Capital One Data Challenge

Pallav Anand
Texas A&M University

I have used R language and Jupyter notebook, R studio platforms for analysis.

Question 1
Programmatically download and load into your favorite analytical tool the trip data for September 2015

Downloading the data and importing in R
data=read.csv('https://s3.amazonaws.com/nyc-tlc/trip+data/green_tripdata_2015-09.csv')
summary(data)

Checking the rows and columns

```
> ncol(data)
[1] 21
> nrow(data)
[1] 1494926
>
```

The dataset has 1494926 rows and 21 columns

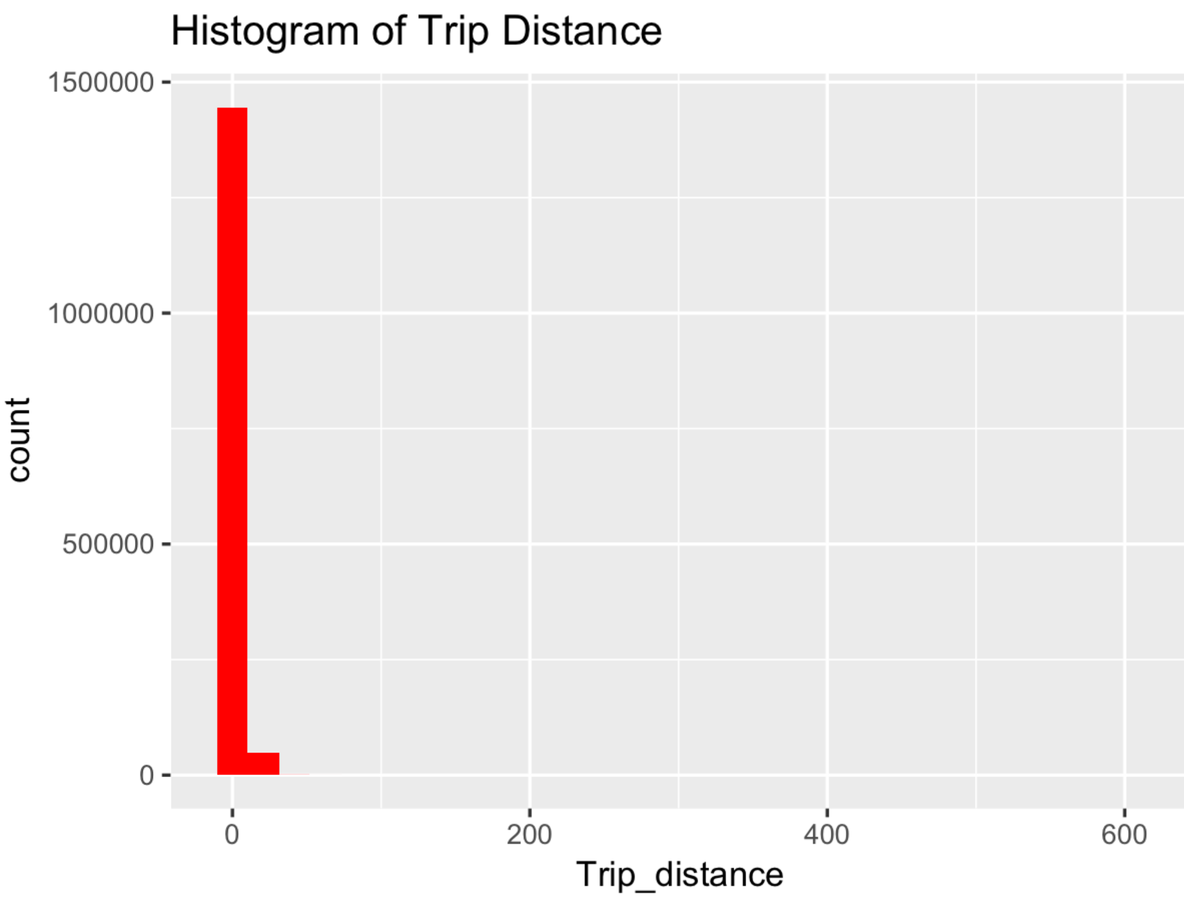
We can see different column names by colnames and get a summary of each variable by summary function.
We can use str function to get an idea about the different variable data types.

```
colnames(data)
1] "VendorID"              "lpep_pickup_datetime"  "Lpep_dropoff_datetime" "Store_and_fwd_flag"    "RateCodeID"            "Pickup_longitude"      "Pickup_latit
8] "Dropoff_longitude"     "Dropoff_latitude"      "Passenger_count"       "Trip_distance"         "Fare_amount"           "Extra"                 "MTA_tax"
5] "Tip_amount"            "Tolls_amount"          "Ehail_fee"             "improvement_surcharge" "Total_amount"          "Payment_type"          "Trip_type"
summary(data)
   VendorID          lpep_pickup_datetime        Lpep_dropoff_datetime Store_and_fwd_flag   RateCodeID      Pickup_longitude Pickup_latitude Dropoff_longitud
lin.   :1.000   2015-09-20 02:00:32:      9   2015-09-28 00:00:00:    172   N:1486192          Min.   : 1.000   Min.   :-83.32   Min.   : 0.00   Min.   :-83.43
st Qu.:2.000   2015-09-05 14:57:48:      8   2015-09-13 00:00:00:    153   Y:   8734          1st Qu.: 1.000   1st Qu.:-73.96   1st Qu.:40.70   1st Qu.:-73.97
ledian :2.000   2015-09-10 17:43:49:      8   2015-09-19 00:00:00:    141                      Median : 1.000   Median :-73.95   Median :40.75   Median :-73.95
lean   :1.782   2015-09-13 00:27:28:      8   2015-09-14 00:00:00:    126                      Mean   : 1.098   Mean   :-73.83   Mean   :40.69   Mean   :-73.84
rd Qu.:2.000   2015-09-13 01:06:29:      8   2015-09-21 00:00:00:    125                      3rd Qu.: 1.000   3rd Qu.:-73.92   3rd Qu.:40.80   3rd Qu.:-73.91
lax.   :2.000   2015-09-26 22:48:40:      8   2015-09-12 00:00:00:    119                      Max.   :99.000   Max.   :  0.00   Max.   :43.18   Max.   :  0.00
               (Other)            :1494877   (Other)            :1494090
assenger_count Trip_distance      Fare_amount          Extra             MTA_tax           Tip_amount         Tolls_amount        Ehail_fee        improvement_surcharge
lin.   :0.000   Min.   :  0.000   Min.   :-475.00   Min.   :-1.0000   Min.   :-0.5000   Min.   :-50.000   Min.   :-15.2900   Mode:logical     Min.   :-0.3000
st Qu.:1.000   1st Qu.:  1.100   1st Qu.:    6.50   1st Qu.: 0.0000   1st Qu.: 0.5000   1st Qu.:  0.000   1st Qu.:  0.0000   NA's:1494926     1st Qu.: 0.3000
ledian :1.000   Median :  1.980   Median :    9.50   Median : 0.5000   Median : 0.5000   Median :  0.000   Median :  0.0000                    Median : 0.3000
lean   :1.371   Mean   :  2.968   Mean   :   12.54   Mean   : 0.3513   Mean   : 0.4866   Mean   :  1.236   Mean   :  0.1231                    Mean   : 0.2921
rd Qu.:1.000   3rd Qu.:  3.740   3rd Qu.:   15.50   3rd Qu.: 0.5000   3rd Qu.: 0.5000   3rd Qu.:  2.000   3rd Qu.:  0.0000                    3rd Qu.: 0.3000
lax.   :9.000   Max.   :603.100   Max.   : 580.50   Max.   :12.0000   Max.   : 0.5000   Max.   :300.000   Max.   : 95.7500                    Max.   : 0.3000

 Payment_type     Trip_type
lin.   :1.000   Min.   :1.000
st Qu.:1.000   1st Qu.:1.000
ledian :2.000   Median :1.000
lean   :1.541   Mean   :1.022
rd Qu.:2.000   3rd Qu.:1.000
lax.   :5.000   Max.   :2.000
               NA's   :4
```

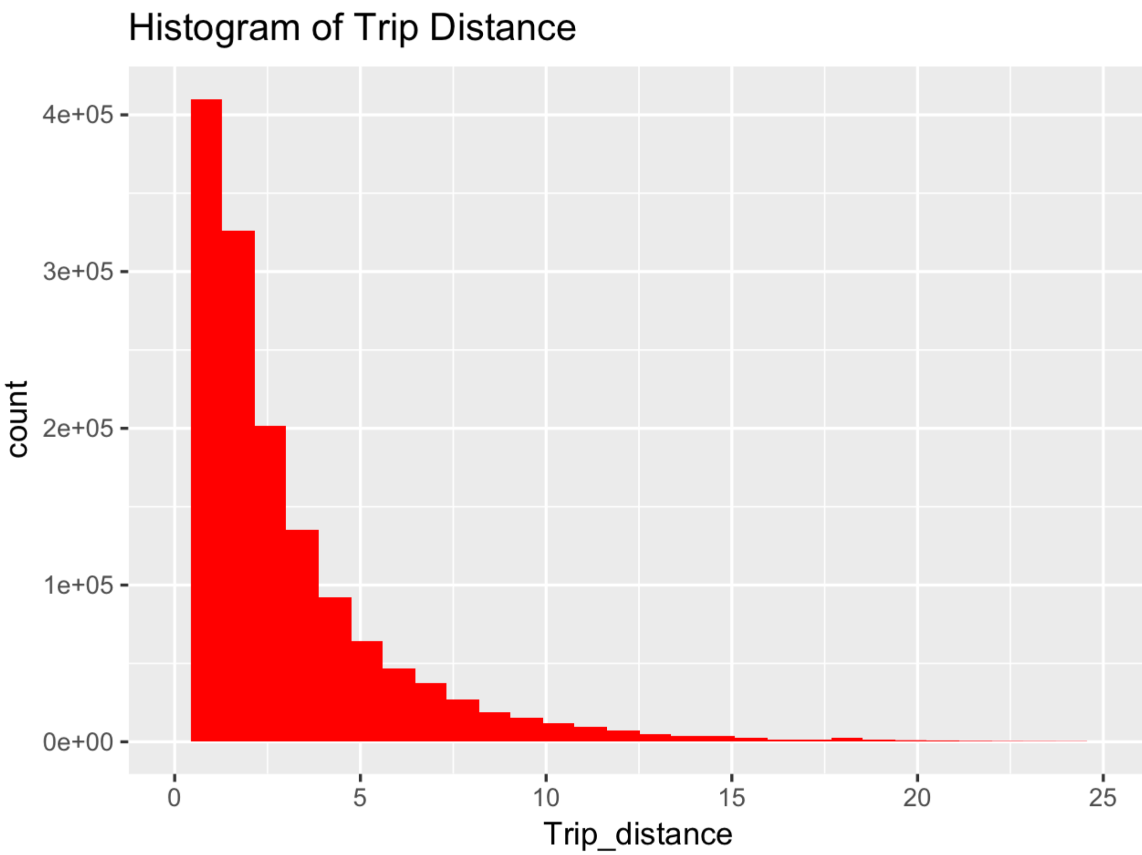
Summary reveals that the data is erroneous because of Fare_amount of -475 and other such issues

Question 2
Plot a histogram of the number of the trip distance ("Trip Distance")
I installed visualization package ggplot2 using
install.packages(“ggplot”)

library(ggplot2)
ggplot(data=data,aes(x=Trip_distance))+geom_histogram(bins=30,fill="red")+ggtitle("Histogram of Trip Distance")



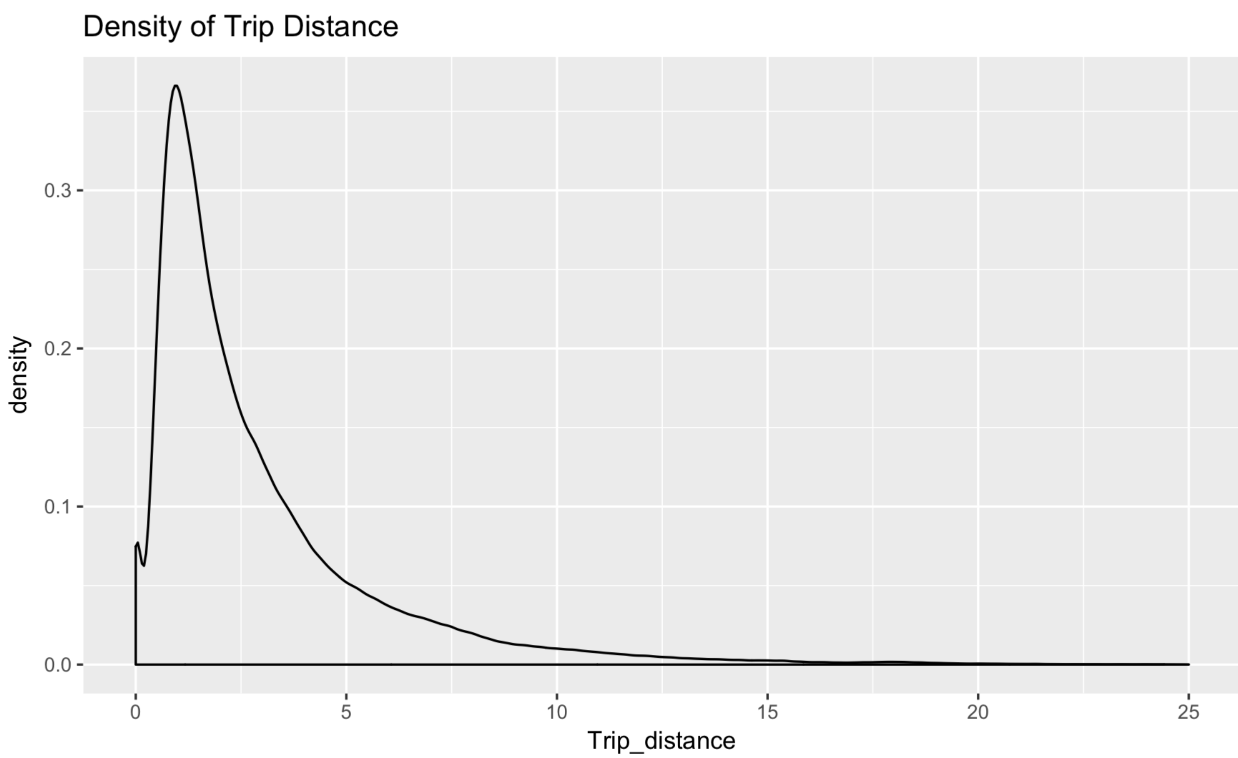
ggplot(data=data,aes(x=Trip_distance))+geom_histogram(bins=30,fill="red")+ggtitle("Histogram of Trip Distance")+xlim(0,25)



Report any structure you find and any hypotheses you have about that structure.

There are many outliers in trip distances. These distances are too large to be feasible and can be erroneous data. I checked for some and they seem too large for the small travel times and fares.

A lot of the trip distances are between 0 and 25. The distribution is not normally distributed and is heavily skewed to the right.



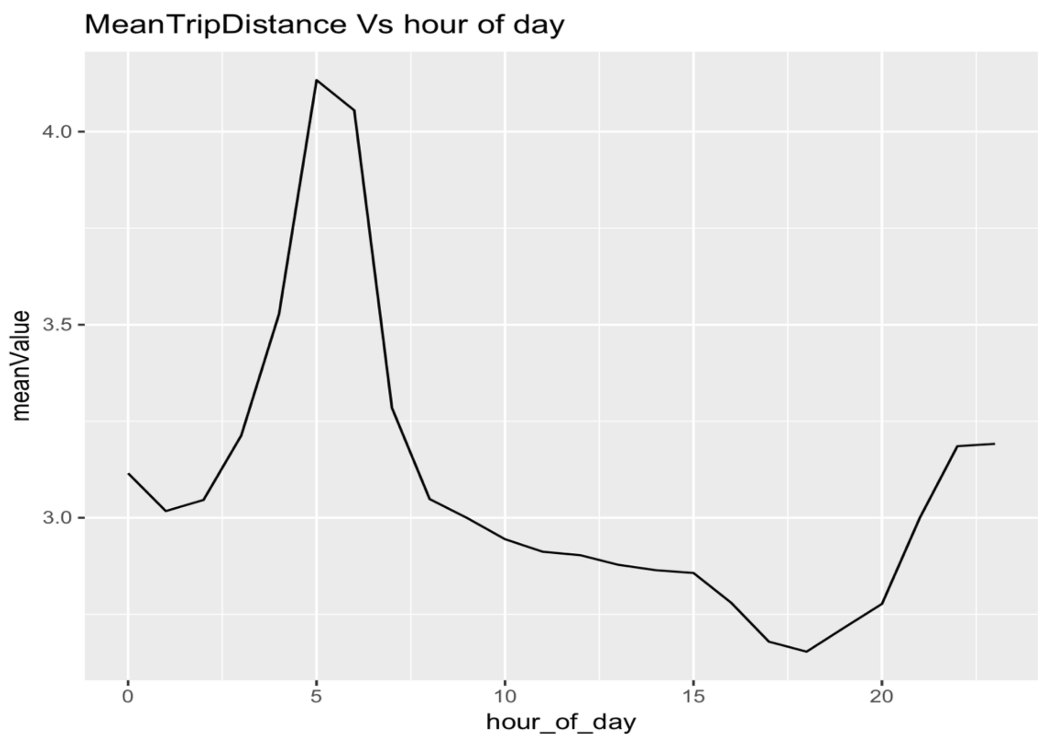
Question 3
Report mean and median trip distance grouped by hour of day.

I used hour function from lubridate package and dplyr package for grouping and aggregation.

install.packages("lubridate")
library(lubridate)

```
data %>% mutate(hour_of_day = hour(as.POSIXct(lpep_pickup_datetime,"%Y-%m-%d %h:%m:%s"))) %>%
group_by(hour_of_day) %>% summarise(meanValue = mean(Trip_distance))
```

hour_of_day	meanValue
0	3.115276
1	3.017347
2	3.046176
3	3.212945
4	3.526555
5	4.133474
6	4.055149
7	3.284394
8	3.048450
9	2.999105
10	2.944482
11	2.912015
12	2.903065
13	2.878294
14	2.864304
15	2.857040
16	2.779852
17	2.679114
18	2.653222
19	2.715597
20	2.777052
21	2.999189
22	3.185394
23	3.191538

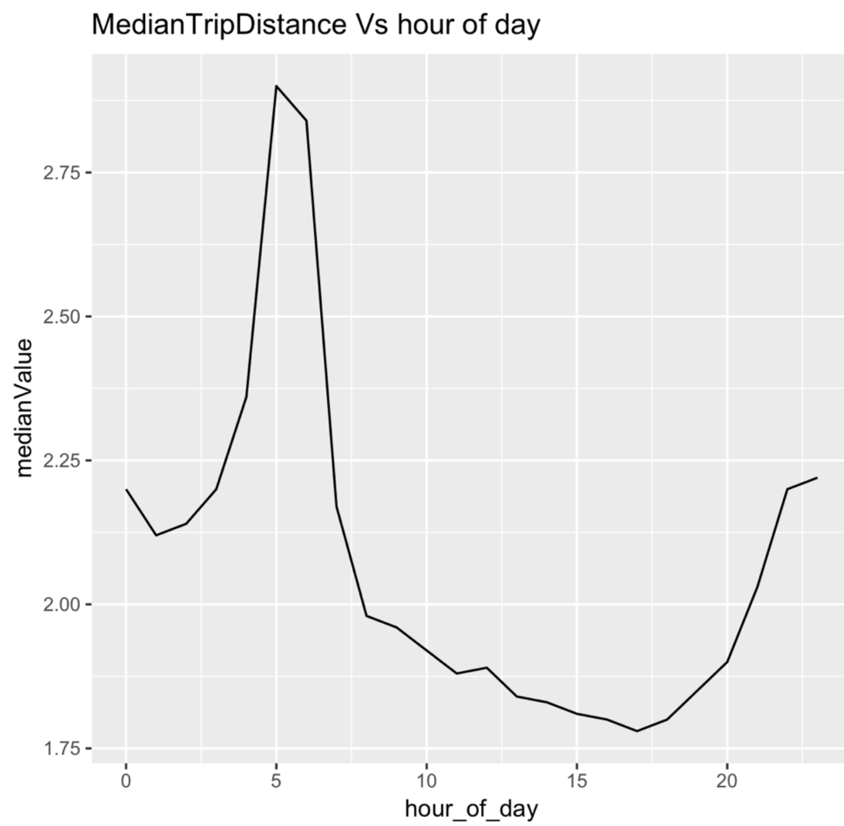


Comments
It seems that people are travelling longer distances during late night and early morning hours.

Median values are also calculated

```
data %>% mutate(hour_of_day = hour(as.POSIXct(lpep_pickup_datetime,"%Y-%m-%d %h:%m:%s"))) %>%
group_by(hour_of_day) %>% summarise(medianValue = median(Trip_distance))
```

hour_of_day	medianValue
0	2.20
1	2.12
2	2.14
3	2.20
4	2.36
5	2.90
6	2.84
7	2.17
8	1.98
9	1.96
10	1.92
11	1.88
12	1.89
13	1.84
14	1.83
15	1.81
16	1.80
17	1.78
18	1.80
19	1.85
20	1.90
21	2.03
22	2.20
23	2.22



We would like to get a rough sense of identifying trips that originate or terminate at one of the NYC area airports. Can you provide a count of how many transactions fit this criteria, the average fair, and any other interesting characteristics of these trips

There are three major airports in the NYC area:
LaGuardia Airport **40.7769, 73.8740**
Newark Liberty International Airport **40.6895, 74.1745**
John F. Kennedy International Airport **40.6413, 73.7781**

(Data obtained from Wikipedia)

I will analyse the pickups and drop-offs for JFK airport.
The issue here is that a small change in decimal of airport longitude and lattitude can mean a large change in the distance. But airports are quite large in size and have multiple terminals so we can consider +/- 0.1 lattitude and longitude differences to be close to the airport.

JFKpick=data[data$Pickup_latitude > 40.6 & data$Pickup_latitude < 40.7 & data$Pickup_longitude < -73.7 & data$Pickup_longitude > -73.8,]

JFKdrop=data[data$Dropoff_latitude > 40.6 & data$Dropoff_latitude < 40.7 & data$Dropoff_longitude < -73.7 & data$Dropoff_longitude > -73.8,]

We observe that a there are **more number of airport drop-offs than pickups**. It is possible that it is difficult to get a cab at a populated airport like JFK and people ask some of their friends/relatives to pick them up.

```
dim(JFKpick)
```

1082 25

```
dim(JFKdrop)
```

19921 25

```
JFKall=rbind(JFKpick,JFKdrop)
```

```
dim(JFKall)
```

21003 25

```
mean(JFKall$Total_amount)
```

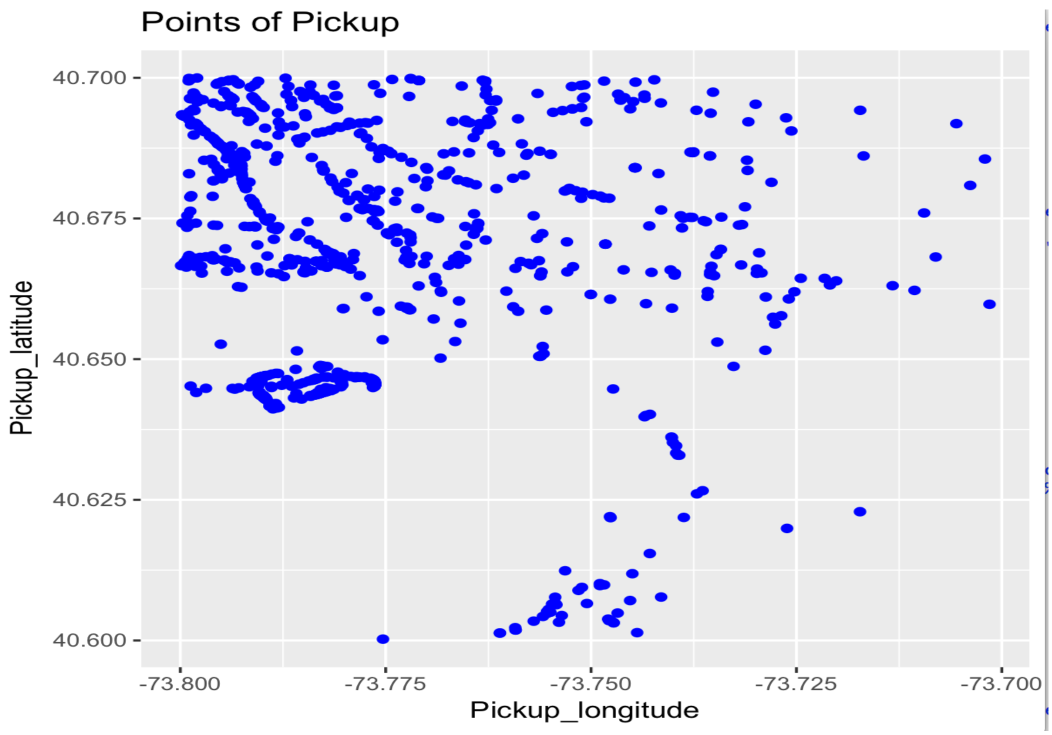
38.3313874208446

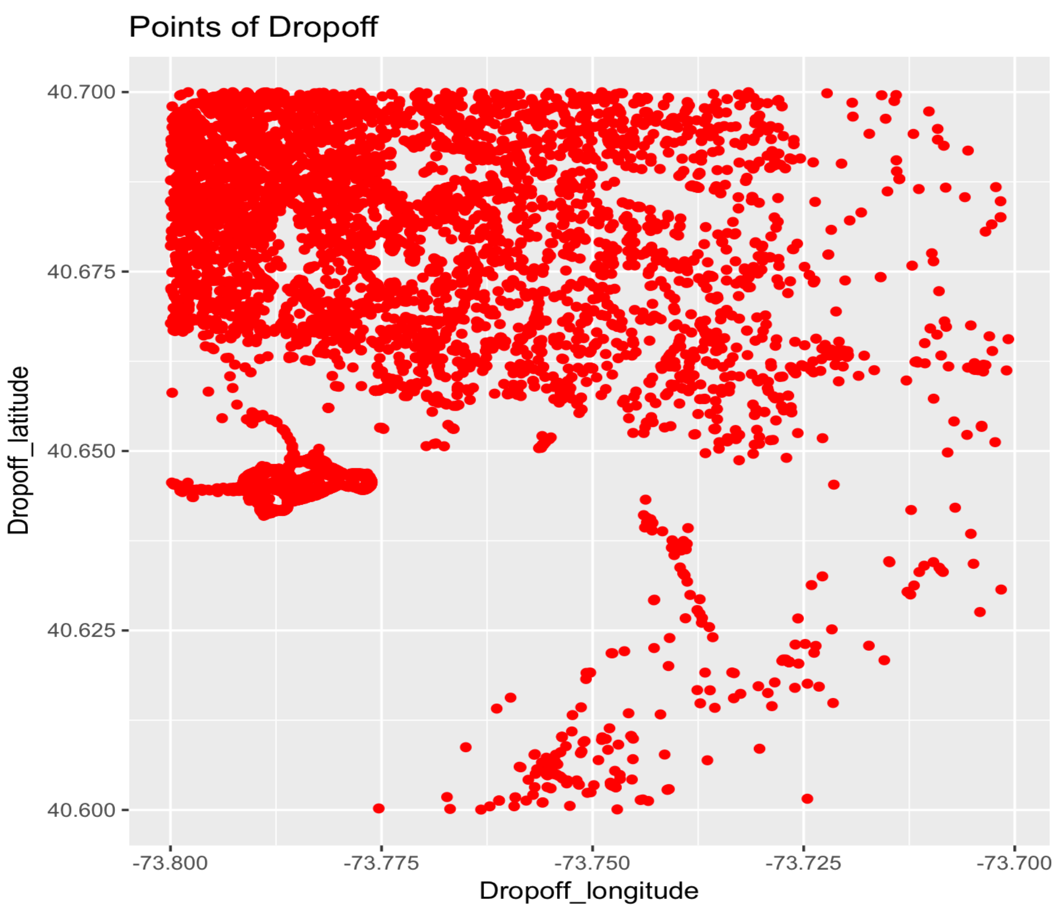
There are around 21000 trips near JFK airport and mean fare of trips near JFK is around 38 dollars.

We can visualize pickup and dropoff points in a scatterplot

ggplot(JFKpick, aes(x=Pickup_longitude, y=Pickup_*latitude)) + geom_point(color='blue')+ggtitle("Points of Pickup")*

ggplot(JFKdrop, aes(x=Dropoff_longitude, y=Dropoff_latitude)) + geom_point(color='red')+ggtitle("Points of Dropoff")


Points of Pickup
40.700
40.675
40.650
40.625
40.600
Pickup_latitude
-73.800
-73.775
-73.750
-73.725
-73.700
Pickup_longitude


Points of Dropoff
40.700
40.675
40.650
40.625
40.600
Dropoff_latitude
-73.800
-73.775
-73.750
-73.725
-73.700
Dropoff_longitude

Question 4
Part A) Build a derived variable for tip as a percentage of the total fare.

We calculate the variable tipPercent as the ratio of Tip_amount and Fare_amount and then calculate the percentage.
*data$tipPercent=data$Tip_amount/data$Fare_amount*100*

```
head(data[,c('Tip_amount','Fare_amount','tipPercent')])
```

Tip_amount	Fare_amount	tipPercent
1.95	7.8	25.00000
0.00	45.0	0.00000
0.50	4.0	12.50000
0.00	5.0	0.00000
0.00	5.0	0.00000
1.36	5.5	24.72727

Part B) Build a predictive model for tip as a percentage of the total fare. Use as much of the data as you like (or all of it). We will validate a sample.

Steps used in modeling

1. Cleaning of data set
2. Partition the data into train and test sets
3. Creation of new variables
4. Transformation of existing variables
5. Implementing various models

1.Cleaning

Many data points seem to be erroneous having zero trip distance and time and negative total fares. I have shown one such example.

ge	Total_amount	Payment_type	Trip_type	tipPercent
	-3.8	3	1	0
	-5.8	4	1	0
	-4.3	3	1	0
	-52.8	4	1	0
	-3.8	3	1	0
	-3.8	3	1	0
	-3.3	4	1	0
	-5.3	3	1	0
	-3.3	3	1	0
	-4.8	3	1	0

I check for NAs in the data frame and removed those which are not feasible.

datacleaned=data[data$Trip_distance>0 & data$Fare_amount>0 & data$Total_amount>0 & data$Triptime>0 & data$Tip_amount>=0,]

Cleaned the dataset of all infeasible values

2.Partioning the data into train and test sets

```
index=sample(1:nrow(datacleaned), size=0.8*nrow(datacleaned))
traindata=datacleaned[index,]
testdata=datacleaned[-index,]
```

3.Creating some new features

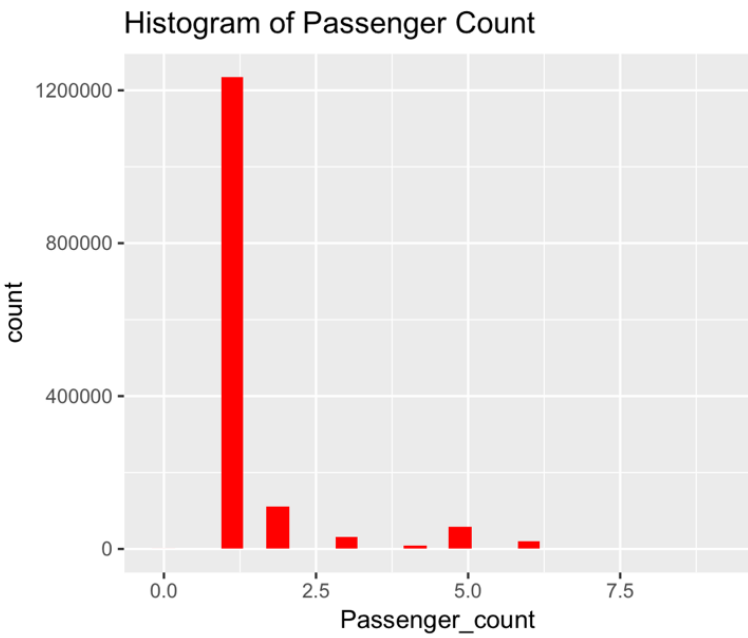
Calculation of Trip Time using lubridate package

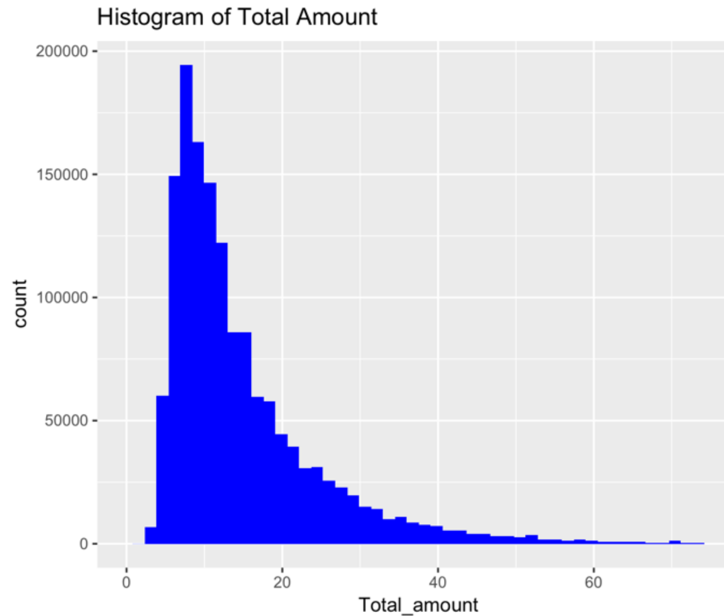
```
data$Triptime=as.POSIXct(data$Lpep_dropoff_datetime,"%Y-%m-%d %h:%m:%s")-
as.POSIXct(data$lpep_pickup_datetime,"%Y-%m-%d %h:%m:%s")
data$Triptime=as.numeric(data$Triptime,units="mins")
```

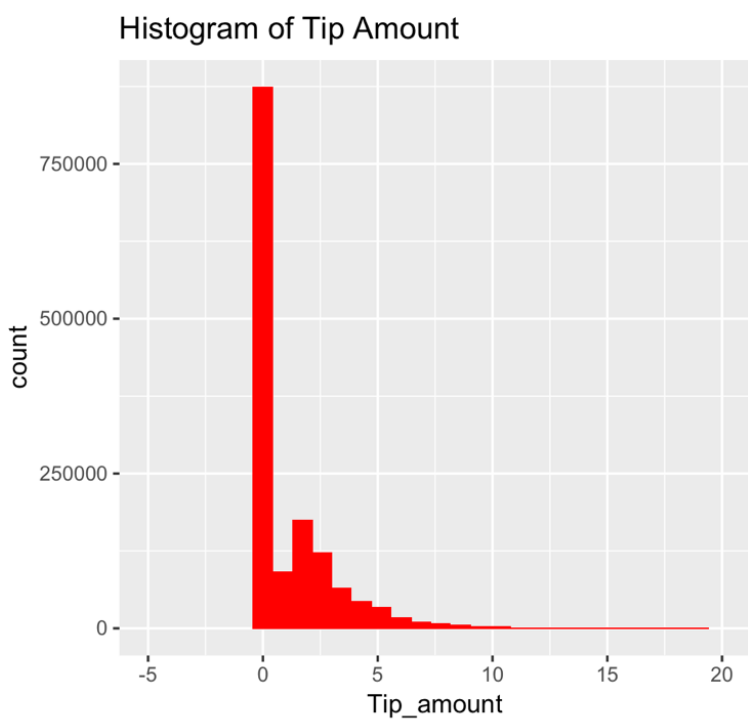
Calculation of Trip Speed by using distance divided by time

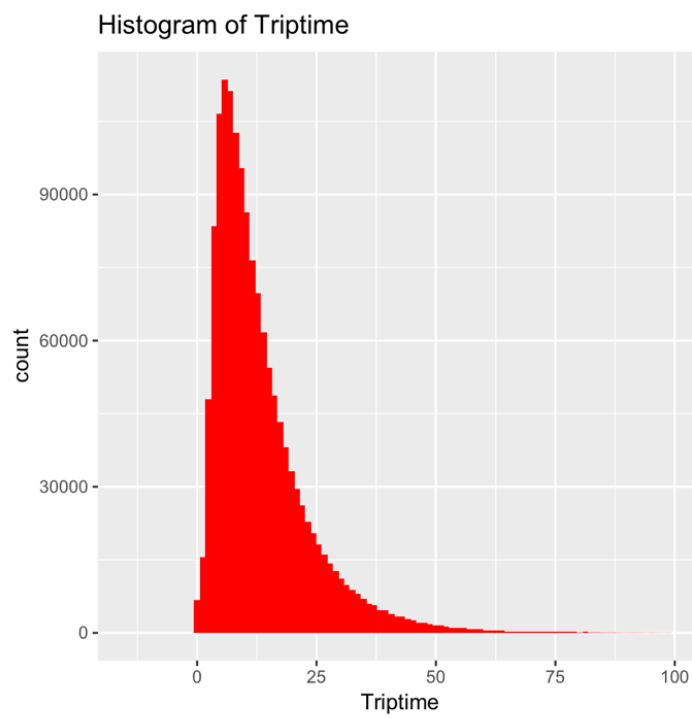
```
data$speed=data$Trip_distance/data$Triptime   # speed in miles per minute
```

4.Checking the distribution of various features in our data/Transformation





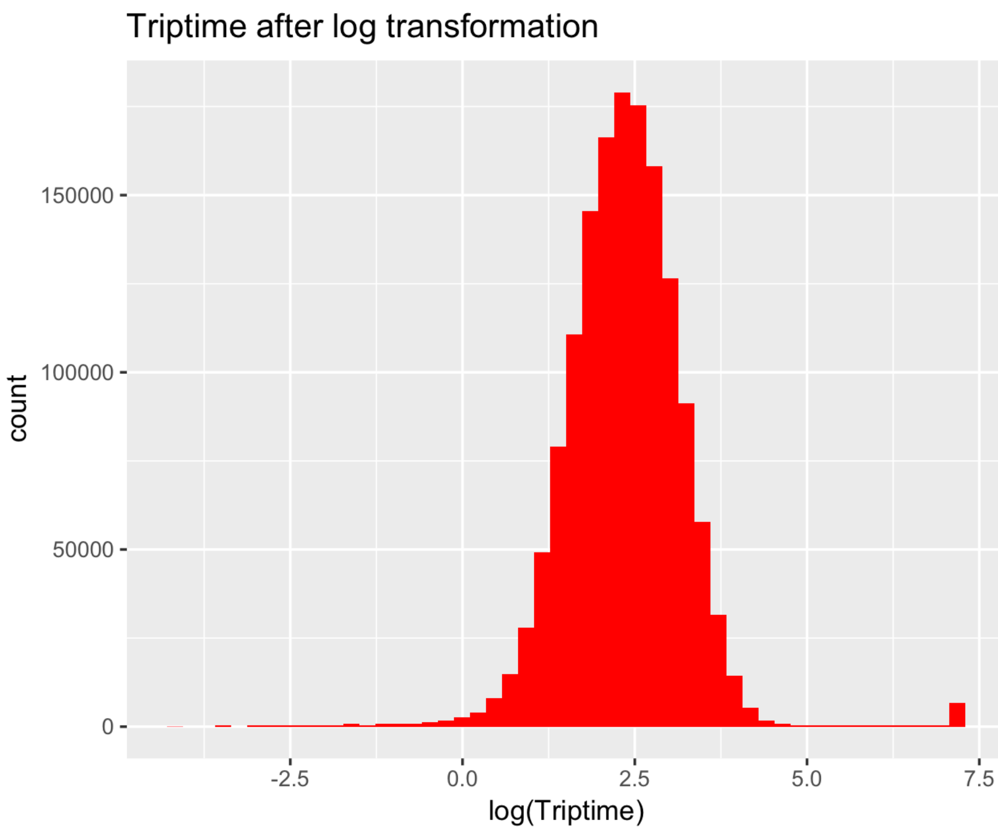




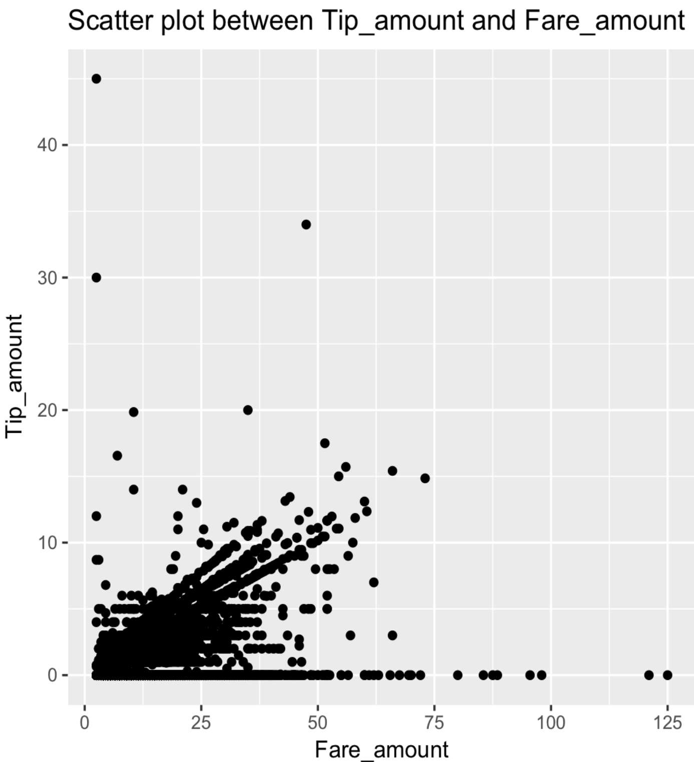
We observe that a lot of our features ar**e skewed to the right**.We can try log transformations to normalize our data.

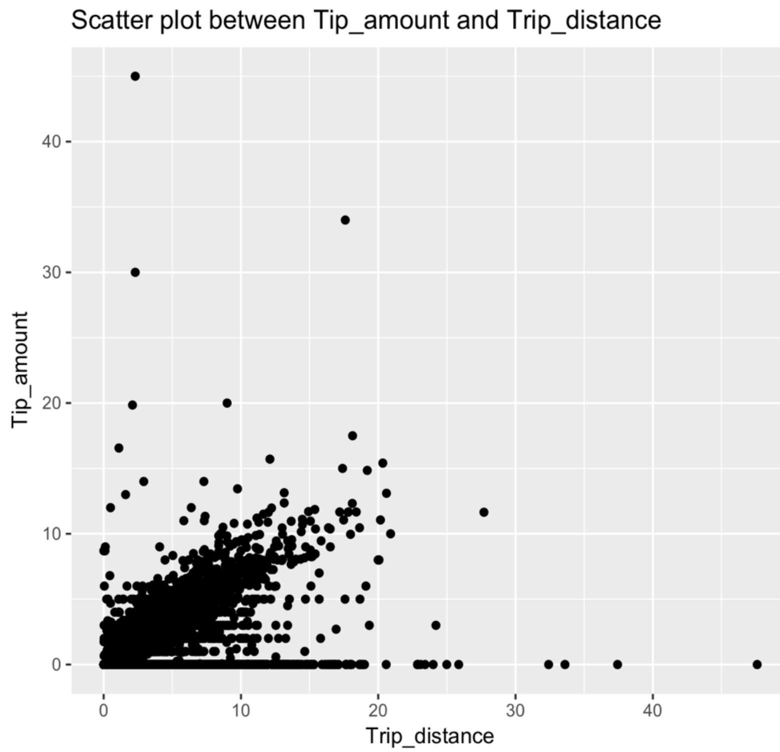
<u>Log Transformation</u>

datacleaned$speedlog=log(datacleaned$speed)
datacleaned$Total_amountlog=log(datacleaned$Total_amount)
datacleaned$Triptimelog=log(datacleaned$Triptime)
datacleaned$Fare_amountlog=log(datacleaned$Fare_amount)
datacleaned$Tip_amountlog=log(datacleaned$Tip_amount)



After log transformation the variables become more normal.





There is a **correlation** between Tip_amount and Fare_amount & Tip_amount and Trip_Distance as expected. So these variables can be used to model Tip_amount.

Linear Regression Model

```
lm.fit=lm(Tip_amount~Payment_type+Fare_amount+Total_amount+Trip_distance,data=traindata)
```

```
summary(lm.fit)
```

```
Call:
lm(formula = Tip_amount ~ Payment_type + Fare_amount + Total_amount +
    Trip_distance, data = traindata)

Residuals:
    Min      1Q  Median      3Q     Max
-71.697  -0.197  -0.018   0.302  72.049

Coefficients:
                Estimate Std. Error  t value Pr(>|t|)
(Intercept)    0.0512701  0.0030890    16.60   <2e-16 ***
Payment_type  -0.5062470  0.0016187  -312.75   <2e-16 ***
Fare_amount   -0.7430810  0.0004203 -1767.98   <2e-16 ***
Total_amount   0.7586510  0.0003577  2121.19   <2e-16 ***
Trip_distance -0.0443389  0.0005438   -81.54   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.7755 on 1174872 degrees of freedom
Multiple R-squared:  0.8853,    Adjusted R-squared:  0.8853
F-statistic: 2.266e+06 on 4 and 1174872 DF,  p-value: < 2.2e-16
```

The value of F-stat is high with a low p-value. The model to predict Tip_amount explains 88.5% variance with an R^2 of 0.8853 on the train data.

Testing the Linear Regression model

```
pred_lr=predict(lm.fit,testdata)
```

```
SSE =sum((pred_lr-testdata$Tip_amount)^2)
SSE
```

176273.86644352

```
SST =sum((pred_lr-mean(pred_lr))^2)
SST
```

1334650.06308689

```
1-(SSE/SST)
```

0.867925030448941

The R^2 for the test data is around 0.87 which means our model is doing pretty good.

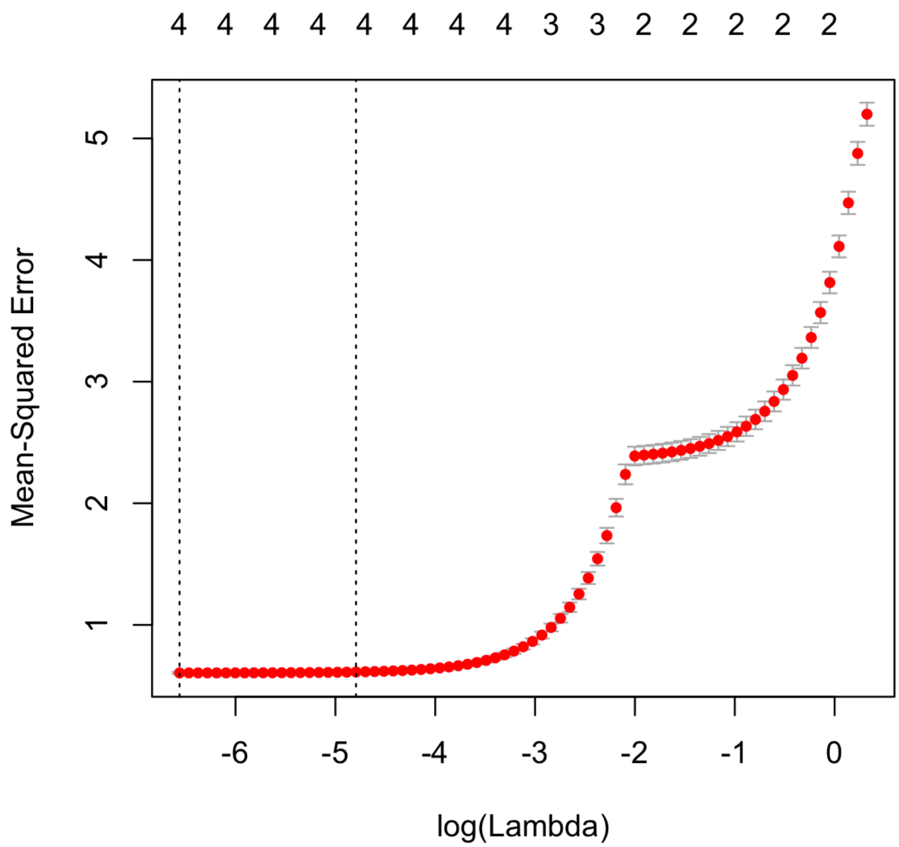
Lasso Regression Model

In lasso regression the coefficients of the regression are regularised so there is less chances of overfitting.Using the variables Payment_type, Fare_amount , Total_amount and Trip_Distance, I fitted a lasso model.

```
xmatrix <- data.matrix(traindata[, c(11, 12, 19, 20)])
y<- as.vector(traindata$Tip_amount)
fit_lasso = cv.glmnet(xmatrix,y, alpha = 1)
```

Using the cv.glmnet function I selected the best lambda value . Lambda is lowest when all the four variables are selected.

plot(fit_lasso)




```
glm(formula = traindata$Tip_amount~traindata$Payment_type+traindata$Total_amount
    +traindata$Trip_distance+traindata$Fare_amount)
```

```
Call:  glm(formula = traindata$Tip_amount ~ traindata$Payment_type + 
    traindata$Total_amount + traindata$Trip_distance + traindata$Fare_amount)

Coefficients:
           (Intercept)  traindata$Payment_type  traindata$Total_amount  
               0.05127                -0.50625                 0.75865  
traindata$Trip_distance   traindata$Fare_amount  
              -0.04434                -0.74308  

Degrees of Freedom: 1174876 Total (i.e. Null);  1174872 Residual
Null Deviance:	    6159000 
Residual Deviance: 706600 	AIC: 2737000
```

<u>Testing the lasso model</u>

```
> SSE =sum((pred_lasso-testdata$Tip_amount)^2)
>
> SSE
[1] 173386
> SST =sum((pred_lasso-mean(pred_lasso))^2)
> SST
[1] 1333527
> 1-(SSE/SST)
[1] 0.8699794
```

There is slight improvement in R^2 for test data (0.869) with the lasso model.

Modeling for Tip_percent

linearfit3=lm(tipPercent~Fare_amount+Trip_distance+Total_amount+Payment_type+Trip_type+speed+Tip_amount,data=datacleaned)

```
summary(linearfit3)
```

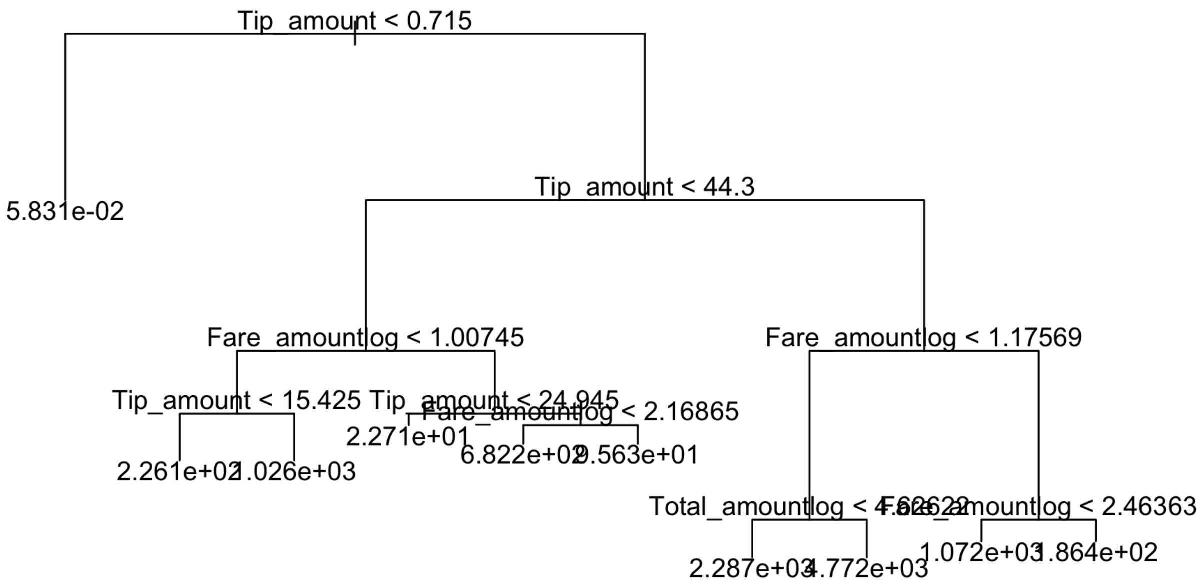
```
Call:
lm(formula = tipPercent ~ Fare_amount + Trip_distance + Total_amount +
    Payment_type + Trip_type + speed + Tip_amount, data = datacleaned)

Residuals:
    Min      1Q  Median      3Q     Max
-1262.1    -4.1    -1.1     3.3  9931.6

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)   11.520476   0.172348  66.844   <2e-16 ***
Fare_amount   -0.200919   0.020923  -9.603   <2e-16 ***
Trip_distance -0.223640   0.014425 -15.503   <2e-16 ***
Total_amount  -0.508031   0.020385 -24.922   <2e-16 ***
Payment_type  -2.336820   0.043312 -53.953   <2e-16 ***
Trip_type      2.246727   0.152456  14.737   <2e-16 ***
speed          0.305631   0.005116  59.746   <2e-16 ***
Tip_amount     8.142004   0.023848 341.406   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 22.23 on 1468589 degrees of freedom
Multiple R-squared:  0.3497,    Adjusted R-squared:  0.3497
F-statistic: 1.128e+05 on 7 and 1468589 DF,  p-value: < 2.2e-16
```

After playing around with linear regression and trying different predictors with and without transformation,we get a low value of R^2. Our goal is to make R^2 as close to one as possible. So we can try non linear models like regression trees.

Tree models

After trying different combinations of predictors and transformations

library(tree)
treemodel2=tree(tipPercent~Fare_amountlog+Total_amountlog+Payment_type+Triptimelog+speedlog+Tip_amount,data=traindata)
plot(treemodel2)




```
summary(treemodel2)
```

```
Regression tree:
tree(formula = tipPercent ~ Fare_amountlog + Total_amountlog +
    Payment_type + Triptimelog + speedlog + Tip_amount, data = traindata)
Variables actually used in tree construction:
[1] "Tip_amount"      "Fare_amountlog"  "Total_amountlog"
Number of terminal nodes:  10
Residual mean deviance:  201 = 236200000 / 1175000
Distribution of residuals:
     Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
-2308.000    -0.058    -0.058     0.000    -0.058  6951.000
```

Testing the tree

```
> treemodel2=tree(tipPercent~Fare_amountlog+Total_amountlog+Payment_type+Triptimelog+
+               speedlog+Tip_amount,data=traindata)
> tree.pred2=predict(treemodel2,newdata=testdata)
> SSE =sum((tree.pred2-testdata$tipPercent)^2)
> SST =sum((tree.pred2-mean(tree.pred2))^2)
> SST
[1] 163950950
> SSE
[1] 31942985
>
> 1-(SSE/SST)
[1] 0.8051674
```

We get a R^2 of around 80.5% with a single tree. A single tree usually has high variance and the results will vary from iteration to itearation.

Following steps can be taken further to improve the model if higher accuracy is needed and the time permits

- Tree Pruning can be done using prune.tree() function. This will reduce the number of nodes in the tree and can help in accuracy sometimes.
- Multiple trees can be grown using random forests These trees are decorrelated and this can reduce overfitting. randomForest package can be used from R
- Boosted trees can be used where trees are grown incrementally and parameter can be tuned using cross validation. Gradient boosting **xgb package and gbm packages** can be tried which give surprisingly good results in my experience earlier.
- Stepwise forward and backward regression can be done to incease the accuracy by selecting best variables.
- Other transformation of predictors can be tried like quadratic or cubic.
- If time permits some **more complicated features** can be tried like **hour_of_day, day_of_week.** It is possible that passengers tend to tip more on Fridays because they are happier during the weekends and less on a Monday. They can also tip more on late night trips because they appreciate the service they are getting. These features can be investigated further.
- We can create a feature for passengers travelling to the airport tip more if the speed is high and they reach earlier.

***Option A*: Distributions**
Build a derived variable representing the average speed over the course of a trip.

Created a variable for speed in the previous section .

data$speed=data$Trip_distance/data$Triptime

Can you perform a test to determine if the average trip speeds are materially the same in all weeks of September? If you decide they are not the same, can you form a hypothesis regarding why they differ?

Difference in speed by week

We can calculate means for each groups weekly by creating a week_of_year variable using week function in the **lubridate** package. We can then perform group mean calculations using mutate function from **dplyr.**

dataclean2=datacleaned %>% group_by(week_of_year = week(lpep_pickup_datetime))%>% mutate(avgspeed= mean(speed))

Statistical tests are chosen on the basis of normal distribution of data, number of groups and independency of samples.

We have five different groups here for mean comparison corresponding to each week in November.
We see from the histograms we made earlier that the speed of the vehicles is not a normal distribution. I use the Kruskal-Wallis rank sum test which is a non parametric test.

```
kruskal.test(dataclean2$speed,dataclean2$week_of_year)
```

```
	Kruskal-Wallis rank sum test

data:  dataclean2$speed and dataclean2$week_of_year
Kruskal-Wallis chi-squared = 3725.1, df = 4, p-value < 2.2e-16
```

Kruskal wallis has a low p value which can be used to reject the null hypothesis.
There is a significant difference in the speed of taxis during different weeks in November.

Hypothesis
There is a public holiday in September (Labor day). Less people will be travelling on this day. Also all the weeks will not have the same no of days as September has 30 days and all the weeks will not have same number of days which can add to the difference in means.

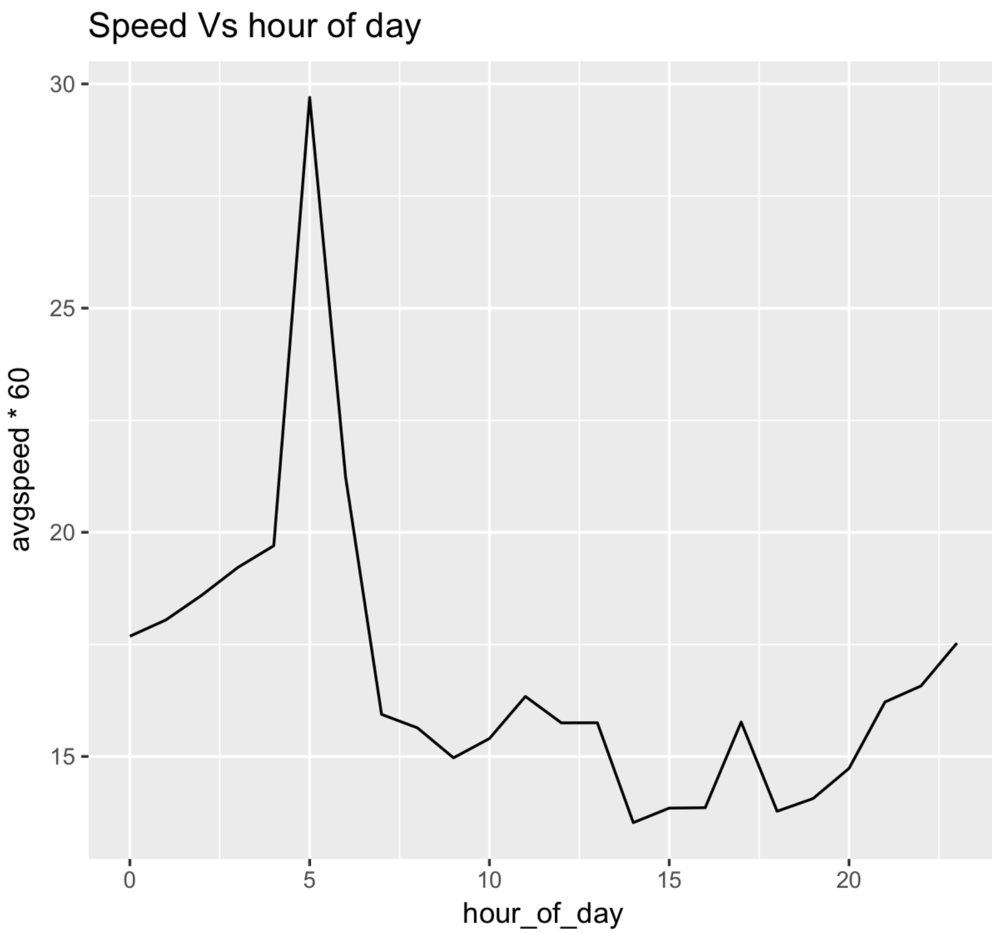
Difference in speed by hour
Hour function from lubridate package is used to form different groups.

dataclean4=datacleaned %>% group_by(hour_of_day= hour(lpep_pickup_datetime))%>% muta te(avgspeed= mean(speed))

```
kruskal.test(dataclean4$speed,dataclean4$hour_of_day)
```

```
        Kruskal-Wallis rank sum test

data:  dataclean4$speed and dataclean4$hour_of_day
Kruskal-Wallis chi-squared = 139310, df = 23, p-value < 2.2e-16
```

There is a significant difference in the hourly average speeds as can be seen by the low p value .



Hypothesis:
We can see higher speeds after 12 in the night which reaches a maximum at 5 in the morning. The speeds during the day are pretty low and drops around at 9 in the morning and during the evening hours possibly due to people travelling to and from work.