Customer Churn Analytics

Data Science Academy

Vinicius Biazon

Customer churn occurs when customers or subscribers stop doing business with a company or service.

One sector in which knowing and predicting cancellation rates is particularly useful is the telecommunications sector, because most customers have several options to choose from within a geographic location.

In this project, I predicted the churn of customers using a telecommunications data set offered for free on the IBM Sample Data Sets portal. I used logistic regression, the decision tree and the random forest as models of Machine Learning.

https://www.ibm.com/communities/analytics/watson-analytics-blog/guide-to-sample-datasets/

Step 1 - Defining the working directory and Loading the packages

```
# Defining the working directory
setwd("D:/Documentos/FCD/BigDataRAzure/Cap06")
getwd()
```

```
## [1] "D:/Documentos/FCD/BigDataRAzure/Cap06"
```

```
# Loading the packages
library(plyr)
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
library(ggplot2)
library(gridExtra)
library(ggthemes)
library(caret)
```

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

```
library(MASS)
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'

## The following object is masked from 'package:gridExtra':
##
##     combine

## The following object is masked from 'package:ggplot2':
##
##     margin
```

```
library(party)
```

```
## Loading required package: grid

## Loading required package: mvtnorm

## Loading required package: modeltools

## Loading required package: stats4

##
## Attaching package: 'modeltools'

## The following object is masked from 'package:plyr':
##
##     empty

## Loading required package: strucchange

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: sandwich
```

Step 2 - Loading and Cleaning Data

```
churn <- read.csv('Telco-Customer-Churn.csv')


# Removing all lines with missing values.
sapply(churn, function(x) sum(is.na(x)))
```

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

```
churn <- churn[complete.cases(churn), ]


# Changing "No internet service" to "No" in six columns:
# "OnlineSecurity", "OnlineBackup", "DeviceProtection", "TechSupport", "streamingTV", "streamingMovies"
cols_recode1 <- c(10:15)
for(i in 1:ncol(churn[,cols_recode1])) {
  churn[,cols_recode1][,i] <- as.factor(mapvalues
                                        (churn[,cols_recode1][,i], from =c("No internet service"),to=c(
}


# Changing "No phone service" to "No" on "MultipleLines" column
churn$MultipleLines <- as.factor(mapvalues(churn$MultipleLines,
                                           from=c("No phone service"),
                                           to=c("No")))


# Grouping "tenure" into five categories:
# "0-12 Month", "12-24 Month", "24-48 Month", "48-60 Month", "> 60 Month"
min(churn$tenure); max(churn$tenure)
```

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

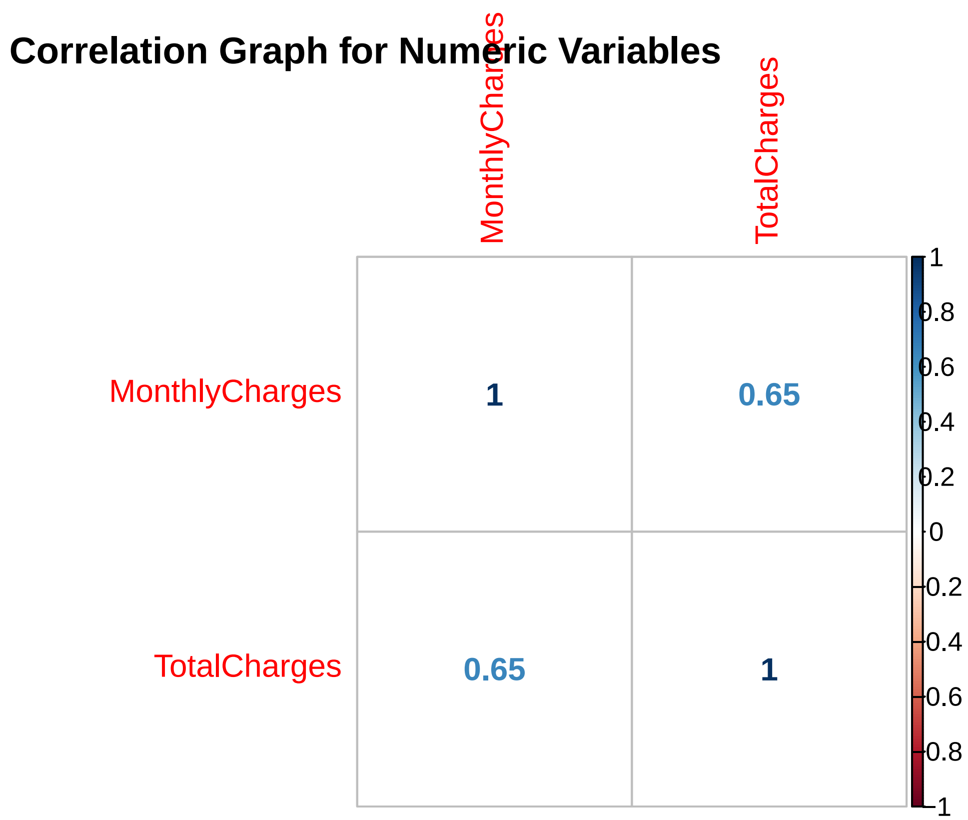
```
## [1] 72
```

```
group_tenure <- function(tenure){
  if (tenure >= 0 & tenure <= 12){
    return('0-12 Month')
  }else if(tenure > 12 & tenure <= 24){
    return('12-24 Month')
  }else if (tenure > 24 & tenure <= 48){
    return('24-48 Month')
  }else if (tenure > 48 & tenure <=60){
    return('48-60 Month')
  }else if (tenure > 60){
    return('> 60 Month')
  }
}
```

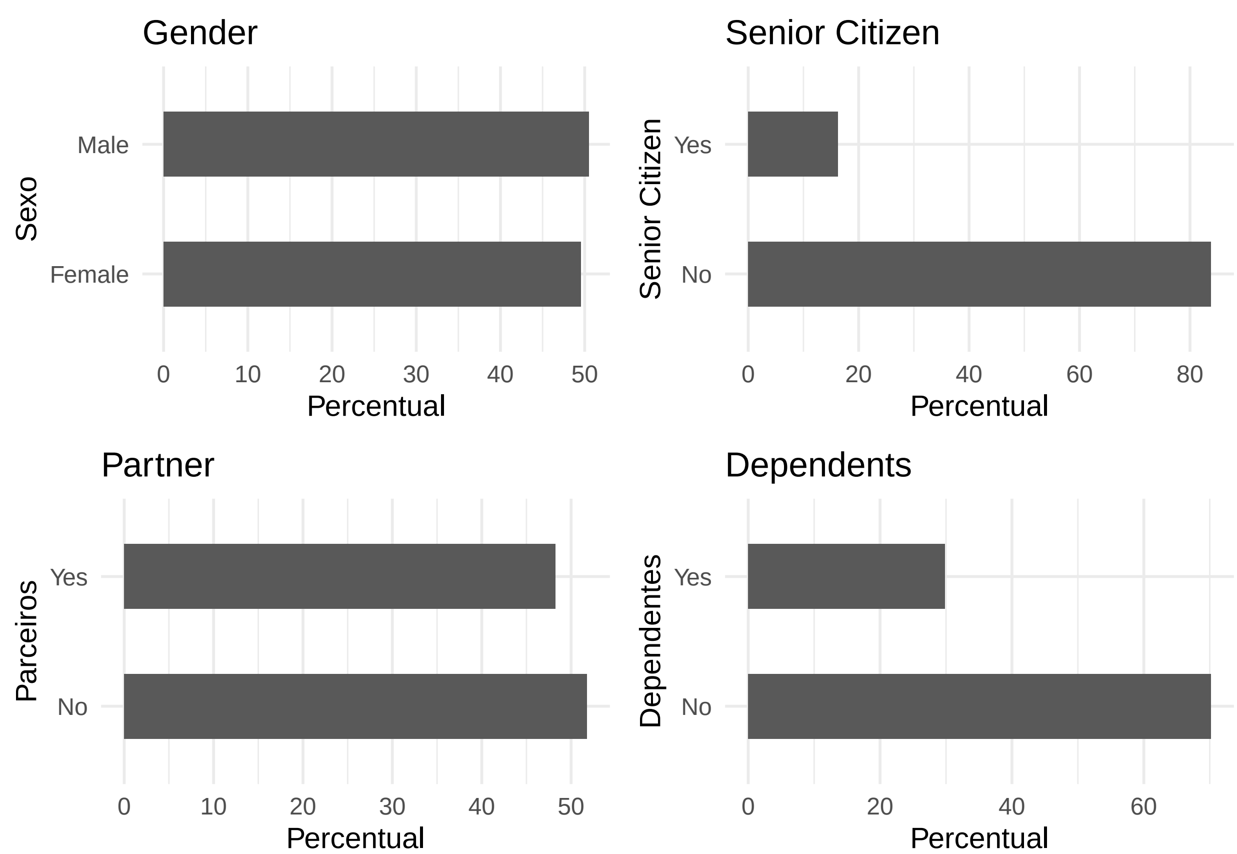
```
churn$tenure_group <- sapply(churn$tenure,group_tenure)
churn$tenure_group <- as.factor(churn$tenure_group)


# Changing "SeniorCitizen" column values from 0 or 1 to "No" or "Yes".
churn$SeniorCitizen <- as.factor(mapvalues(churn$SeniorCitizen,
                                           from=c("0","1"),
                                           to=c("No", "Yes")))



# Removing unnecessary columns for analysis.
churn$customerID <- NULL
churn$tenure <- NULL
```

Step 3 - Exploratory data analysis

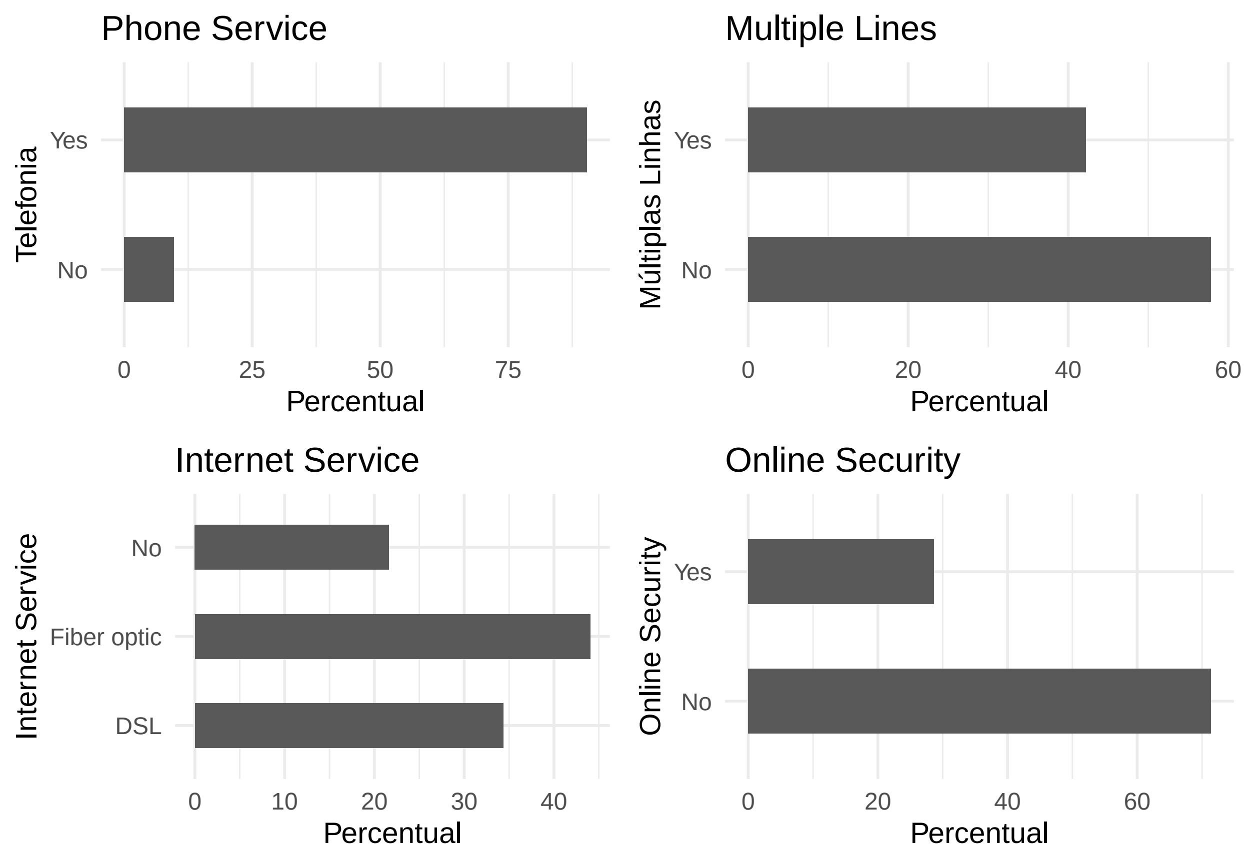
```
# Numeric variables correlation
numeric.var <- sapply(churn, is.numeric)
corr.matrix <- cor(churn[,numeric.var])
corrplot(corr.matrix, main="\n\nCorrelation Graph for Numeric Variables", method="number")
```



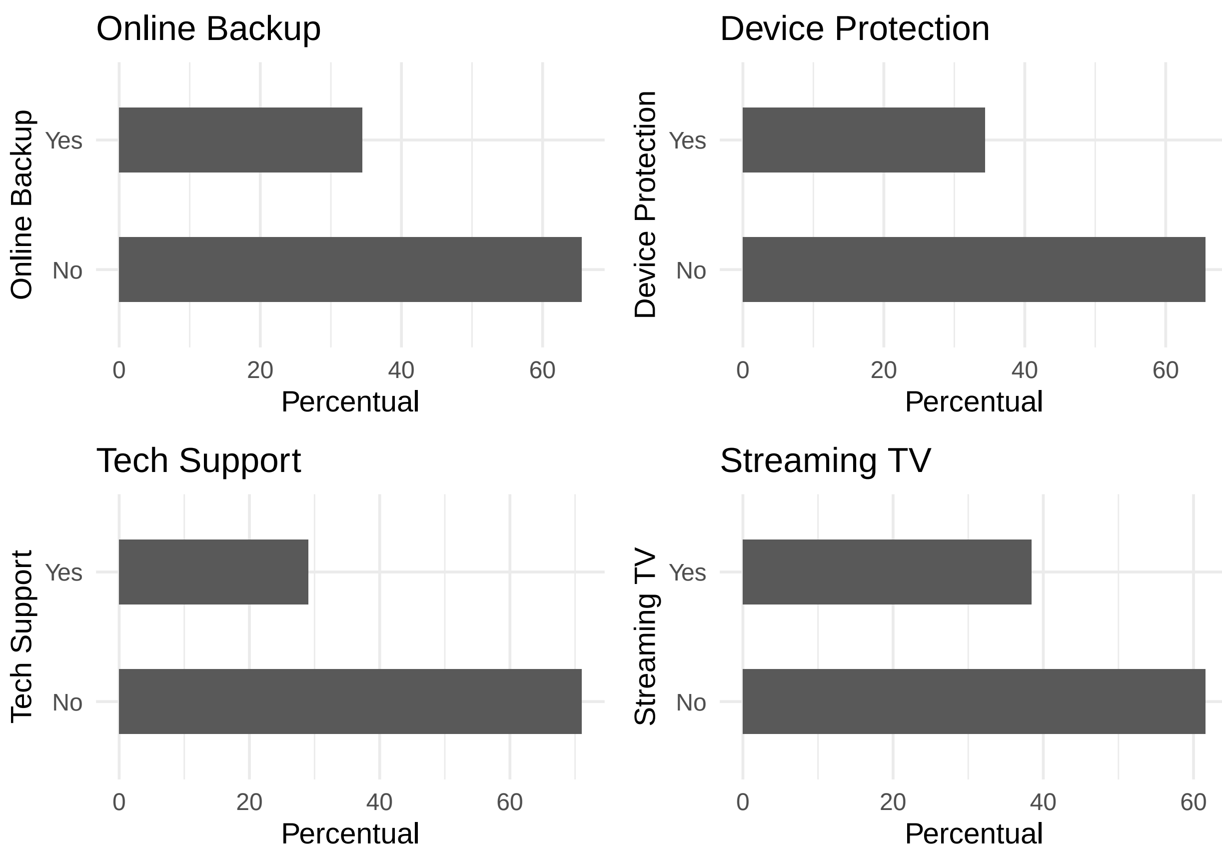
```r
# Removing Total Charges to avoid overfitting
churn$TotalCharges <- NULL


# Categorical variable bar graphs
p1 <- ggplot(churn, aes(x=gender)) + ggtitle("Gender") + xlab("Sexo") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentual") + coord_flip() +
p2 <- ggplot(churn, aes(x=SeniorCitizen)) + ggtitle("Senior Citizen") + xlab("Senior Citizen") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentual") + coord_flip() +
p3 <- ggplot(churn, aes(x=Partner)) + ggtitle("Partner") + xlab("Parceiros") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentual") + coord_flip() +
p4 <- ggplot(churn, aes(x=Dependents)) + ggtitle("Dependents") + xlab("Dependentes") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentual") + coord_flip() +
grid.arrange(p1, p2, p3, p4, ncol=2)
```



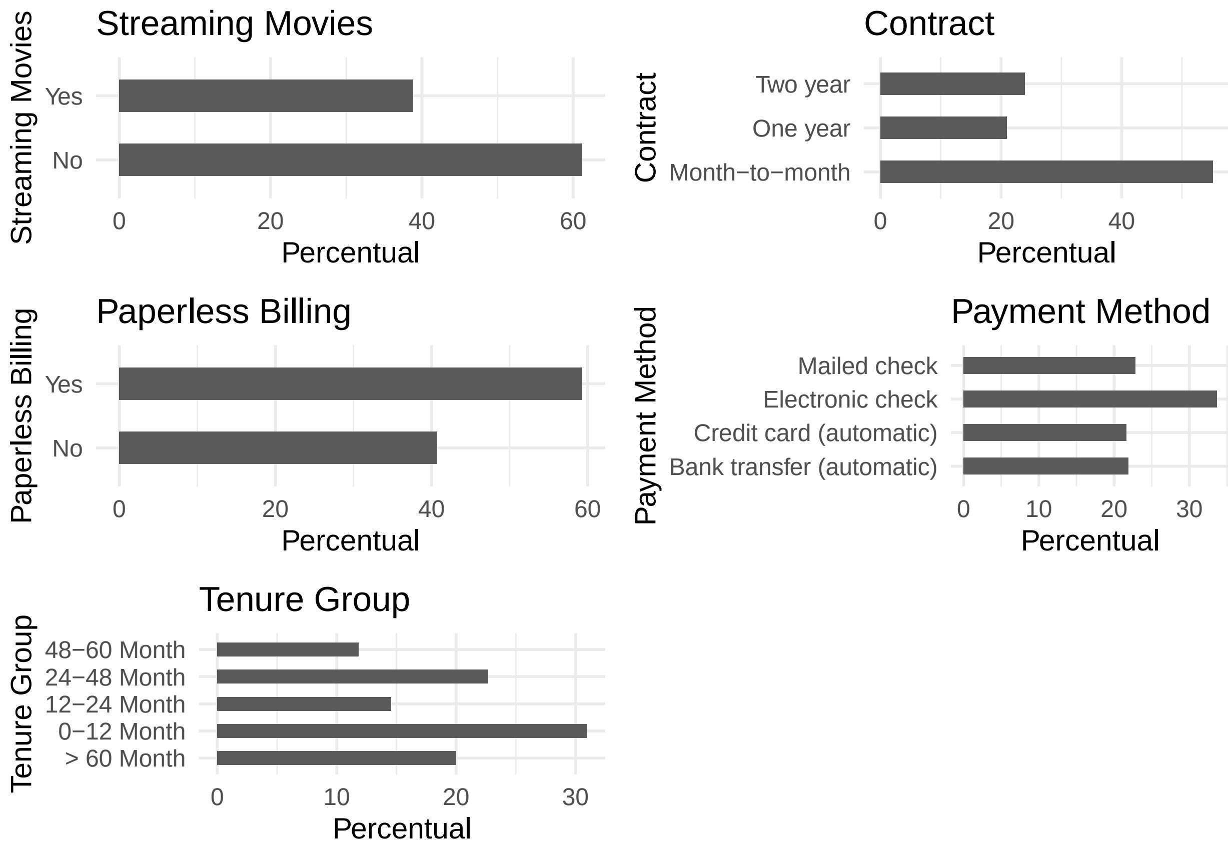

```r
p5 <- ggplot(churn, aes(x=PhoneService)) + ggtitle("Phone Service") + xlab("Telefonia") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentual") + coord_flip() +
p6 <- ggplot(churn, aes(x=MultipleLines)) + ggtitle("Multiple Lines") + xlab("Múltiplas Linhas") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentual") + coord_flip() +
p7 <- ggplot(churn, aes(x=InternetService)) + ggtitle("Internet Service") + xlab("Internet Service") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentual") + coord_flip() +
p8 <- ggplot(churn, aes(x=OnlineSecurity)) + ggtitle("Online Security") + xlab("Online Security") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentual") + coord_flip() +
grid.arrange(p5, p6, p7, p8, ncol=2)
```




```
p9 <- ggplot(churn, aes(x=OnlineBackup)) + ggtitle("Online Backup") + xlab("Online Backup") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentual") + coord_flip() +
p10 <- ggplot(churn, aes(x=DeviceProtection)) + ggtitle("Device Protection") + xlab("Device Protection"
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentual") + coord_flip() +
p11 <- ggplot(churn, aes(x=TechSupport)) + ggtitle("Tech Support") + xlab("Tech Support") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentual") + coord_flip() +
p12 <- ggplot(churn, aes(x=StreamingTV)) + ggtitle("Streaming TV") + xlab("Streaming TV") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentual") + coord_flip() +
grid.arrange(p9, p10, p11, p12, ncol=2)
```



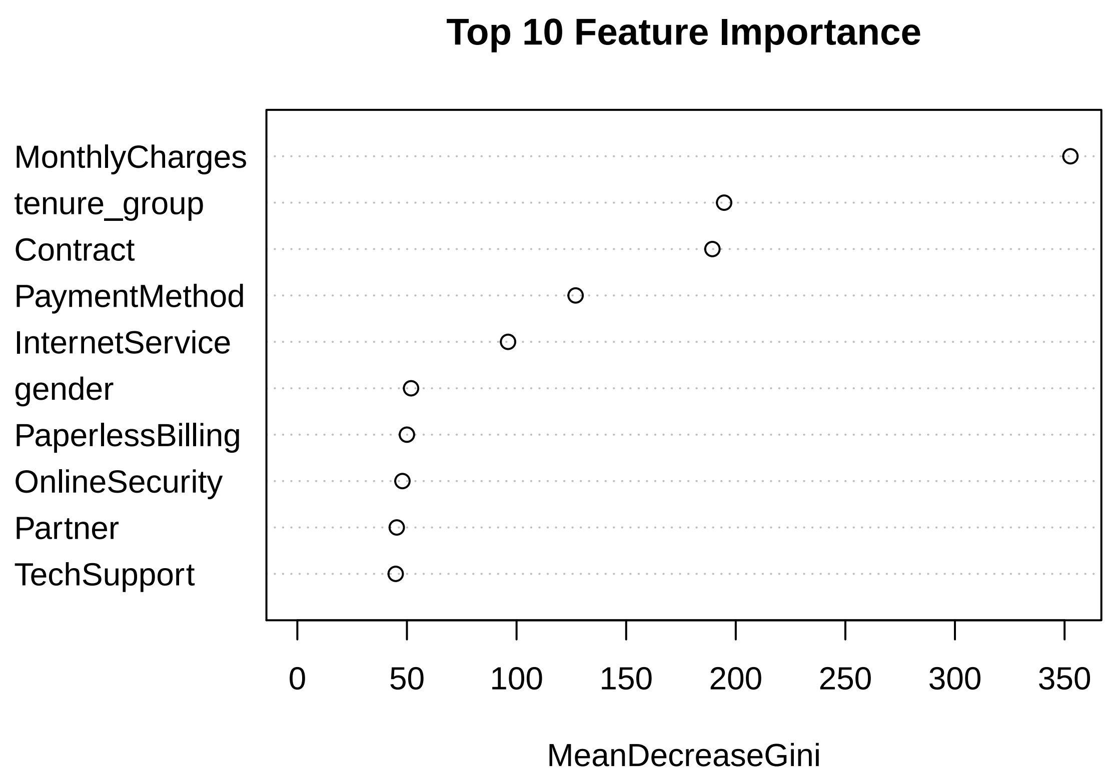

```
p13 <- ggplot(churn, aes(x=StreamingMovies)) + ggtitle("Streaming Movies") + xlab("Streaming Movies") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentual") + coord_flip() +
p14 <- ggplot(churn, aes(x=Contract)) + ggtitle("Contract") + xlab("Contract") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentual") + coord_flip() +
p15 <- ggplot(churn, aes(x=PaperlessBilling)) + ggtitle("Paperless Billing") + xlab("Paperless Billing")
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentual") + coord_flip() +
p16 <- ggplot(churn, aes(x=PaymentMethod)) + ggtitle("Payment Method") + xlab("Payment Method") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentual") + coord_flip() +
p17 <- ggplot(churn, aes(x=tenure_group)) + ggtitle("Tenure Group") + xlab("Tenure Group") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), width = 0.5) + ylab("Percentual") + coord_flip() +
grid.arrange(p13, p14, p15, p16, p17, ncol=2)
```



All categorical variables will be maintained because appear to have a reasonably wide distribution.

Step 4 - Feature Selection with Random Forest

```
# Dividing data into training and test - 70:30 ratio
intrain <- createDataPartition(churn$Churn,p=0.7,list=FALSE)
training <- churn[intrain,]
testing <- churn[-intrain,]

# Feature Selection
rfModel <- randomForest(Churn ~., data = training)
pred_rf <- predict(rfModel, testing)
varImpPlot(rfModel, sort=T, n.var = 10, main = 'Top 10 Feature Importance')
```



Step 5 - Predictive Modeling: Logistic Regression

```r
# Logistic regression model fitting
LogModel <- glm(Churn ~ MonthlyCharges+tenure_group+Contract+PaymentMethod+InternetService, family=binom
print(summary(LogModel))
```

```
##
## Call:
## glm(formula = Churn ~ MonthlyCharges + tenure_group + Contract +
##     PaymentMethod + InternetService, family = binomial(link = "logit"),
##     data = training)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6803  -0.6747  -0.3082   0.7748   3.0948
##
## Coefficients:
##                                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)                          -2.407347   0.304341  -7.910 2.57e-15 ***
## MonthlyCharges                        0.004077   0.003587   1.137 0.255745
## tenure_group0-12 Month                1.760224   0.192781   9.131  < 2e-16 ***
## tenure_group12-24 Month               0.902382   0.194622   4.637 3.54e-06 ***
## tenure_group24-48 Month               0.600870   0.181175   3.317 0.000911 ***
## tenure_group48-60 Month               0.369408   0.198313   1.863 0.062497 .
```

```
## ContractOne year                        -0.986713   0.127329  -7.749 9.24e-15 ***
## ContractTwo year                        -1.933307   0.215977  -8.951  < 2e-16 ***
## PaymentMethodCredit card (automatic) -0.046247   0.132704  -0.348 0.727469
## PaymentMethodElectronic check         0.389502   0.109375   3.561 0.000369 ***
## PaymentMethodMailed check            -0.132750   0.133573  -0.994 0.320304
## InternetServiceFiber optic            0.935872   0.153924   6.080 1.20e-09 ***
## InternetServiceNo                    -0.755452   0.179945  -4.198 2.69e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 5702.8  on 4923  degrees of freedom
## Residual deviance: 4192.9  on 4911  degrees of freedom
## AIC: 4218.9
##
## Number of Fisher Scoring iterations: 6
```

```r
# Accuracy
testing$Churn <- as.character(testing$Churn)
testing$Churn[testing$Churn=="No"] <- "0"
testing$Churn[testing$Churn=="Yes"] <- "1"
fitted.results <- predict(LogModel,newdata=testing,type='response')
fitted.results <- ifelse(fitted.results > 0.5,1,0)
misClasificError <- mean(fitted.results != testing$Churn)
print(paste('Logistic Regression Accuracy',1-misClasificError))
```

```
## [1] "Logistic Regression Accuracy 0.784629981024668"
```

```r
# Logistic Regression Confusion Matrix
print("Logistic Regression - Confusion Matrix"); table(testing$Churn, fitted.results > 0.5)
```

```
## [1] "Logistic Regression - Confusion Matrix"
```

```
##
##     FALSE TRUE
##   0  1429  119
##   1   335  225
```

```r
# Odds Ratio
exp(cbind(OR=coef(LogModel), confint(LogModel)))
```

```
## Waiting for profiling to be done...
```

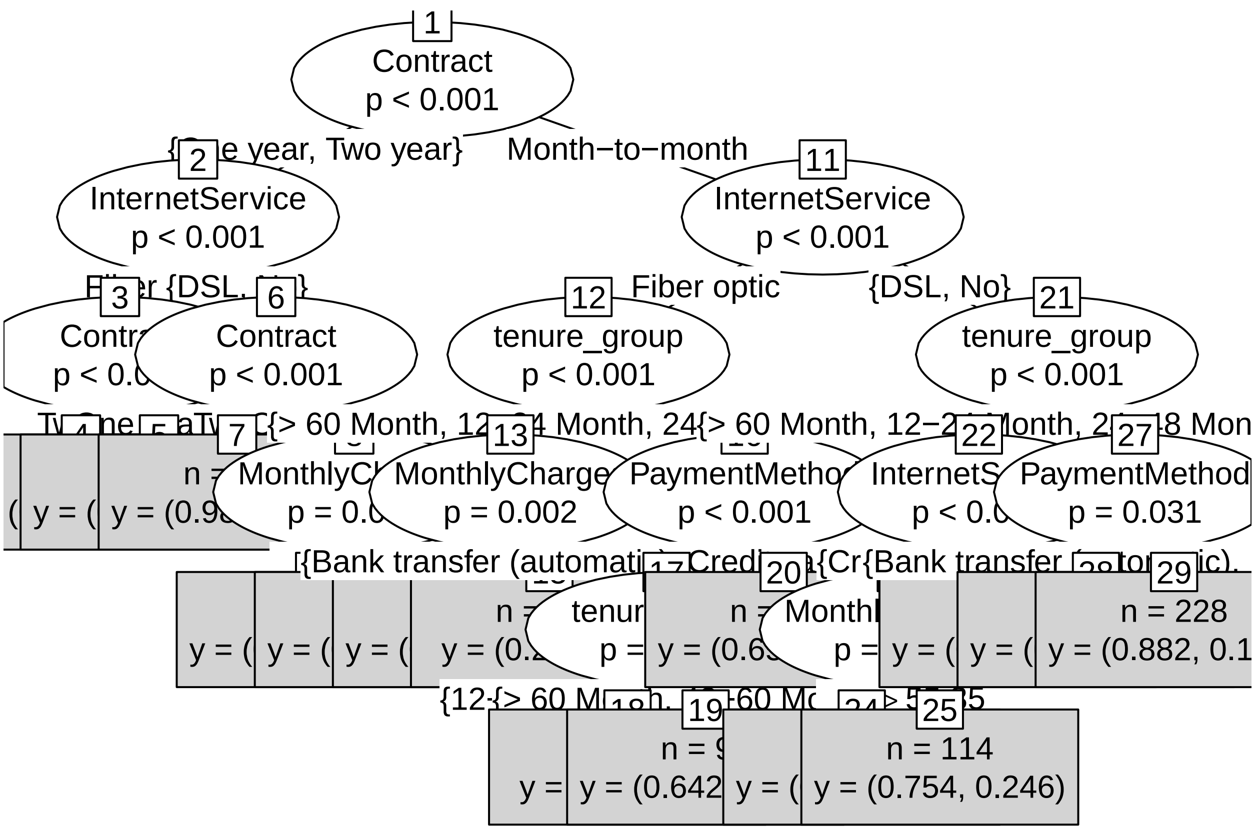
```
##                                          OR      2.5 %    97.5 %
## (Intercept)                      0.09005386 0.04934688 0.1627568
## MonthlyCharges                   1.00408515 0.99706059 1.0111838
## tenure_group0-12 Month           5.81374186 4.00267534 8.5282400
## tenure_group12-24 Month          2.46546899 1.68989604 3.6267653
## tenure_group24-48 Month          1.82370551 1.28383359 2.6141217
## tenure_group48-60 Month          1.44687831 0.98193550 2.1386952
```

```
## ContractOne year                      0.37280004 0.28944123 0.4769661
## ContractTwo year                      0.14466891 0.09329683 0.2179814
## PaymentMethodCredit card (automatic) 0.95480638 0.73586573 1.2383050
## PaymentMethodElectronic check         1.47624588 1.19232973 1.8309412
## PaymentMethodMailed check             0.87568419 0.67408495 1.1381414
## InternetServiceFiber optic            2.54943683 1.88782723 3.4520869
## InternetServiceNo                     0.46979812 0.32945114 0.6673105
```

For each unit increase in the Monthly Charge, there is a 2.5% reduction in the probability of the customer canceling the subscription.

Step 6 - Predictive Modeling: Decision Tree

```r
tree <- ctree(Churn ~ MonthlyCharges+tenure_group+Contract+PaymentMethod+InternetService, training)
plot(tree, type='simple')
```



The Contract is the most important variable for predicting customer churn. If a customer is on a monthly contract, and in the 0 to 12 month tenure group, and using PaperlessBilling, he is more likely to cancel the subscription.

```r
# Decision Tree Confusion Matrix
pred_tree <- predict(tree, testing)
print("Decision Tree Confusion Matrix"); table(Predicted = pred_tree, Actual = testing$Churn)
```

```
## [1] "Decision Tree Confusion Matrix"
```

```
##          Actual
## Predicted    0    1
##       No  1395  294
##       Yes  153  266
```

```
# Accuracy
p1 <- predict(tree, training)
tab1 <- table(Predicted = p1, Actual = training$Churn)
tab2 <- table(Predicted = pred_tree, Actual = testing$Churn)
print(paste('Decision Tree Accuracy',sum(diag(tab2))/sum(tab2)))
```

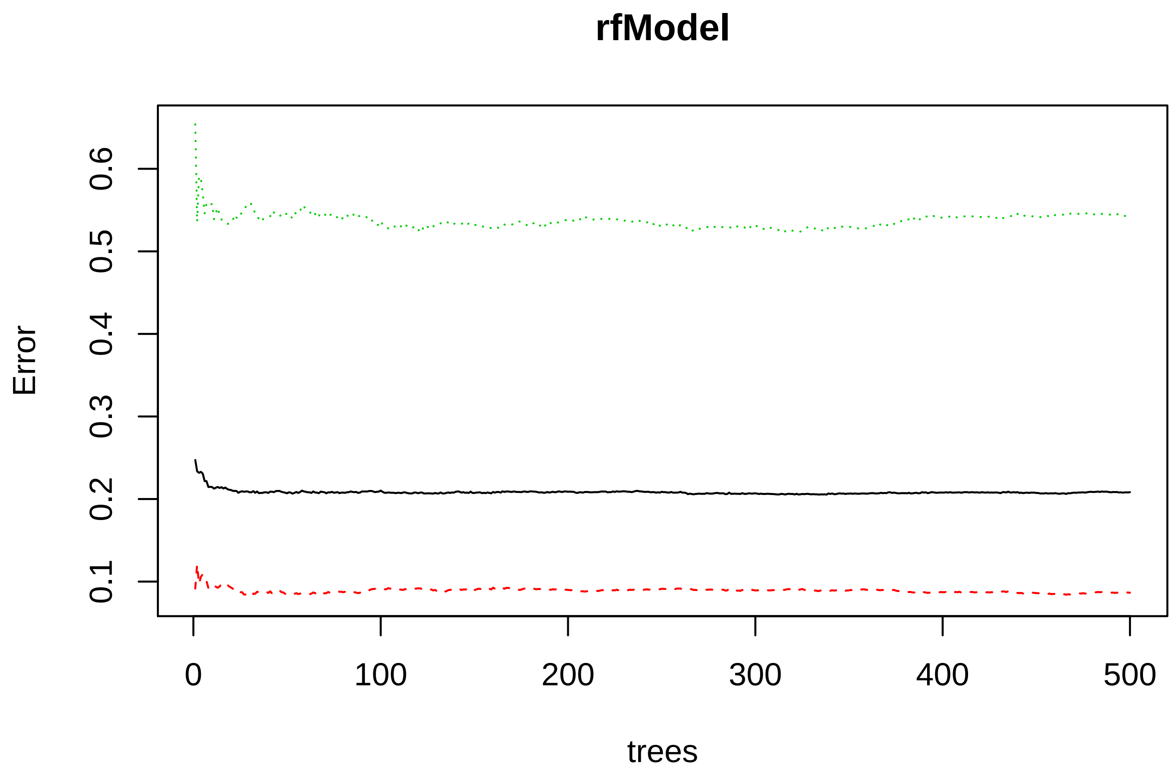
```
## [1] "Decision Tree Accuracy 0.787950664136622"
```

Step 7 - Predictive Modeling: Random Forest

```
rfModel <- randomForest(Churn ~ MonthlyCharges+tenure_group+Contract+PaymentMethod+InternetService, data
print(rfModel)
```

```
##
## Call:
##  randomForest(formula = Churn ~ MonthlyCharges + tenure_group +      Contract + PaymentMethod + Inte
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 20.84%
## Confusion matrix:
##       No Yes class.error
## No  3302 313  0.08658368
## Yes  713 596  0.54469060
```

```
plot(rfModel)
```



The prediction is very good when predicting “No”, but the error rate is much higher when predicting “yes”.

```
# Predicting values with test data
pred_rf <- predict(rfModel, testing)

# Confusion Matrix
print("Random Forest - Confusion Matrix"); table(testing$Churn, pred_rf)
```

```
## [1] "Random Forest - Confusion Matrix"
```

```
##    pred_rf
##       No  Yes
##   0 1423  125
##   1  318  242
```