Project Capstone Churn

Johnny den Braber Lártiga

01-03-2022

INTRODUCTION

On the following document we develop a predictive model for the customer churn on a telecommunication company using different machine learning algorithms. In order to determine if the client is ending or not its contract with the company we first need to understand and characterize the client profile and all of the different services that the company offers to the clients. This project use all of the skills and knowledge that we have acquired through this course.

This report is obtained from an RMarkdown file and has five different sections that show all of the work involved on the development of the different models and the final selection of the most accurate model. Sections 2 and 3 are focused on the preparations and data exploration. Section 4 shows the development of all five models that we tested for this project and Section 5 shows our conclusions and final comments around the project.

We will star this work by uploading the data and the relevant libraries.

DATA WRANGLING

```
library(tidyverse)
library(forcats)
library(stringr)
library(caTools)
library(ggthemes)
library(MASS)
library(party)
```

DATA ASSESSMENT / VISUALIZATIONS

```
library(DT)
library(data.table)
```

```
##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last
```

```
## The following object is masked from 'package:purrr':
##
##     transpose
```

```
library(pander)
library(ggplot2)
library(scales)
```

```
##
## Attaching package: 'scales'
```

```
## The following object is masked from 'package:purrr':
##
##     discard
```

```
## The following object is masked from 'package:readr':
##
##     col_factor
```

```
library(grid)
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(VIM)
```

```
## Loading required package: colorspace
```

```
## VIM is ready to use.
```

```
## Suggestions and bug-reports can be submitted at: https://github.com/statistikat/VIM/issues
```

```
##
## Attaching package: 'VIM'
```

```
## The following object is masked from 'package:datasets':
##
##     sleep
```

```
library(knitr)
library(vcd)
library(caret)
```

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

```
##
## Attaching package: 'caret'
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

MODEL

```
library(xgboost)
```

```
##
## Attaching package: 'xgboost'
```

```
## The following object is masked from 'package:dplyr':
##
##     slice
```

```
library(MLmetrics)
```

```
##
## Attaching package: 'MLmetrics'
```

```
## The following objects are masked from 'package:caret':
##
##     MAE, RMSE
```

```
## The following object is masked from 'package:base':
##
##     Recall
```

```
library(randomForest)
```

```
## randomForest 4.7-1
```

```
## 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:dplyr':
##
##     combine

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

```
library(rpart)
library(rpart.plot)
library(car)
```

```
## Loading required package: carData

##
## Attaching package: 'car'

## The following object is masked from 'package:modeltools':
##
##     Predict

## The following object is masked from 'package:dplyr':
##
##     recode

## The following object is masked from 'package:purrr':
##
##     some
```

```
library(e1071)
library(vcd)
library(ROCR)
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following object is masked from 'package:colorspace':
##
##     coords

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
library(VIM)
library(glmnet)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loaded glmnet 4.1-3
```

```
library(plyr)
```

```
## ------------------------------------------------------------------------------
```

```
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
```

```
## ------------------------------------------------------------------------------
```

```
##
## Attaching package: 'plyr'
```

```
## The following object is masked from 'package:modeltools':
##
##     empty
```

```
## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
## The following object is masked from 'package:purrr':
##
##     compact
```

SECTION II:

DATA PREPARATION

The available data is organized on 7043 different rows, where each one represents each client, and 21 columns that represent each variable associated to the clients. The target variable for these models will be the Churn variable and the explanatory variables will be the remaining 20 variables, which are the following: 'CustomerID', 'Gender', 'SeniorCitizen', 'Partner', 'Dependents', 'Tenure', 'PhoneService', 'MultipleLines, 'InternetService', 'OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV', StreamingMovies', 'Contract', 'PaperlessBilling', 'PaymentMethod', 'MonthlyCharges' and 'TotalCharges'.

First we will identify and treat the missing values with an special interest on the Tenure variable, where the minimum is 1 month and the maximum is 72 months, which were classified on five intervals of 12 moths each. We also changed the Senior Citizen from 0 and 1 to No and Yes. this process was also performed for the Internet Service and Phone Service variables.

The organization and cleaning process is a critical step on data science and the posterior development of models for analysis.

```
churn <- read.csv('C:/Users/brabe/Documents/R_Edx_Capstone_A/churn_jden/TelcoCustomerChurn.csv')
```

```
dim(churn)
```

```
## [1] 7043   21
```

```
str(churn)
```

```
## '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" ...
```

```
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), ]
```

```
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")
}
```

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

```
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)
```

```
churn$SeniorCitizen <- as.factor(mapvalues(churn$SeniorCitizen,
                                           from=c("0","1"),
                                           to=c("No", "Yes")))
```

```
churn$customerID <- NULL
churn$tenure <- NULL
```

Exploratory data analysis and feature selection

Table 1: Correlation Plot for Numeric Variables

	MonthlyCharges	TotalCharges
MonthlyCharges	1.00	0.65
TotalCharges	0.65	1.00

```
churn$TotalCharges <- NULL
```

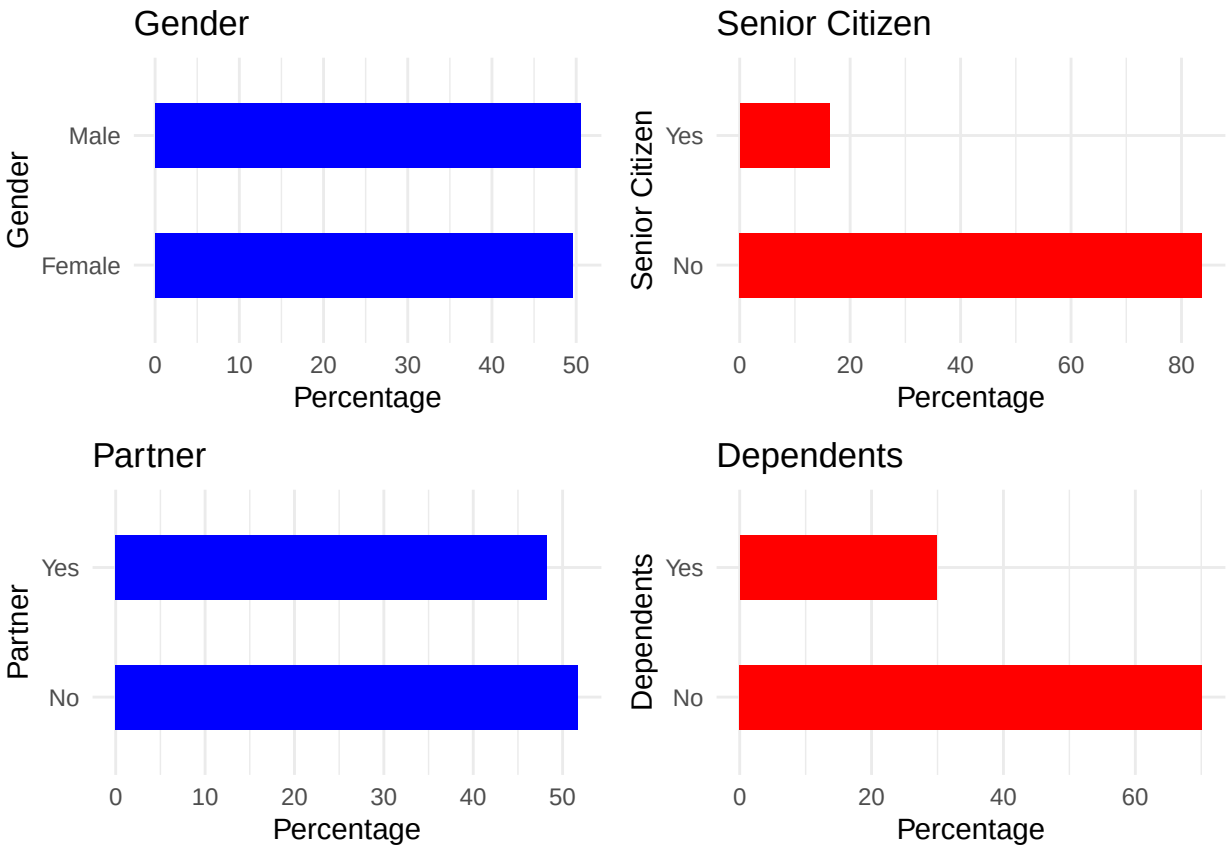
SECTION III

DATA VISUALIZATION

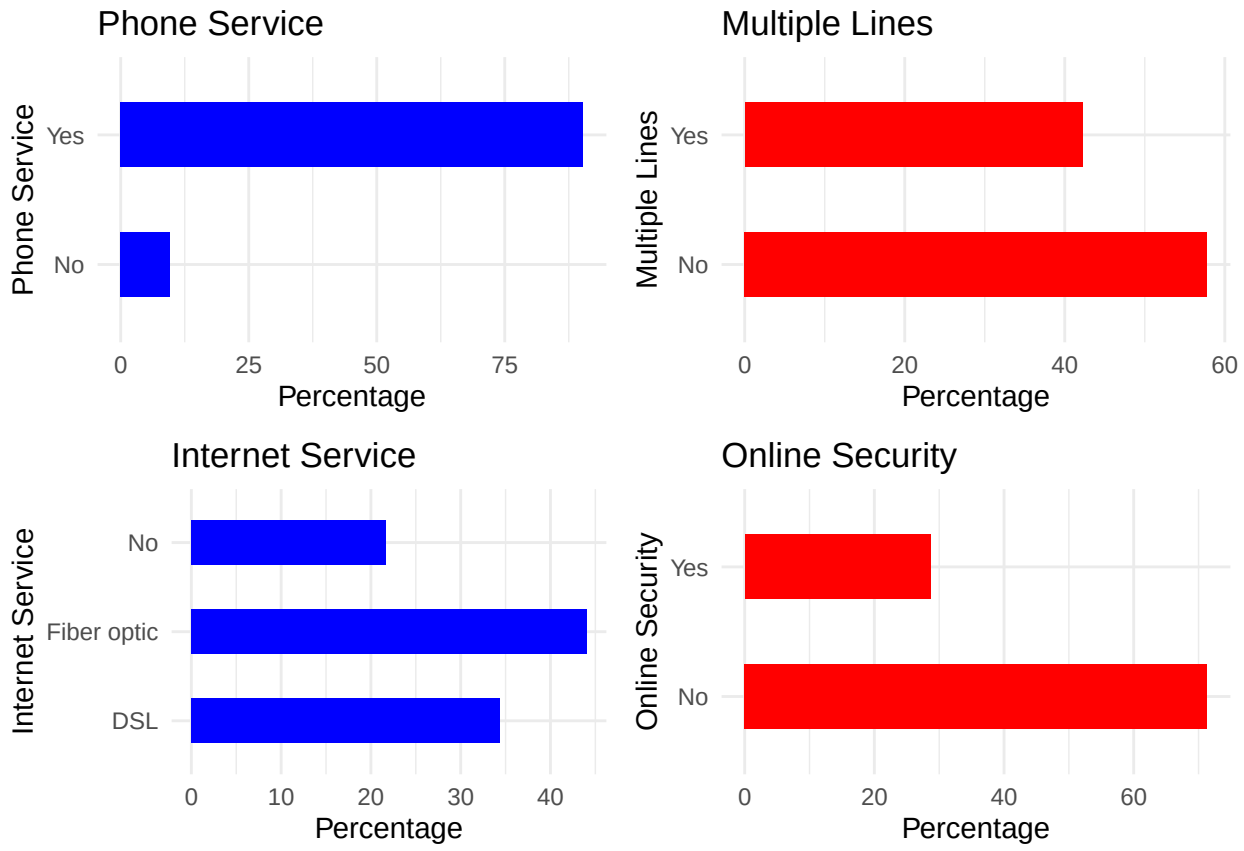
Here we add a set of graphs to get a global overview of the behavior and distribution of the variables that we will use to develop our models, such as Gender, Contract and Tenure Group.

Bar plots of categorical variables.

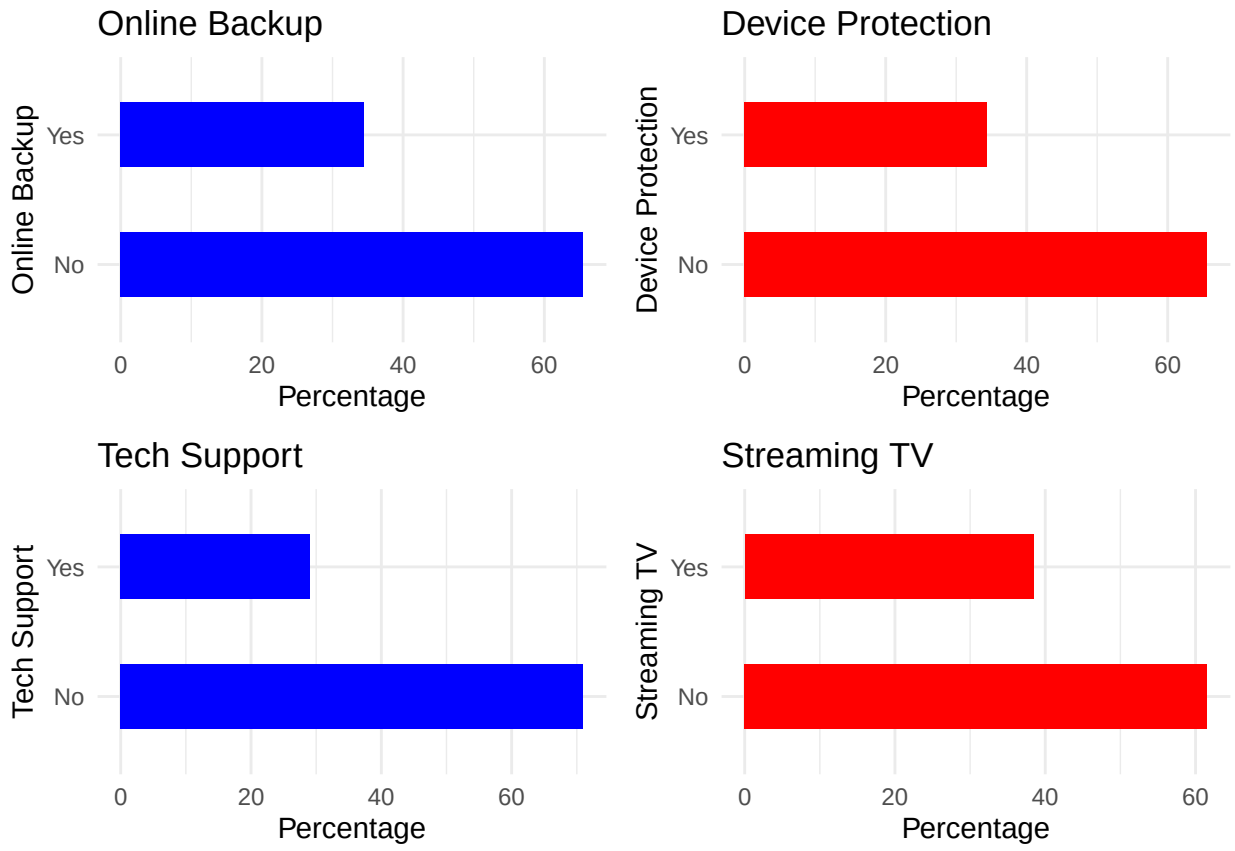
```
graf_1 <- ggplot(churn, aes(x=gender)) +
  ggtitle("Gender") + xlab("Gender") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), fill = 'blue' ,width = 0.5) +
  ylab("Percentage") +
  coord_flip() +
  theme_minimal()
graf_2 <- ggplot(churn, aes(x=SeniorCitizen)) +
  ggtitle("Senior Citizen") + xlab("Senior Citizen") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), fill = 'red' ,width = 0.5) +
  ylab("Percentage") + coord_flip() +
  theme_minimal()
graf_3 <- ggplot(churn, aes(x=Partner)) +
  ggtitle("Partner") + xlab("Partner") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), fill = 'blue' ,width = 0.5) +
  ylab("Percentage") + coord_flip() +
  theme_minimal()
graf_4 <- ggplot(churn, aes(x=Dependents)) +
  ggtitle("Dependents") + xlab("Dependents") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), fill = 'red' ,width = 0.5) +
  ylab("Percentage") + coord_flip() +
  theme_minimal()
grid.arrange(graf_1, graf_2, graf_3, graf_4, ncol=2)
```


Gender
Male
Female
Gender
0
10
20
30
40
50
Percentage
Senior Citizen
Yes
No
Senior Citizen
0
20
40
60
80
Percentage
Partner
Yes
No
Partner
0
10
20
30
40
50
Percentage
Dependents
Yes
No
Dependents
0
20
40
60
Percentage

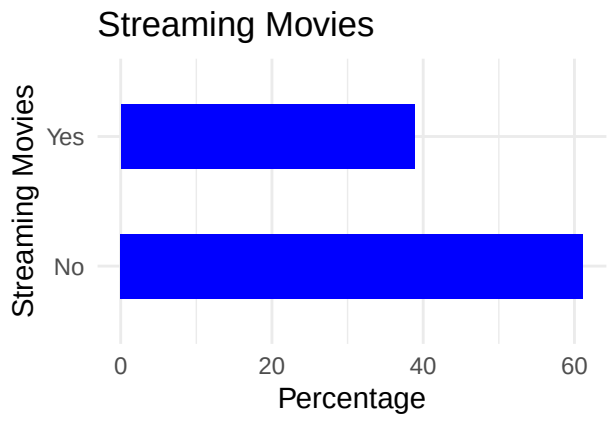
```
graf_5 <- ggplot(churn, aes(x=PhoneService)) +
  ggtitle("Phone Service") + xlab("Phone Service") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), fill = 'blue' ,width = 0.5) +
  ylab("Percentage") + coord_flip() +
  theme_minimal()
graf_6 <- ggplot(churn, aes(x=MultipleLines)) +
  ggtitle("Multiple Lines") + xlab("Multiple Lines") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), fill = 'red' ,width = 0.5) +
  ylab("Percentage") + coord_flip() +
  theme_minimal()
graf_7 <- ggplot(churn, aes(x=InternetService)) + ggtitle("Internet Service") +
  xlab("Internet Service") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), fill = 'blue' ,width = 0.5) +
  ylab("Percentage") + coord_flip() +
  theme_minimal()
graf_8 <- ggplot(churn, aes(x=OnlineSecurity)) + ggtitle("Online Security") +
  xlab("Online Security") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), fill = 'red' ,width = 0.5) +
  ylab("Percentage") + coord_flip() +
  theme_minimal()
grid.arrange(graf_5, graf_6, graf_7, graf_8, ncol=2)
```

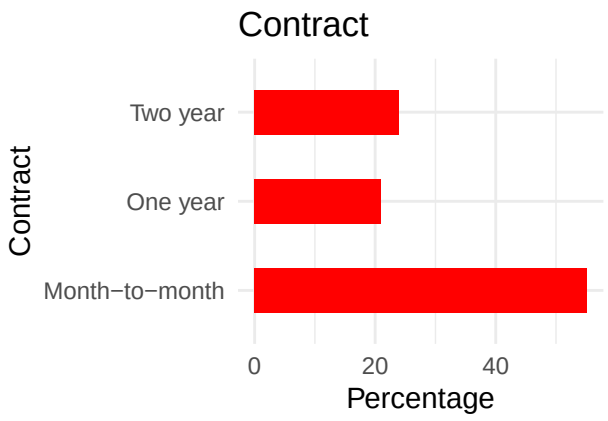


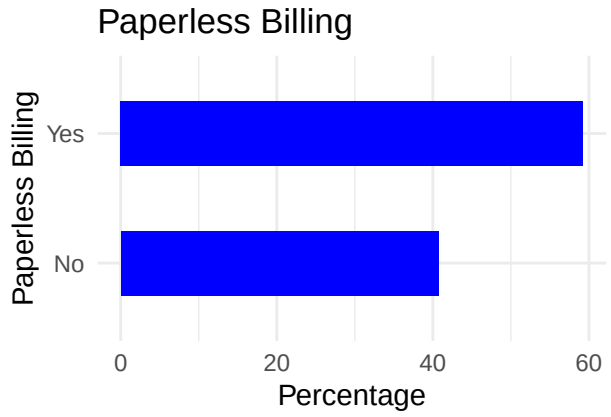
```
graf_9 <- ggplot(churn, aes(x=OnlineBackup)) +
  ggtitle("Online Backup") + xlab("Online Backup") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), fill = 'blue' ,width = 0.5) +
  ylab("Percentage") + coord_flip() +
  theme_minimal()
graf_10 <- ggplot(churn, aes(x=DeviceProtection)) +
  ggtitle("Device Protection") + xlab("Device Protection") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), fill = 'red' ,width = 0.5) +
  ylab("Percentage") + coord_flip() +
  theme_minimal()
graf_11 <- ggplot(churn, aes(x=TechSupport)) +
  ggtitle("Tech Support") + xlab("Tech Support") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), fill = 'blue'  ,width = 0.5) +
  ylab("Percentage") + coord_flip() +
  theme_minimal()
graf_12 <- ggplot(churn, aes(x=StreamingTV)) +
  ggtitle("Streaming TV") + xlab("Streaming TV") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), fill = 'red'  ,width = 0.5) +
  ylab("Percentage") + coord_flip() +
  theme_minimal()
grid.arrange(graf_9, graf_10, graf_11, graf_12, ncol=2)
```



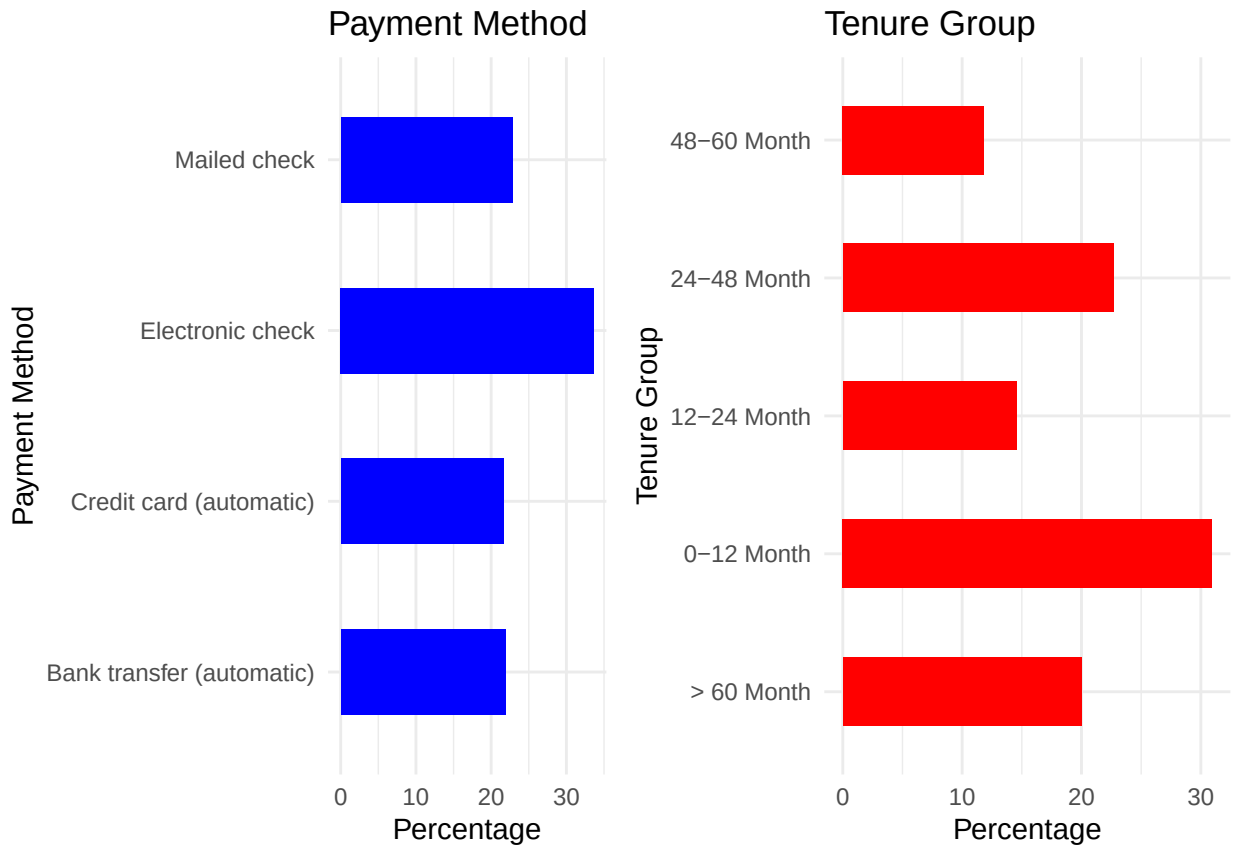
```
graf_13 <- ggplot(churn, aes(x=StreamingMovies)) +
  ggtitle("Streaming Movies") + xlab("Streaming Movies") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), fill = 'blue' ,width = 0.5) +
  ylab("Percentage") + coord_flip() +
  theme_minimal()
graf_14 <- ggplot(churn, aes(x=Contract)) +
  ggtitle("Contract") + xlab("Contract") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), fill = 'red' ,width = 0.5) +
  ylab("Percentage") + coord_flip() +
  theme_minimal()
graf_15 <- ggplot(churn, aes(x=PaperlessBilling)) +
  ggtitle("Paperless Billing") + xlab("Paperless Billing") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), fill = 'blue' ,width = 0.5) +
  ylab("Percentage") + coord_flip() +
  theme_minimal()
grid.arrange(graf_13, graf_14, graf_15, ncol=2)
```







```
graf_16 <- ggplot(churn, aes(x=PaymentMethod)) +
  ggtitle("Payment Method") + xlab("Payment Method") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), fill = 'blue' ,width = 0.5) +
  ylab("Percentage") + coord_flip() +
  theme_minimal()
graf_17 <- ggplot(churn, aes(x=tenure_group)) +
  ggtitle("Tenure Group") + xlab("Tenure Group") +
  geom_bar(aes(y = 100*(..count..)/sum(..count..)), fill = 'red' ,width = 0.5) +
  ylab("Percentage") + coord_flip() +
  theme_minimal()
grid.arrange(graf_16, graf_17, ncol=2)
```



```
library(dplyr)
library(tidyverse)
```

We show a table for the main variable.

```
tabla <- churn %>%
  dplyr::group_by(Churn) %>%
  dplyr::summarize(count=n())
kable(tabla, caption = 'Observation to Churn', aling = 'cc')
```

Table 2: Observation to Churn

Churn	count
No	5163
Yes	1869

SECTION IV

MODELS

Training Set - In machine learning, a training set is a dataset used to train a model. In training the model, specific features are picked out from the training set. These features are then incorporated into the model.

Test Set - The test set is a dataset used to measure how well the model performs at making predictions on that test set.

Section 4 show the different models that we used to obtain the best possible prediction. The models used are the following: On first place we start using a decision tree model, next we used a random forest model, logistic regression model and support vector machine model, to finish with a radial support vector machine model.

```
feauter_1<-churn[1:7032, c("gender", "InternetService","PhoneService","Contract","PaymentMethod","tenu

response <- as.factor(churn$Churn)

feauter_1$Churn=as.factor(churn$Churn)
```

Verifying data.

```
set.seed(500)
ind=createDataPartition(feauter_1$Churn,times=1,p=0.8,list=FALSE)
train_val=feauter_1[ind,]
test_val=feauter_1[-ind,]
```

```
round(prop.table(table(churn$Churn)*100),digits = 1)
```

Here we check the Churn rate in the orginal training data, current training data and test data.

```
##
##  No Yes
## 0.7 0.3
```

```
round(prop.table(table(train_val$Churn)*100),digits = 1)
```

```
##
##  No Yes
## 0.7 0.3
```

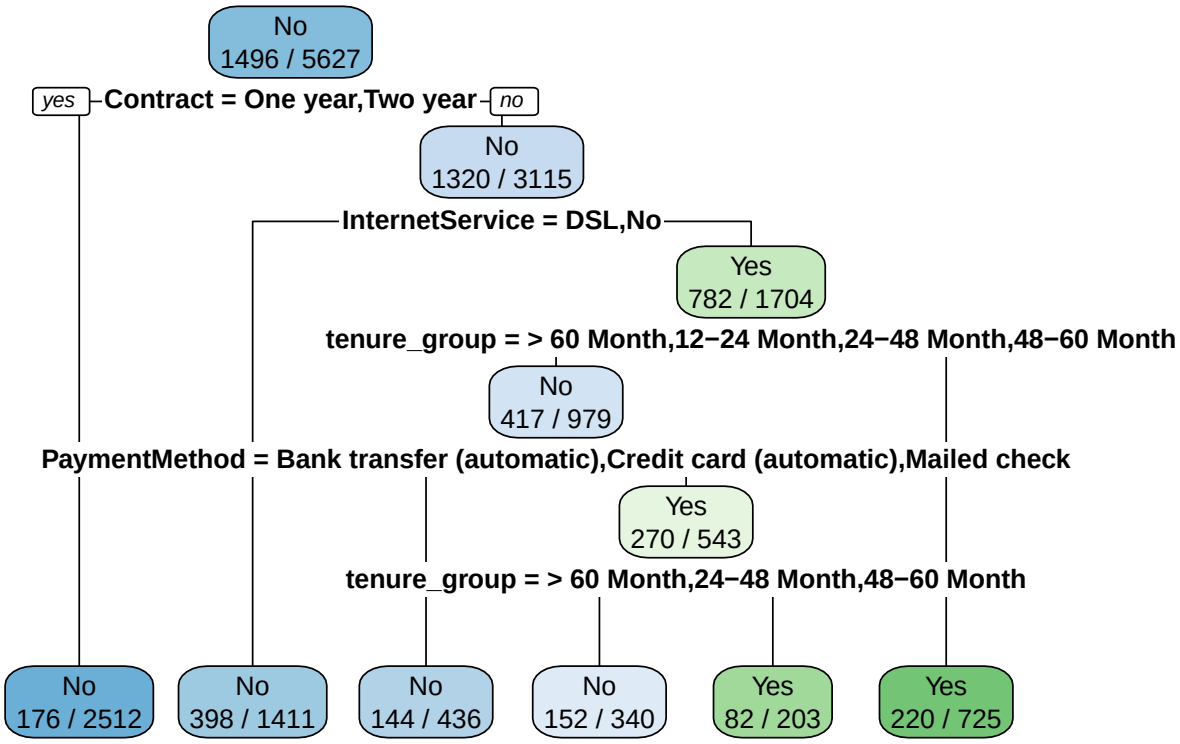
```
round(prop.table(table(test_val$Churn)*100),digits = 1)
```

```
##
##  No Yes
## 0.7 0.3
```

MODEL DECISION TREE (Predictive Analysis and Cross Validation {.tabset})

Model Decision Tree

```
set.seed(1234)
Model_DT=rpart(Churn~.,data=train_val,method="class")
rpart.plot(Model_DT,extra =  3,fallen.leaves = T)
```




```
PRE_TDT=predict(Model_DT,data=train_val,type="class")
confusionMatrix(PRE_TDT,train_val$Churn)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   No  Yes
##        No  3829  870
##        Yes  302  626
##
##                Accuracy : 0.7917
##                  95% CI : (0.7809, 0.8023)
##     No Information Rate : 0.7341
```

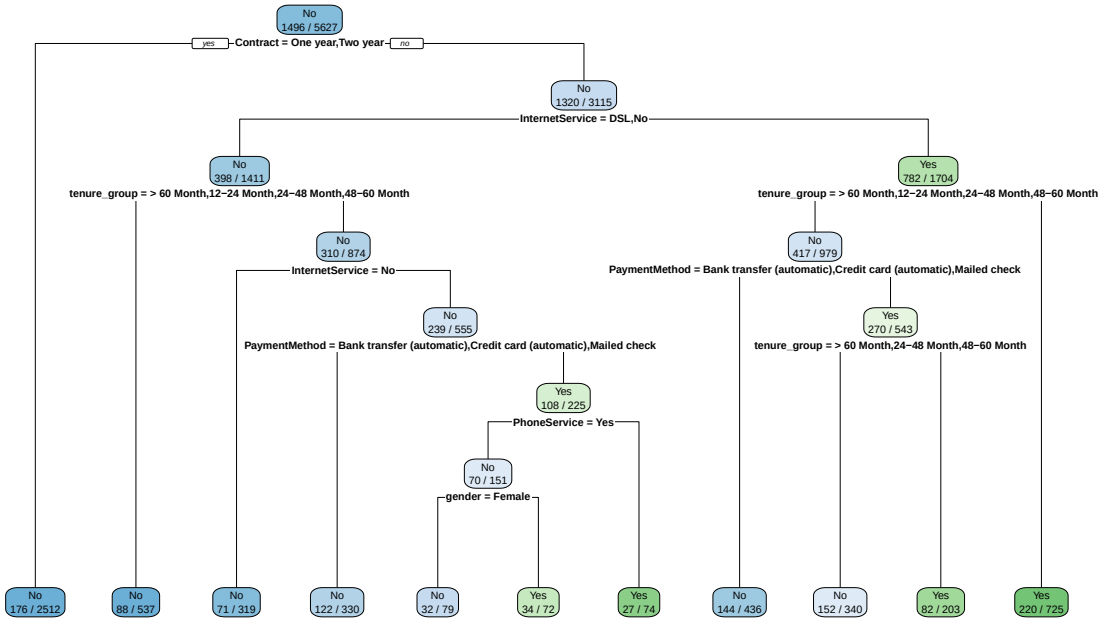
```
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.3929
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9269
##             Specificity : 0.4184
##          Pos Pred Value : 0.8149
##          Neg Pred Value : 0.6746
##              Prevalence : 0.7341
##          Detection Rate : 0.6805
##    Detection Prevalence : 0.8351
##       Balanced Accuracy : 0.6727
##
##        'Positive' Class : No
##
```

```
set.seed(1234)
cv.10 <- createMultiFolds(train_val$Churn, k = 10, times = 10)


### Control
ctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 10,
                     index = cv.10)
train_val <- as.data.frame(train_val)


###Train the data
Model_CDT <- train(x = train_val[,-7], y = train_val[,7], method = "rpart", tuneLength = 30,
                   trControl = ctrl)


rpart.plot(Model_CDT$finalModel,extra =  3,fallen.leaves = T)
```




```
PRE_VDTS=predict(Model_CDT$finalModel,newdata=test_val,type="class")
confusionMatrix(PRE_VDTS,test_val$Churn)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  937 186
##        Yes  95 187
##
##                Accuracy : 0.8
##                  95% CI : (0.7781, 0.8206)
##     No Information Rate : 0.7345
##     P-Value [Acc > NIR] : 6.374e-09
##
##                   Kappa : 0.4439
##
##  Mcnemar's Test P-Value : 7.920e-08
##
##             Sensitivity : 0.9079
##             Specificity : 0.5013
##          Pos Pred Value : 0.8344
##          Neg Pred Value : 0.6631
##              Prevalence : 0.7345
##          Detection Rate : 0.6669
```

```
##    Detection Prevalence : 0.7993
##       Balanced Accuracy : 0.7046
##
##        'Positive' Class : No
##
```

```
col_names <- names(train_val)
train_val[col_names] <- lapply(train_val[col_names] , factor)
test_val[col_names] <- lapply(test_val[col_names] , factor)
```

RANDOM FOREST MODEL

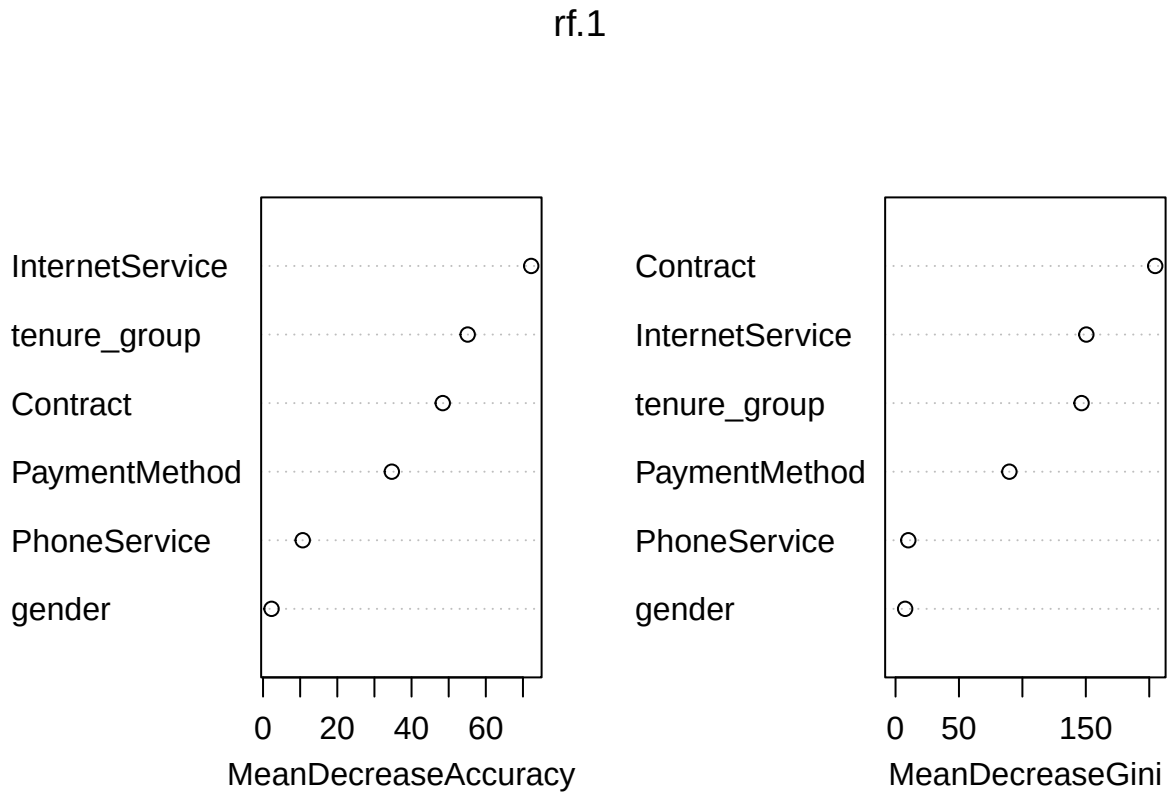
```
set.seed(1234)

rf.1 <- randomForest(x = train_val[,-7],y=train_val[,7], importance = TRUE, ntree = 1000)
rf.1
```

Random Forest

```
##
## Call:
##  randomForest(x = train_val[, -7], y = train_val[, 7], ntree = 1000,      importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 1000
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 20.51%
## Confusion matrix:
##       No Yes class.error
## No  3802 329  0.07964173
## Yes  825 671  0.55147059
```

```
varImpPlot(rf.1)
```

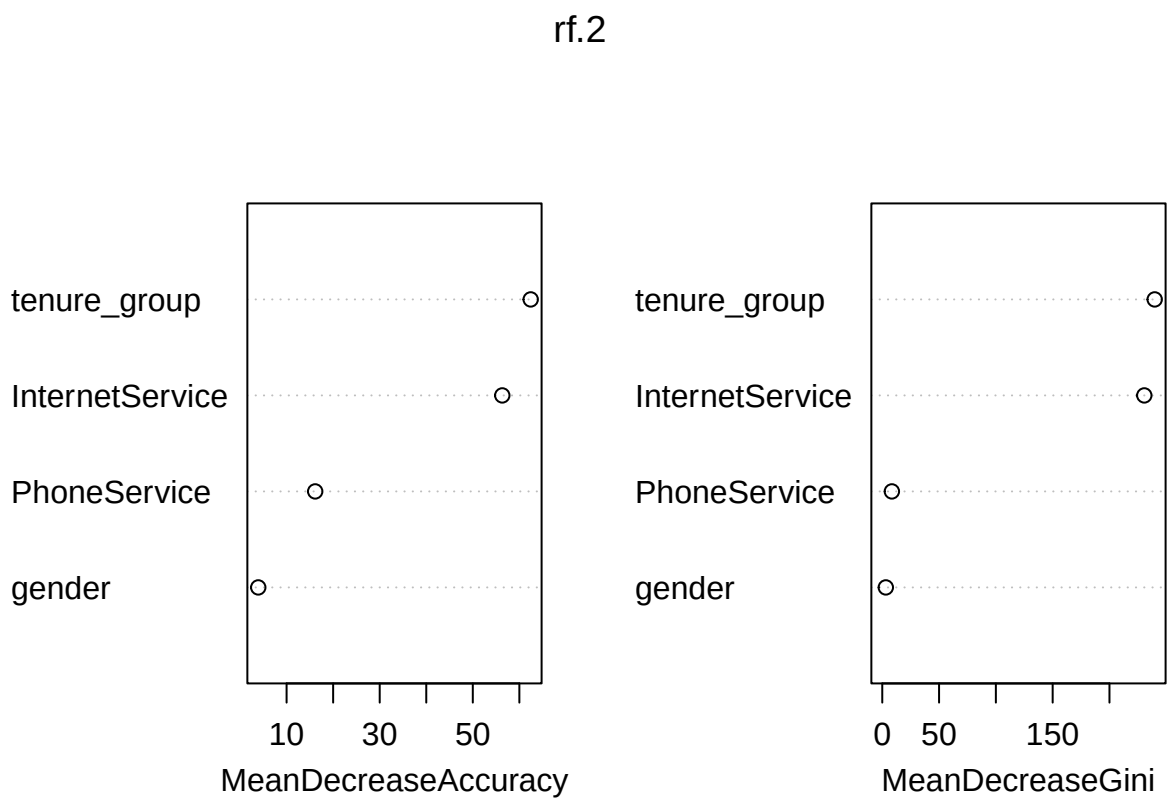


```
train_val1=train_val[,-4:-5]
test_val1=test_val[,-4:-5]


set.seed(1234)
rf.2 <- randomForest(x = train_val1[,-5],y=train_val1[,5], importance = TRUE, ntree = 1000)
rf.2
```

```
##
## Call:
##  randomForest(x = train_val1[, -5], y = train_val1[, 5], ntree = 1000,      importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 1000
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 21.59%
## Confusion matrix:
##       No Yes class.error
## No  3906 225  0.05446623
## Yes  990 506  0.66176471
```

```
varImpPlot(rf.2)
```



```
set.seed(2348)
cv10_1 <- createMultiFolds(train_val1[,5], k = 10, times = 10)


ctrl_1 <- trainControl(method = "repeatedcv", number = 10, repeats = 10,
                       index = cv10_1)


set.seed(1234)
rf.5<- train(x = train_val1[,-5], y = train_val1[,5], method = "rf", tuneLength = 3,
               ntree = 1000, trControl =ctrl_1)
rf.5
```

```
## Random Forest
##
## 5627 samples
##    4 predictor
##    2 classes: 'No', 'Yes'
##
## No pre-processing
## Resampling: Cross-Validated (10 fold, repeated 10 times)
## Summary of sample sizes: 5065, 5064, 5064, 5064, 5064, 5064, ...
## Resampling results across tuning parameters:
##
##   mtry  Accuracy   Kappa
##   2     0.7829790  0.3384866
##   3     0.7817177  0.3619847
##   4     0.7826776  0.3692586
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 2.
```

```
pr.rf=predict(rf.5,newdata = test_val1)
confusionMatrix(pr.rf,test_val1$Churn)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  979 234
##        Yes  53 139
##
##                Accuracy : 0.7957
##                  95% CI : (0.7737, 0.8165)
##     No Information Rate : 0.7345
##     P-Value [Acc > NIR] : 5.654e-08
##
##                   Kappa : 0.3802
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9486
##             Specificity : 0.3727
```

```
##          Pos Pred Value : 0.8071
##          Neg Pred Value : 0.7240
##              Prevalence : 0.7345
##          Detection Rate : 0.6968
##    Detection Prevalence : 0.8633
##       Balanced Accuracy : 0.6606
##
##        'Positive' Class : No
##
```

LOGISTIC REGRESSION MODEL

```
contrasts(train_val$gender)
```

Logistic Regression

```
##        Male
## Female    0
## Male      1
```

```
contrasts(train_val$Contract)
```

```
##                 One year Two year
## Month-to-month         0        0
## One year               1        0
## Two year               0        1
```

```
log.mod <- glm(Churn ~ ., family = binomial(link=logit),
             data = train_val1)


summary(log.mod)
```

```
##
## Call:
## glm(formula = Churn ~ ., family = binomial(link = logit), data = train_val1)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.5984  -0.7468  -0.3666   0.8084   3.0493
##
## Coefficients:
##                              Estimate Std. Error z value Pr(>|z|)
## (Intercept)                 -3.01891    0.16635 -18.148  < 2e-16 ***
## genderMale                   0.01500    0.07013   0.214    0.831
## InternetServiceFiber optic   1.54724    0.09233  16.758  < 2e-16 ***
## InternetServiceNo           -1.08641    0.13537  -8.025 1.01e-15 ***
## PhoneServiceYes             -0.53425    0.13147  -4.064 4.83e-05 ***
## tenure_group0-12 Month       2.94166    0.13666  21.526  < 2e-16 ***
## tenure_group12-24 Month      1.92248    0.14853  12.943  < 2e-16 ***
## tenure_group24-48 Month      1.37384    0.14322   9.593  < 2e-16 ***
## tenure_group48-60 Month      0.87160    0.16799   5.188 2.12e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6517.2  on 5626  degrees of freedom
## Residual deviance: 5010.6  on 5618  degrees of freedom
## AIC: 5028.6
##
## Number of Fisher Scoring iterations: 5
```

```
confint(log.mod)
```

```
##                                   2.5 %     97.5 %
## (Intercept)                 -3.3510774 -2.6986272
## genderMale                  -0.1224459  0.1525250
## InternetServiceFiber optic   1.3680249  1.7300538
## InternetServiceNo           -1.3554410 -0.8242890
## PhoneServiceYes             -0.7912375 -0.2755948
## tenure_group0-12 Month       2.6800256  3.2163219
## tenure_group12-24 Month      1.6360457  2.2189286
## tenure_group24-48 Month      1.0980843  1.6601847
## tenure_group48-60 Month      0.5436570  1.2031041
```

```
train.probs <- predict(log.mod, data=train_val1,type =  "response")#
table(train_val1$Churn,train.probs>0.5)
```

```
##
##       FALSE TRUE
##   No   3906  225
##   Yes   990  506
```

```
test.probs <- predict(log.mod, newdata=test_val1,type =  "response")
table(test_val1$Churn,test.probs>0.5)
```

```
##
##       FALSE TRUE
##   No    979   53
##   Yes   234  139
```

SUPPORT VECTOR MACHINE MODEL

```
set.seed(1274)

liner.tune=tune.svm(Churn~.,data=train_val,kernel="linear",cost=c(0.01,0.1,0.2,0.5,0.7,1,2,3,5,10,15,20
liner.tune
```

Linear Support vector Machine

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##  0.01
##
## - best performance: 0.2130803
```

```
best.linear=liner.tune$best.model

best.test=predict(best.linear,newdata=test_val,type="class")
confusionMatrix(best.test,test_val$Churn)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  924 181
##        Yes 108 192
##
##                Accuracy : 0.7943
##                  95% CI : (0.7722, 0.8152)
##     No Information Rate : 0.7345
##     P-Value [Acc > NIR] : 1.131e-07
##
##                   Kappa : 0.4374
##
##  Mcnemar's Test P-Value : 2.283e-05
##
##             Sensitivity : 0.8953
##             Specificity : 0.5147
##          Pos Pred Value : 0.8362
##          Neg Pred Value : 0.6400
##              Prevalence : 0.7345
##          Detection Rate : 0.6577
##    Detection Prevalence : 0.7865
##       Balanced Accuracy : 0.7050
##
##        'Positive' Class : No
##
```

RADIAL SUPPORT VECTOR MACHINE MODEL: (Non Linear Kernel give us a better accuracy)

```
set.seed(1274)
rd.poly=tune.svm(Churn~.,data=train_val1,kernel="radial",gamma=seq(0.1,5))

summary(rd.poly)
```

bRadial Support vector Machine

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  gamma
##    2.1
##
## - best performance: 0.215567
##
## - Detailed performance results:
##   gamma     error dispersion
## 1   0.1 0.2159238 0.01203596
## 2   1.1 0.2166334 0.01072112
## 3   2.1 0.2155670 0.01176814
## 4   3.1 0.2155670 0.01176814
## 5   4.1 0.2155670 0.01176814
```

```
best.rd=rd.poly$best.model

pre.rd=predict(best.rd,newdata = test_val1)
confusionMatrix(pre.rd,test_val1$Churn)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  No Yes
##        No  942 203
##        Yes  90 170
##
##                Accuracy : 0.7915
##                  95% CI : (0.7693, 0.8124)
##     No Information Rate : 0.7345
##     P-Value [Acc > NIR] : 4.302e-07
##
##                   Kappa : 0.408
##
##  Mcnemar's Test P-Value : 6.025e-11
##
##             Sensitivity : 0.9128
```

```
##             Specificity : 0.4558
##          Pos Pred Value : 0.8227
##          Neg Pred Value : 0.6538
##              Prevalence : 0.7345
##          Detection Rate : 0.6705
##    Detection Prevalence : 0.8149
##       Balanced Accuracy : 0.6843
##
##        'Positive' Class : No
##
```

SECTION V

CONCLUSIONS

The different models that were developed show an acceptable accuracy level for the amount of variables used and the complexity of the model itself.

After the test of the models we can say that the Contract variable is the most relevant variable to predict the end of the service with the company. As such, when the contract of a client has lasted for a year or more, this client is more likely to end its contract rather than a client with a contract that has lasted less than a year, regardless of the gender, the kind of service or the payment method.

On the logistic regression model we can highlight that the selected variables are highly significant for the accuracy on the prediction of the model. On the case of the random forest model we obtained that a 'No' prediction is more accurate than a 'Yes' prediction for the same dataset. For the overall accuracy, all the models present an accuracy of around 0.79.

This next table synthesize the accuracy obtained for each model.

```
models <- c("Decision Tree","Random Forest","Logic Regression", "Support Vector Machine", "Radial Suppo
accur <- c(0.7917, 0.7957, 0.7941, 0.7943, 0.7955)
modelos <- data.frame(models,accur)
kable(modelos,caption="Accuracy by model",
align="lc", col.names = c("Models","Accuracy"))
```

Table 3: Accuracy by model

Models	Accuracy
Decision Tree	0.7917
Random Forest	0.7957
Logic Regression	0.7941
Support Vector Machine	0.7943
Radial Support Vector Machine	0.7955

SECTION VI

BIBLIOGRAPHY

Garet James et.al 2021 An Introduction to Statistical Learning with Application in R 2da ed. edition.

Irizarry A. Rafael and Love I. Michael. Data Analysis for the life Sciences.

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

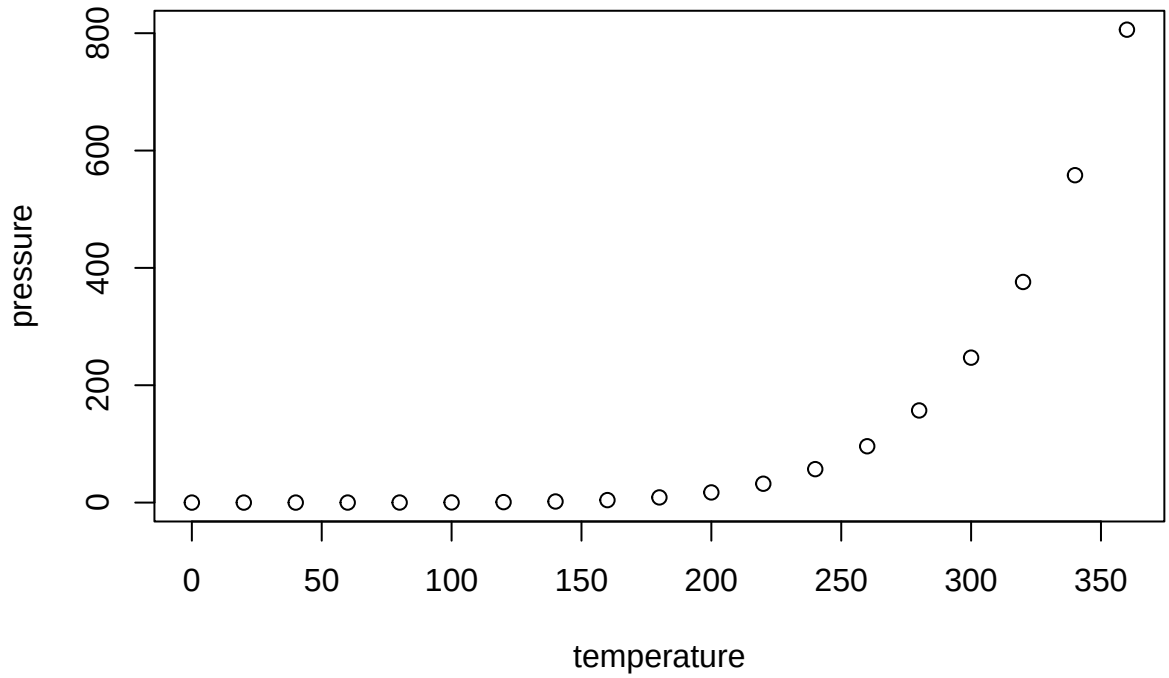
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed           dist
##  Min.   : 4.0   Min.   :  2.00
##  1st Qu.:12.0   1st Qu.: 26.00
##  Median :15.0   Median : 36.00
##  Mean   :15.4   Mean   : 42.98
##  3rd Qu.:19.0   3rd Qu.: 56.00
##  Max.   :25.0   Max.   :120.00
```

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.