Attrition Visualization and Analysis

```
> churn <- read.csv("/Users/joshgrewal/Desktop/churn.txt")
> library(tidyverse)
> library(datasets)
> library(e1071)
> install.packages("corrplot")
> library(corrplot)
> library(ggplot2)
> churnTibble <- as_tibble(churn)
```

```
> churn$Churn <- tolower(churn$Churn) == "true" // Converts
Column to boolean datatype // needed for certain charts
```

ONE

```
> summary(churnTibble)
     State          Account.Length    Area.Code        Phone             Int.l.Plan
 Length:3333        Min.   :  1.0   Min.   :408.0   Length:3333        Length:3333
 Class :character   1st Qu.: 74.0   1st Qu.:408.0   Class :character   Class :character
 Mode  :character   Median :101.0   Median :415.0   Mode  :character   Mode  :character
                    Mean   :101.1   Mean   :437.2
                    3rd Qu.:127.0   3rd Qu.:510.0
                    Max.   :243.0   Max.   :510.0
  VMail.Plan        VMail.Message       Day.Mins       Day.Calls       Day.Charge       Eve.Mins
 Length:3333        Min.   : 0.000   Min.   :  0.0   Min.   :  0.0   Min.   : 0.00   Min.   :  0.0
 Class :character   1st Qu.: 0.000   1st Qu.:143.7   1st Qu.: 87.0   1st Qu.:24.43   1st Qu.:166.6
 Mode  :character   Median : 0.000   Median :179.4   Median :101.0   Median :30.50   Median :201.4
                    Mean   : 8.099   Mean   :179.8   Mean   :100.4   Mean   :30.56   Mean   :201.0
                    3rd Qu.:20.000   3rd Qu.:216.4   3rd Qu.:114.0   3rd Qu.:36.79   3rd Qu.:235.3
                    Max.   :51.000   Max.   :350.8   Max.   :165.0   Max.   :59.64   Max.   :363.7
   Eve.Calls      Eve.Charge      Night.Mins     Night.Calls     Night.Charge       Intl.Mins
 Min.   :  0.0   Min.   : 0.00   Min.   : 23.2   Min.   : 33.0   Min.   : 1.040   Min.   : 0.00
 1st Qu.: 87.0   1st Qu.:14.16   1st Qu.:167.0   1st Qu.: 87.0   1st Qu.: 7.520   1st Qu.: 8.50
 Median :100.0   Median :17.12   Median :201.2   Median :100.0   Median : 9.050   Median :10.30
 Mean   :100.1   Mean   :17.08   Mean   :200.9   Mean   :100.1   Mean   : 9.039   Mean   :10.24
 3rd Qu.:114.0   3rd Qu.:20.00   3rd Qu.:235.3   3rd Qu.:113.0   3rd Qu.:10.590   3rd Qu.:12.10
 Max.   :170.0   Max.   :30.91   Max.   :395.0   Max.   :175.0   Max.   :17.770   Max.   :20.00
   Intl.Calls      Intl.Charge     CustServ.Calls     Churn
 Min.   : 0.000   Min.   :0.000   Min.   :0.000   Length:3333
 1st Qu.: 3.000   1st Qu.:2.300   1st Qu.:1.000   Class :character
 Median : 4.000   Median :2.780   Median :1.000   Mode  :character
 Mean   : 4.479   Mean   :2.765   Mean   :1.563
 3rd Qu.: 6.000   3rd Qu.:3.270   3rd Qu.:2.000
 Max.   :20.000   Max.   :5.400   Max.   :9.000
```

Look at this as sort of an extra graph or intro command. This is used to just get an insight on the dataset and become familiar with it, before using intricate commands to then interpret it heavily. With a summary on the data you can see how each of the columns values look and compare to each other, as the medians and means give you a good sense of what number would be the averages, which you can then base future understanding off of.

TWO

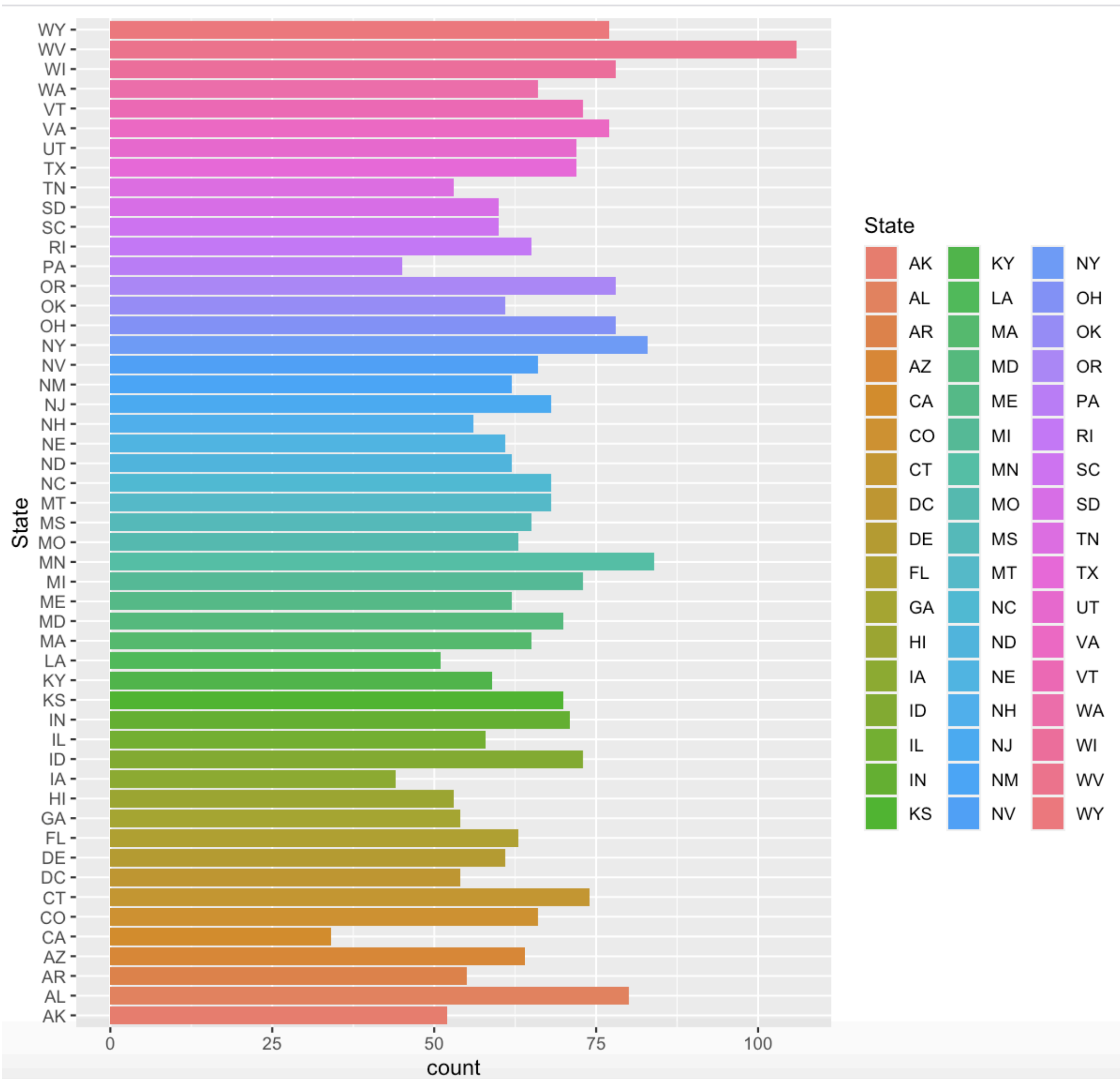
```
> true_churn_by_state <- churn %>% group_by(State) %>%
summarise(True_Churn_Count = sum(Churn))
```

	State	True_Churn_Count
1	NJ	18
2	TX	18
3	MD	17
4	MI	16
5	MN	15
6	NY	15
7	MS	14
8	MT	14
9	NV	14
10	SC	14
11	WA	14
12	KS	13
13	ME	13
14	CT	12
15	AR	11
16	MA	11
17	NC	11
18	OR	11
19	OH	10
20	UT	10
21	WV	10
22	CA	9
23	CO	9
24	DE	9
25	ID	9
26	IN	9
27	NH	9
28	OK	9
29	WY	9
30	AL	8
31	FL	8
32	GA	8
33	KY	8
34	PA	8
35	SD	8
36	VT	8
37	MO	7
38	WI	7
39	ND	6
40	NM	6
41	RI	6

The second graph provides more of a robust interpretation of how many churns there are per state, which will help to interpret hard to read graphs that directly follow this one. The following graph will likely put this information into more of a visual spectacle

THREE

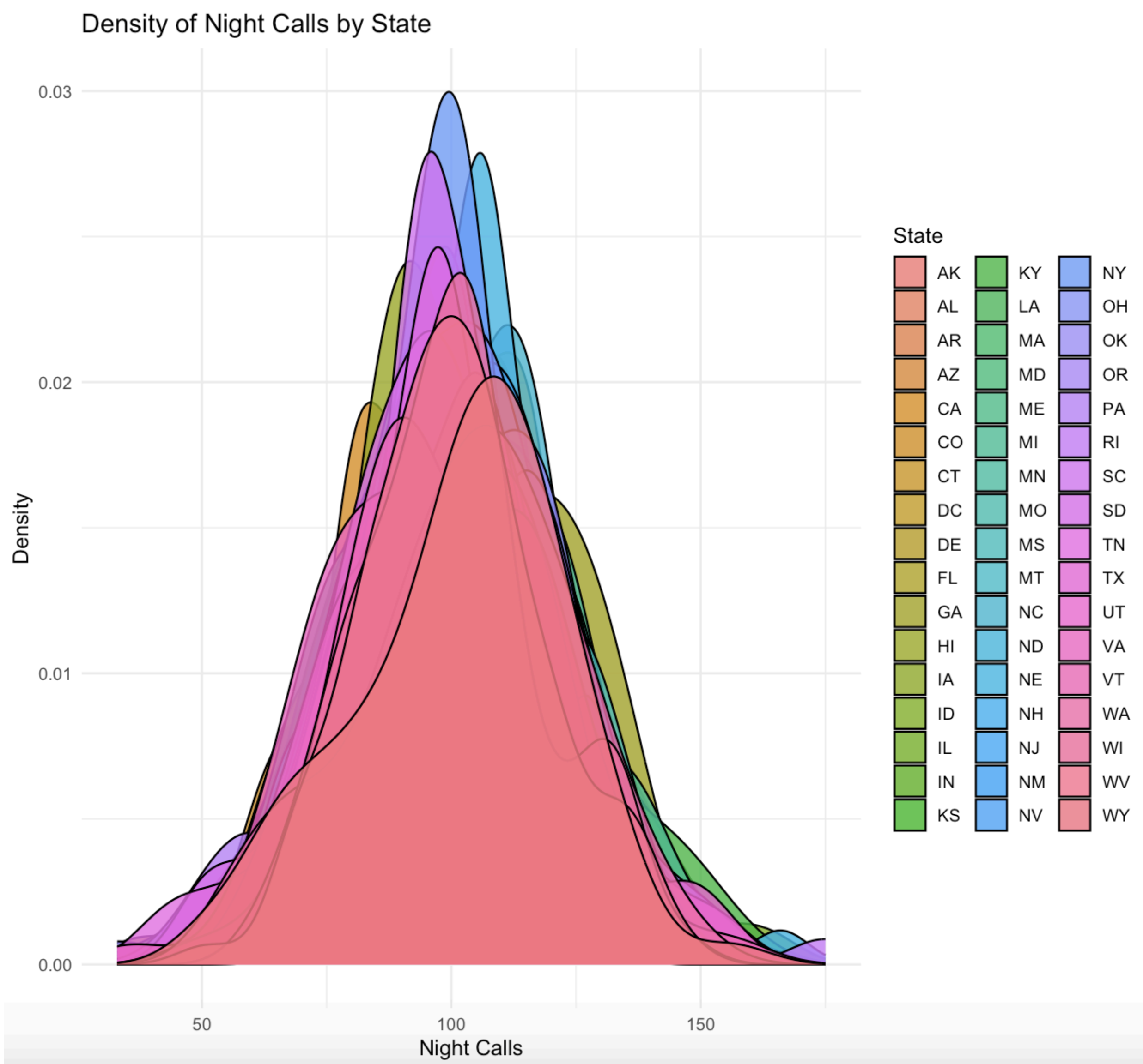
```
> ggplot(churnTibble, aes(y = State, fill = State)) + geom_bar()
```



The bar graph helps to interpret the data by visually displaying the distribution of churned customers across different states. It allows us to quickly identify states with higher or lower numbers of churned customers, providing insights into regional variations in churn rates. Additionally, by sorting the states alphabetically and presenting the data in a clear manner, the bar graph provides easy comparison and identification of patterns or trends in churn behavior across different states. This visualization aids in understanding which states may require more attention in terms of customer retention strategies or where there may be opportunities for improvement.

FOUR

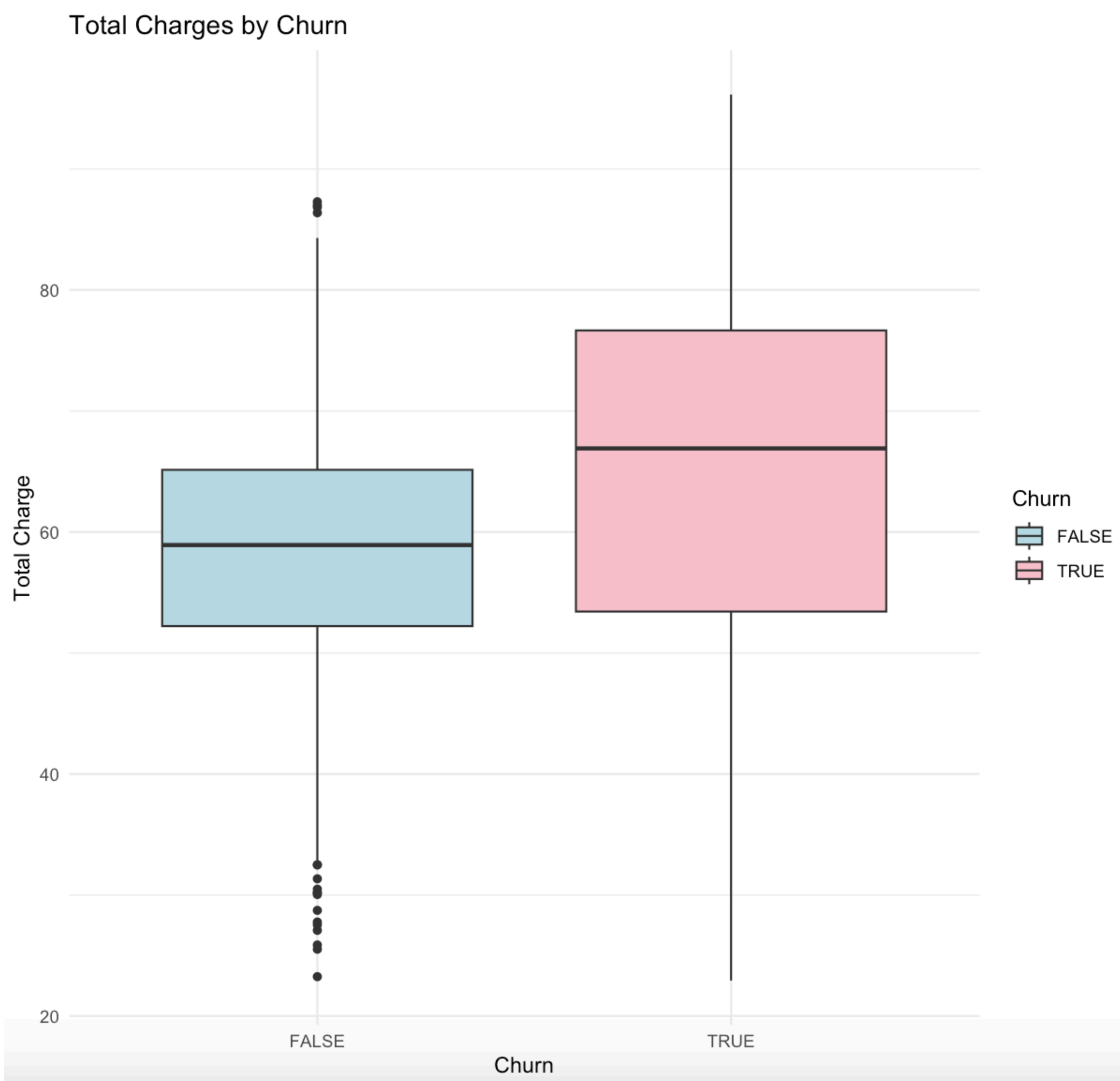
```
> churnDf <- as.data.frame(churn)
> ggplot(churnDf, aes(x = Night.Calls, fill = State)) +
geom_density(alpha = .8) + labs(title = "Density of Night Calls
by State",x = "Night Calls", y = "Density") + theme_minimal()
```



While the density graph comparing night calls across states provides a glimpse into usage patterns, it doesn't conclusively demonstrate a direct link between night calls and churn rates. Churn decisions are influenced by a myriad of factors beyond just call volume, such as overall satisfaction, service quality, and competitive offerings. To truly understand the relationship between night calls and churn rates, a more thorough analysis incorporating these additional variables would be necessary.

FIVE

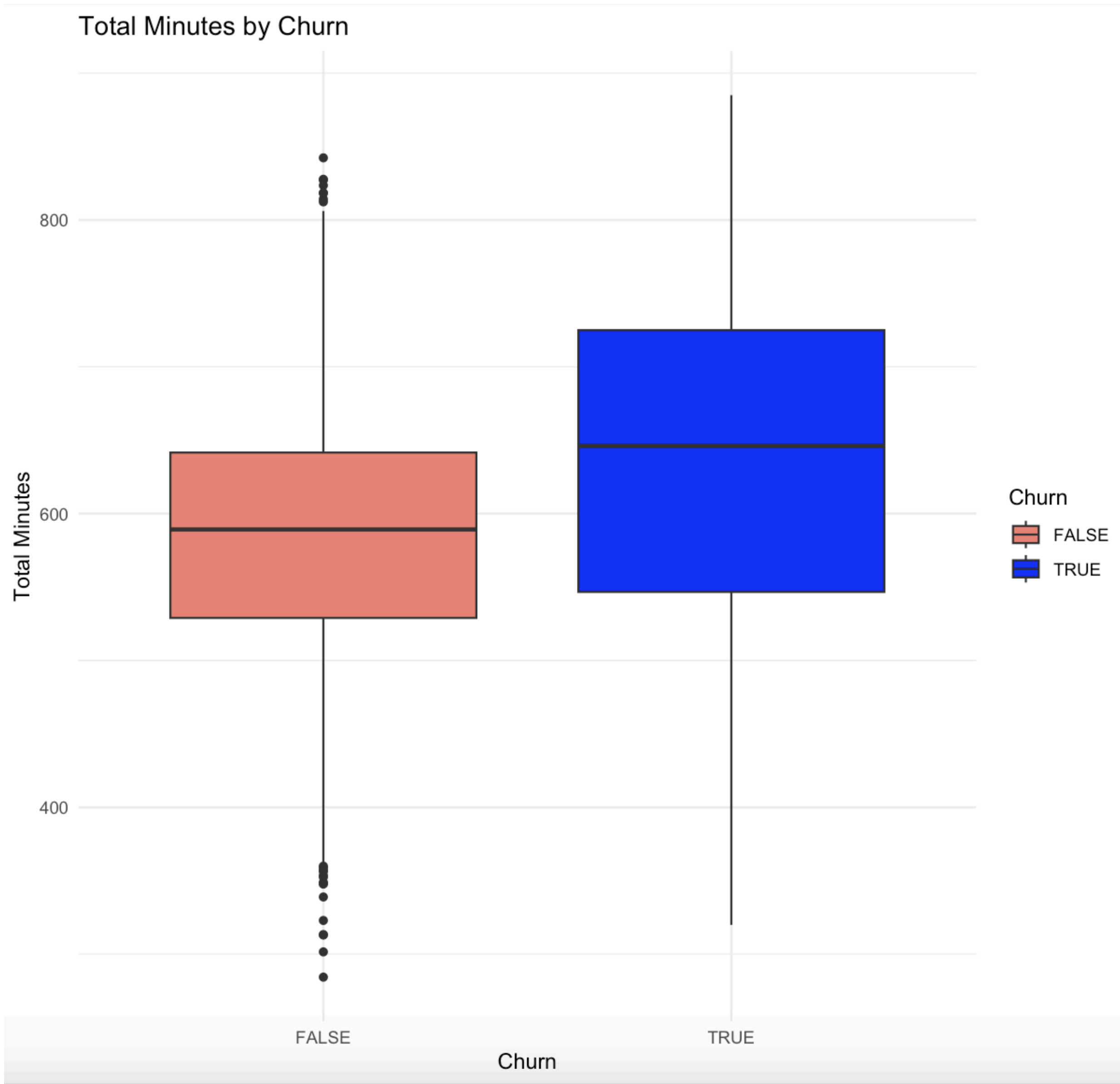
```
> totalCharges <- churn %>% mutate(Total.Charge = Day.Charge +
Eve.Charge + Night.Charge + Intl.Charge)
> charge_comparison <- ggplot(totalCharges, aes(x = Churn, y =
Total.Charge, fill = Churn)) + geom_boxplot() + labs(title =
"Total Charges by Churn",x = "Churn",y = "Total Charge",fill =
"Churn") + scale_fill_manual(values = c("TRUE" = "pink", "FALSE"
= "lightblue")) + theme_minimal()
> print(charge_comparison)
```



Based on the following boxplots, it shows that there may be a correlation between churn and total customer charges. We can see clearly that customers who were charged more in turn had true churn values. As the median for churn in the true section was just below seventy, while the total charges for churn in the false section was a bit lower than 60

SIX

```
> totalMinutes <- churn %>% mutate(Total.Mins = Day.Mins + Eve.Mins + Night.Mins + Intl.Mins)
> Minutes_comparison <- ggplot(totalMinutes, aes(x = Churn, y = Total.Mins, fill = Churn)) + geom_boxplot() + labs(title = "Total Minutes by Churn", x = "Churn", y = "Total Minutes", fill = "Churn") + scale_fill_manual(values = c("TRUE" = "blue", "FALSE" = "salmon")) + theme_minimal()
> print(Minutes_comparison)
```



Although this may seem familiar to the earlier boxplot, this one showcases the total minutes instead of charges by the customer. Similarly, the results are again in favor of the category of being True. Meaning, that customers who spend more time on the phone are more likely to Churn.

SEVEN

```
> df <- churn %>% select(-State, -Account.Length,-VMail.Message
,-Area.Code, -Phone, -Int.l.Plan, -VMail.Plan)
> df <- transform(df, Churn = as.numeric(as.factor(Churn)))
```

```
> df$Churn <- factor(df$Churn., levels = c(2, 1), labels =
c("True", "False"))
> set.seed(1234)
> train <- sample(nrow(df), 0.7 * nrow(df))
> df.train <- df[train, ]
> df.validate <- df[-train, ]
> print(table(df.train$Churn.))
```

```
 True False
  349  1984
```

```
> print(table(df.validate$Churn.))
```

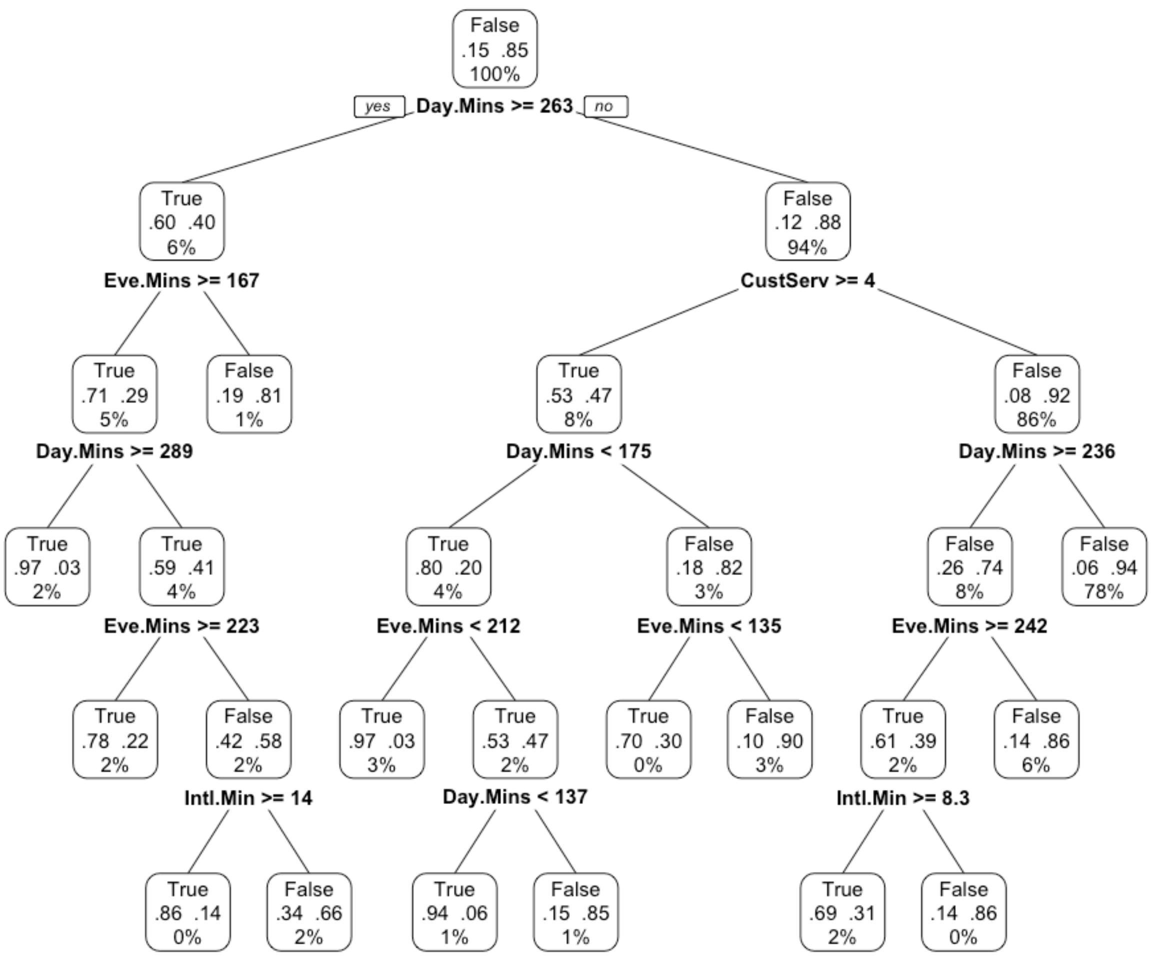
```
 True False
  134   866
```

```
> library(rpart)
> decisiontree <- rpart(Churn. ~ ., data = df.train, method =
"class")
> print(decisiontree$cptable)
```

```
          CP nsplit rel error    xerror       xstd
1 0.08500478      0 1.0000000 1.0000000 0.04936291
2 0.05730659      3 0.7449857 0.8481375 0.04606367
3 0.02005731      4 0.6876791 0.7793696 0.04441612
4 0.01575931      6 0.6475645 0.7593123 0.04391525
5 0.01432665      8 0.6160458 0.7593123 0.04391525
6 0.01146132      9 0.6017192 0.7593123 0.04391525
7 0.01002865     10 0.5902579 0.7449857 0.04355154
8 0.01000000     13 0.5558739 0.7478510 0.04362469
```

```
> decisiontree.pruned <- prune(decisiontree, cp = 0.01)
> library(rpart.plot)
> print(prp(decisiontree.pruned, type = 2, extra = 104, main =
"Decision Tree"))
```

Decision Tree




```
> decisiontree.pred <- predict(decisiontree.pruned, df.validate,
type = "class")
> dtdecisiontreeree.perf <- table(df.validate$Churn.,
decisiontree.pred, dnn = c("Actual", "Predicted"))
> print(decisiontree.perf)
```

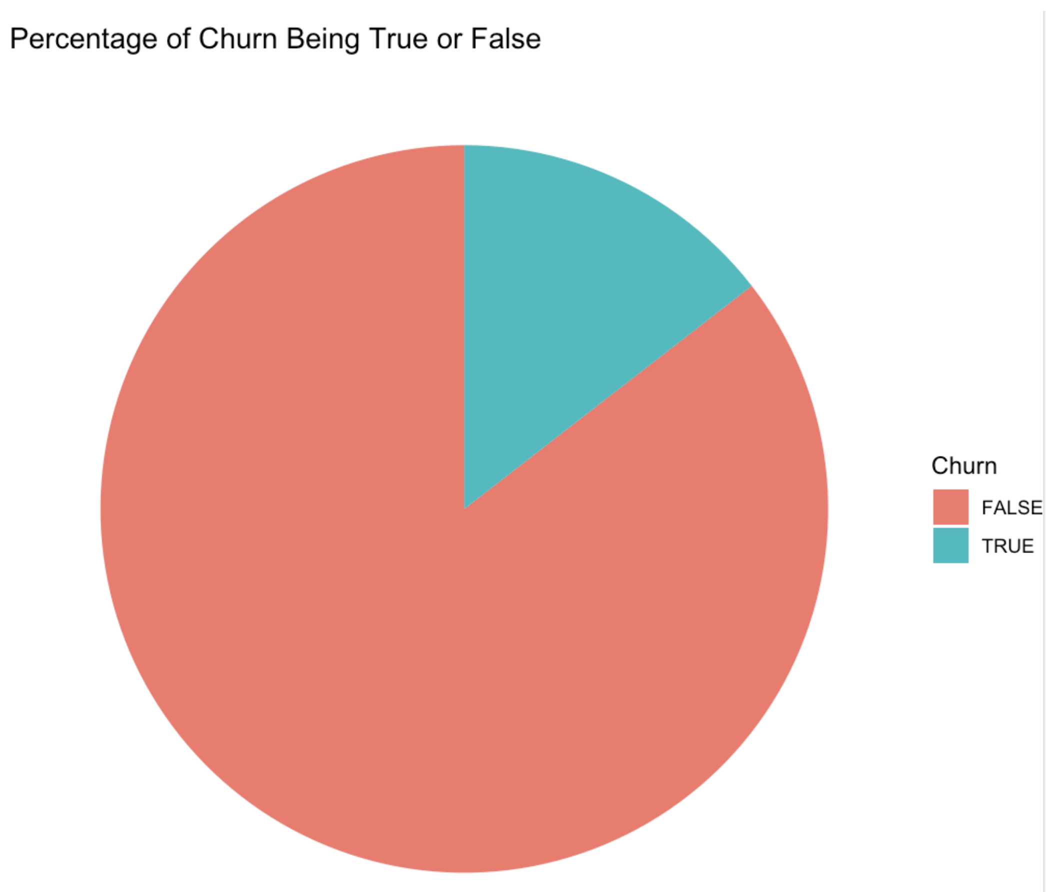
```
       Predicted
Actual  True False
  True    58    76
  False   18   848
```

```
> tn <- decisiontree.perf[1, 1]
> fp <- decisiontree.perf[1, 2]
> fn <- decisiontree.perf[2, 1]
> tp <- decisiontree.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(accuracy)
[1] 0.906
> print(error_rate)
[1] 0.094
> print(sensitivity)
[1] 0.9792148
> print(specificity)
[1] 0.4328358
> print(precision)
[1] 0.9177489
> print(recall)
[1] 0.9792148
> print(f_measure)
[1] 0.947486
```

The decision tree model's output gives us a good idea of how well it predicts churn. With an accuracy of 90.6%, it's pretty good at getting predictions right overall. The sensitivity score of 97.9% means it's great at spotting actual churn cases, while the specificity score of 43.3% suggests it's not as good at recognizing non-churned customers. Its precision score of 91.8% tells us it's reliable when it says a customer will churn, and the recall score of 97.9% means it catches most churned customers. The F-measure, at 94.7%, shows it's doing a decent job overall, balancing precision and recall. These numbers give us a good sense of how well the model is working and how it's handling churn predictions.

Eight

```
> churn_summary <- churn %>% group_by(Churn) %>% summarise(Count = n())
> churn_summary <- mutate(churn_summary, Percent = (Count / sum(Count)) * 100)
> pie_chart <- ggplot(churn_summary, aes(x = "", y = Percent, fill = factor(Churn))) + geom_bar(width = 1, stat = "identity") + coord_polar("y", start = 0) + labs(title = "Percentage of Churn Being True or False", fill = "Churn", y = NULL) + theme_void()
> print(pie_chart)
```



The pie graph provides a sort of breath of fresh air, as it returns to the original dataset and provides a sort of aerial view on how the split is between churned and non churned customers. As you can see the split is smaller than ¼, which showcases just how difficult it is to discern whether churn is true or false from simply looking at the data. This is why predictive algorithms and classification is so important

Nine

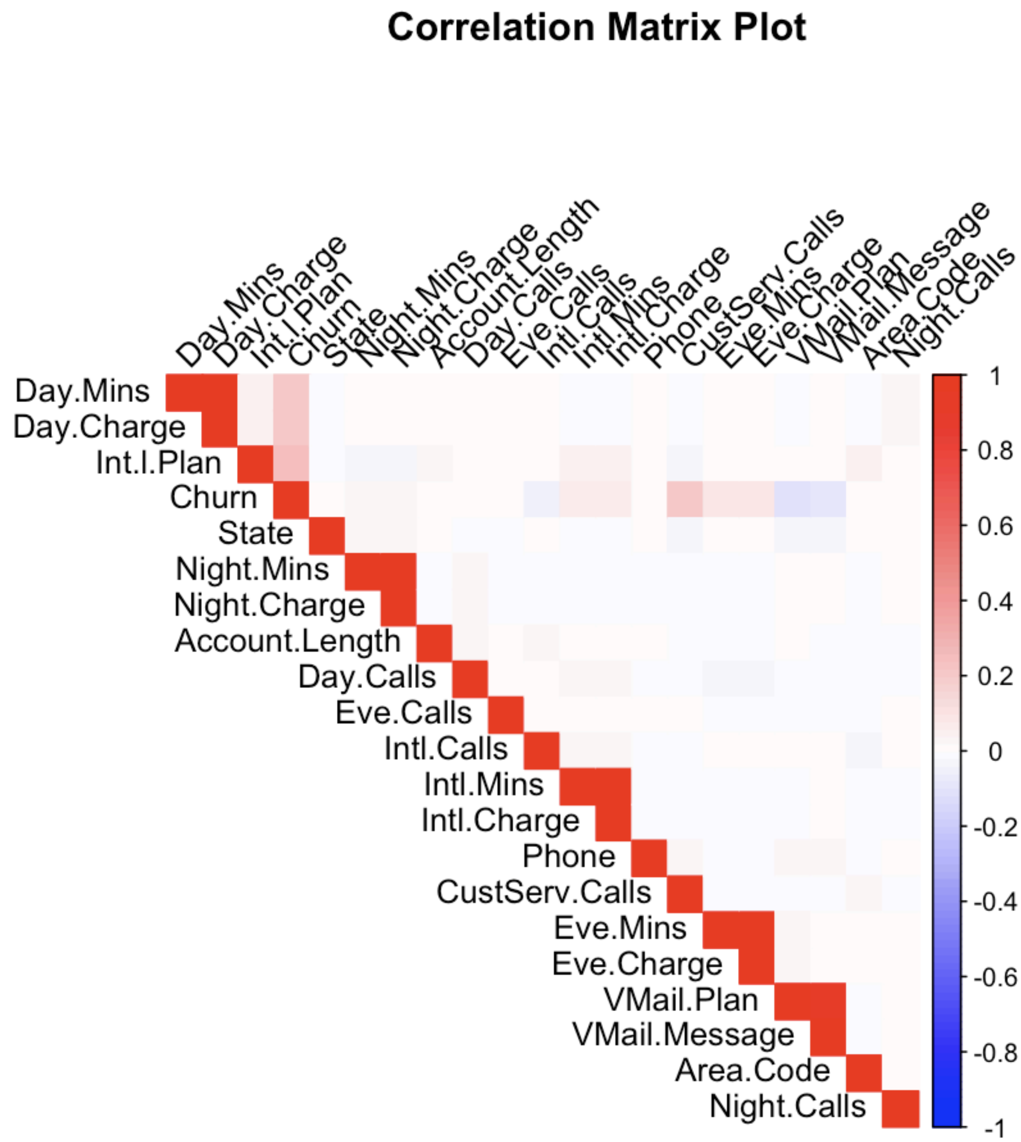
```
> nb.model <- naiveBayes(Churn.~.,data = df.train)
> nb.pred <- predict(nb.model, df.validate)
> nb.perf <- table(df.validate$Churn., nb.pred, dnn=c("Actual",
"Predicted"))
> print(nb.perf)
        Predicted
Actual   True False
  True     48    86
  False    34   832
> tn <- nb.perf[1, 1]
> fp <- nb.perf[1, 2]
> fn <- nb.perf[2, 1]
> tp <- nb.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(accuracy)
[1] 0.88
> print(error_rate)
[1] 0.12
> print(sensitivity)
[1] 0.960739
> print(specificity)
[1] 0.358209
```

```
> print(precision)
[1] 0.9063181
> print(recall)
[1] 0.960739
> print(f_measure)
[1] 0.9327354
```

The results from the Naive Bayesian method show that it's doing a pretty decent job at predicting churn. It correctly predicted churn for 88% of the cases, which is not bad. However, it seems to be better at catching customers who actually churn (96.1% of them) than identifying those who don't churn (35.8% accuracy). This means it might mistakenly label some customers as churners when they're not. Still, when it says a customer will churn, it's right about 90.6% of the time, which is pretty reliable. Overall, it seems to be doing okay, with a balance between how many churned customers it catches and how accurate those predictions are.

TEN

```
> churn <- read.csv("/Users/joshgrewal/Desktop/churn.txt") //
reset the churn dataset
> churn <- transform(churn, Churn =
as.numeric(as.factor(Churn)))
> churn <- transform(churn, State =
as.numeric(as.factor(State)))
> churn <- transform(churn,  Int.l.Plan = as.numeric(as.factor(
Int.l.Plan)))
> churn <- transform(churn,  VMail.Plan = as.numeric(as.factor(
VMail.Plan)))
> View(churn)
> churn <- transform(churn,  Phone =
as.numeric(as.factor(Phone)))
> correlation_matrix <- cor(churn)
> title("Correlation Matrix Plot")
> corrplot(correlation_matrix, method = "color", type = "upper",
order = "hclust", tl.col = "black", tl.srt = 45, col =
colorRampPalette(c("blue", "white", "red"))(100))
> title("Correlation Matrix Plot")
```



Blue hues indicate negative correlations, red hues indicate positive correlations, and white regions signify no correlation. This color scheme enables quick identification of strong correlations (both positive and negative) and highlights potential patterns or associations within the data. As a result, you can easily grasp the correlation between variables, aiding in decision making processes such as feature selection for predictive modeling or identifying potential areas for further analysis.