CSP 571 Project

Cab Ride Data Analytics

A20489131 - Pranjal Naik

A20488668 - Kishan Pate

A20468078 - Jainam Shah

Loading Libraries

```
#install.packages("caret", repos = "http://cran.us.r-project.org")
# install.packages("sqldf")
# install.packages("tidyr")
# install.packages("tidyverse")
# install.packages("ggplot2")
# install.packages("readr")
#install.packages("gmodels")
# install.packages("tm")
# install.packages("SnowballC")
# install.packages("wordcloud")
# install.packages("RColorBrewer")
# install.packages("treemap")
# install.packages("highcharter")
# install.packages("remotes")
# remotes::install_github("cran/DMwR")
#install.packages("corrplot")
#install.packages('rpart.plot')
library(rpart.plot)
```

```
## Loading required package: rpart
```

```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library("DMwR")
```

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

```
## Loading required package: grid
## Registered S3 method overwritten by 'quantmod':
##   method            from
##   as.zoo.data.frame zoo
```

```
library(treemap)
library(highcharter)
library(tidyr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(tidyverse)
```

```
## ── Attaching packages
## ───────────────────────────────────────
## tidyverse 1.3.2 ──
```

```
## ✔ ggplot2 3.4.0     ✔ purrr   0.3.5
## ✔ tibble  3.1.8     ✔ stringr 1.4.1
## ✔ readr   2.1.3     ✔ forcats 0.5.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
```

```
options(gsubfn.engine="R")
library(sqldf)
```

```
## Loading required package: gsubfn
## Loading required package: proto
## Loading required package: RSQLite
```

```
library(ggplot2)
library(readr)
library(gmodels)
library(tm)
```

```
## Loading required package: NLP
##
## Attaching package: 'NLP'
##
## The following object is masked from 'package:ggplot2':
##
##     annotate
```

```
library(SnowballC)
library(wordcloud)
```

```
## Loading required package: RColorBrewer
```

```
library(RColorBrewer)
library(rpart)
library(randomForest)
```

```
## randomForest 4.7-1.1
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
##
## The following object is masked from 'package:ggplot2':
##
##     margin
##
## The following object is masked from 'package:dplyr':
##
##     combine
```

Loading Dataset

```
cab_ds <- read.csv("/Users/Shared/pranjalnaik/Pranjal DPA/rideshare_kaggle.csv")
#head(cab_ds)
summary(cab_ds)
```

```
##       id              timestamp              hour            day       
##  Length:693071      Min.   :1.543e+09   Min.   : 0.00   Min.   : 1.00  
##  Class :character   1st Qu.:1.543e+09   1st Qu.: 6.00   1st Qu.:13.00  
##  Mode  :character   Median :1.544e+09   Median :12.00   Median :17.00  
##                     Mean   :1.544e+09   Mean   :11.62   Mean   :17.79  
##                     3rd Qu.:1.545e+09   3rd Qu.:18.00   3rd Qu.:28.00  
##                     Max.   :1.545e+09   Max.   :23.00   Max.   :30.00  
##                                                                        
##      month         datetime           timezone            source         
##  Min.   :11.00   Length:693071      Length:693071      Length:693071     
##  1st Qu.:11.00   Class :character   Class :character   Class :character  
##  Median :12.00   Mode  :character   Mode  :character   Mode  :character  
##  Mean   :11.59                                                           
##  3rd Qu.:12.00                                                           
##  Max.   :12.00                                                           
##                                                                          
##  destination          cab_type          product_id            name          
##  Length:693071      Length:693071      Length:693071      Length:693071     
##  Class :character   Class :character   Class :character   Class :character  
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character  
##                                                                              
##                                                                              
##                                                                              
##                                                                              
##      price          distance     surge_multiplier    latitude    
##  Min.   : 2.50   Min.   :0.020   Min.   :1.000    Min.   :42.21  
##  1st Qu.: 9.00   1st Qu.:1.280   1st Qu.:1.000    1st Qu.:42.35  
##  Median :13.50   Median :2.160   Median :1.000    Median :42.35  
##  Mean   :16.55   Mean   :2.189   Mean   :1.014    Mean   :42.34  
##  3rd Qu.:22.50   3rd Qu.:2.920   3rd Qu.:1.000    3rd Qu.:42.36  
##  Max.   :97.50   Max.   :7.860   Max.   :3.000    Max.   :42.37  
##  NA's   :55095                                                    
##    longitude       temperature    apparentTemperature short_summary     
##  Min.   :-71.11   Min.   :18.91   Min.   :12.13       Length:693071     
##  1st Qu.:-71.08   1st Qu.:36.45   1st Qu.:31.91       Class :character  
##  Median :-71.06   Median :40.49   Median :35.90       Mode  :character  
##  Mean   :-71.07   Mean   :39.58   Mean   :35.88                         
##  3rd Qu.:-71.05   3rd Qu.:43.58   3rd Qu.:40.08                         
##  Max.   :-71.03   Max.   :57.22   Max.   :57.22                         
##                                                                         
##  long_summary       precipIntensity    precipProbability    humidity     
##  Length:693071      Min.   :0.000000   Min.   :0.0000    Min.   :0.3800  
##  Class :character   1st Qu.:0.000000   1st Qu.:0.0000    1st Qu.:0.6400  
##  Mode  :character   Median :0.000000   Median :0.0000    Median :0.7100  
##                     Mean   :0.008922   Mean   :0.1461    Mean   :0.7411  
##                     3rd Qu.:0.000000   3rd Qu.:0.0000    3rd Qu.:0.8800  
##                     Max.   :0.144700   Max.   :1.0000    Max.   :0.9600  
##                                                                          
##    windSpeed         windGust      windGustTime        visibility    
##  Min.   : 0.450   Min.   : 0.80   Min.   :1.543e+09   Min.   : 0.717  
##  1st Qu.: 3.410   1st Qu.: 4.06   1st Qu.:1.543e+09   1st Qu.: 8.432  
##  Median : 5.910   Median : 7.55   Median :1.544e+09   Median : 9.880  
##  Mean   : 6.186   Mean   : 8.47   Mean   :1.544e+09   Mean   : 8.468
##  3rd Qu.: 8.410   3rd Qu.:11.74   3rd Qu.:1.545e+09   3rd Qu.: 9.996
##  Max.   :15.000   Max.   :27.25   Max.   :1.545e+09   Max.   :10.000
##
##  temperatureHigh temperatureHighTime temperatureLow  temperatureLowTime
##  Min.   :32.68   Min.   :1.543e+09   Min.   :17.85   Min.   :1.543e+09
##  1st Qu.:42.57   1st Qu.:1.543e+09   1st Qu.:30.17   1st Qu.:1.543e+09
##  Median :44.68   Median :1.544e+09   Median :34.18   Median :1.544e+09
##  Mean   :45.04   Mean   :1.544e+09   Mean   :34.15   Mean   :1.544e+09
##  3rd Qu.:46.91   3rd Qu.:1.545e+09   3rd Qu.:38.73   3rd Qu.:1.545e+09
##  Max.   :57.87   Max.   :1.545e+09   Max.   :46.60   Max.   :1.545e+09
##
##  apparentTemperatureHigh apparentTemperatureHighTime apparentTemperatureLow
##  Min.   :22.62           Min.   :1.543e+09           Min.   :11.81
##  1st Qu.:36.57           1st Qu.:1.543e+09           1st Qu.:27.70
##  Median :40.95           Median :1.544e+09           Median :30.03
##  Mean   :41.61           Mean   :1.544e+09           Mean   :30.14
##  3rd Qu.:44.12           3rd Qu.:1.545e+09           3rd Qu.:35.32
##  Max.   :57.20           Max.   :1.545e+09           Max.   :47.25
##
##  apparentTemperatureLowTime     icon              dewPoint        pressure
##  Min.   :1.543e+09          Length:693071      Min.   : 4.39   Min.   : 988.1
##  1st Qu.:1.543e+09          Class :character   1st Qu.:27.49   1st Qu.: 999.8
##  Median :1.544e+09          Mode  :character   Median :30.69   Median :1009.2
##  Mean   :1.544e+09                             Mean   :31.66   Mean   :1010.1
##  3rd Qu.:1.545e+09                             3rd Qu.:38.12   3rd Qu.:1021.9
##  Max.   :1.545e+09                             Max.   :50.67   Max.   :1035.5
##
##   windBearing      cloudCover        uvIndex        visibility.1
##  Min.   :  2.0   Min.   :0.0000   Min.   :0.0000   Min.   : 0.717
##  1st Qu.:124.0   1st Qu.:0.3700   1st Qu.:0.0000   1st Qu.: 8.432
##  Median :258.0   Median :0.8200   Median :0.0000   Median : 9.880
##  Mean   :220.1   Mean   :0.6865   Mean   :0.2489   Mean   : 8.468
##  3rd Qu.:303.0   3rd Qu.:1.0000   3rd Qu.:0.0000   3rd Qu.: 9.996
##  Max.   :356.0   Max.   :1.0000   Max.   :2.0000   Max.   :10.000
##
##      ozone        sunriseTime          sunsetTime          moonPhase
##  Min.   :269.4   Min.   :1.543e+09   Min.   :1.543e+09   Min.   :0.0900
##  1st Qu.:290.9   1st Qu.:1.543e+09   1st Qu.:1.543e+09   1st Qu.:0.3000
##  Median :307.4   Median :1.544e+09   Median :1.544e+09   Median :0.6800
##  Mean   :313.5   Mean   :1.544e+09   Mean   :1.544e+09   Mean   :0.5791
##  3rd Qu.:331.8   3rd Qu.:1.545e+09   3rd Qu.:1.545e+09   3rd Qu.:0.7900
##  Max.   :378.9   Max.   :1.545e+09   Max.   :1.545e+09   Max.   :0.9300
##
##  precipIntensityMax  uvIndexTime        temperatureMin  temperatureMinTime
##  Min.   :0.00000    Min.   :1.543e+09   Min.   :15.63   Min.   :1.543e+09
##  1st Qu.:0.00000    1st Qu.:1.543e+09   1st Qu.:30.17   1st Qu.:1.543e+09
##  Median :0.00040    Median :1.544e+09   Median :34.24   Median :1.544e+09
##  Mean   :0.03737    Mean   :1.544e+09   Mean   :33.46   Mean   :1.544e+09
##  3rd Qu.:0.09160    3rd Qu.:1.545e+09   3rd Qu.:38.88   3rd Qu.:1.545e+09
##  Max.   :0.14590    Max.   :1.545e+09   Max.   :43.10   Max.   :1.545e+09
##
```

```
##  temperatureMax  temperatureMaxTime  apparentTemperatureMin
##  Min.   :33.51   Min.   :1.543e+09   Min.   :11.81
##  1st Qu.:42.57   1st Qu.:1.543e+09   1st Qu.:27.76
##  Median :44.68   Median :1.544e+09   Median :30.13
##  Mean   :45.26   Mean   :1.544e+09   Mean   :29.73
##  3rd Qu.:46.91   3rd Qu.:1.545e+09   3rd Qu.:35.71
##  Max.   :57.87   Max.   :1.545e+09   Max.   :40.05
##
##  apparentTemperatureMinTime apparentTemperatureMax apparentTemperatureMaxTime
##  Min.   :1.543e+09          Min.   :28.95          Min.   :1.543e+09
##  1st Qu.:1.543e+09          1st Qu.:36.57          1st Qu.:1.543e+09
##  Median :1.544e+09          Median :40.95          Median :1.544e+09
##  Mean   :1.544e+09          Mean   :42.00          Mean   :1.544e+09
##  3rd Qu.:1.545e+09          3rd Qu.:44.12          3rd Qu.:1.545e+09
##  Max.   :1.545e+09          Max.   :57.20          Max.   :1.545e+09
##
```

Data Preprocessing

###Performing Data Sanity Checks before proceeding with analysis

```
#Checking the shape of the dataset
row=nrow(cab_ds)
col=ncol(cab_ds)
sprintf("The rows and colums are: %s %s",row,col)
```

```
## [1] "The rows and colums are: 693071 57"
```

```
#See whether missing values or not
sapply(cab_ds, function(x) sum(is.na(x)))
```

```
##                          id                   timestamp 
##                           0                           0 
##                        hour                         day 
##                           0                           0 
##                       month                    datetime 
##                           0                           0 
##                    timezone                      source 
##                           0                           0 
##                 destination                    cab_type 
##                           0                           0 
##                  product_id                        name 
##                           0                           0 
##                       price                    distance 
##                       55095                           0 
##            surge_multiplier                    latitude 
##                           0                           0 
##                   longitude                 temperature 
##                           0                           0 
##         apparentTemperature               short_summary 
##                           0                           0 
##                long_summary             precipIntensity 
##                           0                           0 
##           precipProbability                    humidity 
##                           0                           0 
##                   windSpeed                    windGust 
##                           0                           0 
##                windGustTime                  visibility 
##                           0                           0 
##             temperatureHigh         temperatureHighTime 
##                           0                           0 
##              temperatureLow          temperatureLowTime 
##                           0                           0 
##     apparentTemperatureHigh apparentTemperatureHighTime 
##                           0                           0 
##      apparentTemperatureLow  apparentTemperatureLowTime 
##                           0                           0 
##                        icon                    dewPoint 
##                           0                           0 
##                    pressure                 windBearing 
##                           0                           0 
##                  cloudCover                     uvIndex 
##                           0                           0 
##                visibility.1                       ozone 
##                           0                           0 
##                 sunriseTime                  sunsetTime 
##                           0                           0 
##                   moonPhase          precipIntensityMax 
##                           0                           0 
##                 uvIndexTime              temperatureMin 
##                           0                           0 
##          temperatureMinTime              temperatureMax 
##                           0                           0 
##          temperatureMaxTime      apparentTemperatureMin
##                           0                           0
##  apparentTemperatureMinTime      apparentTemperatureMax
##                           0                           0
##  apparentTemperatureMaxTime
##                           0
```

```
cab_ds_distinct <-  na.omit(cab_ds)
cab_ds_distinct<- cab_ds_distinct %>% distinct()
print(paste("The number of records removed : ", nrow(cab_ds) - nrow(cab_ds_distinct)))
```

```
## [1] "The number of records removed :  55095"
```

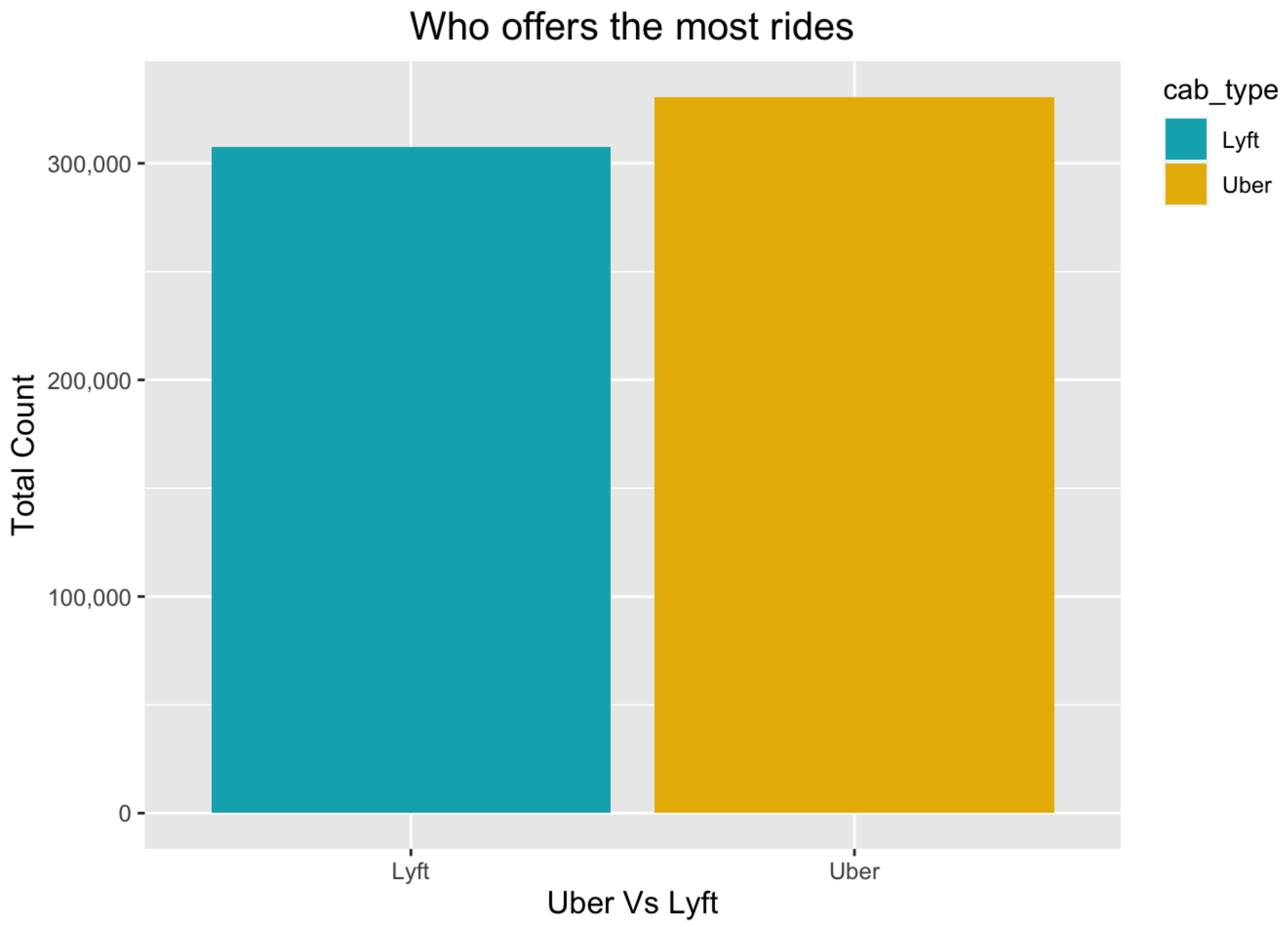
Exploratory Data Analysis

Who offers the most rides,Uber or lyft?

```
 cab_ds_distinct %>% group_by(cab_type) %>%
    summarise("Total_ Count" = length(id),
              'Percentage' = (length(id) / nrow(cab_ds_distinct)) * 100)
```

```
## # A tibble: 2 × 3
##   cab_type `Total_ Count` Percentage
##   <chr>             <int>      <dbl>
## 1 Lyft             307408       48.2
## 2 Uber             330568       51.8
```

```
bp <-ggplot()+
  geom_bar(data=cab_ds_distinct,mapping=aes(x=cab_type, fill=cab_type))+
  scale_y_continuous(breaks = seq(0,1000000,100000),labels=scales::comma)+
  labs(x="Uber Vs Lyft",
       y="Total Count")+
  labs(title="Who offers the most rides")+
  theme(plot.title =element_text(hjust = 0.50,size=15),
        legend.justification = c("right", "top"),
       axis.title = element_text(size=12),
        axis.text = element_text(size=09))+
  theme(plot.caption=element_text(size=10))

bp + scale_fill_manual(values = c("#00AFBB", "#E7B800"))
```

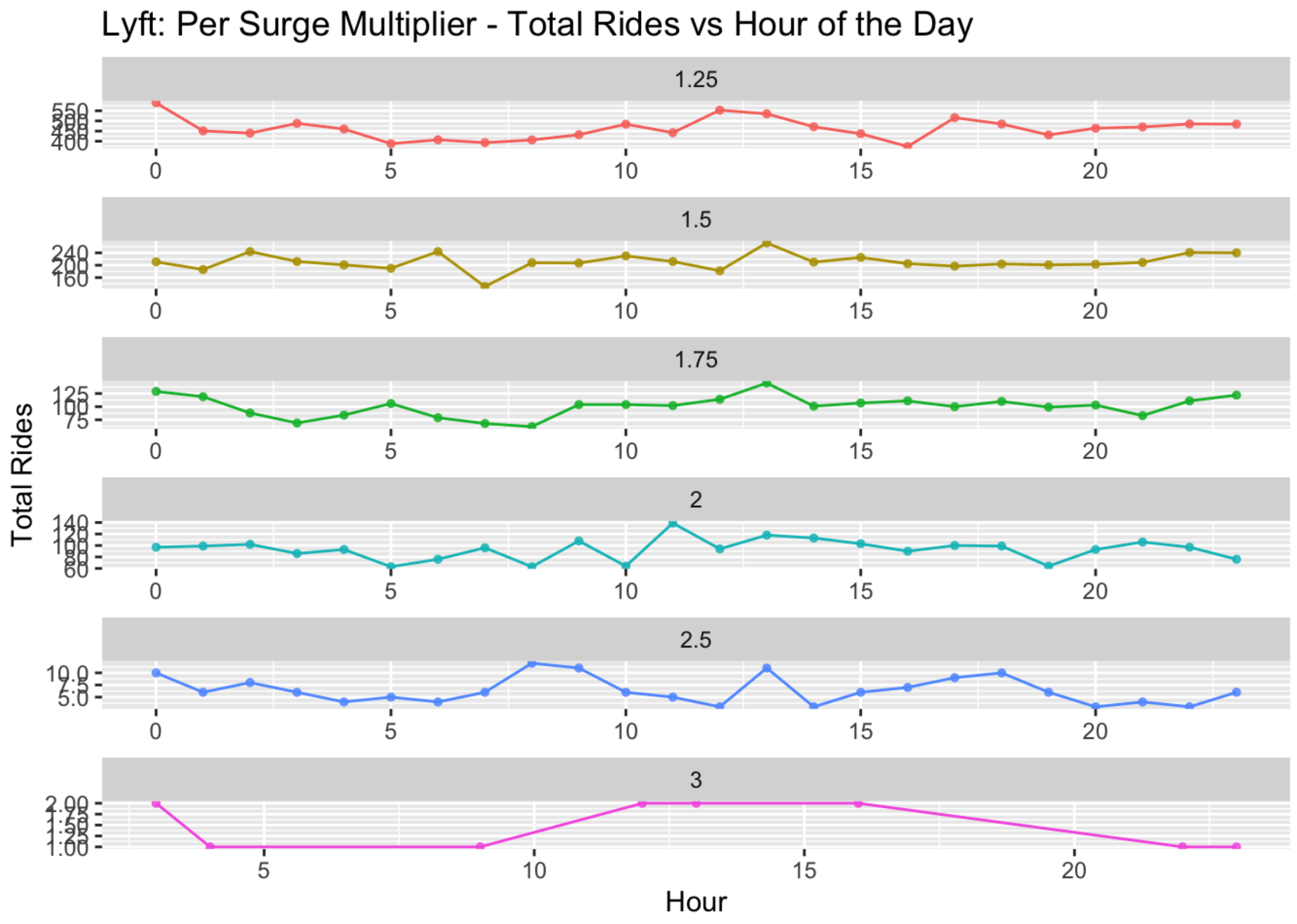


Lyft: Per Surge Multiplier - Total Rides vs Hour of the Day

```
surged_data <- cab_ds %>%
        filter(cab_type == "Lyft", surge_multiplier > 1.00) %>%
        dplyr::group_by(hour, surge_multiplier) %>%
        dplyr::summarize(total_rides = n())
```

```
## `summarise()` has grouped output by 'hour'. You can override using the
## `.groups` argument.
```

```
surged_data$surge_multiplier <- as.factor(surged_data$surge_multiplier)

lyft_surged_data <- ggplot(surged_data, aes(hour, total_rides, color = surge_multiplie
r)) +
        geom_point(alpha=0.8, size=1, aes(color = surge_multiplier)) +
        geom_line(aes(color = surge_multiplier)) + ggtitle("Lyft: Per Surge Multiplier -
Total Rides vs Hour of the Day") +
        facet_wrap(~surge_multiplier, ncol=1, scales="free") + xlab("Hour") + ylab("Tota
l Rides") +
        guides(color=guide_legend(ncol=1)) + theme(legend.position="none",
                                                   panel.border = element_blank(),
                                                   panel.spacing.x = unit(0,"line"))
lyft_surged_data
```



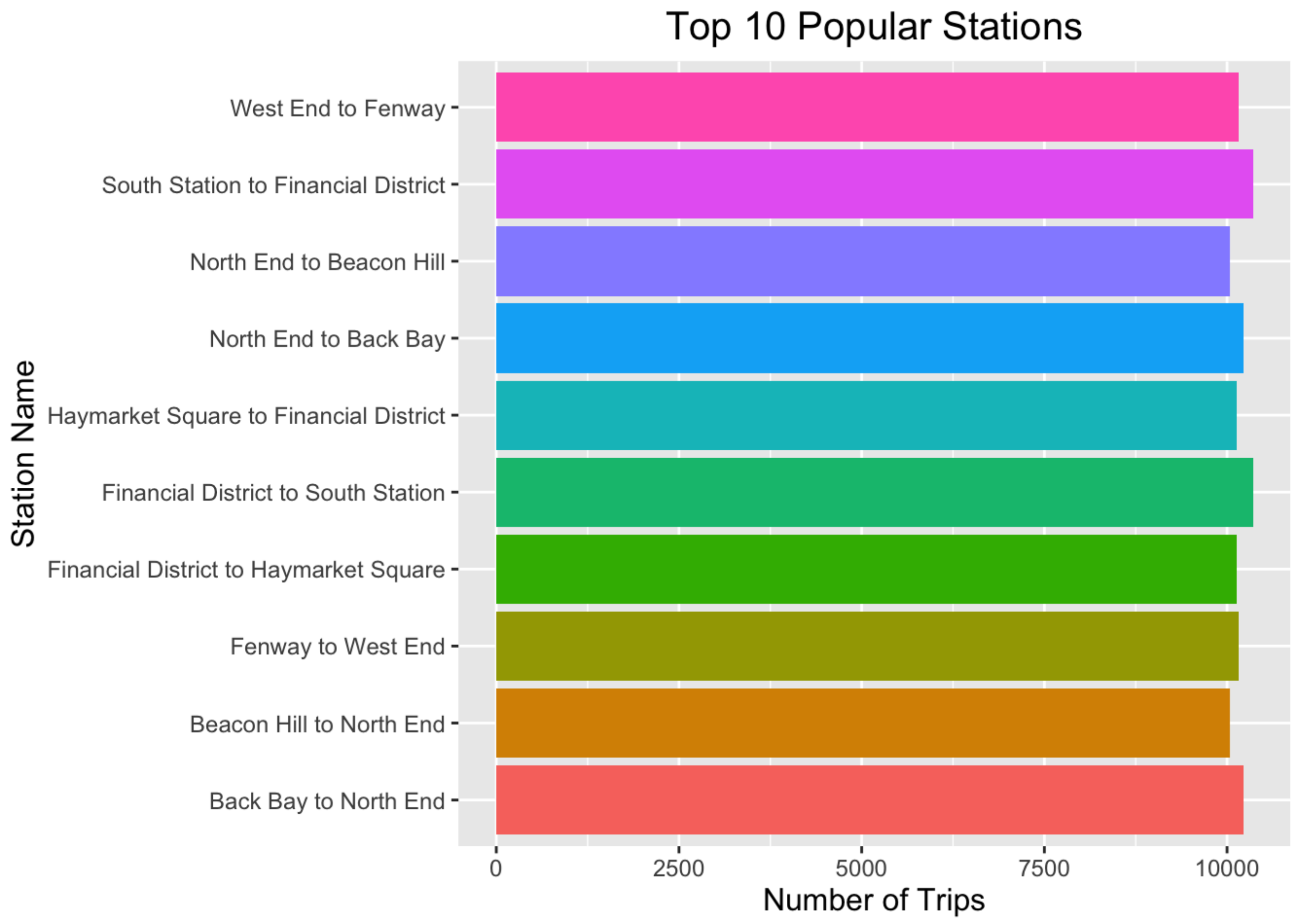
minimum and maximum fare prices

```
df<-sqldf("select source ,destination, cab_type ,avg(price) as average_price,min(price)
 as minimun_price,max(price) as maximum_price from cab_ds group by source, destination,c
ab_type order by cab_type")
CrossTable(cab_ds$surge_multiplier, cab_ds$cab_type)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## | Chi-square contribution |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  693071
##
##
##                         | cab_ds$cab_type
## cab_ds$surge_multiplier |      Lyft |      Uber | Row Total |
## ------------------------|-----------|-----------|-----------|
##                       1 |    286433 |    385663 |    672096 |
##                         |   456.978 |   364.253 |           |
##                         |     0.426 |     0.574 |     0.970 |
##                         |     0.932 |     1.000 |           |
##                         |     0.413 |     0.556 |           |
## ------------------------|-----------|-----------|-----------|
##                    1.25 |     11085 |         0 |     11085 |
##                         |  7738.535 |  6168.307 |           |
##                         |     1.000 |     0.000 |     0.016 |
##                         |     0.036 |     0.000 |           |
##                         |     0.016 |     0.000 |           |
## ------------------------|-----------|-----------|-----------|
##                     1.5 |      5065 |         0 |      5065 |
##                         |  3535.921 |  2818.446 |           |
##                         |     1.000 |     0.000 |     0.007 |
##                         |     0.016 |     0.000 |           |
##                         |     0.007 |     0.000 |           |
## ------------------------|-----------|-----------|-----------|
##                    1.75 |      2420 |         0 |      2420 |
##                         |  1689.423 |  1346.622 |           |
##                         |     1.000 |     0.000 |     0.003 |
##                         |     0.008 |     0.000 |           |
##                         |     0.003 |     0.000 |           |
## ------------------------|-----------|-----------|-----------|
##                       2 |      2239 |         0 |      2239 |
##                         |  1563.065 |  1245.903 |           |
##                         |     1.000 |     0.000 |     0.003 |
##                         |     0.007 |     0.000 |           |
##                         |     0.003 |     0.000 |           |
## ------------------------|-----------|-----------|-----------|
##                     2.5 |       154 |         0 |       154 |
##                         |   107.509 |    85.694 |           |
##                         |     1.000 |     0.000 |     0.000 |
##                         |     0.001 |     0.000 |           |
##                              |     0.000 |     0.000 |           |
## -----------------------------|-----------|-----------|-----------|
##                            3 |        12 |         0 |        12 |
##                              |     8.377 |     6.677 |           |
##                              |     1.000 |     0.000 |     0.000 |
##                              |     0.000 |     0.000 |           |
##                              |     0.000 |     0.000 |           |
## -----------------------------|-----------|-----------|-----------|
##                 Column Total |    307408 |    385663 |    693071 |
##                              |     0.444 |     0.556 |           |
## -----------------------------|-----------|-----------|-----------|
## 
## 
```

Top 10 most Popular Stations

```
popular_station<-sqldf("select source,destination,(source|| ' to ' ||destination) as sta
tion_name,count(id) as number_of_trips from cab_ds group by source,destination order by
 count(id) desc LIMIT 10")
ggplot(data=popular_station,aes(x=number_of_trips, y=station_name, fill=station_name))+g
eom_bar(stat='identity')+
  labs(x="Number of Trips",y="Station Name")+
  labs(title=" Top 10 Popular Stations ")+
  theme(plot.title =element_text(hjust = 0.5,size=15),
        legend.position = c(2.50, .50),
        legend.justification = c("right", "top"),
       axis.title = element_text(size=12),
        axis.text = element_text(size=09))+
  theme(plot.caption=element_text(size=10))
```

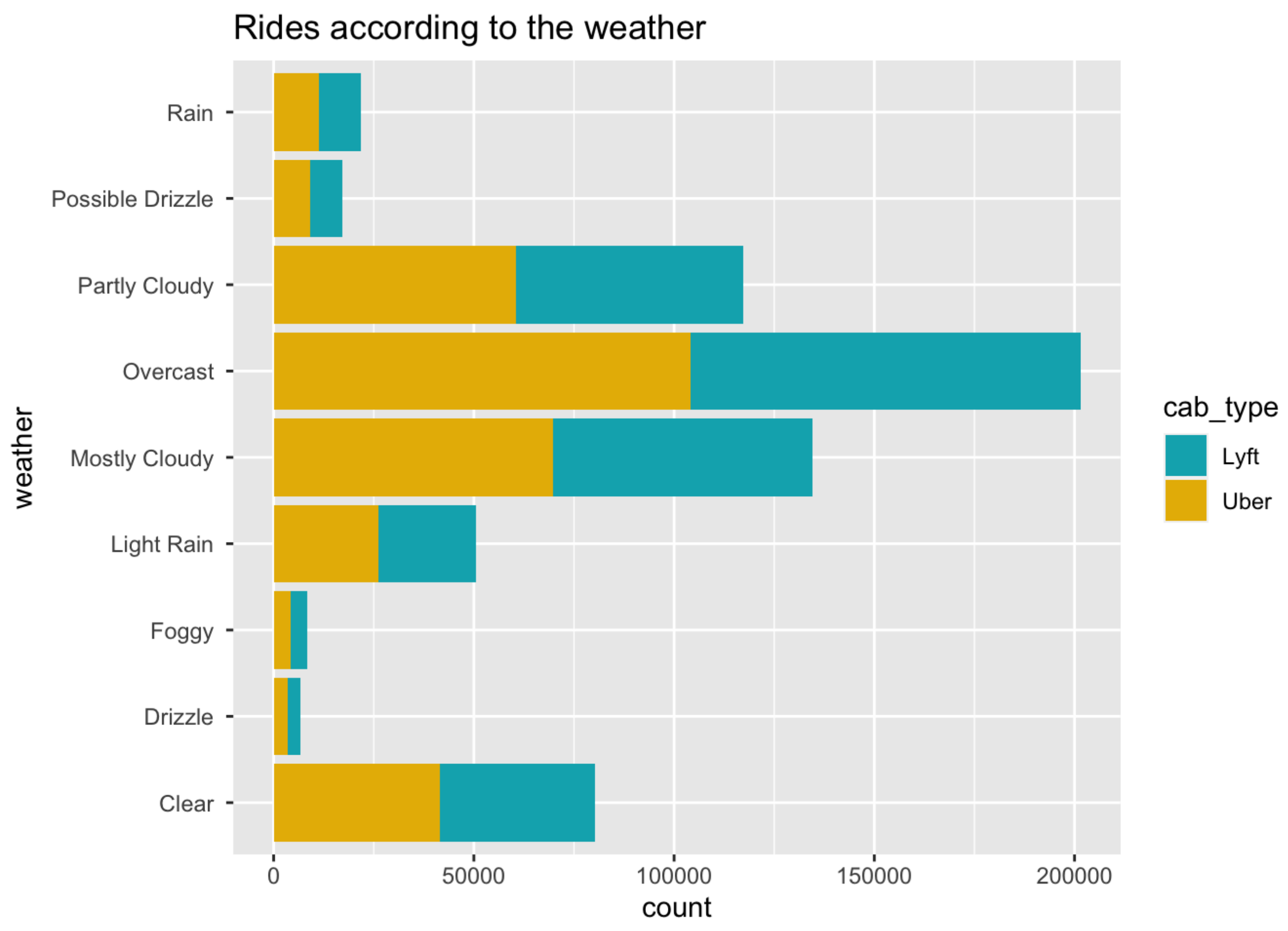


Weather affects the rides

```
cab_ds_distinct %>% group_by(short_summary) %>%
summarise(count = length(id),'Percentage' = (length(id) / nrow(cab_ds_distinct)) * 100)
```

```
## # A tibble: 9 × 3
##   short_summary         count Percentage
##   <chr>                 <int>      <dbl>
## 1 " Clear "             80256      12.6
## 2 " Drizzle "            6725       1.05
## 3 " Foggy "              8292       1.30
## 4 " Light Rain "        50488       7.91
## 5 " Mostly Cloudy "    134603      21.1
## 6 " Overcast "         201429      31.6
## 7 " Partly Cloudy "    117226      18.4
## 8 " Possible Drizzle "  17176       2.69
## 9 " Rain "              21781       3.41
```

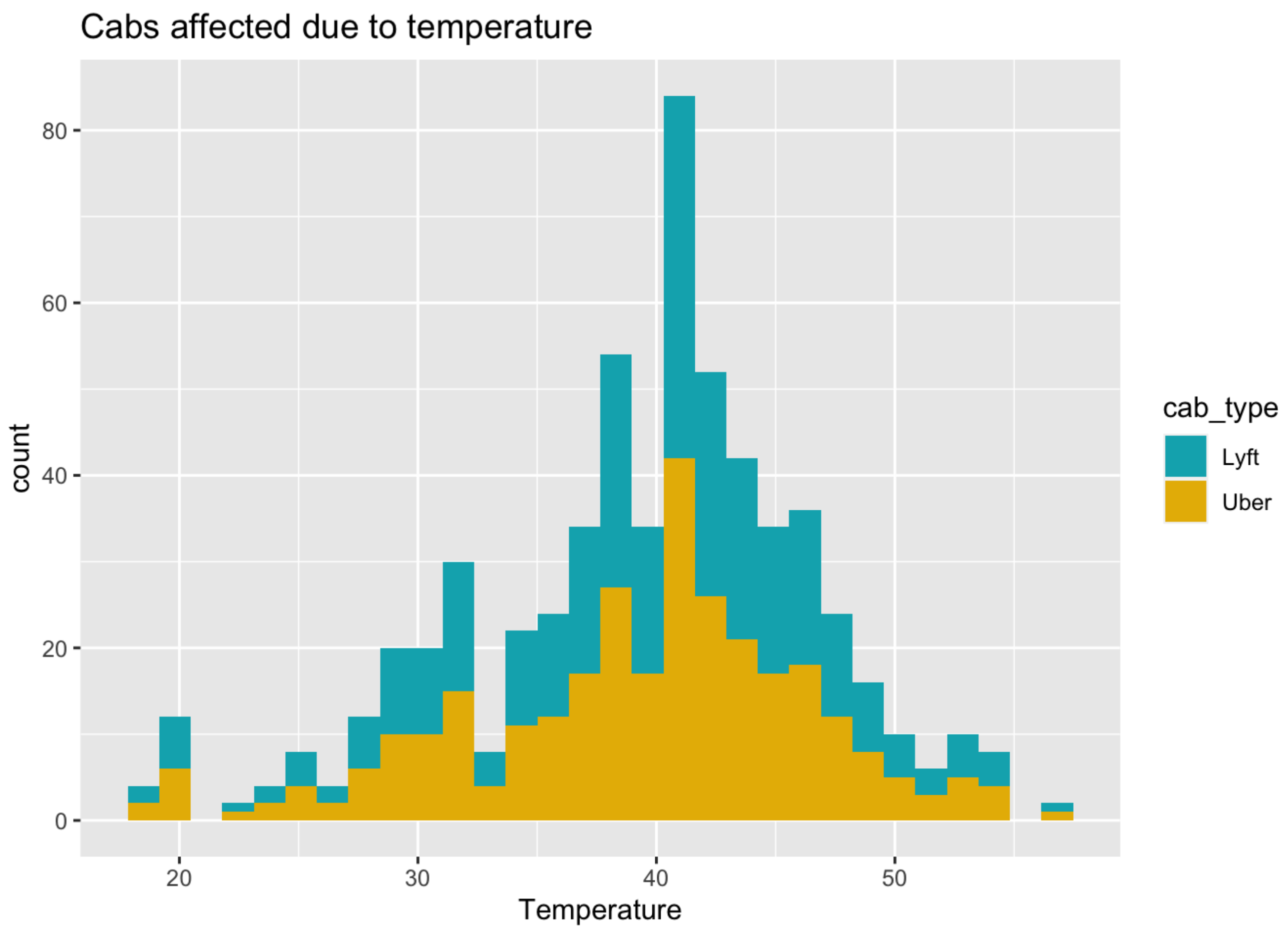
```
bp <- cab_ds_distinct %>%
    ggplot(aes(short_summary, fill=cab_type)) +
    labs(x="weather", title="Rides according to the weather") +
    geom_bar()+ coord_flip()

bp + scale_fill_manual(values = c("#00AFBB", "#E7B800"))
```



Temperature affects the ride’s price

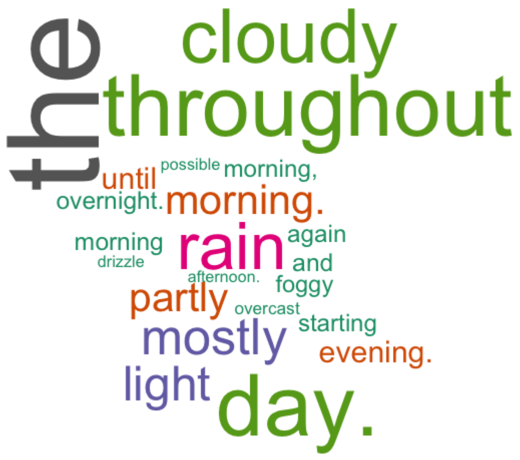
```
df2<-sqldf("select temperature, price , cab_type from cab_ds group by cab_type,temperatu
re")
bp <- df2 %>%
    ggplot(aes(temperature, fill=cab_type)) +
    labs(x="Temperature", title="Cabs affected due to temperature") +
    geom_histogram()
bp + scale_fill_manual(values = c("#00AFBB", "#E7B800"))
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



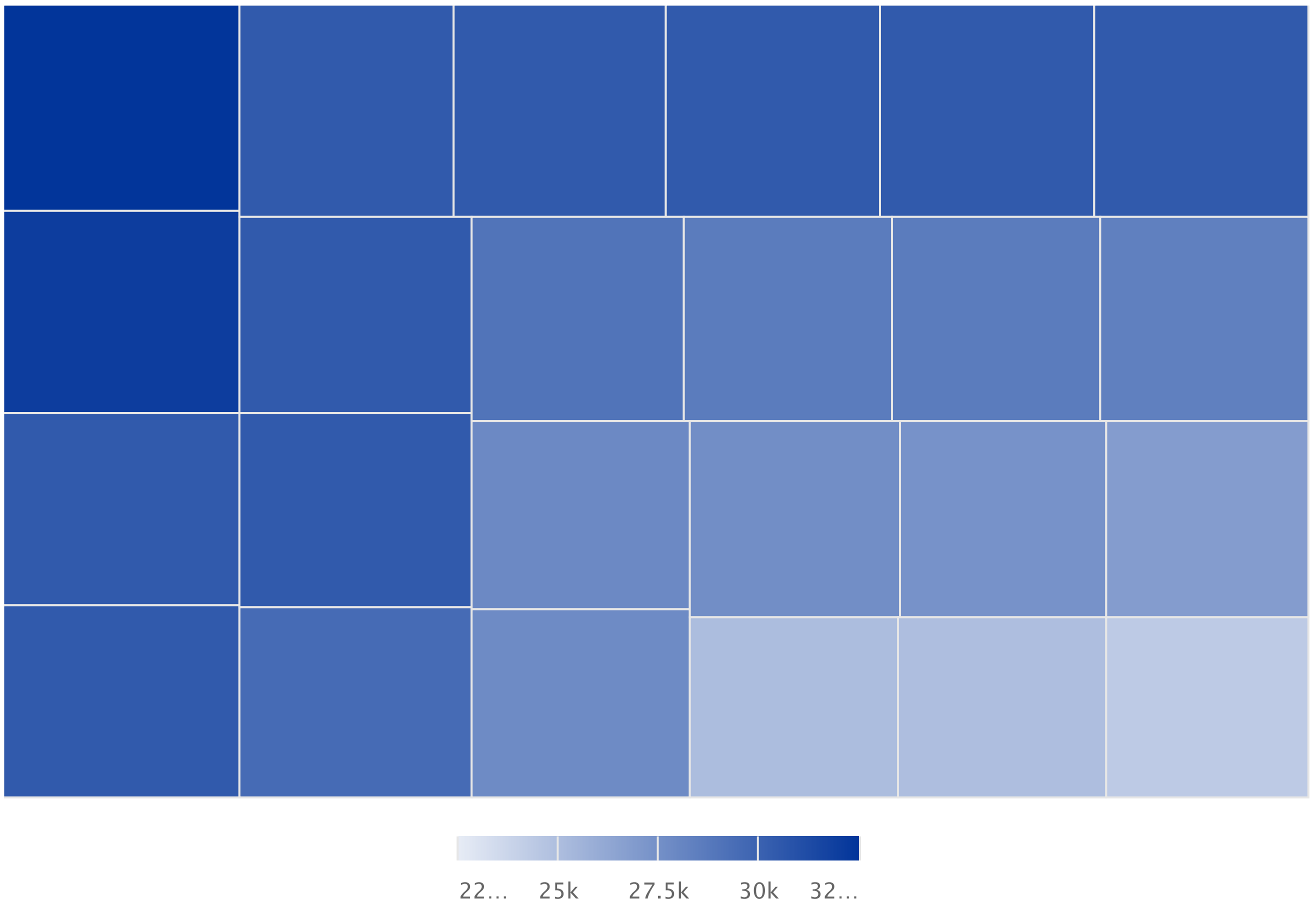
weather the passengers opt for cabs

```
document_tm <- Corpus(VectorSource(cab_ds$long_summary))
mat <- as.matrix(TermDocumentMatrix(document_tm))
vec <- sort(rowSums(mat), decreasing = TRUE)
word_corpus <- data.frame(word = names(vec), freq = vec)
set.seed(3)
wordcloud(word_corpus$word, freq = word_corpus$freq, colors = brewer.pal(8, "Dark2"))
```



Time division on basis of hour

```
cab_ds %>% group_by(hour) %>% summarise(n = n()) %>% arrange(desc(n)) %>% hchart(type =
"treemap", hcaes(name = hour, x = hour, value = n, color = n))
```



Price range between Uber and Lyft

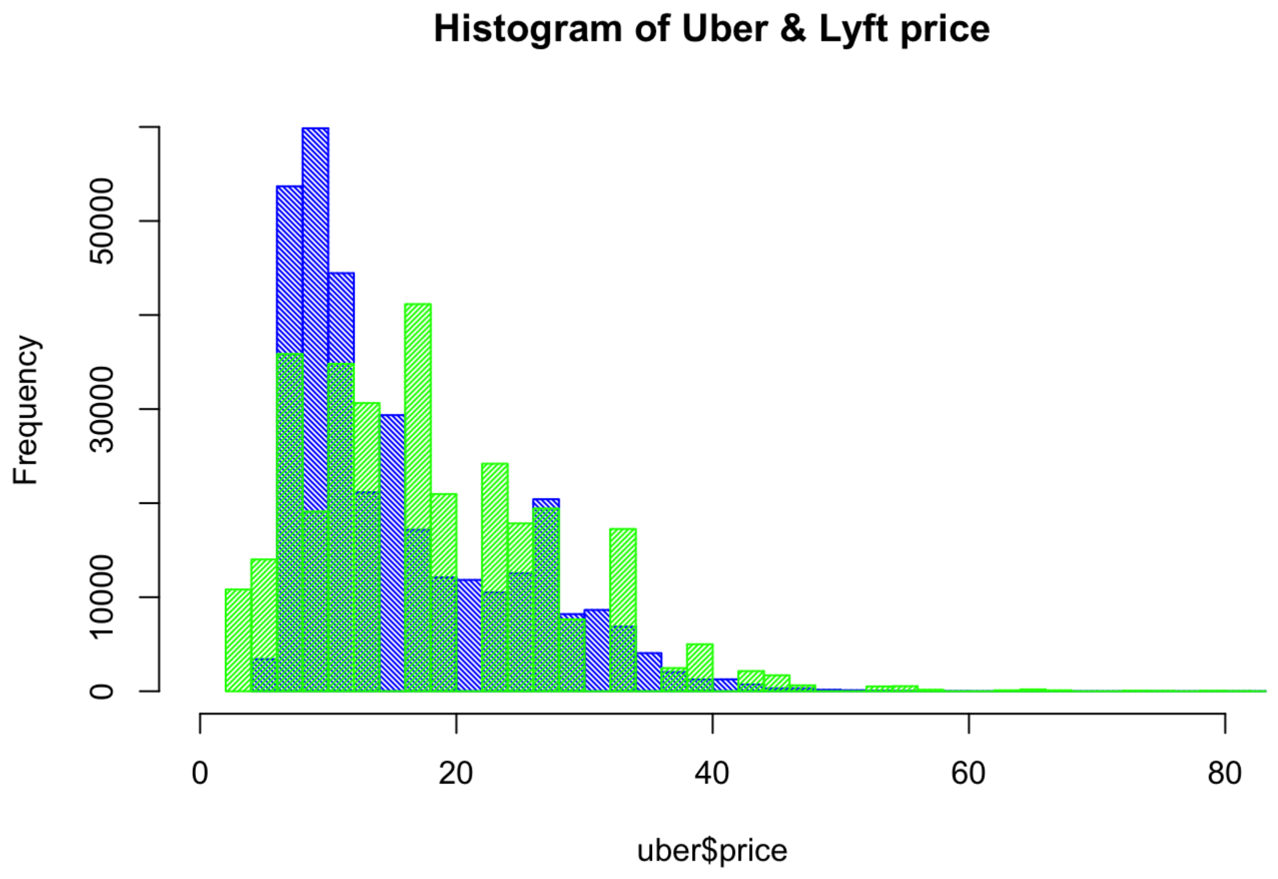
```
lyft<-sqldf("select * from cab_ds where cab_type='Lyft'")
uber<-sqldf("select * from cab_ds where cab_type='Uber'")
summary(lyft$price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##    2.50    9.00   16.50   17.35   22.50   97.50 
```

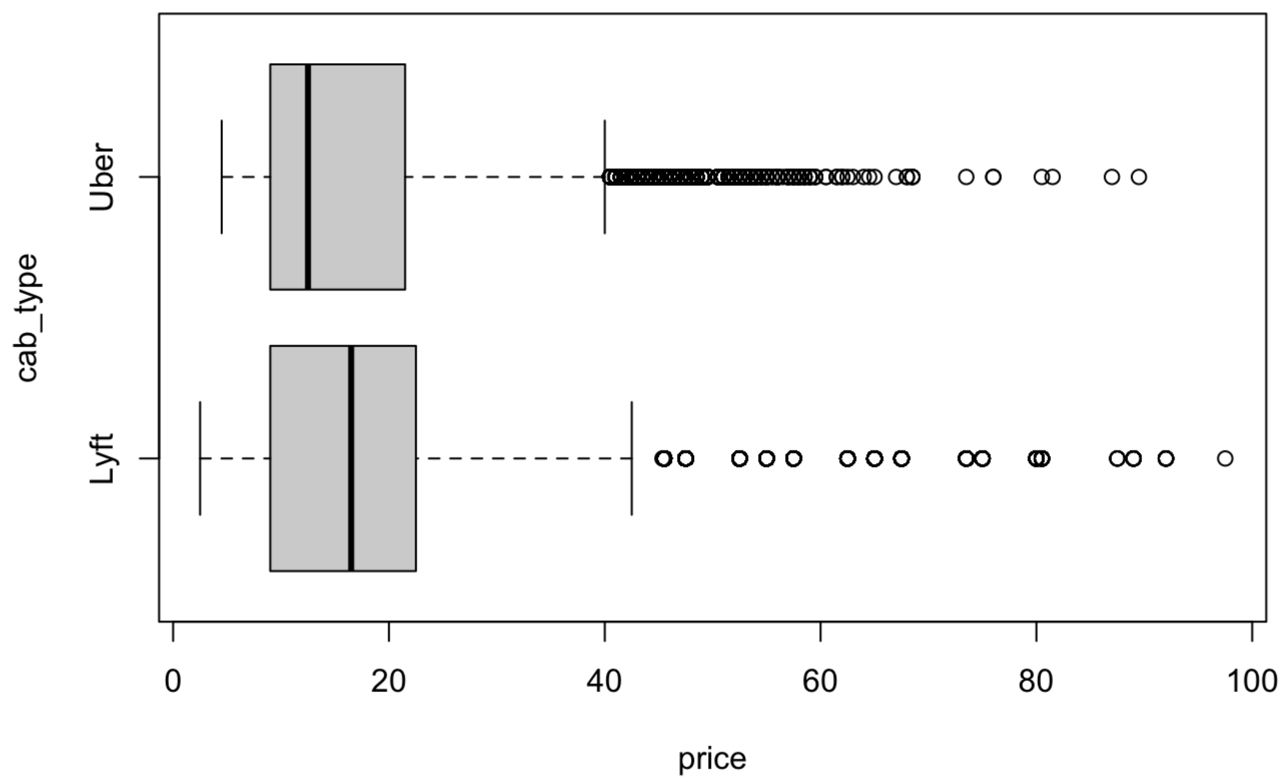
```
summary(uber$price)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's 
##     4.5     9.0    12.5    15.8    21.5    89.5   55095 
```

```
hist(uber$price, col = "blue", density = 50, angle = 135, breaks = 40, xlim = c(0,80), m
ain = "Histogram of Uber & Lyft price")
hist(lyft$price, col = "green", density = 50, add = TRUE, breaks = 40)
```

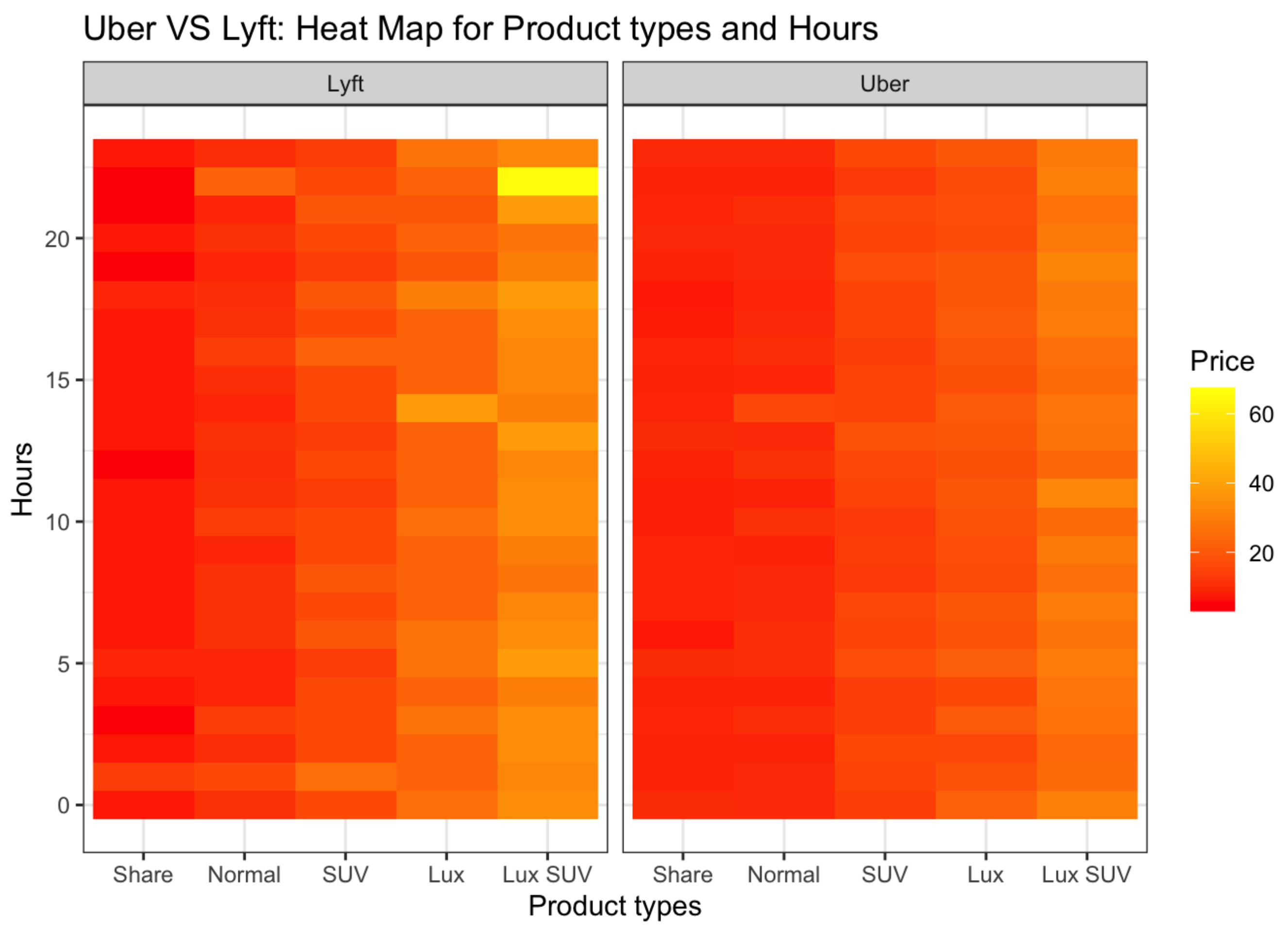


```
boxplot(cab_ds$price~cab_ds$cab_type,xlab='price', ylab='cab_type', data= cab_ds, horizo
ntal = TRUE)
```



Heatmap for specific location and hours

```
bt<-cab_ds %>% select(price,cab_type,name,distance,short_summary,hour,source,destinatio
n) %>% filter(name!="WAV") %>% filter(name!="Lux") %>% filter(price>=0)
bt$name_f<-factor(bt$name,
                            levels=c("UberPool","Shared","UberX","Lyft","UberXL","Lyft X
L","Black","Lux Black","Black SUV","Lux Black XL"))
levels(bt$name_f) <- list("Share" = c("UberPool","Shared"),
                             "Normal" =  c("UberX","Lyft"),
                             "SUV" = c("UberXL","Lyft XL"),
                             "Lux" = c("Black","Lux Black"),
                             "Lux SUV"= c("Black SUV","Lux Black XL"))

bt<-bt %>% select(price,cab_type,name,name_f,distance,short_summary,hour,source,destinat
ion) %>% filter(name!="WAV") %>% filter(name!="Lux") %>% filter(price>=0)
bt1<-bt %>% select(price,cab_type,name_f,hour,source, destination) %>% filter(destinatio
n=="Northeastern University") %>% filter(source=="Theatre District")  %>% filter(price>=
0)
ggplot(bt1, aes(name_f,hour ))+
  geom_raster(aes(fill = price))+
  scale_fill_gradientn(colours=c("red","yellow"),name="Price")+
  labs(title ="Uber VS Lyft: Heat Map for Product types and Hours", x = "Product types",
y = "Hours")+
  theme_bw()+facet_wrap(~cab_type)
```



Data Modelling

Loading pre processed Data and factoring required columns

Split data to train and test

```
wd <- weekdays(as.POSIXlt(cab_ds$datetime), abbreviate = TRUE)

cab_ds['Fri'] = as.integer(wd=='Fri')
cab_ds['Sat'] = as.integer(wd=='Sat')
cab_ds['Sun'] = as.integer(wd=='Sun')

#change short Summary of weather to binary variables
ss_data <- unique(cab_ds$short_summary)
for (i in ss_data)
      {
        cab_ds[i] = as.integer(cab_ds$name == i)

       }

for (p in unique(cab_ds$name))
      {
          cab_ds[p] = as.integer(cab_ds$name == p)
      }

lyft<-sqldf("select [distance],[surge_multiplier],[Fri], [Sat],[Sun],[Shared],[Lyft XL],
[Lux Black XL], [LUX],[Lux Black],[ Mostly Cloudy ], [ Rain ], [ Partly Cloudy ],[ Overc
ast ], [ Light Rain ], [ Foggy ], [ Possible Drizzle ],[ Drizzle ], price from cab_ds wh
ere cab_type='Lyft'")
uber<-sqldf("select [distance],[surge_multiplier],[Fri], [Sat],[Sun],[UberPool],[UberX
L],[Black],[Black SUV], [WAV],[ Mostly Cloudy ], [ Rain ], [ Partly Cloudy ],[ Overcast
 ], [ Light Rain ], [ Foggy ], [ Possible Drizzle ],[ Drizzle ], price from cab_ds where
cab_type='Uber'")

colnames(uber)[9] ="Black_SUV"
colnames(uber)[11] ="Mostly_Cloudy"
colnames(uber)[12] ="Rain"
colnames(uber)[13] ="Partly_Cloudy"
colnames(uber)[14] ="Overcast"
colnames(uber)[15] ="Light_Rain"
colnames(uber)[16] ="Foggy"
colnames(uber)[17] ="Possible_Drizzle"
colnames(uber)[18] ="Drizzle"


colnames(lyft)[7] ="Lyft_XL"
colnames(lyft)[8] ="Lux_Black_XL"
colnames(lyft)[10] ="Lux_Black"
colnames(lyft)[11] ="Mostly_Cloudy"
colnames(lyft)[12] ="Rain"
colnames(lyft)[13] ="Partly_Cloudy"
colnames(lyft)[14] ="Overcast"
colnames(lyft)[15] ="Light_Rain"
colnames(lyft)[16] ="Foggy"
colnames(lyft)[17] ="Possible_Drizzle"
colnames(lyft)[18] ="Drizzle"

#Uber
```

```
#selecting on numeric data
numeric_index = sapply(uber,is.numeric)
numeric_data = uber[,numeric_index]

#divide into train & test
train_index = sample(1:nrow(uber), 0.9 * nrow(uber))
uber_train = uber[train_index,]
uber_test = uber[-train_index,]

uber_train<-na.omit(uber_train)
sapply(uber_train, function(x) sum(is.na(x)))
```

```
##         distance surge_multiplier              Fri              Sat 
##                0                0                0                0 
##              Sun         UberPool           UberXL            Black 
##                0                0                0                0 
##        Black_SUV              WAV    Mostly_Cloudy             Rain 
##                0                0                0                0 
##    Partly_Cloudy         Overcast       Light_Rain            Foggy 
##                0                0                0                0 
## Possible_Drizzle          Drizzle            price 
##                0                0                0
```

```
uber_test<-na.omit(uber_test)
sapply(uber_test, function(x) sum(is.na(x)))
```

```
##         distance surge_multiplier              Fri              Sat 
##                0                0                0                0 
##              Sun         UberPool           UberXL            Black 
##                0                0                0                0 
##        Black_SUV              WAV    Mostly_Cloudy             Rain 
##                0                0                0                0 
##    Partly_Cloudy         Overcast       Light_Rain            Foggy 
##                0                0                0                0 
## Possible_Drizzle          Drizzle            price 
##                0                0                0
```

```
#lyft
#selecting on numeric data
numeric_index = sapply(lyft,is.numeric)
numeric_data = uber[,numeric_index]

#divide into train & test
train_index = sample(1:nrow(lyft), 0.9 * nrow(lyft))
lyft_train = lyft[train_index,]
lyft_test = lyft[-train_index,]

lyft_train<-na.omit(lyft_train)
sapply(lyft_train, function(x) sum(is.na(x)))
```

```
##          distance surge_multiplier              Fri              Sat 
##                 0                0                0                0 
##               Sun           Shared          Lyft_XL     Lux_Black_XL 
##                 0                0                0                0 
##               Lux        Lux_Black    Mostly_Cloudy             Rain 
##                 0                0                0                0 
##     Partly_Cloudy         Overcast       Light_Rain            Foggy 
##                 0                0                0                0 
##  Possible_Drizzle          Drizzle            price 
##                 0                0                0
```

```
lyft_test<-na.omit(lyft_test)
sapply(lyft_test, function(x) sum(is.na(x)))
```

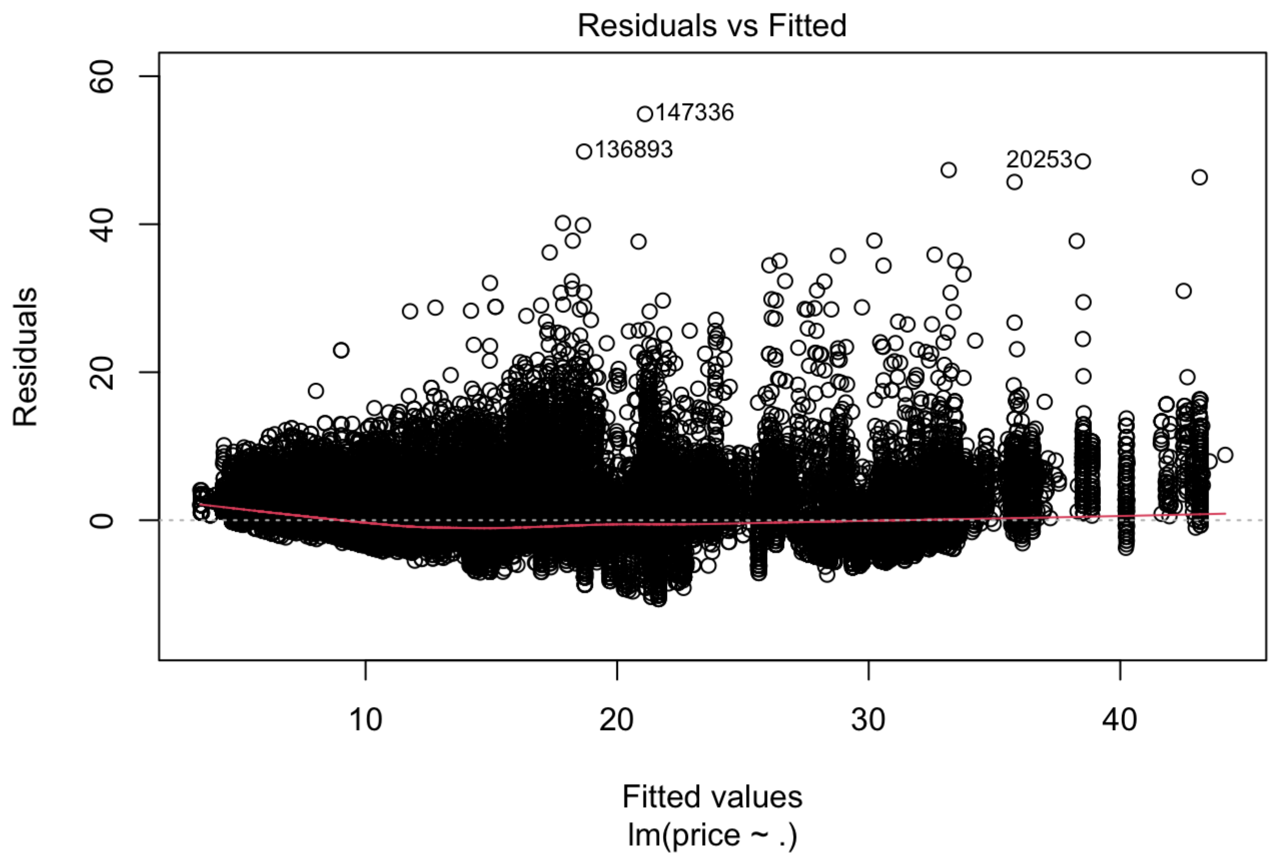
```
##          distance surge_multiplier              Fri              Sat 
##                 0                0                0                0 
##               Sun           Shared          Lyft_XL     Lux_Black_XL 
##                 0                0                0                0 
##               Lux        Lux_Black    Mostly_Cloudy             Rain 
##                 0                0                0                0 
##     Partly_Cloudy         Overcast       Light_Rain            Foggy 
##                 0                0                0                0 
##  Possible_Drizzle          Drizzle            price 
##                 0                0                0
```

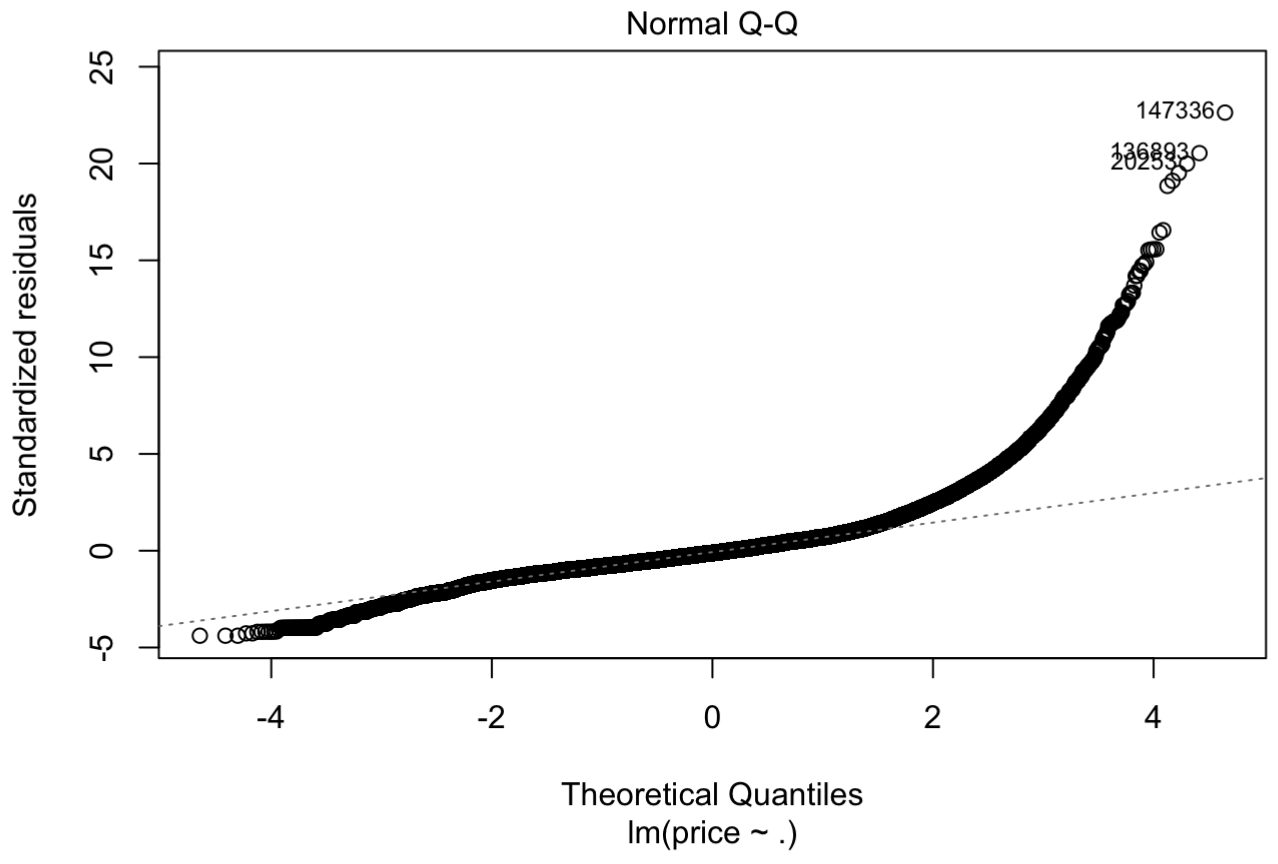
Linear Regression

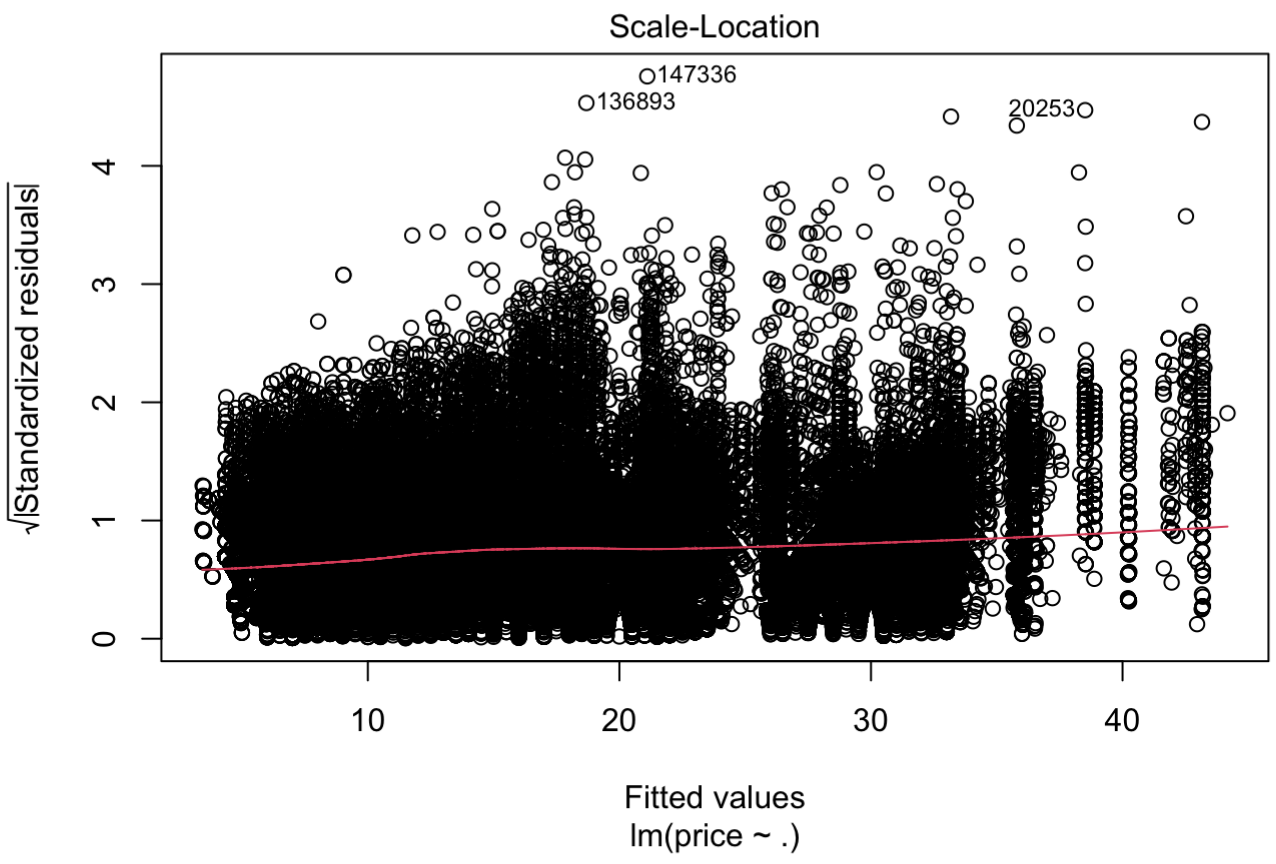
```
#Uber
uber_lm_model = lm(price ~., data = uber_train)
summary(uber_lm_model)
```

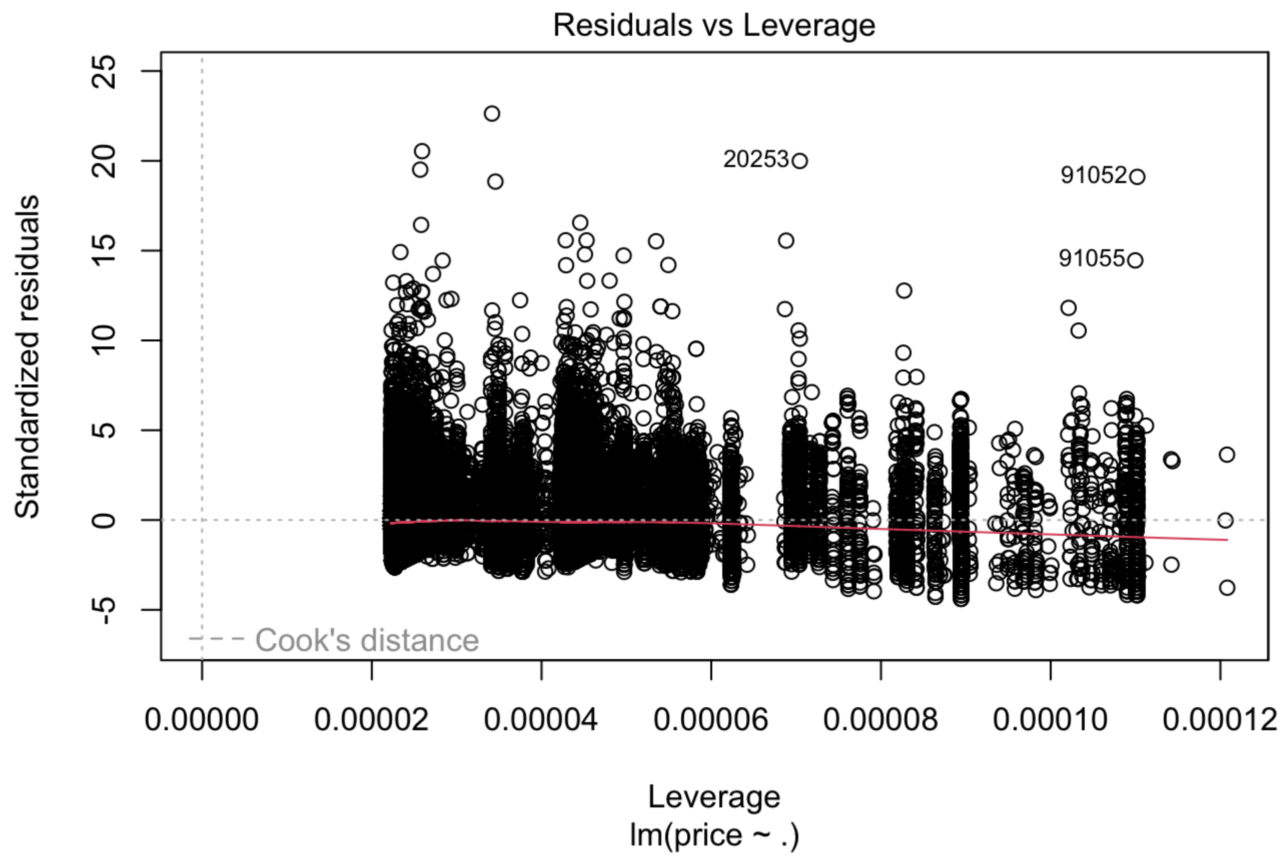
```
##
## Call:
## lm(formula = price ~ ., data = uber_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -10.648  -1.420  -0.283   1.074  54.892
##
## Coefficients: (9 not defined because of singularities)
##                   Estimate Std. Error  t value Pr(>|t|)
## (Intercept)       4.407800   0.014150  311.500   <2e-16 ***
## distance          2.445569   0.003772  648.374   <2e-16 ***
## surge_multiplier        NA         NA       NA       NA
## Fri              -0.022536   0.013601   -1.657   0.0975 .
## Sat               0.012649   0.013599    0.930   0.3523
## Sun              -0.031185   0.013512   -2.308   0.0210 *
## UberPool         -1.003611   0.015404  -65.151   <2e-16 ***
## UberXL            5.915176   0.015413  383.776   <2e-16 ***
## Black            10.773591   0.015407  699.253   <2e-16 ***
## Black_SUV        20.529696   0.015424 1331.030   <2e-16 ***
## WAV               0.002248   0.015417    0.146   0.8841
## Mostly_Cloudy           NA         NA       NA       NA
## Rain                    NA         NA       NA       NA
## Partly_Cloudy           NA         NA       NA       NA
## Overcast                NA         NA       NA       NA
## Light_Rain              NA         NA       NA       NA
## Foggy                   NA         NA       NA       NA
## Possible_Drizzle        NA         NA       NA       NA
## Drizzle                 NA         NA       NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.425 on 297500 degrees of freedom
## Multiple R-squared:  0.9198, Adjusted R-squared:  0.9198
## F-statistic: 3.79e+05 on 9 and 297500 DF,  p-value: < 2.2e-16
```

```
plot (uber_lm_model)
```


Residuals vs Fitted
147336
136893
20253
Residuals
60
40
20
0
10
20
30
40
Fitted values
lm(price ~ .)


Normal Q-Q
147336
136893
20253
Standardized residuals
25
20
15
10
5
0
-5
-4
-2
0
2
4
Theoretical Quantiles
lm(price ~ .)


Scale-Location
√|Standardized residuals|
Fitted values
lm(price ~ .)
147336
136893
20253
0
1
2
3
4
10
20
30
40


Residuals vs Leverage
Standardized residuals
Leverage
lm(price ~ .)
20253
91052
91055
Cook's distance
-5
0
5
10
15
20
25
0.00000
0.00002
0.00004
0.00006
0.00008
0.00010
0.00012

```
#prediction
uber_pred = predict(uber_lm_model, uber_test[,1:18])
```

```
## Warning in predict.lm(uber_lm_model, uber_test[, 1:18]): prediction from a rank-
## deficient fit may be misleading
```

```
#Correlation Matrix
actuals_predicts <- data.frame(cbind(actuals=uber_test$price, predicteds=uber_pred))
correlation_accuracy <- cor(actuals_predicts)
correlation_accuracy
```

```
##              actuals predicteds
## actuals    1.0000000  0.9591117
## predicteds 0.9591117  1.0000000
```

```
#Evaluation
mat_lr_uber<- regr.eval(uber_test[,19], uber_pred)#, stats = c('mape','rmse'))
print(mat_lr_uber)
```

```
##       mae       mse      rmse      mape
## 1.6697108 5.8347082 2.4155141 0.1191017
```

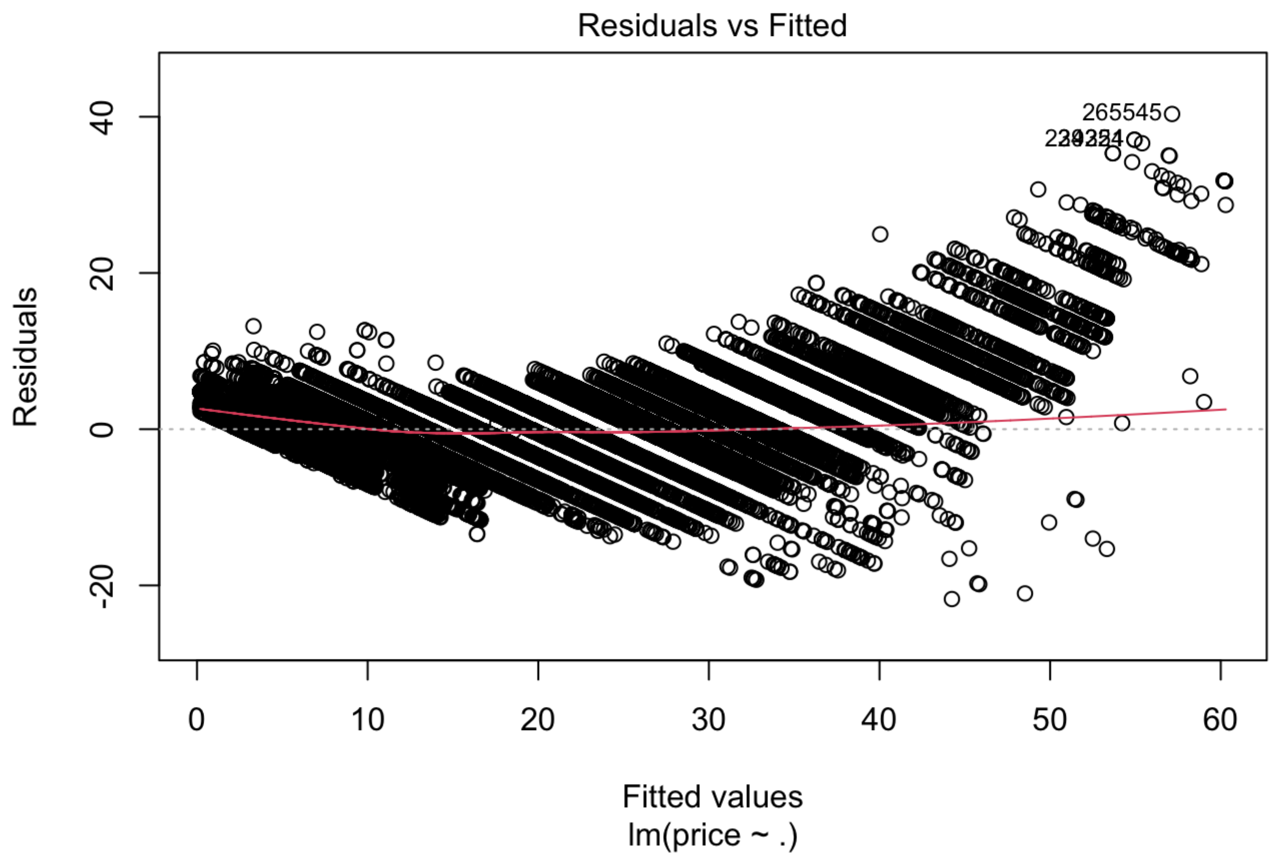
```
errors = abs(uber_pred - uber_test$price)
mape = 100 * (errors / uber_test$price)
uber_lr_accuracy = 100 - mean(mape)
sprintf("The Accuracy of Linear Regression for Uber :%f",uber_lr_accuracy)
```

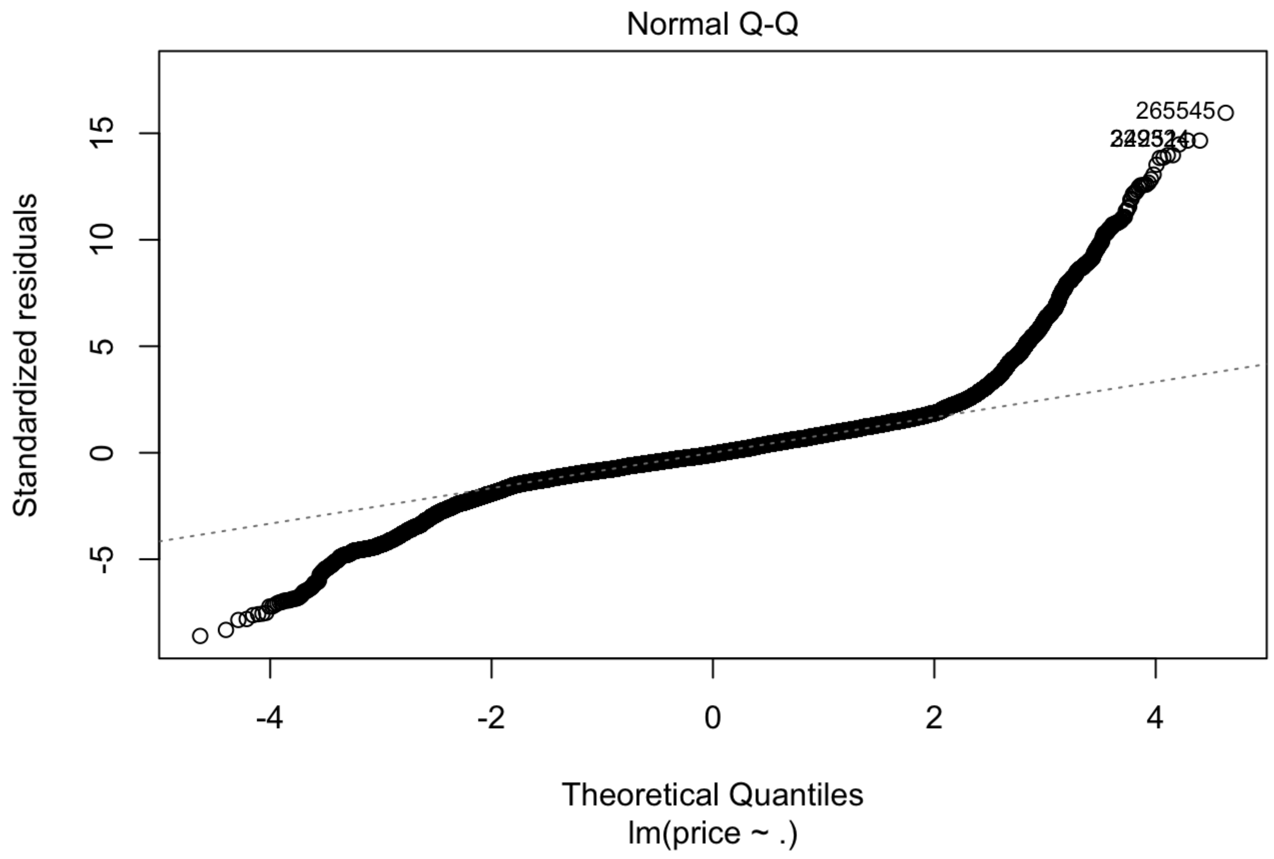
```
## [1] "The Accuracy of Linear Regression for Uber :88.089834"
```

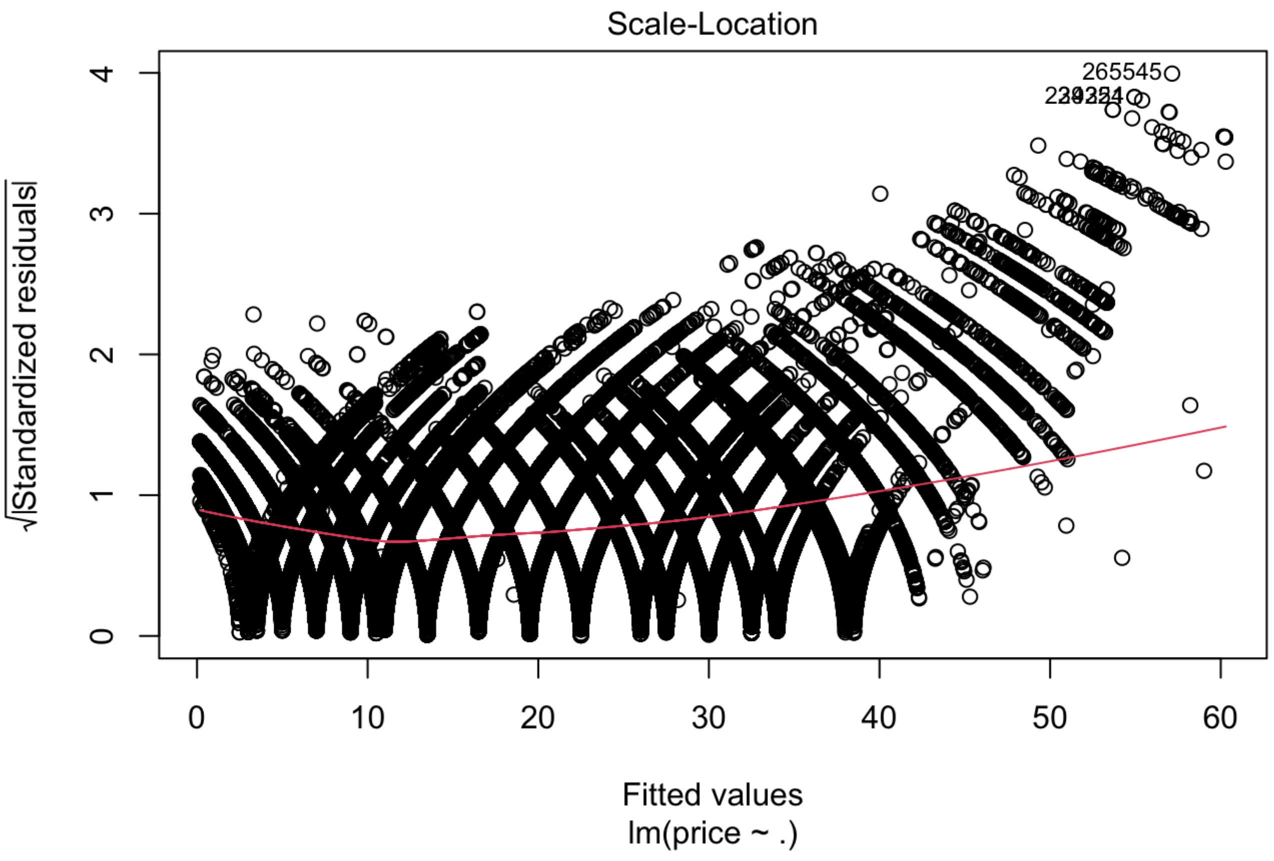
```
#-----------------------------------------------------------------------------------------
----------------------------------------
#-----------------------------------------------------------------------------------------
----------------------------------------

#lyft
lyft_lm_model = lm(price ~., data = lyft_train)
summary(lyft_lm_model)
```

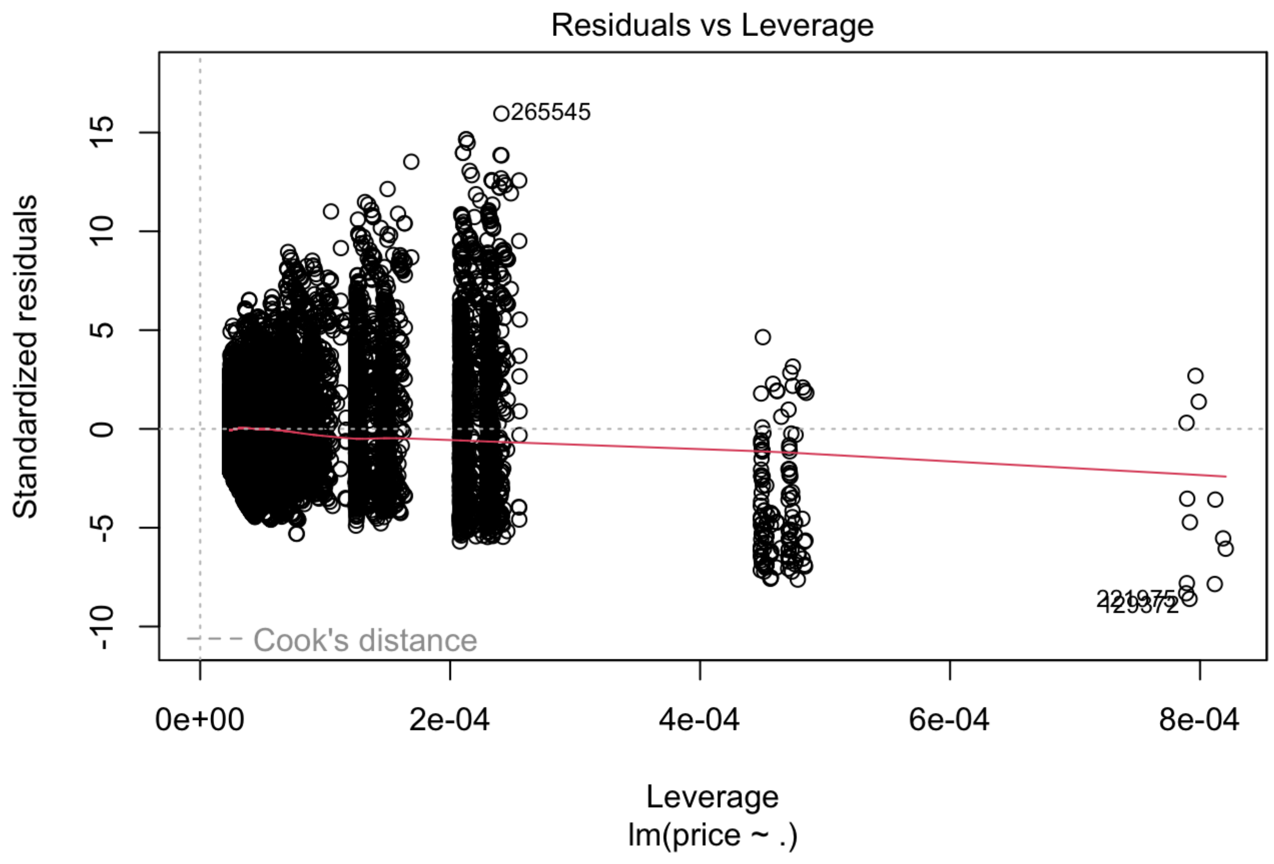
```
##
## Call:
## lm(formula = price ~ ., data = lyft_train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -21.744  -1.417  -0.165   1.422  40.354
##
## Coefficients: (8 not defined because of singularities)
##                   Estimate Std. Error  t value Pr(>|t|)
## (Intercept)      -1.640e+01  3.988e-02 -411.217   <2e-16 ***
## distance          3.243e+00  4.428e-03  732.461   <2e-16 ***
## surge_multiplier  1.822e+01  3.568e-02  510.849   <2e-16 ***
## Fri              -1.054e-02  1.467e-02   -0.719    0.472
## Sat               1.304e-02  1.465e-02    0.890    0.374
## Sun               7.566e-04  1.461e-02    0.052    0.959
## Shared           -2.895e+00  1.670e-02 -173.332   <2e-16 ***
## Lyft_XL           5.696e+00  1.666e-02  341.886   <2e-16 ***
## Lux_Black_XL      2.273e+01  1.666e-02 1363.977   <2e-16 ***
## Lux               8.171e+00  1.665e-02  490.736   <2e-16 ***
## Lux_Black         1.347e+01  1.665e-02  808.881   <2e-16 ***
## Mostly_Cloudy            NA         NA       NA       NA
## Rain                     NA         NA       NA       NA
## Partly_Cloudy            NA         NA       NA       NA
## Overcast                 NA         NA       NA       NA
## Light_Rain               NA         NA       NA       NA
## Foggy                    NA         NA       NA       NA
## Possible_Drizzle         NA         NA       NA       NA
## Drizzle                  NA         NA       NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.529 on 276656 degrees of freedom
## Multiple R-squared:  0.9363, Adjusted R-squared:  0.9363
## F-statistic: 4.068e+05 on 10 and 276656 DF,  p-value: < 2.2e-16
```

```
plot(lyft_lm_model)
```


Residuals vs Fitted
Residuals
Fitted values
lm(price ~ .)
265545
239224
-20
0
20
40
0
10
20
30
40
50
60


Normal Q-Q
Standardized residuals
Theoretical Quantiles
lm(price ~ .)
265545
249524
-5
0
5
10
15
-4
-2
0
2
4


Scale-Location
√|Standardized residuals|
Fitted values
lm(price ~ .)
265545
230254
0
1
2
3
4
0
10
20
30
40
50
60


Residuals vs Leverage
Standardized residuals
Leverage
lm(price ~ .)
265545
221975
429372
Cook's distance
-10
-5
0
5
10
15
0e+00
2e-04
4e-04
6e-04
8e-04

```
#prediction
lyft_pred = predict(lyft_lm_model, lyft_test[,1:18])
```

```
## Warning in predict.lm(lyft_lm_model, lyft_test[, 1:18]): prediction from a rank-
## deficient fit may be misleading
```

```
#Correlation Matrix
actuals_predicts <- data.frame(cbind(actuals=lyft_test$price, predicteds=lyft_pred))
correlation_accuracy <- cor(actuals_predicts)
correlation_accuracy
```

```
##              actuals predicteds
## actuals    1.0000000  0.9683375
## predicteds 0.9683375  1.0000000
```

```
#Evaluation
mat_lr_lyft<- regr.eval(lyft_test[,19], lyft_pred)#, stats = c('mape','rmse'))
print(mat_lr_lyft)
```

```
##       mae       mse      rmse      mape
## 1.8068009 6.2327136 2.4965403 0.1493054
```

```
errors = abs(lyft_pred - lyft_test$price)
mape = 100 * (errors / lyft_test$price)
lyft_lr_accuracy = 100 - mean(mape)
sprintf("The Accuracy of Linear Regression for Lyft :%f",lyft_lr_accuracy)
```

```
## [1] "The Accuracy of Linear Regression for Lyft :85.069462"
```

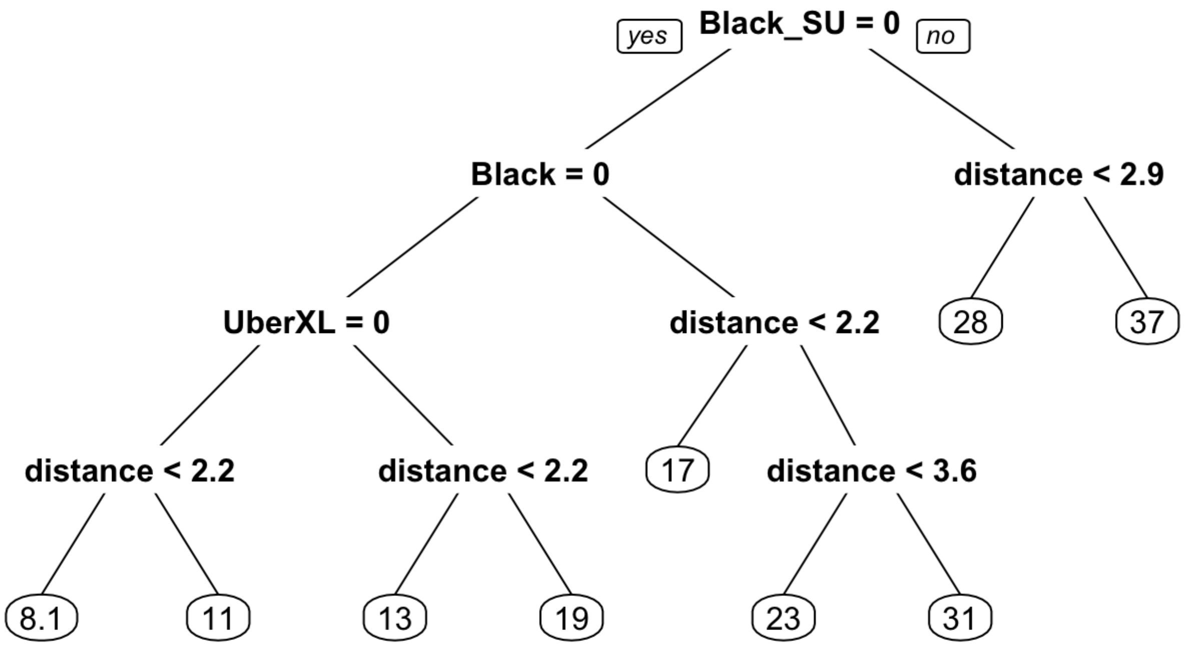
Decision Tree

```
#Uber
uber_rpart_model = rpart(price ~., data = uber_train, method="anova")
summary(uber_rpart_model)
```

```
## Call:
## rpart(formula = price ~ ., data = uber_train, method = "anova")
##   n= 297510
## 
##           CP nsplit  rel error     xerror         xstd
## 1 0.57182252      0 1.00000000 1.00001009 0.0028723813
## 2 0.16623160      1 0.42817748 0.42818237 0.0016622376
## 3 0.06681990      2 0.26194589 0.26195252 0.0012964846
## 4 0.03144997      3 0.19512599 0.19513268 0.0010827101
## 5 0.02763747      4 0.16367602 0.16368297 0.0010058718
## 6 0.01931217      5 0.13603855 0.13604616 0.0008734574
## 7 0.01245846      6 0.11672639 0.11673341 0.0007865145
## 8 0.01026684      7 0.10426792 0.10427501 0.0007714845
## 9 0.01000000      8 0.09400108 0.09525306 0.0007232491
## 
## Variable importance
## Black_SUV     Black  distance    UberXL 
##        63        18        11         7 
## 
## Node number 1: 297510 observations,    complexity param=0.5718225
##   mean=15.7961, MSE=73.33058 
##   left son=2 (248034 obs) right son=3 (49476 obs)
##   Primary splits:
##       Black_SUV < 0.5   to the left,  improve=0.57182250, (0 missing)
##       UberPool  < 0.5   to the right, improve=0.13580220, (0 missing)
##       WAV       < 0.5   to the right, improve=0.09929631, (0 missing)
##       distance  < 2.295 to the left,  improve=0.07445334, (0 missing)
##       Black     < 0.5   to the left,  improve=0.06162716, (0 missing)
## 
## Node number 2: 248034 observations,    complexity param=0.1662316
##   mean=12.90399, MSE=32.96959 
##   left son=4 (198344 obs) right son=5 (49690 obs)
##   Primary splits:
##       Black    < 0.5   to the left,  improve=0.44348150, (0 missing)
##       distance < 2.195 to the left,  improve=0.13931610, (0 missing)
##       UberPool < 0.5   to the right, improve=0.13112140, (0 missing)
##       WAV      < 0.5   to the right, improve=0.07483935, (0 missing)
##       UberXL   < 0.5   to the left,  improve=0.05861073, (0 missing)
## 
## Node number 3: 49476 observations,    complexity param=0.02763747
##   mean=30.29488, MSE=23.52232 
##   left son=6 (37701 obs) right son=7 (11775 obs)
##   Primary splits:
##       distance < 2.865 to the left,  improve=5.18096e-01, (0 missing)
##       Sat      < 0.5   to the left,  improve=5.43687e-05, (0 missing)
##       Fri      < 0.5   to the right, improve=3.19882e-05, (0 missing)
##       Sun      < 0.5   to the left,  improve=6.88000e-06, (0 missing)
## 
## Node number 4: 198344 observations,    complexity param=0.0668199
##   mean=10.99009, MSE=16.76863 
##   left son=8 (148728 obs) right son=9 (49616 obs)
##   Primary splits:
```

```
##       UberXL   < 0.5   to the left,  improve=4.383045e-01, (0 missing)
##       distance < 2.195 to the left,  improve=1.814092e-01, (0 missing)
##       UberPool < 0.5   to the right, improve=9.995433e-02, (0 missing)
##       WAV      < 0.5   to the right, improve=3.002805e-02, (0 missing)
##       Fri      < 0.5   to the right, improve=1.172281e-05, (0 missing)
##
## Node number 5: 49690 observations,    complexity param=0.03144997
##   mean=20.54357, MSE=24.65318
##   left son=10 (25181 obs) right son=11 (24509 obs)
##   Primary splits:
##       distance < 2.195 to the left,  improve=5.600991e-01, (0 missing)
##       Fri      < 0.5   to the right, improve=3.978983e-05, (0 missing)
##       Sat      < 0.5   to the right, improve=1.947026e-05, (0 missing)
##       Sun      < 0.5   to the left,  improve=1.001368e-06, (0 missing)
##
## Node number 6: 37701 observations
##   mean=28.34392, MSE=7.632643
##
## Node number 7: 11775 observations
##   mean=36.54144, MSE=23.19123
##
## Node number 8: 148728 observations,    complexity param=0.01245846
##   mean=9.424234, MSE=5.735292
##   left son=16 (75588 obs) right son=17 (73140 obs)
##   Primary splits:
##       distance < 2.195 to the left,  improve=3.186419e-01, (0 missing)
##       UberPool < 0.5   to the right, improve=3.957829e-02, (0 missing)
##       WAV      < 0.5   to the left,  improve=9.862602e-03, (0 missing)
##       Sat      < 0.5   to the right, improve=4.397628e-05, (0 missing)
##       Sun      < 0.5   to the left,  improve=2.553613e-05, (0 missing)
##   Surrogate splits:
##       Sun < 0.5   to the left,  agree=0.509, adj=0.001, (0 split)
##
## Node number 9: 49616 observations,    complexity param=0.01931217
##   mean=15.68386, MSE=20.46067
##   left son=18 (25185 obs) right son=19 (24431 obs)
##   Primary splits:
##       distance < 2.195 to the left,  improve=4.150268e-01, (0 missing)
##       Fri      < 0.5   to the right, improve=1.219591e-05, (0 missing)
##       Sat      < 0.5   to the left,  improve=7.710355e-06, (0 missing)
##       Sun      < 0.5   to the right, improve=1.881805e-06, (0 missing)
##
## Node number 10: 25181 observations
##   mean=16.87755, MSE=3.94707
##
## Node number 11: 24509 observations,    complexity param=0.01026684
##   mean=24.31011, MSE=17.93197
##   left son=22 (20484 obs) right son=23 (4025 obs)
##   Primary splits:
##       distance < 3.575 to the left,  improve=5.096473e-01, (0 missing)
##       Fri      < 0.5   to the right, improve=2.516569e-05, (0 missing)
##       Sat      < 0.5   to the right, improve=7.776256e-08, (0 missing)
##       Sun      < 0.5   to the left,  improve=7.515796e-09, (0 missing)
##
## Node number 16: 75588 observations
##   mean=8.094453, MSE=2.347904
##
## Node number 17: 73140 observations
##   mean=10.79852, MSE=5.51988
##
## Node number 18: 25185 observations
##   mean=12.81376, MSE=6.920754
##
## Node number 19: 24431 observations
##   mean=18.64254, MSE=17.17293
##
## Node number 22: 20484 observations
##   mean=22.97005, MSE=6.733298
##
## Node number 23: 4025 observations
##   mean=31.12994, MSE=19.27517
```

```
#identify best cp value to use
best <- uber_rpart_model$cptable[which.min(uber_rpart_model$cptable[,"xerror"]),"CP"]

#produce a pruned tree based on the best cp value
pruned_tree <- prune(uber_rpart_model, cp=best)

#plot the pruned tree
prp(pruned_tree)
```



```
#prediction
uber_pred_rpart = predict(uber_rpart_model, uber_test[,-19])

#Correlation Matrix
actuals_predicts <- data.frame(cbind(actuals=uber_test$price, predicteds=uber_pred_rpar
t))
correlation_accuracy <- cor(actuals_predicts)
correlation_accuracy
```

```
##              actuals predicteds
## actuals    1.0000000  0.9520703
## predicteds 0.9520703  1.0000000
```

```
#Evaluation
mat_dt_uber<- regr.eval(uber_test[,19], uber_pred_rpart)#, stats = c('mape','rmse'))
print(mat_dt_uber)
```

```
##       mae       mse      rmse      mape
## 1.7995829 6.8132767 2.6102254 0.1230596
```

```
errors = abs(uber_pred_rpart - uber_test$price)
mape = 100 * (errors / uber_test$price)
uber_dt_accuracy = 100 - mean(mape)
sprintf("The Accuracy of Decision Tree for Uber :%f",uber_dt_accuracy)
```

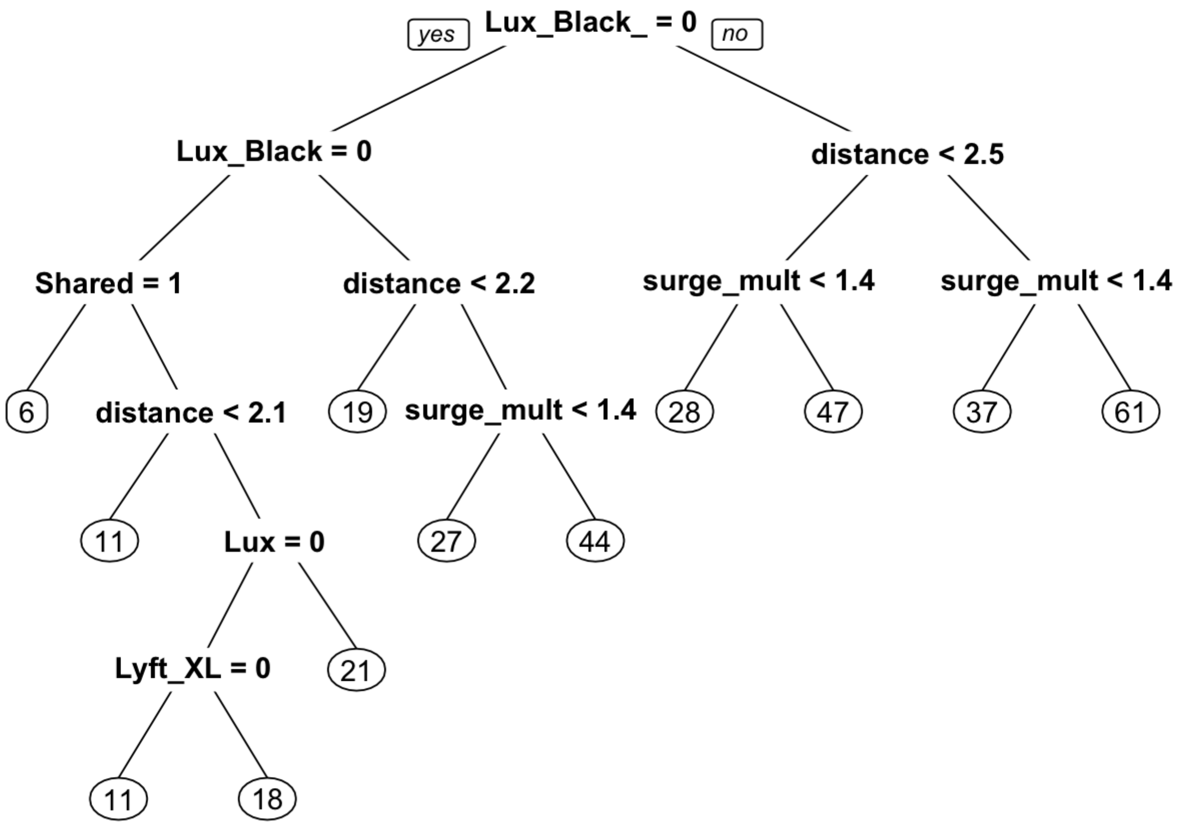
```
## [1] "The Accuracy of Decision Tree for Uber :87.694036"
```

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

# lyft
lyft_rpart_model = rpart(price ~., data = lyft_train, method="anova")
summary(lyft_rpart_model)
```

```
## Call:
## rpart(formula = price ~ ., data = lyft_train, method = "anova")
##   n= 276667
##
##            CP nsplit rel error    xerror         xstd
## 1  0.44585340      0 1.0000000 1.0000058 0.0035430159
## 2  0.15776524      1 0.5541466 0.5541521 0.0023072276
## 3  0.08367959      2 0.3963814 0.3963891 0.0019125673
## 4  0.03891566      3 0.3127018 0.3127098 0.0018211360
## 5  0.03231274      4 0.2737861 0.2738188 0.0017487835
## 6  0.03101511      5 0.2414734 0.2416787 0.0016521927
## 7  0.02459246      6 0.2104583 0.2106961 0.0014428580
## 8  0.02088014      7 0.1858658 0.1860868 0.0014020353
## 9  0.01746894      8 0.1649857 0.1651956 0.0013715584
## 10 0.01156837      9 0.1475167 0.1477901 0.0010403555
## 11 0.01103924     10 0.1359484 0.1353521 0.0008943156
## 12 0.01000000     11 0.1249091 0.1261764 0.0007299040
##
## Variable importance
##     Lux_Black_XL        Lux_Black         distance           Shared
##               51               18               12               10
## surge_multiplier              Lux          Lyft_XL
##                5                3                2
##
## Node number 1: 276667 observations,    complexity param=0.4458534
##   mean=17.34354, MSE=100.4256
##   left son=2 (230640 obs) right son=3 (46027 obs)
##   Primary splits:
##       Lux_Black_XL     < 0.5   to the left,  improve=0.44585340, (0 missing)
##       Shared           < 0.5   to the right, improve=0.25535070, (0 missing)
##       distance         < 2.445 to the left,  improve=0.09106581, (0 missing)
##       surge_multiplier < 1.125 to the left,  improve=0.08077146, (0 missing)
##       Lux_Black        < 0.5   to the left,  improve=0.06536165, (0 missing)
##
## Node number 2: 230640 observations,    complexity param=0.1577652
##   mean=14.35433, MSE=56.38975
##   left son=4 (184461 obs) right son=5 (46179 obs)
##   Primary splits:
##       Lux_Black        < 0.5   to the left,  improve=0.33703760, (0 missing)
##       Shared           < 0.5   to the right, improve=0.30765310, (0 missing)
##       distance         < 2.175 to the left,  improve=0.13826520, (0 missing)
##       surge_multiplier < 1.125 to the left,  improve=0.09972451, (0 missing)
##       Lux              < 0.5   to the left,  improve=0.05171951, (0 missing)
##
## Node number 3: 46027 observations,    complexity param=0.03101511
##   mean=32.32242, MSE=51.94622
##   left son=6 (28747 obs) right son=7 (17280 obs)
##   Primary splits:
##       distance         < 2.495 to the left,  improve=3.604198e-01, (0 missing)
##       surge_multiplier < 1.375 to the left,  improve=3.554192e-01, (0 missing)
##       Fri              < 0.5   to the right, improve=4.740028e-05, (0 missing)
##       Sat              < 0.5   to the left,  improve=9.382979e-06, (0 missing)
##       Sun              < 0.5   to the left,  improve=4.275176e-07, (0 missing)
## 
## Node number 4: 184461 observations,    complexity param=0.08367959
##   mean=12.17306, MSE=36.22639 
##   left son=8 (46168 obs) right son=9 (138293 obs)
##   Primary splits:
##       Shared           < 0.5   to the right, improve=0.34792990, (0 missing)
##       Lux              < 0.5   to the left,  improve=0.28836720, (0 missing)
##       distance         < 1.955 to the left,  improve=0.15749800, (0 missing)
##       surge_multiplier < 1.125 to the left,  improve=0.11212680, (0 missing)
##       Lyft_XL          < 0.5   to the left,  improve=0.08942717, (0 missing)
## 
## Node number 5: 46179 observations,    complexity param=0.03231274
##   mean=23.06736, MSE=42.00945 
##   left son=10 (23531 obs) right son=11 (22648 obs)
##   Primary splits:
##       distance         < 2.175 to the left,  improve=4.627902e-01, (0 missing)
##       surge_multiplier < 1.375 to the left,  improve=2.271646e-01, (0 missing)
##       Sun              < 0.5   to the left,  improve=3.186539e-05, (0 missing)
##       Fri              < 0.5   to the left,  improve=9.812630e-06, (0 missing)
##       Sat              < 0.5   to the right, improve=1.424171e-06, (0 missing)
##   Surrogate splits:
##       surge_multiplier < 1.125 to the left,  agree=0.521, adj=0.023, (0 split)
##       Sun              < 0.5   to the left,  agree=0.510, adj=0.000, (0 split)
## 
## Node number 6: 28747 observations,    complexity param=0.01156837
##   mean=28.9677, MSE=19.8281 
##   left son=12 (27781 obs) right son=13 (966 obs)
##   Primary splits:
##       surge_multiplier < 1.375 to the left,  improve=5.638976e-01, (0 missing)
##       distance         < 1.955 to the left,  improve=1.489673e-01, (0 missing)
##       Fri              < 0.5   to the right, improve=3.106035e-04, (0 missing)
##       Sun              < 0.5   to the right, improve=2.727286e-06, (0 missing)
##       Sat              < 0.5   to the right, improve=7.353013e-07, (0 missing)
## 
## Node number 7: 17280 observations,    complexity param=0.01746894
##   mean=37.90334, MSE=55.50882 
##   left son=14 (16450 obs) right son=15 (830 obs)
##   Primary splits:
##       surge_multiplier < 1.375 to the left,  improve=5.060141e-01, (0 missing)
##       distance         < 3.555 to the left,  improve=2.237248e-01, (0 missing)
##       Sun              < 0.5   to the right, improve=5.680013e-05, (0 missing)
##       Sat              < 0.5   to the right, improve=7.306841e-06, (0 missing)
##       Fri              < 0.5   to the left,  improve=6.602096e-06, (0 missing)
## 
## Node number 8: 46168 observations
##   mean=6.028537, MSE=4.441998 
## 
## Node number 9: 138293 observations,    complexity param=0.03891566
##   mean=14.22436, MSE=30.02528 
##   left son=18 (68334 obs) right son=19 (69959 obs)
##   Primary splits:
```

```
##       distance         < 2.125 to the left,  improve=2.603988e-01, (0 missing)
##       Lux              < 0.5   to the left,  improve=2.095525e-01, (0 missing)
##       surge_multiplier < 1.375 to the left,  improve=1.093318e-01, (0 missing)
##       Lyft_XL          < 0.5   to the left,  improve=1.901981e-02, (0 missing)
##       Sat              < 0.5   to the right, improve=9.215372e-06, (0 missing)
##   Surrogate splits:
##       surge_multiplier < 1.125 to the left,  agree=0.507, adj=0.002, (0 split)
##
## Node number 10: 23531 observations
##   mean=18.74162, MSE=10.94892
##
## Node number 11: 22648 observations,    complexity param=0.01103924
##   mean=27.56175, MSE=34.63985
##   left son=22 (21586 obs) right son=23 (1062 obs)
##   Primary splits:
##       surge_multiplier < 1.375 to the left,  improve=3.909626e-01, (0 missing)
##       distance         < 3.545 to the left,  improve=3.072661e-01, (0 missing)
##       Fri              < 0.5   to the left,  improve=5.558182e-05, (0 missing)
##       Sat              < 0.5   to the right, improve=1.157420e-05, (0 missing)
##       Sun              < 0.5   to the left,  improve=9.455698e-08, (0 missing)
##
## Node number 12: 27781 observations
##   mean=28.34417, MSE=7.205983
##
## Node number 13: 966 observations
##   mean=46.89959, MSE=50.09137
##
## Node number 14: 16450 observations
##   mean=36.71287, MSE=23.00623
##
## Node number 15: 830 observations
##   mean=61.49759, MSE=114.9096
##
## Node number 18: 68334 observations
##   mean=11.39514, MSE=11.97631
##
## Node number 19: 69959 observations,    complexity param=0.02459246
##   mean=16.98786, MSE=32.19954
##   left son=38 (46591 obs) right son=39 (23368 obs)
##   Primary splits:
##       Lux              < 0.5   to the left,  improve=0.3033266000, (0 missing)
##       surge_multiplier < 1.375 to the left,  improve=0.1417080000, (0 missing)
##       distance         < 3.545 to the left,  improve=0.1207882000, (0 missing)
##       Lyft_XL          < 0.5   to the left,  improve=0.0269986000, (0 missing)
##       Sat              < 0.5   to the right, improve=0.0000834332, (0 missing)
##   Surrogate splits:
##       Lyft_XL < 0.5   to the right, agree=0.666, adj=0.001, (0 split)
##
## Node number 22: 21586 observations
##   mean=26.74548, MSE=17.96197
##
## Node number 23: 1062 observations
##   mean=44.15301, MSE=84.81816
##
## Node number 38: 46591 observations,    complexity param=0.02088014
##   mean=14.77456, MSE=23.21915
##   left son=76 (23334 obs) right son=77 (23257 obs)
##   Primary splits:
##       Lyft_XL          < 0.5   to the left,  improve=5.362742e-01, (0 missing)
##       surge_multiplier < 1.375 to the left,  improve=1.488868e-01, (0 missing)
##       distance         < 3.545 to the left,  improve=1.257325e-01, (0 missing)
##       Sat              < 0.5   to the right, improve=8.349234e-05, (0 missing)
##       Fri              < 0.5   to the right, improve=5.437046e-06, (0 missing)
##   Surrogate splits:
##       distance < 3.205 to the left,  agree=0.502, adj=0.002, (0 split)
##       Sat      < 0.5   to the right, agree=0.502, adj=0.002, (0 split)
##
## Node number 39: 23368 observations
##   mean=21.40072, MSE=20.86427
##
## Node number 76: 23334 observations
##   mean=11.25167, MSE=5.12571
##
## Node number 77: 23257 observations
##   mean=18.30911, MSE=16.4276
```

```
#identify best cp value to use
best <- lyft_rpart_model$cptable[which.min(lyft_rpart_model$cptable[,"xerror"]),"CP"]
#produce a pruned tree based on the best cp value
pruned_tree <- prune(lyft_rpart_model, cp=best)

#plot the pruned tree
prp(pruned_tree)
```



```
#prediction
lyft_pred_rpart = predict(lyft_rpart_model, lyft_test[,-19])

#Correlation Matrix
actuals_predicts <- data.frame(cbind(actuals=lyft_test$price, predicteds=lyft_pred_rpar
t))
correlation_accuracy <- cor(actuals_predicts)
correlation_accuracy
```

```
##              actuals predicteds
## actuals    1.0000000  0.9348973
## predicteds 0.9348973  1.0000000
```

```
#Evaluation
mat_dt_lyft<- regr.eval(lyft_test[,19], lyft_pred_rpart)#, stats = c('mape','rmse'))
print(mat_dt_lyft)
```

```
##       mae        mse       rmse       mape
##  2.6463307 12.5978293  3.5493421  0.1969981
```

```
errors = abs(lyft_pred_rpart - lyft_test$price)
mape = 100 * (errors / lyft_test$price)
lyft_dt_accuracy = 100 - mean(mape)
sprintf("The Accuracy of Decision Tree for Lyft :%f",lyft_dt_accuracy)
```

```
## [1] "The Accuracy of Decision Tree for Lyft :80.300189"
```

Random Forest

```
#Uber
#head(uber_train)
uber_rmforest_model = randomForest(price ~., data = uber_train, importance = TRUE, ntree
= 100)
summary(uber_rmforest_model)
```

```
##                 Length Class  Mode
## call                 5 -none- call
## type                 1 -none- character
## predicted       297510 -none- numeric
## mse                100 -none- numeric
## rsq                100 -none- numeric
## oob.times       297510 -none- numeric
## importance          36 -none- numeric
## importanceSD        18 -none- numeric
## localImportance      0 -none- NULL
## proximity            0 -none- NULL
## ntree                1 -none- numeric
## mtry                 1 -none- numeric
## forest              11 -none- list
## coefs                0 -none- NULL
## y               297510 -none- numeric
## test                 0 -none- NULL
## inbag                0 -none- NULL
## terms                3 terms  call
```

```
#prediction
uber_pred_rmforest = predict(uber_rmforest_model, uber_test[,-19])

#Correlation Matrix
actuals_predicts <- data.frame(cbind(actuals=uber_test$price, predicteds=uber_pred_rmfor
est))
correlation_accuracy <- cor(actuals_predicts)
correlation_accuracy
```

```
##              actuals predicteds
## actuals    1.0000000  0.9576262
## predicteds 0.9576262  1.0000000
```

```
#Evaluation
mat_rf_uber<- regr.eval(uber_test[,19], uber_pred_rmforest)#, stats = c('mape','rmse'))
print(mat_rf_uber)
```

```
##       mae       mse      rmse      mape
## 1.7504893 6.6244389 2.5737985 0.1288409
```

```
errors = abs(uber_pred_rmforest - uber_test$price)
mape = 100 * (errors / uber_test$price)
uber_rf_accuracy = 100 - mean(mape)
sprintf("The Accuracy of Random Forest for Uber :%f",uber_rf_accuracy)
```

```
## [1] "The Accuracy of Random Forest for Uber :87.115908"
```

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



lyft_rmforest_model = randomForest(price ~., data = lyft_train, importance = TRUE, ntree
= 100)
summary(lyft_rmforest_model)
```

```
##                 Length Class  Mode
## call                 5 -none- call
## type                 1 -none- character
## predicted       276667 -none- numeric
## mse                100 -none- numeric
## rsq                100 -none- numeric
## oob.times       276667 -none- numeric
## importance          36 -none- numeric
## importanceSD        18 -none- numeric
## localImportance      0 -none- NULL
## proximity            0 -none- NULL
## ntree                1 -none- numeric
## mtry                 1 -none- numeric
## forest              11 -none- list
## coefs                0 -none- NULL
## y               276667 -none- numeric
## test                 0 -none- NULL
## inbag                0 -none- NULL
## terms                3 terms  call
```

```
#prediction
lyft_pred_rmforest = predict(lyft_rmforest_model, lyft_test[,-19])

#Correlation Matrix
actuals_predicts <- data.frame(cbind(actuals=lyft_test$price, predicteds=lyft_pred_rmfor
est))
correlation_accuracy <- cor(actuals_predicts)
correlation_accuracy
```

```
##              actuals predicteds
## actuals    1.0000000  0.9766981
## predicteds 0.9766981  1.0000000
```

```
#Evaluation
mat_rf_lyft<- regr.eval(lyft_test[,19], lyft_pred_rmforest)#, stats = c('mape','rmse'))
print(mat_rf_lyft)
```

```
##       mae       mse      rmse      mape
## 1.7126200 5.4800038 2.3409408 0.1326613
```

```
errors = abs(lyft_pred_rmforest - lyft_test$price)
mape = 100 * (errors / lyft_test$price)
lyft_rf_accuracy = 100 - mean(mape)
sprintf("The Accuracy of Random Forest for Lyft :%f",lyft_rf_accuracy)
```

```
## [1] "The Accuracy of Random Forest for Lyft :86.733867"
```

Model Evaluation

```
# Uber
print("***************Uber Statitics*************")
```

```
## [1] "***************Uber Statitics*************"
```

```
tab <- matrix(c(mat_lr_uber["mae"],mat_dt_uber["mae"],mat_rf_uber["mae"],mat_lr_uber["ms
e"],mat_dt_uber["mse"],mat_rf_uber["mse"],mat_lr_uber["rmse"],mat_dt_uber["rmse"],mat_rf
_uber["rmse"],mat_lr_uber["mape"],mat_dt_uber["mape"],mat_rf_uber["mape"], uber_lr_accur
acy,uber_dt_accuracy,uber_rf_accuracy), ncol=3, byrow=TRUE)
colnames(tab) <- c("Linear Regression",'Decision Tree','Random Forest')
rownames(tab) <- c('MAE','MSE','RMSE','MAPE',"Accuracy")
uber_tab <- as.table(tab)
uber_tab
```

```
##          Linear Regression Decision Tree Random Forest
## MAE              1.6697108     1.7995829     1.7504893
## MSE              5.8347082     6.8132767     6.6244389
## RMSE             2.4155141     2.6102254     2.5737985
## MAPE             0.1191017     0.1230596     0.1288409
## Accuracy        88.0898340    87.6940363    87.1159081
```

```
print("***************lyft Statitics*************")
```

```
## [1] "***************lyft Statitics*************"
```

```
# Lyft
tab <- matrix(c(mat_lr_lyft["mae"],mat_dt_lyft["mae"],mat_rf_lyft["mae"],mat_lr_lyft["ms
e"],mat_dt_lyft["mse"],mat_rf_lyft["mse"],mat_lr_lyft["rmse"],mat_dt_lyft["rmse"],mat_rf
_lyft["rmse"],mat_lr_lyft["mape"],mat_dt_lyft["mape"],mat_rf_lyft["mape"], lyft_lr_accur
acy,lyft_dt_accuracy,lyft_rf_accuracy), ncol=3, byrow=TRUE)
colnames(tab) <- c("Linear Regression",'Decision Tree','Random Forest')
rownames(tab) <- c('MAE','MSE','RMSE','MAPE',"Accuracy")
lyft_tab <- as.table(tab)
lyft_tab
```

```
##          Linear Regression Decision Tree Random Forest
## MAE              1.8068009     2.6463307     1.7126200
## MSE              6.2327136    12.5978293     5.4800038
## RMSE             2.4965403     3.5493421     2.3409408
## MAPE             0.1493054     0.1969981     0.1326613
## Accuracy        85.0694620    80.3001895    86.7338674
```

Based on different metrics, we can conclude that the Random Forest model is the best model and thus we could consider this as our final model.