Alex Gibbons

May 9th, 2024

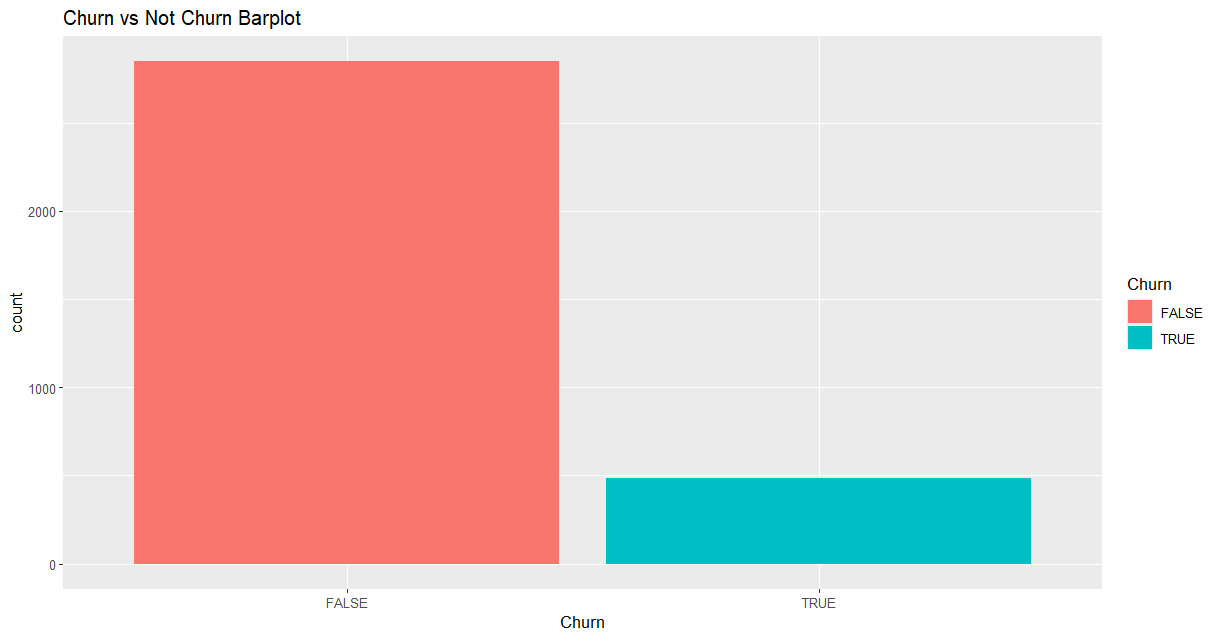
CMPT 363 – Data Mining

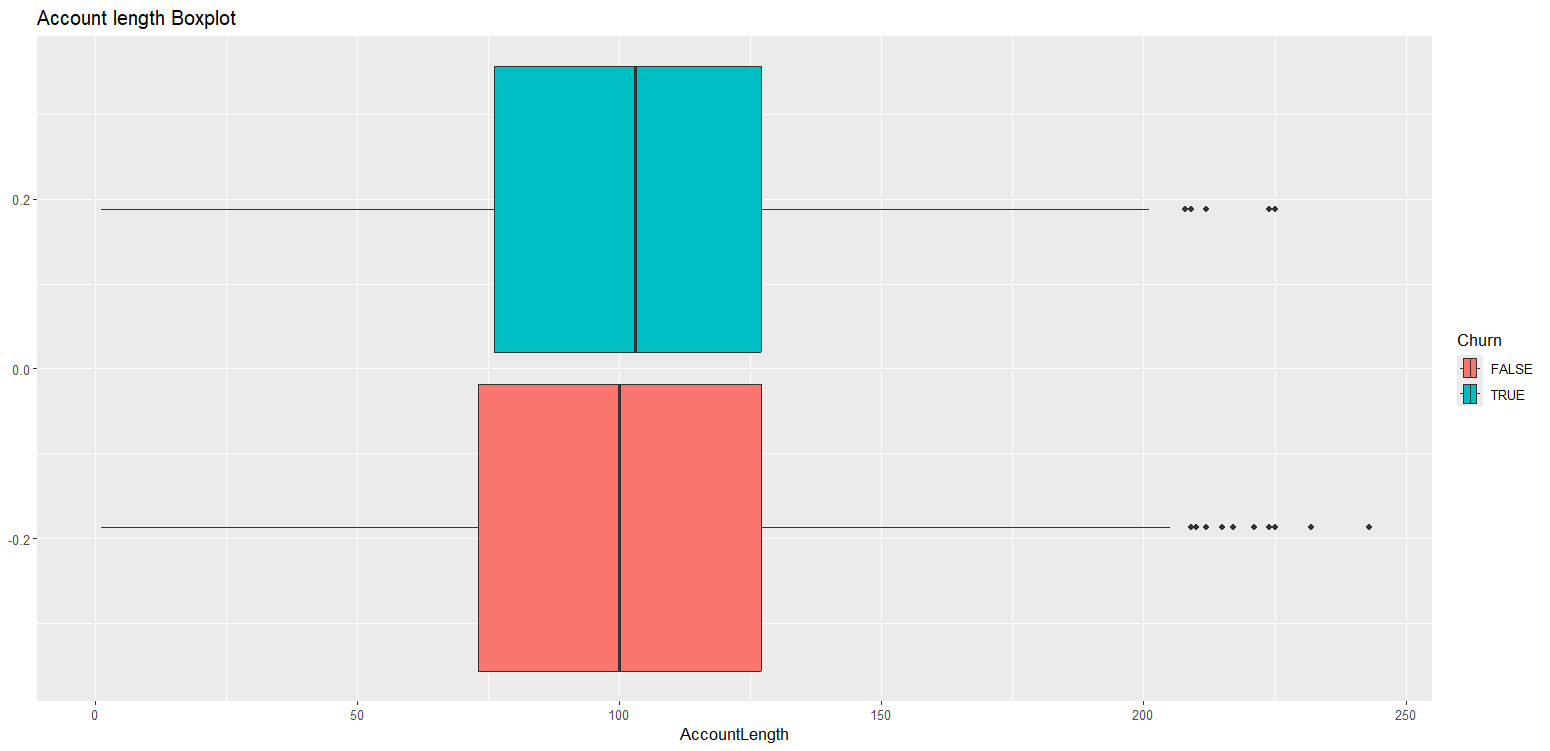
Dr. Ankur Agrawal

Churn Analysis

**Findings:**

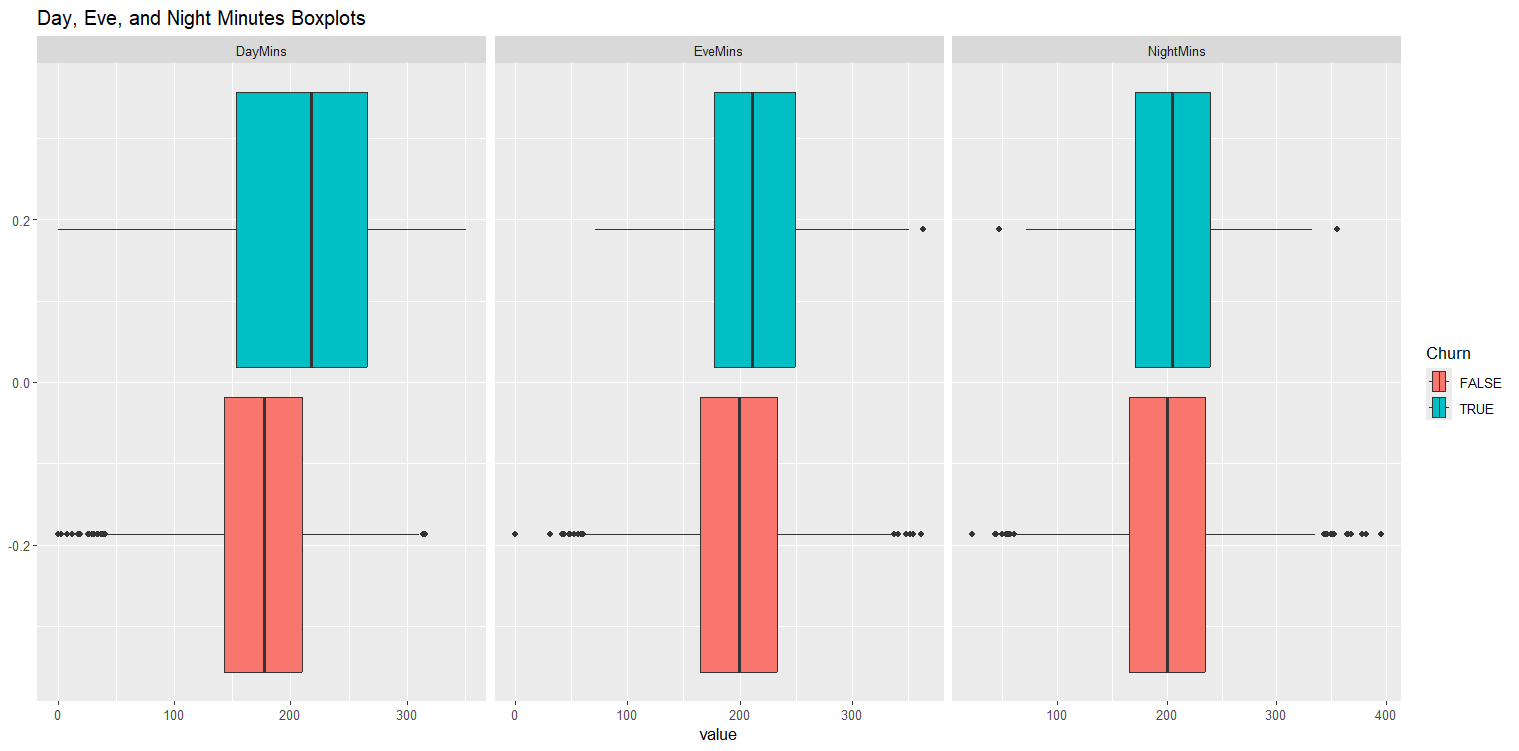
The first graph just shows the counts for customers that did not churn vs who did churn. As we can see the number of customers that did not churn is much higher, but number of those that did is not an insignificant amount.



 A screen shot of a computer

Description automatically generated

The data I received did not indicate any correlation between account length and churning as customers in both categories had relatively the same values.



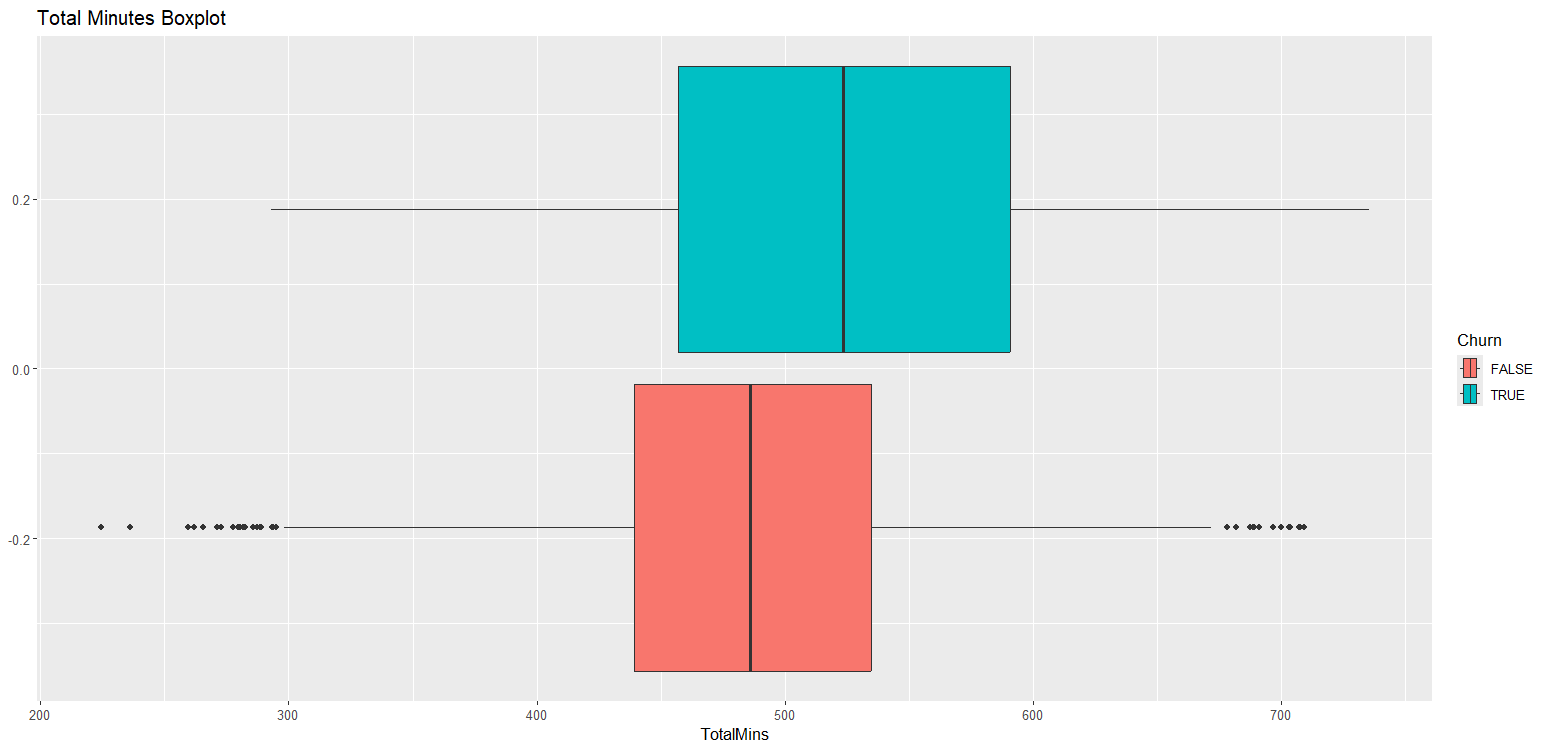
A screen shot of a computer

Description automatically generatedA screenshot of a computer screen

Description automatically generatedA screen shot of a computer

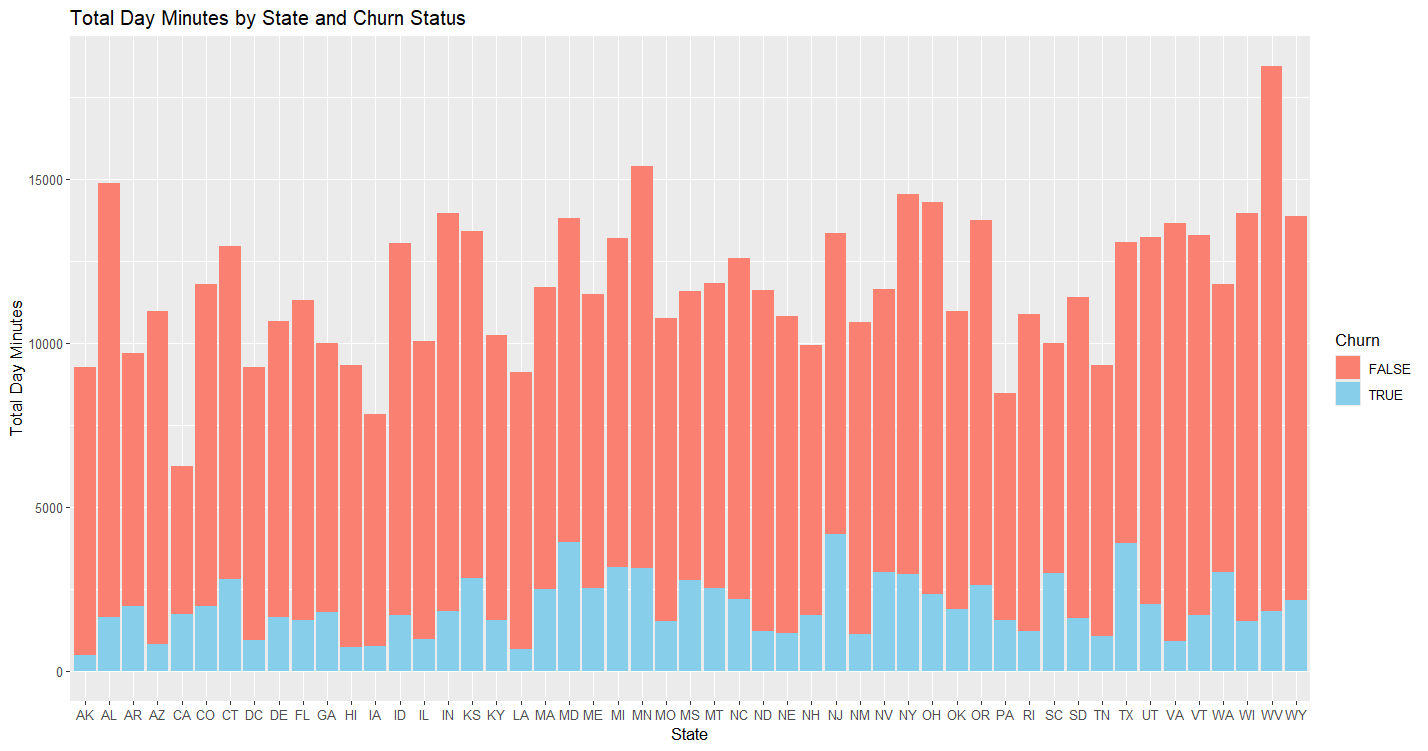
Description automatically generated

DayMins seems to have the highest correlation between whether a customer will churn or not when compared to Eve and Night.

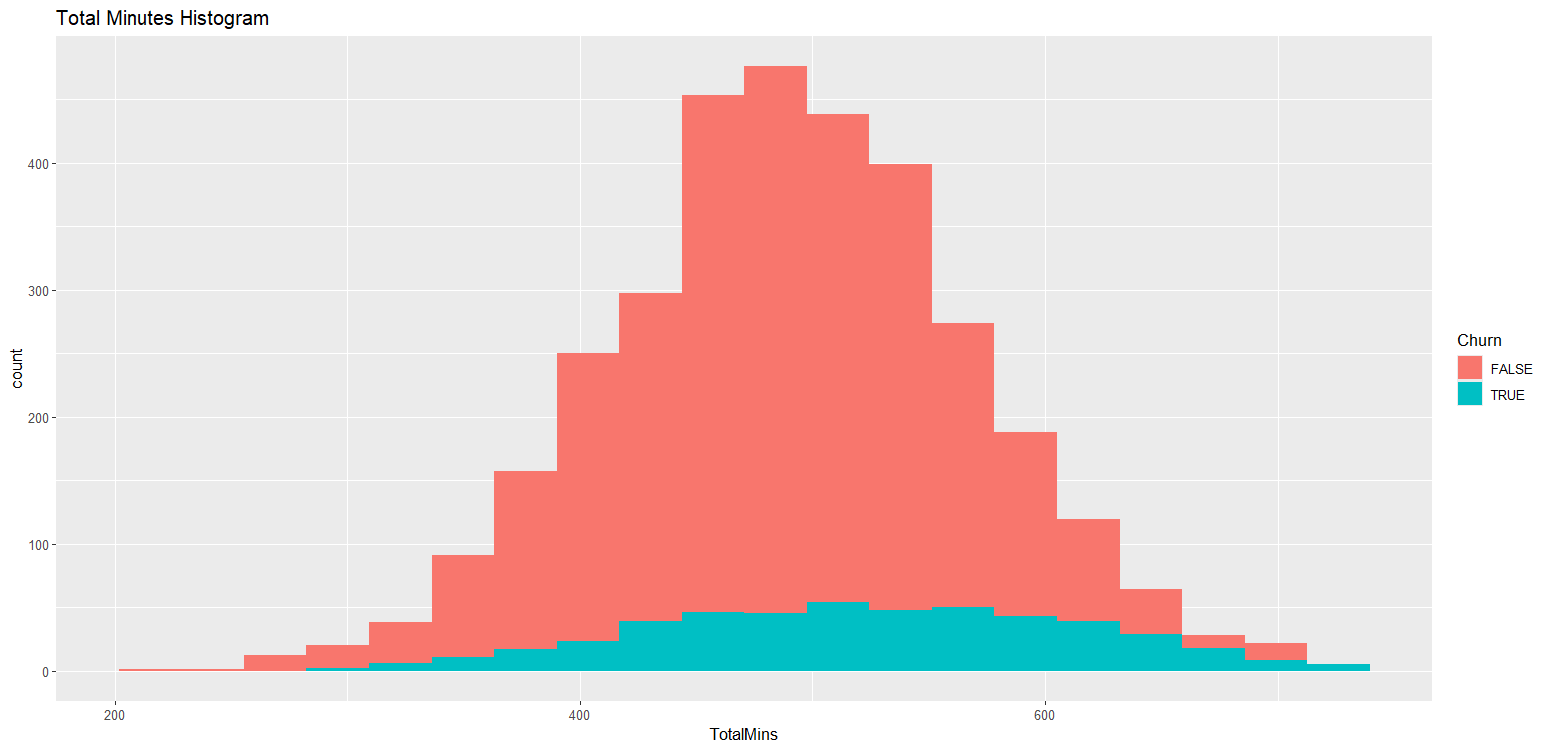
A screenshot of a computer screen

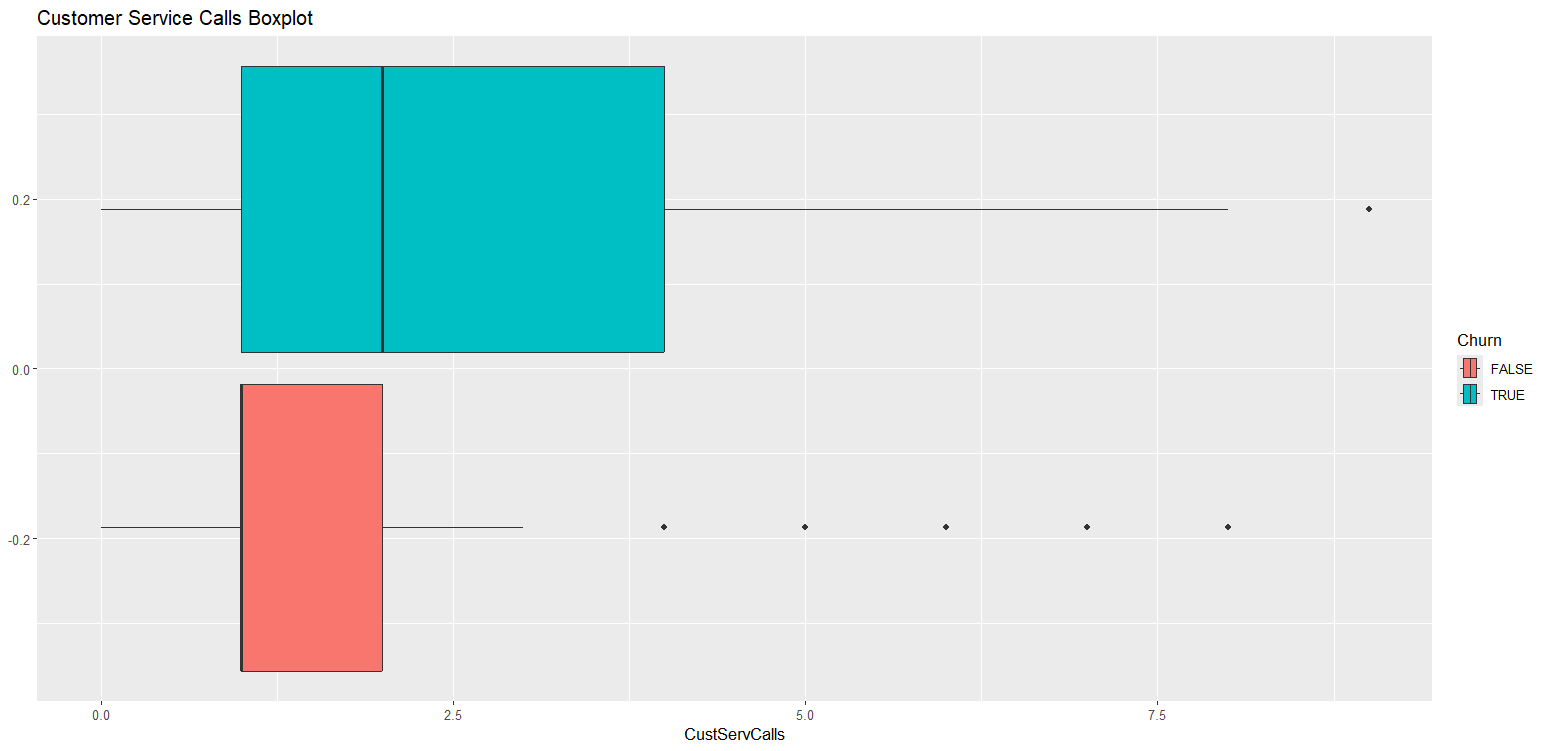
Description automatically generated

I decided to get a sum of all the minutes categories as well. The results are similar to that of DayMins

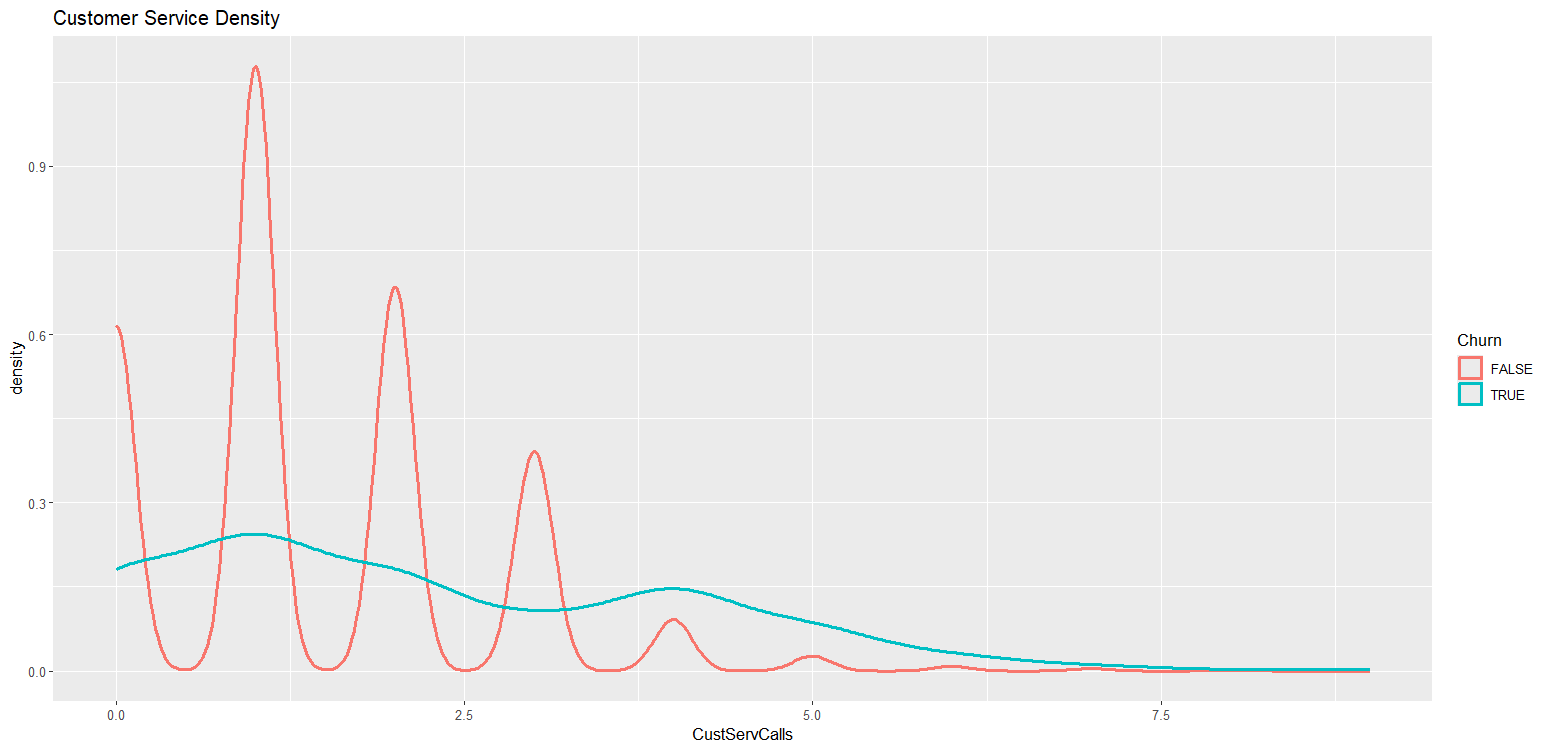


When I compared the total day number of minutes by state, I noticed that West Virginia was the most popular state, while New Jersey had the highest Churn customers.

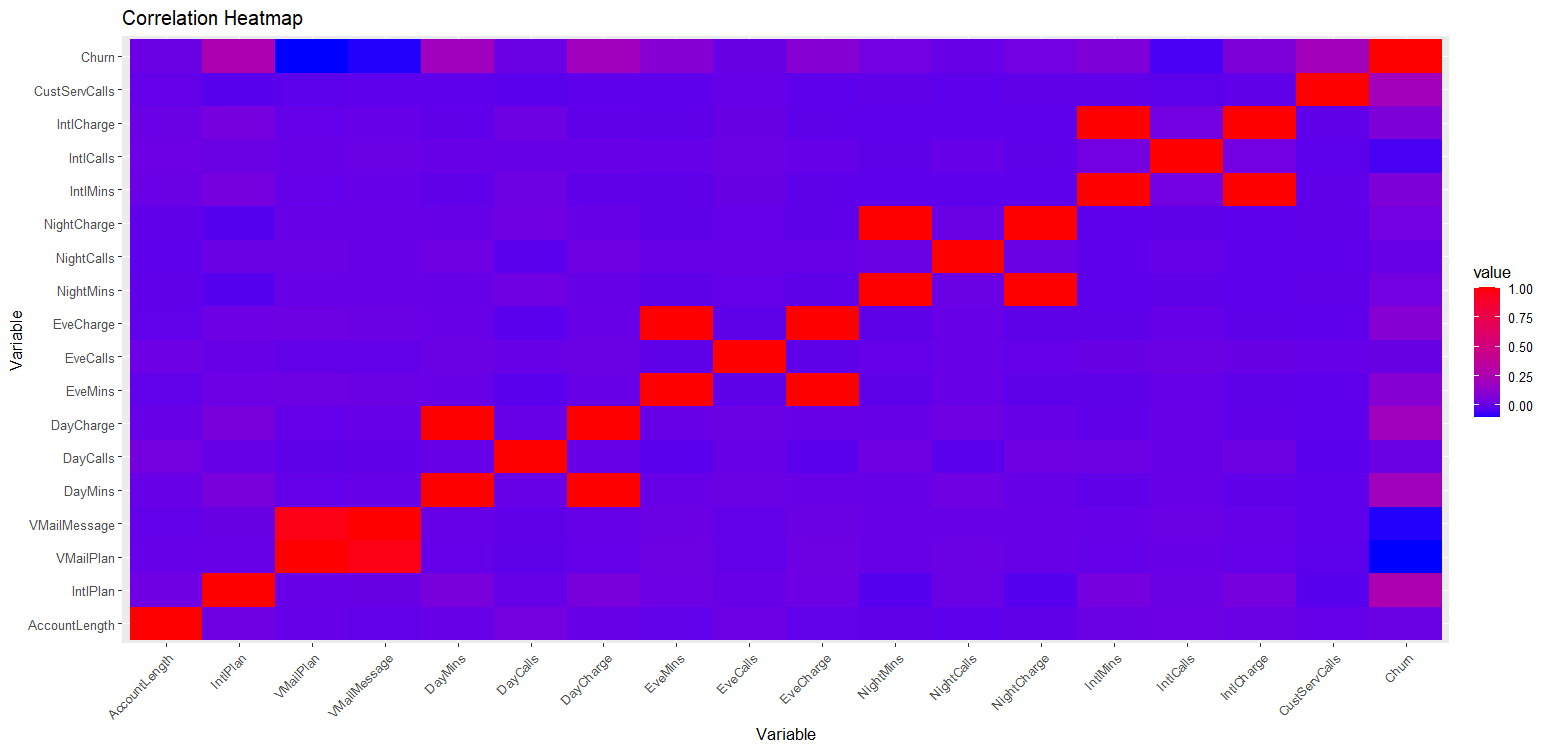


A screen shot of a computer screen

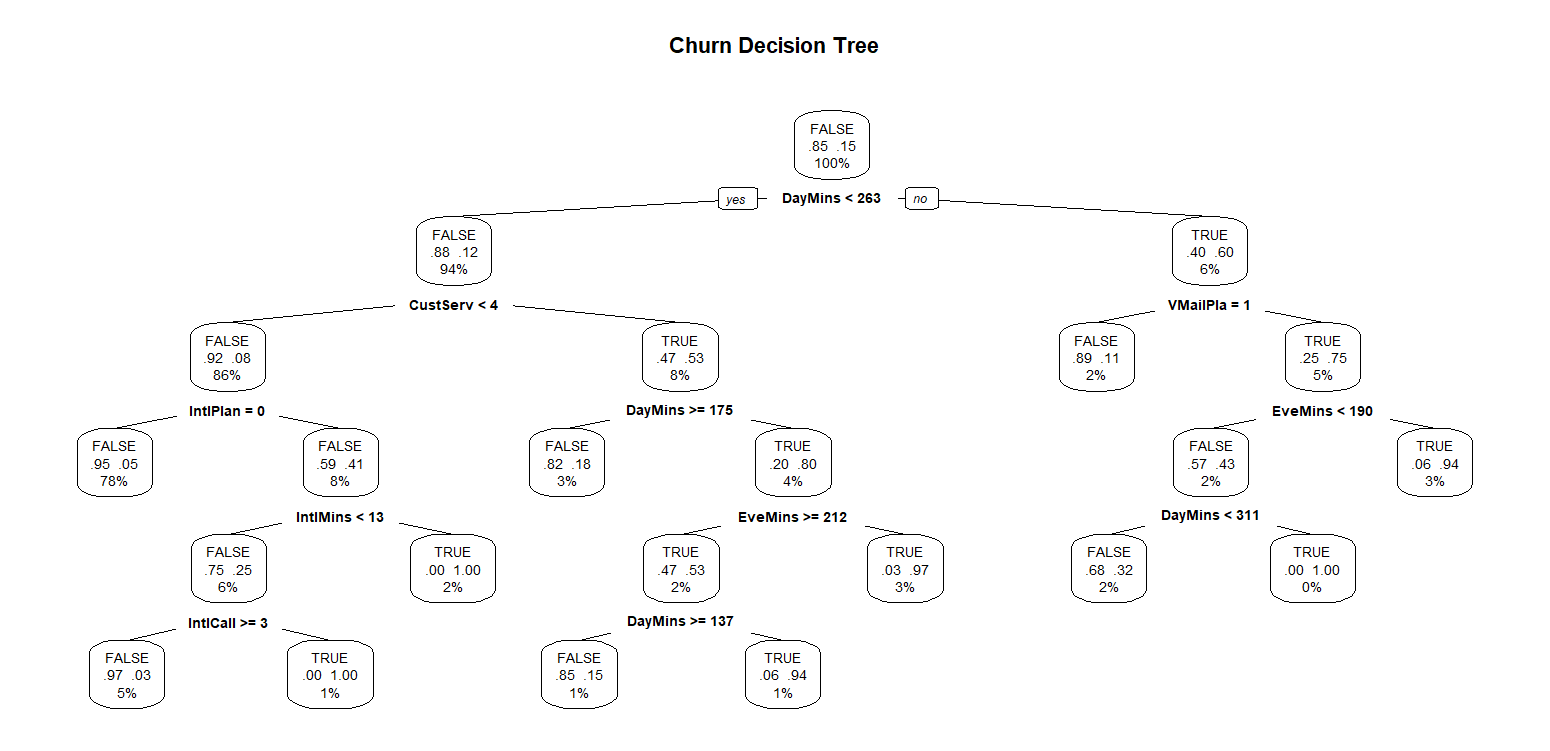
Description automatically generated

I noticed a big relation between number of customer service calls made to number of churn customers.

As for the density of Customer Service Calls, there is significant fluctuation between customers that did not churn, peaking at around 1 call per customer. Whereas those who churned stays relatively level across the board.



There is nearly a 1 to 1 correlation between each time of day’s(day, eve, night, and intl) mins and charge. As well as with VMailPlan and VMailMessage which is all to be expected. What was unexpecting was that no other variables correlated much with any other.



A screenshot of a computer program

Description automatically generatedA blue screen with white text

Description automatically generated

A screenshot of a computer screen

Description automatically generated

For my decision tree, the results of the CP table told me that 9 branches would be the most optimal. The Performance metrics at first glance show this to be true, however it still received a very low sensitivity score. This I believe to be because of the large difference in the number of entries that did not churn vs those that did. To get a better understanding of how it came to this decision, I wanted to see how each variable ranked in importance to the tree’s decisions. As you can see DayMins, IntlPlan, and CustServCall were the most significant. (DayCharge is also ranked high in importance, but as I mentioned before there is a near 1 to 1 correlation)

I also decided to run an SVM to compare against the dtrees results:

A screenshot of a computer

Description automatically generated

It performed slightly worse but still had very similar results.

**Conclusion:**

The results of this analysis were surprising. I had assumed that during my analysis I would find more indications that would lead me to believe whether a customer would churn or not, but I did not. Before using Classification, I could tell there was some correlation between number of minutes a customer used and whether they would churn, but I believe these values to be slightly skewed due to the large difference in number of customers that did not churn compared to those that did. The only true variable that stuck out to me was the number of customer service calls. I was originally surprised with the results before I noticed the sensitivity rate showing that both classifications had many false negatives. I originally tried to create k-clusters but the algorithm never seemed to find its k clusters. I believe if one wanted to get a proper analysis on data of this type, they would need to get a sample with significantly more customers the churned.

**Code:**

# Alex Gibbons

# Churn Project

# CMPT363 - Data Mining

# Dr. Ankur Agrawal

library("tidyverse")

library("rpart")

library("rpart.plot")

library("dplyr")

library("gridExtra")

library("caret")

library("reshape2")

library("e1071")

churn <- as.tibble(read\_csv("C:\\Users\\Kelly\\Desktop\\MCSP24\\CMPT363 - Data Mining\\Assignment8\\churn2.txt",col\_names=TRUE))

churn %>% apply(MARGIN = 2, FUN = unique) %>% str(strict.width = "cut")

names(churn) <- gsub("'| |\\?","",names(churn))

# Barplot churn or no churn

churn\_barplot = ggplot(churn, aes(x = Churn, fill = Churn)) +

geom\_bar() + ggtitle("Churn vs Not Churn Barplot")

print(churn\_barplot)

# Boxplot Day Eve and Night minutes

churn\_long <- churn %>%

pivot\_longer(cols = c(DayMins, EveMins, NightMins), names\_to = "variable")

combined\_plot <- ggplot(churn\_long, aes(x = value, fill = Churn)) +

geom\_boxplot() +

facet\_wrap(~ variable, scales = "free\_x", nrow = 1) +

ggtitle("Day and Night Minutes Boxplots")

print(combined\_plot)

day\_mins\_churn\_data <- churn %>% filter(Churn == TRUE) %>% select(DayMins)

day\_mins\_not\_churn\_data <- churn %>% filter(Churn == FALSE) %>% select(DayMins)

print("Churn:")

print(summary(day\_mins\_churn\_data))

print("Not Churn:")

print(summary(day\_mins\_not\_churn\_data))

eve\_mins\_churn\_data <- churn %>% filter(Churn == TRUE) %>% select(EveMins)

eve\_mins\_not\_churn\_data <- churn %>% filter(Churn == FALSE) %>% select(EveMins)

print("Churn:")

print(summary(eve\_mins\_churn\_data))

print("Not Churn:")

print(summary(eve\_mins\_not\_churn\_data))

night\_mins\_churn\_data <- churn %>% filter(Churn == TRUE) %>% select(NightMins)

night\_mins\_not\_churn\_data <- churn %>% filter(Churn == FALSE) %>% select(NightMins)

print("Churn:")

print(summary(night\_mins\_churn\_data))

print("Not Churn:")

print(summary(night\_mins\_not\_churn\_data))

# Boxplot total minutes

minutes\_tibble <- tibble(churn$DayMins,churn$NightMins,churn$IntlMins,churn$EveCalls)

total\_mins <- rowSums(minutes\_tibble)

churn\_with\_totals <- mutate(churn,TotalMins = total\_mins)

churn\_minutes\_boxplot <- ggplot(churn\_with\_totals, aes(x = TotalMins, fill = Churn)) +

geom\_boxplot() + ggtitle("Total Minutes Boxplot")

print(churn\_minutes\_boxplot)

total\_mins\_churn\_data <- churn\_with\_totals %>% filter(Churn == TRUE) %>% select(TotalMins)

total\_mins\_not\_churn\_data <- churn\_with\_totals %>% filter(Churn == FALSE) %>% select(TotalMins)

print("Churn:")

print(summary(total\_mins\_churn\_data))

print("Not Churn:")

print(summary(total\_mins\_not\_churn\_data))

# Cust Service Calls Boxplot

cust\_serv\_boxplot <- ggplot(churn, aes(x = CustServCalls, fill = Churn)) +

geom\_boxplot() + ggtitle("Customer Service Calls Boxplot")

print(cust\_serv\_boxplot)

cust\_serv\_churn\_data <- churn %>% filter(Churn == TRUE) %>% select(CustServCalls)

cust\_serv\_not\_churn\_data <- churn %>% filter(Churn == FALSE) %>% select(CustServCalls)

print("Churn:")

print(summary(cust\_serv\_churn\_data))

print("Not Churn:")

print(summary(cust\_serv\_not\_churn\_data))

# Cust Serv Density

cust\_serv\_denisity <- ggplot(churn, aes(x = CustServCalls, color = Churn)) +

geom\_density(size = 1) + ggtitle("Customer Service Density")

print(cust\_serv\_denisity)

# Account length Boxplot

length\_boxplot <- ggplot(churn, aes(x = AccountLength, fill = Churn)) +

geom\_boxplot() + ggtitle("Account length Boxplot")

print(length\_boxplot)

# Day Min by State Churn

day\_minutes\_by\_state <- churn %>%

group\_by(State, Churn) %>%

summarise(DayMins = sum(DayMins))

min\_state\_bar <- ggplot(day\_minutes\_by\_state, aes(x = State, y = DayMins, fill = Churn)) +

geom\_bar(stat = "identity", position = "stack") +

labs(title = "Total Day Minutes by State and Churn Status", x = "State", y = "Total Day Minutes") +

scale\_fill\_manual(values = c("skyblue","salmon"))

print(min\_state\_bar)

# Tot Min Histogram

total\_mins\_histo <- ggplot(churn\_with\_totals, aes(x = TotalMins, fill = Churn)) +

geom\_histogram(bins = 20) + ggtitle("Total Minutes Histogram")

print(total\_mins\_histo)

# Heat map

df<-churn

df <- df %>% select(-c("State","AreaCode","Phone"))

df$IntlPlan <- ifelse(df$IntlPlan == "yes",1,0)

df$VMailPlan <- ifelse(df$VMailPlan == "yes",1,0)

correlation\_mat <- cor(df)

melted\_correlation <- melt(correlation\_mat)

heatmap <- ggplot(melted\_correlation, aes(Var1, Var2, fill = value)) +

geom\_tile() +

scale\_fill\_gradient(low = "blue", high = "red") +

labs(title = "Correlation Heatmap", x = "Variable", y = "Variable") +

theme(axis.text.x = element\_text(angle = 45, hjust = 1))

print(heatmap)

# Decision Tree

df$Churn <- factor(df$Churn, levels = c(FALSE,TRUE))

set.seed(1234)

train <- sample(nrow(df),0.7\*nrow(df))

df.train <- df[train,]

df.validate <- df[-train,]

dtree <- rpart(Churn ~ ., data=df.train,method="class",parms=list(split="information"))

print("Dtree results: ")

print(dtree$cptable)

dtree.pruned <- prune(dtree, cp=0.01862464)

prp(dtree.pruned,type = 2, extra = 104, main="Churn Decision Tree")

dtree.pred <- predict(dtree.pruned, df.validate, type ="class")

dtree.perf <- table(df.validate$Churn,dtree.pred, dnn=c("Actual","Predicted"))

print(dtree.perf)

tn <- dtree.perf[1,1]

fp <- dtree.perf[1,2]

fn <- dtree.perf[2,1]

tp <- dtree.perf[2,2]

accuracy <- (tp+tn)/(tp+tn+fp+fn)

error.rate <- (fp+fn)/(tp+tn+fp+fn)

sensitivity <- tp/(tp+fn)

specificity <- tn/(tn+fp)

precision <- tp/(tp+fp)

recall <- tp/(tp+fn)

f.measure <- (2\*precision\*recall)/(precision+recall)

print(paste("Accuracy: ",accuracy))

print(paste("Error rate: ",error.rate))

print(paste("Sensitivity: ",sensitivity))

print(paste("Specificity: ",specificity))

print(paste("Precision: ",precision))

print(paste("Recall: ",recall))

print(paste("F-measure: ",f.measure))

print(varImp(dtree))

svm.model <- svm(Churn ~.,data=df.train)

svm.pred <- predict(svm.model,na.omit(df.validate))

svm.perf <- table(na.omit(df.validate)$Churn,svm.pred,dnn=c("Actual","Predicted"))

print(svm.perf)

# Evaluation metrics

tn <- svm.perf[1,1]

fp <- svm.perf[1,2]

fn <- svm.perf[2,1]

tp <- svm.perf[2,2]

accuracy <- (tp+tn)/(tp+tn+fp+fn)

error.rate <- (fp+fn)/(tp+tn+fp+fn)

sensitivity <- tp/(tp+fn)

specificity <- tn/(tn+fp)

precision <- tp/(tp+fp)

recall <- tp/(tp+fn)

f.measure <- (2\*precision\*recall)/(precision+recall)

print(paste("Accuracy: ",accuracy))

print(paste("Error rate: ",error.rate))

print(paste("Sensitivity: ",sensitivity))

print(paste("Specificity: ",specificity))

print(paste("Precision: ",precision))

print(paste("Recall: ",recall))

print(paste("F-measure: ",f.measure))