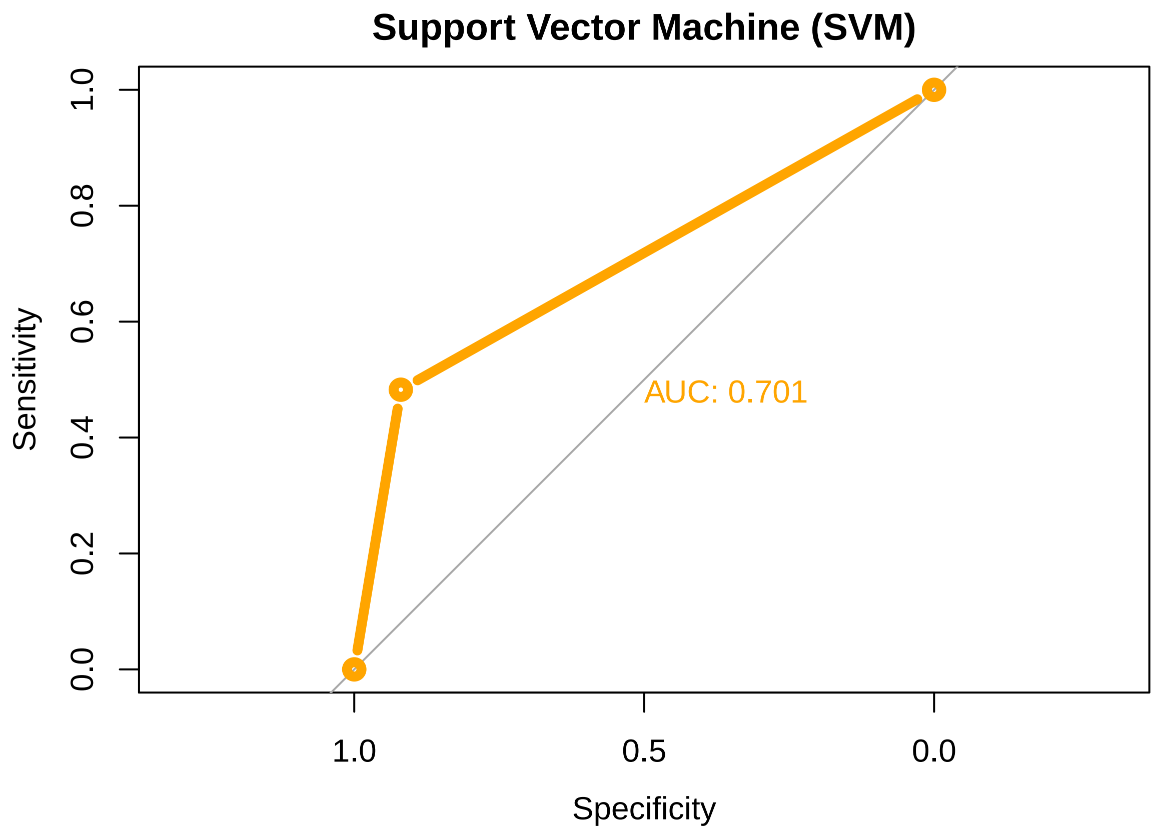
```
#  ROC plot of Decision Tree
plot.roc (as.numeric(churn_test$Churn), as.numeric(tree_pred),
          main="Decision Tree",lwd=5, type="b", print.auc=TRUE, col ="green")
```




```
# ROC plot of Random Forest
plot.roc (as.numeric(RF_test$Churn), as.numeric(rf_pred),
          main="Random Forest",lwd=5, type="b", print.auc=TRUE, col ="blue")
```




```
# ROC plot of SVM
plot.roc (as.numeric(SVM_test$Churn), as.numeric(SVM_prd),
          main="Support Vector Machine (SVM)",lwd=5, type="b", print.auc=TRUE, col ="orange")
```



The goal of the project is to predict churn of customers for telecom company in order to retain them by offering specials deals. In this case, the most appropriate model for churn prediction will be the *Random Forest* model due to balanced values of such parameters as Accuracy, Sensitivity, Specificity and Prevalence.

CONCLUSION

This section includes a brief summary of the report, its potential impact, its limitations, and future work.

Brief Summary

This project covers building and comparing machine learning algorithms to predict churn of a telecom company based on dataset, which was downloaded from website www.kaggle.com. The following 3 different models were implemented: Decision Tree, Random Forest and Support Vector Machine. Based on model results comparison, the Random forest model was identified as the most preferable for churn prediction due to balanced parameters from confusion matrix.

Potential Impact

The potential impact of the project is a possibility of predicting churn for telecom company by using advanced models based on the parameters from dataset.

Limitations

There are some limitations for the project:

1. The AUC all models is around 0.7 which is not too high.
2. Accuracy of all models is not very high as well (~ 0.8).
3. Limited capacity of computer.

Future work

Because of imbalanced dataset, some machine learning models such as kNN (k-nearest neighbors algorithm), SMOTE (Synthetic Minority Over-sampling Technique) or Artificial Neural Networks might be useful to try to improve current results of prediction.