Business Analytics Project(Group-7)

AJEET KUMAR SANDELA
LOKESH JETANGI
MAHESH KONDURI
SOUMYA GUN REDDY

Abstract:

Churn is a challenge for telecom companies since it is harder to bring in new customers than to keep hold of their current ones. Recent data reveals that current customers account for a considerable portion of firm revenues, which has drawn a lot of attention to customer churn modeling. Businesses routinely use Data Mining techniques to identify consumers who are likely to leave, as they are interested in doing so. We were able to identify project participants who were likely to leave by using the data at hand, and we provided them with sufficient support to persuade them to do so.

INTRODUCTION:

Finding a balance between customer acquisition and retention is tough because both are crucial factors that directly affect a company's profitability. Since the churn rate affects the business's sales, customer retention is probably going to be a key factor.

Network problems, poor customer service, pricey monthly plans, and other factors can all lead customers to switch service providers.

These problems may be resolved, among other things, by providing discounts or better service to clients in order to prevent them from switching service providers. These days, when we have the analytical skills to sort through complex data, evaluate it, and draw out pertinent information, we can use this knowledge to identify patterns and predict what will happen in the future.

This project's objective is to use a predictive model to evaluate data and identify trends in order to predict when a long-term customer will switch service providers. Regression, Nave bias, KNN, and other prediction models can all be applied. We'll use logistic regression to build our model.

DATA & Overview:

The training data (‘Churn_Training.csv’) is available to download from the course portal along with this assignment. There are 19 predictors, mostly numeric:

1. state (categorical),
2. account_length,
3. area_code,
4. international_plan (yes/no),
5. voice_mail_plan (yes/no),
6. number_vmail_messages,
7. total_day_minutes,
8. total_day_calls,
9. total_day_charge,
10. total_eve_minutes,
11. total_eve_calls,
12. total_eve_charge,
13. total_night_minutes,

14. total_night_calls,
15. total_night_charge,

16. total_intl_minutes,
17. total_intl_calls,
18. total_intl_charge
19. number_customer_service_calls.

ABC Wireless company has provided data related to demographics, such as state, account length, area code, international plan, and voice-mail plan, as well as calling behavior metrics including the number of messages, total day minutes, total day calls, total day charge, total evening minutes, total evening calls, total evening charges, total night minutes, total night calls, total night charges, total international minutes, total international calls, total international charges, and number of calls to customer service. These data points may help infer factors responsible for customer churn.

Business Analytics PROJECT

GROUP-7

2023-04-29

R Markdown

```
#install.packages("RANN")
#install.packages("pROC")
#install.packages("rpart")
#install.packages("rpart.plot")

library(readr)
library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.
0.0 ──
## ✓ dplyr     1.1.0     ✓ purrr     1.0.1
## ✓ forcats   1.0.0     ✓ stringr   1.5.0
## ✓ ggplot2   3.4.1     ✓ tibble    3.1.8
## ✓ lubridate 1.9.2     ✓ tidyr     1.3.0
## ── Conflicts ────────────────────────────────────────── tidyverse_conflict
s() ──
## ✗ dplyr::filter() masks stats::filter()
## ✗ dplyr::lag()    masks stats::lag()
## ℹ Use the ]8;;http://conflicted.r-lib.org/conflicted package]8;; to force
all conflicts to become errors

library(caret)

## Warning: package 'caret' was built under R version 4.2.3

## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift

library(pROC)

## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
##
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```r
library(ggcorrplot)
library(gmodels)
```

```
##
## Attaching package: 'gmodels'
##
## The following object is masked from 'package:pROC':
##
##     ci
```

```r
library(rpart)
library(RANN)
```

```
## Warning: package 'RANN' was built under R version 4.2.3
```

```r
library(rpart.plot)
```

```
## Warning: package 'rpart.plot' was built under R version 4.2.3
```

```r
#Importing the Customer churn dataset
customer_churn <- read.csv("C:/Users/jetan/Downloads/Churn_Train.csv")
head(customer_churn) #printing the top rows of the dataset
```

```
##   state account_length     area_code international_plan voice_mail_plan
## 1    NV            125 area_code_510                 no              no
## 2    HI            108 area_code_415                 no              no
## 3    DC             82 area_code_415                 no              no
## 4    HI             NA area_code_408                 no             yes
## 5    OH             83 area_code_415                 no              no
## 6    MO             89 area_code_415                 no              no
##   number_vmail_messages total_day_minutes total_day_calls total_day_charge
## 1                     0            2013.4              99            28.66
## 2                     0             291.6              99            49.57
## 3                     0             300.3             109            51.05
## 4                    30             110.3              71            18.75
## 5                     0             337.4             120            57.36
## 6                     0             178.7              81            30.38
##   total_eve_minutes total_eve_calls total_eve_charge total_night_minutes
## 1            1107.6             107            14.93               243.3
## 2             221.1              93            18.79               229.2
## 3             181.0             100            15.39               270.1
## 4             182.4             108            15.50               183.8
## 5             227.4             116            19.33               153.9
## 6                NA              74            19.86               131.9
##   total_night_calls total_night_charge total_intl_minutes total_intl_calls
## 1                92              10.95               10.9                7
## 2               110              10.31               14.0                9
## 3                73              12.15               11.7                4
## 4                88               8.27               11.0                8
## 5               114               6.93               15.8                7
## 6               120               5.94                9.1                4
```

```
##   total_intl_charge number_customer_service_calls churn
## 1              2.94                             0    no
## 2              3.78                             2   yes
## 3              3.16                             0   yes
## 4              2.97                             2    no
## 5              4.27                             0   yes
## 6              2.46                             1    no
```

```
#structure of the dataset
str(customer_churn)
```

```
## 'data.frame':    3333 obs. of  20 variables:
##  $ state                         : chr  "NV" "HI" "DC" "HI" ...
##  $ account_length                : int  125 108 82 NA 83 89 135 28 86 65 ..
.
##  $ area_code                     : chr  "area_code_510" "area_code_415" "ar
ea_code_415" "area_code_408" ...
##  $ international_plan            : chr  "no" "no" "no" "no" ...
##  $ voice_mail_plan               : chr  "no" "no" "no" "yes" ...
##  $ number_vmail_messages         : int  0 0 0 30 0 0 0 0 0 0 ...
##  $ total_day_minutes             : num  2013 292 300 110 337 ...
##  $ total_day_calls               : int  99 99 109 71 120 81 81 87 115 137 .
..
##  $ total_day_charge              : num  28.7 49.6 51 18.8 57.4 ...
##  $ total_eve_minutes             : num  1108 221 181 182 227 ...
##  $ total_eve_calls               : int  107 93 100 108 116 74 114 92 112 83
...
##  $ total_eve_charge              : num  14.9 18.8 15.4 15.5 19.3 ...
##  $ total_night_minutes           : num  243 229 270 184 154 ...
##  $ total_night_calls             : int  92 110 73 88 114 120 82 112 95 111
...
##  $ total_night_charge            : num  10.95 10.31 12.15 8.27 6.93 ...
##  $ total_intl_minutes            : num  10.9 14 11.7 11 15.8 9.1 10.3 10.1
9.8 12.7 ...
##  $ total_intl_calls              : int  7 9 4 8 7 4 6 3 7 6 ...
##  $ total_intl_charge             : num  2.94 3.78 3.16 2.97 4.27 2.46 2.78
2.73 2.65 3.43 ...
##  $ number_customer_service_calls: int  0 2 0 2 0 1 1 3 2 4 ...
##  $ churn                         : chr  "no" "yes" "yes" "no" ...
```

```
#Checking the summary of the dataset
summary(customer_churn)
```

```
##     state           account_length     area_code         international_pla
n
##  Length:3333        Min.   :-209.00   Length:3333        Length:3333
##  Class :character   1st Qu.:  72.00   Class :character   Class :character
##  Mode  :character   Median : 100.00   Mode  :character   Mode  :character
##                     Mean   :  97.32
##                     3rd Qu.: 127.00
##                     Max.   : 243.00
```

```
##                      NA's   :501
##  voice_mail_plan    number_vmail_messages total_day_minutes total_day_call
s
##  Length:3333        Min.   :-10.000       Min.   :   0.0    Min.   :  0.0
##  Class :character   1st Qu.:  0.000       1st Qu.: 149.3    1st Qu.: 87.0
##  Mode  :character   Median :  0.000       Median : 190.5    Median :101.0
##                     Mean   :  7.333       Mean   : 418.9    Mean   :100.3
##                     3rd Qu.: 16.000       3rd Qu.: 237.8    3rd Qu.:114.0
##                     Max.   : 51.000       Max.   :2185.1    Max.   :165.0
##                     NA's   :200           NA's   :200       NA's   :200
##  total_day_charge total_eve_minutes total_eve_calls total_eve_charge
##  Min.   : 0.00    Min.   :   0.0    Min.   :  0.0   Min.   : 0.00
##  1st Qu.:24.45    1st Qu.: 170.5    1st Qu.: 87.0   1st Qu.:14.14
##  Median :30.65    Median : 209.9    Median :100.0   Median :17.09
##  Mean   :30.63    Mean   : 324.3    Mean   :100.1   Mean   :17.08
##  3rd Qu.:36.84    3rd Qu.: 257.6    3rd Qu.:114.0   3rd Qu.:20.00
##  Max.   :59.64    Max.   :1244.2    Max.   :170.0   Max.   :30.91
##  NA's   :200      NA's   :301       NA's   :200     NA's   :200
##  total_night_minutes total_night_calls total_night_charge total_intl_minut
es
##  Min.   : 23.2       Min.   : 33.0     Min.   : 1.040     Min.   : 0.00
##  1st Qu.:167.3       1st Qu.: 87.0     1st Qu.: 7.530     1st Qu.: 8.50
##  Median :201.4       Median :100.0     Median : 9.060     Median :10.30
##  Mean   :201.2       Mean   :100.1     Mean   : 9.054     Mean   :10.23
##  3rd Qu.:235.3       3rd Qu.:113.0     3rd Qu.:10.590     3rd Qu.:12.10
##  Max.   :395.0       Max.   :175.0     Max.   :17.770     Max.   :20.00
##  NA's   :200                           NA's   :200        NA's   :200
##  total_intl_calls total_intl_charge number_customer_service_calls
##  Min.   : 0.00    Min.   :0.000     Min.   :0.000
##  1st Qu.: 3.00    1st Qu.:2.300     1st Qu.:1.000
##  Median : 4.00    Median :2.780     Median :1.000
##  Mean   : 4.47    Mean   :2.762     Mean   :1.561
##  3rd Qu.: 6.00    3rd Qu.:3.270     3rd Qu.:2.000
##  Max.   :20.00    Max.   :5.400     Max.   :9.000
##  NA's   :301      NA's   :200       NA's   :200
##     churn
##  Length:3333
##  Class :character
##  Mode  :character
##
##
##
##
```

```
#As per the summary observed the dataset contains Negative Values, Missing va
lues and Outliers So we are trying to minimize the error rate without directl
y eliminating them from the dataset as this is a small dataset

# BY OBSERVING THE structure and summary of the dataset we are converting the
#char variables to factors
```

```r
#Conversion
customer_churn <- customer_churn %>% mutate_if(is.character, as.factor)
str(customer_churn) #checking the conversion status
```

```
## 'data.frame':    3333 obs. of  20 variables:
##  $ state                         : Factor w/ 51 levels "AK","AL","AR",..: 3
4 12 8 12 36 25 28 39 13 16 ...
##  $ account_length                : int  125 108 82 NA 83 89 135 28 86 65 ..
.
##  $ area_code                     : Factor w/ 3 levels "area_code_408",..: 3
2 2 1 2 2 2 2 1 2 ...
##  $ international_plan            : Factor w/ 2 levels "no","yes": 1 1 1 1 1
1 1 1 1 1 ...
##  $ voice_mail_plan               : Factor w/ 2 levels "no","yes": 1 1 1 2 1
1 1 1 1 1 ...
##  $ number_vmail_messages         : int  0 0 0 30 0 0 0 0 0 0 ...
##  $ total_day_minutes             : num  2013 292 300 110 337 ...
##  $ total_day_calls               : int  99 99 109 71 120 81 81 87 115 137 .
..
##  $ total_day_charge              : num  28.7 49.6 51 18.8 57.4 ...
##  $ total_eve_minutes             : num  1108 221 181 182 227 ...
##  $ total_eve_calls               : int  107 93 100 108 116 74 114 92 112 83
...
##  $ total_eve_charge              : num  14.9 18.8 15.4 15.5 19.3 ...
##  $ total_night_minutes           : num  243 229 270 184 154 ...
##  $ total_night_calls             : int  92 110 73 88 114 120 82 112 95 111
...
##  $ total_night_charge            : num  10.95 10.31 12.15 8.27 6.93 ...
##  $ total_intl_minutes            : num  10.9 14 11.7 11 15.8 9.1 10.3 10.1
9.8 12.7 ...
##  $ total_intl_calls              : int  7 9 4 8 7 4 6 3 7 6 ...
##  $ total_intl_charge             : num  2.94 3.78 3.16 2.97 4.27 2.46 2.78
2.73 2.65 3.43 ...
##  $ number_customer_service_calls: int  0 2 0 2 0 1 1 3 2 4 ...
##  $ churn                         : Factor w/ 2 levels "no","yes": 1 2 2 1 2
1 1 1 1 2 ...
```

```r
#Identifying the Negative values column-wise
sapply(customer_churn, function(x) sum(x < 0, na.rm = TRUE))
```

```
##                          state                 account_length
##                              0                             51
##                      area_code             international_plan
##                              0                              0
##                voice_mail_plan          number_vmail_messages
##                              0                            201
##              total_day_minutes                total_day_calls
##                              0                              0
##               total_day_charge              total_eve_minutes
##                              0                              0
```

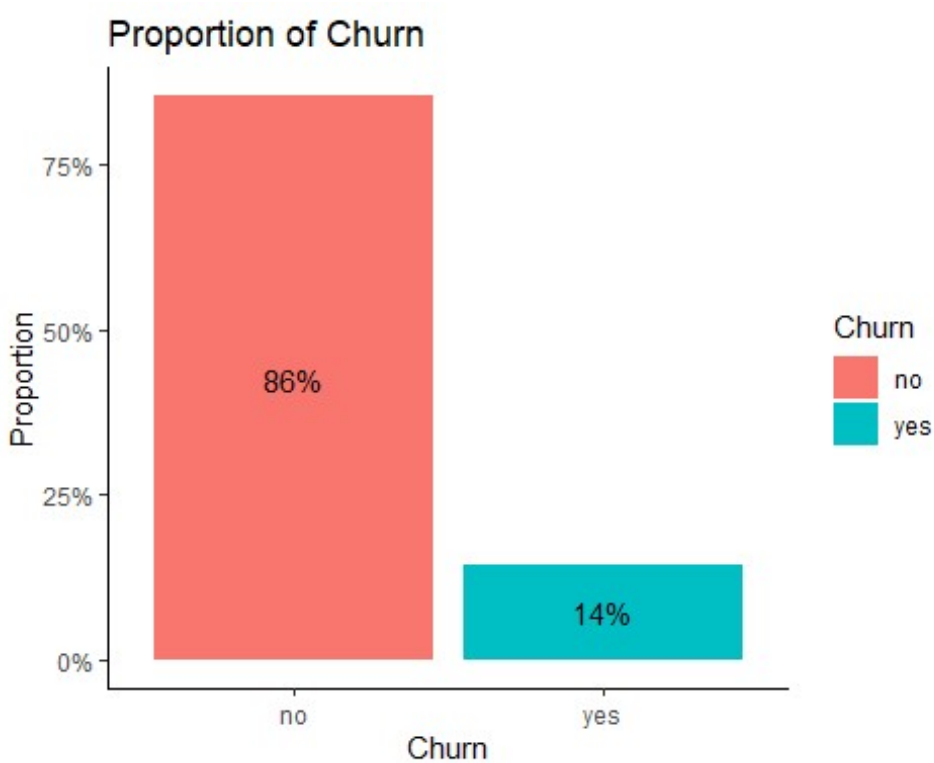
```
##               total_eve_calls              total_eve_charge
##                             0                             0
##           total_night_minutes             total_night_calls
##                             0                             0
##            total_night_charge            total_intl_minutes
##                             0                             0
##              total_intl_calls             total_intl_charge
##                             0                             0
## number_customer_service_calls                         churn
##                             0                             0
```

```
#Identifying the missing values
missing_values_in_dataset <- is.na(customer_churn)

# Count the number of missing values in each column
colSums(missing_values_in_dataset)
```

```
##                          state                account_length
##                              0                           501
##                      area_code             international_plan
##                              0                             0
##                voice_mail_plan          number_vmail_messages
##                              0                           200
##              total_day_minutes                total_day_calls
##                            200                           200
##               total_day_charge              total_eve_minutes
##                            200                           301
##               total_eve_calls               total_eve_charge
##                            200                           200
##           total_night_minutes              total_night_calls
##                            200                             0
##            total_night_charge             total_intl_minutes
##                            200                           200
##              total_intl_calls              total_intl_charge
##                            301                           200
## number_customer_service_calls                         churn
##                            200                             0
```

```
# Replace negative values in numeric columns with their absolute value
customer_churn[which(customer_churn < 0 & is.numeric(customer_churn))] <-
  abs(customer_churn[which(customer_churn < 0 & is.numeric(customer_churn))])

# Impute missing numeric values with median imputation
library(caret)

imputation_model <- preProcess(customer_churn %>% select_if(is.numeric), meth
od = "medianImpute")
imputed_data <- predict(imputation_model, customer_churn %>% select_if(is.num
eric))
```

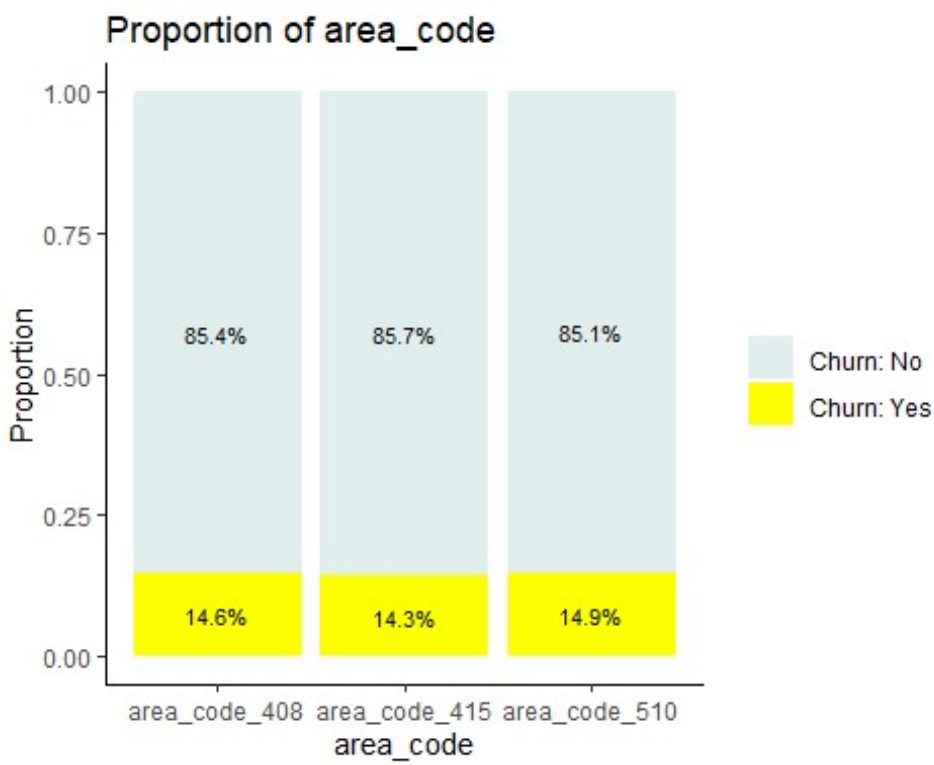
```r
# Replace missing values in the original data with the imputed values
customer_churn <- customer_churn %>%
  select(-where(is.numeric)) %>%
  bind_cols(imputed_data)

#library(ggplot2)

customer_churn %>%
  count(churn) %>%
  mutate(prop = n / sum(n)) %>%
  ggplot(aes(x = churn, y = prop, fill = churn)) +
  geom_col() +
  geom_text(aes(label = scales::percent(prop)),
            position = position_stack(vjust = 0.5)) +
  scale_y_continuous(labels = scales::percent_format()) +
  labs(x = "Churn", y = "Proportion", fill = "Churn") +
  ggtitle("Proportion of Churn") +
  theme_classic()
```




```r
#From the above graph we can see that only around 14% of population are Churn
ed,      rest 86% are retained in the telecom network.

#PROPORTION OF AREA_CODE
customer_churn %>%
  select(area_code, churn) %>%
  na.omit() %>%
  group_by(area_code, churn) %>%
```

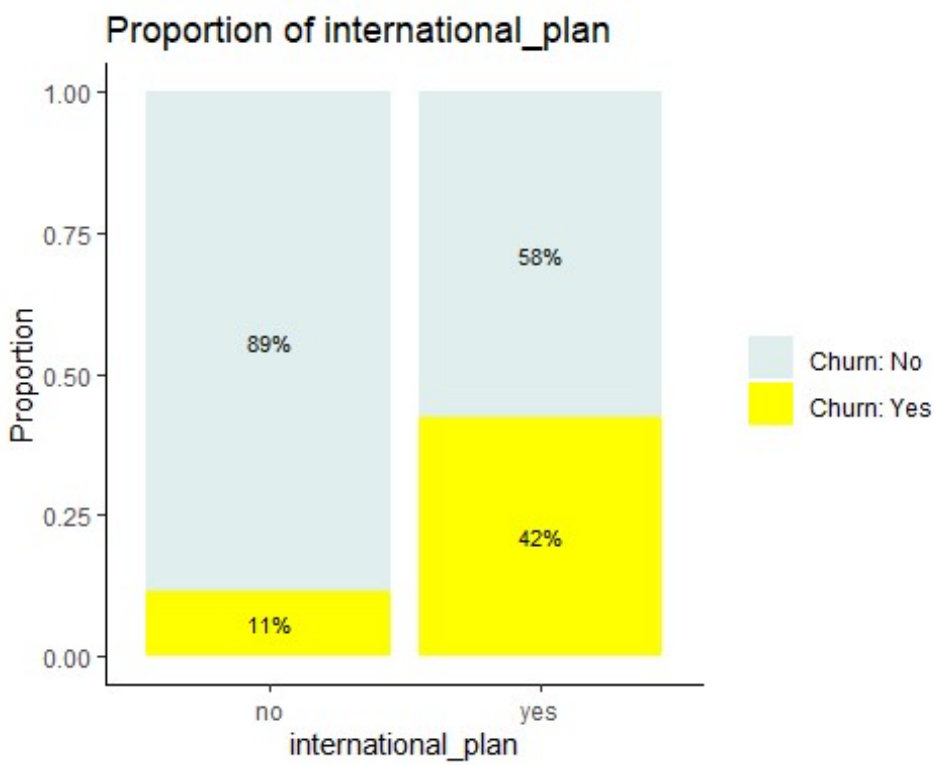
```
  summarize(count = n()) %>%
  mutate(prop = count / sum(count)) %>%
  ggplot(aes(x = area_code, y = prop, fill = churn)) +
  geom_col() +
  geom_text(aes(label = scales::percent(prop, accuracy = 0.1)),
            position = position_stack(vjust = 0.5),
            size = 3) +
  scale_fill_manual(labels = c("Churn: No", "Churn: Yes"),
                    values = c("azure2", "yellow")) +
  labs(y = "Proportion", title = "Proportion of area_code") +
  theme_classic() +
  theme(legend.title = element_blank())
```

```
## `summarise()` has grouped output by 'area_code'. You can override using th
e
## `.groups` argument.
```

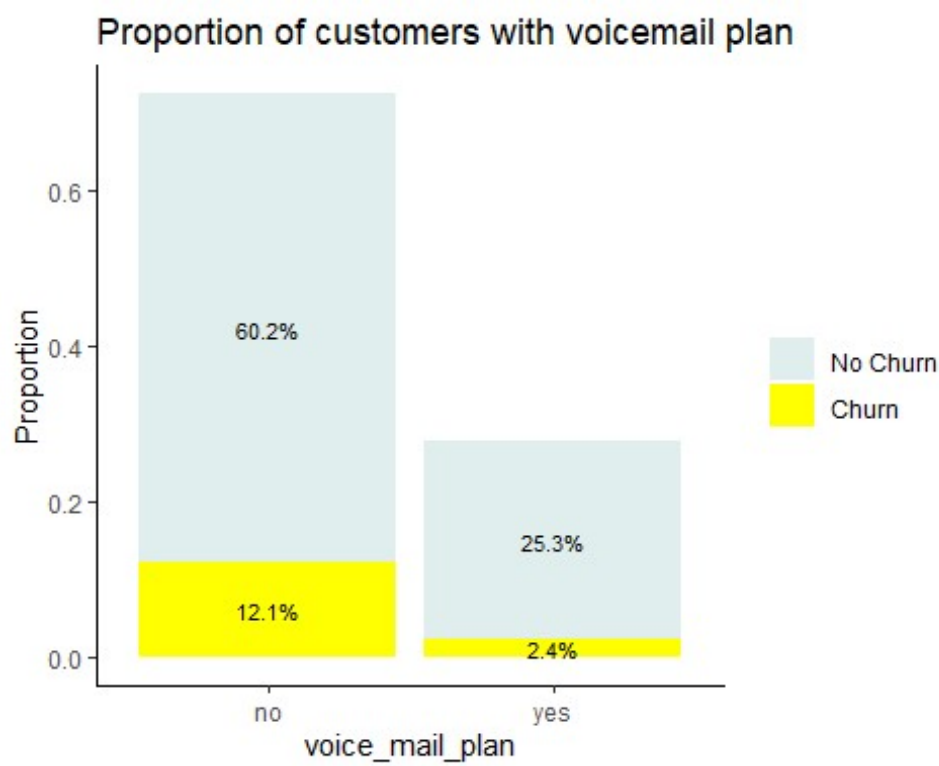



```
#proportion of international_plan
customer_churn %>%
  count(international_plan, churn) %>%
  group_by(international_plan) %>%
  mutate(prop = n / sum(n)) %>%
  ggplot(aes(x = international_plan, y = prop, fill = churn)) +
  geom_col() +
  geom_text(aes(label = paste0(format(prop * 100, digits = 1), "%")),
            position = position_stack(vjust = 0.5),
            size = 3) +
```

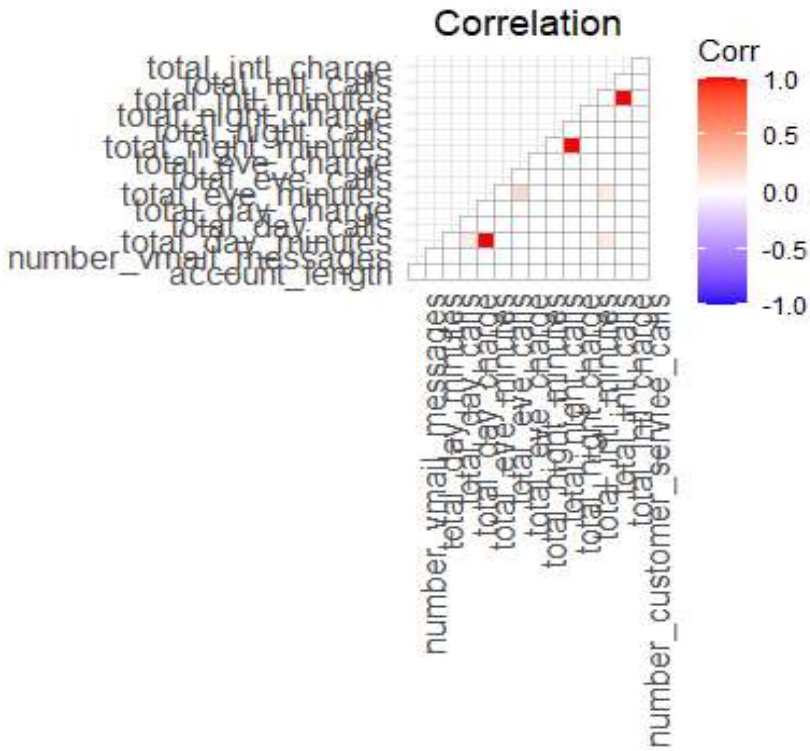
```
  labs(y = "Proportion", title = "Proportion of international_plan") +
  scale_fill_manual(labels = c("Churn: No", "Churn: Yes"),
                    values = c("azure2", "yellow")) +
  theme_classic() +
  theme(legend.title = element_blank())
```




```
#PROPORTION OF VOICE_MAIL_PLAN
as.data.frame(prop.table(table(customer_churn[c("voice_mail_plan", "churn")])
)) %>%
  ggplot(aes(x = voice_mail_plan, y = Freq, fill = churn)) +
  geom_col() +
  geom_text(aes(label = paste0(round(Freq * 100, 1), "%")),
            position = position_stack(vjust = 0.5),
            size = 2.8) +
  theme_classic() +
  labs(y = "Proportion", title = "Proportion of customers with voicemail plan
") +
  theme(legend.title = element_blank()) +
  scale_fill_manual(labels = c("No Churn", "Churn"),
                    values = c("azure2", "yellow"))
```




```
#Correlation matrix
# Compute correlation matrix for numeric columns
Churn_Data_cor <- round(cor(customer_churn %>% select_if(is.numeric)), 1)

# Visualize correlation using ggcorrplot
ggcorrplot(Churn_Data_cor, title = "Correlation", type = "lower") +
  theme(plot.title = element_text(hjust = 0.5),
        axis.text.x = element_text(angle = 90))
```



Since total minutes and total charges for the day, evening, night, and international services are closely related, we may choose to omit them to avoid "multicollinearity" problems.

Model Approach:

Binary classification is the process of dividing an example into two categories using a classifier. We will use both logistic regression and decision trees to tackle this problem, comparing the performance matrices of both models to determine which performs better as the goal variable in this data is categorical and the outcome for this model is a likelihood or probability of odds between 0 and 1.

Logistic Regression - Regression with logit

Pre-processing the data

Now, Dividing the dataset into sets for training (80%) and validation (20%)

Our model, which we will be testing across the training set, will be fitted

```
# Create dummy variables for categorical columns, except "state" and "churn"
customer_churn <- customer_churn %>%
  select(-state, -churn) %>%
  fastDummies::dummy_cols(remove_selected_columns = TRUE) %>%
  mutate(state = customer_churn$state, churn = customer_churn$churn)
```

```r
#Splitting dataset into training (80%) and validation (20%)
set.seed(123)
index <- createDataPartition(customer_churn$churn, p=0.8, list=FALSE)
Churn_Data_train <- customer_churn[index,]
Churn_Data_test <- customer_churn[-index,]

# Data scaling
scaling <- preProcess(Churn_Data_train %>% select_if(is.numeric), method = c(
"center", "scale"))
training_normalized <- predict(scaling, Churn_Data_train %>% select_if(is.num
eric))
testing_normalized <- predict(scaling, Churn_Data_test %>% select_if(is.numer
ic))

# Add churn column back to the normalized data frames
training_normalized$churn <- Churn_Data_train$churn
testing_normalized$churn <- Churn_Data_test$churn

#Logistic Regression Model building
logistic_model1 <- glm(churn ~ ., data = training_normalized , family= "binom
ial")

#Summary
summary(logistic_model1)

##
## Call:
## glm(formula = churn ~ ., family = "binomial", data = training_normalized)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0918  -0.5152  -0.3500  -0.2059   3.1693
##
## Coefficients: (3 not defined because of singularities)
##                                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)                   -2.290e+00  7.918e-02 -28.917  < 2e-16 ***
## account_length                 6.003e-02  6.318e-02   0.950   0.3421
## number_vmail_messages         -2.625e-02  1.499e-01  -0.175   0.8610
## total_day_minutes             -3.171e+00  1.311e+00  -2.418   0.0156 *
## total_day_calls                5.406e-02  6.156e-02   0.878   0.3798
## total_day_charge               9.133e-01  1.228e-01   7.435 1.05e-13 ***
## total_eve_minutes              3.023e+00  1.296e+00   2.333   0.0196 *
## total_eve_calls               -8.137e-04  6.181e-02  -0.013   0.9895
## total_eve_charge              -9.630e-02  2.078e-01  -0.463   0.6431
## total_night_minutes            3.181e+01  4.746e+01   0.670   0.5027
## total_night_calls             -3.185e-03  6.160e-02  -0.052   0.9588
## total_night_charge            -3.160e+01  4.746e+01  -0.666   0.5055
## total_intl_minutes             4.649e+00  1.673e+01   0.278   0.7811
## total_intl_calls              -1.547e-01  6.662e-02  -2.323   0.0202 *
## total_intl_charge             -4.446e+00  1.673e+01  -0.266   0.7904
```

```
## number_customer_service_calls  6.520e-01  5.689e-02  11.461  < 2e-16 ***
## area_code_area_code_408       -1.218e-02  7.662e-02  -0.159   0.8737
## area_code_area_code_415       -4.075e-04  7.506e-02  -0.005   0.9957
## area_code_area_code_510               NA         NA      NA       NA
## international_plan_no         -5.983e-01  4.858e-02 -12.316  < 2e-16 ***
## international_plan_yes                NA         NA      NA       NA
## voice_mail_plan_no             4.306e-01  1.492e-01   2.886   0.0039 **
## voice_mail_plan_yes                   NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2209.0  on 2666  degrees of freedom
## Residual deviance: 1754.9  on 2647  degrees of freedom
## AIC: 1794.9
##
## Number of Fisher Scoring iterations: 6
```

```
#Predictions
set.seed(1234)
predictions <-predict(object = logistic_model1,testing_normalized, type ="res
ponse")
```

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

```
sequence1 <-data.frame(pred_cutoff =seq(0.5,0.9,0.1), pred_accuracy
=rep(0,5))

for (i in 1:5){
 logistic_model11 <-as.factor(ifelse(predictions > sequence1$pred_cutoff[i],
"yes",
"no"))
 sequence1[i,2] <-confusionMatrix(logistic_model11,Churn_Data_test$churn
)$overall[1]
}
#probability predictions along with accuracy
sequence1
```

```
##   pred_cutoff pred_accuracy
## 1         0.5     0.8648649
## 2         0.6     0.8633634
## 3         0.7     0.8618619
## 4         0.8     0.8618619
## 5         0.9     0.8603604
```

```
#Assigning labels based on maximum probability prediction
Model_Pre_lables <-as.factor(ifelse(predictions>sequence1$pred_cutoff
                             [which.max(sequence1$pred_accuracy)] ,"yes","no")
```

```
)
CrossTable(x=testing_normalized$churn, y = Model_Pre_lables, prop.chisq
=FALSE)

##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  666
##
##
##                          | Model_Pre_lables
## testing_normalized$churn |        no |       yes | Row Total |
## -------------------------|-----------|-----------|-----------|
##                      no |       558 |        12 |       570 |
##                         |     0.979 |     0.021 |     0.856 |
##                         |     0.877 |     0.400 |           |
##                         |     0.838 |     0.018 |           |
## -------------------------|-----------|-----------|-----------|
##                     yes |        78 |        18 |        96 |
##                         |     0.812 |     0.188 |     0.144 |
##                         |     0.123 |     0.600 |           |
##                         |     0.117 |     0.027 |           |
## -------------------------|-----------|-----------|-----------|
##             Column Total |       636 |        30 |       666 |
##                         |     0.955 |     0.045 |           |
## -------------------------|-----------|-----------|-----------|
##
##

confusionMatrix(Model_Pre_lables,testing_normalized$churn)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  558  78
##        yes  12  18
##
##                Accuracy : 0.8649
##                  95% CI : (0.8365, 0.8899)
##     No Information Rate : 0.8559
##     P-Value [Acc > NIR] : 0.2748
```

```
##
##                   Kappa : 0.2331
##
##  Mcnemar's Test P-Value : 7.303e-12
##
##             Sensitivity : 0.9789
##             Specificity : 0.1875
##          Pos Pred Value : 0.8774
##          Neg Pred Value : 0.6000
##              Prevalence : 0.8559
##          Detection Rate : 0.8378
##    Detection Prevalence : 0.9550
##       Balanced Accuracy : 0.5832
##
##        'Positive' Class : no
##
```

```
#Roc curve for logistic model
roc(Churn_Data_test$churn, predictions)
```

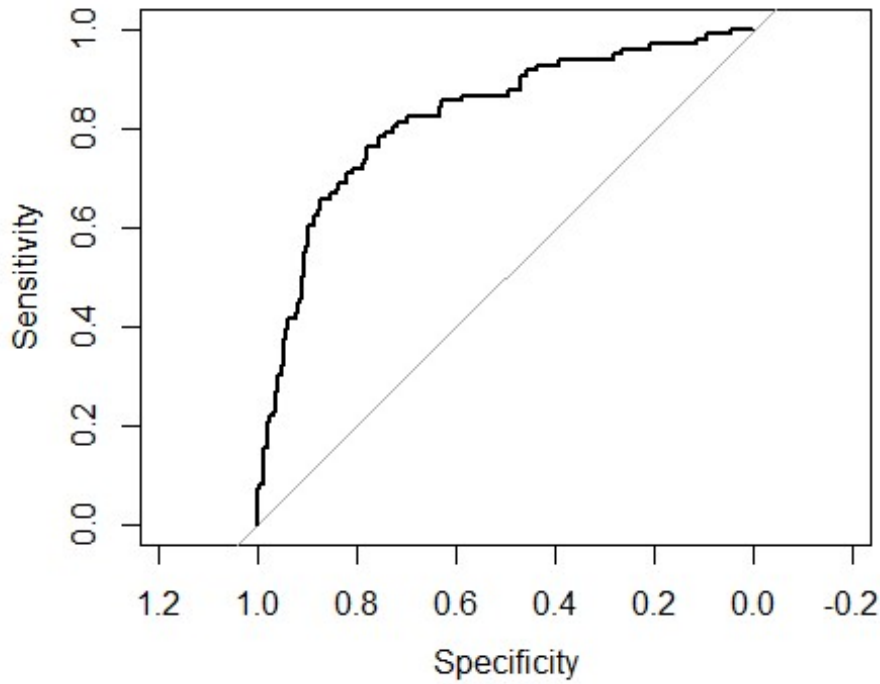
```
## Setting levels: control = no, case = yes
```

```
## Setting direction: controls < cases
```

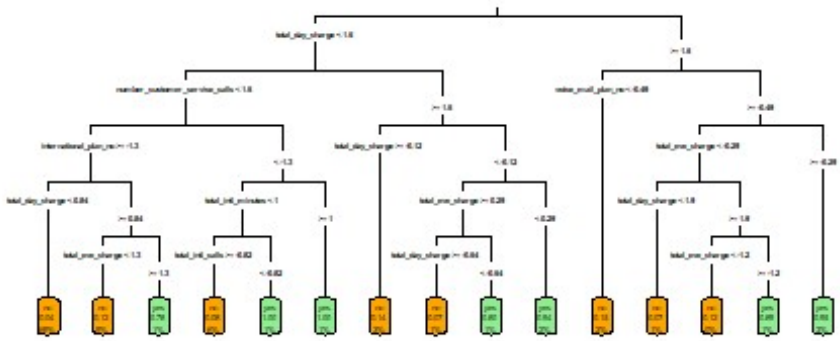
```
##
## Call:
## roc.default(response = Churn_Data_test$churn, predictor = predictions)
##
## Data: predictions in 570 controls (Churn_Data_test$churn no) < 96 cases (C
hurn_Data_test$churn yes).
## Area under the curve: 0.8211
```

```
plot.roc(roc(Churn_Data_test$churn, predictions))
```

```
## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```




```
##Decision Tree Model building
decision_model <-rpart(churn ~ .,data = training_normalized, method ="class")
rpart.plot(decision_model, type =3, box.palette =c("orange", "lightgreen"),
           fallen.leaves =TRUE)
```



The above summary plot depicts how the model splits each variable into branches or nodes using Entropy value. The algorithm calculates the change in homogeneity that would result from a split on each feature, using a measure known as information gain, to determine the optimal feature to split upon.

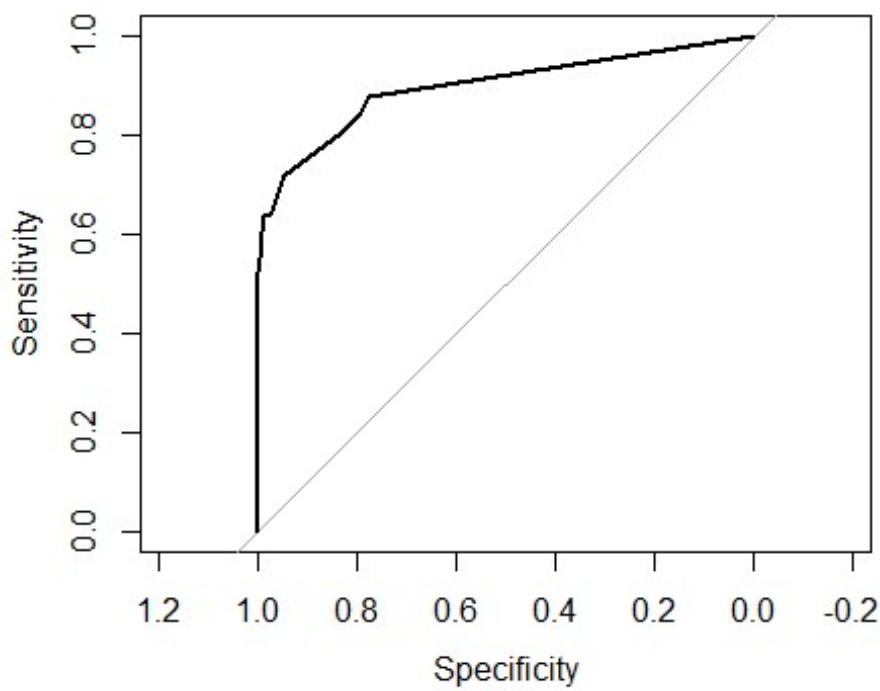
```
#Predict values based on decision_model.
pred_labels <-predict(object = decision_model,testing_normalized, type ="clas
s")
predictions <-predict(object = decision_model,testing_normalized)

#Efficiency Metrics
CrossTable(x=testing_normalized$churn, y = pred_labels, prop.chisq =FALSE)

##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  666
##
##
```

```
##                            | pred_labels
## testing_normalized$churn |        no |       yes | Row Total |
## -------------------------|-----------|-----------|-----------|
##                       no |       563 |         7 |       570 |
##                          |     0.988 |     0.012 |     0.856 |
##                          |     0.941 |     0.103 |           |
##                          |     0.845 |     0.011 |           |
## -------------------------|-----------|-----------|-----------|
##                      yes |        35 |        61 |        96 |
##                          |     0.365 |     0.635 |     0.144 |
##                          |     0.059 |     0.897 |           |
##                          |     0.053 |     0.092 |           |
## -------------------------|-----------|-----------|-----------|
##             Column Total |       598 |        68 |       666 |
##                          |     0.898 |     0.102 |           |
## -------------------------|-----------|-----------|-----------|
##
##
```

```
confusionMatrix(pred_labels,testing_normalized$churn)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  no yes
##        no  563  35
##        yes   7  61
##
##                Accuracy : 0.9369
##                  95% CI : (0.9157, 0.9542)
##     No Information Rate : 0.8559
##     P-Value [Acc > NIR] : 3.705e-11
##
##                   Kappa : 0.7091
##
##  Mcnemar's Test P-Value : 3.097e-05
##
##             Sensitivity : 0.9877
##             Specificity : 0.6354
##          Pos Pred Value : 0.9415
##          Neg Pred Value : 0.8971
##              Prevalence : 0.8559
##          Detection Rate : 0.8453
##    Detection Prevalence : 0.8979
##       Balanced Accuracy : 0.8116
##
##        'Positive' Class : no


##
```

```
#AUC for decision model
roc(Churn_Data_test$churn, predictions[,2])

## Setting levels: control = no, case = yes

## Setting direction: controls < cases

##
## Call:
## roc.default(response = Churn_Data_test$churn, predictor = predictions[,
2])
##
## Data: predictions[, 2] in 570 controls (Churn_Data_test$churn no) < 96 cas
es (Churn_Data_test$churn yes).
## Area under the curve: 0.8957

plot.roc(roc(Churn_Data_test$churn, predictions[,2]))

## Setting levels: control = no, case = yes
## Setting direction: controls < cases
```



Conclusion:
Based on the results of logistic regression and decision tree models, it was found that the decision tree had higher AUC and accuracy values. Therefore, the decision tree model was selected as the best model for future predictions.

Making Predictions based on Customers_To_Predictdata

```
# Load the Customers_To_Predict dataset
load("C:/Users/jetan/Downloads/Customers_To_Predict.RData")
Customers_To_Predict_data <- Customers_To_Predict

# Remove the 'state' column and create dummy variables for categorical featur
es
Customers_To_Predict <- Customers_To_Predict %>%
  dplyr::select(-state) %>%
  fastDummies::dummy_cols(remove_selected_columns = TRUE)

# Scale the data
Customers_To_Predict <- as.data.frame(scale(Customers_To_Predict))

# Make predictions using the decision_model
predict_labels <- predict(object = decision_model, Customers_To_Predict, type
= "class")

# Add the predicted churn labels to the original dataset and create a frequen
cy table
Customers_To_Predict <- Customers_To_Predict_data %>%
  dplyr::mutate(Churn_Prob = predict_labels)

table(Customers_To_Predict$Churn_Prob)

##
##   no  yes
## 1443  157
```

We're using a set of data that includes a list of customers whose attrition we need to forecast. Out of the 1600 total subscribers, we were able to predict that 157 users will switch from ABC Wireless to another network.

Contributions

Sno.	Name	Contribution
1.	Ajeet Kumar Sandela	We are all involved in Data Cleaning, preparation, model building, performance evaluation, presentation and report
2.	Lokesh Jetangi	
3.	Mahesh Konduri	
4.	Soumya Gun reddy	