Telco Churn

In this Project we shall predict customer churn for a Telecommunications company “Telco”. All credit for the dataset goes to Kaggle User “BlastChar” and can be accessed here: https://www.kaggle.com/blastchar/telco-customer-churn (https://www.kaggle.com/blastchar/telco-customer-churn)

Let’s start off by loading our dataset and the required libraries.

```
#Let's load libraries and import the dataset!

if (!require(data.table)) install.packages('data.table')
```

```
## Loading required package: data.table
```

```
## Warning: package 'data.table' was built under R version 3.6.1
```

```
library(data.table)
if (!require(dplyr)) install.packages('dplyr')
```

```
## Loading required package: dplyr
```

```
## Warning: package 'dplyr' was built under R version 3.6.1
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:data.table':
##
##     between, first, last
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(dplyr)
if (!require(ggplot2)) install.packages('ggplot2')
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 3.6.1
```

```
library(ggplot2)
if (!require(caret)) install.packages('caret')
```

```
## Loading required package: caret
```

```
## Warning: package 'caret' was built under R version 3.6.1
```

```
## Loading required package: lattice
```

```
library(caret)
if (!require(class)) install.packages('class')
```

```
## Loading required package: class
```

```
## Warning: package 'class' was built under R version 3.6.1
```

```
library(class)
library(readr)
```

```
## Warning: package 'readr' was built under R version 3.6.1
```

```
library(readr)
WA_F <- read_csv("https://raw.githubusercontent.com/lukejohnsaid1989/Telco-Churn-Project/mast
er/Telco%20Customer%20ChurnWA_Fn-UseC_-Telco-Customer-Churn.csv")
```

```
## Parsed with column specification:
## cols(
##   .default = col_character(),
##   SeniorCitizen = col_double(),
##   tenure = col_double(),
##   MonthlyCharges = col_double(),
##   TotalCharges = col_double()
## )
```

```
## See spec(...) for full column specifications.
```

```
mydata <-WA_F
```

Sections:

1. Introduction and Dataset exploration

2. Variable Analysis and Data Exploration / Cleanup

3. Model Building

4. Prediction Results and Accuracy Evaluation

5. Conclusions and Future Work

SECTION 1: INTRODUCTION

We will create a Logistic Regression model to predict customer churn.

We will be analysing variable by variable and modifying the data in a way that can be easily interpreted by a Logistic Regression model. Variables with text data will be split into “N” columns where “N” is the number of levels of the variable. These columns will be populated with 1s and 0s depending on the level of the original Variable column. Variables with numeric data can be fit into the regression model as they are.

All variables will then be standardized. This will give the variables zero-mean and unit-variance. This is done to mitigate the issue created by different ranges shown by different variables.

Summing this up, we will:

- Load our data
- Take a look at each variable to:
  - Check for NAs (and replace if necessary)
  - Manipulate Data to make it interpretable by a regression model
- Select which features we will include into our regression model
- Standardize all our data
- Split our data into Train and Test sets
- Fit our Logistic Regression Model
- Use the Train set to predict our Test Set
- Evaluate the accuray of our model

Dataset Exploration Let us now take a look at our dataset.

```
#Let's take a look at our dataset
head(mydata)
```

customerID <chr>	gen… <chr>	SeniorCitizen <dbl>	Partner <chr>	Depende… <chr>	tenure <dbl>	PhoneService <chr>	MultipleLines <chr>
7590-VHVEG	Female	0	Yes	No	1	No	No phone servi
5575-GNVDE	Male	0	No	No	34	Yes	No
3668-QPYBK	Male	0	No	No	2	Yes	No
7795-CFOCW	Male	0	No	No	45	No	No phone servi
9237-HQITU	Female	0	No	No	2	Yes	No
9305-CDSKC	Female	0	No	No	8	Yes	Yes

6 rows | 1-8 of 21 columns

Let’s check for NAs. As per the below output, we have 11 NAs.

```
#Check for NAs
sum(is.na(mydata))
```

```
## [1] 11
```

```
#We have 11 Na's
```

Let us take a look at our **output Variable**; which is the column “Churn”. Let’s check for NAs, and plot. There are much less people who churned than who didn’t. As per the below output, we have no NAs.

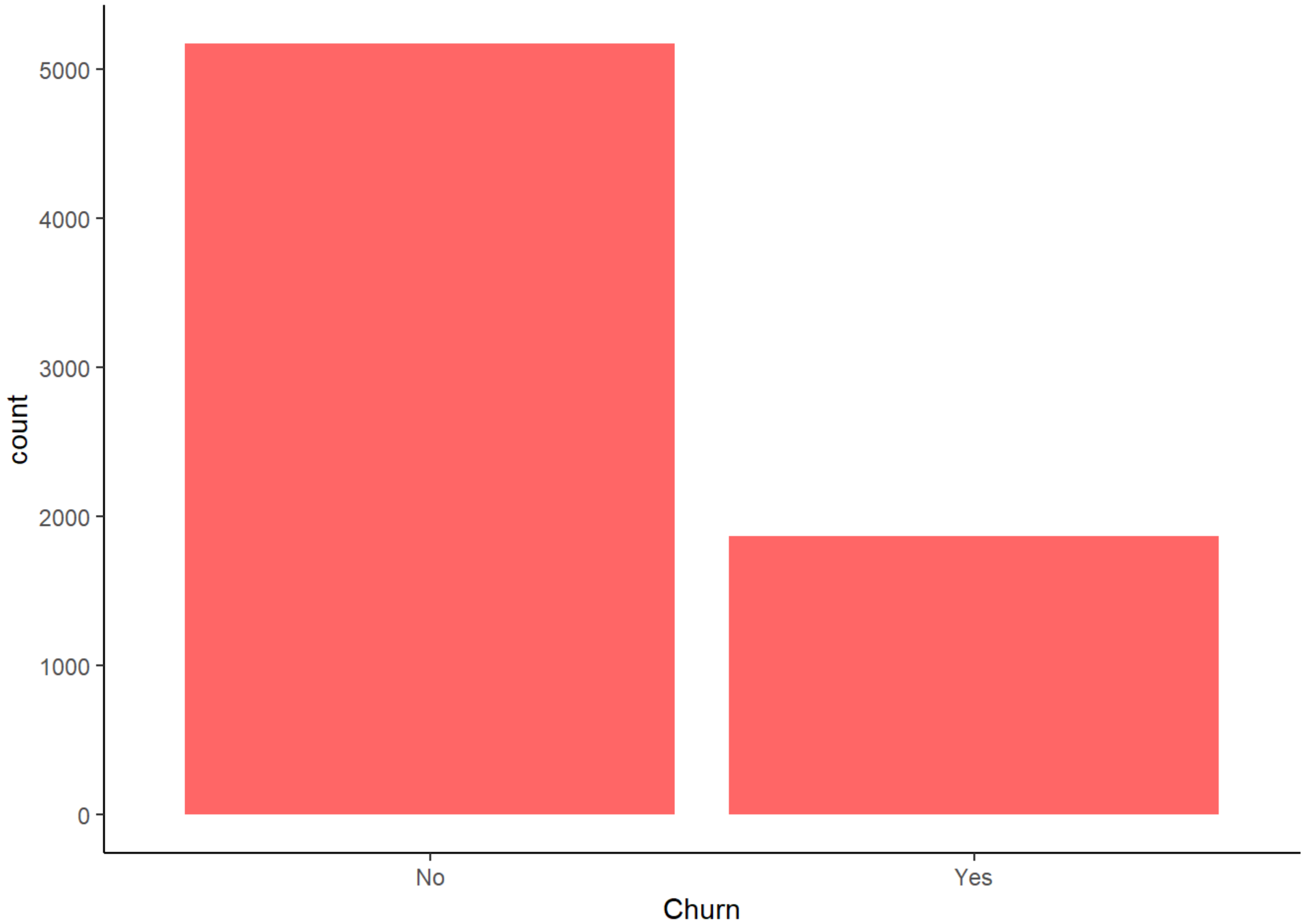
```
#Let's look at our output

base::table(mydata$Churn)
```

```
## 
##   No  Yes 
## 5174 1869
```

```
sum(is.na(mydata$Churn))
```

```
## [1] 0
```

```
ggplot(mydata, aes(mydata$Churn)) + geom_bar(fill = "#FF6666") + theme_classic() + xlab("Chur
n")
```



Since we know that we will be using Logistic Regression - we need to switch our output Variable to 0s and 1s. The value of 1 represents customer churn.

I added a new column “ChurnLog” which contains binary data for Churn.

```
mydata$ChurnLog = 0
mydata$ChurnLog[mydata$Churn == "Yes"] = 1
mydata$ChurnLog[mydata$Churn == "No"] = 0

head(mydata)
```

customerID <chr>	gen... <chr>	SeniorCitizen <dbl>	Partner <chr>	Depende... <chr>	tenure <dbl>	PhoneService <chr>	MultipleLines <chr>
7590-VHVEG	Female	0	Yes	No	1	No	No phone servi
5575-GNVDE	Male	0	No	No	34	Yes	No
3668-QPYBK	Male	0	No	No	2	Yes	No
7795-CFOCW	Male	0	No	No	45	No	No phone servi
9237-HQITU	Female	0	No	No	2	Yes	No
9305-CDSKC	Female	0	No	No	8	Yes	Yes

6 rows | 1-8 of 22 columns

Let's take a look at our column names. Creating a vector with the names of columns is useful if you want to call upon it to check the exact names of the columns.

```
ColVec <- colnames(mydata)
ColVec
```

```
##  [1] "customerID"       "gender"           "SeniorCitizen"
##  [4] "Partner"          "Dependents"       "tenure"
##  [7] "PhoneService"     "MultipleLines"    "InternetService"
## [10] "OnlineSecurity"   "OnlineBackup"     "DeviceProtection"
## [13] "TechSupport"      "StreamingTV"      "StreamingMovies"
## [16] "Contract"         "PaperlessBilling" "PaymentMethod"
## [19] "MonthlyCharges"   "TotalCharges"     "Churn"
## [22] "ChurnLog"
```

SECTION 2: VARIABLE ANALYSIS AND DATA EXPLORATION / CLEANUP

Variable Analysis

```
>>> *GENDER*
```

First we check our levels. We have two, “Male” and “Female”. Then we check for NAs. We have no NAs.

```
base::table(mydata$gender)
```

```
##
## Female   Male
##   3488   3555
```

```
sum(is.na(mydata$gender))
```

```
## [1] 0
```

I created a dataframe using Dummy Variables. This dataframe contains two columns with binary values representing the Gender variable from mydata. This will be used to develop the logistic regression model.

```
Genderframe <-as.data.frame(mydata$gender)
dmy <- dummyVars(" ~ .", data = Genderframe)
Genderframe2 <- data.frame(predict(dmy, newdata = Genderframe))
head(Genderframe2)
```

	X.mydata.gender.Female <dbl>	X.mydata.gender.Male <dbl>
1	1	0
2	0	1
3	0	1
4	0	1
5	1	0
6	1	0

6 rows

Variable Analysis

```
>>> *PARTNER*
```

First we check our levels. We have two, “Yes” and “No”. Then we check for NAs. I added a new column “PartnerLog” which contains binary data for Partner. As per the output - we have no NAs.

```
mydata$PartnerLog = 0
sum(is.na(mydata$Partner))
```

```
## [1] 0
```

```
mydata$PartnerLog[mydata$Partner == "Yes"] = 1
mydata$PartnerLog[mydata$Partner == "No"] = 0
head(mydata)
```

customerID <chr>	gen... <chr>	SeniorCitizen <dbl>	Partner <chr>	Depende... <chr>	tenure <dbl>	PhoneService <chr>	MultipleLines <chr>
7590-VHVEG	Female	0	Yes	No	1	No	No phone servi
5575-GNVDE	Male	0	No	No	34	Yes	No
3668-QPYBK	Male	0	No	No	2	Yes	No
7795-CFOCW	Male	0	No	No	45	No	No phone servi
9237-HQITU	Female	0	No	No	2	Yes	No
9305-CDSKC	Female	0	No	No	8	Yes	Yes

6 rows | 1-8 of 23 columns

Variable Analysis

```
>>> *SENIOR CITIZEN*
```

Let’s take a look at Senior Citizen. It is already in 1s and 0s, so we need not modify it. Also we have no NAs.

```
base::table(mydata$SeniorCitizen)
```

```
##
##    0    1
## 5901 1142
```

```
sum(is.na(mydata$SeniorCitizen))
```

```
## [1] 0
```

Variable Analysis

```
>>>*DEPENDENTS*
```

Now let us take a look at dependents. As per the code ouput below, we have no NAs. We will create another column with Binary Data, as we did for Gender.

```
base::table(mydata$Dependents)
```

```
##
##   No  Yes
## 4933 2110
```

```
sum(is.na(mydata$Dependents))
```

```
## [1] 0
```

```
mydata$DependentsLog = 0
mydata$DependentsLog[mydata$Dependents == "Yes"] = 1
mydata$DependentsLog[mydata$Dependents == "No"] = 0
tail(mydata)
```

customerID <chr>	gen… <chr>	SeniorCitizen <dbl>	Partner <chr>	Depende… <chr>	tenure <dbl>	PhoneService <chr>	MultipleLines <chr>
2569-WGERO	Female	0	No	No	72	Yes	No
6840-RESVB	Male	0	Yes	Yes	24	Yes	Yes
2234-XADUH	Female	0	Yes	Yes	72	Yes	Yes
4801-JZAZL	Female	0	Yes	Yes	11	No	No phone servi
8361-LTMKD	Male	1	Yes	No	4	Yes	Yes
3186-AJIEK	Male	0	No	No	66	Yes	No

6 rows | 1-8 of 24 columns

Variable Analysis

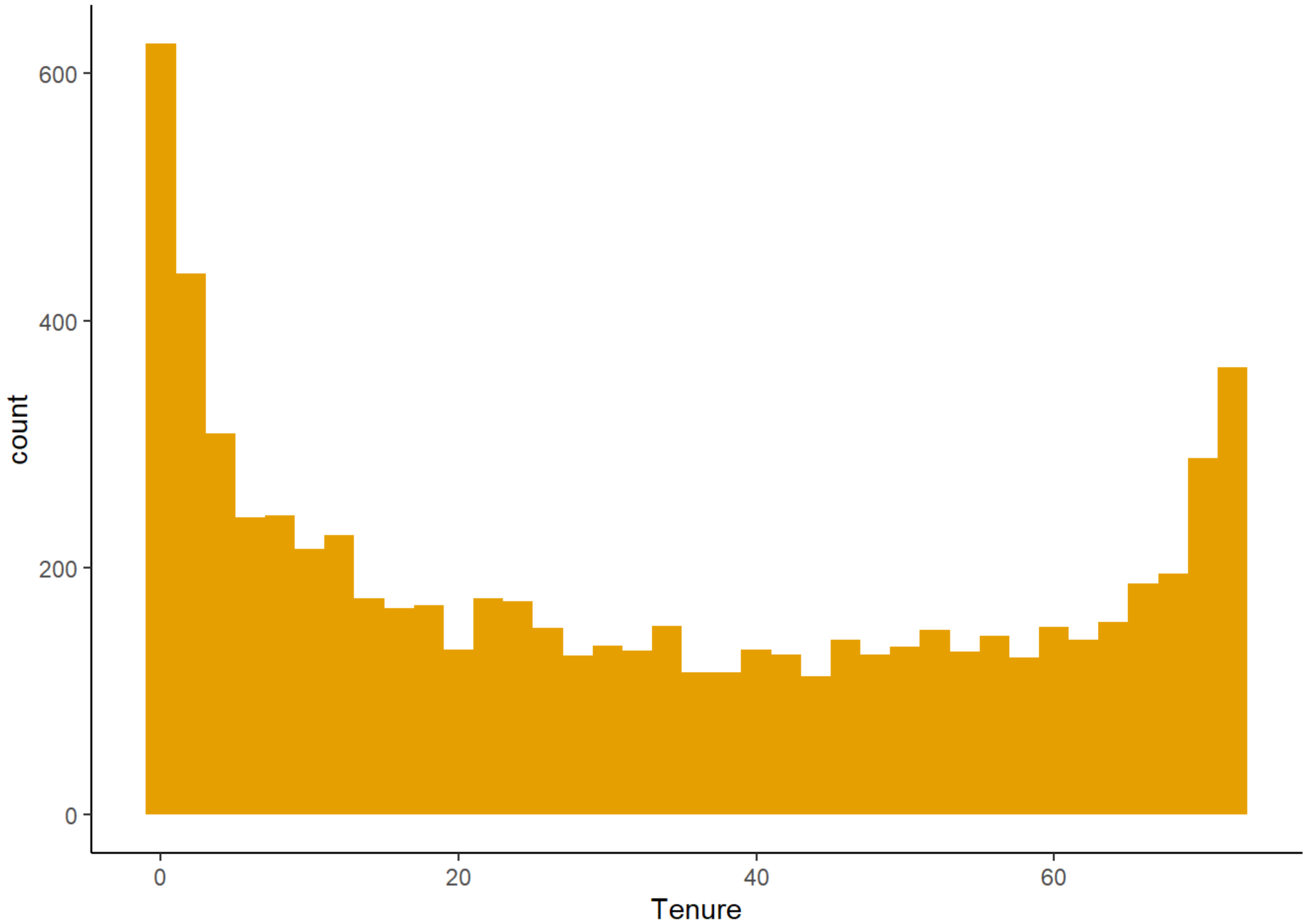
```
>>>*TENURE*
```

Let’s take a look at tenure. As per the code output, we have no NAs. The distribution is not Normal. When looking at the violin plot it is evident that there is a higher probability that people with lower tenure will Churn.

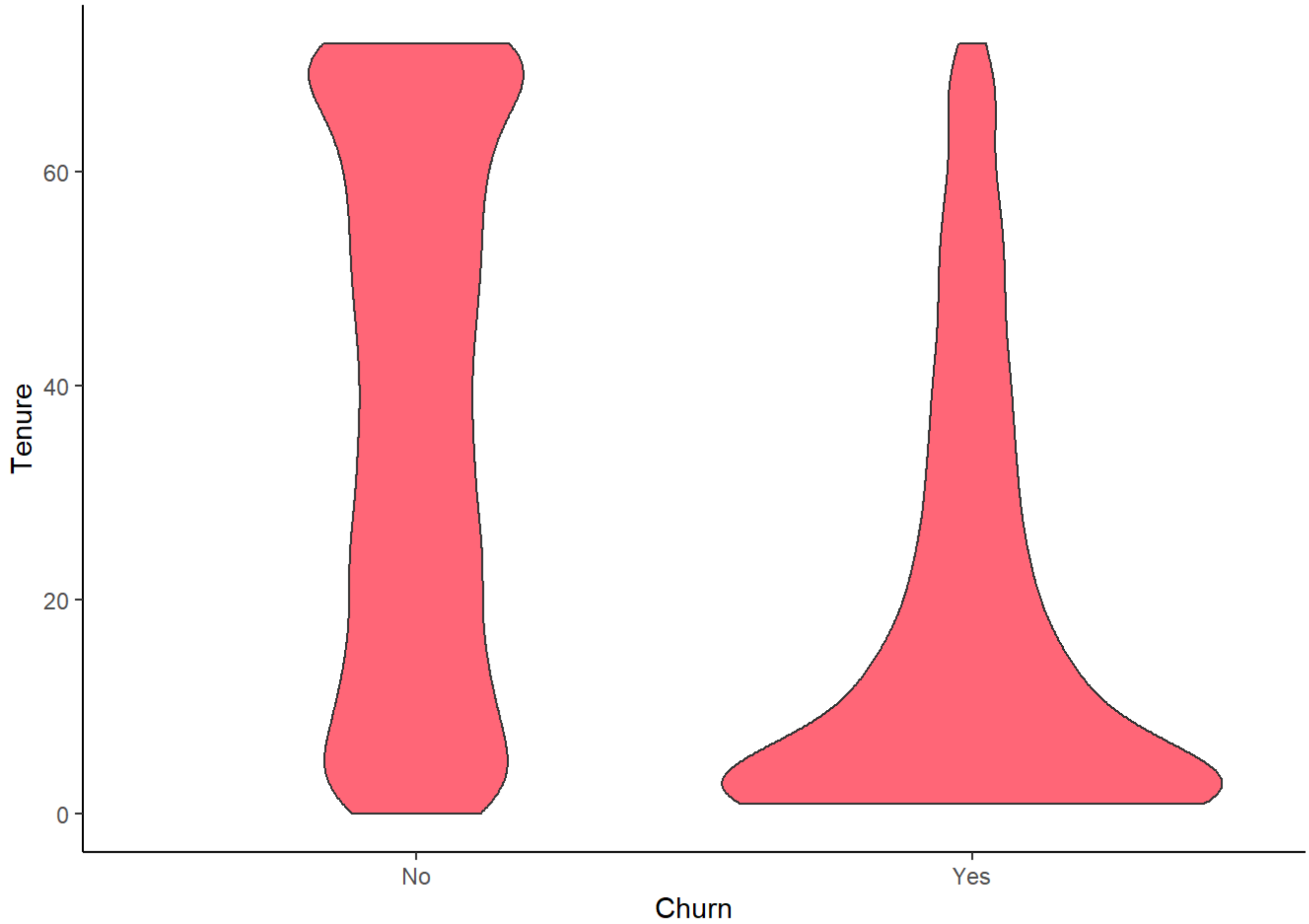
```
sum(is.na(mydata$tenure))
```

```
## [1] 0
```

```
ggplot(mydata, aes(mydata$tenure)) + geom_histogram(binwidth = 2, fill = "#E69F00") + theme_classic() +xlab("Tenure") +  theme_classic()
```




```
ggplot(mydata, aes(mydata$Churn, mydata$tenure)) + geom_violin(fill = "#FF6677") + xlab("Churn") +ylab("Tenure") + theme_classic()
```



Now Lets create a dataframe to split the tenure data into:

1. Values with zero value
2. Values betwen Between 1 and 20
3. Values between Between 21 and 40
4. Values between Between 41 and 60
5. Values over 61

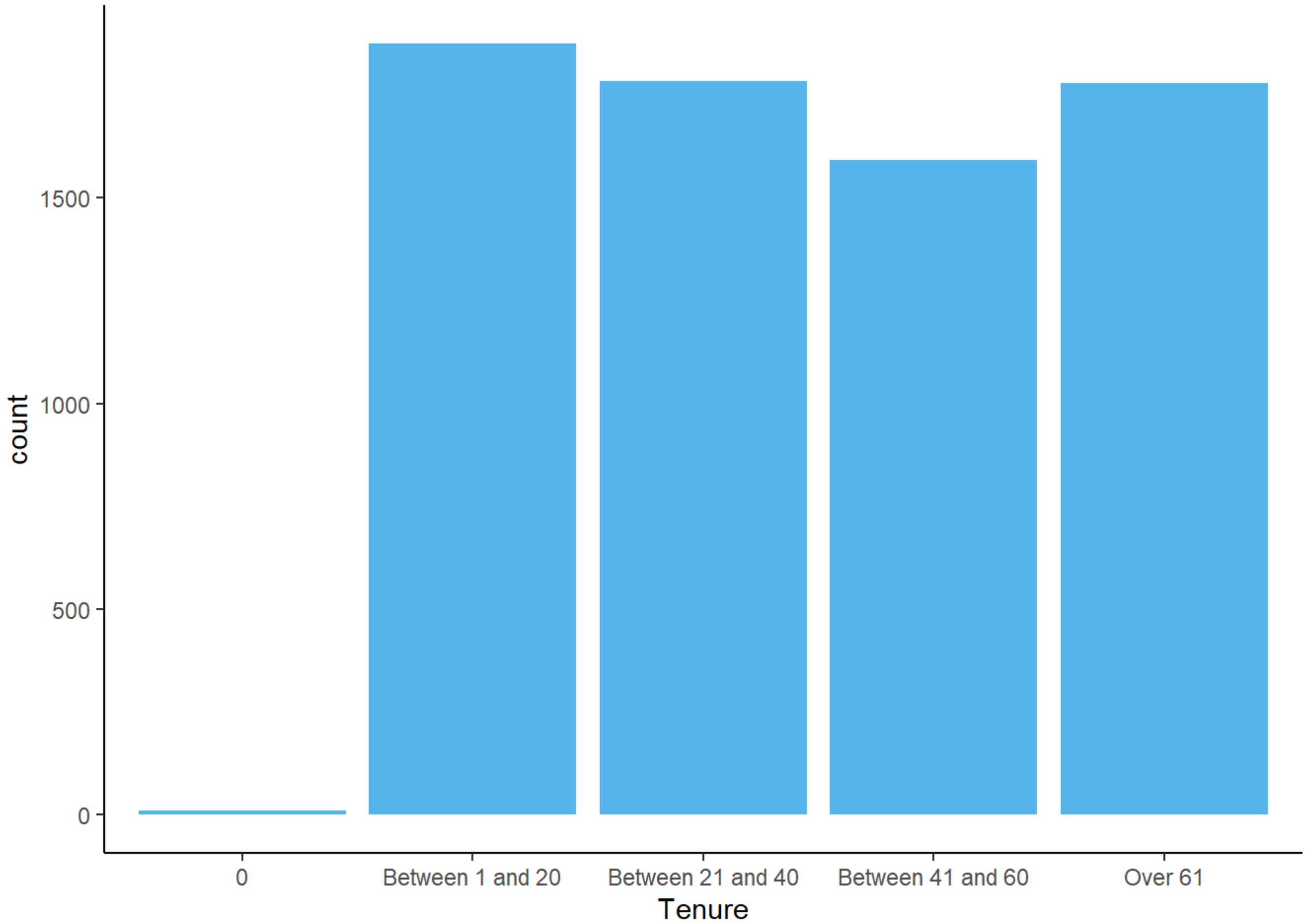
```
TenureFrame <- as.data.frame(mydata$tenure)
x = 1
while (x<=nrow(TenureFrame)){
  if ((TenureFrame$`mydata$tenure`[x] > 0) & (TenureFrame$`mydata$tenure`[x] <= 20))
        {TenureFrame$`mydata$tenure`[x] = "Between 1 and 20"}
  else if ((TenureFrame$`mydata$tenure`[x] > 20) & (TenureFrame$`mydata$tenure`[x] <= 40))
        {TenureFrame$`mydata$tenure`[x] = "Between 21 and 40"}
  else if ((TenureFrame$`mydata$tenure`[x] > 40) & (TenureFrame$`mydata$tenure`[x] <= 60))
        {TenureFrame$`mydata$tenure`[x] = "Between 41 and 60"}
  else if (TenureFrame$`mydata$tenure`[x] > 60)
  {TenureFrame$`mydata$tenure`[x] = "Over 61"}
  x = x + 1
}
```

```
base::table(TenureFrame$`mydata$tenure`)
```

```
##
##                 0  Between 1 and 20 Between 21 and 40 Between 41 and 60
##                11              1875              1784              1593
##           Over 61
##              1780
```

Let’s plot out Tenure data again.

```
ggplot(mydata, aes(TenureFrame$`mydata$tenure`)) + geom_bar(fill = "#56B4E9") + theme_classic
() + xlab("Tenure")
```



Now let’s use Dummy Vars to create a dataframe with columns containing binary data representing the tenure periods defined. Loading up this data frame, we can see that we have the 5 columns we defined.

```
# let's create dummy var!

dmy <- dummyVars(" ~ .", data = TenureFrame)
TenureFrame2 <- data.frame(predict(dmy, newdata = TenureFrame))
head(TenureFrame2)
```

	X.mydata.tenure.0 <dbl>	X.mydata.tenure.Between.1.and.20 <dbl>	X.mydata.tenure.Between.
1	0	1	
2	0	0	
3	0	1	

	X.mydata.tenure.0 <dbl>	X.mydata.tenure.Between.1.and.20 <dbl>	X.mydata.tenure.Between.
4	0	0	
5	0	1	
6	0	0	

6 rows | 1-4 of 6 columns

Variable Analysis

```
>>>*ANALYSIS OF SERVICES AVAILABLE*
```

The following Variables are all related to the Phone and Internet services:

```
Phone Service
MultipleLines
InternetService
OnlineSecurity
OnlineBackup
DeviceProtection
TechSupport
StreamingTV
SreamingMovies
```

We will first check for NAs in each variable. Then we will take a look at the levels of each Variable. No NAs as per the below output. Also, it can be noted that we have a lot of duplicate information. In each service related to Internet service (such as Online backup and Streaming Movies), we have a level specifying “No Internet Service”. This is the same amount (1526) showing “No” in the “Internet Service” Variable.

```
base::table(mydata$PhoneService)
```

```
##
##   No  Yes
##  682 6361
```

```
sum(is.na(mydata$PhoneService))
```

```
## [1] 0
```

```
base::table(mydata$MultipleLines)
```

```
##
##               No No phone service              Yes
##             3390              682             2971
```

```
sum(is.na(mydata$MultipleLines))
```

```
## [1] 0
```

```
base::table(mydata$InternetService)
```

```
##
##         DSL Fiber optic          No
##        2421        3096        1526
```

```
sum(is.na(mydata$InternetService))
```

```
## [1] 0
```

```
base::table(mydata$OnlineSecurity)
```

```
##
##                  No No internet service                 Yes
##                3498                1526                2019
```

```
sum(is.na(mydata$OnlineSecurity))
```

```
## [1] 0
```

```
base::table(mydata$OnlineBackup)
```

```
##
##                  No No internet service                 Yes
##                3088                1526                2429
```

```
sum(is.na(mydata$OnlineBackup))
```

```
## [1] 0
```

```
base::table(mydata$DeviceProtection)
```

```
##
##                  No No internet service                 Yes
##                3095                1526                2422
```

```
sum(is.na(mydata$DeviceProtection))
```

```
## [1] 0
```

```
base::table(mydata$TechSupport)
```

```
## 
##                  No No internet service                 Yes 
##                3473                1526                2044
```

```
sum(is.na(mydata$TechSupport))
```

```
## [1] 0
```

```
base::table(mydata$StreamingTV)
```

```
## 
##                  No No internet service                 Yes 
##                2810                1526                2707
```

```
sum(is.na(mydata$StreamingTV))
```

```
## [1] 0
```

```
base::table(mydata$StreamingMovies)
```

```
## 
##                  No No internet service                 Yes 
##                2785                1526                2732
```

```
sum(is.na(mydata$StreamingMovies))
```

```
## [1] 0
```

We will now use Dummy Variables to create a dataframe with binary data on each of these Variables.

!We must then remove colums containing duplicate data!

```
ServiceFrame <- cbind.data.frame(mydata$PhoneService, mydata$MultipleLines, mydata$InternetSe
rvice, mydata$OnlineSecurity, mydata$OnlineBackup, mydata$DeviceProtection, mydata$TechSuppor
t, mydata$StreamingTV, mydata$StreamingMovies)
ncol(ServiceFrame)
```

```
## [1] 9
```

```
dmy <- dummyVars(" ~ .", data = ServiceFrame)
ServiceFrame2 <- data.frame(predict(dmy, newdata = ServiceFrame))
head(ServiceFrame2 )
```

	X.mydata.PhoneService.No <dbl>	X.mydata.PhoneService.Yes <dbl>	X.mydata.MultipleLines.No <dbl>
1	1	0	0
2	0	1	1
3	0	1	1
4	1	0	0
5	0	1	1
6	0	1	0

6 rows | 1-4 of 27 columns

It can be observed that there are a lot of columns with duplicate information. The following are examples:

X.mydata.PhoneService.No X.mydata.MultipleLines.No.phone.service We will remove the redunant columns:

```
ServiceFrame2ColVec <-colnames(ServiceFrame2)
ServiceFrame2ColVec
```

```
##  [1] "X.mydata.PhoneService.No"
##  [2] "X.mydata.PhoneService.Yes"
##  [3] "X.mydata.MultipleLines.No"
##  [4] "X.mydata.MultipleLines.No.phone.service"
##  [5] "X.mydata.MultipleLines.Yes"
##  [6] "X.mydata.InternetService.DSL"
##  [7] "X.mydata.InternetService.Fiber.optic"
##  [8] "X.mydata.InternetService.No"
##  [9] "X.mydata.OnlineSecurity.No"
## [10] "X.mydata.OnlineSecurity.No.internet.service"
## [11] "X.mydata.OnlineSecurity.Yes"
## [12] "X.mydata.OnlineBackup.No"
## [13] "X.mydata.OnlineBackup.No.internet.service"
## [14] "X.mydata.OnlineBackup.Yes"
## [15] "X.mydata.DeviceProtection.No"
## [16] "X.mydata.DeviceProtection.No.internet.service"
## [17] "X.mydata.DeviceProtection.Yes"
## [18] "X.mydata.TechSupport.No"
## [19] "X.mydata.TechSupport.No.internet.service"
## [20] "X.mydata.TechSupport.Yes"
## [21] "X.mydata.StreamingTV.No"
## [22] "X.mydata.StreamingTV.No.internet.service"
## [23] "X.mydata.StreamingTV.Yes"
## [24] "X.mydata.StreamingMovies.No"
## [25] "X.mydata.StreamingMovies.No.internet.service"
## [26] "X.mydata.StreamingMovies.Yes"
```

```
ServiceFrame2 <-subset(ServiceFrame2, select =-X.mydata.MultipleLines.No.phone.service)
ServiceFrame2 <-subset(ServiceFrame2, select =-X.mydata.OnlineSecurity.No.internet.service)
ServiceFrame2 <-subset(ServiceFrame2, select =-X.mydata.TechSupport.No.internet.service)
ServiceFrame2 <-subset(ServiceFrame2, select =-X.mydata.StreamingTV.No.internet.service)
ServiceFrame2 <-subset(ServiceFrame2, select =-X.mydata.StreamingMovies.No.internet.service)
ServiceFrame2 <-subset(ServiceFrame2, select =-X.mydata.OnlineBackup.No.internet.service)
ServiceFrame2 <-subset(ServiceFrame2, select =-X.mydata.DeviceProtection.No.internet.service)

head(ServiceFrame2)
```

	X.mydata.PhoneService.No <dbl>	X.mydata.PhoneService.Yes <dbl>	X.mydata.MultipleLines.No <dbl>
1	1	0	0
2	0	1	1
3	0	1	1
4	1	0	0
5	0	1	1
6	0	1	0

6 rows | 1-4 of 20 columns

```
ncol(ServiceFrame2)
```

```
## [1] 19
```

Variable Analysis

```
>>>*CONTRACT*
```

Let’s take a look at contract. We have no NAs, and we have 3 levels. Again, lets use Dummy Variables to creata a dataframe with binary data.

```
base::table(mydata$Contract)
```

```
##
## Month-to-month       One year       Two year
##           3875           1473           1695
```

```
sum(is.na(mydata$Contract))
```

```
## [1] 0
```

```
ContractFrame <- cbind.data.frame(mydata$Contract)
dmy <- dummyVars(" ~ .", data = ContractFrame)
ContractFrame2 <- data.frame(predict(dmy, newdata = ContractFrame))
head(ContractFrame2)
```

	X.mydata.Contract.Month.to.month <dbl>	X.mydata.Contract.One.year <dbl>	X.mydata.Contr
1	1	0	
2	0	1	
3	1	0	
4	0	1	
5	1	0	
6	1	0	

6 rows

Variable Analysis

```
>>>*PAPERLESS BILLING*
```

We have no NAs in this variable. Let’s create a column with binary data.

```
base::table(mydata$PaperlessBilling)
```

```
##
##   No  Yes
## 2872 4171
```

```
sum(is.na(mydata$PaperlessBilling))
```

```
## [1] 0
```

```
mydata$PaperlessBillingLog[mydata$PaperlessBilling == "Yes"] = 1
```

```
## Warning: Unknown or uninitialised column: 'PaperlessBillingLog'.
```

```
mydata$PaperlessBillingLog[mydata$PaperlessBilling == "No"] = 0

head(mydata)
```

customerID <chr>	gen... <chr>	SeniorCitizen <dbl>	Partner <chr>	Depende... <chr>	tenure <dbl>	PhoneService <chr>	MultipleLines <chr>
7590-VHVEG	Female	0	Yes	No	1	No	No phone servi
5575-GNVDE	Male	0	No	No	34	Yes	No
3668-QPYBK	Male	0	No	No	2	Yes	No
7795-CFOCW	Male	0	No	No	45	No	No phone servi
9237-HQITU	Female	0	No	No	2	Yes	No
9305-CDSKC	Female	0	No	No	8	Yes	Yes

6 rows | 1-8 of 25 columns

Variable Analysis

```
>>>*PAYMENT METHOD*
```

Let's take a look at payment method. We have no NAs, and 4 levels. Let's create another dataframe with binary values.

```
base::table(mydata$PaymentMethod)
```

```
##
## Bank transfer (automatic)   Credit card (automatic)
##                      1544                      1522
##          Electronic check              Mailed check
##                      2365                      1612
```

```
sum(is.na(mydata$PaymentMethod))
```

```
## [1] 0
```

```
PaymentFrame <- as.data.frame(mydata$PaymentMethod)

dmy <- dummyVars(" ~ .", data = PaymentFrame)
PaymentFrame2 <- data.frame(predict(dmy, newdata = PaymentFrame))
head(PaymentFrame2)
```

	X.mydata.PaymentMethod.Bank.transfer..automatic. <dbl>	X.mydata.PaymentMeth
1	0	
2	0	
3	0	
4	1	
5	0	
6	0	

6 rows | 1-3 of 5 columns

Variable Analysis

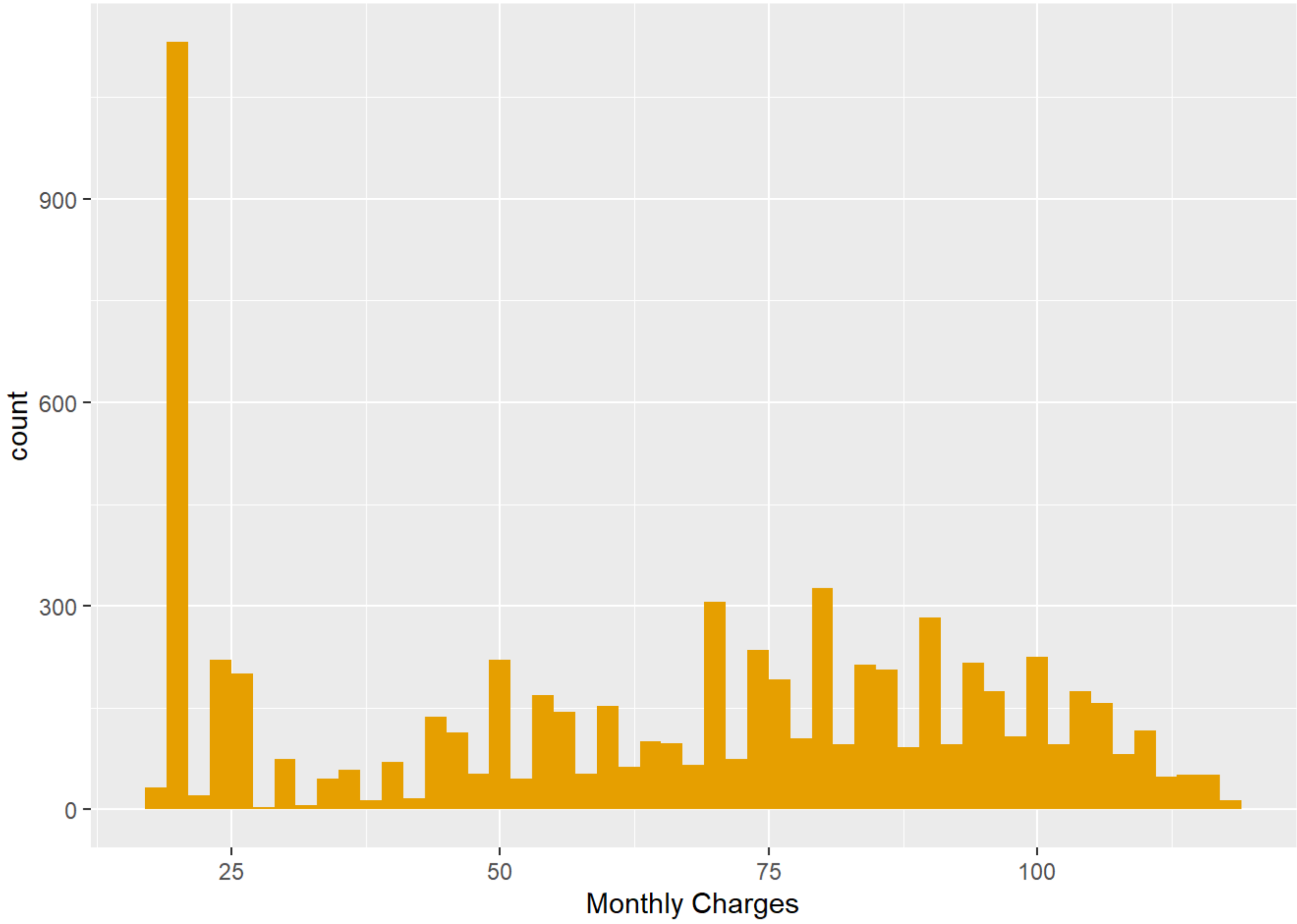
```
>>>*MONTHLY CHARGES*
```

Let’s look at monthly charges. We have no NAs. Talking a look at the Violin graph, it appears that those with high monthly charges are more likely to churn.

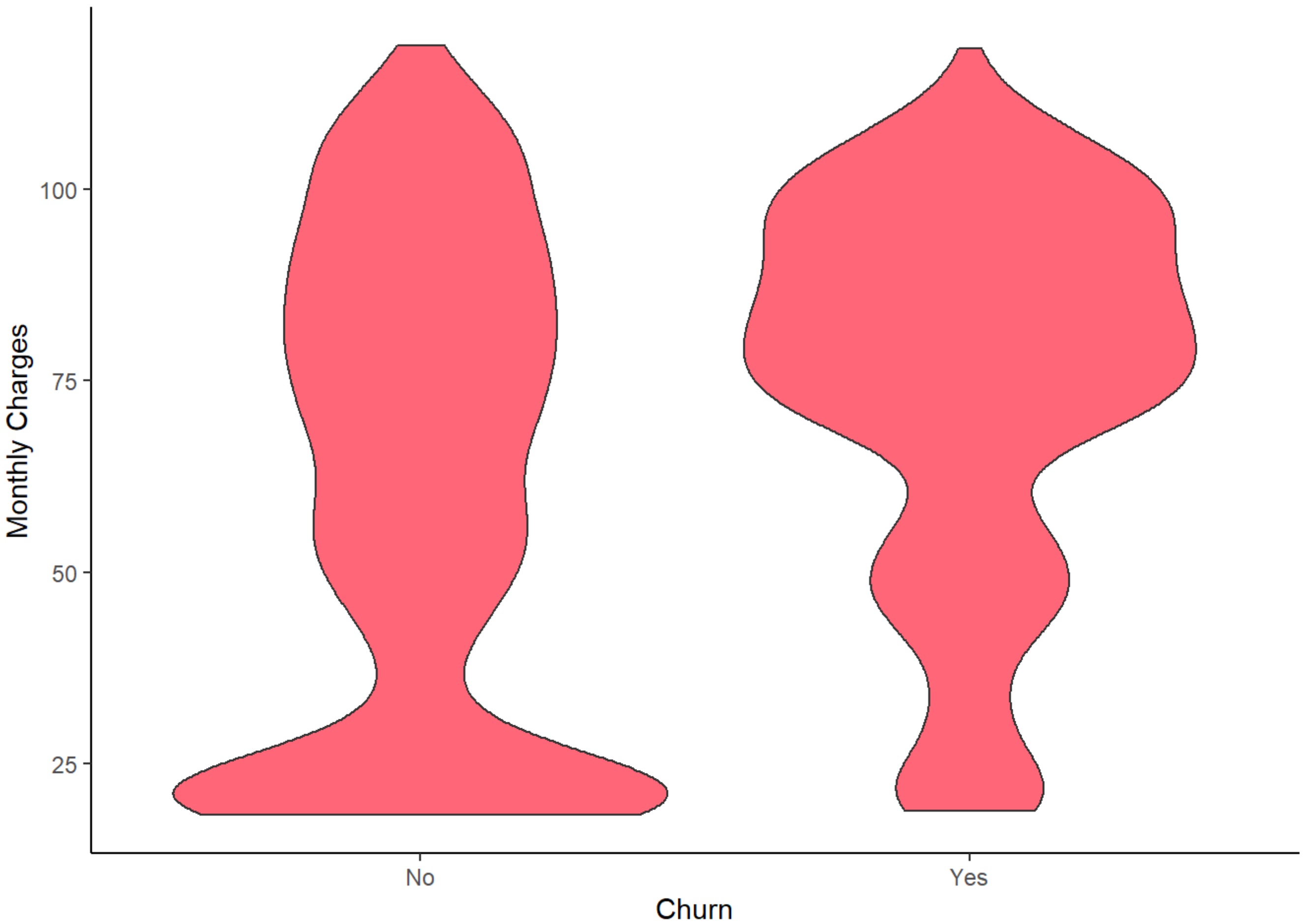
```
sum(is.na(mydata$MonthlyCharges))
```

```
## [1] 0
```

```
ggplot(mydata, aes(mydata$MonthlyCharges)) + geom_histogram(binwidth = 2, fill = "#E69F00") +
xlab("Monthly Charges")
```




```
ggplot(mydata, aes(mydata$Churn, mydata$MonthlyCharges)) + geom_violin(fill = "#FF6677") + xl
ab("Churn") +ylab("Monthly Charges") + theme_classic()
```



Variable Analysis

```
>>>*TOTAL CHARGES*
```

Let’s take a look at total charges. We have 11 Nas.

```
sum(is.na(mydata$TotalCharges))
```

```
## [1] 11
```

VALUE IMPUTATION

We need to replace the 11 Na values that we have in the “Total Charges” Column.

Let’s take a look at the relationship between Monthly Charges and Total Charges. Let’s create a column to Analyse this relationship. We will divide the Total Charges by the tenure and verify if it is similar to the Monthly Charges. Indeed they are very similar. In this regard, for the values which are missing from “Total Charges” we can simply multiply the monthly charges with the tenure.

```
mydata$TotalChargesDivTenure <- (mydata$TotalCharges/mydata$tenure)

test <-cbind.data.frame(mydata$MonthlyCharges,mydata$TotalChargesDivTenure)
head(test)
```

	mydata$MonthlyCharges <dbl>	**mydata$TotalChargesDivTenure** <dbl>
1	29.85	29.85000
2	56.95	55.57353
3	53.85	54.07500
4	42.30	40.90556
5	70.70	75.82500
6	99.65	102.56250

6 rows

```
## Missing value replacement

x=1

while (x <= nrow(mydata))
{
  if ((is.na(mydata$TotalCharges[x])) == TRUE)

  { mydata$TotalCharges[x] <- (mydata$MonthlyCharges[x]*mydata$tenure[x])}

  x = x + 1
}
```

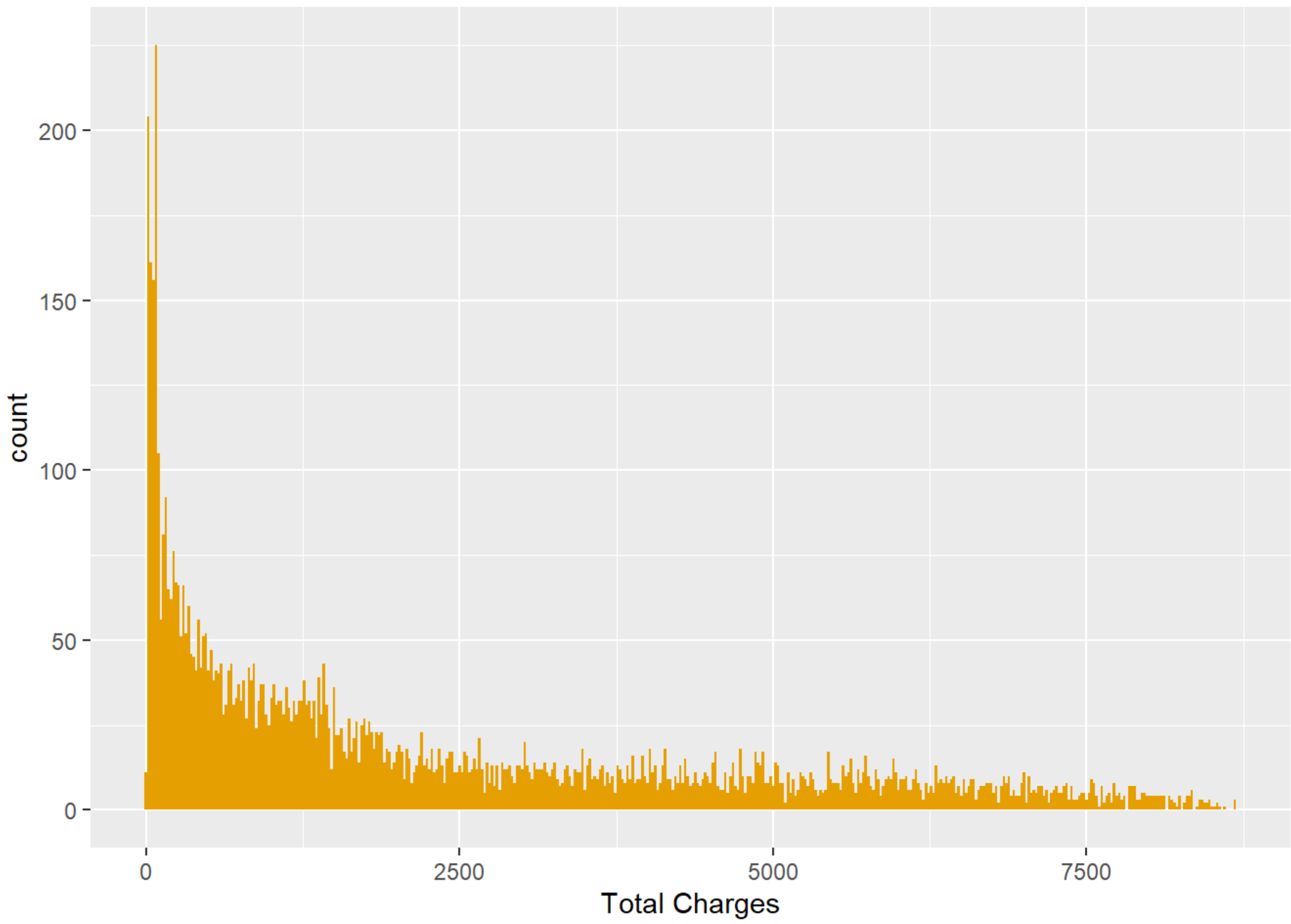
Check NAs. Now that NAs are gone, we can take a better look at our Total Charges Variable.

```
sum(is.na(mydata$TotalCharges))
```

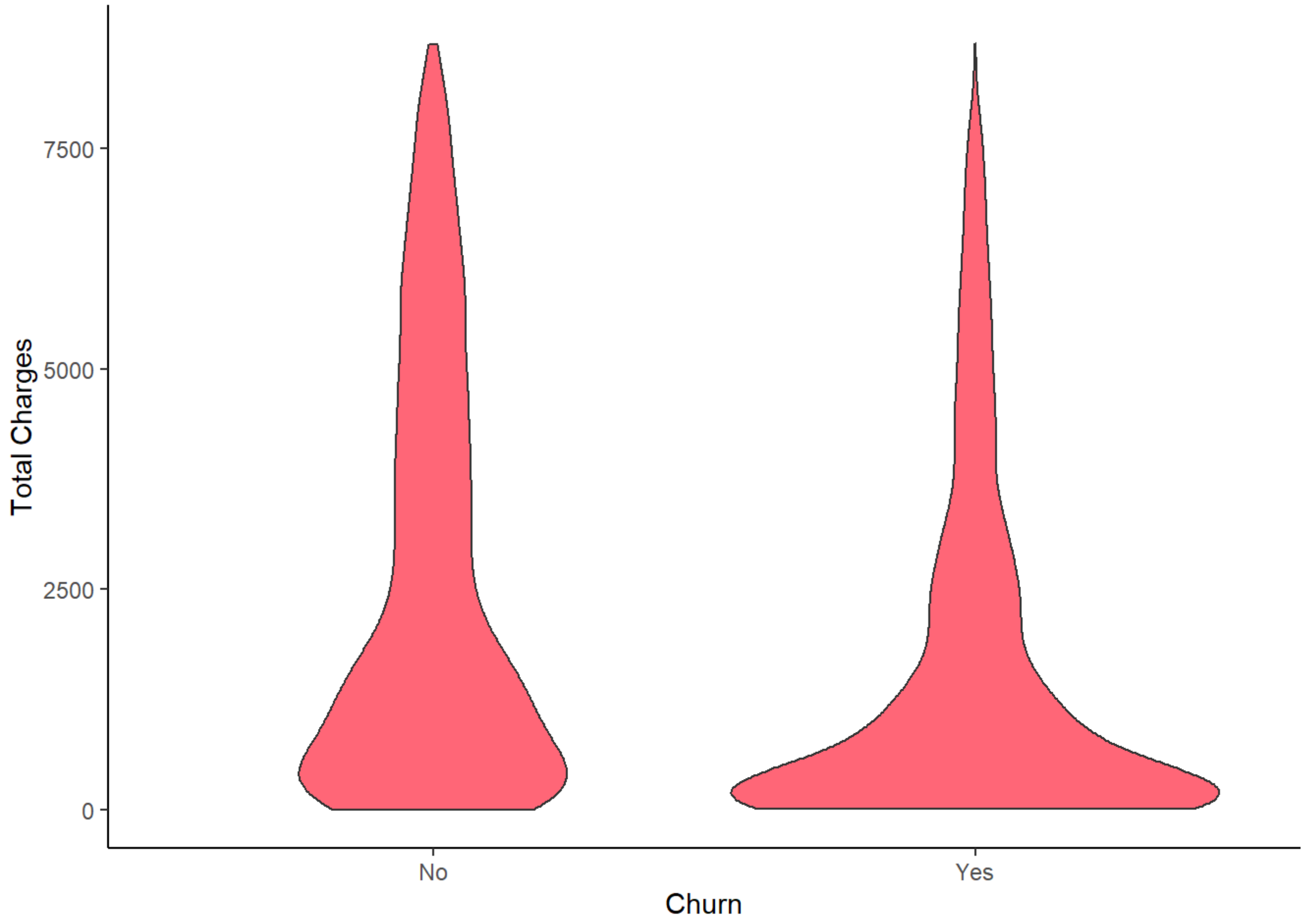
```
## [1] 0
```

Let's take a look at our Total Charges:

```
ggplot(mydata, aes(mydata$TotalCharges)) + geom_histogram(binwidth = 20, fill = "#E69F00") +
 xlab("Total Charges")
```




```
ggplot(mydata, aes(mydata$Churn, mydata$TotalCharges)) + geom_violin(fill = "#FF6677") + xlab
("Churn") +ylab("Total Charges") + theme_classic()
```



Feature Selection

We will now select which features to include into our model. Let's check the column names from the original dataframe "mydata".

- 'customerID' - No inclusion
- 'gender' - No inclusion
- 'SeniorCitizen' - Include
- 'Partner' - No inclusion
- 'Dependents'- No inclusion
- 'tenure' - Include
- 'PhoneService' - No inclusion
- 'MultipleLines' - No inclusion
- 'InternetService' - No inclusion
- 'OnlineSecurity'- No inclusion
- 'OnlineBackup' - No inclusion
- 'DeviceProtection' - No inclusion
- 'TechSupport' - No inclusion
- 'StreamingTV' - No inclusion
- 'StreamingMovies' - No inclusion
- 'Contract' - No inclusion
- 'PaperlessBilling' - No inclusion

- 'PaymentMethod' - No inclusion
- 'MonthlyCharges' - Include
- 'TotalCharges' - Include
- 'Churn' - No inclusion
- 'ChurnLog' - Include
- 'PartnerLog' - Include
- 'DependentsLog' - Include
- 'PaperlessBillingLog' - Include
- 'TotalChargesDivTenure' - No inclusion

We will bind these columns with the dataframes we created: * PaymentFrame2 * TenureFrame2 * ContractFrame2 * Genderframe2 * ServiceFrame2

We will name our data with the selected features: newdata

```
head(mydata)
```

customerID <chr>	gen... <chr>	SeniorCitizen <dbl>	Partner <chr>	Depende... <chr>	tenure <dbl>	PhoneService <chr>	MultipleLines <chr>
7590-VHVEG	Female	0	Yes	No	1	No	No phone servi
5575-GNVDE	Male	0	No	No	34	Yes	No
3668-QPYBK	Male	0	No	No	2	Yes	No
7795-CFOCW	Male	0	No	No	45	No	No phone servi
9237-HQITU	Female	0	No	No	2	Yes	No
9305-CDSKC	Female	0	No	No	8	Yes	Yes

6 rows | 1-8 of 26 columns

```
ColVec <- colnames(mydata)
ColVec
```

```
##  [1] "customerID"           "gender"
##  [3] "SeniorCitizen"        "Partner"
##  [5] "Dependents"           "tenure"
##  [7] "PhoneService"         "MultipleLines"
##  [9] "InternetService"      "OnlineSecurity"
## [11] "OnlineBackup"         "DeviceProtection"
## [13] "TechSupport"          "StreamingTV"
## [15] "StreamingMovies"      "Contract"
## [17] "PaperlessBilling"     "PaymentMethod"
## [19] "MonthlyCharges"       "TotalCharges"
## [21] "Churn"                "ChurnLog"
## [23] "PartnerLog"           "DependentsLog"
## [25] "PaperlessBillingLog"  "TotalChargesDivTenure"
```

```
newdata <- cbind.data.frame(mydata$SeniorCitizen, mydata$PartnerLog, mydata$DependentsLog, my
data$tenure, Genderframe2, PaymentFrame2, ServiceFrame2, TenureFrame2, mydata$PaperlessBillin
gLog, ContractFrame2, mydata$MonthlyCharges, mydata$TotalCharges, mydata$ChurnLog)
head(newdata)
```

	mydata$SeniorCitizen <dbl>	mydata$PartnerLog <dbl>	mydata$DependentsLog <dbl>	mydata$tenure <dbl>
1	0	1	0	1
2	0	0	0	34
3	0	0	0	2
4	0	0	0	45
5	0	0	0	2
6	0	0	0	8

6 rows | 1-5 of 42 columns

Standardizing the Data

This will give the variables zero-mean and unit-variance. This is required for interpretation for a Logistic Regression model.

```
x = 1

while (x<=length(colnames(newdata))) {
  newdata[,x] <- scale(as.numeric(newdata[,x], center = TRUE, scale = TRUE))
  x = x+1
}

newdata$`mydata$ChurnLog` <-mydata$ChurnLog

head(newdata)
```

	mydata$SeniorCitizen <dbl>	mydata$PartnerLog <dbl>	mydata$DependentsLog <dbl>	mydata$tenure <dbl>
1	-0.4398853	1.0344568	-0.6539655	-1.27735389
2	-0.4398853	-0.9665537	-0.6539655	0.06632271
3	-0.4398853	-0.9665537	-0.6539655	-1.23663642
4	-0.4398853	-0.9665537	-0.6539655	0.51421491
5	-0.4398853	-0.9665537	-0.6539655	-1.23663642
6	-0.4398853	-0.9665537	-0.6539655	-0.99233158

6 rows | 1-5 of 42 columns

```
bound = 0.5
train = newdata[1:(bound*(nrow(newdata))),]
test = newdata[(((nrow(mydata))*bound)+1):nrow(newdata),]
print("The number of training rows are")
```

```
## [1] "The number of training rows are"
```

```
nrow(train)
```

```
## [1] 3521
```

```
print("The number of test rows are")
```

```
## [1] "The number of test rows are"
```

```
nrow(test)
```

```
## [1] 3521
```

SECTION 3: MODEL BUILDING

We will use Logistic Regression as the machine learning model.

```
model <- glm( as.numeric(`mydata$ChurnLog`) ~.
              ,family=binomial(link='logit'),data=train, control = list(maxit = 200))
```

SECTION 4: PREDICTION RESULTS AND ACCURACY EVALUATION

> > > Make a prediction

We will now use our test data to make a prediction.

```
predictChurn <- predict(model, newdata = test,type='response')
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type
## == : prediction from a rank-deficient fit may be misleading
```

Let’s round our prediction up, since we want a 1 or 0 as an output.

```
RoundpredictChurn <- round(predictChurn)
head(RoundpredictChurn)
```

```
## 3522 3523 3524 3525 3526 3527
##    0    0    0    1    0    0
```

Measure Accuracy

```
conf_mat <- base::table(test$`mydata$ChurnLog`,RoundpredictChurn)
conf_mat
```

```
##    RoundpredictChurn
##        0    1
##   0 2298  271
##   1  440  512
```

```
cat("Test accuracy: ", sum(diag(conf_mat))/sum(conf_mat))
```

```
## Test accuracy:  0.7980687
```

SECTION 5: CONCLUSIONS AND FUTURE WORK

Several other machine learning models could be applied such as K Nearest Neighbour or Descision Trees. A comparison of the accuracy of these models could be provided, and the most accurate model would be selected. Logistic regression was chosen since it plots the data on a curve which fits a Probabilistic curve with two binary outputs. This is reflective of customer churn behaviour.