Green Taxi Trip Analysis & Predictive Modelling

BY: KETAN WALIA

Step 1

- Programmatically download and load into your favorite analytical tool the trip data for September 2015.
- Report how many rows and columns of data you have loaded. Solution:

a) The data was programmatically downloaded in an automated fashion and was loaded into R Studio. The code used is:

```
url<- "https://s3.amazonaws.com/nyc-tlc/trip+data/green_tripdata_2015-09.csv"
d<-getwd()
setwd(file.path(d))
local<-file.path('NYC')
download.file(url,local)
data<-read.csv(local,sep=',')
```

b) Number of Rows: 1494926 Number of Columns: 21

Code Used:

```
dim(data)
cat("Number of Rows in dataset:",dim(data)[1])
cat("Number of Columns in dataset:",dim(data)[2])
```

Step 2

- Plot a histogram of the number of the trip distance ("Trip Distance").
- Report any structure you find and any hypotheses you have about that structure.

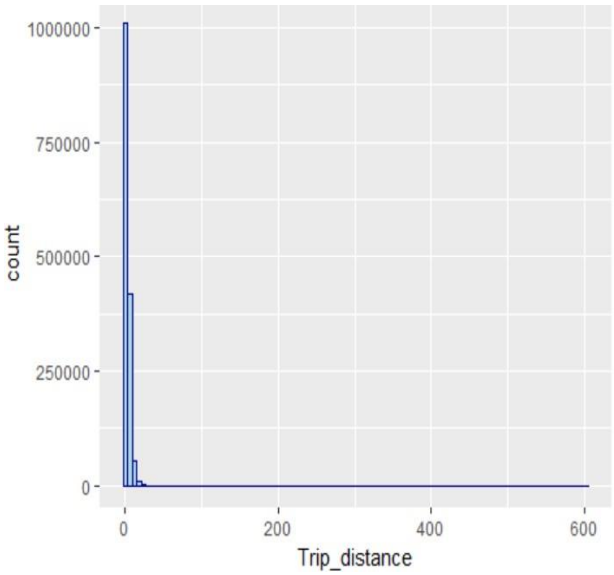
Solution:

Code Used:

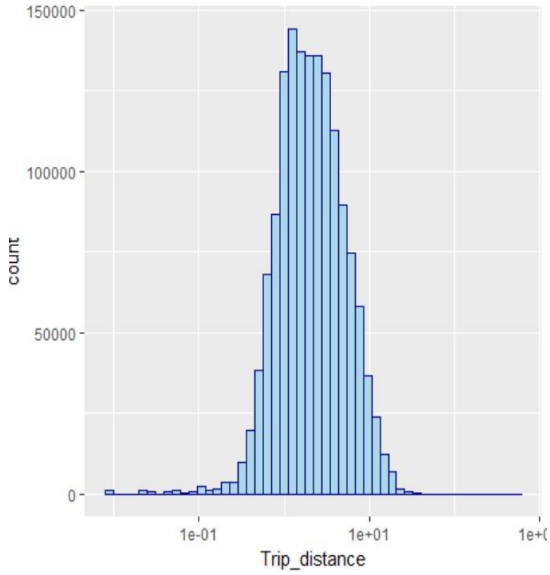
```
ggplot(data, aes(x=Trip_distance ))+
  geom_histogram(color="darkblue", fill="lightblue",bins=500)

ggplot(data, aes(x=Trip_distance ))+
  geom_histogram(color="darkblue", fill="lightblue",bins=50)+scale_x_log10()
```

a) Unit of Trip Distance= Miles

Histogram without Transformation



Histogram with Log Transformation



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

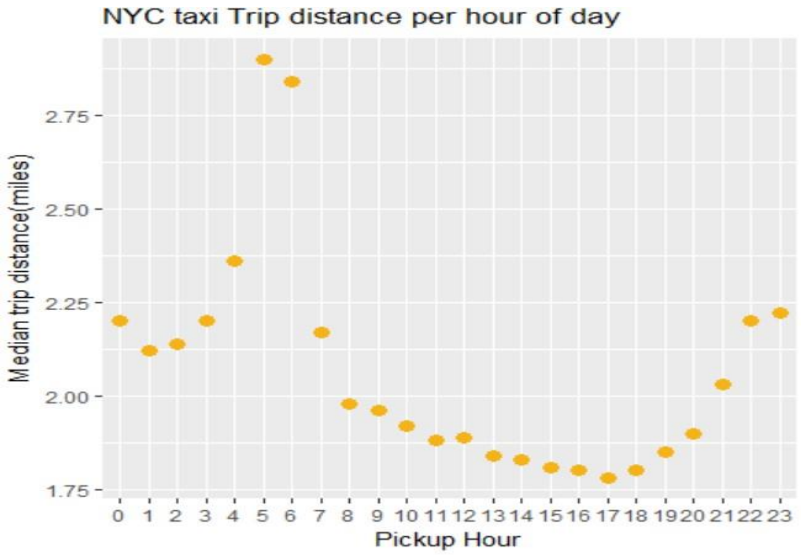
The frequency distribution of variable "Trip Distance" resembles **Log-Normal Distribution**. The frequency distribution of "Trip Distance" is skewed to the right. Consequently, its median is smaller than its mean. Summary of distribution is:

```
summary(data$Trip_distance)
 Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
0.000   1.100   1.980   2.968   3.740 603.100
```

Majority of the trips lie between 0 to 1.98 miles range and the frequency of the trips decreases as the Trip distance. There is a huge gap between the Minimum and Maximum value. The distribution indicates potential outliers towards the tail of the distribution. Presence of outliers also emerges after the Log transformation is done on the variable and its frequency distribution is plotted. It is evident that there are certain trips far away from the central tendency of the data. This pattern implies that people use Green taxi for small commutes.

In order to develop and narrow the hypothesis further it is interesting to observe **Median Trip Distance as a function of Pickup Hour**

Visualization:



From the above plot, it is evident that Trip distance also varies as a function of hour of the day. Trip distance is particularly higher for office hours i.e morning hours and evening hours. It implies that people living close to their work place (or any other place they regularly visit) mainly use green taxi for commute.

Hypothesis:

On an average, people use Green Taxi mainly to commute between home & work in the distance range of 0 to 3 miles.

Code Used:

```
train.data %>%
  group_by(pickup_hour) %>%
  summarise(median_trip = median(Trip_distance))%>%
  ggplot(aes(x = as.factor(pickup_hour), y = median_trip)) +
  geom_point(colour = "#F5B317", size = 3)+
  ggtitle("NYC taxi Trip distance per hour of day") +
  xlab("Pickup Hour") +
  ylab("Median trip distance(miles)")
```

==========

Step3

- **Report mean and median trip distance grouped by hour of day.**
- **We'd 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.**

Solution: a) Report mean and median trip distance grouped by hour of day CODE

USED:

```
train.data %>%
  group_by(pickup_hour) %>%
  summarise(median_trip = median(Trip_distance),mean_trip = mean(Trip_distance))
```

Result:

	pickup_hour	median_trip_distance	mean_trip_distance
1	0	2.20	3.115276
2	1	2.12	3.017347
3	2	2.14	3.046176
4	3	2.20	3.212945
5	4	2.36	3.526555
6	5	2.90	4.133474
7	6	2.84	4.055149
8	7	2.17	3.284394
9	8	1.98	3.048450
10	9	1.96	2.999105
11	10	1.92	2.944482
12	11	1.88	2.912015
13	12	1.89	2.903065
14	13	1.84	2.878294
15	14	1.83	2.864304
16	15	1.81	2.857040
17	16	1.80	2.779852
18	17	1.78	2.679114
19	18	1.80	2.653222
20	19	1.85	2.715597
21	20	1.90	2.777052
22	21	2.03	2.999189
23	22	2.20	3.185394
24	23	2.22	3.191538

b) Can you provide a count of how many transactions fit this criteria, the average fair, and any other interesting characteristics of these trips.

Solution: It was realized that the RateCode ID: 2 & 3 corresponds to JFK and Newark airports.

Reference-

http://www.nyc.gov/html/tlc/downloads/pdf/data_dictionary_trip_records_green.pdf

http://www.nyc.gov/html/tlc/html/passenger/taxicab_rate.shtml **Code Used:**

```
airport_trips<-subset(train.data,(RateCodeID==2)|(RateCodeID==3))
View(airport_trips%>%group_by(RateCodeID)%>%summarise(Transaction_count=n(),avg_fare=mean(Fare_amount)))
View(airport_trips%>%summarise(Transaction_count=n(),avg_fare=mean(Fare_amount)))
```

Results:

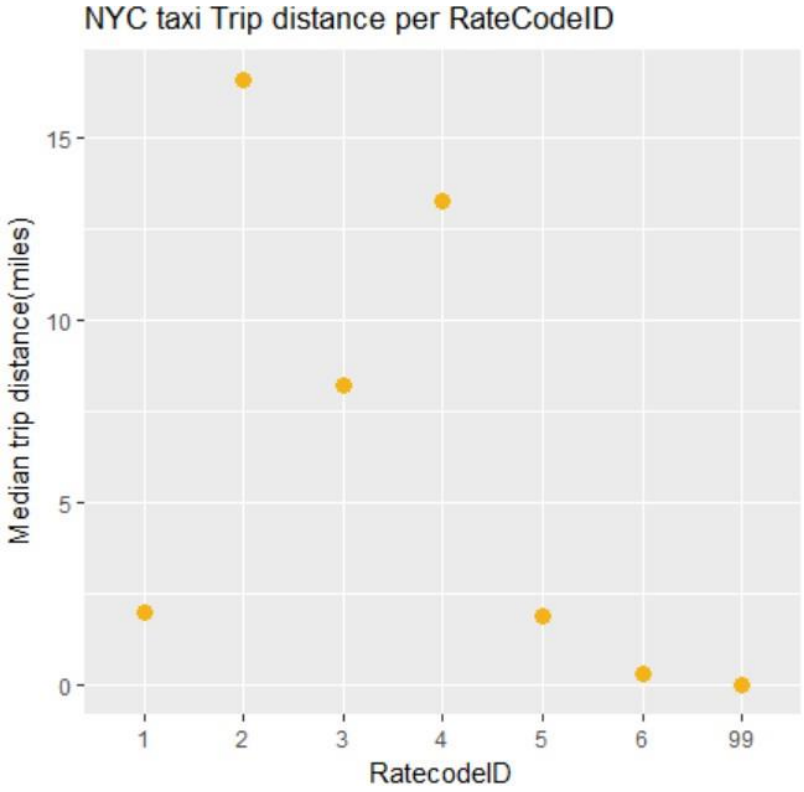
	RateCodeID	Transaction_count	avg_fare
1	2	4435	49.02187
2	3	1117	48.79857

Aggregated results: Taking RateCodeID 1 & 2 together

	Transaction_count	avg_fare
1	5552	48.97695

Further Investigation:

Hypothesis: *Median Trip distance varies with respect to different RateCodeID.*



Inference: The Median Trip Distance is comparatively much higher for RateCodeID: 2 & 3 corresponding to JFK and Newark airport. The Median Trip Distance for RateCodeID: 4 is infact higher than that for Newark airport which is justifiable for the fact that these trips are Out of town trips and thus have more distance to cover.

Code Used:

```
train.data %>%
  group_by(RateCodeID) %>%
  summarise(median_trip_dist = median(Trip_distance))%>%
  ggplot(aes(x = as.factor(RateCodeID), y = median_trip_dist)) +
  geom_point(colour = "#F5B317", size = 3)+
  ggtitle("NYC taxi Trip distance per RateCodeID") +
  xlab("RatecodeID") +
  ylab("Median trip distance(miles)")
```

Step 4

- Build a derived variable for tip as a percentage of the total fare.

- 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.

SOLUTION:

a) Build a derived variable for tip as a percentage of the total fare.

The variable was derived using Tip amount and Total amount pertaining to a trip.

Code Used:

```
train.data$Tip_Perc<-(train.data$Tip_amount/train.data$Total_amount)*100
```

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.

During exploratory analysis it was observed that certain variables could be derived from existing set of variables and could be used to better predictive modelling. These variables were derived based on logical understanding, data visualization & referring to the NYC taxi website.

Derived Variables

Following variables were derived or modified:

1) **Time_Phase:** The variable "Pickup_hour" was categorized into 4 bins based on frequency distribution of trips corresponding to different hours in a day. Hours ranging in between : 0-7 were coded as "1"

7-2 as "2"

2-8 as '3'

8-0 as '4'

Code Used:

```
train.data$time_phase<-ifelse(train.data$pickup_hour %in% 0:7,1,
                              ifelse(train.data$pickup_hour %in% 7:2,2,
                                     ifelse(train.data$pickup_hour %in% 2:8,3,4)))
```

2) **Pick & Drop Longitudes & Latitudes:**

These variables were rounded off to 2 decimal places. The idea was that the places which are pretty close to each other represents common characteristics as such grouping them locally would capture its trend and would help the model to learn better.

Code Used:

```
train.data<-train.data %>%
  mutate(pick_lat = round(Pickup_latitude, 2),
         pick_long = round(Pickup_longitude,2),
         drop_lat = round(Dropoff_latitude, 2),
         drop_long = round(Dropoff_longitude,2)

  )
```

3) **Pick_Cnt & Drop_Cnt:**

These variables represent the counts of trips pertaining to a particular location. They capture popularity for different locations and Tip amount varies with it.

Code Used:

```
train.data<-train.data %>% mutate(pick_var = paste(pick_long,pick_lat),drop_var=paste(drop_long,drop_lat))
p<-train.data%>% group_by(pick_var)%>%summarise(pick_cnt=n())
d<-train.data%>% group_by(drop_var)%>%summarise(drop_cnt=n())
```

4) **Date Variable**s: Following variables were derived from the timestamps-
-pickup_week_day
-pickup_month
-pickup_hour
-pickup_date
-dropoff_week_day
-dropoff_month
-dropoff_hour
-dropoff_date

Code Used:

```
train.data <- data %>%
  mutate(pickup_week_day = wday(lpep_pickup_datetime, label = T),
         pickup_month = month(lpep_pickup_datetime, label = T),
         pickup_hour = hour(lpep_pickup_datetime),
         pickup_date = as_date(lpep_pickup_datetime),
         dropoff_week_day = wday(Lpep_dropoff_datetime, label = T),
         dropoff_month = month(Lpep_dropoff_datetime, label = T),
         dropoff_hour = hour(Lpep_dropoff_datetime),
         dropoff_date = as_date(Lpep_dropoff_datetime)
  )
```

5) **Week Number**: Trips were assigned week number which according to date of a particular trip.

Code Used:

```
train.data$days_sep<-train.data$pickup_date-as.Date("2015-09-01")
train.data$days_sep<-as.integer(train.data$days_sep)

train.data$week<-1

a<-which(train.data$days_sep %in% seq(0,7))
b<-which(train.data$days_sep %in% seq(8,14))
c<-which(train.data$days_sep %in% seq(15,21))
d<-which(train.data$days_sep %in% seq(22,31))

train.data$week[a]<-"w1"
train.data$week[b]<-"w2"
train.data$week[c]<-"w3"
train.data$week[d]<-"w4"
str(train.data)
```

6) **Trip Duration: Duration of each trip was derived using the code:**

```
train.data$Trip_Duration<-strptime(train.data$Lpep_dropoff_datetime,
                                   format="%Y-%m-%d %H:%M:%S")-strptime(train.data$lpep_pickup_datetime,
                                                                        format="%Y-%m-%d %H:%M:%S")
```

7) **Speed: Variable Speed was derived based on the formula {Speed=Trip_Distance/Trip_duration}**

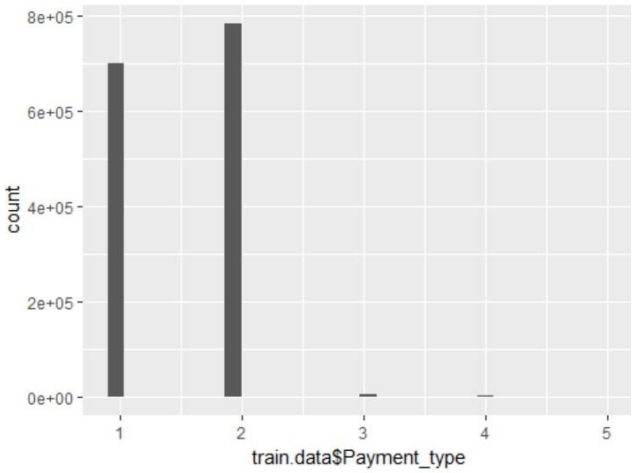
Code:

```
train.data$Speed<-train.data$Trip_distance/train.data$Trip_Duration
```

DATA CLEANING

1) Remove all the trips with total amount <$2.5 as this is the minimum amount chargable for any trip as per the official website for NYC taxi-
http://www.nyc.gov/html/tlc/html/passenger/taxicab_rate.shtml
2) Remove the variable Ehail_Fee as number of missing values is 1494926 i.e almost all values are missing for this variable.
3) Remove all the trips with Trip Duration or Trip Distance or Fare_amount or Passenger_count or Extra <=0
4) Taking only the absolute value of variable Tip Amount. Assumption being that negative Tip amount was entered negative by mistake.
5) Remove all the trips with dropoff month=”October”
6) Removing variables with Payment_Type in (3,4,5) as the number of observations pertaining to these codes are very rare. The dictionary meaning of these codes are**:**

3= No charge 4= Dispute 5= Unknown 6= Voided trip. Given the dataset it is very difficult to detect if the trip would fall under these categories.



	Payment_type	count
1	1	701287
2	2	783699
3	3	5498
4	4	4368
5	5	74

6) Removing redundant variables.

7) Remove all the trips with Speed<=0 or Speed>250 mph. 250 mph is the maximum speed Green taxi can reach which again is very unrealistic in NYC. This aspect needs to be worked upon further in future.

Code for Above Section:

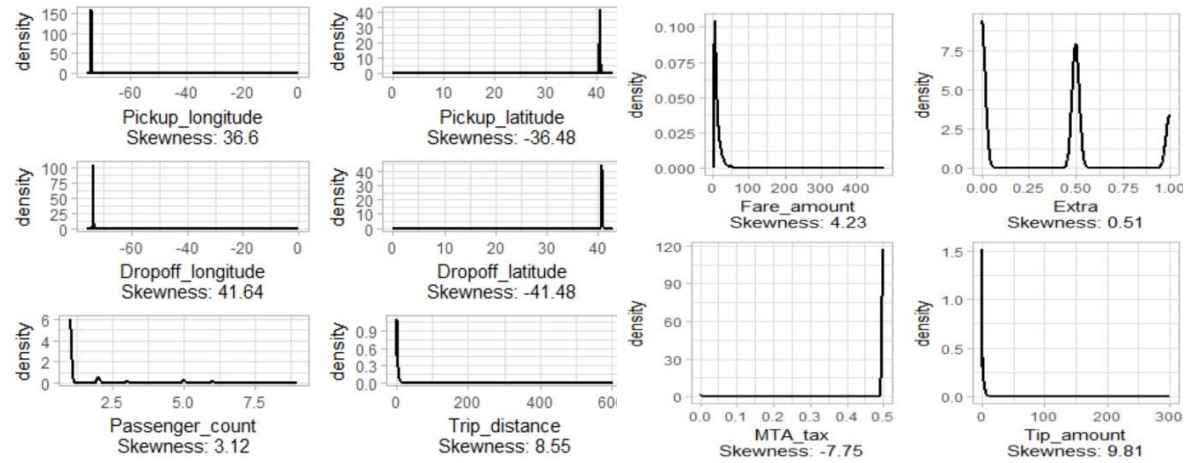
```
#1) PART 4:a) Build a derived variable for tip as a percentage of the total fare.

train.data$Tip_Perc<-(train.data$Tip_amount/train.data$Total_amount)*100

#2)
train.data$Trip_Duration<-strptime(train.data$Lpep_dropoff_datetime,
                                   format="%Y-%m-%d %H:%M:%S")-strptime(train.data$lpep_pickup_datetime,
                                                                        format="%Y-%m-%d %H:%M:%S")

#3)
train.data$Trip_Duration<-as.numeric(train.data$Trip_Duration)/60

#4)
##Remove rows with trip less than $2.5
train.data<-train.data[train.data$Total_amount>=2.5,]
#1494926 - 1487767=7159

#5)
summary(train.data$Ehail_fee)<-NULL


#6) Remove rows having trip duration less than or equal to zero
train.data<-train.data[-which(train.data$Trip_Duration<=0),]

#7) Deriving variable named "Speed=Distance/Time"
train.data$Speed<-train.data$Trip_distance/train.data$Trip_Duration

#8) Remove rows having Trip_distance less than or equal to zero
train.data<-train.data[-which(train.data$Trip_distance==0),]

#9) Assumption that negative Tip amount was entered negative by mistake. Thus, considering
## only its absolute value
train.data$Tip_amount<-abs(train.data$Tip_amount)
```

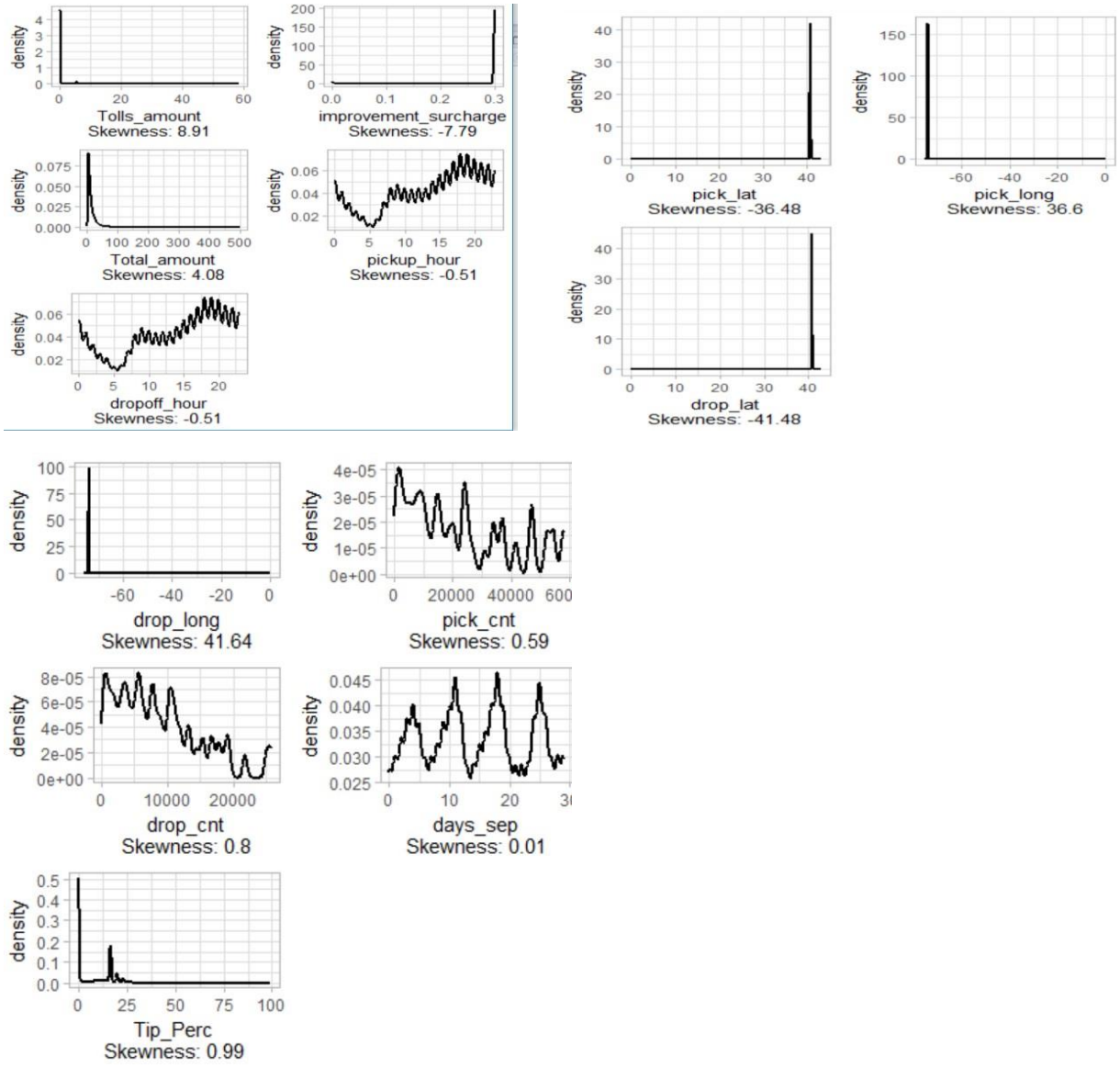
```
#13)
train.data$VendorID<-as.factor(train.data$VendorID)
train.data$RateCodeID<-as.factor(train.data$RateCodeID)
train.data$Payment_type<-as.factor(train.data$Payment_type)
train.data$Trip_type<-as.factor(train.data$Trip_type)
train.data$pickup_week_day<-factor(train.data$pickup_week_day)

#14)
train.data<-train.data[-which(train.data$Payment_type==3),]
train.data<-train.data[-which(train.data$Payment_type==4),]
train.data<-train.data[-which(train.data$Payment_type==5),]
```

DATA EXPLORATION

Categorical Variable Distribution:

Continous Variable Distribution:





The distribution of most of the numeric variables is highly skewed. Many of the variables resembles lognormal distribution including the response variable “Tip_Perc”. For these variables i.e [Trip_distance,Fare_amount,Trip_Duration,Speed,pick_cnt,pick_cnt] log transformation was done. Three functions were written in R to create the above plots.

Code Used:

```r
## Writing function to plot bar plots

plotHist <- function(data_in, i) {
  data <- data.frame(x=data_in[[i]])
  p <- ggplot(data=data, aes(x=factor(x))) + stat_count() + xlab(colnames(data_in)[i]) + theme_light() +
    theme(axis.text.x = element_text(angle = 90, hjust =1))
  return (p)
}


#Writing function to plot densities
plotDen <- function(data_in, i){
  data <- data.frame(x=data_in[[i]]) #, SalePrice = data_in$SalePrice)
  p <- ggplot(data= data) + geom_line(aes(x = x), stat = 'density', size = 1,alpha = 1.0) +
    xlab(paste0((colnames(data_in)[i]), '\n', 'Skewness: ',round(skewness(data_in[[i]], na.rm = TRUE), 2)))
  + theme_light()
  return(p)

}

## Function calling the previous two functions
doPlots <- function(data_in, fun, ii, ncol=3) {
  pp <- list()
  for (i in ii) {
    p <- fun(data_in=data_in, i=i)
    pp <- c(pp, list(p))
  }
  do.call("grid.arrange", c(pp, ncol=ncol))
}




train.data.cat<-train.data[,(which(sapply(train.data,is.factor)))]
train.data.numeric<-train.data[,(which(sapply(train.data,is.numeric)))]

library(ggplot2)
library(gridExtra)
library(e1071)
doPlots(train.data.cat, fun = plotHist, ii = 1:6, ncol = 2)

doPlots(train.data.numeric, fun = plotDen, ii = 1:6, ncol = 2)
```

Outlier Detection & Missing Value

Two functions were written to detect & remove univariate instances lying 3.5 SD away from mean. The total number of 7362 outliers were detected. Out of these 214 were removed. These were observations for which variable Speed was more than mean+4 SD away. Two functions were written in R for univariate outlier detection & removal. These functions are: "findOutlier' & 'removeOutlier'. For rest of the observations multivariate outlier detection using Cooks' distance was used. (This is described in the later section of this report while Modelling). There are actually very few observations having missing values so row deletion as in this case it would not have significant impact on model generalization.

INSIGHT

An interesting insight was uncovered in the distribution of the percentage tip. It was found that among those who paid tip 99.9% of these payments were done by credit cards. When referred to data dictionary it was mentioned that-

"Tip amount – This field is automatically populated for credit card tips. Cash tips are not included." It doesn't mean however that no tip was paid. The data is simply missing.

Constraint

The constraint here is that around 50% of the data pertaining to response variable is missing.

Challenge

In this case it could be detrimental to blindly build a model on available instances i.e belonging to trips paid by Credit Card only and assume that it would generalize to other trips i.e the one paid by cash.

Approach

It is thus required to set up set of Experiments and test if the trips paid by credit card and cash are actually different or not.

Experiments

Following Experiments were done to address the mentioned challenge:

Experiment 1:-

Test if the trips paid by Credit Card are significantly different from those paid by Cash.

Assumption:

It was assumed that Observations for which Payment_Type = Credit Card comes from class1 otherwise class 2

Hypothesis:

H0: *The observations from class 1 could be well differentiated from class 2.*

H1: *The observations from class 1 could not be well differentiated from class 2.*

Methodology: The **classifier Random Forest** was used to classify the instances coming from the two classes. It was assumed that if the two classes are indeed different Random Forest would give low misclassification rate for both the classes.

Since the number of observations in the dataset are too large for the local machine a random sample was extracted from the full dataset. The random sample extracted had 350000 rows. This sample size was chosen based on the capacity of the local machine and also to represent patterns of the full dataset.

Code Used:

Random Forest was implemented using the Library: Ranger in R. It is a fast implementation of Random Forests. The library Random Forest was used to tune the parameters of the model.

Random Forest was tuned to select the parameters:

"mtry"- the number of variables selected to build model for each tree.

```
library(randomForest)
bestmtry<-tuneRF(train1_numeric1,train1_numeric1$Payment_type,stepFactor = 1.2,
                 improve = 0.01,plot=TRUE,trace=TRUE)
#rf<-randomForest(Payment_type~.,data=train1_numeric1,mtry=4,ntree=50,proximity=FALSE)

library(ranger)
rf<-ranger(Payment_type~.,data=train1_numeric1,mtry=4, num.trees=150,importance = "impurity")
print("Performance of Random Forest to classify between trips paid by Credit Card or Cash")
rf
```

"Num.tree"- number of trees to be built

<u>Results of Random Forest Classification:</u>

```
        OOB estimate of  error rate: 35.44%
Confusion matrix:
       1      2 class.error
1 102177  63591   0.3836144
2  60355 123590   0.3281144
```

Inference:

It is evident from the high misclassification rate for both the classes that Random Forest has not been able to separate the two class well enough. So based on results of Random Forest we reject the Null Hypothesis. High misclassification rate for both the classes indicate that observations from supposedly different classes are not separable enough. As such we reject the null hypothesis that The observations from class 1 could be well differentiated from class 2.

This inference leads to **our second Experiment**. The idea is that since the observations are not very different from each other they might be rather similar to each other.

Intuition for Experiment 2:-

The intent was to test if the trips paid by Credit Card are significantly similar to those paid by Cash. The intuition was that if trips paid by payment type Credit Card are different from those paid by cash then different clusters would be dominated by one of the types otherwise if trips from both payment types are indeed similar trips from one payment type would not dominate thee other **Experiment 2:**

Test if the trips paid by Credit Card are significantly similar to those paid by Cash.

Assumption:

The observations come from "k" number of gaussian clusters such that all the instances in the feature space belongs to one particular cluster.

Hypothesis:

H0: *For a given cluster the number of trips paid by Credit Card and those paid by Cash are in comparatively equal proportion.*

H1: *For a given cluster the number of trips paid by Credit Card and those paid by Cash are not in comparatively equal proportion i.e cluster of points for trips paid by cash are distinct from those paid by credit card.*

<u>Data Preparation for Clustering:</u>

Converting categorical variables to dummy variables using CARET package in R CODE:

```
train_Dummy<-dummyVars("~.",data=train1_numeric1)
final_train_use<-as.data.frame(predict(train_Dummy,train1_numeric1))
```

Methodology: The instances were clustered using clustering algorithm-Gaussian Mixture Model(GMM). The R library "ClusterR" was used for faster implementation of the algorithm**.**

<u>Improving Computation Efficiency:</u>

"ClusterR" package in R takes advantage of 'RcppArmadillo' to speed up the computationally intensive parts of the functions. Also, the code was run on parallel cores:

```
library(doSNOW)
sampleCount<-2
cluster<-makeCluster(sampleCount)
registerDoSNOW(cluster)
```

Optimal number of clusters were chose **based on "AIC" criteria** and **distance metric** chosen was **Mahalanobis distance**. The **optimal number of clusters** ultimately chosen were**: 9 Code Used:**

```
set.seed(101)
p<-Optimal_Clusters_GMM(final_train_use, max_clusters=50, criterion = "AIC",
                        dist_mode = "maha_dist", seed_mode = "random_subset", km_iter = 50,
                        em_iter = 100, verbose = FALSE, var_floor = 1e-10, plot_data = TRUE,
                        seed = 1)


g=GMM(final_train_use, gaussian_comps = 9,
      dist_mode = "maha_dist", seed_mode = "random_subset", km_iter = 50,
      em_iter = 100, verbose = FALSE, var_floor = 1e-10,
      seed = 1)
```

Results: The cross table below shows the frequency distribution of observations from each Payment type (1 & 2) with respect to each cluster(0 through 9). Each column represents one cluster and each represent Payment Type.

	0	1	2	3	4	5	6	7	8
1	5477	75018	2451	1276	18710	1036	33206	16064	12530
2	4633	84502	1414	3889	20821	1449	36238	17724	13275

Inference: From the above results, it can be observed that neither of the cluster is dominated by instances from a particular Payment Type**.** Thus, we **do not reject the Null Hypothesis <u>Conclusion</u>**

Since all the clusters have roughly similar proportion of instances from each Payment type it can be established that if one trip is paid by cash and other trip is paid by credit card it doesn't mean that those trips are significantly different from each other. This also conforms with Hypothesis 1 results using Random Forest in the previous step. It also means that a model built just for the trips paid by Credit Card should naturally extend to trips paid by Cash as payment type has no effect in distinguishing between the two.

Also, the provided dataset has the value for the variable "Tip_Amount" for all the instances with Payment type Credit Card. Hence, predictive model would be built just on this data and should be robust enough to do predictions for instances with Payment Type "Cash" as well.

Model Building

NOTE: In model building stage variables "Total Amount" & "Tip Amount" are excluded from dataset and then models are trained. There are two reasons for doing the same:

A) For observations in dataset for which payment has been done through cash the value of variable "Tip Amount" is missing. "Tip Amount" significantly contributes to the calculation of variable "Total Amount" which consequently in case of trips paid through cash is having a portion of information missing. Thus building a model on observations paid through credit card including the variable Final amount would not be robust enough to generalize to the samples for which payment is done through cash. As such both the variables: "Tip Amount" & " Total Amount" are excluded from the data.

B) In a real scenario the value of "Total Amount" & "Tip Amount" would not be available until a trip is over. So to predict "Tip Percentage" for an ongoing trip the model trained on data including both the mentioned variables is not applicable.

MODEL BUILDING AFTER EXCLUDING VARIABLE "FINAL AMOUNT" & "Tip Amount" from the dataset

Log transformation was done on the response variable to make the distribution resemble a normal distribution and fit the Regression Model better.

```
## Taking Log transformation of the response variable
final_train_use_model$Tip_Perc<-log(final_train_use_model$Tip_Perc+1)
```

Before going ahead with modeling **remaining outliers were removed using Cooks distance**. The observations for which cooks distance was more than 4 times mean value were removed.

Code Used:

```
mod<-lm(Tip_Perc~.,data=final_train_use)
print("Performance of Multiple Linear Regression Model to predict Tip Percentage for a given trip")
summary(mod)

##Multivariate Outliers
cooksd <- cooks.distance(mod)
hq = 4*mean(cooksd, na.rm=T)
pl<-ifelse(cooksd>hq,F,T)
length(pl[pl==0])
final_train_use<-final_train_use[pl,]
final_train_use_model<-final_train_use_model[pl,]
```

Regression Models Fitted:

a) Random Forest – Non Linear Regression.

b) Multiple Linear Regression Model- Linear Regression.

c) Lasso Regression- Linear Regression with L1 Penalty.

Random Forest Model Building

Random Forest was implemented using the Library: Ranger in R. It is a fast implementation of Random Forests.

Random Forest library was tuned to select the parameters:

"mtry"- the number of variables selected to build model for each tree.

The value of mtry corresponding to least error rate = 4

"Num.tree"- number of trees to be built

The value of num.tree corresponding to least & robust error rate = 150

```
library(ranger)
library(RandomForest)
bestmtry<-tuneRF(final_train_use_model1,final_train_use_model1$Tip_Perc,
                 stepFactor = 1.2,improve = 0.01,plot=TRUE,trace=TRUE)
rf<-ranger(Tip_Perc~.,data=final_train_use_model,mtry=4, num.trees=150,importance = "impurity")
```

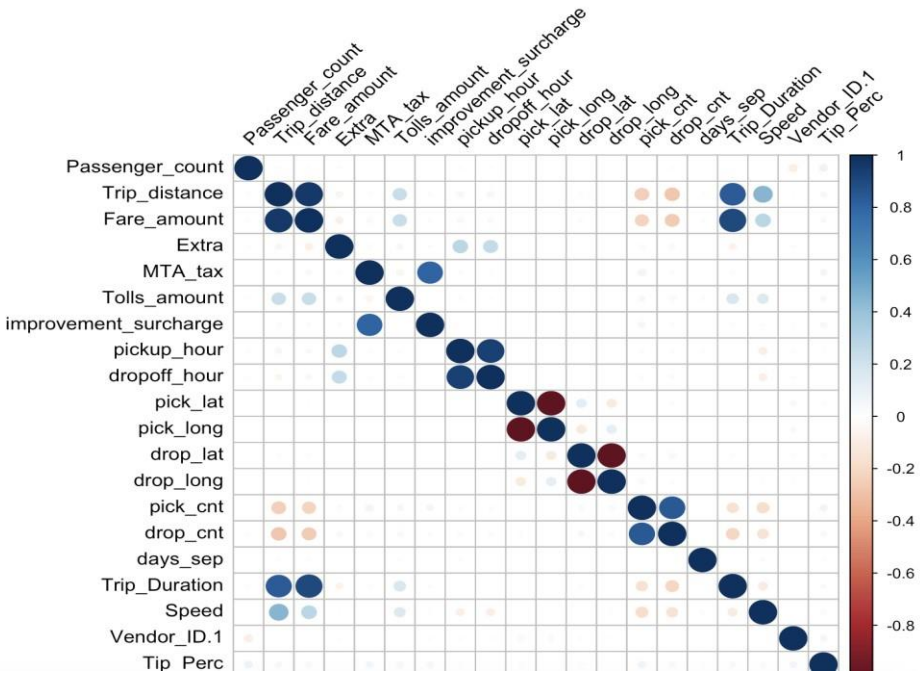
Results: MSE on OOB samples is very high and adjusted R square value is pretty low around 0.083. The model doesn't seem to fit well to the data. The model is saved for giving predictions for test

```
Call:
 ranger(Tip_Perc ~ ., data = final_train_use_model, mtry = 4,      num.trees = 150, importance = "impurity")
                                as.matrix(x, ...)
Type:                             Regression
Number of trees:                  150
Sample size:                      158503
Number of independent variables:  24
Mtry:                             4
Target node size:                 5
Variable importance mode:         impurity
OOB prediction error (MSE):       0.7599283
R squared (OOB):                  0.08356836
```

instances.

saveRDS(rf1,file="RFmodel.RDS")

Multiple Linear Regression



Correlation plot below shows that only few of the variables are strongly correlated to each other. Most of the variables are independent in the data matrix **MLR Model:**

```
mod<-lm(Tip_Perc~.,data=final_train_use)
```

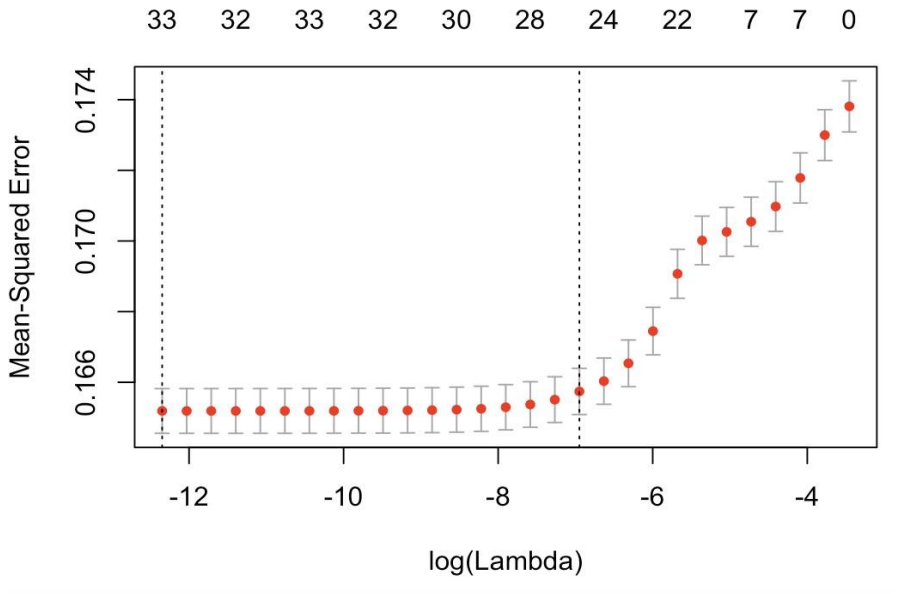
Results:

```
Residual standard error: 0.4064 on 158470 degrees of freedom
Multiple R-squared:  0.04989,   Adjusted R-squared:  0.0497
F-statistic:   260 on 32 and 158470 DF,  p-value: < 2.2e-16
```

The values of R square and adjusted R square are around 0.049 which is very low. The model actually explain only 5% of the variance in the data.

The model is statistically significant as evident from p value<0.05 obtained from the F test. Majority of beta coefficients are statistically significant at 0.05 significance level. Beta value for 24 variables out of 32 is statistically significant at 0.05 level of significance.

LASSO Regression



The model was implemented using GLMNET library in R. The model parameter lambda was tuned using grid search over 10 fold cross validation. The error rate was recorded for different lambda values and the one used to fit the model was the one corresponding to:

lowest error rate +1 standard error

```
Call:  glmnet(x = as.matrix(subset(final_train_use, select = -c(Tip_Perc))),
pha = 1, lambda = CV$lambda.1se)

     Df    %Dev    Lambda
[1,] 27 0.04726 0.0009595
```

Results:

The deviance ratio is again very low for LASSO Regression as well signifying that the model is not ale to explain the variance of the data well enough. The algorithm inspite of putting L1 penalty in the

objective function and constraining the beta coefficients to zero for 6 variables is not able to capture the structure of the data well enough for making accurate predictions.

MODEL BUILDING INCLUDING VARIABLE "FINAL AMOUNT" in the model.

Predictive model was built by training Random Forest on training dataset including variable "Final Amount" and then was tested on the OOB test data having same variables as the training dataset.

NOTE: The code for training this model is "NYC_Train_Final_Amt.R" available in the same folder as this report and also the code for making predictions on new samples is included in same folder as "NYC_Test_Final_Amt.R"

Training Results for Random Forest:

```
Call:
 ranger(Tip_Perc ~ ., data = final_train_use_model, mtry = 4,      num.trees = 150, importance = "impurity")

Type:                             Regression
Number of trees:                  150
Sample size:                      163200
Number of independent variables:  25
Mtry:                             4
Target node size:                 5
Variable importance mode:         impurity
OOB prediction error (MSE):       0.3665969
R squared (OOB):                  0.6222466
```

From the above results it is observed that the model fits significantly better than the Random Forest model previously trained on data with out the variable "Final Amount". The results shown above are pertaining to out of box(OOB) samples i.e observations which are not used to train a particular tree in the forest. Thus, prediction error on OOB samples are similar to the error on test data. Here we see that both OOB prediction error and R squared value is considerably better. The variable "Final Amount" had significant effect in explaining the variance of data and also helps the model to generalize better for the trips paid through Credit Card.

Testing Results:

As expected the model does not generalize better to the trips paid through cash. The MSE for test data is 87.52. We can conclude that model fits comparatively better to data for payment type = Credit card but does not generalize well to data for payment type = Cash

Conclusion

All the three models (excluding variable "Final Amount") used above performs rather badly and none of the model is able to fit the data appropriately. Among the three models Random Forest gives the least error rate on the test set. Random Forest model (including variable "Final Amount") used above performs rather badly on test data. The response variable & predicted variable were transformed to normal scale for calculating MSE.

Code Used:

```
rf3<-readRDS("RFmodel.RDS")
t<-predict(rf3,final_test_use_model)
p<-log(final_test_use_model$Tip_Perc+1)
ms<-(sum((t$predictions-p)^2))/length(t$predictions)
print("Performance of Random FOrest on Test Set")
ms
```

MSE of Random Forest on Test Set is 79.84546

Step 5:

Distributions:

- Build a derived variable representing the average speed over the course of a trip.
- 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?
- Can you build up a hypothesis of average trip speed as a function of time of day?

a) Build a derived variable representing the average speed over the course of a trip.

Solution: The Speed variable was derived in the data preparation step itself. The logic used was:

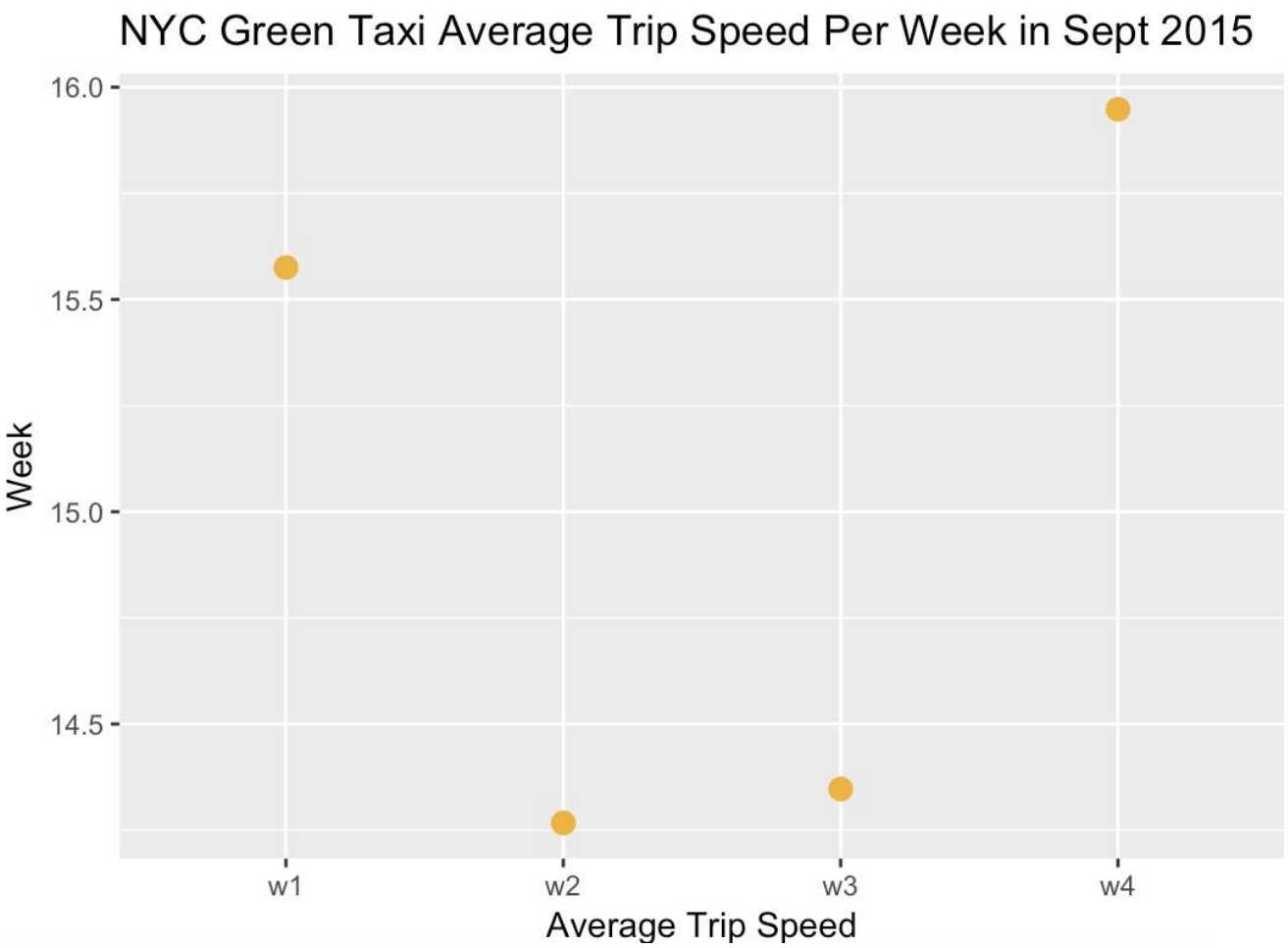
Speed(miles per minutes)=Trip_Distance(miles)/Trip_duration(minutes).

Code Used:

```
train.data$Speed<-train.data$Trip_distance/train.data$Trip_Duration
```

b) 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?

Solution: Looking at the plot below there seems to be significant differences in average speed for majority of the weeks compared in pairs.



Test Use to determine if the average trip speeds are materially the same in all weeks of September:
One-Way Anova

One-way Anova was conducted to compare the average trip speeds pertaining to different weeks in September. The p value obtained was much lesser than 0.05 indicating that average trip speeds pertaining to different weeks in September are indeed different at 0.05 significance level.

Results:

```
                Df   Sum Sq Mean Sq F value   Pr(>F)
week             3      222    74.1   6.443 0.000234 ***
Residuals  1461050 16802964    11.5
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Further Investigation:

As the ANOVA test is significant, **Tukey HSD** test was used for performing multiple pairwise-comparison between the means of groups.

Results:

```
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = Speed ~ week, data = d)

$week
               diff          lwr           upr     p adj
w2-w1 -0.021806658 -0.042253150 -1.360167e-03 0.0312465
w3-w1 -0.020474763 -0.040869043 -8.048257e-05 0.0486694
w4-w1  0.006211046 -0.013850427  2.627252e-02 0.8565779
w3-w2  0.001331896 -0.019396359  2.206015e-02 0.9984008
w4-w2  0.028017705  0.007616807  4.841860e-02 0.0023638
w4-w3  0.026685809  0.006337240  4.703438e-02 0.0041986
```

Inference

From the pairwise results it was observed that:

Average speed does not differ significantly for week4 & week1 and also for week3 & week2 at 0.05 significance level.

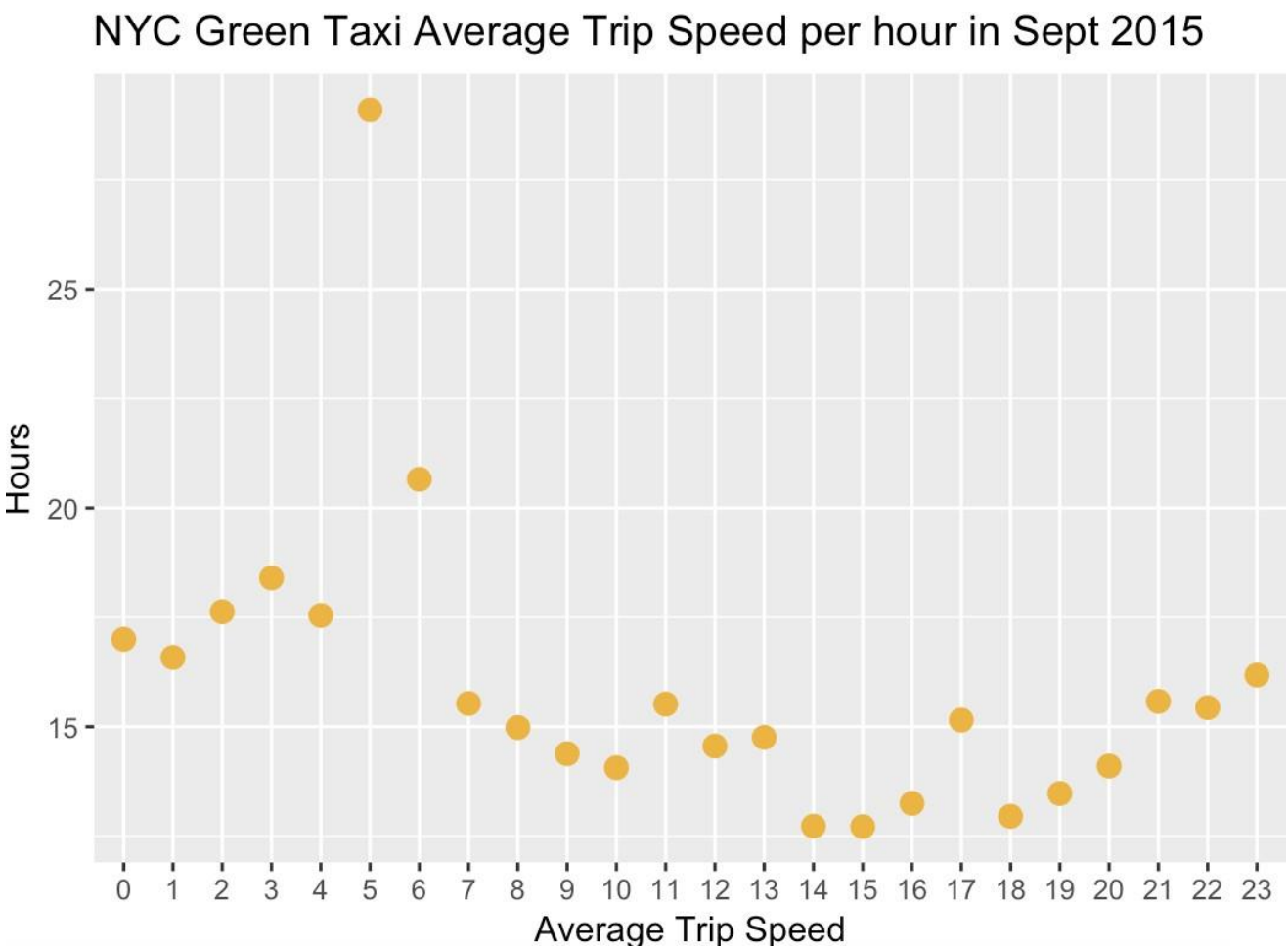
However, average speed does differ significantly for rest of the week pairs i.e week2 & week1, week3 & week1, week4 & week2, week4 & week3 at 0.05 significance level.

1) 2)

Code Used:

```
d %>%
  group_by(week) %>%
  summarise(av_speed=mean(Speed*60))%>%
  ggplot(aes(x = as.factor(week), y = av_speed)) +
  geom_point(colour = "#F5B317", size = 3)+
  ggtitle("NYC Green Taxi Average Trip Speed Per Week in Sept 2015") +
  xlab("Average Trip Speed") +
  ylab("Week")

res.aov <- aov(Speed ~ week, data = d)
summary(res.aov)
TukeyHSD(res.aov)
```

c) Can you build up a hypothesis of average trip speed as a function of time of day?



The aggregated average speed per hour for Sept 2015 is 15.9 mph.

Looking at the above plot following three hypothesis were formed:

Hypothesis1: *Average speed significantly differs for different hours of the day* **Hypothesis2:** *Traffic is faster early morning as compared to evenings.*
Hypothesis3: *Average speed is different for different phases of the day i.e. Early morning (2-7 hours), Morning(7-11 hours), Afternoon(11-17 hours), Evening(17-21), Night(22-24 hours)*

Code Used:

```
df2<-train.data%>%group_by(as.factor(pickup_hour))%>%summarise(av_speed=mean(Speed*60))
train.data %>%
  group_by(pickup_hour=as.factor(pickup_hour)) %>%
  summarise(av_speed=mean(Speed*60))%>%
  ggplot(aes(x = pickup_hour, y = av_speed)) +
  geom_point(colour = "#F5B317", size = 3)+
  ggtitle("NYC Green Taxi Average Trip Speed per hour in Sept 2015") +
  xlab("Average Trip Speed") +
  ylab("Hours")
```

Future Work:

Given more time following are list of things I would want to work upon further:

A) The performance of all three models are below par. Given time I would like to build multiple predictive models like- Gradient Boosting Models, SVM, Neural Network and stack these models along with Random Forest to build an ensemble. This could help both in model fitting and reducing the variance in prediction.

B) The main challenge of the problem is getting the right features having good predictive power and association with the response variable. The dataset does not include any variable which describe the experience of the passenger in a particular trip which plays an important part for passenger to decide on the tip amount to be given. So, I would want to extract features pertaining to customer rating and also customer complaints and then build predictive models. Also, I would focus on deriving more features from the existing dataset. A number of features like: pick up clusters, drop off clusters, intra and inter-borough traffic, popularity of different clusters could be included for model building.

C) While clustering observations the only method used in this study is Gaussian Mixture Model. In future, I would recommend using clustering techniques like Spectral Clustering & DBSCAN. Spectral Clustering has the ability to learn the manifold structure of the data as such if the data is non linear spectral clustering could give clusters totally different from the ones given by GMM. Also, different distance measures like- Gower distance & manhattan distance needs to be used and tested.

D) In the future, I would also recommend to work upon outlier detection. One of the easy ways would be use the clusters formed by GMM in this study and use distance measures to detect outliers. For robust outlier detection I would suggest to combine different methods of multivariate outlier detection like: Clustering using GMM, Local Outlier Factor, Isolation Forests.