Capital One Data Challenge

Xinyan Yang (xinyany2@illinois.edu)

February 26, 2018

Note about this report:

To make the report concise, some code are hidden, especially those code with respect to plotting.

First let's load the data.

```
#load data
full_data = read_csv("green_tripdata_2015-09.csv")
```

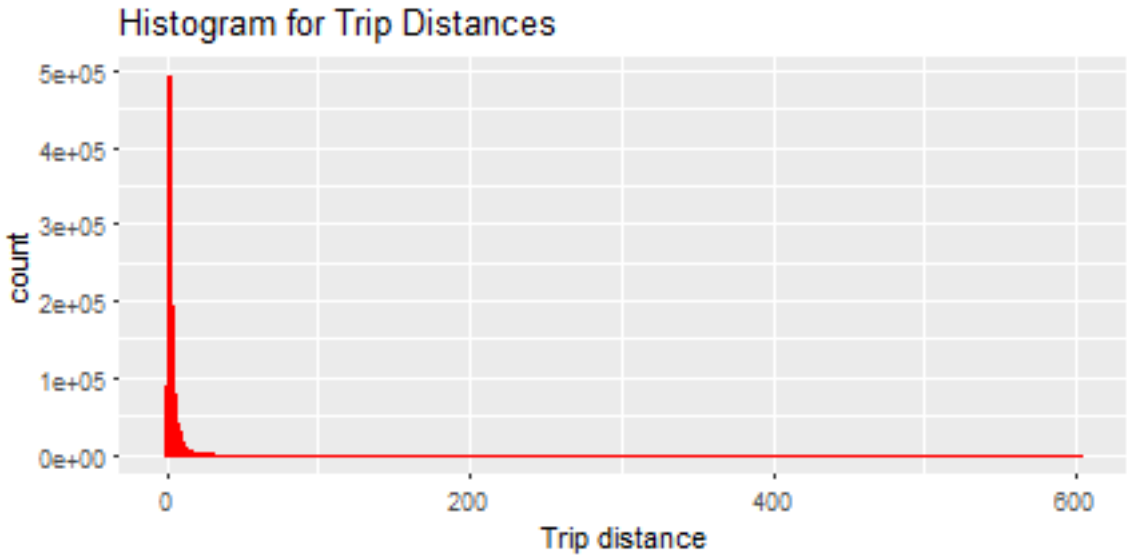
Question 1 Load data

```
#obtain the dimension of original data
dim(full_data)
```

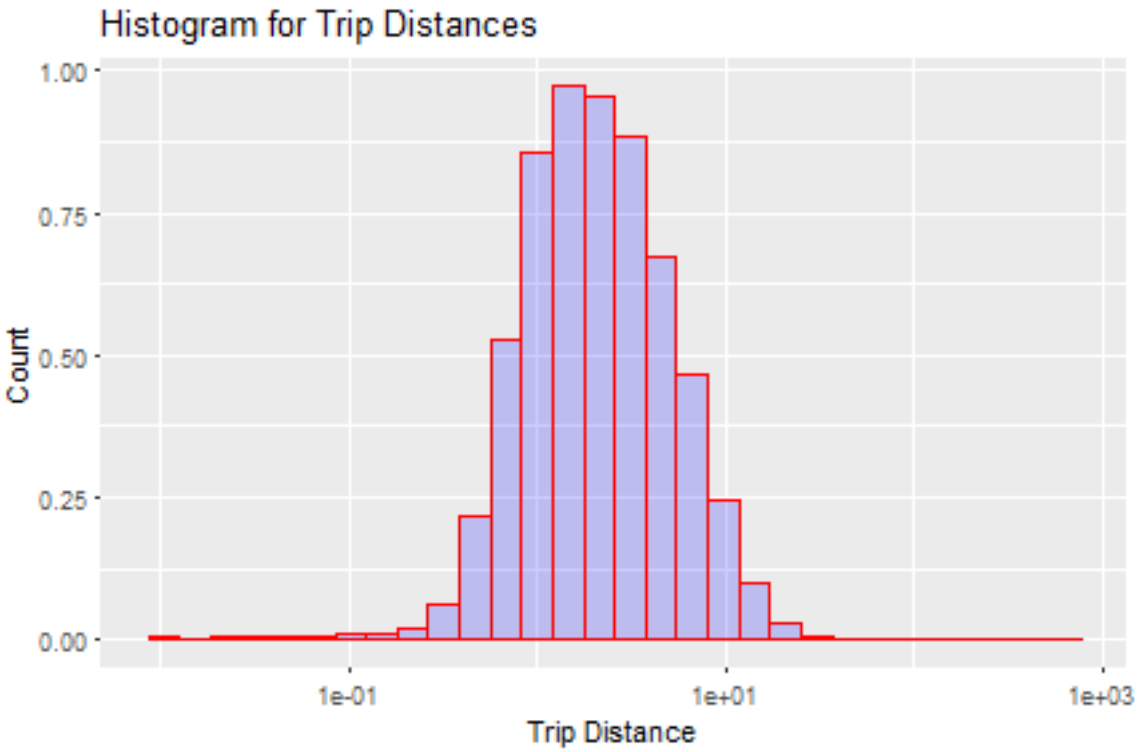
```
## [1] 1494926      21
```

We can see that the original dataset has in total 1,494,926 rows and 21 columns.

Question 2 Histogram of Trip Distance



The above histogram shows that the trip_distance has a right-skewed distribution. We can see that some records have such a long distance that we cannot see the pattern at samll trip distances clearly. Therefore I choose to plot it on a log scale.



Note that x-axis is logarithmic and y-axis is density. We can get the following findings.

1. Under the log scale, trip distance has a more normal-like distribution with most cases fall in the group of 1-10 mile. **2.** Outside of the majority group, there are more short-distance rides than long-distance rides.

Question 3

a) Report mean and median trip distance grouped by hour of day.

Here I define the hour of day based on the pickup time instead of dropoff time.

```
#compute the hour of day
full_data["hour_of_day"] = hour(full_data$lpep_pickup_datetime)
```

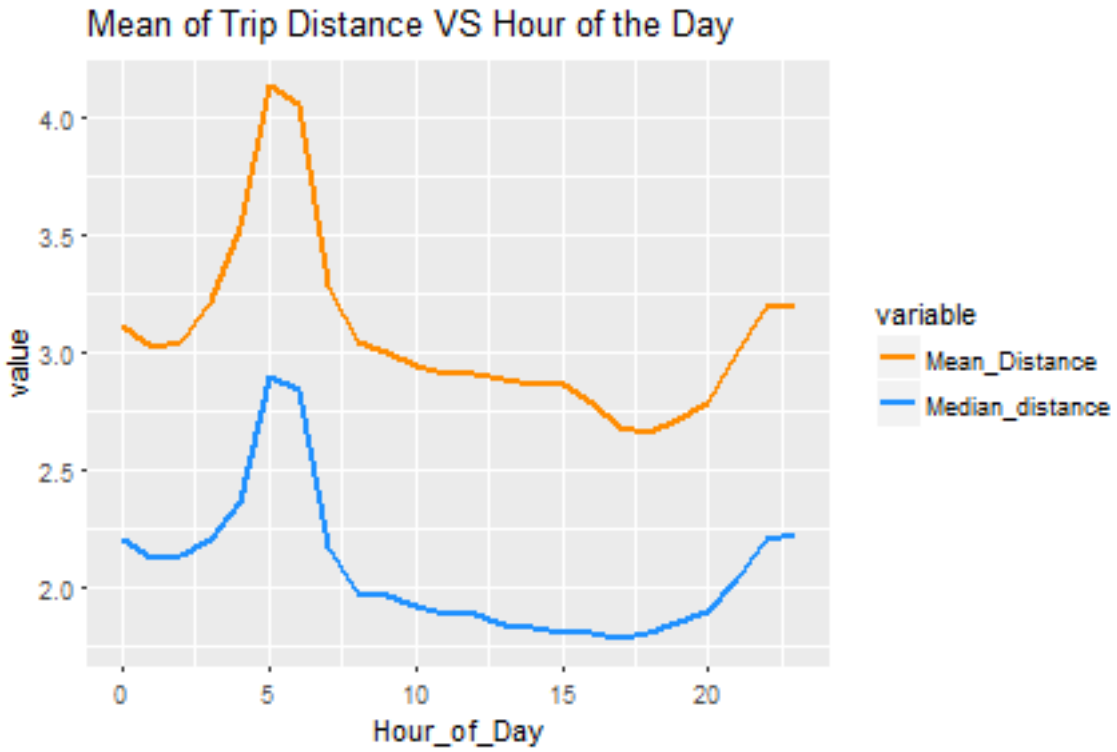
```
mean_calc = aggregate(full_data[, 11], by = list(full_data$hour_of_day), FUN = mean)
median_calc = aggregate(full_data[, 11], by = list(full_data$hour_of_day), FUN = median)
table_distance = merge(mean_calc, median_calc, by = 'Group.1')
```

```
colnames(table_distance) = c("Hour_of_Day", "Mean_Distance", "Median_distance")
knitr::kable(round(table_distance, 2))
```

Hour_of_Day	Mean_Distance	Median_distance
0	3.12	2.20
1	3.02	2.12
2	3.05	2.14
3	3.21	2.20
4	3.53	2.36
5	4.13	2.90

Hour_of_Day	Mean_Distance	Median_distance
6	4.06	2.84
7	3.28	2.17
8	3.05	1.98
9	3.00	1.96
10	2.94	1.92
11	2.91	1.88
12	2.90	1.89
13	2.88	1.84
14	2.86	1.83
15	2.86	1.81
16	2.78	1.80
17	2.68	1.78
18	2.65	1.80
19	2.72	1.85
20	2.78	1.90
21	3.00	2.03
22	3.19	2.20
23	3.19	2.22

To help understand the table above, I plot the relationship between mean distance, median distance and hour of the day on a scatter plot.



From this plot, we can see that the peak of trip distance appears at 5 AM. Then it keeps decreasing until 6 PM. For this phenomenon, I would hypothesize that on the early morning, either there are less public transportation in further areas or people from further areas tend to take taxis to avoid being late for work, school or important appointments.

b) Provide a count, the average fare of how many transactions originate or terminate at one of the NYC area airports.

To identify the trips that originate or terminate at one of the NYC area airports, we need to know the coordinates of these two airports. We could get the latitude and logitude information from Google. Then we calculate the distance between both pickup location and dropoff location with two airports using distCosine function. We then can define a trip related to airport if any of these two distances less than 2,000 meters.

```
#define the location of two airports
jfk_coord = tibble(lon = -73.7781, lat = 40.6413)
laguardia_coord = tibble(lon = -73.8740, lat = 40.7769)

#calculate the distance between pickup location, dropoff location and two airports
# the distanece is in meters
pick_coord = tibble(full_data$Pickup_longitude, full_data$Pickup_latitude)
drop_coord = tibble(full_data$Dropoff_longitude, full_data$Dropoff_latitude)

jfk_pick_dist = distCosine(pick_coord, jfk_coord)
jfk_drop_dist = distCosine(drop_coord, jfk_coord)
laguardia_pick_dist = distCosine(pick_coord, laguardia_coord)
laguardia_drop_dist = distCosine(drop_coord, laguardia_coord)

# define the trip as an airport trip if the distance is within 2,000 meters
full_data["jfk_ride"] = (jfk_pick_dist <= 2000) | (jfk_drop_dist <= 2000)
full_data["laguardia_ride"] = (laguardia_pick_dist <= 2000) | (laguardia_drop_dist <= 2000)
```

```
#sum the total number of airport rides
sum(full_data$jfk_ride)
```

```
## [1] 13053
```

```
sum(full_data$laguardia_ride)
```

```
## [1] 37140
```

As you can see, if we use 2,000 meters as the radius, there would be 13053 trips related to JFK airport and 37140 trips related to LaGuardia airport.

```
#compute the average fare of these airport rides
mean(full_data[which(full_data$jfk_ride), ]$Fare_amount)
```

```
## [1] 41.41232
```

```
mean(full_data[which(full_data$laguardia_ride), ]$Fare_amount)
```

```
## [1] 16.9119
```

The average fare related to JFK airport trips is 41.41$ and the average fare related to LaGuardia airport trips is 16.919.

The result is reasonable since JFK airport is far away from the Manhattan area, where most trips start or end up there. At the same time, with a close distance to Manhattan area, LaGuardia airport have almost 3 times trips than JFK airport, and they usually have a smaller fare.

Question 4

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

First we need to remove those observations with total_amount equals to 0 since it would make the computation of tip ratio become NA.

```
total_nonzero = (full_data$Total_amount != 0)
full_data = full_data[which(total_nonzero), ]
full_data["tip_ratio"] = full_data$Tip_amount / full_data$Total_amount
```

b) Build a predictive model for tip as a percentage of the total fare.

Next we would do the following steps to complete our final model.

1) Data Preprocessing: Remove the missing values **2) Feature Engineering**: Define relevant features **3) Model Selection and Validation**: Perform model selection and tunning **4) Prediction and Conclusions**: Use final model to do the prediction and give some suggestions or conclusins based on the result.

The First Step: Data Preprocessing

Let's check if our dataset has any missing values

```
#check if there are any missing values
sum(is.na(full_data))
```

```
## [1] 1490758
```

```
sum(is.na(full_data[, -17]))
```

```
## [1] 4
```

```
#create a new dataset excluding the missing values
newdata = na.omit(full_data[, -17])
sum(is.na(newdata))
```

```
## [1] 0
```

We find that the total column of Ehail-free are NAs, so we remove this variable and check again, this time we have 4 NA value in total. Since we have a fairly large dataset, I simply create a new dataset without these 4 missing records. And we would use this *newdata* to continue our analysis.

The Second Step: Feature Engineering

This step is important since it directly decides the result of our final model.

I first check the correlation matrix between tip_ratio and other variables in the current dataset and find that most variables have a small linear relationship with our target variable tip_ratio except **Payment_type** and **Total_amount**.

Then I create some **time variables** which may related to the target variable or not.

New variables: Day_of_month, Day_of_week, Trip_duration, Speed.

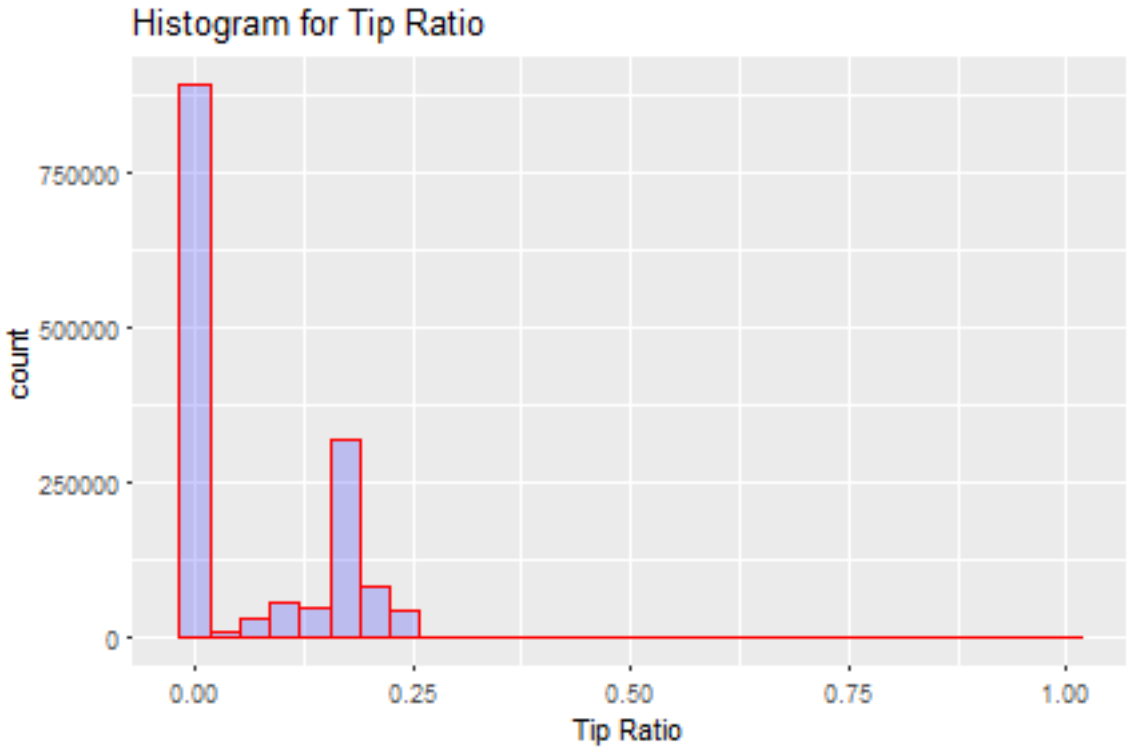
```
#all time variables would use the pickup datetime
# Create Days_of_month variable
newdata["day_of_month"] = day(newdata$lpep_pickup_datetime)
#create day_of_week variable
newdata["day_of_week"] = weekdays(newdata$lpep_pickup_datetime)
#create trip_duration variable, it's measured in minutes
```

```
newdata["Trip_duration"] = as.numeric((newdata$Lpep_dropoff_datetime - newdata$lpep_pickup_datetime)/60)
#create speed variable using trip_distance/trip_duration
newdata["speed"] = newdata$Trip_distance / newdata$Trip_duration
```

Until here, we have some variables which can be put into model, they are: *Payment_type, Fare_amount, hour_of_day, airtrip, day_of_month, day_of_week, trip_duration, airtrip and speed.*

Next we should explore the relationship between these variables and our target variable to decide whether it would be appropriate to throw them in our model and which type of model should we build.

Let's **check the distribution of target variable tip_ratio**:




```
sum(newdata$tip_ratio == 0)
```

```
## [1] 887980
```

This histogram shows that more than half of the trips have no tip, and this skewed distribution would definitely cause some influence on the model if we directly build a regression model. Therefore I would try the following steps to build our final model:

1. Build a classification model to identify if the trip would have a tip(1) or not(0). 2. If we get 0 in step 1, the tip_ratio would be 0. If we get 1 in step 1, predict the value of tip_ratio using a regression model.

Let's build a binary variable to indicate whether the trip has tip or not.

```
newdata["tip_or_not"] = ifelse(newdata$tip_ratio == 0, 0, 1)
```

Then we can **check the relationship between payment_type and tip_ratio.** We have in total 6 different types of payment.

```
sum(newdata[newdata$tip_or_not == 1, ]$Payment_type == 1)
```

```
## [1] 602730
```

```
sum(newdata[newdata$tip_or_not == 1, ]$Payment_type == 2)
```

```
## [1] 2
```

```
sum(newdata[newdata$tip_or_not == 1, ]$Payment_type == 3)
```

```
## [1] 35
```

```
sum(newdata[newdata$tip_or_not == 1, ]$Payment_type == 4)
```

```
## [1] 3
```

```
sum(newdata[newdata$tip_or_not == 1, ]$Payment_type == 5)
```

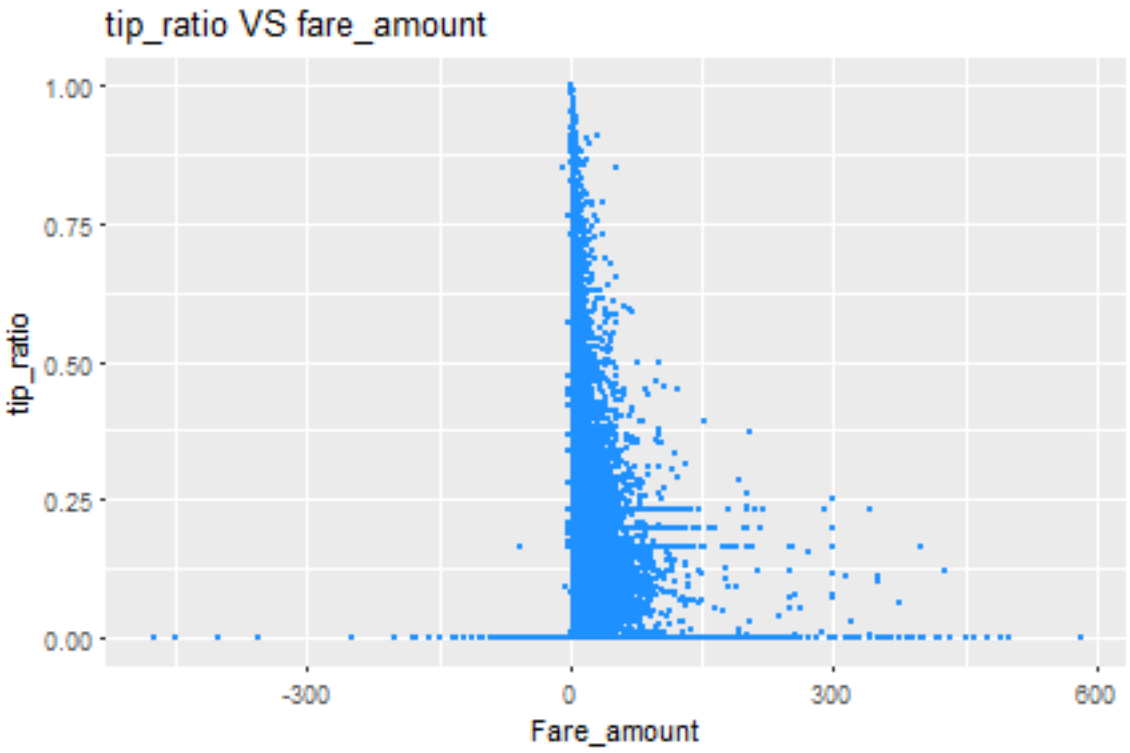
```
## [1] 0
```

```
sum(newdata[newdata$tip_or_not == 1, ]$Payment_type == 6)
```

```
## [1] 0
```

As you can see, almost all of them used payment_type 1, which refers to the credit card. Since this variable has such unbalanced variance, there's no sense to put it in the regression model. However, it can be put into the classification model.

Then we can **check the relationship between fare_amount and tip_ratio.**



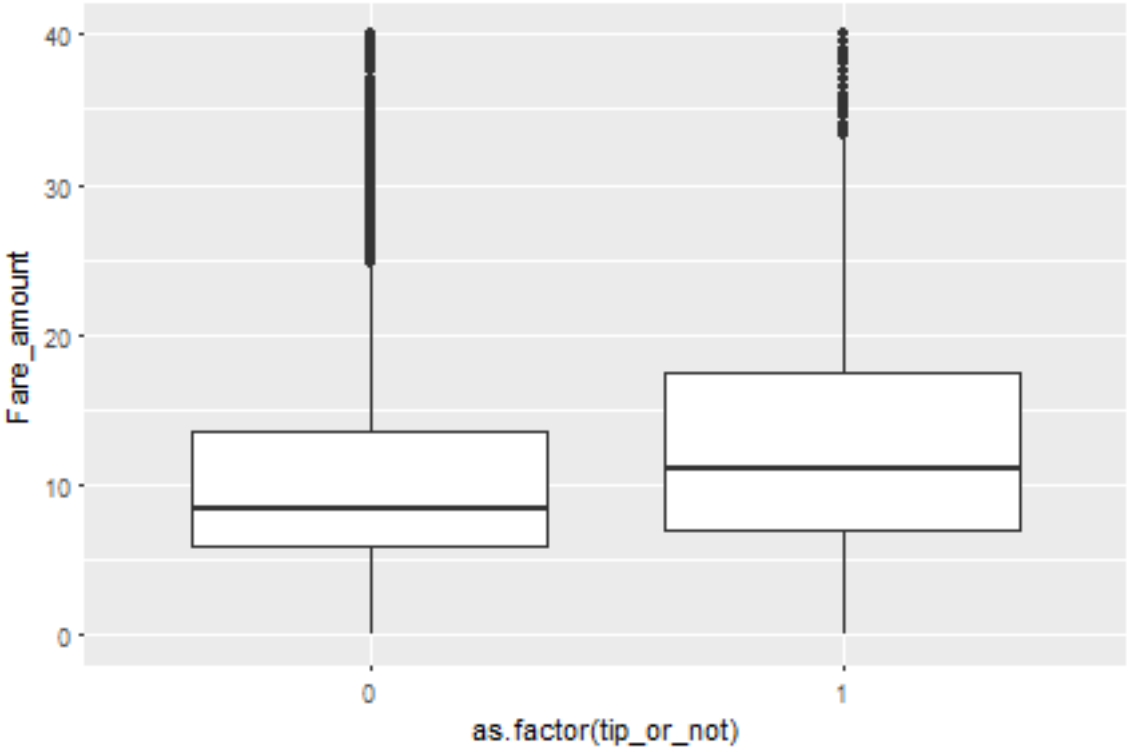

```
sum(newdata$Fare_amount <= 0)
```

```
## [1] 2713
```

Here we notice a strange phenomenon that there are actually 2713 records in our dataset having a negative or zero fare amount. These points may be outliers but I actually have no idea why it happens, but I would prefer to delete them from the dataset.

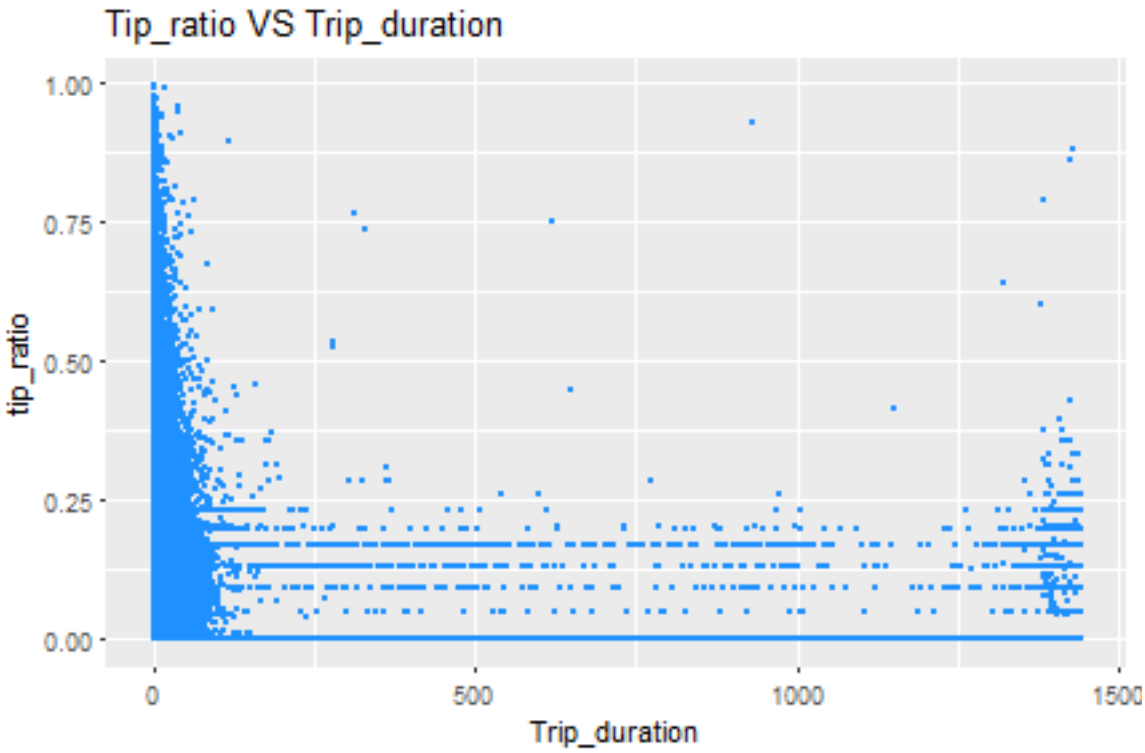
```
newdata = newdata[which(newdata$Fare_amount > 0), ]
```

Let's see if there are any difference in fare_amount between tip group and non-tip group.

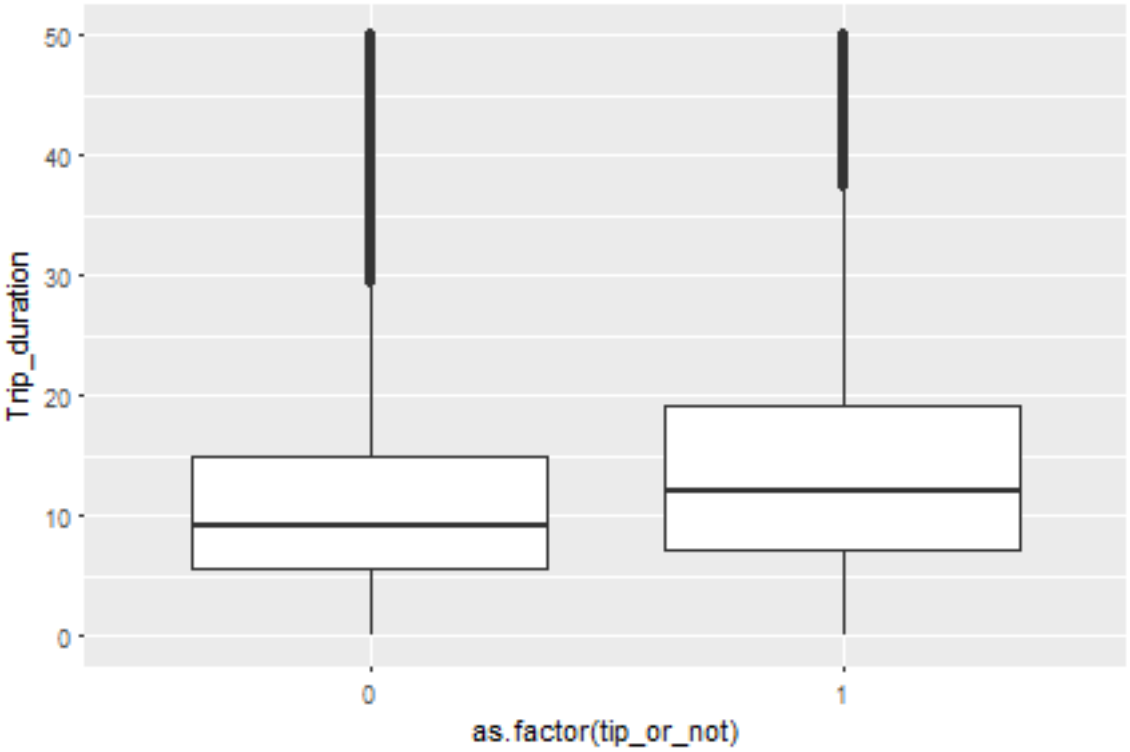


We can see that the tip group has a higher 1st quantile, median value, 3rd quantile than the non-tip group on the fare_amount variable. Therefore we would consider put it into our classification model.

Next let's check the **relationship between trip-duration and tip__ratio.**

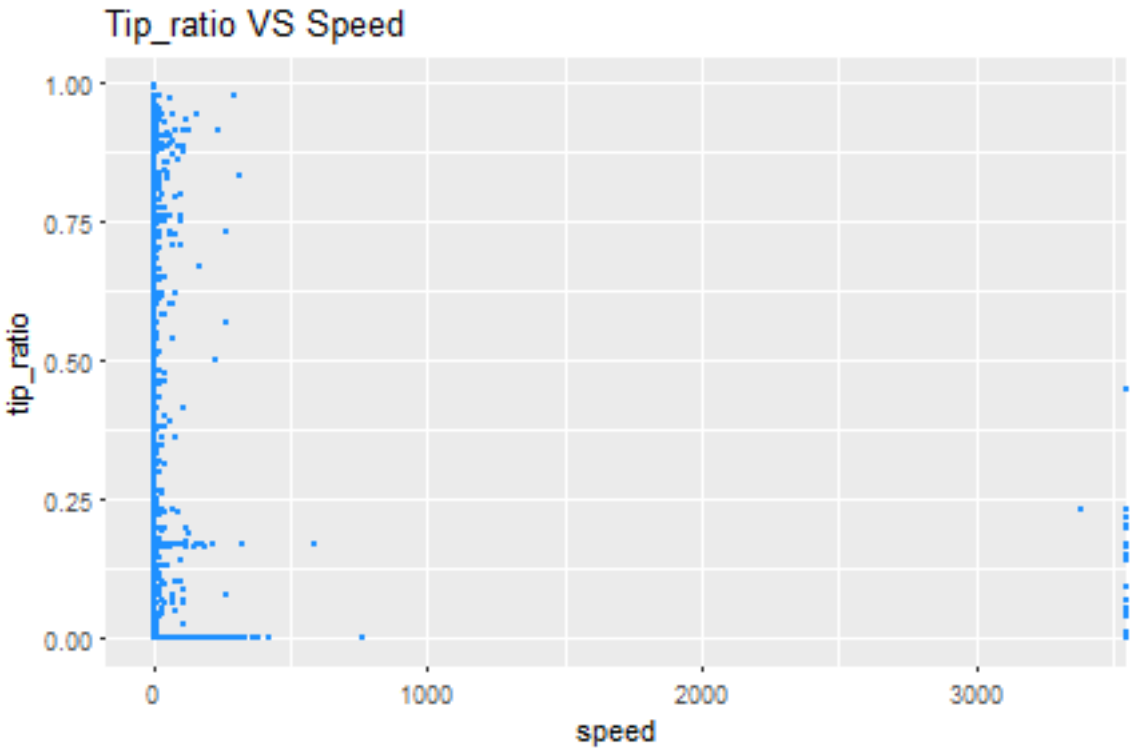


It's weird that there are many trips with duration more than 15 hours(which means more than 900 minutes in this plot) in NYC. However, we choose to keep these instances since there's a little increasing trend when trip_duration are more than 1250 minutes.



This plot still shows that the tip group has a higher 1st quantile, median value, 3rd quantile than the non-tip

group on the trip_duration variable. Therefore we would consider put it into our classification model Finally let's check **the relationship between speed and tip_ratio.**



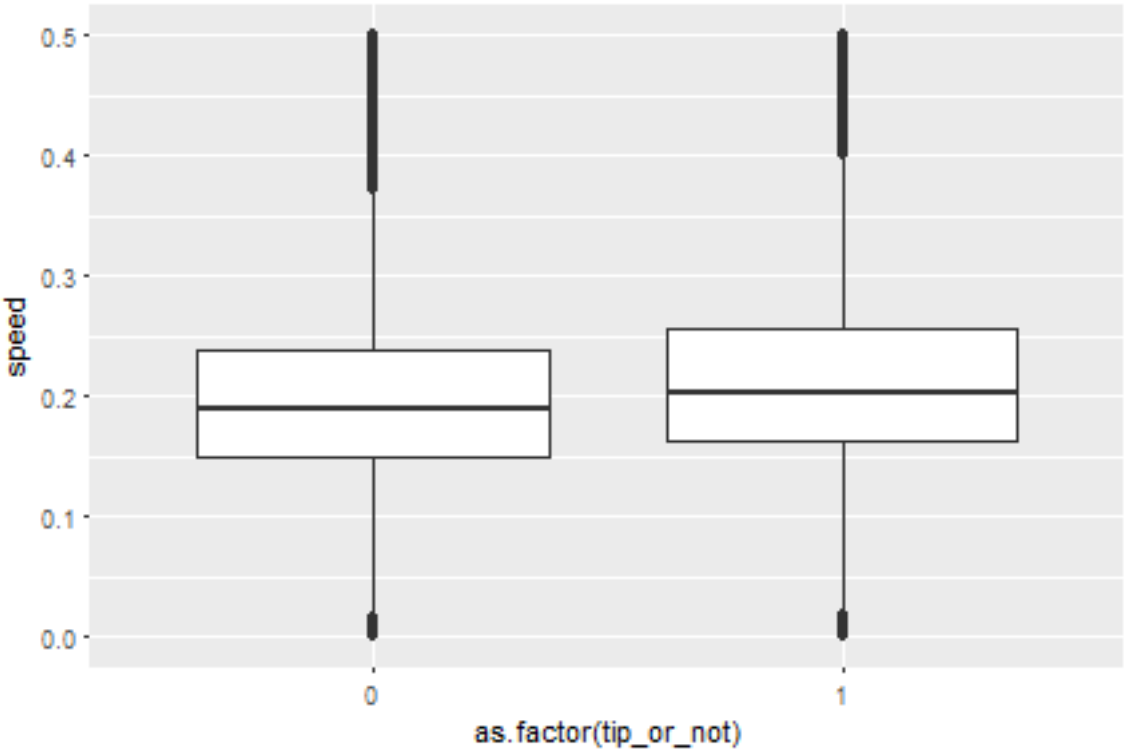

```
summary(newdata$speed)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
##  0.0000  0.1554  0.1962     Inf  0.2512     Inf     855
```

This plot apparently indicates that there are some outliers in the dataset, since the maximum value of speed reaches infinity and also, there are 855 NAs in the speed variable. According to some common sense, I choose to mannully delete those observations with speed greater than 2.5 miles per minute or NAs.

```
newdata = newdata[which(newdata$speed < 2.5),]
newdata = na.omit(newdata)
```

Then we can check the difference of speed between tip group and non-tip group.



The plot suggests that the tip group has a higher 1st quantile, median value, 3rd quantile than the non-tip group on the speed variable. Therefore we would consider put it into our classification model.

Conclusion:

After all this analysis above, we choose the following variables to put into our model.

Classification model:

Features: Payment_type, Fare_amount, hour_of_day, airtrip, day_of_month, day_of_week, trip_duration and speed

Response: tip_or_not(binary variable)

Regression model:

Features: Fare_amount, hour_of_day, airtrip, day_of_month, day_of_week, trip_duration, airtrip and speed

Response: tip_ratio

The Third Step: Model Selection and Validation

A few notes before we tune the model:

1. We randomly choose 40,000 cases to be our training dataset and 40,000 cases to be our test dataset since the model accuracy is good enough under this size. If we include more cases, the improvement would be samll compared to the large computation cost.
2. We use 5-fold cross-validation to be our resampling method to improve the stability of model.
3. For the limitation of time, we use random forest to train the model.

```
#Randomly choose data as our training set and test set
set.seed(1000)
newdata$day_of_week = as.factor(newdata$day_of_week)
```

```
newdata$Payment_type = as.factor(newdata$Payment_type)
train_index = sample(nrow(newdata), size = 40000)
trn = newdata[train_index, ]
tst_index = sample(nrow(newdata[-train_index, ]), size = 40000)
tst = newdata[-train_index,][tst_index, ]
```

```
class_trn = trn[, c(19, 12, 21, 26:29)]
class_trn$tip_or_not = as.factor(class_trn$tip_or_not)
class_tst = tst[, c(19, 12, 21, 26:29)]
class_tst$tip_or_not = as.factor(class_tst$tip_or_not)
```

Classification model

```
set.seed(1000)
rf_class = train(
    form = tip_or_not ~ .,
    data = class_trn,
    trControl = trainControl(method = "cv", number = 5),
    method = "rf",
    verbose = FALSE,
    tuneGrid = expand.grid(.mtry = 3),
    importance = TRUE
)
```

```
rf_class
```

```
## Random Forest
##
## 40000 samples
##     6 predictor
##     2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 32000, 32000, 32000, 32000, 32000
## Resampling results:
##
##   Accuracy  Kappa
##   0.93445   0.8674139
##
## Tuning parameter 'mtry' was held constant at a value of 3
```

The tunning parameter in random forest with caret package would be mtry, which means the number of variables to possibly split at in each node. We can see that the best mtry would be 3 with a cross-validation accuracy of 93.5%.

Let's see how it performs on the test dataset.

```
class.predict = predict(rf_class, newdata = class_tst[, -7])
class.accuracy = mean(class.predict == class_tst$tip_or_not)
class.accuracy
```

```
## [1] 0.9344
```

The classification model has an accuracy of 0.9344, which is nearly the same as the CV score. Let's see the feature importance of this classification model.

```
varImp(rf_class)
```

```
## rf variable importance
##
##                       Importance
## Payment_type2            100.0000
## Payment_type4             23.1416
## Payment_type3             22.0351
## speed                      4.8166
## Trip_duration              3.6889
## Payment_type5              3.2092
## hour_of_day                2.7592
## Fare_amount                2.6210
## day_of_weekWednesday       0.2923
## day_of_weekTuesday         0.2870
## day_of_weekMonday          0.2011
## day_of_weekSunday          0.1258
## day_of_weekSaturday        0.1079
## day_of_weekThursday        0.0000
```

This variance importance plot indicates that the most important variable for classification model is payment_type.

Regression model

Next we will still use random forest to build our regression model. Note that in this case, all the traning samples are those have a tip_amount > 0.

```
set.seed(1000)
rf_reg = train(
    form = tip_ratio ~ .,
    data = reg_trn,
    trControl = trainControl(method = "cv", number = 5),
    method = "rf",
    verbose = FALSE,
    tuneGrid = expand.grid(.mtry = 2),
    importance = TRUE
)
```

```
#define the root mean squared error function to compute the accuracy
calc_rmse = function(pre_mod, actual){
    sqrt(mean((pre_mod - actual)^2))
}
```

Use this regression model to predict the tip

```
reg.predict = predict(rf_reg, newdata = reg_tst[, -6])
reg.rmse = calc_rmse(reg.predict, reg_tst$tip_ratio)
reg.rmse
```

```
## [1] 0.05188736
```

We can see that the rmse of testing dataset is 0.0518874.

Let's check the feature importance of this regression model:

```
varImp(rf_reg)
```

```
## rf variable importance
```

```
##
##                       Overall
## Fare_amount          100.000
## Trip_duration         72.452
## speed                 65.988
## hour_of_day           29.654
## day_of_weekWednesday  29.289
## day_of_weekSaturday   14.650
## day_of_weekSunday     13.099
## day_of_weekThursday    4.817
## day_of_weekTuesday     1.260
## day_of_weekMonday      0.000
```

As we can see, the most three important variables for this regression model would be: Fare_amount, speed, Trip_duration.

The final Step: Prediction and Conclusions

In this final step, we will combine our regression model and classification model into a function to predict on the complete dataset.

Note: If you want to use the following function to predict another dataset, make sure you do two things before you use the function.

1. Make sure the dataset has the exactly same structure as the original dataset, I will remove the tip_amount variable before predicting.
2. Make sure you have run all the data structure related chunks above and build the rf_class and rf_reg model.

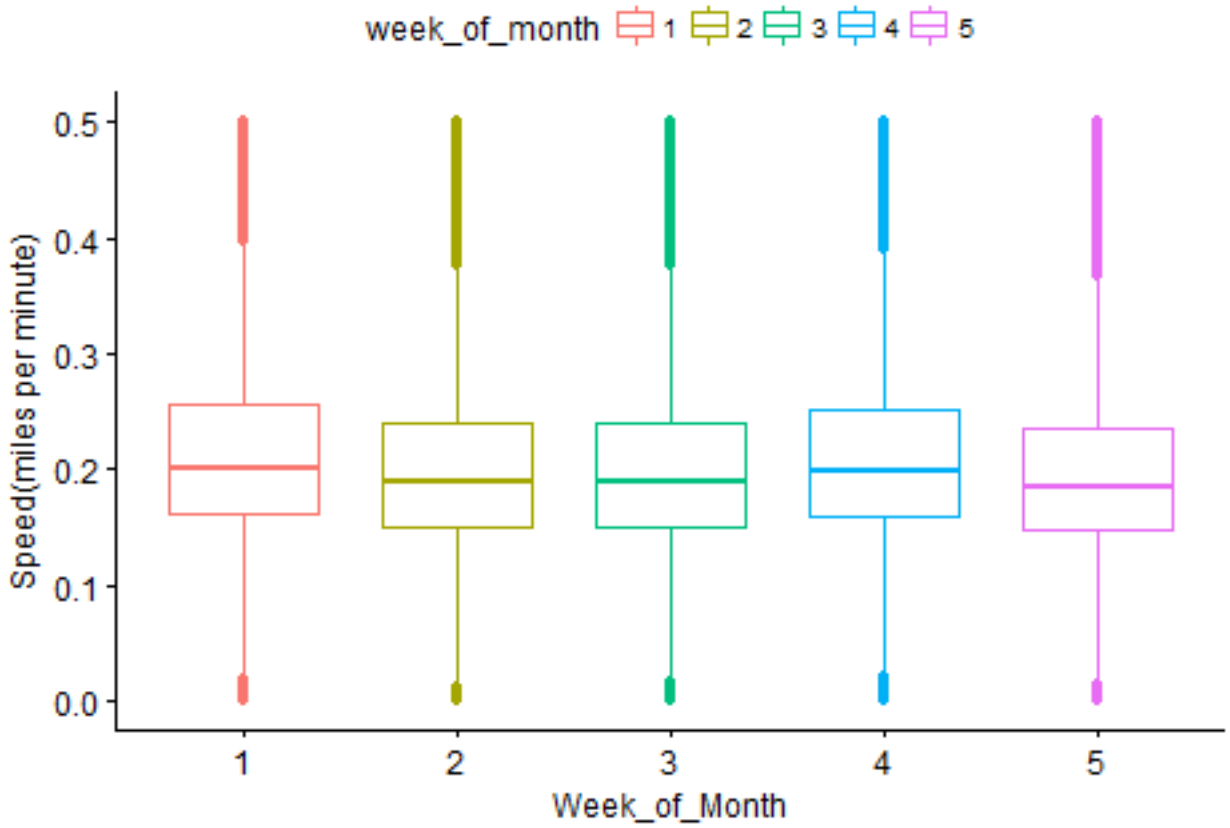
```
predict_tip = function(data){
    #classification model
    #first transform the data to have the same variables as our training model
    predict_tip_ratio = rep(0, nrow(data))

    data["hour_of_day"] = hour(data$lpep_pickup_datetime)
    data["tip_ratio"] = data$Tip_amount / data$Total_amount
    data["day_of_week"] = as.factor(weekdays(data$lpep_pickup_datetime))
    data["Trip_duration"] = as.numeric((data$Lpep_dropoff_datetime -    data$lpep_pickup_datetime)/60)
    data["speed"] = data$Trip_distance / data$Trip_duration

    class_data = data[, c("Payment_type", "Fare_amount","hour_of_day","day_of_week","Trip_duration","sp
    reg_data = data[, c("Fare_amount","hour_of_day","day_of_week","Trip_duration","speed")]
    predict_tip_ratio = predict(rf_class, newdata = class_data)

    #if the classification result is 1, we will continue use the regression model to predict the tip_ra
    for (k in 1:nrow(data)) {
        if (predict_tip_ratio[k] == 1) {
           predict_tip_ratio[k] == predict(rf_reg, newdata = reg_data[k, ])
        }
    }
    return(predict_tip_ratio)
}
```

Question 5 Option 1: Distributin

As we've done in the previous question, some trips have a really large average speed. However, according to the New York State Vehicle and Traffic Law (V&T Law), the vehicles in the state of New York are allowed to have a maximum speed of 65 MPH at certain highways and 55 MPH at all others.We mannully select the records with an average speed less than 2.5 miles per minute and I believe this selection rule can cover almost all trips with a speed they can reach. In other words, any trips with an average speed more than 2.5 miles per hour can be viewed as outliers and we won't consider these points in our following analysis.

a) Perform a One-way ANOVA test to compare the average speed of different weeks of September.

```
#extract the week of month
newdata["week_of_month"] = ceiling(day(newdata$lpep_pickup_datetime) / 7)
#transform the variable to be a categorical variable
newdata$week_of_month = as.factor(newdata$week_of_month)
```



From this plot, we can see some differences between the speed of different weeks of Spetember. Generally, the speed of first and the fourth week would have a slightly higher speed than the other weeks.

```
#perform one-way ANOVA to compare the average speed of different weeks of September
aov_mod = aov(speed ~ week_of_month, data = newdata)
# Summary of the analysis
summary(aov_mod)
```

```
##                   Df Sum Sq Mean Sq F value Pr(>F)
```

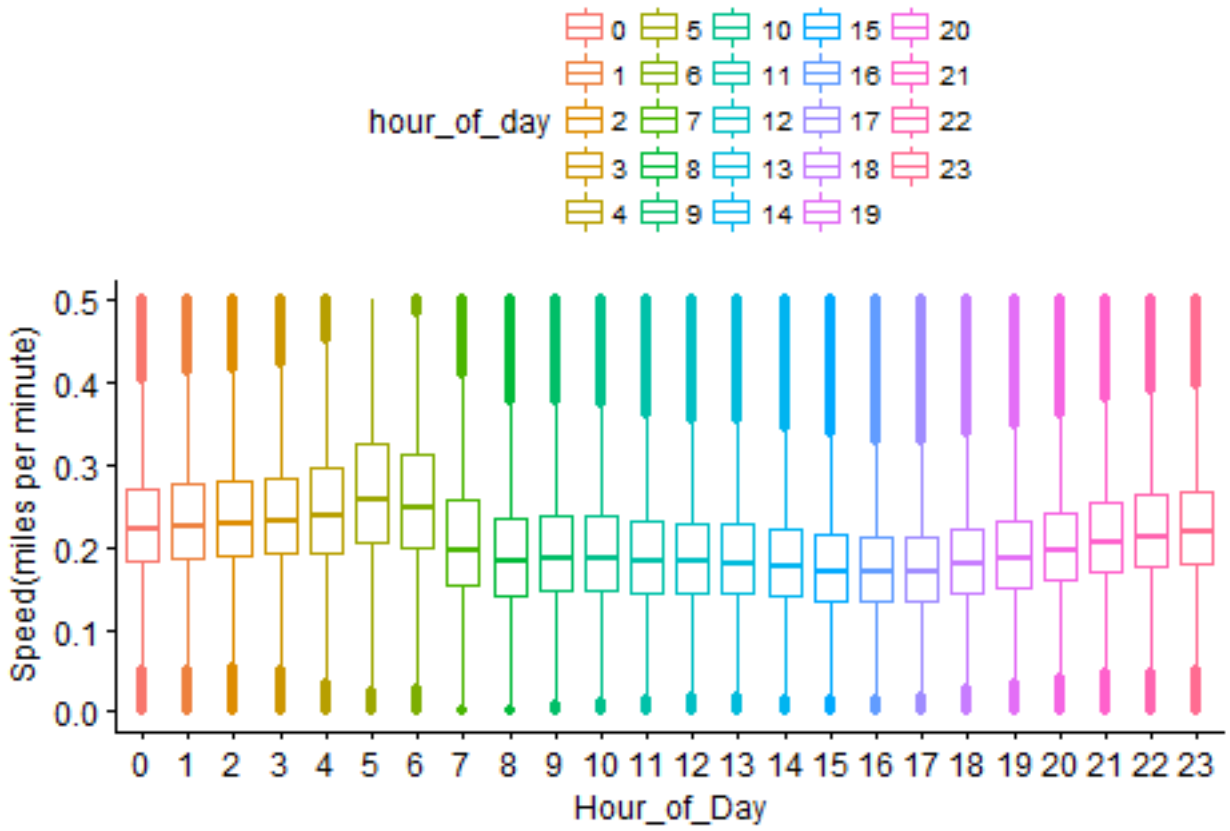
```
## week_of_month       4     78  19.606    1815 <2e-16 ***
## Residuals     1484614  16034   0.011
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Since the p-value of one-way ANOVA test is smaller than 0.05, we can conclude that there significant differences between the speed of different weeks of September.

I would hypothesize that this result may have something to do with the external factors like the weather, holiday, big-event and so on, for all of these factors would influence the traffic on the road. For example, on rainy or foggy days, the traffic could be slower than average speed.

b) Build up a hypothesis of average trip speed as a function of time of day.

Let's first compare the average speed between different time of day.




```
#anova test
aov_mod2 = aov(speed ~ hour_of_day, data = newdata)
# Summary of the analysis
summary(aov_mod2)
```

```
##                  Df Sum Sq Mean Sq F value Pr(>F)
## hour_of_day       1    171  170.64   15892 <2e-16 ***
## Residuals   1484617  15942    0.01
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

From this plot we can see that the average speed reaches a maximum value at early morning and becomes smaller in the evening. The p value of anova test indicates that there are significant differences between the speed of different hours.