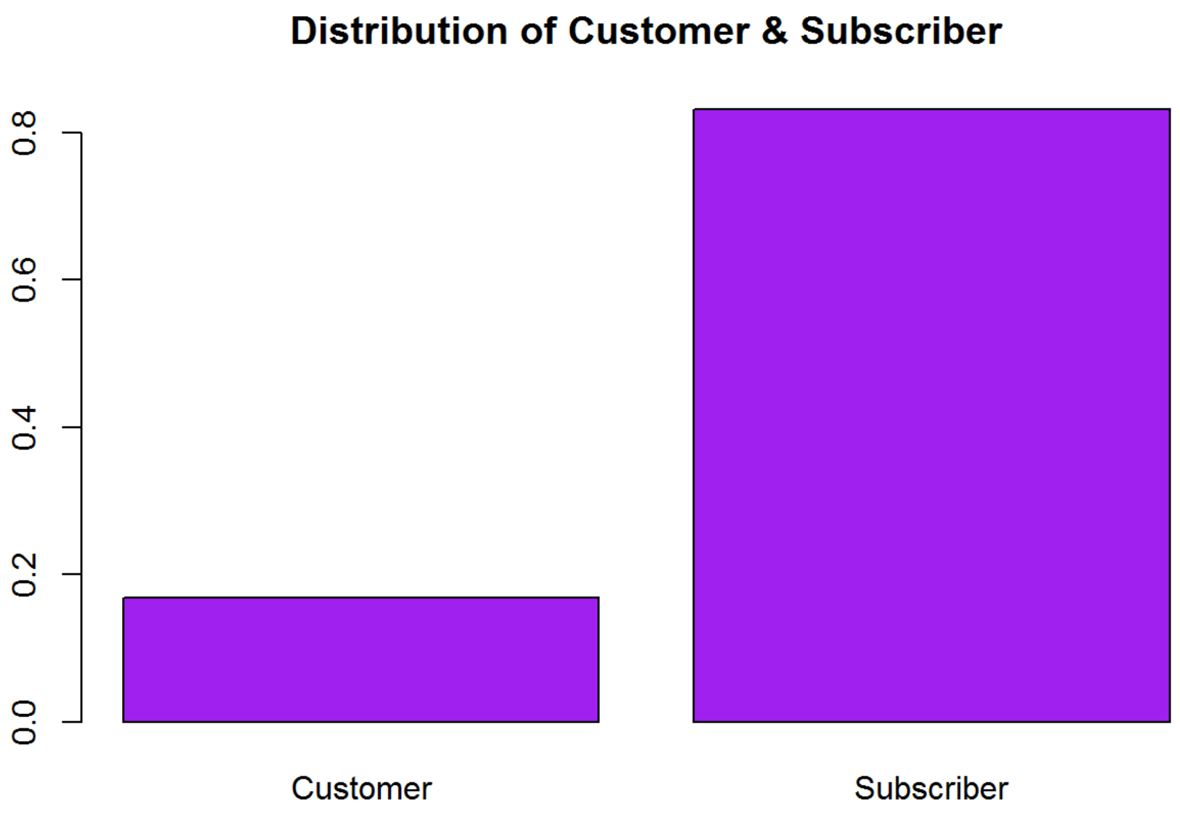
SF Bike Share

Sourav Ghosh

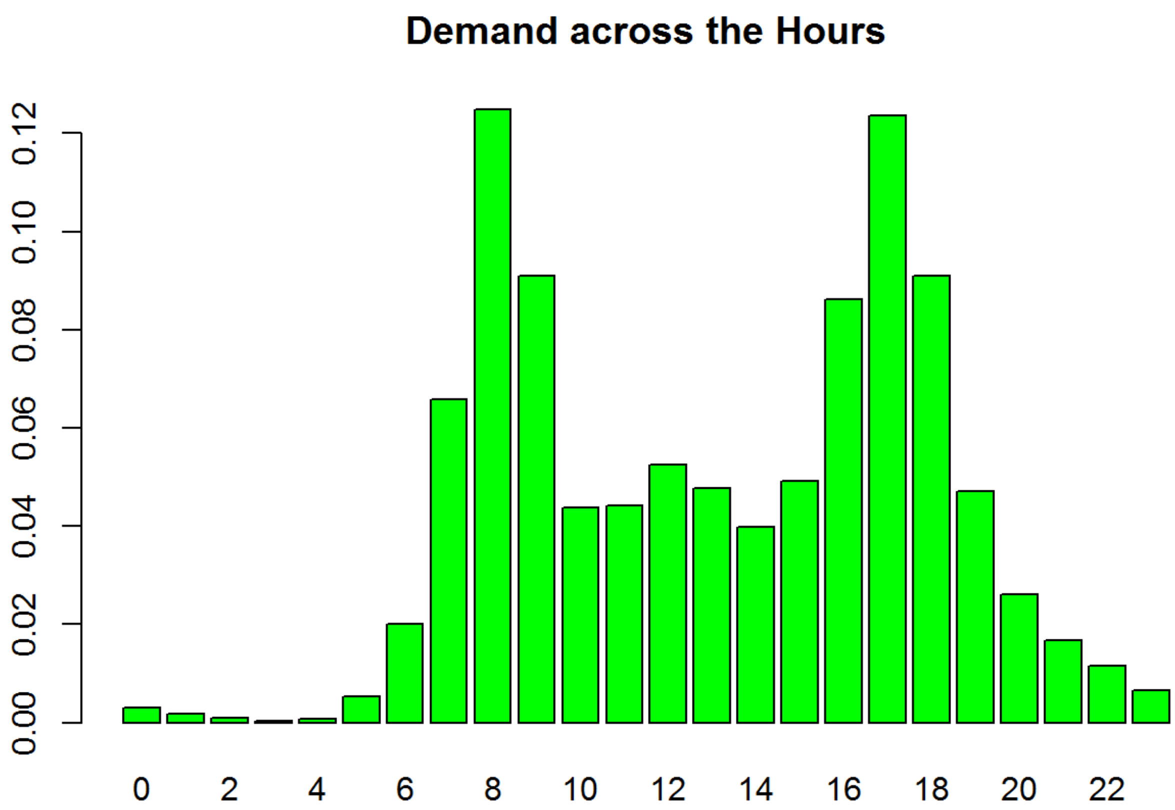
August 28, 2017

```
library(sqldf)
library(lubridate)
library(ggplot2)
library(dplyr)
library(chron)
library(classInt)
```

```
df <- read.csv("C:/Users/SGHOSH13/Documents/U-Exercise/data/201408_trip_data (5).cs
v", stringsAsFactors = F, check.names = F)

names(df) <- c("Trip_ID","Duration", "Start_Date", "Start_Station", "Start_Termina
l",
               "End_Date", "End_Station", "End_Terminal", "BikeID", "Subscriber_Ty
pe",
               "Zip_Code")

#Double-checking for missing values (though it is mentioned as there isn't any)
sapply(df,function(x) sum(is.na(x)))

#Extract Date and Time and store under different Variables

df$Start_Date <- parse_date_time(df$Start_Date, '%m/%d/%Y %H:%M', exact = TRUE)
df$Start_Date1 <- as.Date(df$Start_Date)
df$Start_Hour <- hour(df$Start_Date)

df$End_Date <- parse_date_time(df$End_Date, '%m/%d/%Y %H:%M', exact = TRUE)
df$End_Date1 <- as.Date(df$End_Date)
df$End_Hour <- hour(df$End_Date)

#Finding which day a particular date falls into
df$StartDayofWk <- wday(df$Start_Date1, label=TRUE, abbr = F)

#Finding whether the day falls on Weekday or Weekend
df$isWeekend = chron::is.weekend(df$Start_Date1)
df$isWeekend <- factor(df$isWeekend,levels=c(T,F), labels=c("Weekend","Weekday"))

#Ratio of Customers to Subscribers
barplot(prop.table(table(df$Subscriber_Type)), col = "purple", main = "Distribution
of Customer & Subscriber")
```

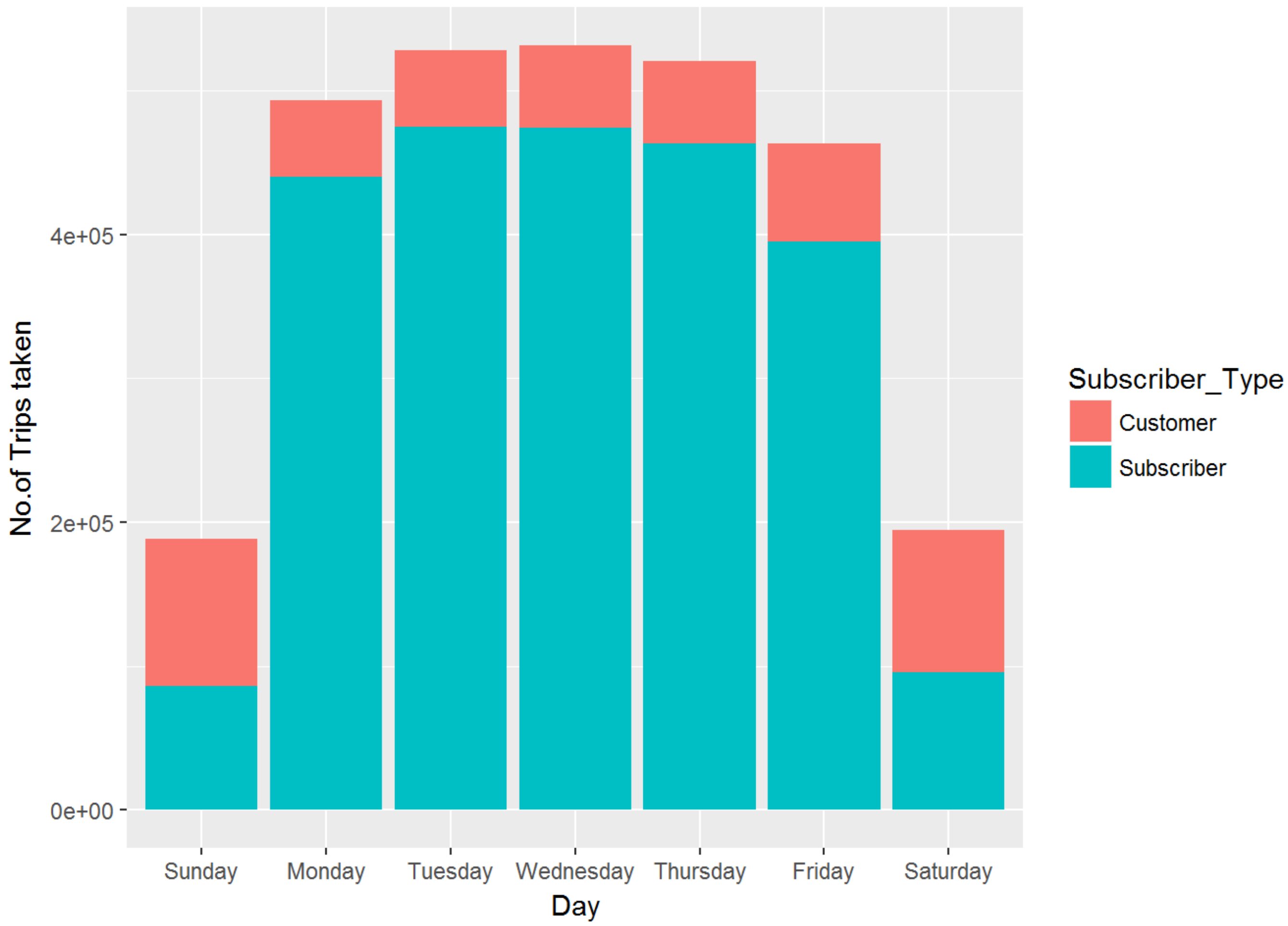



```r
#Seeing Proportion of Demand across Hours
barplot(prop.table(table(df$Start_Hour)), col = "green", main = "Demand across the Hours")
```

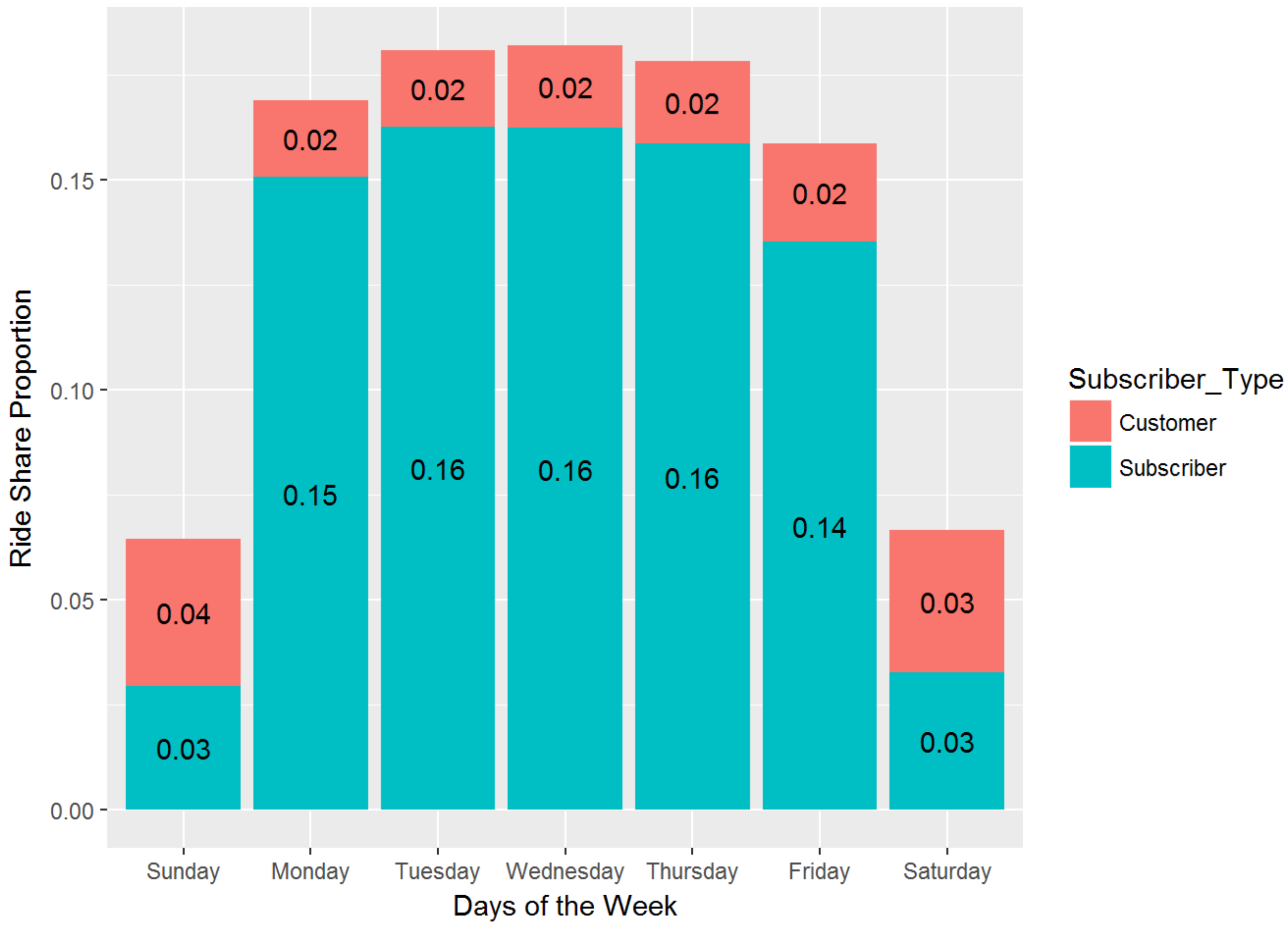


Demand across the Hours
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0
2
4
6
8
10
12
14
16
18
20
22

```
#Average Trip Duration in Minutes by Subscriber Type
customerType <- sqldf("SELECT Subscriber_Type, AVG(Duration)/60 AS Average_Trip_Tim
e_mins
                       FROM df
                       GROUP BY Subscriber_Type")
customerType

#Find out top Prime Starting Station by no. of Trips starting from it
hotStartZone <- sqldf("SELECT Start_Station, COUNT(*) AS Total_Trips
                       FROM df
                       GROUP BY Start_Station
                       ORDER BY Total_Trips DESC")

head(hotStartZone)

#Find out top Prime Ending Station by no. of Trips starting from it
hotEndZone <-  sqldf( "SELECT End_Station, COUNT(*) AS Total_Trips
                       FROM df
                       GROUP BY End_Station
                       ORDER BY Total_Trips DESC")

head(hotEndZone)

#Some plots for Exploratory Data Analysis which has been Summarised towards the end
of the Script

plt1 <- ggplot(df, aes(x=StartDayofWk, y = length(df), fill=Subscriber_Type)) +
        geom_bar(stat="identity") +
        xlab("Day") +
        ylab("No.of Trips taken")

plt1
```

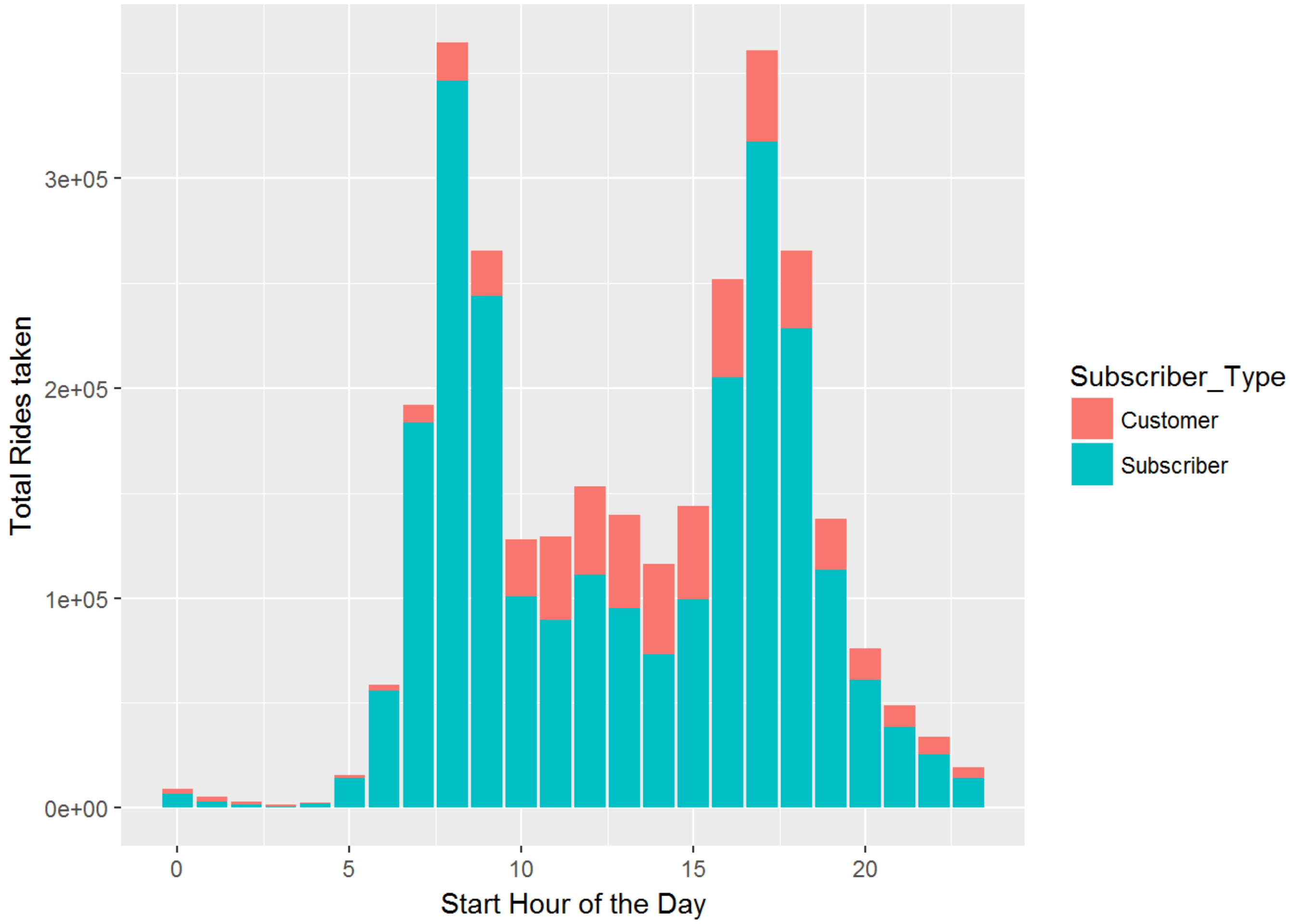



```
stackBar1 <- df %>%
                  count(StartDayofWk, Subscriber_Type) %>%
                  mutate(prop = n/sum(n))

plt2 <- ggplot(stackBar1, aes(x=StartDayofWk, y=prop, fill=Subscriber_Type)) +
        geom_bar(stat="identity") +
        xlab("Days of the Week") +
        ylab("Ride Share Proportion") +
        geom_text(aes(label=round(prop,2)) , position = position_stack(vjust = 0.5
))

plt2
```

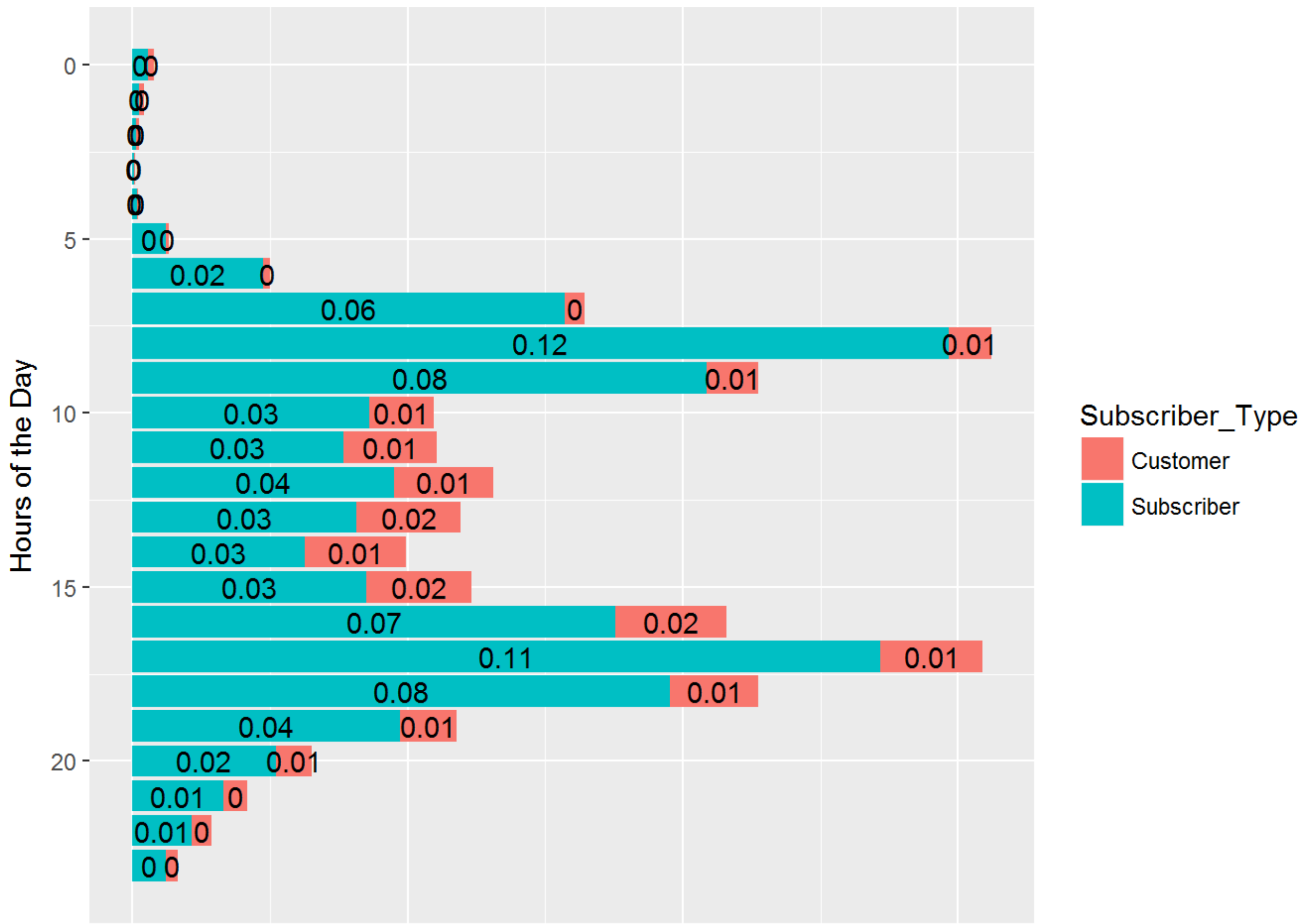



```
stackBar2 <- df %>%
                  count(isWeekend, Subscriber_Type) %>%
                  mutate(prop = n/sum(n))

plt3 <- ggplot(stackBar2, aes(x=isWeekend, y=prop, fill=Subscriber_Type)) +
        geom_bar(stat="identity") +
        xlab("WeekDays vs Weekends") +
        ylab("Ride Share Proportion") +
        geom_text(aes(label=round(prop,2)) , position = position_stack(vjust = 0.5
))

plt3
```

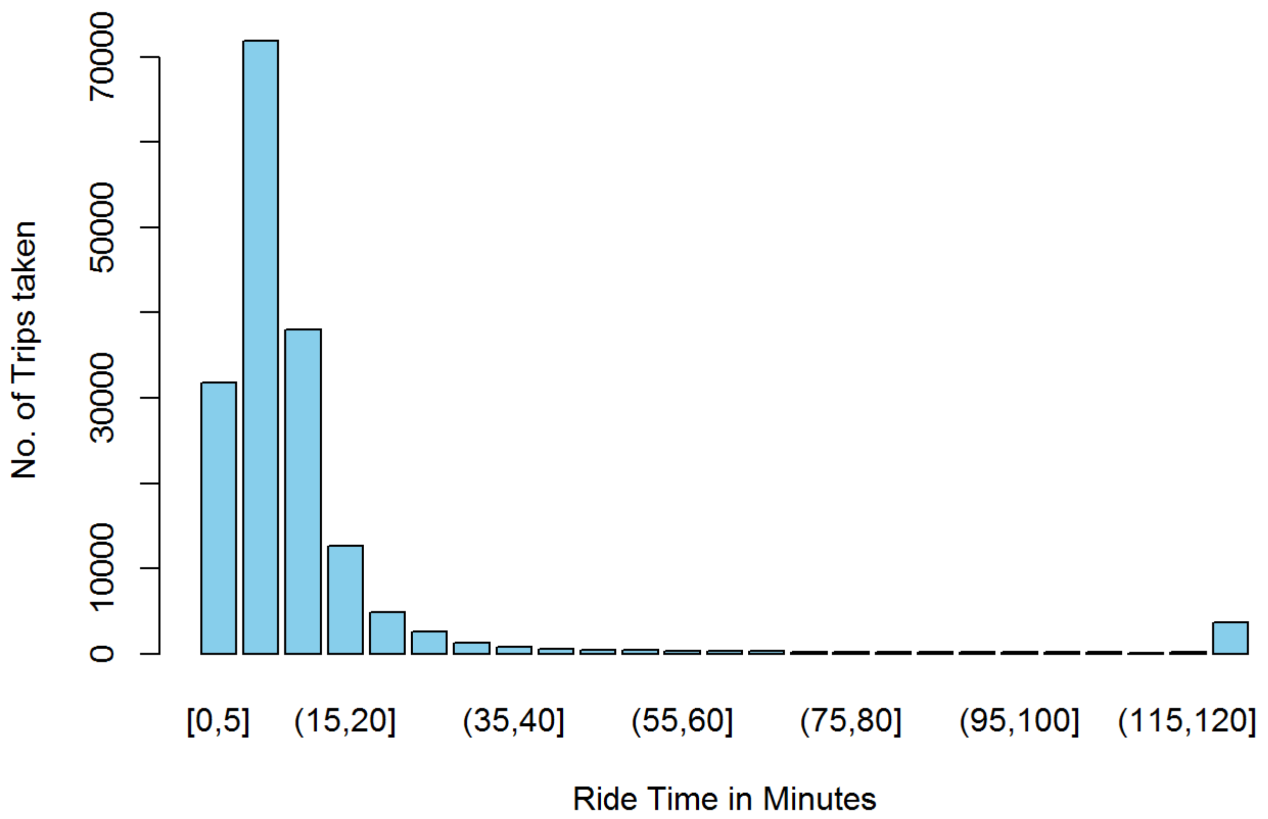

```r
# Analysis for Trips taken across the Hours of a Day

plt4 <- ggplot(df, aes(x=Start_Hour, y = length(df), fill=Subscriber_Type)) +
        geom_bar(stat="identity") +
        xlab("Start Hour of the Day") +
        ylab("Total Rides taken")

plt4
```



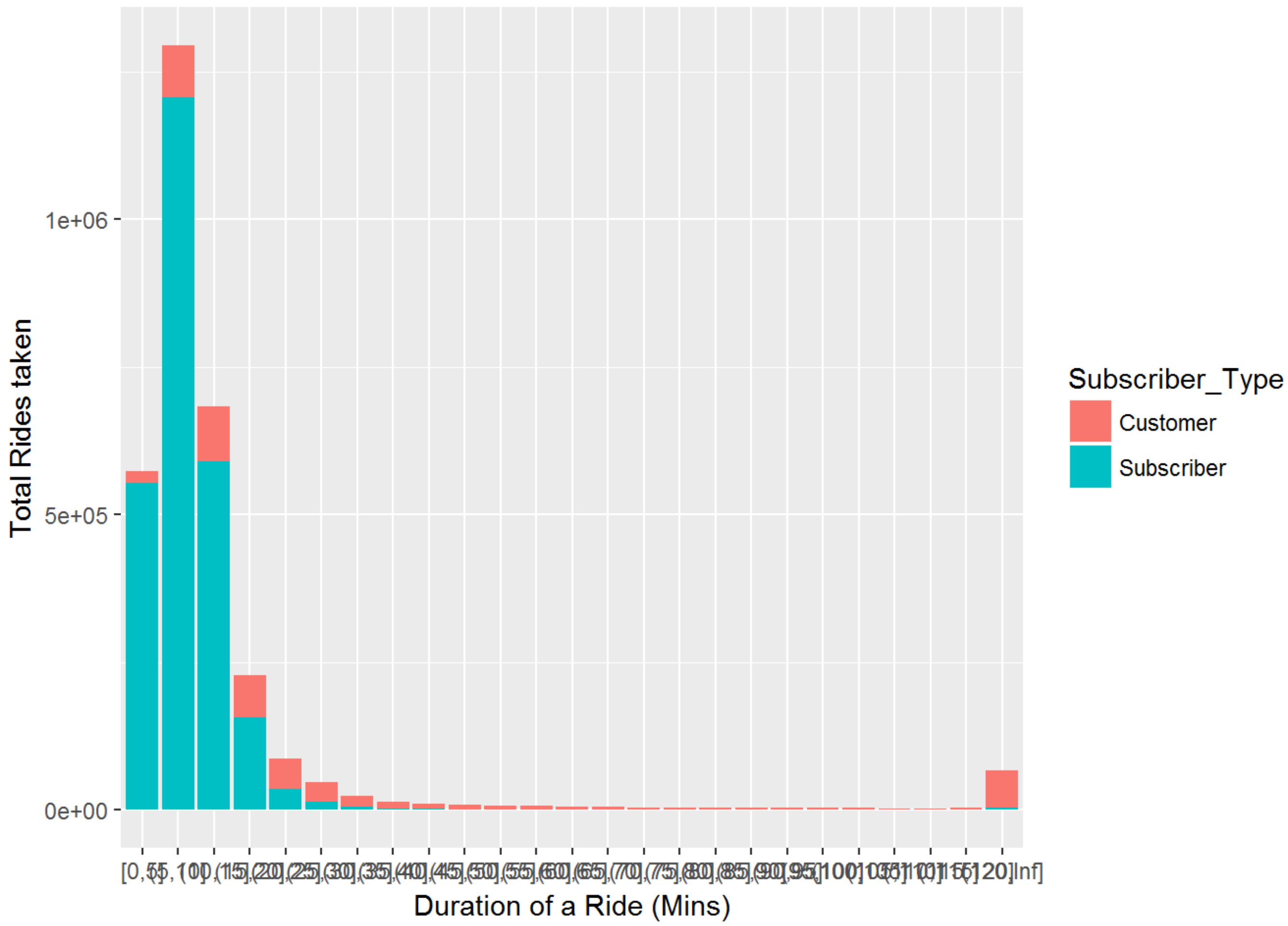

```
stackBar3 <- df %>%
                  count(Start_Hour, Subscriber_Type) %>%
                  mutate(prop = n/sum(n))

plt5 <- ggplot(stackBar3, aes(x=Start_Hour, y=prop, fill=Subscriber_Type)) +
        geom_bar(stat="identity") +
        xlab("Hours of the Day") +
        ylab("Ride Share Proportion") +
        geom_text(aes(label=round(prop,2)) , position = position_stack(vjust = 0.5
)) +
        theme(axis.title.x=element_blank(),
              axis.text.x=element_blank(),
              axis.ticks.x=element_blank()) +
        coord_flip() + scale_x_reverse()

plt5
```



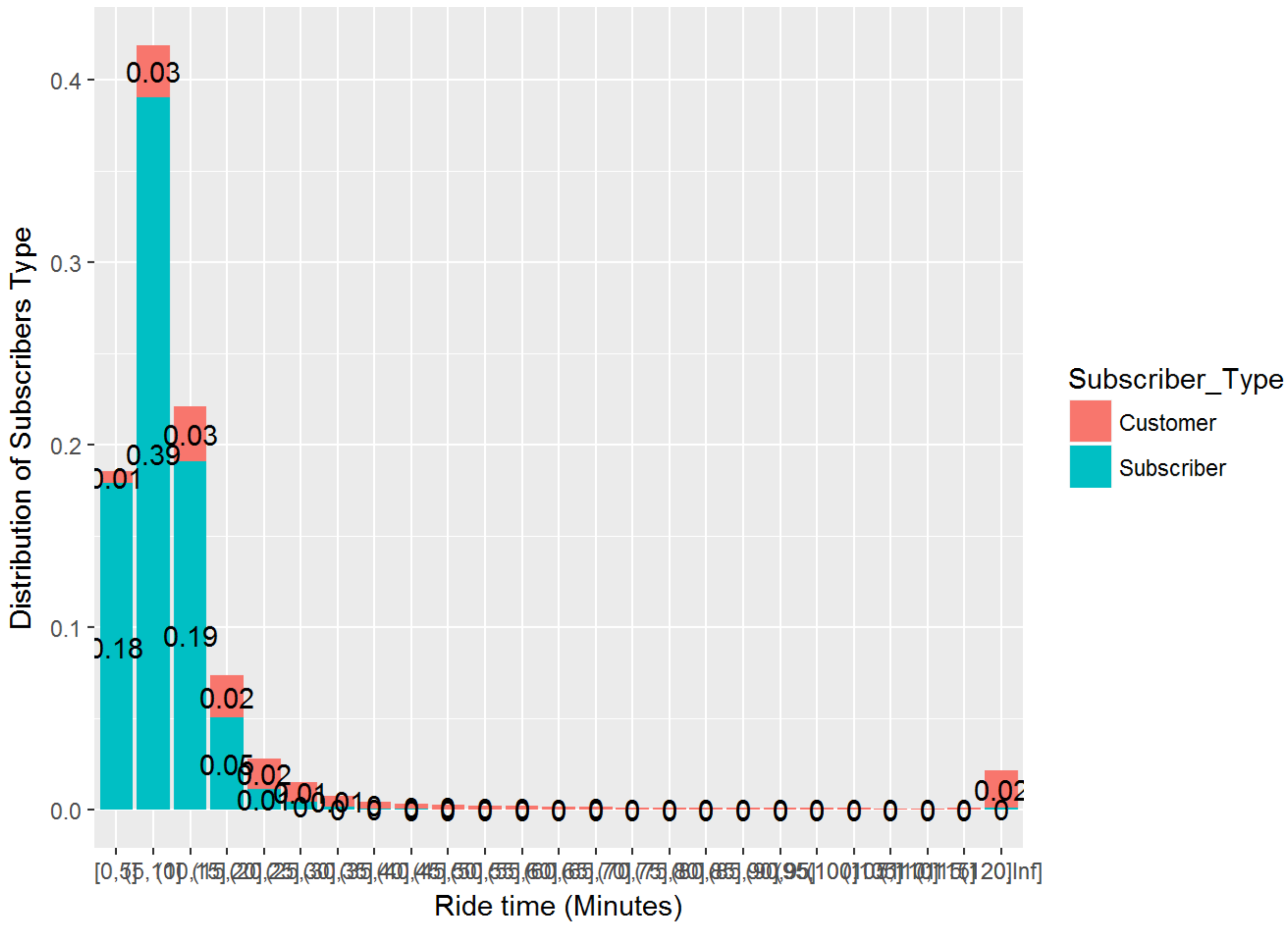

```
#Analysis according to Length of the Trip
df$byTime <- cut((df$Duration)/60, c(seq(0,120,5),Inf), include.lowest=TRUE)

count.duration <- tapply((df$Duration)/60, df$byTime, length)

barplot(count.duration, xlab = "Ride Time in Minutes", ylab = "No. of Trips taken",
col = "skyblue")
```



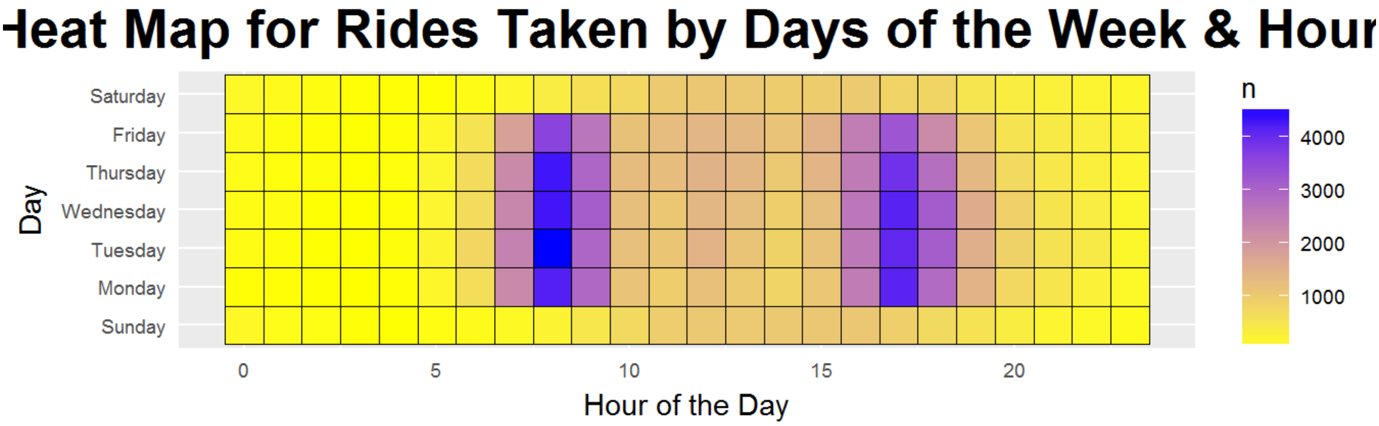

```
plt6 <- ggplot(df, aes(x=byTime, y = length(df), fill=Subscriber_Type)) +
        geom_bar(stat="identity") +
        xlab("Duration of a Ride (Mins)") +
        ylab("Total Rides taken")

plt6
```



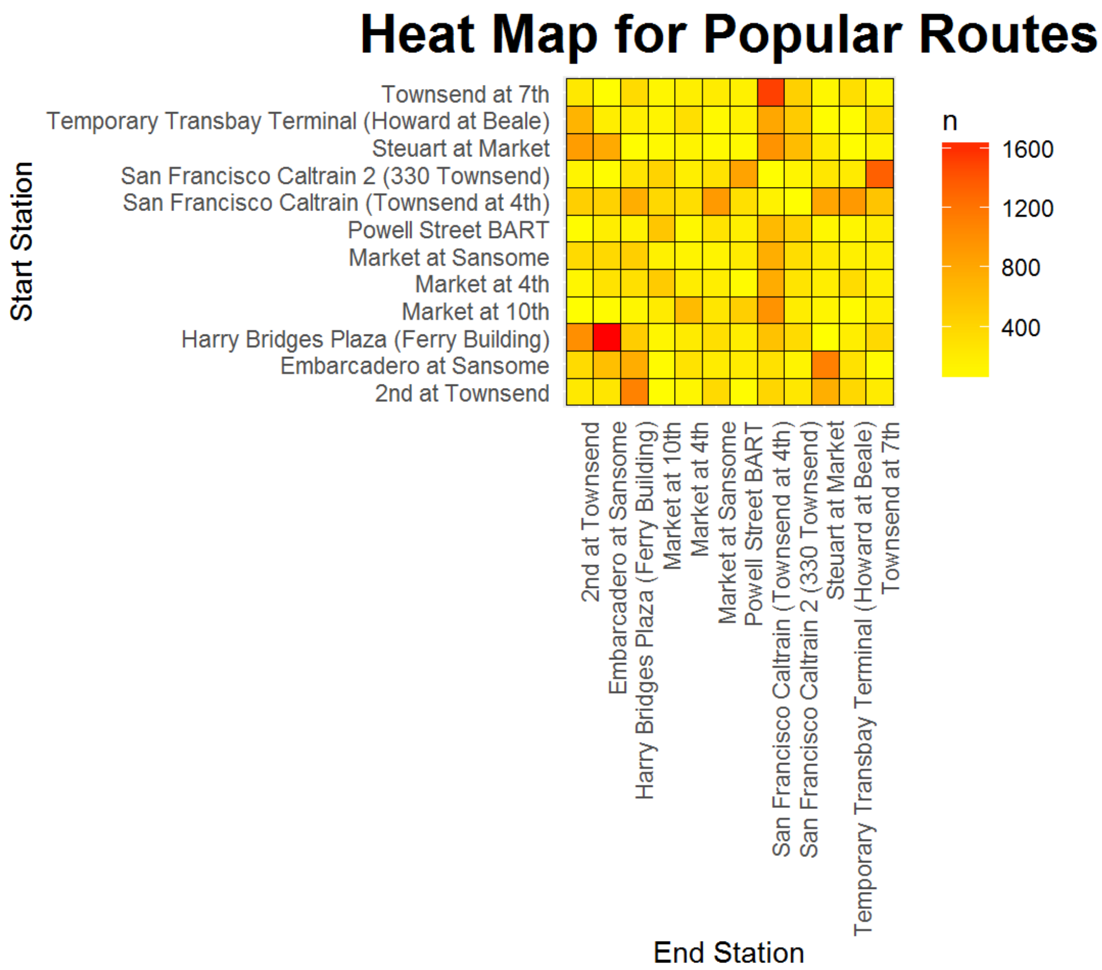

```
#Distribution of Ride time vs Subscriber Type
stackBar3 <- df %>%
                  count(byTime, Subscriber_Type) %>%
                  mutate(prop = n/sum(n))

plt7  <-  ggplot(stackBar3, aes(x=byTime, y=prop, fill=Subscriber_Type)) +
          geom_bar(stat="identity") +
          xlab("Ride time (Minutes)") +
          ylab("Distribution of Subscribers Type") +
          geom_text(aes(label=round(prop,2)) , position = position_stack(vjust = 0
.5))
plt7
```




```
# Heat Map for Popular Routes

# No. of Trips Starting at which hour by days of the week
weekDays <- count(df, StartDayofWk, Start_Hour)

hm1 <-  ggplot(weekDays, aes(x=Start_Hour, y=StartDayofWk, fill=n)) +
        geom_tile(color="black", size=0.05) +
        coord_equal() + scale_fill_gradient(low = "yellow", high = "blue") +
        labs(x="Hour of the Day", y="Day", title="Heat Map for Rides Taken by Days
of the Week & Hour") +
        theme(plot.title=element_text(size = 20, face = "bold", hjust = 0.5)) +
        theme(axis.ticks=element_blank()) +
        theme(axis.text=element_text(size=7)) +
        theme(legend.title=element_text(size=10)) +
        theme(legend.text=element_text(size=7))

hm1
```



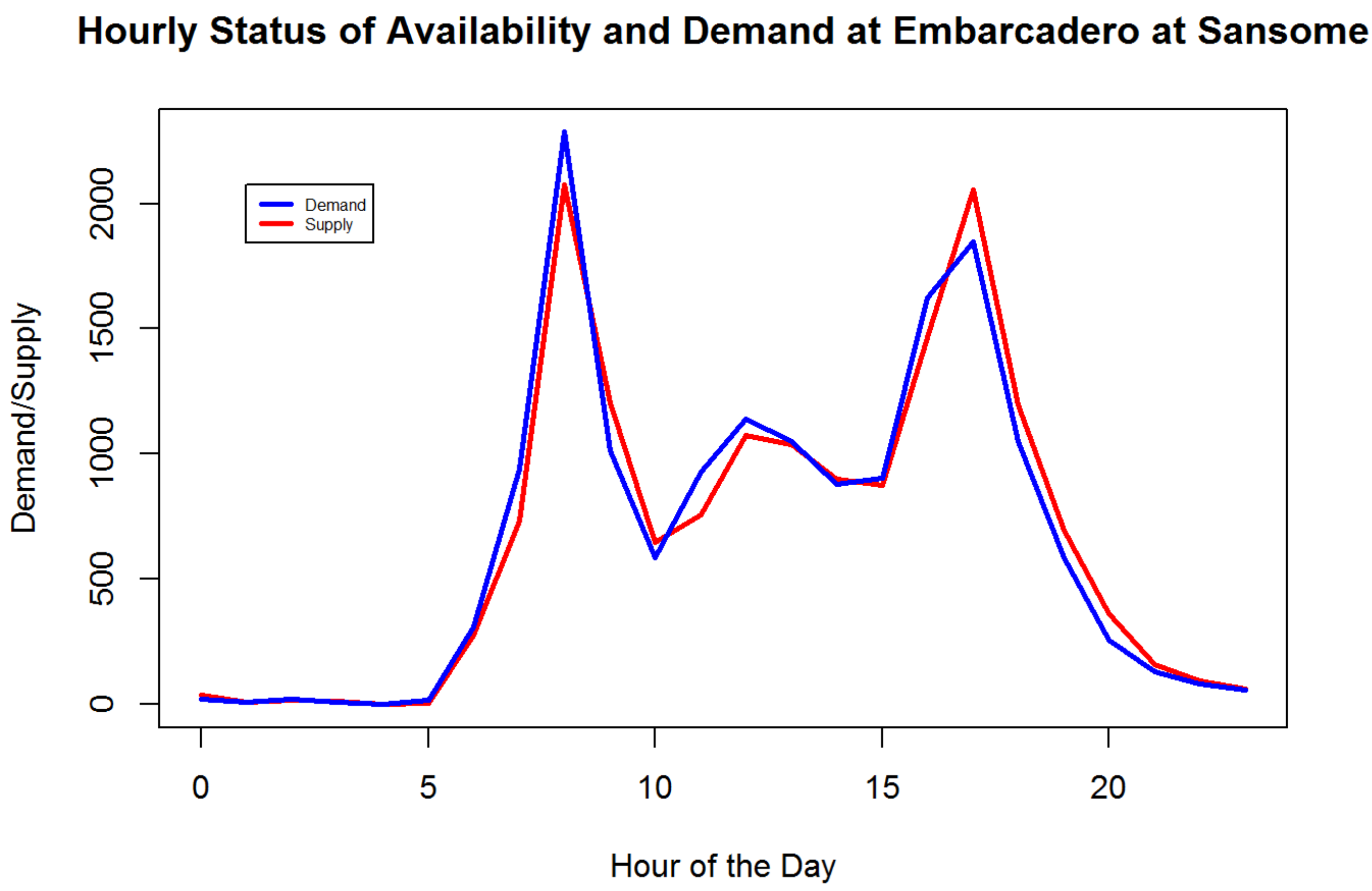

```
#Find the most Popular Routes
pop_routes <- count(df, Start_Station, End_Station)
pop_routes <- pop_routes[order(-pop_routes$n),]

head(pop_routes)

#Find top 12 Start and End Stations
pop_startStations <- as.character(hotStartZone$Start_Station[1:12])
pop_stopStations  <- as.character(hotEndZone$End_Station[1:12])

#Subset those observations that is part of the Top 12 Start and End Stations
subDat <- subset(pop_routes, pop_routes$Start_Station %in% pop_startStations )
subDat <- subset(subDat, subDat$End_Station %in% pop_stopStations )

#Generate Heat Map for the Popular Routes
hm3 <- ggplot(subDat, aes(x=End_Station, y=Start_Station, fill=n)) +
       geom_tile(color="black", size=0.05) +
       coord_equal() + scale_fill_gradient(low = "yellow", high = "red") +
       labs(x="End Station", y="Start Station", title="Heat Map for Popular Routes"
) +
       theme(plot.title=element_text(size = 20, face = "bold", hjust = 0.5)) +
       theme(axis.ticks=element_blank()) +
       theme(axis.text.x = element_text(angle = 90, hjust = 1, element_text(size=6
)))
       theme(legend.title=element_text(size=7)) +
       theme(legend.text=element_text(size=6))
hm3
```

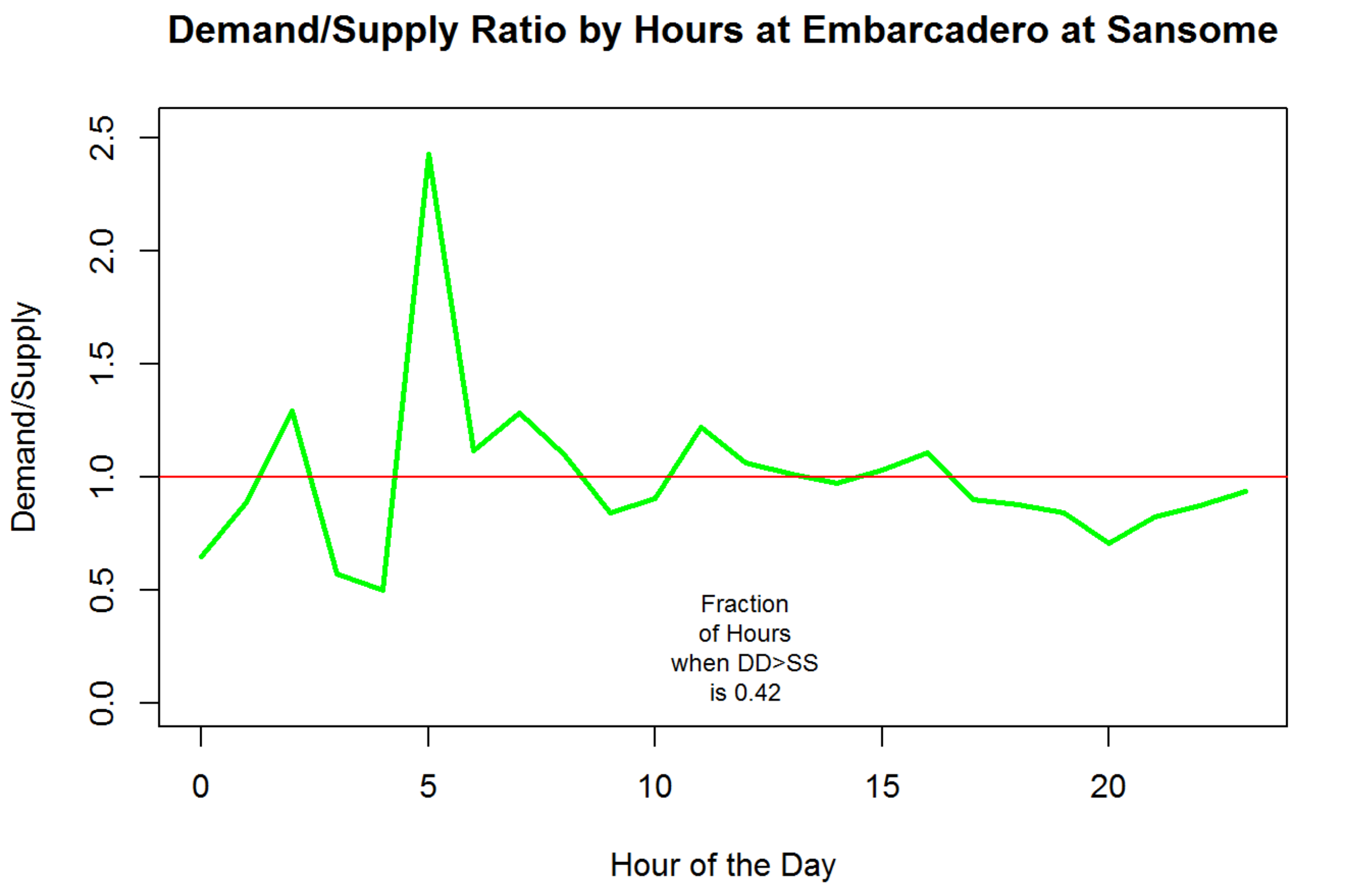



```
df$Start_Hour <- as.integer(df$Start_Hour)
df$End_Hour <- as.integer(df$End_Hour)

#Demand Supply at every Station

#Find Status Demand / Supply at Each Station

#Create the Table Mapping Station ID to its Name
StationNameTable <- sqldf( "SELECT DISTINCT Start_Station, Start_Terminal
                            FROM df
                            ORDER BY Start_Terminal")

StationStatus <- data.frame()

stationIDs <- sort(unique(df$End_Terminal))

for(i in 1:70){

  Station_data  <- subset(df, df$Start_Terminal == stationIDs[i] | df$End_Terminal
== stationIDs[i])
  NameofStation <- StationNameTable$Start_Station[i]
  StationCode   <- StationNameTable$Start_Terminal[i]

  #Find out the Demand at this Station at every hour
  DD <- sqldf("SELECT Start_Hour AS Hour, COUNT(Start_Hour) AS Demand
              FROM Station_data
              GROUP BY Start_Hour
              ORDER BY Start_Hour")
```

```
  #Find out the Supply at this Station at every hour
  SS <- sqldf("SELECT End_Hour AS Hour, COUNT(End_Hour) AS Supply
              FROM Station_data
              GROUP BY End_Hour
              ORDER BY End_Hour")

  #Merge the Demand and Supply Tables
  DDSS <- merge(SS, DD, by="Hour")

  #Check whether Demand is more than Supply

  condition   <- DDSS$Demand > DDSS$Supply
  status <- ifelse(condition, 1, 0)
  avgBalancedHours <- (sum(status))/24

  StatusofStation <- cbind.data.frame(StationCode, NameofStation, avgBalancedHours)

  StationStatus <- rbind.data.frame(StatusofStation, StationStatus)

  NameofStation = avgBalancedHours = StatusofStation = condition = status <- NULL
}

StationStatus <- StationStatus[order(-StationStatus$avgBalancedHours),]
names(StationStatus) <- c("Station ID", "Station_Name", "Rel Freq(DD>SS)")
StationStatus[,3] <- round(StationStatus[ ,3],2)

# These are the Stations where Demand is relatively high to the Supply over the Hou
rs
# with the peak time can be observered from the Plot (where the plot is above the l
ine y=1)
# It is done in the next Step

head(StationStatus,10)

#To check if Demand is met with Supply at the most Busiest Stations

# A for-loop structure can be used to iterate over all the desired Stations
# (which is not done here for simplicity) instead a function is written to check
# the Status of a Individual Station when invoked by its Station ID which is deploy
ed
# as a tool using rshiny for Part(c) of this Exercise

DemandSupply_StatusAtStation <- function(StationID){

  if(StationID %in%  unique(df$Start_Terminal)){

    Station_data <- subset(df, df$Start_Terminal == StationID | df$End_Terminal ==
StationID)
    NameofStation <- Station_data[1,4]

    #Find out the Demand at this Station at every hour
    Station_dataDD <- sqldf("SELECT Start_Hour, COUNT(Start_Hour) AS Demand
                            FROM Station_data
                            GROUP BY Start_Hour
                            ORDER BY Start_Hour")

    #Find out the Supply at this Station at every hour
```

```
    Station_dataSS <- sqldf("SELECT End_Hour, COUNT(End_Hour) AS Supply
                             FROM Station_data
                             GROUP BY End_Hour
                             ORDER BY End_Hour")

    #Merge the Demand and Supply Tables
    Station_dataDDSS <- sqldf("SELECT b.End_Hour, a.Demand, b.Supply
                               FROM Station_dataDD a
                               INNER JOIN Station_dataSS b
                               ON a.Start_Hour = b.End_Hour")

    #Plot the Demand and Supply by Hours
    plot (range(Station_dataDDSS$End_Hour), range(c(Station_dataDDSS$Supply,Statio
n_dataDDSS$Demand)),type='n',
          xlab='Hour of the Day',ylab='Demand/Supply',
          main = paste0("Hourly Status of Availability and Demand at ", NameofStat
ion))
          lines(Station_dataDDSS$End_Hour, Station_dataDDSS$Supply, col='red',lwd=
2.5, pch = 2)
          lines(Station_dataDDSS$End_Hour, Station_dataDDSS$Demand, col='blue',lwd
=2.5, pch = 10)
          legend(1,max(Station_dataDDSS$Supply), c('Demand', 'Supply'), lty=c(1,1)
, lwd=c(2.5,2.5),
          col=c('blue','red'), cex = 0.5)

    #Check whether Demand is more than Supply

    condition   <- Station_dataDDSS$Demand > Station_dataDDSS$Supply
    status <- ifelse(condition, 1, 0)
    avgBalancedHours <- (sum(status))/nrow(Station_dataDDSS)

    plot (range(Station_dataDDSS$End_Hour), range(c(0,max(Station_dataDDSS$Demand/
Station_dataDDSS$Supply)+0.1)),type='n',
          xlab='Hour of the Day',ylab='Demand/Supply',
          main = paste0("Demand/Supply Ratio by Hours at ", NameofStation))

    lines(Station_dataDDSS$End_Hour, Station_dataDDSS$Demand/Station_dataDDSS$Suppl
y, col='green',lwd=2.5, pch = 12)
    abline(a= 1, b=0, col = "red")
    text(12, 0.25, cex = 0.75,
         paste0("Fraction\nof Hours\nwhen DD>SS\nis ", round(avgBalancedHours,2)))
    }else{
    print("Please enter a Valid Station ID and call the Function")
  }
}

# Test the above function
# Pass any number to check the Status of a particular Station
# If ID is not a valid one it asks to re-enter
# Notice there are 2 plots generated

DemandSupply_StatusAtStation(50)
```

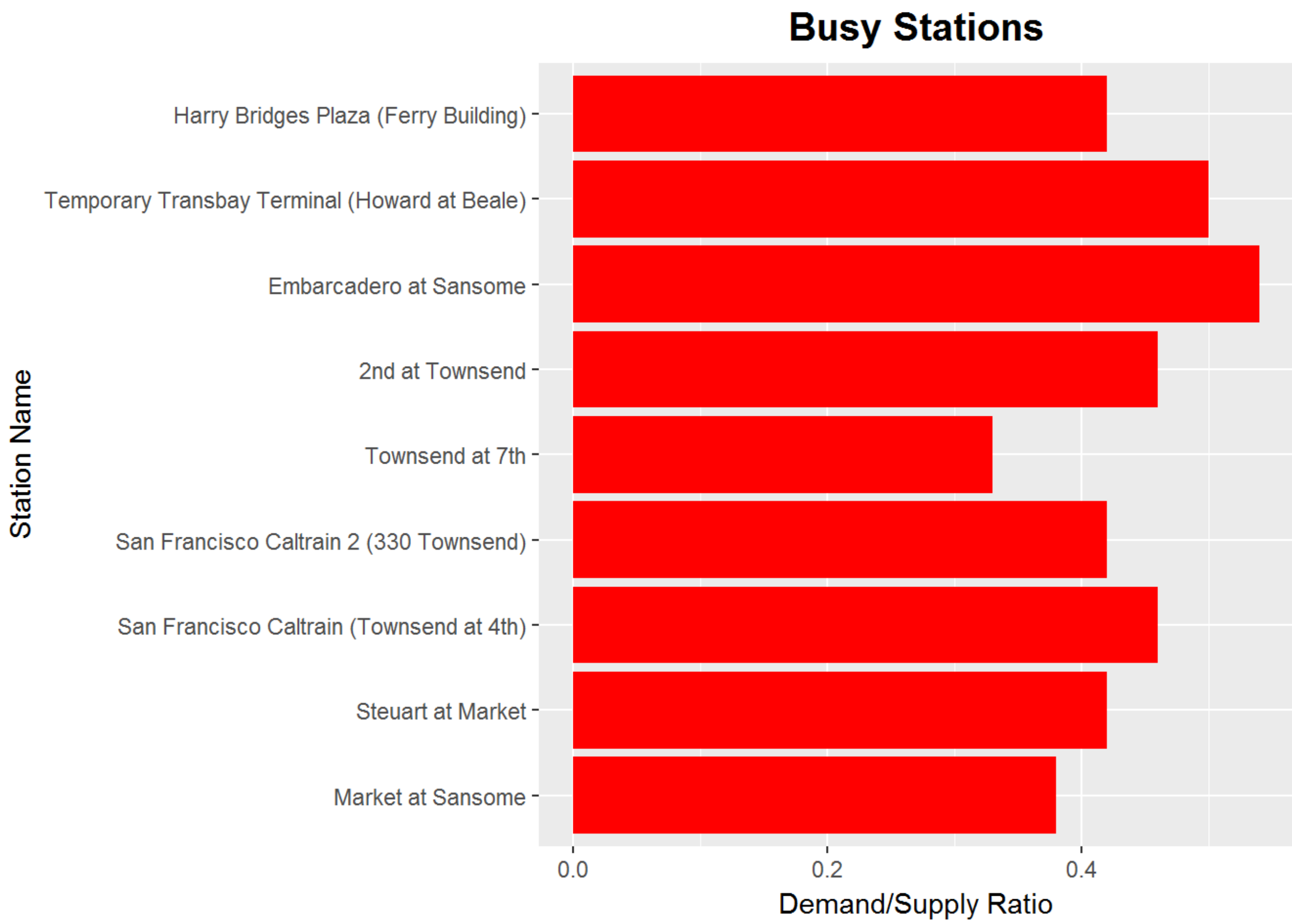




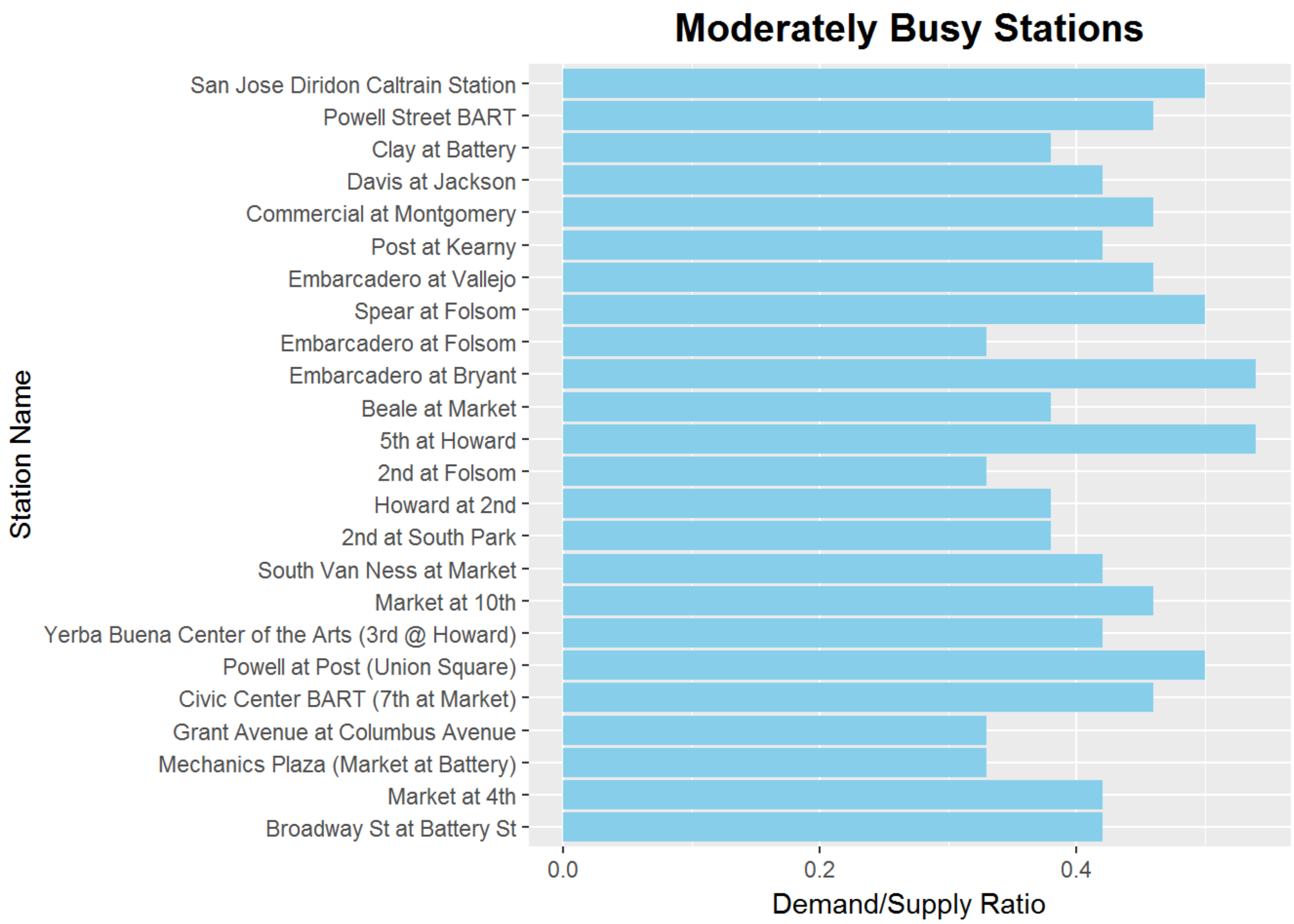

```
#Ananlysing whether the Shortage in Demand happens across the Busiest/Idle Stations
StartStopbyStation <- sqldf("SELECT a.Start_Station AS Station_Name, a.Total_Trips
AS Starts, b.Total_Trips AS Stops
```

```
AS Starts, b.Total_Trips AS Stops
                              FROM hotStartZone a
                              INNER JOIN hotEndZone b
                              ON a.Start_Station = b.End_Station")

#Finding Total Start and Stops for Each Station
StartStopbyStation$Total <- StartStopbyStation$Starts + StartStopbyStation$Stops

# Jenk-Fisher natural breaks optimization is used here to classify all the 70 Stations
# into 3 Groups namely Busy, Moderately Busy or Idle depending upon the above Table

v = StartStopbyStation$Total

#Break the Stations into 3 Clusters based on Jenks-Fisher Algorithm

j_int <- classIntervals(v,n=3,style = "jenks")

print(classIntervals(v,n=3,style = "fisher"))
print(classIntervals(v,n=3,style = "jenks"))

jenks.tests(j_int)
#both has classified into the exact same groups with high Goodness of Fit

#Using the Information from the Jenks Classifier I divide the Stations into 3 Groups

StartStopbyStation$Group[StartStopbyStation$Total >10362 ] <- "Busy"
StartStopbyStation$Group[StartStopbyStation$Total >3683 & StartStopbyStation$Total <= 10362 ] <- "Moderate"
StartStopbyStation$Group[StartStopbyStation$Total <= 3683 ] <- "Idle"

#Merge with Demand/Supply Status Table

StationByGroupTab <- merge(StationStatus, StartStopbyStation , by="Station_Name")
StationByGroupTab <- StationByGroupTab[ ,-c(4:6)]

StationByGroupTab <- StationByGroupTab[ , c("Station ID", "Station_Name", "Rel Freq(DD>SS)", "Group" )]

#Creating Table for Busy Stations
BusyStations <- StationByGroupTab[StationByGroupTab$Group == "Busy", ]
median(BusyStations$`Rel Freq(DD>SS)`)

#Creating Table for Moderate Stations
ModerateStations <- StationByGroupTab[StationByGroupTab$Group == "Moderate", ]
median(ModerateStations$`Rel Freq(DD>SS)`)

#Creating Table for Idle Stations
IdleStations <- StationByGroupTab[StationByGroupTab$Group == "Idle", ]
median(IdleStations$`Rel Freq(DD>SS)`)

#Plotting Busy Stations
plt_busy <- ggplot(BusyStations, aes(x=BusyStations$Station_Name, y = BusyStations$`Rel Freq(DD>SS)`)) +
            geom_bar(stat="identity", fill = "red") +
            xlab("Station Name") +
            ylab("Demand/Supply Ratio")+
```

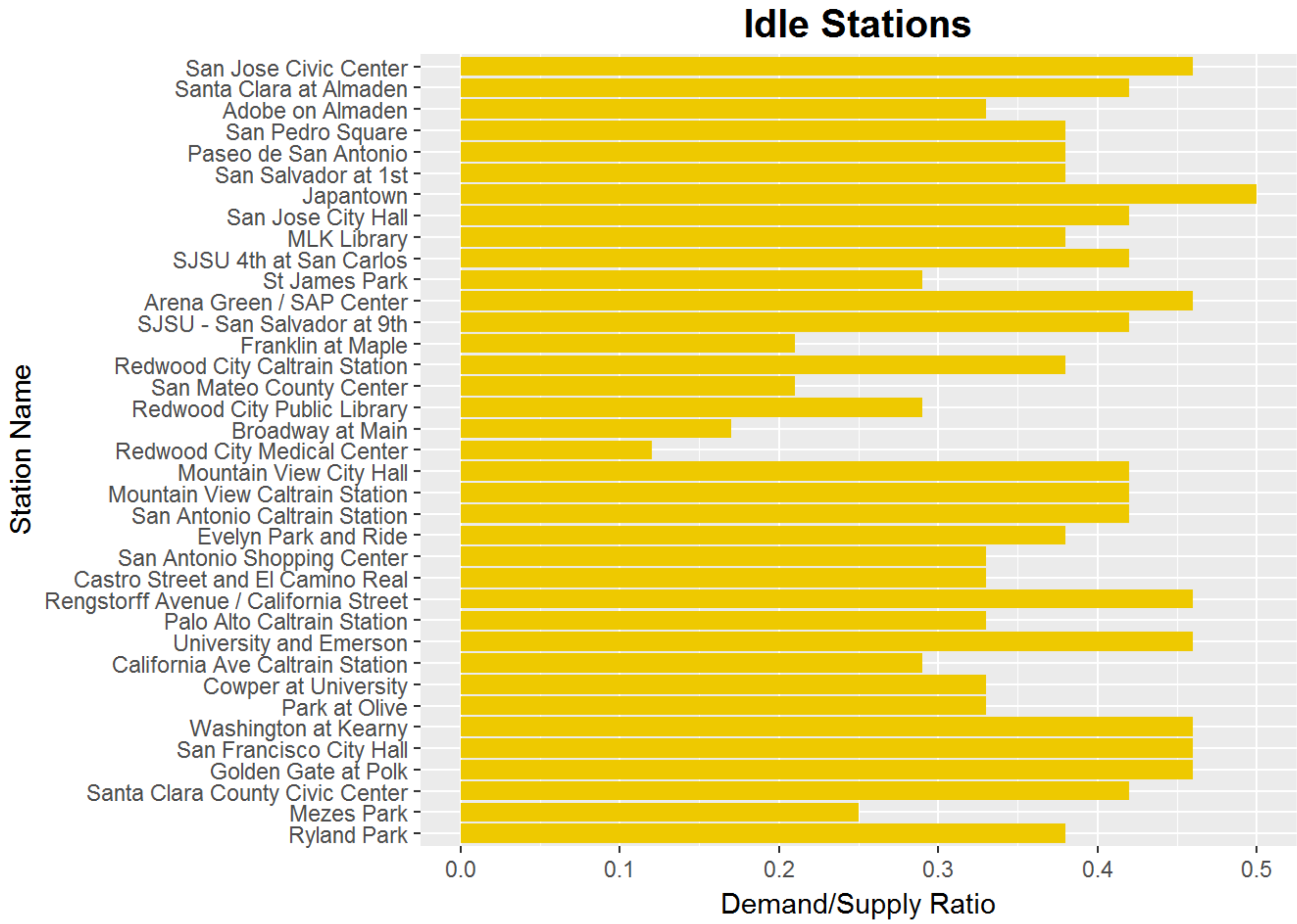
```
            ylab("Demand/Supply Ratio")+
            labs(title="Busy Stations")+
            theme(plot.title=element_text(size = 15, face = "bold", hjust = 0.5))
+
            coord_flip()

plt_busy
```



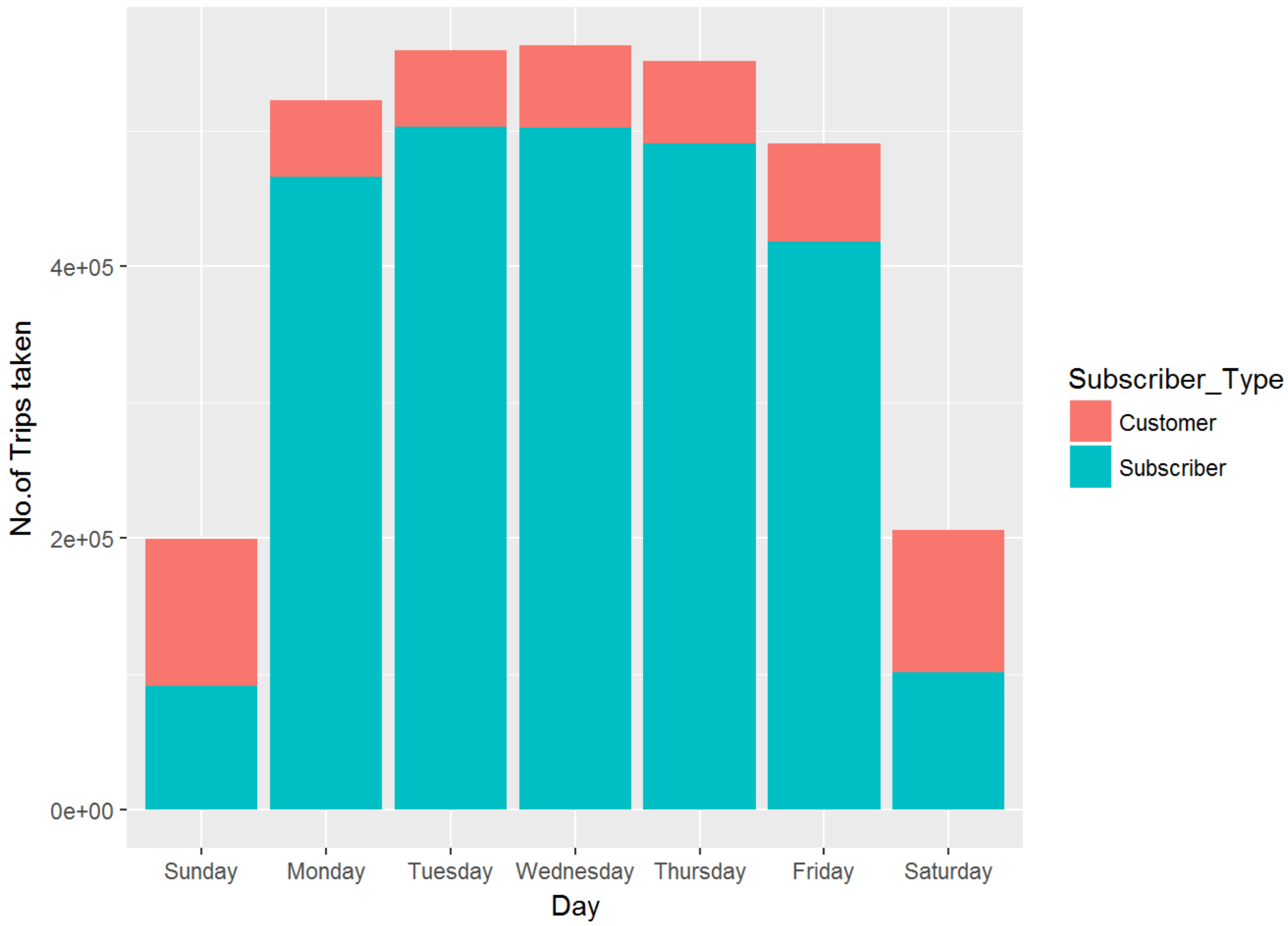

```
#Plotting Moderate Stations
plt_moderate <- ggplot(ModerateStations, aes(x=ModerateStations$Station_Name, y =
ModerateStations$`Rel Freq(DD>SS)`)) +
                geom_bar(stat="identity", fill = "skyblue") +
                xlab("Station Name") +
                ylab("Demand/Supply Ratio")+
                labs(title="Moderately Busy Stations")+
                theme(plot.title=element_text(size = 15, face = "bold", hjust = 0.
5)) +
                coord_flip()

plt_moderate
```



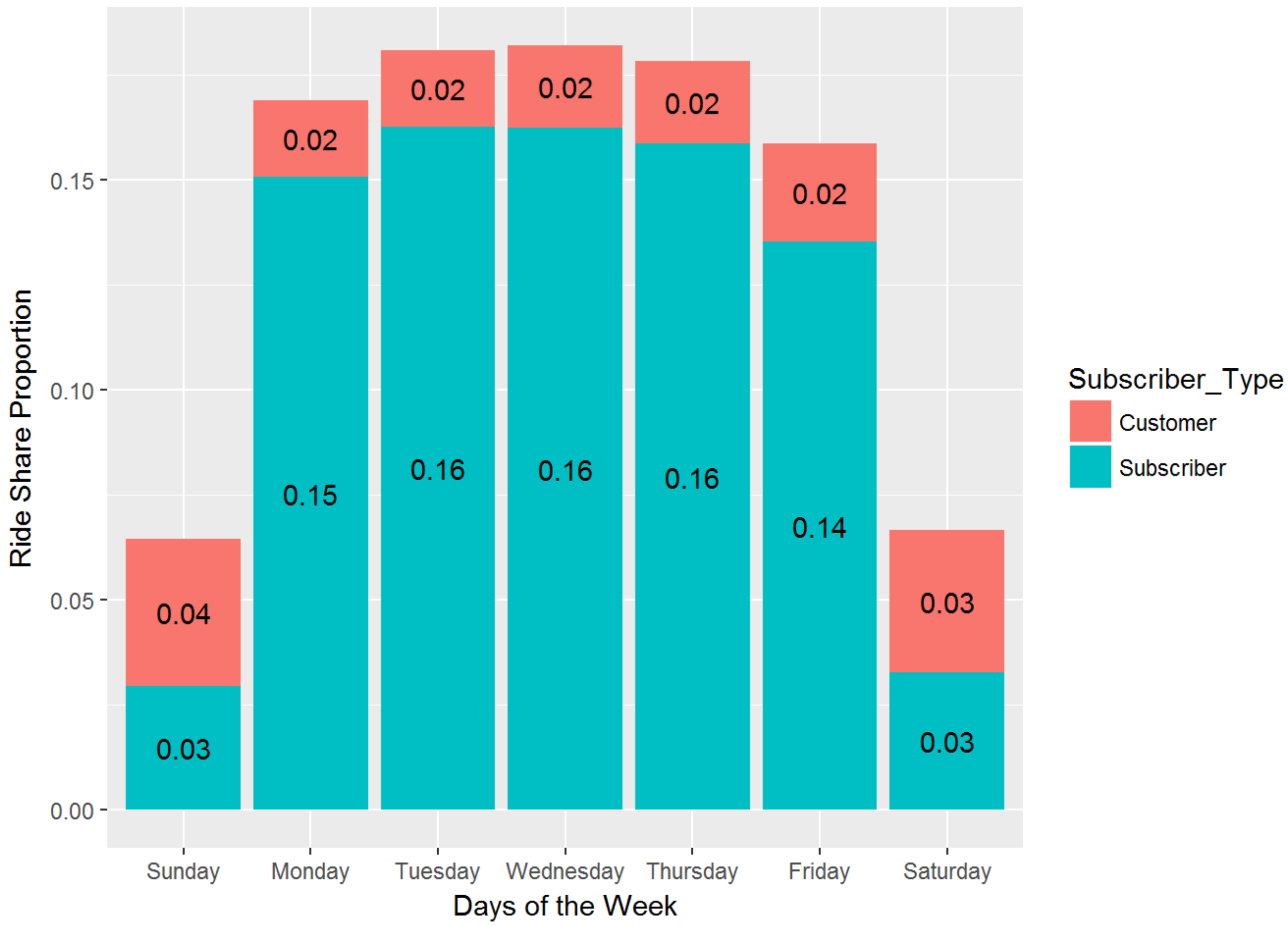

```
#Plotting Idle Stations
plt_idle <- ggplot(IdleStations, aes(x=IdleStations$Station_Name, y = IdleStations
$`Rel Freq(DD>SS)`)) +
            geom_bar(stat="identity", fill = "gold2") +
            xlab("Station Name") +
            ylab("Demand/Supply Ratio") +
            labs(title="Idle Stations")+
            theme(plot.title=element_text(size = 15, face = "bold", hjust = 0.5))
+
            coord_flip()

plt_idle
```



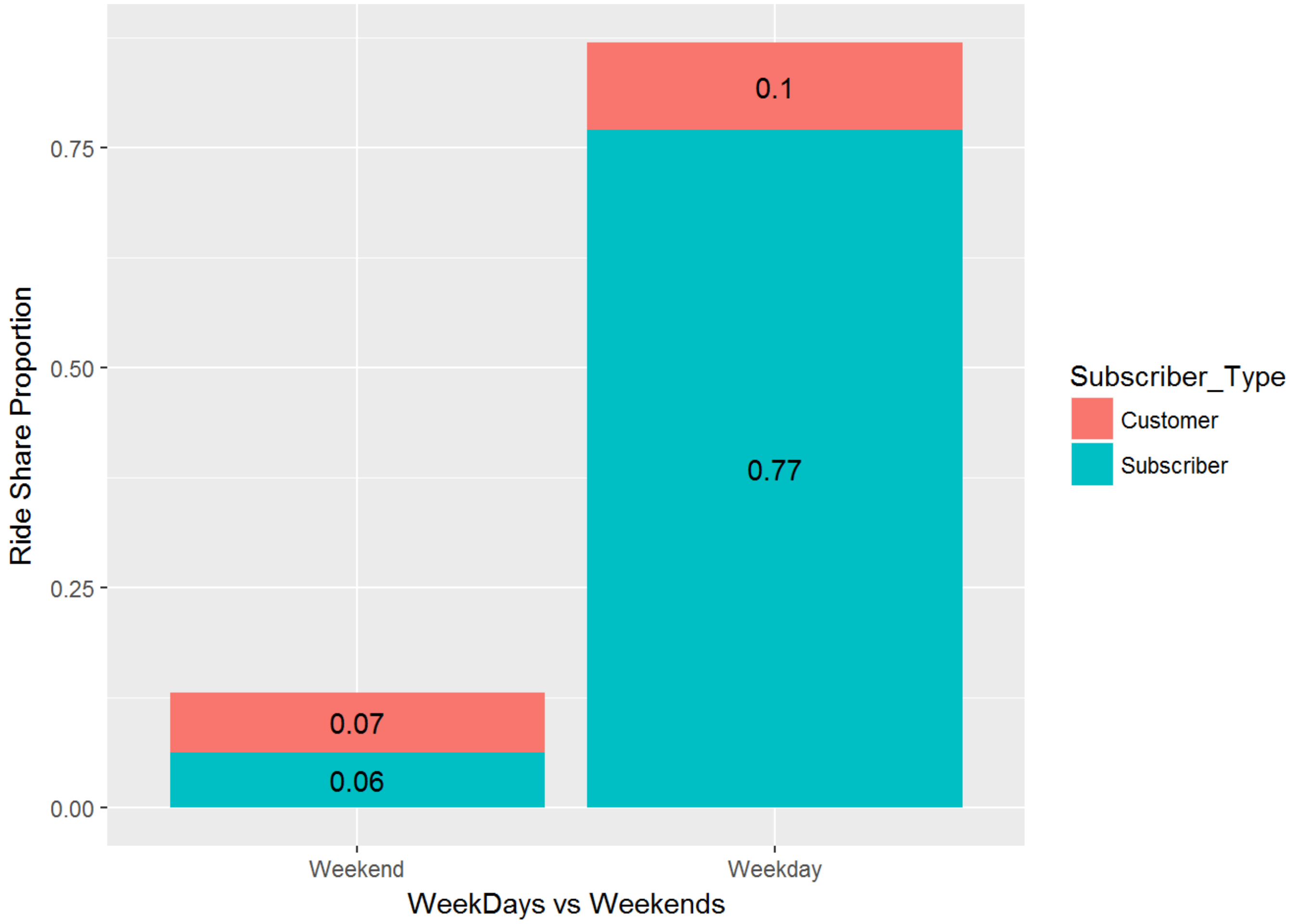

```
# Summary

# Important Plots

# Number of Trips Taken over the Days of the Weeks by Subscriber_Type
# Inference : The Demand is high in the WeekDays than it is on the Weekends

plt1
```



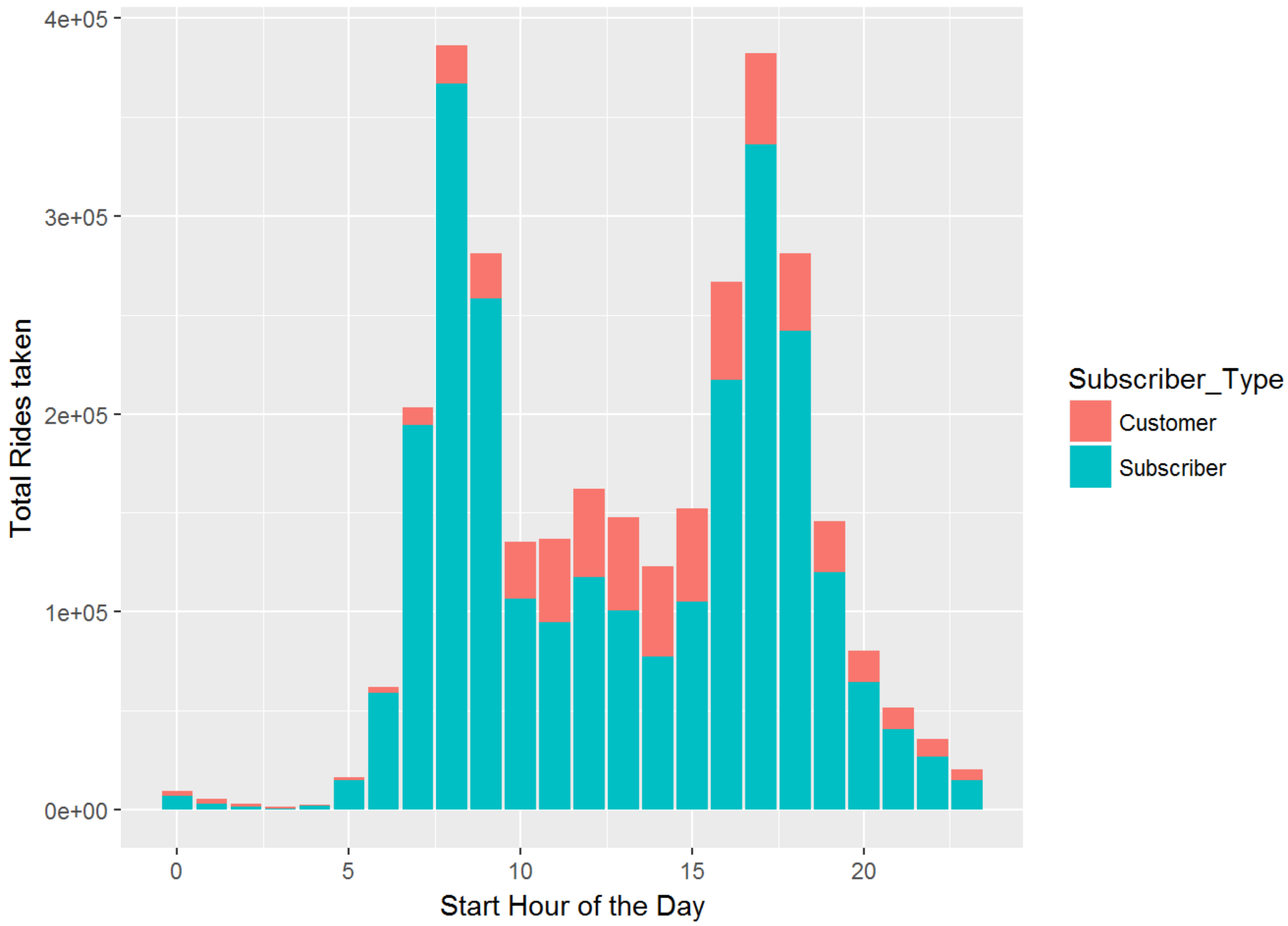

```
# Stacked Bar Chart of the above plot
# Inference : Customers have more Demand in the Weekends as they take over half of the Rides

plt2
```



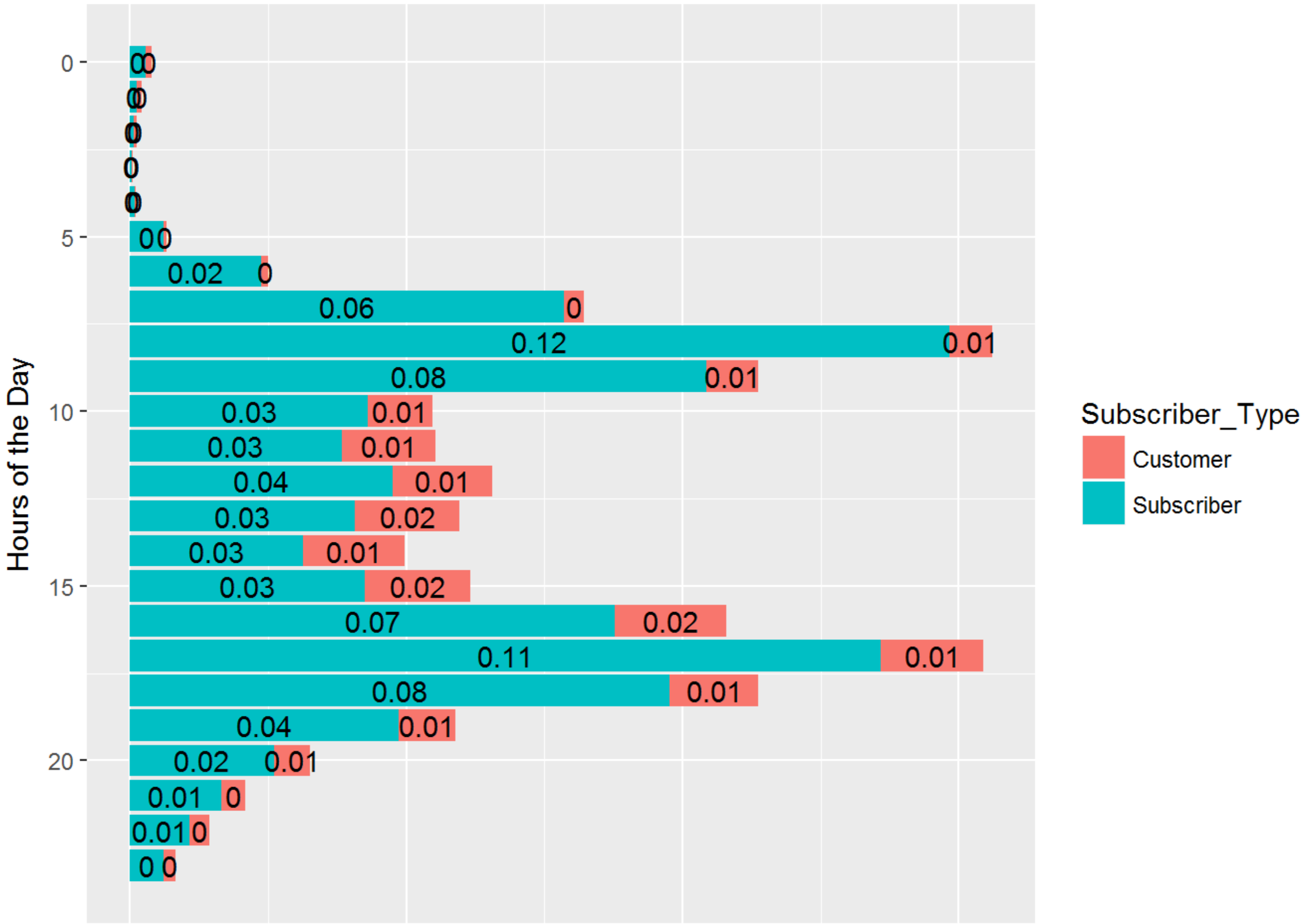

```
# Stacked Bar Chart of Weekday vs Weekends (Grouped)
# Inference : Same as above

plt3
```



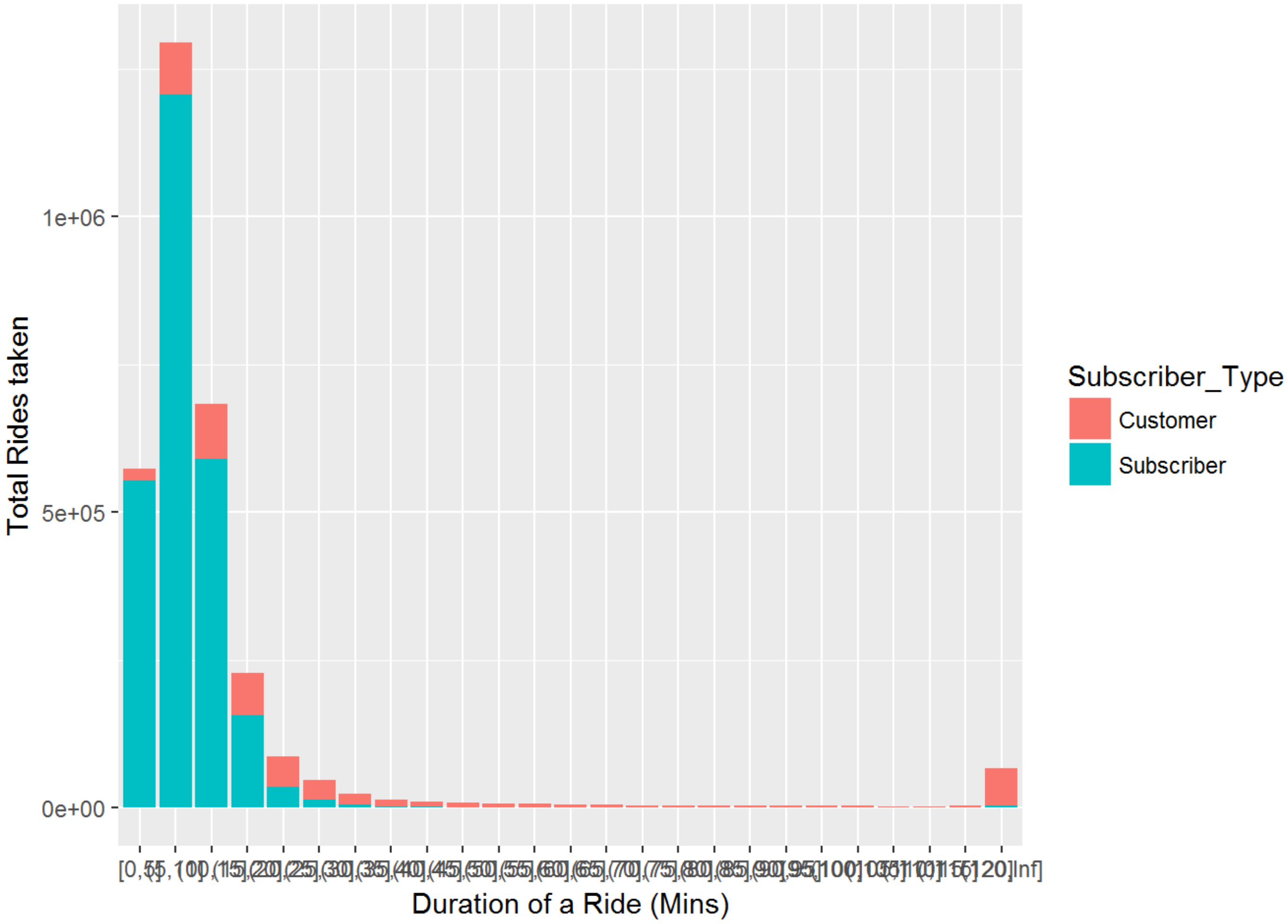

```
# Barplot of Rides taken across the Hours of a Day
# Inference : Most Rides are taken between 8-9 a.m. and the lowest in 3-4 a.m.,
# so probably take Bike Rides to avoid traffic that could be there due to office/sc
hool hours
# N.B. This doesnt show us whether its Weekday or Weekend, to infer about that I ha
ve plotted a
# Heat Map to understand the Demand patterns

plt4
```



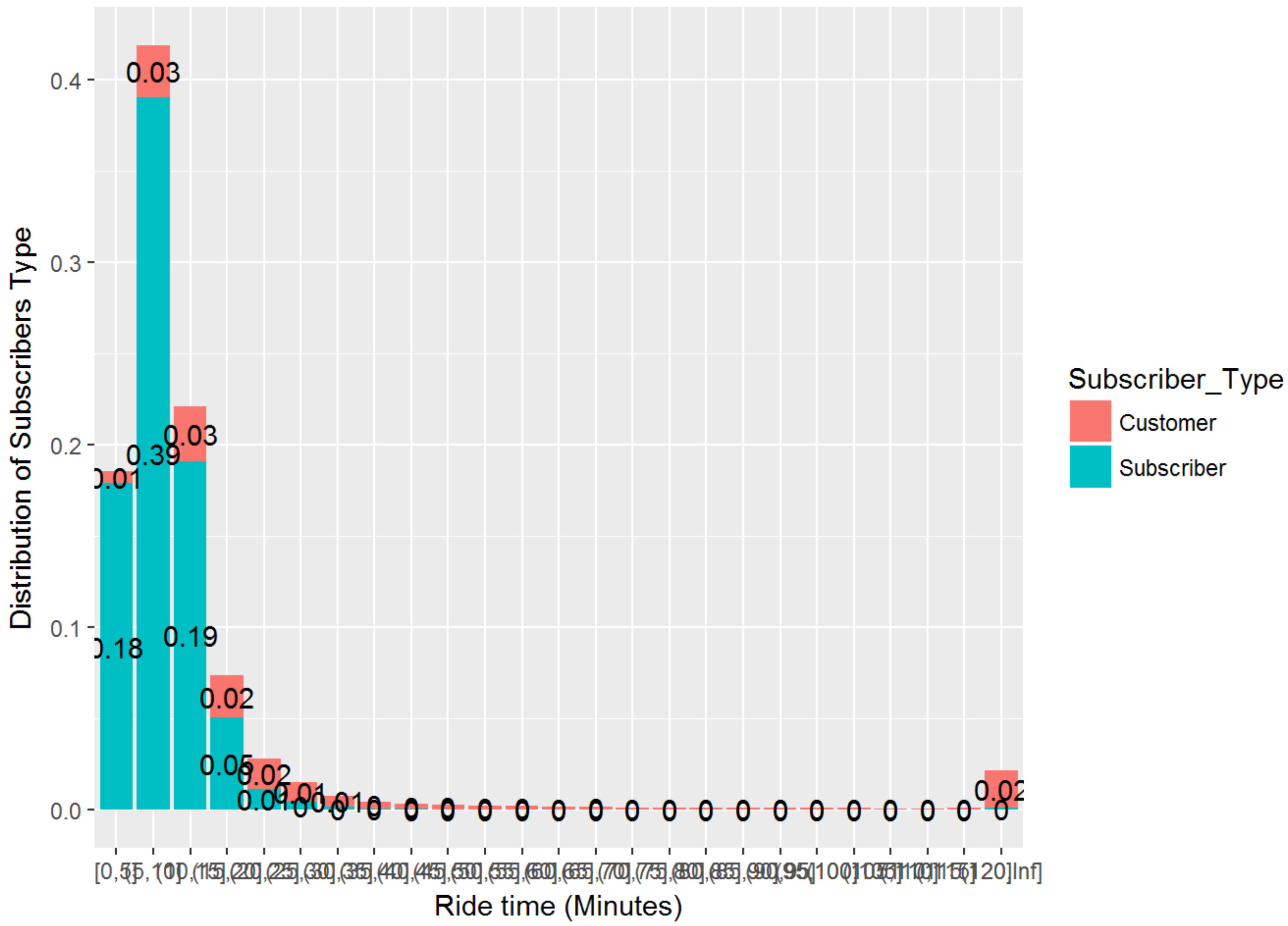

```
# Stacked Bar Chart of the above
# Inference : Subscribers are more active during the peak hours and vice-versa

plt5
```



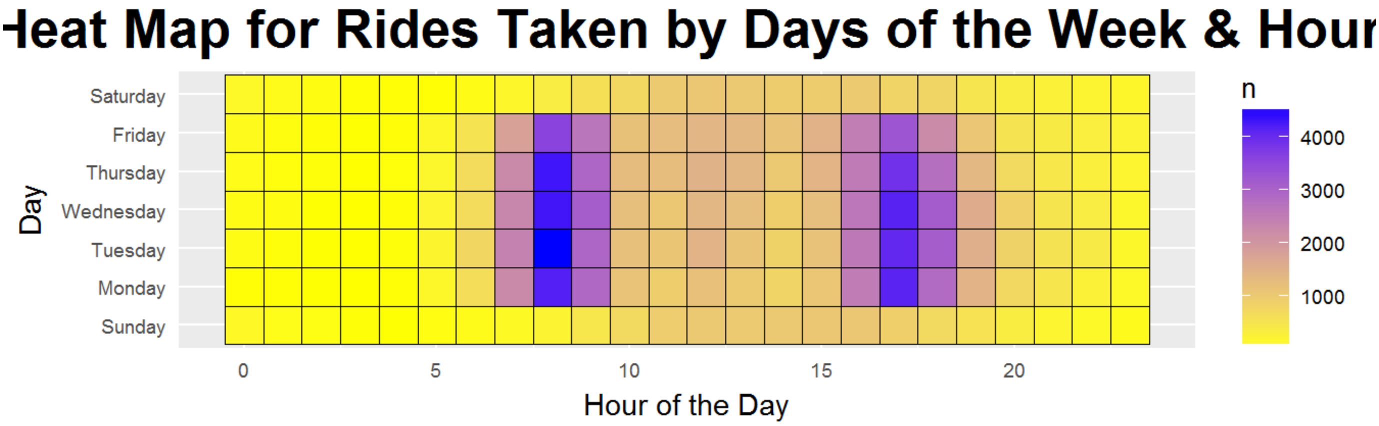

```
# Bar plot of Number of Rides taken when divided into groups of 5 mins each
# Inference : Most of the rides generally last for short duration less than 15 mins

plt6
```



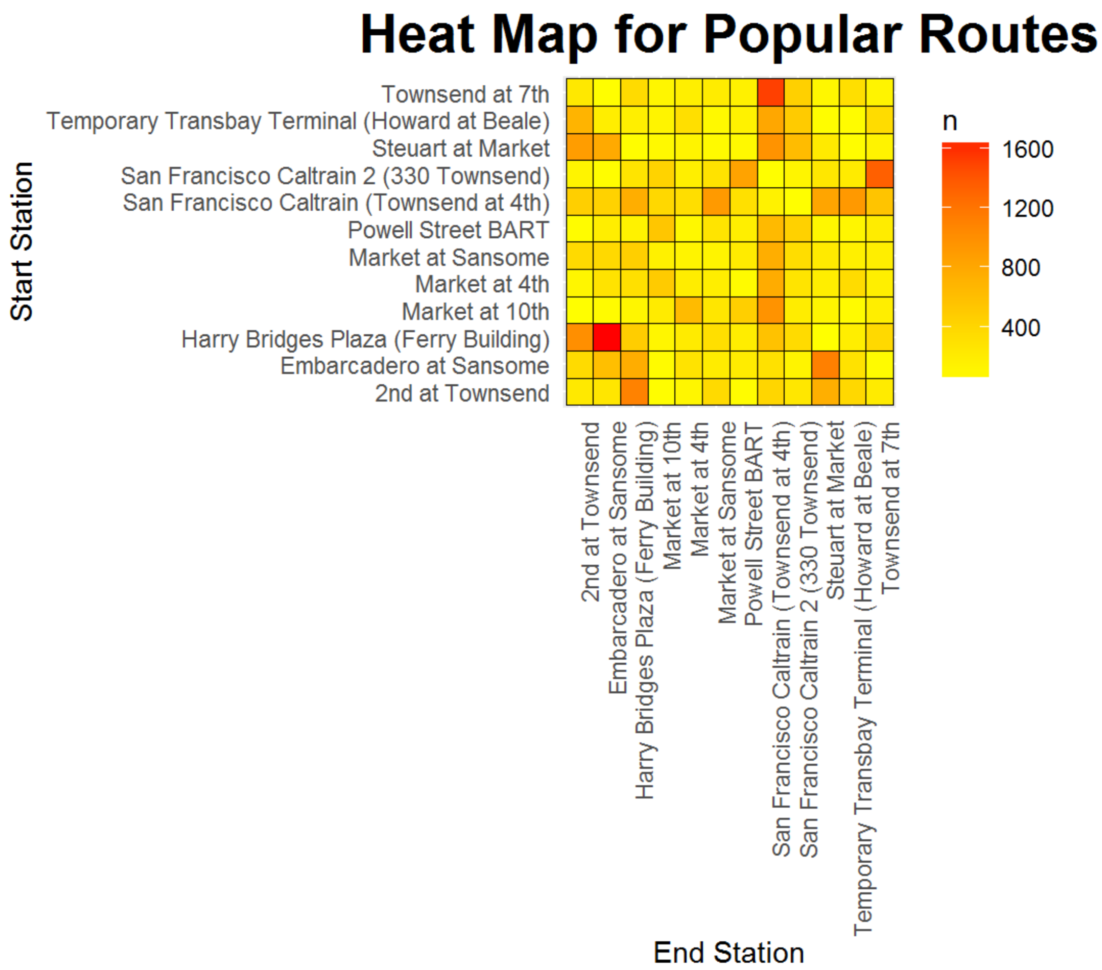

```
# Stacked Bar Chart to show the average duration of a trip by Subscriber Type
# Inference : Subscribers generally use the service for Short Trips less than 5-10m
ins

plt7
```



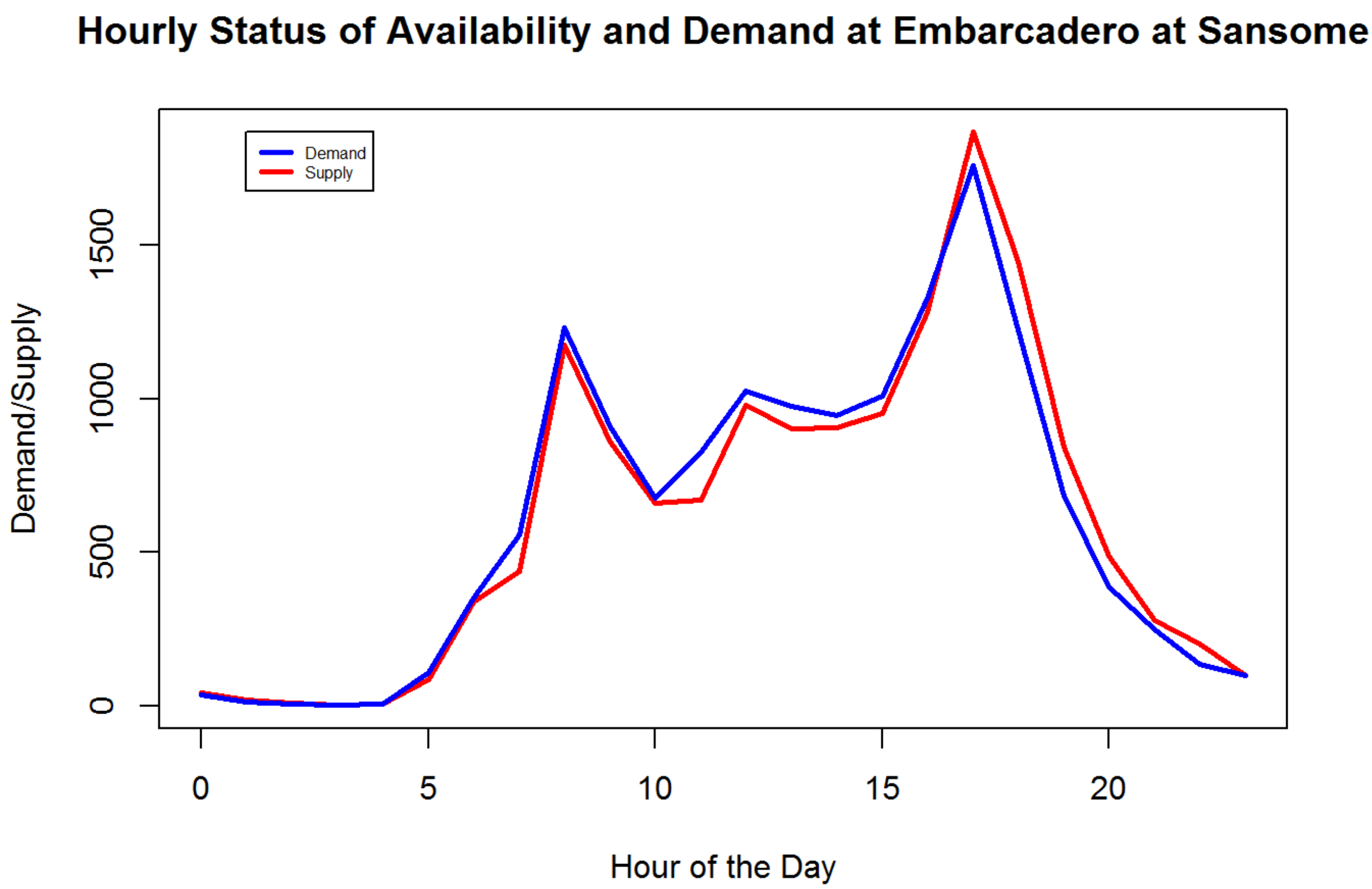

```
# Heat Map to see when across the Demand the # of Trips taken is the most

# Inference : It reveals that the maximum rides happend over the Weekdays and at 8-9 a.m./
# 5-6 p.m.

hm1
```

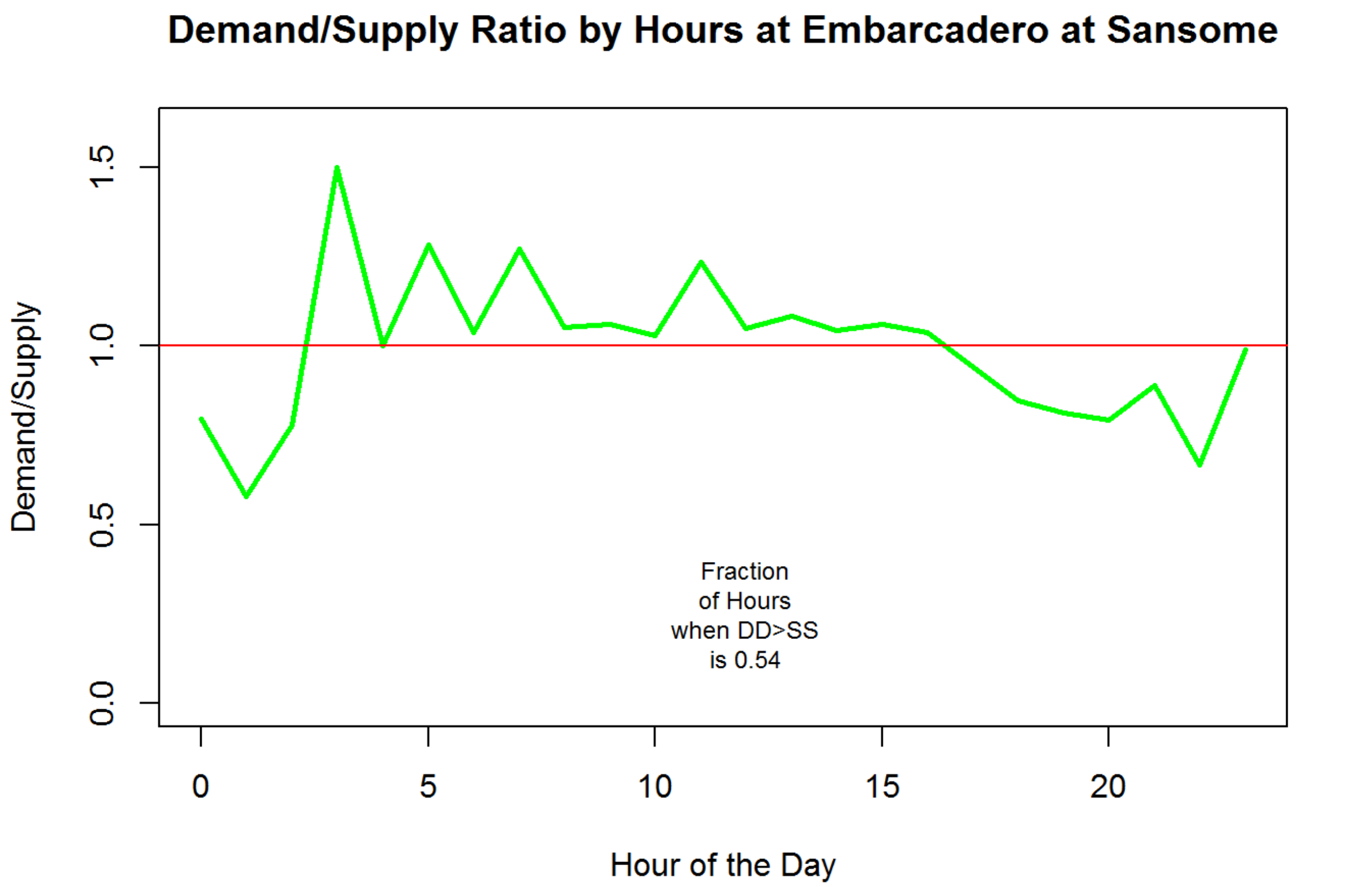



```
# A Heat Map to show which routes are the most Popular by Start and End terminal

# Inference : So a more reddish box will indicate maximum trips have been taken whe
n
# the Ride Starts a particular Station and ends at the particular Station)

hm3
```

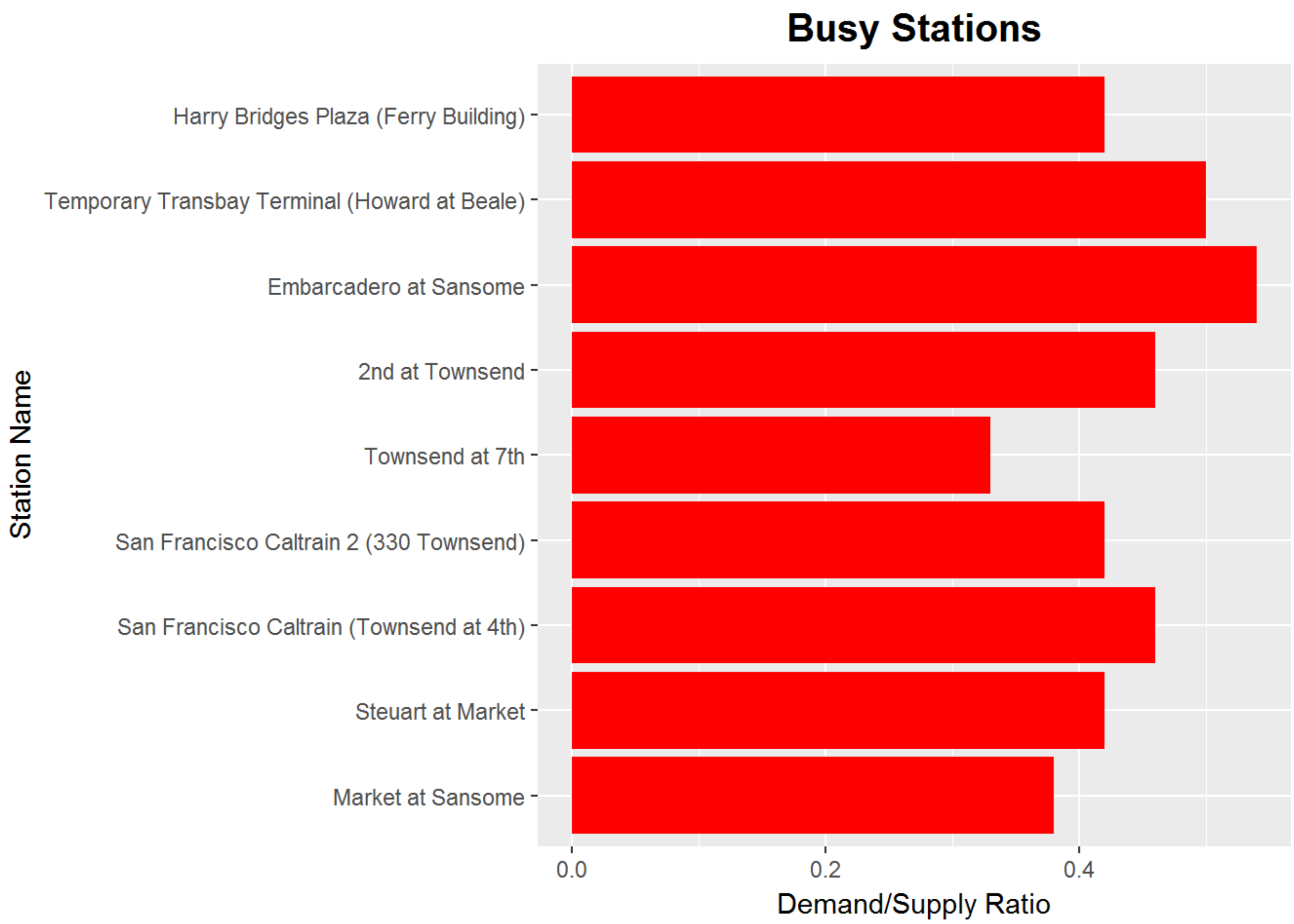



```
# To show the Stations where the Demand is relatively higher than the Supply,
# of course Bikes are already stationed there (since the Trip is recorded) hence th
ere is
# no issue but these Stations needs to be monitored on an hourly basis

head(StationStatus,5)

# If we want to see the Hourly status of a particular Station
# N.B. 2 Plots are generated
DemandSupply_StatusAtStation(60) # pass the Station code as an argument in the func
tion
```

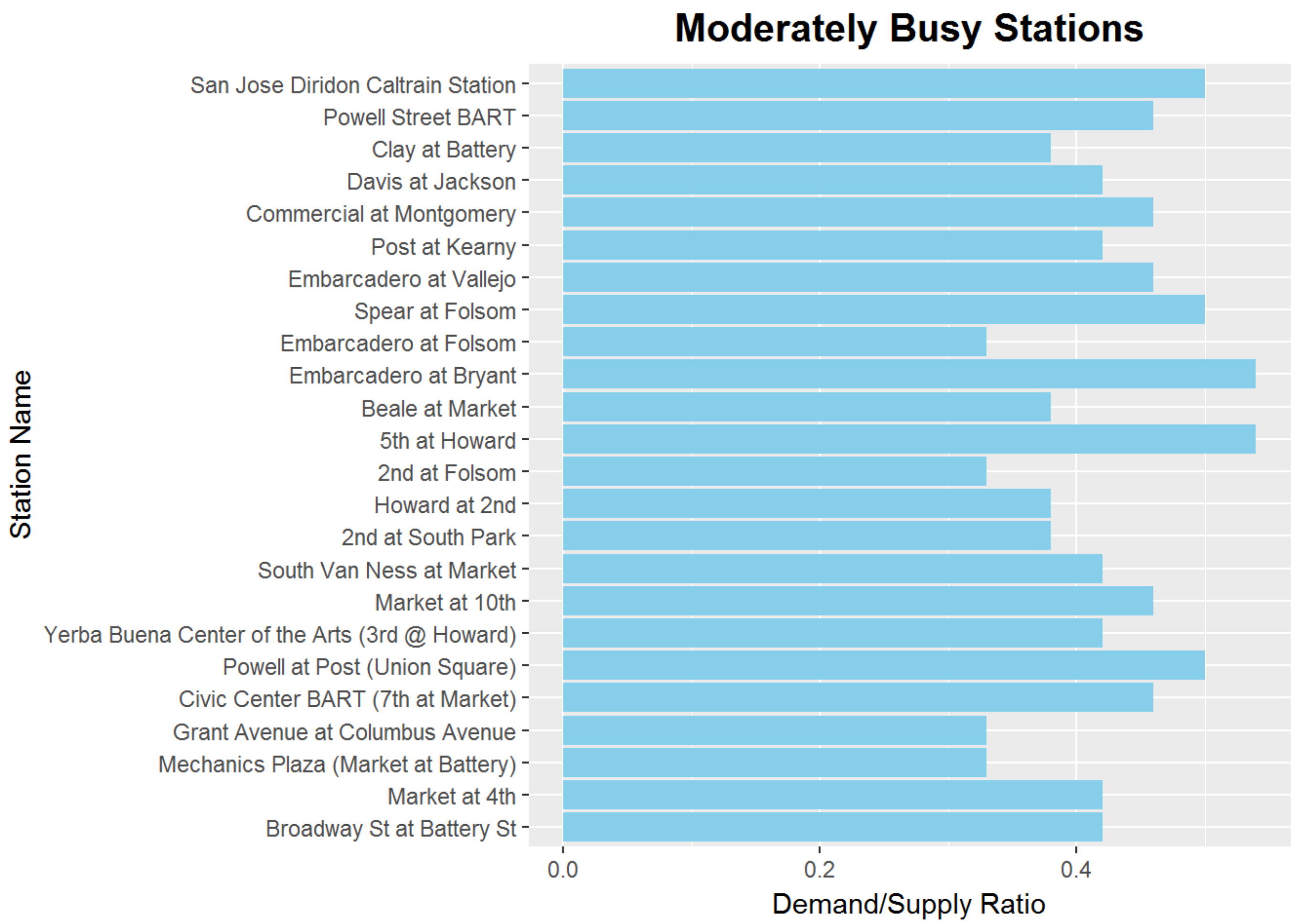




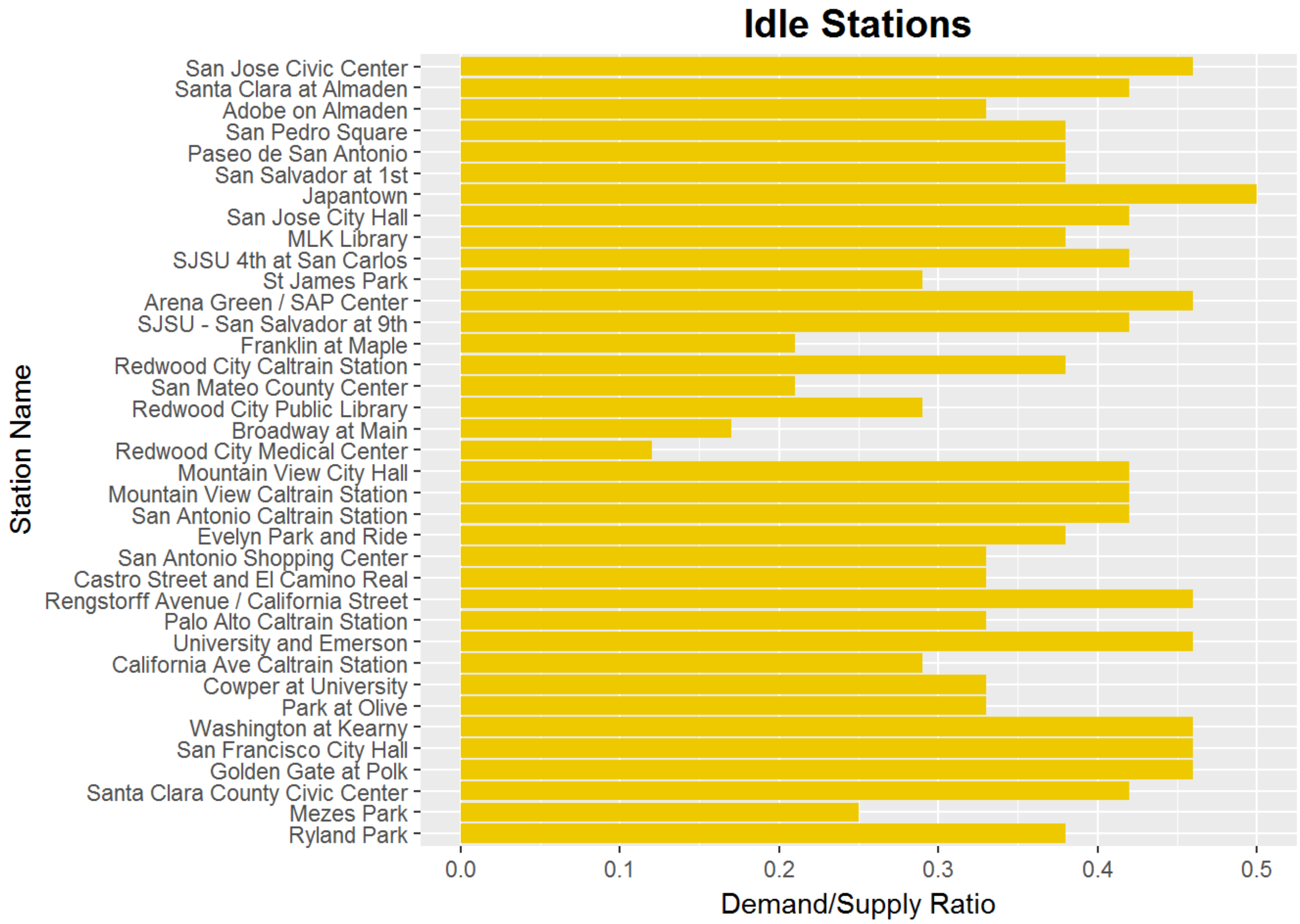

```
#Plotting Demand/Supply Ratio for Busy Stations
plt_busy
```




```
#Plotting Demand/Supply Ratio for Moderately busy Stations
plt_moderate
```




```
#Plotting Demand/Supply Ratio for Idle Stations
plt_idle
```




```
# The Strategy would be to identify the Busy Stations which are having high Demand:
Supply
# ratio and to deploy more Bikes here from the Idle/Moderately Busy Stations as it
would
# increase the no. of rides.
```