Final Project

By Meghavi Vaghela

Telco Customer Churn Prediction using XGBoost Model

Course: CSC 8015 Data Mining &Predictive Analytics
Tools: R studio
Language: R
Date: 01 May 2024

Contents

Introduction

A brief description of the motivations behind the project.

Reducing churn is important for businesses because retaining existing customers is often more cost-effective than acquiring new ones. Additionally, loyal customers tend to spend more over time and may also act as advocates for the brand, helping to attract new customers through positive word-of-mouth.

Main motivations for me are gaining valuable skills like 'Predictive Analytics Skills' and skills applicable across various 'Business domains'.

Project Goals

This project explores the churn dataset 'Telco customer Churn' to identify the key drivers of churn and builds the best predictive model to predict churn. A strategy to reduce churn is presented and the proposal is evaluated against the model.

- Investigate and analyze the key factors contributing to customer churn within the business domain, with the aim of understanding the underlying drivers and patterns of churn behavior.
- Evaluate the effectiveness of different machine learning algorithms and feature engineering techniques in predicting customer churn, with the goal of identifying the most suitable approach.
- Develop a predictive model to accurately identify customers at risk of churning.

Why it is interesting or important?

- Customer churn prediction involves analyzing large and complex datasets encompassing various customer attributes, behaviors, and interactions.
- This presents an exciting challenge to leverage advanced analytics techniques and machine learning algorithms to extract meaningful insights and predictions.

The necessity of applying Data Mining techniques?

- To build models that predict the likelihood of customer churn based on historical data.
- By analyzing historical customer data, businesses can identify factors and variables that are predictive of churn behavior, such as transaction history, usage patterns, and customer interactions.
- Feature selection methods help identify the most important variables that contribute to churn prediction

In short, applying data mining techniques enables businesses to extract actionable insights from large volumes of data, build accurate predictive models, and develop effective retention strategies to mitigate churn and improve customer retention.

Keywords of the project

- Customer Churn
- Predictive Analytics
- Machine Learning
- Data Mining
- Feature engineering
- Best model selection
- Ensemble learning techniques
- Univariate Analysis
- Bivariate Analysis
- Data pre-processing
- Customer engagement
- Exploratory Data Analytics

Background

Background information regarding the dataset and overall information

This sample data module tracks a fictional telco company's customer churn based on a variety of possible factors. Company provided home phone and Internet services to 7043 customers **The churn column indicates whether or not the customer left within the last month**. Other columns include gender, dependents, monthly charges, and many with information about the types of services each customer has. Source: IBM. Location: Team content > Samples > Data. The Telco customer churn data module is composed of 5 uploaded files:

Telco_customer_churn_demographics.xlsx

Telco_customer_churn_location.xlsx

Telco_customer_churn_population.xlsx

Telco_customer_churn_services.xlsx

Telco_customer_churn_status.xlsx

Resource: https://community.ibm.com/community/user/businessanalytics/blogs/steven-macko/2019/07/11/telco-customer-churn-1113

Main focus in this project will be on the customer churn in response to the services that is provided by the company that is available in Telco_customer_churn_services.xlsx.

What is the research question /problem to be addressed?

"Predict behavior to retain customers by analysis of services provided to the customers."

Is the available data sufficient to deliver the intended message?

The dataset that is used here is a part of big dataset. The big dataset has other focuses like costumer status, demographics and location. There is a scope for improvement in the predictions using other focus areas as well.

Dataset description

Sources of the data

This dataset can be downloaded easily from kaggle platform.
https://www.kaggle.com/datasets/blastchar/telco-customer-churn/data

What dataset present and what it means for me

Each row represents a unique customer, with features such as customer ID, gender, senior citizen status, partnership status, tenure (months with the company), phone service, internet service type, online security, online backup, device protection, tech support, streaming TV and movies, contract type, paperless billing preference, payment method, monthly charges, total charges, and churn status.

For analysis, this dataset offers valuable insights into customer behaviour and churn patterns within the telecommunications industry. Working on a customer churn prediction project as a student can provide me with a rich learning experience that encompasses data pre-processing, Data visualization, Machine learning, model evaluation, feature engineering and Hyper Parameter Tuning.

Explore the data

EDA

There is no particular order for the above stages, but we can start by the loading of dataset that is available in '.csv' format from above mentioned link to R studio.

- Import important libraries.

R command:

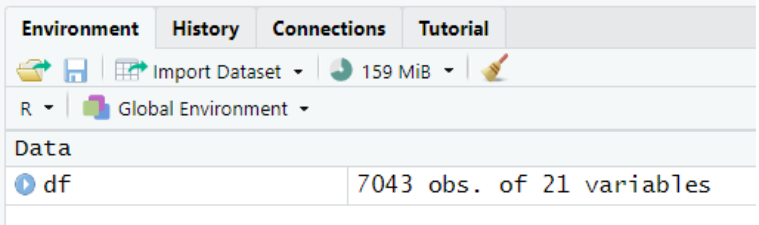
```
# Author: Meghavi Vaghela
# About: Customer Churn Prediction

# ---------------------------------LIBRARIES-----------------------
# install package
install.packages("cowplot")
install.packages("ggplot2")
install.packages("ggcorrplot")
install.packages("xgboost")
install.packages("pROC")
install.packages("forecast")


# Load important libraries
# library(data.table)
library(ggplot2)
library(cowplot)
library(ggcorrplot)
library(xgboost)
library(lattice)
library(caret)
library(pROC)
library(forecast)

```

- Load the dataset 'Telco-Customer-Churn.csv'.

R command:

```
# ---------------------------------DATASET IMPORT------------------
# Load data set
df = read.csv("Telco-Customer-Churn.csv")
```

Environment window:



- **Explore the dataset**

  - **Dimension**

    R command:

```
# View data set dimensions and summary statistics
dim(df)
head(df)
str(df)
summary(df)
```

    Console output:

```
> # View data set dimensions and summary statistics
> dim(df)
[1] 7043   21
> head(df)
  customerID gender SeniorCitizen Partner Dependents tenure PhoneService    MultipleLines
1 7590-VHVEG Female             0     Yes         No      1           No No phone service
2 5575-GNVDE   Male             0      No         No     34          Yes               No
3 3668-QPYBK   Male             0      No         No      2          Yes               No
4 7795-CFOCW   Male             0      No         No     45           No No phone service
5 9237-HQITU Female             0      No         No      2          Yes               No
6 9305-CDSKC Female             0      No         No      8          Yes              Yes
  InternetService OnlineSecurity OnlineBackup DeviceProtection TechSupport StreamingTV StreamingMovies
1             DSL             No          Yes               No          No          No              No
2             DSL            Yes           No              Yes          No          No              No
3             DSL            Yes          Yes               No          No          No              No
4             DSL            Yes           No              Yes         Yes          No              No
5     Fiber optic             No           No               No          No          No              No
6     Fiber optic             No           No              Yes          No         Yes             Yes
        Contract PaperlessBilling             PaymentMethod MonthlyCharges TotalCharges Churn
1 Month-to-month              Yes          Electronic check          29.85        29.85    No
2       One year               No              Mailed check          56.95      1889.50    No
3 Month-to-month              Yes              Mailed check          53.85       108.15   Yes
4       One year               No Bank transfer (automatic)          42.30      1840.75    No
5 Month-to-month              Yes          Electronic check          70.70       151.65   Yes
6 Month-to-month              Yes          Electronic check          99.65       820.50   Yes
```

  - **First six rows**


```
> # View data set dimensions and summary statistics
> dim(df)
[1] 7043   21
> head(df)
  customerID gender SeniorCitizen Partner Dependents tenure PhoneService    MultipleLines
1 7590-VHVEG Female             0     Yes         No      1           No No phone service
2 5575-GNVDE   Male             0      No         No     34          Yes               No
3 3668-QPYBK   Male             0      No         No      2          Yes               No
4 7795-CFOCW   Male             0      No         No     45           No No phone service
5 9237-HQITU Female             0      No         No      2          Yes               No
6 9305-CDSKC Female             0      No         No      8          Yes              Yes
  InternetService OnlineSecurity OnlineBackup DeviceProtection TechSupport StreamingTV StreamingMovies
1             DSL             No          Yes               No          No          No              No
2             DSL            Yes           No              Yes          No          No              No
3             DSL            Yes          Yes               No          No          No              No
4             DSL            Yes           No              Yes         Yes          No              No
5     Fiber optic             No           No               No          No          No              No
6     Fiber optic             No           No              Yes          No         Yes             Yes
        Contract PaperlessBilling             PaymentMethod MonthlyCharges TotalCharges Churn
1 Month-to-month              Yes          Electronic check          29.85        29.85    No
2       One year               No              Mailed check          56.95      1889.50    No
3 Month-to-month              Yes              Mailed check          53.85       108.15   Yes
4       One year               No Bank transfer (automatic)          42.30      1840.75    No
5 Month-to-month              Yes          Electronic check          70.70       151.65   Yes
6 Month-to-month              Yes          Electronic check          99.65       820.50   Yes
```

- **Structure of each variable**

```
> str(df)
'data.frame':	7043 obs. of  21 variables:
 $ customerID      : chr  "7590-VHVEG" "5575-GNVDE" "3668-QPYBK" "7795-CFOCW" ...
 $ gender          : chr  "Female" "Male" "Male" "Male" ...
 $ SeniorCitizen   : int  0 0 0 0 0 0 0 0 0 0 ...
 $ Partner         : chr  "Yes" "No" "No" "No" ...
 $ Dependents      : chr  "No" "No" "No" "No" ...
 $ tenure          : int  1 34 2 45 2 8 22 10 28 62 ...
 $ PhoneService    : chr  "No" "Yes" "Yes" "No" ...
 $ MultipleLines   : chr  "No phone service" "No" "No" "No phone service" ...
 $ InternetService : chr  "DSL" "DSL" "DSL" "DSL" ...
 $ OnlineSecurity  : chr  "No" "Yes" "Yes" "Yes" ...
 $ OnlineBackup    : chr  "Yes" "No" "Yes" "No" ...
 $ DeviceProtection: chr  "No" "Yes" "No" "Yes" ...
 $ TechSupport     : chr  "No" "No" "No" "Yes" ...
 $ StreamingTV     : chr  "No" "No" "No" "No" ...
 $ StreamingMovies : chr  "No" "No" "No" "No" ...
 $ Contract        : chr  "Month-to-month" "One year" "Month-to-month" "One year" ...
 $ PaperlessBilling: chr  "Yes" "No" "Yes" "No" ...
 $ PaymentMethod   : chr  "Electronic check" "Mailed check" "Mailed check" "Bank transfer (automatic)"
 $ MonthlyCharges  : num  29.9 57 53.9 42.3 70.7 ...
 $ TotalCharges    : num  29.9 1889.5 108.2 1840.8 151.7 ...
 $ Churn           : chr  "No" "No" "Yes" "No" ...
```

- **Summary**

```
> summary(df)
  customerID           gender          SeniorCitizen      Partner           Dependents
 Length:7043        Length:7043        Min.   :0.0000   Length:7043        Length:7043
 Class :character   Class :character   1st Qu.:0.0000   Class :character   Class :character
 Mode  :character   Mode  :character   Median :0.0000   Mode  :character   Mode  :character
                                       Mean   :0.1621
                                       3rd Qu.:0.0000
                                       Max.   :1.0000

     tenure      PhoneService       MultipleLines      InternetService    OnlineSecurity
 Min.   : 0.00   Length:7043        Length:7043        Length:7043        Length:7043
 1st Qu.: 9.00   Class :character   Class :character   Class :character   Class :character
 Median :29.00   Mode  :character   Mode  :character   Mode  :character   Mode  :character
 Mean   :32.37
 3rd Qu.:55.00
 Max.   :72.00

 OnlineBackup       DeviceProtection   TechSupport        StreamingTV        StreamingMovies
 Length:7043        Length:7043        Length:7043        Length:7043        Length:7043
 Class :character   Class :character   Class :character   Class :character   Class :character
 Mode  :character   Mode  :character   Mode  :character   Mode  :character   Mode  :character



   Contract         PaperlessBilling   PaymentMethod      MonthlyCharges    TotalCharges
 Length:7043        Length:7043        Length:7043        Min.   : 18.25   Min.   :  18.8
 Class :character   Class :character   Class :character   1st Qu.: 35.50   1st Qu.: 401.4
 Mode  :character   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
```

Does the data set need some cleaning or customization in order to fit with the model's requirements?

1. As we can see from the summary above, there are 11 NA's in the TotalCharges column. We can handle the missing values by removing those rows because here in our dataset the dimension is 7043 rows and 21 columns, so rows with NULL value percentage: 0.15%, which is insignificant.

2. Also, there is a customer ID column which is not useful for data analysis and model building, so we can remove that in data pre-processing or before using it for the model building. [go to: Clean data for the model selection]
3. Senior citizen column has binary values rather than "Yes" or "No". So, we can replace for the better analysis.
4. Furthermore, features like Multiple Lines, Online Security, Online Backup, Device protection, Tech support, Streaming TV and Streaming Movies have two unique values "No service" and "No". This can be replaced by only "No". This changes are made later on in the "Clean data for the model selection". We can keep it as it is for the EDA.

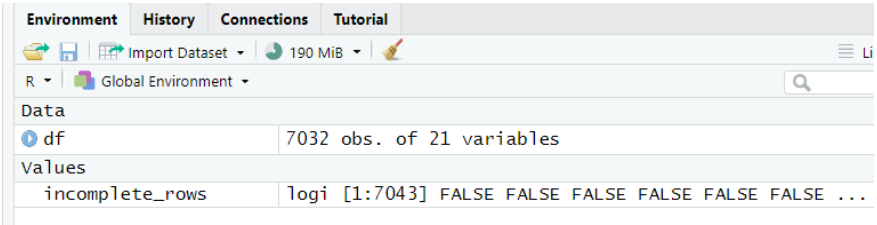
Data cleaning

➔ Remove rows with missing values

R command:

```
# ---------------------------------DATASET CLEANING-----------------
# Check for rows with NULL values in a data frame
incomplete_rows <- !complete.cases(df)

# Print the rows with NULL values
df[incomplete_rows, ]

# Neglect rows with NULL values because there are only 11 rows out of 7043
# Rows with NULL value percentage: 0.15%,
# which can be ignored
df <- df[complete.cases(df), ]
```

Environment window:



➔ Replace senior citizen column values 0 to "No" and 1 to "Yes".

R command:

```
# Change senior citizen column binary values
# Set "Yes" if senior citizen == 1 or "No" if it is 0
df$SeniorCitizen <- replace(df$SeniorCitizen, df$SeniorCitizen == 0, "No")
df$SeniorCitizen <- replace(df$SeniorCitizen, df$SeniorCitizen == 1, "Yes")
```

```
# Check for the change
View(df)
```

Data Frame tab:

Go to file/function | Addins

MeghaviVaghela_FinalProject_RCode.R × | df ×

Filter

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines
27	6467-CHFZW	Male	No	Yes	Yes	47	Yes	Yes
28	8665-UTDHZ	Male	No	Yes	Yes	1	No	No phone service
29	5248-YGIJN	Male	No	Yes	No	72	Yes	Yes
30	8773-HHUOZ	Female	No	No	Yes	17	Yes	No
31	3841-NFECX	Female	Yes	Yes	No	71	Yes	Yes
32	4929-XIHVW	Male	Yes	Yes	No	2	Yes	No
33	6827-IEAUQ	Female	No	Yes	Yes	27	Yes	No
34	7310-EGVHZ	Male	No	No	No	1	Yes	No
35	3413-BMNZE	Male	Yes	No	No	1	Yes	No
36	6234-RAAPL	Female	No	Yes	Yes	72	Yes	Yes

There might be some questions that arise to know more about dataset as following.

- **Which features are categorical and which are continuous?**

R command:

```
# Check for unique values in each feature to know about categorical and
# continuous features.
sapply(df, function(x) { if (length(unique(x)) <= 10) {return(unique(x))} else
  {return("CONTINUOUS DATA")} } )
```

Console output:

```
> # Check for unique values in each feature to know about categorical and continuous
+   features.
> sapply(df, function(x) { if (length(unique(x)) <= 10) {return(unique(x))} else
+   {return("CONTINUOUS DATA")} } )
$customerID
[1] "CONTINUOUS DATA"

$gender
[1] "Female" "Male"

$SeniorCitizen
[1] "No"  "Yes"

$Partner
[1] "Yes" "No"

$Dependents
[1] "No"  "Yes"

$tenure
[1] "CONTINUOUS DATA"

$PhoneService
[1] "No"  "Yes"

$MultipleLines
[1] "No phone service" "No"               "Yes"

$InternetService
[1] "DSL"         "Fiber optic" "No"

$OnlineSecurity
[1] "No"                  "Yes"                 "No internet service"

$OnlineBackup
[1] "Yes"                 "No"                  "No internet service"

$DeviceProtection
[1] "No"                  "Yes"                 "No internet service"

$TechSupport
[1] "No"                  "Yes"                 "No internet service"

$StreamingTV
[1] "No"                  "Yes"                 "No internet service"
```

```
$StreamingMovies
[1] "No"                  "Yes"                 "No internet service"

$Contract
[1] "Month-to-month" "One year"       "Two year"

$PaperlessBilling
[1] "Yes" "No"

$PaymentMethod
[1] "Electronic check"          "Mailed check"              "Bank transfer (automatic)"
[4] "Credit card (automatic)"

$MonthlyCharges
[1] "CONTINUOUS DATA"

$TotalCharges
[1] "CONTINUOUS DATA"

$Churn
[1] "No"  "Yes"

>
```

There are 3 continuous variables: Tenure, Monthly Charges and Total Charges. All other variables are having 2 or 3 unique values.

Univariate analysis visualization

Bar plots of specific variables

Churn Status

- **What is the overall churn rate in the dataset?**

R command:

```
# UNIVARIATE ANALYSIS
# -------------------


# CHURN PERCENT PLOT
#
#
# Count the frequency of churn and non-churn
churn_counts <- table(df$Churn)

# Calculate percentages
percent_churn <- round((churn_counts / sum(churn_counts)) * 100, 2)

# Create bar plot
x <- barplot(percent_churn, col = c("lightblue", "indianred"),
        main = "Churn Percentage", ylab = "Percentage", xlab = "Churn Status",
        names.arg = c("Not Churn", "Churn"), ylim = c(0, 100))
# Add text labels for percentages
text(x = x, y = percent_churn + 5, labels = paste0(percent_churn, "%"),
     cex = 1, col = "black")
```

Environment window:

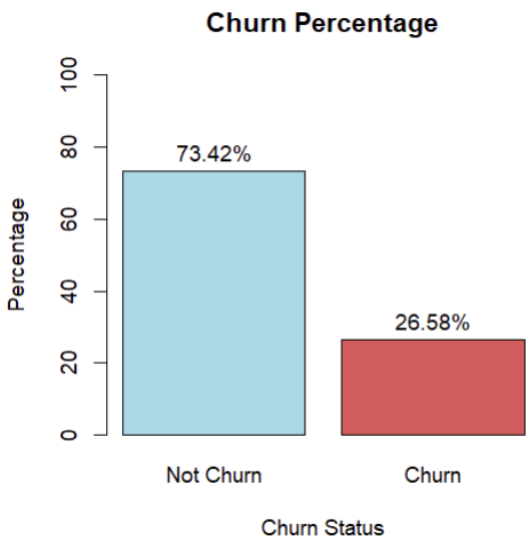
Environment | History | Connections | Tutorial

Import Dataset | 375 MiB

R | Global Environment

Data	
df	7032 obs. of 21 variables
x	num [1:2, 1] 0.7 1.9
Values	
churn_counts	'table' int [1:2(1d)] 5163 1869
incomplete_rows	logi [1:7043] FALSE FALSE FALSE FALSE FALSE FALSE
percent_churn	'table' num [1:2(1d)] 73.4 26.6

Plot:



26.58% of the customers left the platform within the last month. The recommendations after analysis should be such that the percentage churn reduces.

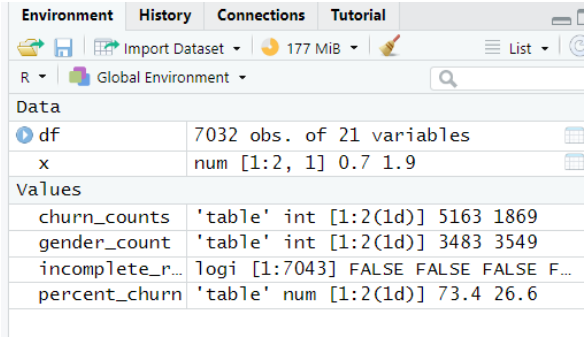
Gender ratio

- **What is the gender ratio for the given dataset?**

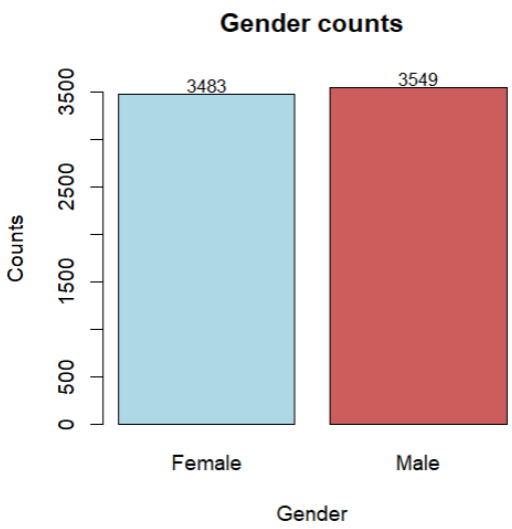
R command:

```
# GENDER RATIO
#
#
# Count the frequency of female and male present in dataset
gender_count <- table(df$gender)

# Create bar plot
x <- barplot(gender_count, col = c("lightblue", "indianred"),
             main = "Gender counts", ylab = "Counts", xlab = "Gender",
             names.arg = c("Female", "Male"), ylim = c(0, 3700))
text(x = x, y = gender_count + 100, labels = gender_count, cex = 0.9, col = "black")
```

Environment window:



Plot:



The dataset has almost similar records for both the genders male and female.

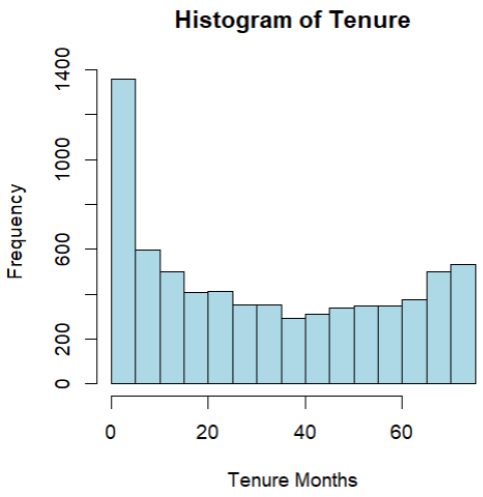
Tenure data distribution

- **What type of pattern does tenure data distribution follow?**

R command:

```
# TENURE DISTRIBUTION
#
#
#
hist(df$tenure, main = "Histogram of Tenure", xlab = "Tenure Months", ylab = "Frequency",
     col = "lightblue")
```

Plot:



As depicted in the histogram, majority of the customer has tenure 0 to 12 months.

Bivariate analysis

Bar plots for categorical variables and Box plots for continuous variables

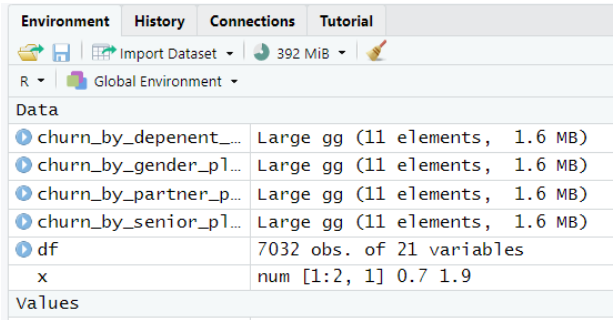
Categorical data with respect to churn

R command:

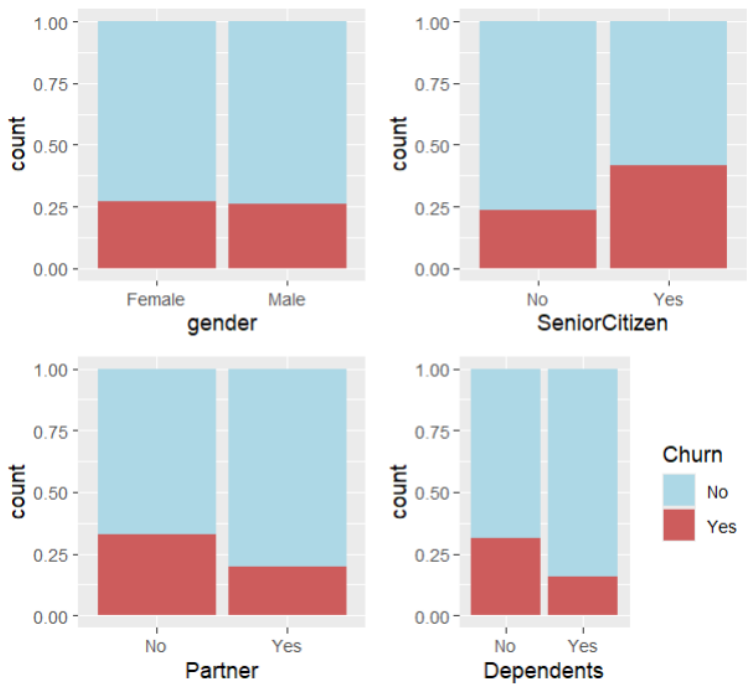
```
# BIVARIATE ANALYSIS
# -------------------

# CATEGORICAL VARIABLES VS. CHURN PLOTS
#
#
# PLOT 1: Gender vs. Churn
# Create bar plot
churn_by_gender_plot <- ggplot(df, aes(x = gender, fill = Churn)) +
                                geom_bar(position = 'fill') +
                                scale_fill_manual(values = c("lightblue", "indianred")) +
                                theme(legend.position = "none")


# PLOT 2: Senior citizen vs. Churn
# Create bar plot
churn_by_senior_plot <- ggplot(df, aes(x = SeniorCitizen, fill = Churn)) +
                                geom_bar(position = 'fill') +
                                scale_fill_manual(values = c("lightblue", "indianred")) +
                                theme(legend.position = "none")


# PLOT 3: Partner vs. Churn
# Create bar plot
churn_by_partner_plot <- ggplot(df, aes(x = Partner, fill = Churn)) +
                                geom_bar(position = 'fill') +
                                scale_fill_manual(values = c("lightblue", "indianred")) +
                                theme(legend.position = "none")


# PLOT 4: Dependent vs. Churn
# Create bar plot
churn_by_depenent_plot <- ggplot(df, aes(x = Dependents, fill = Churn)) +
                                  geom_bar(position = 'fill') +
                                  scale_fill_manual(values = c("lightblue", "indianred"))
```

```
# Create Multiplot grid for bivariate analysis using bar plots
plot_grid(churn_by_gender_plot, churn_by_senior_plot, churn_by_partner_plot,
          churn_by_depenent_plot, ncol = 2)
```

Environment window:



Plot:



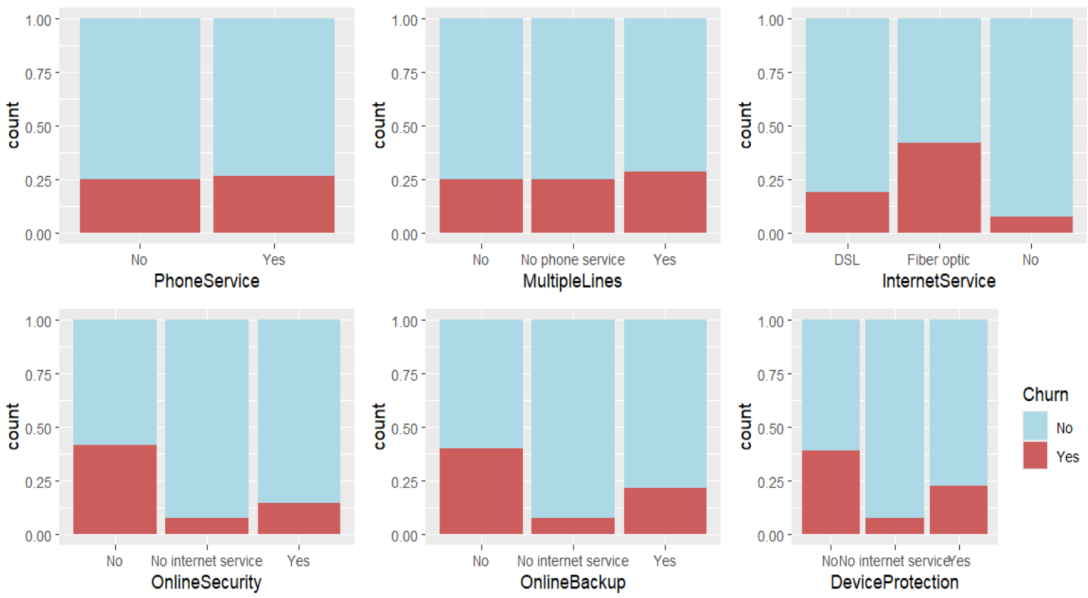
- Churn rate is almost equal in the case of Male and Female.
- Customers having no partner and no dependents has left the platform more as compared to those who have partner and dependents.
- Senior citizen is more likely to leave.

R command:

```r
# PLOT 5: Phone Service vs. Churn
# Create bar plot
churn_by_PhoneService_plot <- ggplot(df, aes(x = PhoneService, fill = Churn)) +
                                    geom_bar(position = 'fill') +
                                    scale_fill_manual(values = c("lightblue", "indianred")) +
                                    theme(legend.position = "none")


# PLOT 6: Multiple lines vs. Churn
# Create bar plot
churn_by_MultipleLines_plot <- ggplot(df, aes(x = MultipleLines, fill = Churn)) +
                                    geom_bar(position = 'fill') +
                                    scale_fill_manual(values = c("lightblue", "indianred")) +
                                    theme(legend.position = "none")



# PLOT 7: Internet service vs. Churn
# Create bar plot
churn_by_InternetService_plot <- ggplot(df, aes(x = InternetService, fill = Churn)) +
                                      geom_bar(position = 'fill') +
                                      scale_fill_manual(values = c("lightblue", "indianred")) +
                                      theme(legend.position = "none")
```

```r
# PLOT 8: Online security vs. Churn
# Create bar plot
churn_by_OnlineSecurity_plot <- ggplot(df, aes(x = OnlineSecurity, fill = Churn)) +
                                      geom_bar(position = 'fill') +
                                      scale_fill_manual(values = c("lightblue", "indianred")) +
                                      theme(legend.position = "none")


# PLOT 9: Online backup vs. Churn
# Create bar plot
churn_by_OnlineBackup_plot <- ggplot(df, aes(x = OnlineBackup, fill = Churn)) +
                                    geom_bar(position = 'fill') +
                                    scale_fill_manual(values = c("lightblue", "indianred")) +
                                    theme(legend.position = "none")


# PLOT 10: Device protection vs. Churn
# Create bar plot
churn_by_DeviceProtection_plot <- ggplot(df, aes(x = DeviceProtection, fill = Churn)) +
                                        geom_bar(position = 'fill') +
                                        scale_fill_manual(values = c("lightblue", "indianred"))
```

Plot:



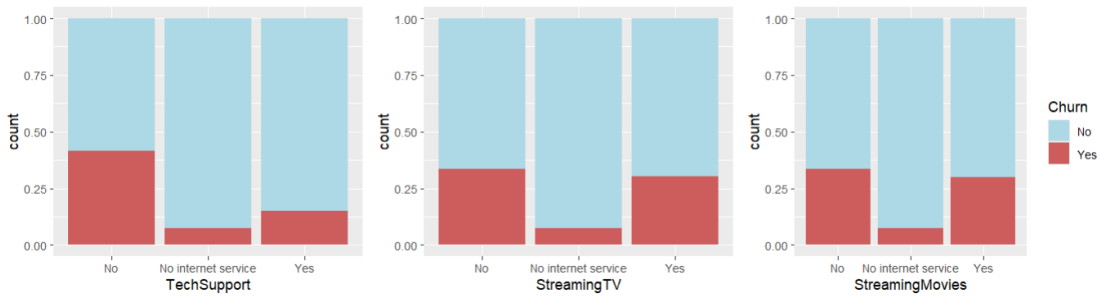
- Customers with Fiber Optic based Internet Service has high churn rate.

- Churn rate is almost same for customers with or without Phone Service.
- It is observed that the customers who do not have services like Online Security, Online Backup and Device Protection has left the platform last month much more than those who have these services.

R command:

```
# PLOT 11: Tech support vs. Churn
# Create bar plot
churn_by_TechSupport_plot <- ggplot(df, aes(x = TechSupport, fill = Churn)) +
                                        geom_bar(position = 'fill') +
                                        scale_fill_manual(values = c("lightblue", "indianred")) +
                                        theme(legend.position = "none")


# PLOT 12: Streaming TV vs. Churn
# Create bar plot
churn_by_StreamingTV_plot <- ggplot(df, aes(x = StreamingTV, fill = Churn)) +
                                        geom_bar(position = 'fill') +
                                        scale_fill_manual(values = c("lightblue", "indianred")) +
                                        theme(legend.position = "none")


# PLOT 13: Streaming Movies vs. Churn
# Create bar plot
churn_by_StreamingMovies_plot <- ggplot(df, aes(x = StreamingMovies, fill = Churn)) +
                                          geom_bar(position = 'fill') +
                                          scale_fill_manual(values = c("lightblue", "indianred"))
```

Plot:

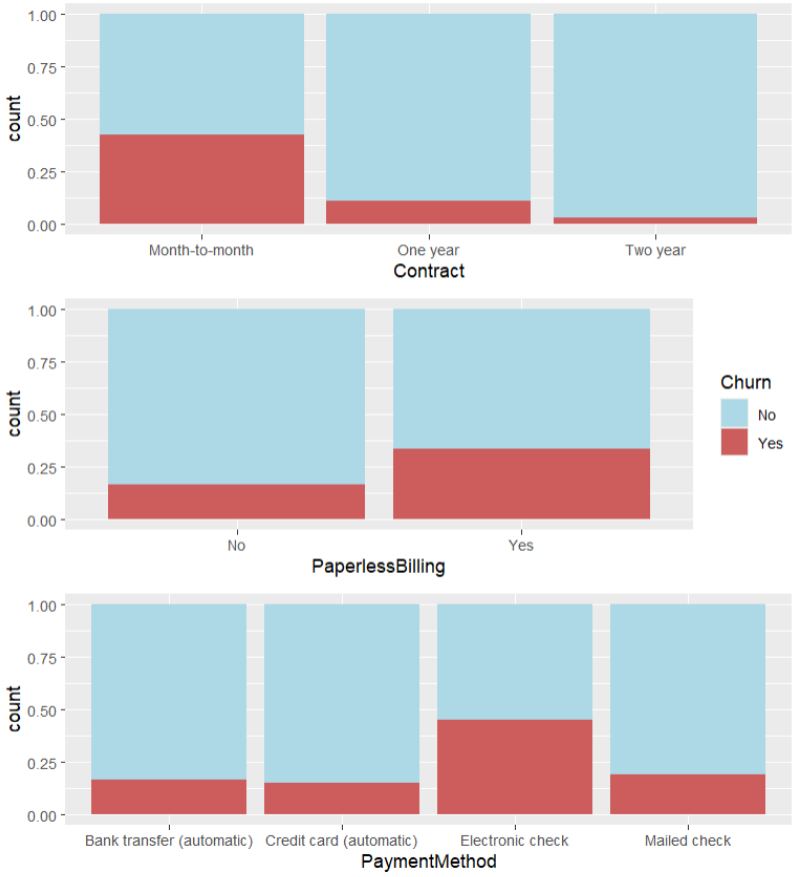


- Churn rate is almost same for the customers with or without Streaming services like Streaming TV and Streaming Movies.
- The customers who do not have Tech Support Service has much more churn rate than those who have it.

R command:

```r
# PLOT 14: Contract vs. Churn
# Create bar plot
churn_by_Contract_plot <- ggplot(df, aes(x = Contract, fill = Churn)) +
                                  geom_bar(position = 'fill') +
                                  scale_fill_manual(values = c("lightblue", "indianred")) +
                                  theme(legend.position = "none")



# PLOT 15: Paperless billing vs. Churn
# Create bar plot
churn_by_PaperlessBilling_plot <- ggplot(df, aes(x = PaperlessBilling, fill = Churn)) +
                                        geom_bar(position = 'fill') +
                                        scale_fill_manual(values = c("lightblue", "indianred"))



# PLOT 16: Payment method vs. Churn
# Create bar plot
churn_by_PaymentMethod_plot <- ggplot(df, aes(x = PaymentMethod, fill = Churn)) +
                                      geom_bar(position = 'fill') +
                                      scale_fill_manual(values = c("lightblue", "indianred")) +
                                      theme(legend.position = "none")
```

Plot:



- Month-to-month contract based customers show much more churn rate than other contracts.
- The customers who give Electronic check and have the Paperless Billing system for the payment has much more churn rate than others.
- From above visualization we can say that below are the most significant features of customer to decide if customer will churn or not:
    - Senior Citizen
    - No partner

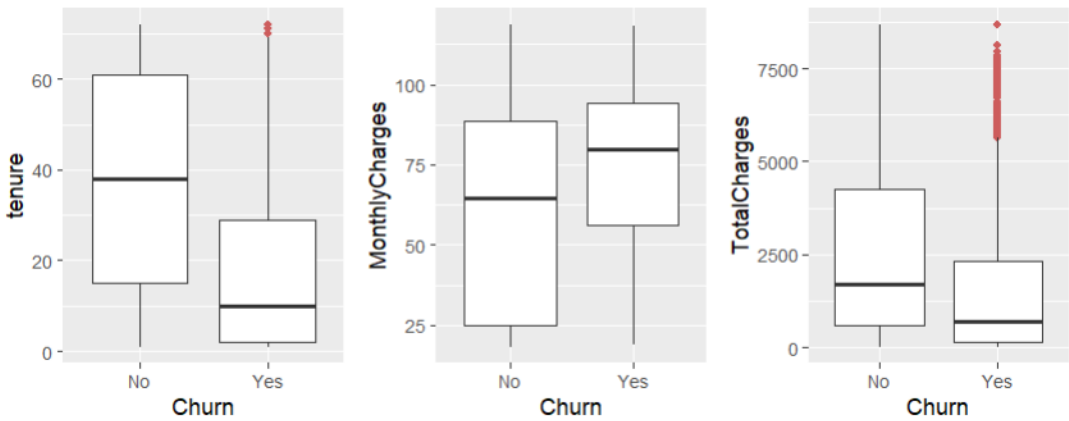
- No dependents
- Fiber optic based Internet Service
- Has Internet Service but no Online Security, no Online Backup, no Device Protection and no Tech Support service.
- Monthly contract
- Payment through Electronic Check

Continuous data with respect to churn

R command:

```
# CONTINUOUS VARIABLES VS. CHURN BOX PLOTS
#
#
# PLOT 1: Tenure vs. Churn
#
#
# Create box plot
churn_by_Tenure_boxplot <- ggplot(data = df, aes(x = Churn, y = tenure)) +
                                geom_boxplot(outlier.color = "indianred")

# PLOT 2: Monthly charges vs. Churn
#
#
# Create box plot
churn_by_MonthlyCharges_boxplot <- ggplot(data = df, aes(x = Churn, y = MonthlyCharges)) +
                                      geom_boxplot(outlier.color = "indianred")


# PLOT 3: Total charges vs. Churn
#
#
# Create box plot
churn_by_TotalCharges_boxplot <- ggplot(data = df, aes(x = Churn, y = TotalCharges)) +
                                    geom_boxplot(outlier.color = "indianred")


# Create Multiplot grid for bivariate analysis using box plots
plot_grid(churn_by_Tenure_boxplot, churn_by_MonthlyCharges_boxplot,
          churn_by_TotalCharges_boxplot, ncol = 3)
```

Plot:



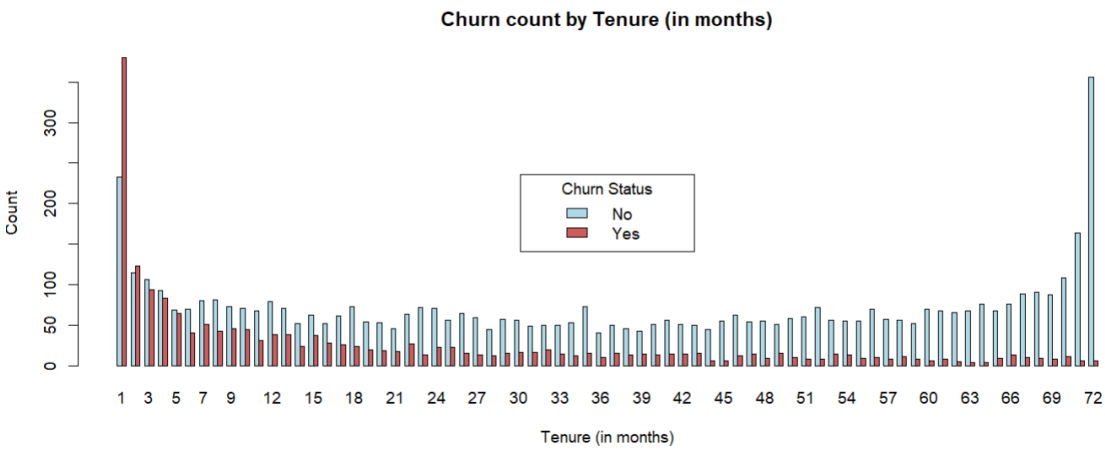
- The median tenure of customers who have left in last month is around 12 months.

- The customers who have churned has high monthly charges with median value more than 75.
- The median Total charges is low for the customers who churned.

R command:

```
# CONTINUOUS VARIABLES VS. CHURN PLOTS (Another way to visualize)
#
#
# PLOT 1: Tenure vs. Churn
#
# Count the frequency of churn by Payment method
churn_counts <- table(df$Churn, df$tenure)

# Create bar plot
churn_by_Tenure_plot <- barplot(churn_counts,
                                beside = TRUE,
                                col = c("lightblue", "indianred"),
                                main = "Churn count by Tenure (in months)",
                                xlab = "Tenure (in months)", ylab = "Count",
                                legend = rownames(churn_counts),
                                args.legend = list(title = "Churn Status", x = "center", y = NULL))
```

Plot:

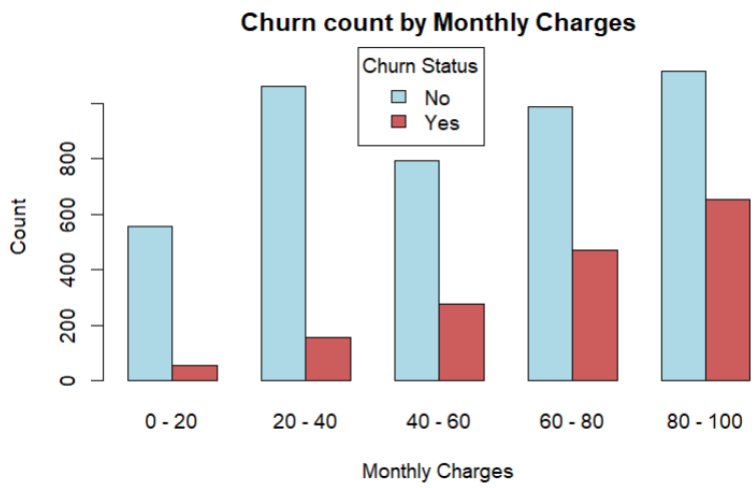


- As the tenure month increases, the churn rate shows positive skewness (right-skewed distribution) and the retention rate shows Bi-modal/U-shaped distribution.

R command:

```
# PLOT 2: Monthly charges vs. Churn
#
#
# Create bins for grouping monthly charges
bins <- cut(df$MonthlyCharges,
            breaks = seq(0, ceiling(max(df$MonthlyCharges)), by = 20),
            right = FALSE,
            labels = paste0(seq(0, ceiling(max(df$MonthlyCharges)) - 20, by = 20), " - ",
                            seq(20, ceiling(max(df$MonthlyCharges)), by = 20)))

# Count the frequency of churn by Payment method
churn_counts <- table(df$Churn, bins)


# Create bar plot
churn_by_MonthlyCharges_plot <- barplot(churn_counts, beside = TRUE, col = c("lightblue", "indianred"),
                                        main = "Churn count by Monthly Charges",
                                        xlab = "Monthly Charges", ylab = "Count",
                                        legend = rownames(churn_counts),
                                        args.legend = list(title = "Churn Status", x = 9, y = 1200))
```

Plot:

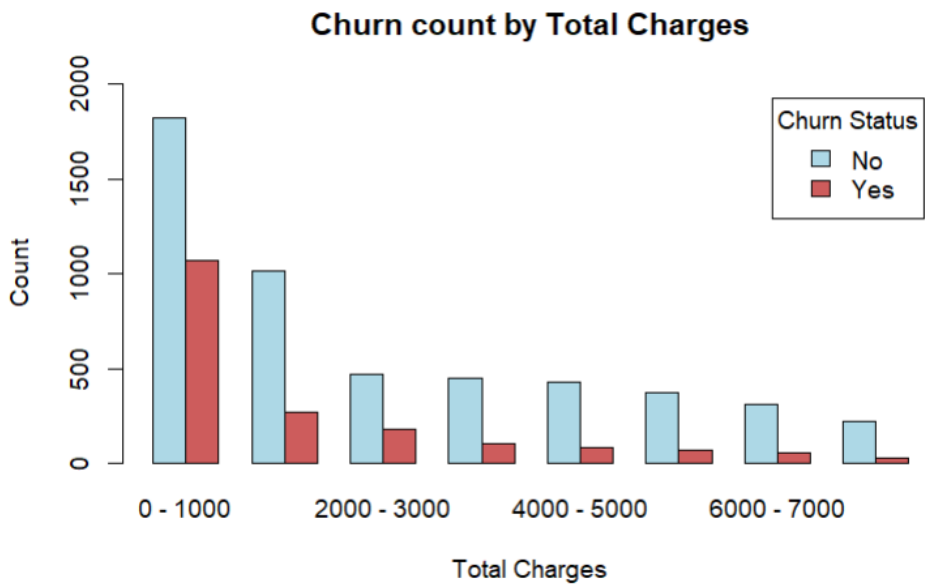


- As the Monthly charges increases, the churn rate shows negative skewness (left-skewed distribution).

R command:

```
# PLOT 3: Total charges vs. Churn
#
#
# Create bins for grouping monthly charges
bins <- cut(df$TotalCharges,
            breaks = seq(0, ceiling(max(df$TotalCharges)), by = 1000),
            right = FALSE,
            labels = paste0(seq(0, ceiling(max(df$TotalCharges)) - 1000, by = 1000), " - ",
                            seq(1000, ceiling(max(df$TotalCharges)), by = 1000)))


# Count the frequency of churn by Payment method
churn_counts <- table(df$Churn, bins)

# Create bar plot
churn_by_TotalCharges_plot <- barplot(churn_counts, beside = TRUE, col = c("lightblue", "indianred"),
                                      main = "Churn count by Total Charges",
                                      xlab = "Total Charges", ylab = "Count",
                                      ylim = c(0, 2000),
                                      legend = rownames(churn_counts),
                                    args.legend = list(title = "Churn Status"))
```

Plot:



- As the Total charges increases, the churn rate shows positive skewness (right-skewed distribution).

Multivariate analysis

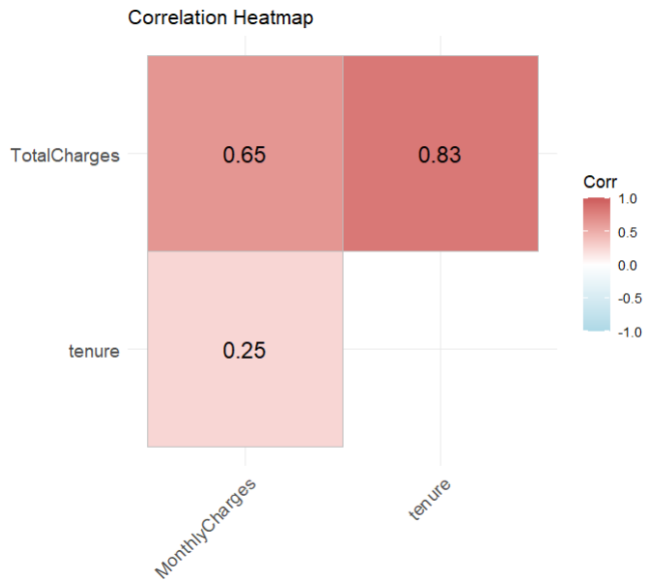
Correlation matrix for continuous variables

Correlation Heatmap

R command:

```
# MULTIVARIATE ANALYSIS
#
#
# PLOT: correlation heatmap for 3 continuous variables
correlation_matrix <- cor(df[c("tenure", "MonthlyCharges", "TotalCharges")])

# Create the correlation heatmap
ggcorrplot(correlation_matrix,
           type = "upper", # Show only upper triangle of the correlation matrix
           hc.order = TRUE,
           lab = TRUE, # Correlation value show
           lab_size = 5, # Value size
           method = "square",
           colors = c("lightblue", "white", "indianred"),
           title = "Correlation Heatmap"
)
```

Heatmap:



- Total Charges has positive correlation with Monthly Charges and tenure.

Model's baseline

Type of Data Mining Algorithm used

- The data mining algorithm used to build the model is XGBoost (Extreme Gradient Boosting).
- XGBoost is a powerful machine learning algorithm known for its efficiency and effectiveness in predictive modeling tasks.
- It belongs to the family of gradient boosting algorithms, which iteratively combine weak learners (typically decision trees) to create a strong predictive model.

Why I preferred this algorithm?

- XGBoost employs an ensemble learning technique that sequentially builds a series of decision trees, with each tree aiming to correct the errors made by the previous ones.
- It optimizes model performance by minimizing a specified loss function, such as mean squared error or log loss, and incorporating regularization techniques to prevent overfitting.

Apart from above facts, below is the accuracy results based decision to select this algorithm for the used dataset.

Clean data for the model selection

- Replace the value “No phone service” or “No Internet Service” with “No”.

R command:

```
# Change the names of "No service" to "No" for the ease of the analysis
df$MultipleLines <- replace(df$MultipleLines, df$MultipleLines == "No phone service", "No")
df$OnlineSecurity <- replace(df$OnlineSecurity, df$OnlineSecurity == "No internet service", "No")
df$OnlineBackup <- replace(df$OnlineBackup, df$OnlineBackup == "No internet service", "No")
df$DeviceProtection <- replace(df$DeviceProtection, df$DeviceProtection == "No internet service", "No")
df$TechSupport <- replace(df$TechSupport, df$TechSupport == "No internet service", "No")
df$StreamingTV <- replace(df$StreamingTV, df$StreamingTV == "No internet service", "No")
df$StreamingMovies <- replace(df$StreamingMovies, df$StreamingMovies == "No internet service", "No")
View(df)
```

Data Frame tab:

MeghaviVaghela_FinalProject_RCode.R × df × df_train ×

Filter

	customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService	OnlineSecurity
1	7590-VHVEG	Female	No	Yes	No	1	No	No	DSL	No
2	5575-GNVDE	Male	No	No	No	34	Yes	No	DSL	Yes
3	3668-QPYBK	Male	No	No	No	2	Yes	No	DSL	Yes
4	7795-CFOCW	Male	No	No	No	45	No	No	DSL	Yes
5	9237-HQITU	Female	No	No	No	2	Yes	No	Fiber optic	No
6	9305-CDSKC	Female	No	No	No	8	Yes	Yes	Fiber optic	No
7	1452-KIOVK	Male	No	No	Yes	22	Yes	Yes	Fiber optic	No
8	6713-OKOMC	Female	No	No	No	10	No	No	DSL	Yes
9	7892-POOKP	Female	No	Yes	No	28	Yes	Yes	Fiber optic	No
10	6388-TABGU	Male	No	No	Yes	62	Yes	No	DSL	Yes
11	9763-GRSKD	Male	No	Yes	Yes	13	Yes	No	DSL	Yes
12	7469-LKBCI	Male	No	No	No	16	Yes	No	No	No
13	8091-TTVAX	Male	No	Yes	No	58	Yes	Yes	Fiber optic	No
14	0280-XJGEX	Male	No	No	No	49	Yes	Yes	Fiber optic	No

- Remove customer ID column

R command:

```
# ---------------------------------- Select best model---------------------
# -----------------------------------(for given dataset)------------------

# train and test data set
set.seed(123)
df_train <- df[, -c(1)]
View(df_train)
```

Data Frame tab:

MeghaviVaghela_FinalProject_RCode.R × df_train × df ×

Filter

	gender	SeniorCitizen	Partner	Dependents	tenure	Ph
1	Female	No	Yes	No	1	No
2	Male	No	No	No	34	Ye
3	Male	No	No	No	2	Ye
4	Male	No	No	No	45	No

Data split

R command:

```
# split the data with 70% training set and 30% test set
train_ind_all <- createDataPartition(df_train$Churn, p = 0.7, list = FALSE)

# Create the train dataset
X_train_all <- df_train[train_ind_all, ]

# Create the test dataset
X_test_all <- df_train[-train_ind_all, ]
```

Environment window:

X_test_all	2108 obs. of 20 variables
X_train_all	4924 obs. of 20 variables
Values	
bins	Factor w/ 8 levels "0 - 1000" "1000 - 2000" : 1 2

Cross-validation

R command:

```
# 3-fold cross validation
modelCtrl <- trainControl(method = "cv", number = 3, verboseIter = FALSE)
```

Environment window:

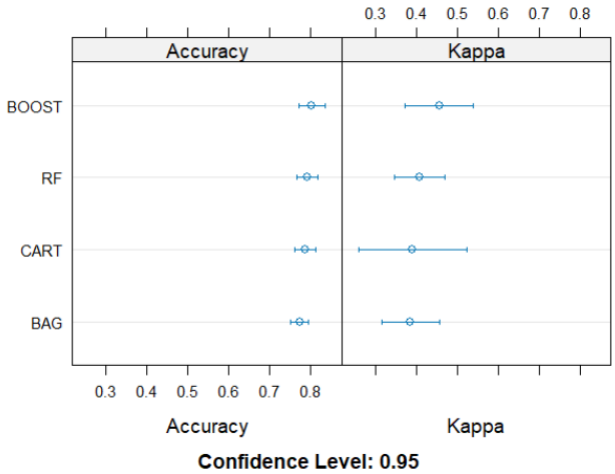
modelCtrl	List of 27
train_ind_all	int [1:4924, 1] 1 2 3 5 6 8 11 14 15 16 ...

Apply CART, Tree Bag, XGBoost, and Random Forest algorithms

R command:

```
# CART
set.seed(123)
cart.model <- train(Churn ~., method = "rpart", data = X_train_all,
                    parms = list(split = "gini"), trControl = modelCtrl)

# Tree Bag algorithm
set.seed(123)
bag.model <- train(Churn ~., method = "treebag", data = X_train_all,
                   trControl = modelCtrl)

# Boosting algorithm
set.seed(123)
xgb.model <- train(Churn ~., method = "xgbTree", data = X_train_all,
                   trControl = modelCtrl)

# Random Forest
set.seed(123)
rf.model <- train(Churn ~., method = "rf", data = X_train_all,
                  trControl = modelCtrl)

results <- resamples(list(CART=cart.model, BAG=bag.model,
                          BOOST=xgb.model, RF=rf.model))
dotplot(results)
```

Plot:



Summary

R command:

```
summary(results)
```

Console output:

```
> summary(results)

Call:
summary.resamples(object = results)

Models: CART, BAG, BOOST, RF
Number of resamples: 3

Accuracy
           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
CART  0.7758831 0.7846509 0.7934186 0.7877766 0.7937233 0.7940280    0
BAG   0.7667479 0.7691143 0.7714808 0.7739657 0.7775746 0.7836685    0
BOOST 0.7904994 0.7968341 0.8031688 0.8034145 0.8098720 0.8165753    0
RF    0.7813642 0.7867820 0.7921999 0.7916349 0.7967703 0.8013406    0

Kappa
           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
CART  0.3328402 0.3688197 0.4047993 0.3917332 0.4211797 0.4375601    0
BAG   0.3564000 0.3730952 0.3897904 0.3862230 0.4011346 0.4124787    0
BOOST 0.4232961 0.4392979 0.4552997 0.4561925 0.4726407 0.4899817    0
RF    0.3909861 0.3941534 0.3973208 0.4085509 0.4173333 0.4373458    0

>
```

- A **high kappa value** is often observed as a desirable metric for model algorithm selection because it indicates strong agreement between the predicted and actual classifications made by a classification model.
- Kappa (also known as Cohen's kappa) is a statistic that measures inter-rater agreement or the agreement between two raters when assigning categorical labels.
- A **high accuracy** is also significant metric for model algorithm selection.
- Among all above algorithms, Boost algorithm has highest mean accuracy 0.80 and mean Kappa 0.45.
- **Hence, the 'XGBoost' algorithm is selected.**

The pros and cons of chosen algorithm

Pros:

1. **High Accuracy:** XGBoost is known for its high predictive accuracy and performance. It can effectively capture complex relationships between features and target variables, leading to more accurate predictions of customer churn.
2. **Handles Non-linearity:** XGBoost is capable of capturing non-linear relationships and interactions between predictor variables, making it suitable for modeling complex datasets commonly encountered in customer churn prediction tasks.
3. **Regularization:** XGBoost incorporates regularization techniques such as L1 and L2 regularization, which help prevent overfitting and improve the generalization ability of the model. This is particularly beneficial in customer churn prediction, where overfitting can lead to inaccurate predictions.
4. **Feature Importance:** XGBoost provides insights into feature importance, allowing analysts to identify the most influential variables in predicting customer churn. This helps in understanding the underlying drivers of churn and informing targeted retention strategies.
5. **Scalability:** XGBoost is highly scalable and efficient, making it suitable for large datasets with a high volume of features. It can handle millions of observations and thousands of features efficiently, enabling fast model training and prediction.

Cons:

1. **Parameter Tuning:** XGBoost has several hyper parameters that need to be tuned to optimize model performance. Finding the optimal combination of hyper parameters can be time-consuming and computationally intensive, requiring careful experimentation and validation.
2. **Black Box Model:** Like other ensemble methods, XGBoost is considered a black box model, meaning it lacks interpretability compared to simpler models such as logistic regression. While it provides accurate predictions, understanding the underlying mechanisms driving those predictions may be challenging.
3. **Sensitivity to Outliers:** XGBoost can be sensitive to outliers in the data, potentially leading to suboptimal model performance if outliers are not properly handled during pre-processing. Outliers may disproportionately influence the splitting decisions made by the algorithm, affecting the overall model output.
4. **Resource Intensive:** Training an XGBoost model with large datasets and complex features can require significant computational resources, including memory and processing power. This may limit its practicality for deployment in resource-constrained environments or real-time applications.
5. **Data Pre-processing Requirements:** XGBoost performs best when the input data is pre-processed and cleaned appropriately. Data pre-processing tasks such as handling missing values, encoding categorical variables, and scaling features are essential for maximizing the performance of the model.

Model building

XGBoost model without hyper parameter tuning

- ➔ Create a model as mentioned above in [Apply CART, Tree Bag, XGBoost, and Random Forest algorithms]
- ➔ Create Prediction on test data with removed churn target variable and display unique values of the prediction.

R command:

```
# Predict on test data with removed churn target variable
xgb.model_pred <- predict(xgb.model, newdata = X_test_all[, -c(20)])
unique(xgb.model_pred)
```

Console output:

```
> # Predict on test data with removed churn target variable
> xgb.model_pred <- predict(xgb.model, newdata = X_test_all[, -c(20)])
> unique(xgb.model_pred)
[1] No  Yes
Levels: No Yes
>
```

Evaluate the model1

- ➔ Evaluate the "xgbTree" model above.

R command:

```
# Area Under the ROC Curve (AUC)
auc2 <- pROC::auc(roc(X_test_all$Churn, as.numeric(xgb.model_pred)))
auc2

# Confusion matrix and other evaluations
conf_mat2 <- table(xgb.model_pred, X_test_all$Churn)

accuracy2 <- sum(diag(conf_mat2)) / sum(conf_mat2)
precision2 <- conf_mat2[2, 2] / sum(conf_mat2[, 2])
recall2 <- conf_mat2[2, 2] / sum(conf_mat2[2, ])
f1_score2 <- 2 * precision1 * recall1 / (precision1 + recall1)

# Print the metrics
print(conf_mat2)
print(paste("Accuracy:", accuracy2))
print(paste("Precision:", precision2))
print(paste("Recall:", recall2))
print(paste("F1-score:", f1_score2))
```

Console output:

```
> # Area Under the ROC Curve (AUC)
> auc2 <- pROC::auc(roc(X_test_all$Churn, as.numeric(xgb.model_pred)))
Setting levels: control = No, case = Yes
Setting direction: controls < cases
> auc2
Area under the curve: 0.7235
> # Confusion matrix and other evaluations
> conf_mat2 <- table(xgb.model_pred, X_test_all$Churn) # best_cutoff value here is 0.6
> accuracy2 <- sum(diag(conf_mat2)) / sum(conf_mat2)
> precision2 <- conf_mat2[2, 2] / sum(conf_mat2[, 2])
> recall2 <- conf_mat2[2, 2] / sum(conf_mat2[2, ])
> f1_score2 <- 2 * precision1 * recall1 / (precision1 + recall1)
> # Print the metrics
> print(conf_mat2)

xgb.model_pred  No Yes
           No  940 173
           Yes  92 200
> print(paste("Accuracy:", accuracy2))
[1] "Accuracy: 0.811387900355872"
> print(paste("Precision:", precision2))
[1] "Precision: 0.536193029490617"
> print(paste("Recall:", recall2))
[1] "Recall: 0.684931506849315"
> print(paste("F1-score:", f1_score2))
[1] "F1-score: 0.835696990660671"
>
```

"Setting direction: controls < cases": This part indicates the direction of comparison for the ROC curve. It tells that the ROC curve is being plotted so that higher values of your model's prediction are associated with the "Yes" or "case" category.

"Setting levels: control = No, case = Yes": This tells how the levels of outcome variable are being interpreted. In this case, "No" likely refers to one level of your outcome variable (indicating non-churn), and "Yes" refers to the other level (indicating churn).

XGBoost model with hyper parameter tuning

Below is the model created by 'xgb.train' command. 'xgb.train' is an advanced interface for training an 'xgboost' model. The 'xgboost' function is a simpler wrapper for 'xgb.train'. [reference: R Documentation > xgb.train {xgboost} > Description, https://www.rdocumentation.org/packages/xgboost/versions/1.7.7.1/topics/xgb.train]

In this way of XGBoost model building, the 'matrix' format is required for data.

Note: When you convert your data into MATRIX format in R, it's optimized for use with the XGBoost algorithm, allowing for faster computation and reduced memory usage, especially with large datasets.

Pre-process the dataset

- ➔ Select all continuous variables
- ➔ Convert into numeric
- ➔ Apply scaling on data frame
- ➔ Select all categorical variables
- ➔ Apply ONE-HOT encoding which converts character values like 'yes' or 'no' into numbers
- ➔ Feature engineering is done here with the creation of the dummy variables.
- ➔ Bind data frames converted from matrix and create final data frame.
- ➔ Display the final data frame.

R command:

```
# STEP 1: convert continuous variables into numeric using as.numeric()
#
# select all continuous variables
continuous_df <- df[c("tenure", "MonthlyCharges", "TotalCharges")]

# convert into numeric
continuous_df_numeric <- sapply(continuous_df, as.numeric)

# apply scaling on data frame
continuous_df_scaled <- scale(continuous_df_numeric)


# STEP 2: convert categorical variables into matrix with all numeric values
#
# select all categorical variables
categorical_df <- df[, !(names(df) %in% c("tenure", "MonthlyCharges", "TotalCharges", "customerID"))]

# Apply ONE-HOT encoding which converts character values like 'yes' or 'no' into numbers
encoded_matrix <- model.matrix(~.-1, data = data.frame(categorical_df))



# Final data for model building
# bind data frames converted from matrix
final_df <- cbind(data.frame(continuous_df_scaled), data.frame(encoded_matrix))

# Display final_df
head(final_df)
```

Console output:

```
> # Display final_df
> head(final_df)
       tenure MonthlyCharges TotalCharges genderFemale genderMale SeniorCitizenYes PartnerYes DependentsYes
1 -1.28015700     -1.1616113   -0.9941234            1          0                0         1             0
2  0.06429811     -0.2608594   -0.1737275            0          1                0         0             0
3 -1.23941594     -0.3638974   -0.9595809            0          1                0         0             0
4  0.51244982     -0.7477972   -0.1952338            0          1                0         0             0
5 -1.23941594      0.1961642   -0.9403906            1          0                0         0             0
6 -0.99496955      1.1584066   -0.6453233            1          0                0         0             0
  PhoneServiceYes MultipleLinesYes InternetServiceFiber.optic InternetServiceNo OnlineSecurityYes OnlineBackupYes
1               0                0                         0                 0                 0               1
2               1                0                         0                 0                 1               0
3               1                0                         0                 0                 1               1
4               0                0                         0                 0                 1               0
5               1                0                         1                 0                 0               0
6               1                1                         1                 0                 0               0
  DeviceProtectionYes TechSupportYes StreamingTVYes StreamingMoviesYes ContractOne.year ContractTwo.year
1                   0             0             0                 0               0               0
2                   1             0             0                 0               1               0
3                   0             0             0                 0               0               0
4                   1             1             0                 0               1               0
5                   0             0             0                 0               0               0
6                   1             0             1                 1               0               0
  PaperlessBillingYes PaymentMethodCredit.card..automatic. PaymentMethodElectronic.check
1                   1                                   0                            1
2                   0                                   0                            0
3                   1                                   0                            0
4                   0                                   0                            0
5                   1                                   0                            1
6                   1                                   0                            1
  PaymentMethodMailed.check ChurnYes
1                         0        0
2                         1        0
3                         1        1
4                         0        0
5                         0        1
6                         0        1
>
```

Split the data

- ➔ Set the seed for same partition of set.
- ➔ Create train and test dataset with 70% train and 30% test data.
- ➔ Display the dimension of both.
- ➔ Separate target variables for predictors.

R command:

```
# set seed
set.seed(123)

# split the data with 70% training set and 30% test set
train_ind <- createDataPartition(final_df$ChurnYes, p = 0.7, list = FALSE)

# Create the train dataset
X_train <- final_df[train_ind, ]

# Create the test dataset
X_test <- final_df[-train_ind, ]

# Check the dim
dim(X_train)
dim(X_test)


# Separate target variable for predictors
y_train <- X_train$ChurnYes
X_train_pred <- subset(X_train, select = -c(ChurnYes))
y_test <- X_test$ChurnYes
X_test_pred <- subset(X_test, select = -c(ChurnYes))
```

Console output:

```
> # Check the dim
> dim(X_train)
[1] 4923   25
> dim(X_test)
[1] 2109   25
```

After feature engineering there are now 25 variables to work with, that too in matrix format which adds in ease of the computations.

Build model2

- ➔ Convert data sets into DMatrix format
- ➔ Calculate class imbalance ratio to handle it in the 'params'.
- ➔ Train the model
- ➔ Predict on test data
- ➔ Convert the predictions that can be any decimal value between 0 to 1, into 0 or 1 based on the cut-off value 0.5.
- ➔ Display the unique values in the predictions.

R command:

```
# Convert data to DMatrix format
dtrain <- xgb.DMatrix(data = as.matrix(X_train_pred), label = y_train)
dtest <- xgb.DMatrix(data = as.matrix(X_test_pred), label = y_test)

# Calculate class imbalance ratio
imbalance_ratio <- sum(y_train == 0) / sum(y_train == 1)


# Train XGBoost model
params <- list(
  objective = "binary:logistic",
  eval_metric = "logloss",
  scale_pos_weight = imbalance_ratio,
  eta = 0.1,
  max_depth = 6,
  min_child_weight = 1,
  subsample = 0.8,
  colsample_bytree = 0.8,
  reg_lambda = 1,
  reg_alpha = 0
)

xgb_model <- xgb.train(params = params, data = dtrain, nrounds = 100)

# Predict on test data
xgb_pred <- predict(xgb_model, newdata = dtest)


# Predict on test data with cutoff = 0.5
xgb_model_pred <- ifelse(xgb_pred > 0.5, 1, 0)
unique(xgb_model_pred)
```

Console output:

```
> unique(xgb_model_pred)
[1] 1 0
>
```

In above prediction, ‘1’ means ‘Churn = Yes’ and ‘0’ means ‘Churn = No’.

Evaluate the model2

R command:

```r
# Area Under the ROC Curve (AUC)
auc1 <- pROC::auc(roc(X_test$ChurnYes, xgb_model_pred))
auc1

# Convert predicted values to factor
xgb_model_pred_factor <- factor(xgb_model_pred, levels = c(0, 1))

# Convert actual values to factor
actual_val <- factor(X_test$ChurnYes, levels = c(0, 1))

# Calculate confusion matrix and evaluation metrics
conf_mat1 <- confusionMatrix(data = xgb_model_pred_factor, reference = actual_val)

accuracy1 <- conf_mat1$overall["Accuracy"]
precision1 <- conf_mat1$byClass["Precision"]
recall1 <- conf_mat1$byClass["Recall"]
f1_score1 <- conf_mat1$byClass["F1"]

# Print the metrics
print(conf_mat1)
print(paste("Accuracy:", accuracy1))
print(paste("Precision:", precision1))
print(paste("Recall:", recall1))
print(paste("F1-score:", f1_score1))
```

Console output:

```
> # Area Under the ROC Curve (AUC)
> auc1 <- pROC::auc(roc(X_test$ChurnYes, xgb_model_pred))
Setting levels: control = 0, case = 1
Setting direction: controls < cases
> auc1
Area under the curve: 0.7703
> # Convert predicted values to factor
> xgb_model_pred_factor <- factor(xgb_model_pred, levels = c(0, 1))
> # Convert actual values to factor
> actual_val <- factor(X_test$ChurnYes, levels = c(0, 1))
> # Calculate confusion matrix and evaluation metrics
> conf_mat1 <- confusionMatrix(data = xgb_model_pred_factor, reference = actual_val)
> accuracy1 <- conf_mat1$overall["Accuracy"]
> precision1 <- conf_mat1$byClass["Precision"]
> recall1 <- conf_mat1$byClass["Recall"]
> f1_score1 <- conf_mat1$byClass["F1"]
> # Print the metrics
> print(conf_mat1)
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 1208  134
         1  341  426

               Accuracy : 0.7748
                 95% CI : (0.7563, 0.7924)
    No Information Rate : 0.7345
    P-Value [Acc > NIR] : 1.141e-05

                  Kappa : 0.4835

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

            Sensitivity : 0.7799
            Specificity : 0.7607
         Pos Pred Value : 0.9001
         Neg Pred Value : 0.5554
             Prevalence : 0.7345
         Detection Rate : 0.5728
   Detection Prevalence : 0.6363
      Balanced Accuracy : 0.7703

       'Positive' Class : 0

> print(paste("Accuracy:", accuracy1))
[1] "Accuracy: 0.774774774774775"
> print(paste("Precision:", precision1))
[1] "Precision: 0.900149031296572"
> print(paste("Recall:", recall1))
[1] "Recall: 0.779857972885733"
> print(paste("F1-score:", f1_score1))
```

```
[1] "F1-score: 0.835696990660671"
```

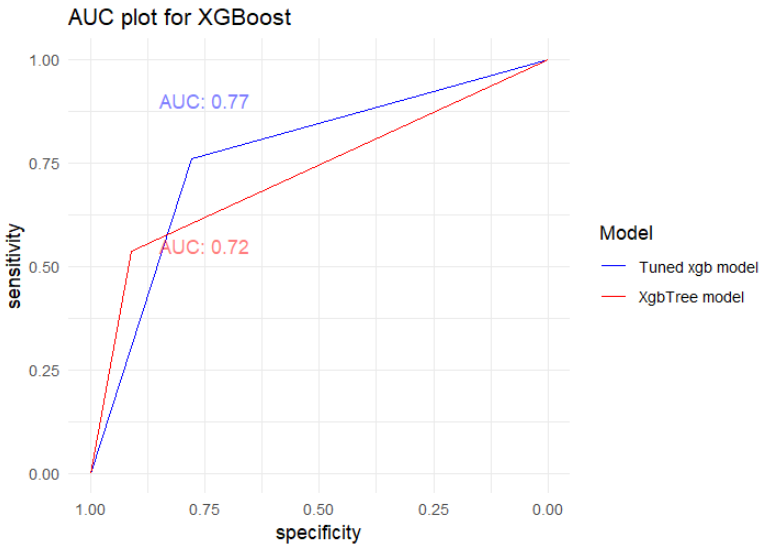
```
>
```

AUC Plot: Compare models

R command:

```r
# ---------------------------------------AUC PLOT: COMPARE MODELS------

# AUC plot
# ref: https://rstudio-pubs-static.s3.amazonaws.com/277278_427ca6a7ce7c4eb688506efc7a6c2435.html
# Multiple roc: ggroc(list(roc1, roc2))

xgb.roc <- roc(response = y_test, predictor = as.numeric(xgb_model_pred))
xgbTree.roc <- roc(response = X_test_all$Churn, predictor = as.numeric(xgb.model_pred))

ggroc(list(xgb.roc, xgbTree.roc)) +
  ggtitle("AUC plot for XGBoost") +
  scale_color_manual(values = c("blue", "red"),
                     labels = c("Tuned xgb model", "XgbTree model")) +
  theme_minimal() +
  annotate("text", x = 0.75, y = 0.9, label = paste("AUC:", round(xgb.roc$auc, 2)),
           alpha = 0.5, color = "blue", size = 4) +
  annotate("text", x = 0.75, y = 0.55, label = paste("AUC:", round(xgbTree.roc$auc, 2)),
           alpha = 0.5, color = "red", size = 4) +
  labs(color = "Model")
```

Plot:



- From above depicted plot, it is clear that the modle2 with hyper parameters like learning rate, The optimization objective, The evaluation metric, **scale_pos_weight,** Maximum depth of a tree and Subsample ratio is tuned and gives the better Area Under Curve(AUC) value.
- AUC measures the ability of the model to distinguish between positive and negative classes. A higher AUC indicates better discrimination, meaning the model is better at correctly ranking the probabilities of positive instances higher than negative instances.

- It provides insights into the model's discrimination ability, robustness to class imbalance, and its overall performance in binary classification tasks.

Conclusion.

Model Selection from all possible models is done in the report by the summary results of all the models. As per the results below, the BOOST model shows highest accuracy and kappa value.

```
> summary(results)

Call:
summary.resamples(object = results)

Models: CART, BAG, BOOST, RF
Number of resamples: 3

Accuracy
          Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
CART  0.7758831 0.7846509 0.7934186 0.7877766 0.7937233 0.7940280    0
BAG   0.7667479 0.7691143 0.7714808 0.7739657 0.7775746 0.7836685    0
BOOST 0.7904994 0.7968341 0.8031688 0.8034145 0.8098720 0.8165753    0
RF    0.7813642 0.7867820 0.7921999 0.7916349 0.7967703 0.8013406    0

Kappa
           Min.   1st Qu.    Median      Mean   3rd Qu.      Max. NA's
CART  0.3328402 0.3688197 0.4047993 0.3917332 0.4211797 0.4375601    0
BAG   0.3564000 0.3730952 0.3897904 0.3862230 0.4011346 0.4124787    0
BOOST 0.4232961 0.4392979 0.4552997 0.4561925 0.4726407 0.4899817    0
RF    0.3909861 0.3941534 0.3973208 0.4085509 0.4173333 0.4373458    0

>
```

After evaluation of both XGBoost models, model2 is giving promising results.

Compare table for models

Parameters	XGBoost general model	XGBoost tuned model
Accuracy	0.81	0.78
Precision	0.53	0.90
Recall	0.68	0.78
F1-score	0.83	0.84
AUC	0.72	0.77

Recommendations

Based on the analysis of your customer churn dataset, here are some key pieces of advice for the owner of the company:

1. **Focus on Retention Strategies for Seniors and Singles**: Seniors and customers without partners or dependents are more likely to leave. Tailor retention strategies specifically for these demographics to improve customer loyalty.

2. **Address Issues with Fiber Optic Internet Service**: Customers with Fiber Optic internet service are churning at a higher rate. Investigate and resolve any issues related to Fiber Optic service quality to retain these customers.

3. **Offer Comprehensive Service Packages**: Customers who lack essential services like Online Security, Online Backup, and Device Protection are more likely to churn. Consider bundling these services or offering incentives to encourage their adoption.

4. **Highlight the Importance of Tech Support**: Customers without Tech Support services are churning more. Emphasize the value of tech support in customer communications and consider offering promotions to encourage sign-up.
5. **Encourage Long-Term Contracts**: Month-to-month contract customers show higher churn rates. Encourage customers to sign longer contracts with benefits such as discounts or additional services to improve retention.
6. **Review Payment Methods and Billing Systems**: Customers paying via Electronic Check and opting for Paperless Billing have higher churn rates. Evaluate these payment methods and billing systems to identify any issues and improve customer satisfaction.
7. **Monitor Customer Tenure and Engagement**: Keep track of customer tenure and engagement levels. Customers with shorter tenure and higher monthly charges are more likely to churn. Implement strategies to increase engagement and satisfaction for these customers.
8. **Understand Skewness in Churn Rate**: Note the skewness in churn rate distribution concerning tenure, monthly charges, and total charges. Use this insight to tailor retention efforts based on different customer segments and their specific characteristics.

By focusing on these areas and implementing targeted strategies, the company can reduce churn rates and improve overall customer retention.

Struggles and difficulties using R-Studio to build the model.

- Using R-Studio for XGBoost model building, specifically for customer churn, can be both rewarding and challenging.
- Preprocessing data to suit XGBoost's requirements demanded meticulous handling, especially with categorical variables.
- Tuning hyperparameters for optimal performance was time-consuming due to the numerous options available.
- Debugging errors in model training required deep understanding of both XGBoost and R.
- Additionally, visualizing model performance and feature importance within R-Studio was perplexing, requiring familiarity with ggplot2 or other visualization packages.
- Despite these challenges, leveraging XGBoost in R-Studio lead me to the robust churn prediction models with careful attention and practice.

Suggest other software(s) works properly with your model? Why?

- Some databases, like SQL Server with the Machine Learning Services, allow anyone to run R code directly within the database environment. It is because, this enables in-database scoring, where predictions are made directly on data stored in the database.

- Frameworks like Shiny to create interactive web applications. These applications can be deployed on web servers for widespread access. It is because, users can input data and get predictions from my model.

Reference

1. "Customer churn prediction in telecom industry using xgboost" by Chen, Y., & Zhang, J. (2018). In Proceedings of the 27th ACM International Conference on Information and Knowledge Management (pp. 2295-2298). This paper discusses the application of XGBoost for customer churn prediction in the telecom industry, focusing on feature engineering and model evaluation.
2. "Customer churn prediction in telecommunication industry using xgboost" by Mishra, S., Agrawal, R. K., & Choudhary, S. R. (2020). In 2020 2nd International Conference on Intelligent Computing, Instrumentation and Control Technologies (ICICICT) (pp. 113-118). IEEE. This paper presents a study on customer churn prediction in the telecom industry using XGBoost, comparing its performance with other machine learning algorithms.
3. AUC plot for comparing the models reference: https://rstudio-pubs-static.s3.amazonaws.com/277278_427ca6a7ce7c4eb688506efc7a6c2435.html
4. Details about using the xgb.train function with hyper parameter tuning reference: https://www.rdocumentation.org/packages/xgboost/versions/1.7.7.1/topics/xgb.train
5. Original dataset reference that explains a lot about the dataset overall: https://community.ibm.com/community/user/businessanalytics/blogs/steven-macko/2019/07/11/telco-customer-churn-1113