NYC Taxi Data Analysis, Modelling and Visualisation

Lakshmi Rajasekhar

September 4, 2018

- 1. Introduction
- 2. Data
- 3. Exploratory Data Analysis (EDA) and Pre-processing
    - i. Structure of the Data and appropriate attribute conversions
    - ii. Data Inconsisencies and missing values
    - iii. Other Interesting Summary Statistics
    - iv. Feature Engineering
    - v. Exploratory Data Analysis (based on given questions)
- 4. Data Modelling
    - Train Test Division
    - Final Data Cleaning
    - **Q5. Features influencing fare and tip**
    - Data Standradization
    - Data Modelling for fare_amount
- 5. Conclusions
- 6. References:

2013 NYC Taxi Data Analysis, Modelling and Visualisation

1. Introduction

The main objective of this study is to identify potential opportunities to increase revenue and cut costs for NYC taxi companies. In 2013 in New York City, only medallion taxis (yellow taxi cabs) licensed by the Taxi and Limousine Commission (TLC), could pick-up and drop-off passengers in any of the five boroughs of NYC (although currently in 2018, there are many more + Uber).
The five boroughs of NYC are as shown in the figure below. Manhattan is the most densely populated of the five boroughs. The borough of Queens has two international airports, namely, John F Kennedy (JFK) and La Guardia (LGA). JFK is one of the world's busiest airports. Due to these factors, Taxi business can be profitable helping make NYC more connected.

NYCBoroughs

2. Data

This study uses historical data for NYC medallion taxi trips and fare data provided by Elula (public NYC data). Due to volume of the data, as suggested, this study focusses on one month of data (September 2013, random sampling choice). No major holidays or events were found that could affect usual trip patterns in September 2013. Also, the weather was noted to be pleasant (start of fall).
The NYC Taxi data has two files; trip details and trip fare details. The two datasets were combined to create a single dataset without duplicate variables, for ease of use. After combining, the dataset has a total of 21 variables and 14+ million trip records for September 2013.
The NYC government website (www.nyc.gov) lists description of most of the variables given in this dataset, as shown below. In 2013, there were two approved T_PEP (Technology Passenger Enhancements Project) vendors, namely, CMT and VeriFone Inc.

Field Name	Description
VendorID	A code indicating the TPEP provider that provided the record. 1= Creative Mobile Technologies, LLC; 2= VeriFone Inc.
tpep_pickup_datetime	The date and time when the meter was engaged.
tpep_dropoff_datetime	The date and time when the meter was disengaged.
Passenger_count	The number of passengers in the vehicle. This is a driver-entered value.
Trip_distance	The elapsed trip distance in miles reported by the taximeter.
RateCodeID	The final rate code in effect at the end of the trip. 1= Standard rate 2=JFK 3=Newark 4=Nassau or Westchester 5=Negotiated fare 6=Group ride
Store_and_fwd_flag	This flag indicates whether the trip record was held in vehicle memory before sending to the vendor, aka "store and forward," because the vehicle did not have a connection to the server. Y= store and forward trip N= not a store and forward trip
Payment_type	A numeric code signifying how the passenger paid for the trip. 1= Credit card 2= Cash 3= No charge 4= Dispute 5= Unknown 6= Voided trip
Fare_amount	The time-and-distance fare calculated by the meter.
MTA_tax	$0.50 MTA tax that is automatically triggered based on the metered rate in use.
Improvement_surcharge	$0.30 improvement surcharge assessed trips at the flag drop. The improvement surcharge began being levied in 2015.
Tip_amount	Tip amount – This field is automatically populated for credit card tips. Cash tips are not included.
Tolls_amount	Total amount of all tolls paid in trip.
Total_amount	The total amount charged to passengers. Does not include cash tips.

Attribute Description

```
# # First read in the csv files using fread for fast reading and then convert to R object (.rds)
 for faster reloads later
# trip_data = fread("trip_data/trip_data_9.csv", data.table = F)
# trip_fare = fread("trip_fare/trip_fare_9.csv", data.table = F)
# # column bind both the datasets into one dataset
# taxi_data = cbind(trip_data, trip_fare)
# # remove duplicate columns
# taxi_data = taxi_data[, !duplicated(colnames(taxi_data))]
# # save as .rds for easier read-in later
# saveRDS(taxi_data, "taxi_data_9.rds")

# Read-in .rds file
taxi_data = readRDS("taxi_data_9.rds")
```

3. Exploratory Data Analysis (EDA) and Pre-processing

EDA is an essential part of data science projects to understand the data further and identify data inconsistencies before performing any predictive modelling on the data.

i. Structure of the Data and appropriate attribute conversions

First, understanding the strucure of the dataset would help us convert the attributes (variables) appropriately.

```
str(taxi_data)
```

```
## 'data.frame':    14107693 obs. of  21 variables:
##  $ medallion         : chr  "0CF8B9F42FED24FA1CA8AACA36D1A25B" "4D3E527682E42F1FACDFBF2D56757
AC6" "D28DD94BE7682B7434EC5CA4D523A788" "CFF1FDD049E5433E6FBCC96EDA9E66A5" ...
##  $ hack_license      : chr  "A04B37232EB2478C7A831A6C587C15B4" "8C1A6E14ED1D019FBD227970CC619
496" "96F7564919171F55F551F0F9B96B5199" "9BF57AD72288939D6385149524149269" ...
##  $ vendor_id         : chr  "CMT" "CMT" "CMT" "CMT" ...
##  $ rate_code         : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ store_and_fwd_flag: chr  "N" "N" "N" "N" ...
##  $ pickup_datetime   : chr  "2013-09-01 16:35:05" "2013-09-01 17:44:05" "2013-09-01 16:36:20"
"2013-09-01 07:54:47" ...
##  $ dropoff_datetime  : chr  "2013-09-01 16:47:53" "2013-09-01 17:58:37" "2013-09-01 16:50:53"
"2013-09-01 08:09:17" ...
##  $ passenger_count   : int  2 1 1 1 1 1 2 1 1 ...
##  $ trip_time_in_secs : int  767 871 872 869 468 1113 1006 787 671 1024 ...
##  $ trip_distance     : num  2.6 3.5 3.1 9.6 1.5 8.3 9.9 3.9 2.2 2.8 ...
##  $ pickup_longitude  : num  -74 -74 -74 -74 -74 ...
##  $ pickup_latitude   : num  40.7 40.7 40.7 40.7 40.8 ...
##  $ dropoff_longitude : num  -74 -74 -74 -73.9 -74 ...
##  $ dropoff_latitude  : num  40.8 40.8 40.8 40.8 40.8 ...
##  $ payment_type      : chr  "CRD" "CRD" "CRD" "CRD" ...
##  $ fare_amount       : num  11.5 14.5 14 27 8 25.5 28 13.5 10.5 14 ...
##  $ surcharge         : num  0 0 0 0 0.5 0.5 0.5 0.5 0.5 0 ...
##  $ mta_tax           : num  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 ...
##  $ tip_amount        : num  2.4 3.75 1 5 2 5.3 7.25 2 1.5 2 ...
##  $ tolls_amount      : num  0 0 0 5.33 0 0 0 0 0 0 ...
##  $ total_amount      : num  14.4 18.8 15.5 37.8 11 ...
```

```
# Convert appropriate attributes to categorical
cat_attrs = c("medallion","hack_license","vendor_id", "payment_type", "store_and_fwd_flag")

taxi_data[, cat_attrs] = as.data.frame(apply(taxi_data[, cat_attrs], 2, function(x) as.factor(a
s.character(x))))
taxi_data$rate_code = as.factor(as.character(taxi_data$rate_code))

#convert datetimes appropriately using lubridate library
taxi_data$pickup_datetime = ymd_hms(taxi_data$pickup_datetime)
taxi_data$dropoff_datetime = ymd_hms(taxi_data$dropoff_datetime)

# check the structure of the data again
str(taxi_data)
```

```
## 'data.frame':    14107693 obs. of  21 variables:
##  $ medallion         : Factor w/ 13438 levels "00005007A9F30E289E760362F69E4EAD",..: 705 4054
11071 10946 13086 12732 8243 11514 7558 5572 ...
##  $ hack_license      : Factor w/ 33338 levels "0002555BBE359440D6CEB34B699D3932",..: 20844 18
182 19611 20255 6135 4291 15533 2582 22527 3572 ...
##  $ vendor_id         : Factor w/ 2 levels "CMT","VTS": 1 1 1 1 1 1 1 1 1 1 ...
##  $ rate_code         : Factor w/ 12 levels "0","1","2","210",..: 2 2 2 2 2 2 2 2 2 2 ...
##  $ store_and_fwd_flag: Factor w/ 3 levels "","N","Y": 2 2 2 2 2 3 2 2 2 2 ...
##  $ pickup_datetime   : POSIXct, format: "2013-09-01 16:35:05" "2013-09-01 17:44:05" ...
##  $ dropoff_datetime  : POSIXct, format: "2013-09-01 16:47:53" "2013-09-01 17:58:37" ...
##  $ passenger_count   : int  2 1 1 1 1 1 1 2 1 1 ...
##  $ trip_time_in_secs : int  767 871 872 869 468 1113 1006 787 671 1024 ...
##  $ trip_distance     : num  2.6 3.5 3.1 9.6 1.5 8.3 9.9 3.9 2.2 2.8 ...
##  $ pickup_longitude  : num  -74 -74 -74 -74 -74 ...
##  $ pickup_latitude   : num  40.7 40.7 40.7 40.7 40.8 ...
##  $ dropoff_longitude : num  -74 -74 -74 -73.9 -74 ...
##  $ dropoff_latitude  : num  40.8 40.8 40.8 40.8 40.8 ...
##  $ payment_type      : Factor w/ 5 levels "CRD","CSH","DIS",..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ fare_amount       : num  11.5 14.5 14 27 8 25.5 28 13.5 10.5 14 ...
##  $ surcharge         : num  0 0 0 0 0.5 0.5 0.5 0.5 0.5 0 ...
##  $ mta_tax           : num  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 ...
##  $ tip_amount        : num  2.4 3.75 1 5 2 5.3 7.25 2 1.5 2 ...
##  $ tolls_amount      : num  0 0 0 5.33 0 0 0 0 0 0 ...
##  $ total_amount      : num  14.4 18.8 15.5 37.8 11 ...
```

ii. Data Inconsisencies and missing values

```
summary(taxi_data)
```

```
##                              medallion
##  20BA941F62CC07F1FA3EF3E122B1E9B2:    2002
##  2905ABD21DF99EA5B9741EA063266632:    1944
##  19E063791B0DF5A558B8488180DDAB67:    1872
##  21B98CAC5B31414B9446D381D38EEC7F:    1862
##  7417B824E9E41523D455BEAC98738B7D:    1862
##  5466D714601371299033C01FB08BB93B:    1860
##  (Other)                         :14096291
##                             hack_license      vendor_id       rate_code
##  D85749E8852FCC66A990E40605607B2F:    1628   CMT:6948757   1      :13769396
##  6B38951F9D76FA1C201EDAD717050CF7:    1547   VTS:7158936   2      :  269276
##  7B19DE6D4D54999531BEB27F758F71F6:    1506                 5      :   40548
##  74CC809D28AE726DDB32249C044DA4F8:    1480                 3      :   22561
##  CE625FD96D0FAFC812A6957139B354A1:    1457                 4      :    5118
##  3D757E111C78F5CAC83D44A92885D490:    1408                 0      :     562
##  (Other)                         :14098667                 (Other):     232
##  store_and_fwd_flag pickup_datetime               dropoff_datetime
##   :7159367          Min.   :2013-09-01 00:00:00   Min.   :2013-09-01 00:01:27
##  N:6802879          1st Qu.:2013-09-08 23:00:00   1st Qu.:2013-09-08 23:11:21
##  Y: 145447          Median :2013-09-16 08:06:51   Median :2013-09-16 08:20:45
##                     Mean   :2013-09-16 07:13:48   Mean   :2013-09-16 07:26:54
##                     3rd Qu.:2013-09-23 13:41:25   3rd Qu.:2013-09-23 13:57:45
##                     Max.   :2013-09-30 23:59:59   Max.   :2013-10-04 02:58:42
##
##  passenger_count   trip_time_in_secs trip_distance     pickup_longitude
##  Min.   :  0.000   Min.   :    0.0   Min.   :  0.000   Min.   :-1213.88
##  1st Qu.:  1.000   1st Qu.:  381.0   1st Qu.:  1.100   1st Qu.:  -73.99
##  Median :  1.000   Median :  642.0   Median :  1.800   Median :  -73.98
##  Mean   :  1.711   Mean   :  784.7   Mean   :  2.983   Mean   :  -73.19
##  3rd Qu.:  2.000   3rd Qu.: 1020.0   3rd Qu.:  3.300   3rd Qu.:  -73.97
##  Max.   :208.000   Max.   :10680.0   Max.   :100.000   Max.   :  147.72
##
##  pickup_latitude    dropoff_longitude  dropoff_latitude   payment_type
##  Min.   :-3084.31   Min.   :-2335.36   Min.   :-3113.54   CRD:7755731
##  1st Qu.:   40.73   1st Qu.:  -73.99   1st Qu.:   40.73   CSH:6287011
##  Median :   40.75   Median :  -73.98   Median :   40.75   DIS:  10331
##  Mean   :   40.32   Mean   :  -73.15   Mean   :   40.29   NOC:  33950
##  3rd Qu.:   40.77   3rd Qu.:  -73.96   3rd Qu.:   40.77   UNK:  20670
##  Max.   :  473.97   Max.   :  108.44   Max.   :  405.55
##                     NA's   :78         NA's   :78
##   fare_amount       surcharge          mta_tax         tip_amount
##  Min.   :  2.50   Min.   : 0.0000   Min.   :0.0000   Min.   :  0.000
##  1st Qu.:  6.50   1st Qu.: 0.0000   1st Qu.:0.5000   1st Qu.:  0.000
##  Median :  9.50   Median : 0.0000   Median :0.5000   Median :  1.000
##  Mean   : 12.73   Mean   : 0.3173   Mean   :0.4981   Mean   :  1.425
##  3rd Qu.: 14.50   3rd Qu.: 0.5000   3rd Qu.:0.5000   3rd Qu.:  2.000
##  Max.   :500.00   Max.   :14.5000   Max.   :0.5000   Max.   :200.000
##
##   tolls_amount      total_amount
##  Min.   : 0.0000   Min.   :  2.50
##  1st Qu.: 0.0000   1st Qu.:  8.00
##  Median : 0.0000   Median : 11.50
##  Mean   : 0.2723   Mean   : 15.24
```

```
##  3rd Qu.: 0.0000   3rd Qu.: 17.00
##  Max.   :20.0000   Max.   :508.25
##
```

We can point out some possible data consistencies by looking at the above summary statistics as follows:
1. **pickup and dropoff latitude and longitude** - There seems to be extreme outlier values (E.g., -1213.88) which are impossible as a latitude or longitude value.There are also 78 missing NA values in dropoff variables.
2. **rate_code** - Expected values for rate_code are 1 to 6 as described in the attribute description above. The data seems to have multiple rate_codes including rate_code = 0.
3. **store_and_fwd_flag** - There are missing records but we will have to look at the usability of this variable before deciding to deal with missing values.
4. **payment_type** - There are “DIS” type payments which mean disputed payments which might actually mean losses for the driver/taxi company as it is not clearly mentioned if the passnger paid the disputed amount or not.
5. **Dropoff datetime** - The maximum entry is a dropoff time for 4th of October (early hours of 1st of October are acceptable, but not this).

Looking at each of the above data inconsistensies:

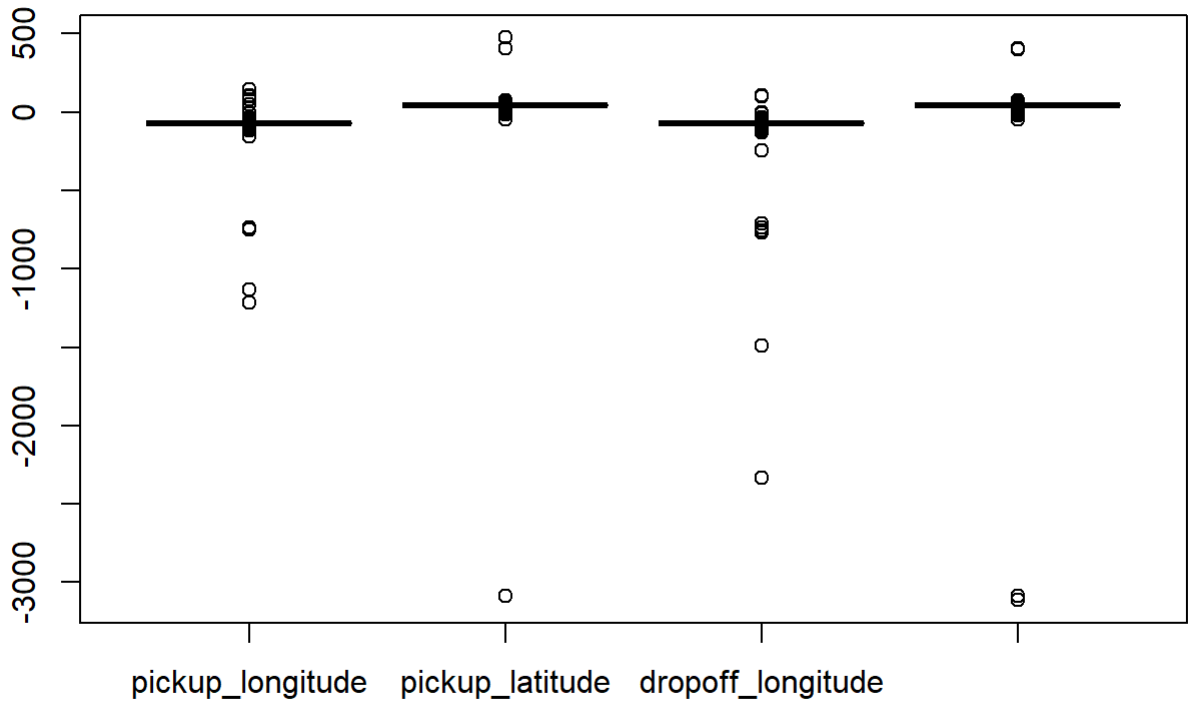
1. **Latitude and Longitude**
   The latitude and longitudes data has a lot of inconsistencies, based on detailed exploration.

a. There are zero values and NA values in the geolocation variables. This might mostly be signal reception issues. There are 78 missing records for drop-off latitude and longitude.
b. Longitudes vary between -180 to 180 degrees and latitudes vary from -90 to 90 degrees. Any values beyond this can be marked as missing values too.
c. There are also some values closer to 0 like -0.002 etc which should also be signal issue.

```
geo_variables = c("pickup_longitude", "pickup_latitude", "dropoff_longitude", "dropoff_latitude"
)

summary(taxi_data[,geo_variables])
```

```
##  pickup_longitude   pickup_latitude    dropoff_longitude  dropoff_latitude
##  Min.   :-1213.88   Min.   :-3084.31   Min.   :-2335.36   Min.   :-3113.54
##  1st Qu.:  -73.99   1st Qu.:   40.73   1st Qu.:  -73.99   1st Qu.:   40.73
##  Median :  -73.98   Median :   40.75   Median :  -73.98   Median :   40.75
##  Mean   :  -73.19   Mean   :   40.32   Mean   :  -73.15   Mean   :   40.29
##  3rd Qu.:  -73.97   3rd Qu.:   40.77   3rd Qu.:  -73.96   3rd Qu.:   40.77
##  Max.   :  147.72   Max.   :  473.97   Max.   :  108.44   Max.   :  405.55
##                                        NA's   :78         NA's   :78
```

```
boxplot(taxi_data[,geo_variables])
```



Due to various inconsistencies noted above, I plan to find the 1.5 X IQR (inter-quartile range) of each of the coordinate variable and mark anything outside 1.5 X IQR from both first and third quartiles as NA. I guess this is a descent approach given that the data is only for NYC and 1.5XIQR from the first and third quartiles would defintely cover more than its neighbouring areas to cover for long-distance trips.

```
# Function that accepts a vector, calculates IQR and returns row indices of the records satusfyi
ng the condition
# condition used = (value > 1.5IQR*4th quartile) or (value< 1.5IQR * 2nd quartile)
find_geoOutlier = function(x){
  iqr = 1.5*IQR(x, na.rm = T)
  qntl = quantile(x, na.rm = T)
  y = which((!is.na(x)) & ((abs(x) < (abs(qntl[2]) - abs(iqr*qntl[2]))) | (abs(x) > (abs(qntl[4
])+abs(iqr*qntl[4])))))
  return(y)
}

# setting all values beyond these values as NA
rows = find_geoOutlier(taxi_data$pickup_latitude)
taxi_data$pickup_latitude[rows] = NA
rows = find_geoOutlier(taxi_data$pickup_longitude)
taxi_data$pickup_longitude[rows] = NA
rows = find_geoOutlier(taxi_data$dropoff_latitude)
taxi_data$dropoff_latitude[rows] = NA
rows = find_geoOutlier(taxi_data$dropoff_longitude)
taxi_data$dropoff_longitude[rows] = NA

summary(taxi_data[, geo_variables])
```

```
##  pickup_longitude pickup_latitude  dropoff_longitude dropoff_latitude
##  Min.   :-76.66   Min.   :38.73    Min.   :-77.06    Min.   :38.70
##  1st Qu.:-73.99   1st Qu.:40.73    1st Qu.:-73.99    1st Qu.:40.73
##  Median :-73.98   Median :40.75    Median :-73.98    Median :40.75
##  Mean   :-73.98   Mean   :40.75    Mean   :-73.97    Mean   :40.75
##  3rd Qu.:-73.97   3rd Qu.:40.77    3rd Qu.:-73.96    3rd Qu.:40.77
##  Max.   :-71.19   Max.   :42.78    Max.   :-70.94    Max.   :42.88
##  NA's   :149539   NA's   :149371   NA's   :158491    NA's   :158353
```

```
colSums(is.na(taxi_data))
```

```
##          medallion       hack_license          vendor_id          rate_code
##                  0                  0                  0                  0
## store_and_fwd_flag    pickup_datetime   dropoff_datetime    passenger_count
##                  0                  0                  0                  0
##  trip_time_in_secs      trip_distance   pickup_longitude    pickup_latitude
##                  0                  0             149539             149371
##  dropoff_longitude   dropoff_latitude       payment_type        fare_amount
##             158491             158353                  0                  0
##          surcharge            mta_tax         tip_amount       tolls_amount
##                  0                  0                  0                  0
##       total_amount
##                  0
```

```
sum(is.na(taxi_data[,geo_variables]))*100/(4*nrow(taxi_data))
```

```
## [1] 1.091167
```

About 1% of data is now NA values in the geo-coordinate variables. Since the trip distance is already given, I would not be using geo-coordinates for modelling. Also, I do not plan to drop records with geo-coordinate values as NA. It might still be useful for further EDA of other variables.

2. **rate_code**
   Rate codes are entered by the taxi driver during the trip. Hence, the probability of human data entry error is very high.

```
# unique rate codes in data
levels(taxi_data$rate_code)
```

```
##  [1] "0"   "1"   "2"   "210" "28"  "3"   "4"   "5"   "6"   "7"   "8"   "9"
```

```
# frequency table
table(taxi_data$rate_code)
```

```
##
##        0        1        2      210       28        3        4        5
##      562 13769396   269276       11        1    22561     5118    40548
##        6        7        8        9
##      201        2       14        3
```

Instead of rate_codes 1 - 6, there are other values listed. I did some research online but could not confirm if the rate codes for 2013 were different from what the nyc gov website has today. It might be real rate_codes in 2013 but rate_codes like ‘210’ and ‘28’ looks like multiple stop trips where the taxi driver manually changes the rate_codes after each destination. For now, I just let the data be without any changes due to lack of proper information.

3. **store_and_fwd_flag**
   This variable should have two two options: “Y” - to denote that the trip data was stored and then forwarded to the server when netwrok issues were restored, “N” - to denote that trip data was sent immediately.

```
prop.table(table(taxi_data$store_and_fwd_flag))
```

```
##
##                     N          Y
## 0.50747964 0.48221059 0.01030977
```

The table above shows more than 50% missing values. 48% of the data was collected when there were no network issues. This leads me to assume that the missing values might be a ‘Y’. But imputing based on plain assumption is invalid.

```
temp = taxi_data[taxi_data$store_and_fwd_flag == "",]
#summary(temp)
cat("Distribution when no network connection")
```

```
## Distribution when no network connection
```

```
table(temp$payment_type)
```

```
##
##     CRD     CSH     DIS     NOC     UNK
## 3975698 3162978       6      15   20670
```

```
temp = taxi_data[taxi_data$store_and_fwd_flag == "N",]
#summary(temp)
cat("Distribution when network connection is intact")
```

```
## Distribution when network connection is intact
```

```
table(temp$payment_type)
```

```
##
##     CRD     CSH     DIS     NOC     UNK
## 3700830 3059306    9978   32765       0
```

I studied the summary statistics of NYC data seperated into two sets based on store_an_fwd_flag == ‘N’ and “” (missing). I was expecting to see more data inconsistencies in the other variables (like lat and long) in the “” group than the “N” group. Although that was not the case, one striking observation I had was with the ‘payment_type’ variable: 1. With network issues, there are more ‘UNK’ (unknown payments) and less ‘NOC’ (no charges).
2. Without network issues, there are more ‘NOC’ and zero ‘UNK’.

Although it is difficult to make conclusions, it might be because cash payments are recorded as ‘UNK’ during network connection issues.
Overall I feel this variable is not that useful from modelling perspective but might be useful from exploration and cleaning perspective. Hence keeping the variable for now.

4. **Dropoff datetime**

```
taxi_data[(month(taxi_data$dropoff_datetime)>9 & day(taxi_data$dropoff_datetime)>1),]
```

```
##                                medallion                    hack_license
## 10220692 6E3369BED599B9EAF4C80E504BA487D7 75ED245E92E47F3829B68B2CC469EF17
##          vendor_id rate_code store_and_fwd_flag     pickup_datetime
## 10220692       CMT         2                  N 2013-09-23 02:10:22
##             dropoff_datetime passenger_count trip_time_in_secs trip_distance
## 10220692 2013-10-04 02:58:42               3               650           3.8
##          pickup_longitude pickup_latitude dropoff_longitude dropoff_latitude
## 10220692        -73.98559        40.76337         -73.99435          40.6161
##          payment_type fare_amount surcharge mta_tax tip_amount tolls_amount
## 10220692          NOC        13.5       0.5     0.5          0            0
##          total_amount
## 10220692         14.5
```

Looking at the data, there is only one record which has a dropoff date of 4th of October. This is unusual given that the pickup date is 23rd of September. This might be a data entry error. Rest of the data which has dates in October are in the early hours of 1st of October with a pickup on 30th of Spetember which are valid entries. Dropping records/variables during EDA might affect the analysis itself and hence not making any decision to drop them till data modelling.

iii. Other Interesting Summary Statistics

```
cat("Total number of yellow taxis in September 2013: ", length(unique(taxi_data$medallion)))
```

```
## Total number of yellow taxis in September 2013:  13438
```

```
cat("\nTotal number of yellow taxi drivers in September 2013: ", length(unique(taxi_data$hack_li
cense)))
```

```
##
## Total number of yellow taxi drivers in September 2013:  33338
```

iv. Feature Engineering

This section lists all the new attributes derived from existing data based on observations while doing the EDA and modelling (iteratively added them here as and when required). I decided to put all the newly created attributes in one section for easy reference later.

```
# 'f' in attribute names denotes 'newly created feature'. A convention I am following to keep tr
ack of engineered features.

# Extract the day of the week
taxi_data$fWeekday = factor(weekdays(taxi_data$pickup_datetime), levels = c("Monday", "Tuesday",
 "Wednesday", "Thursday", "Friday", "Saturday","Sunday"))
# Extract day (day of the month)
taxi_data$fDay = day(taxi_data$pickup_datetime)
# Extract hour from time
taxi_data$fHour = hour(taxi_data$pickup_datetime)
# Extract Weekday or Weekend identifier
taxi_data$fWendOrWday = ifelse(taxi_data$fWeekday == "Saturday", "Weekend", ifelse(taxi_data$fWe
ekday== "Sunday", "Weekend", "Weekday"))
```

v. Exploratory Data Analysis (based on given questions)

EDA Q1: Busiest Locations and Hours

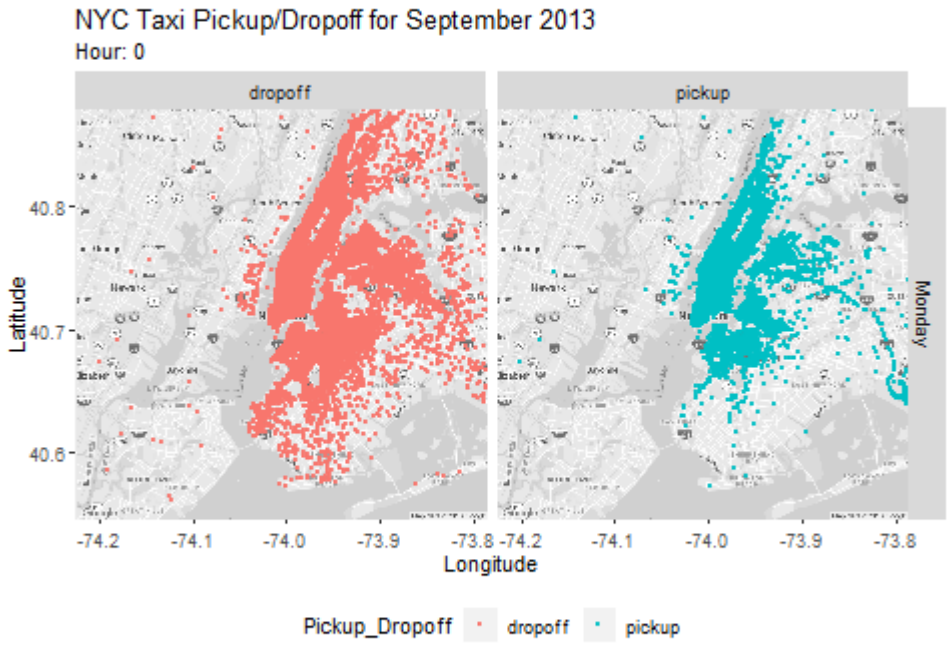
Since location of pickup and dropoff is not given, mapping lat and long coordinates back to the five boroughs of NYC would be a tedious task. In the interest of time, I plan to visually analyse the busy locations based on time of day and day of week for various trips.

Volume of Trips

```
# combine pickup and dropoff latitudes and longitudes into long format data for plotting
temp_pickup = taxi_data %>% select(pickup_longitude, pickup_latitude, fHour, fWeekday) %>% renam
e(longitude = pickup_longitude , latitude = pickup_latitude) %>% mutate("Pickup_Dropoff" = "pick
up")
temp_dropoff = taxi_data %>% select(dropoff_longitude, dropoff_latitude, fHour, fWeekday)  %>% r
ename(longitude = dropoff_longitude , latitude = dropoff_latitude) %>% mutate("Pickup_Dropoff" =
 "dropoff")

#Bind them into a single dataframe before plotting lat and long coordinates
temp = rbind(temp_pickup, temp_dropoff)
rm(temp_dropoff, temp_pickup)

nyc_map = get_map("New York City", maptype='roadmap', zoom = 11, source="google", color = "bw")
```

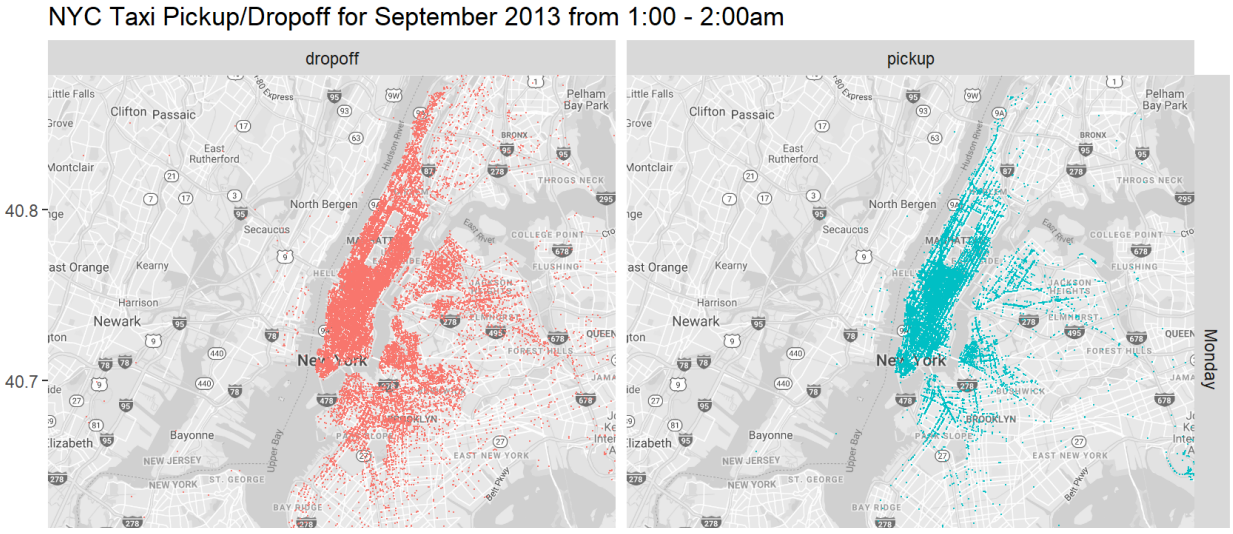
The day of week for the below animation is chosen as “Monday” for better visibility.

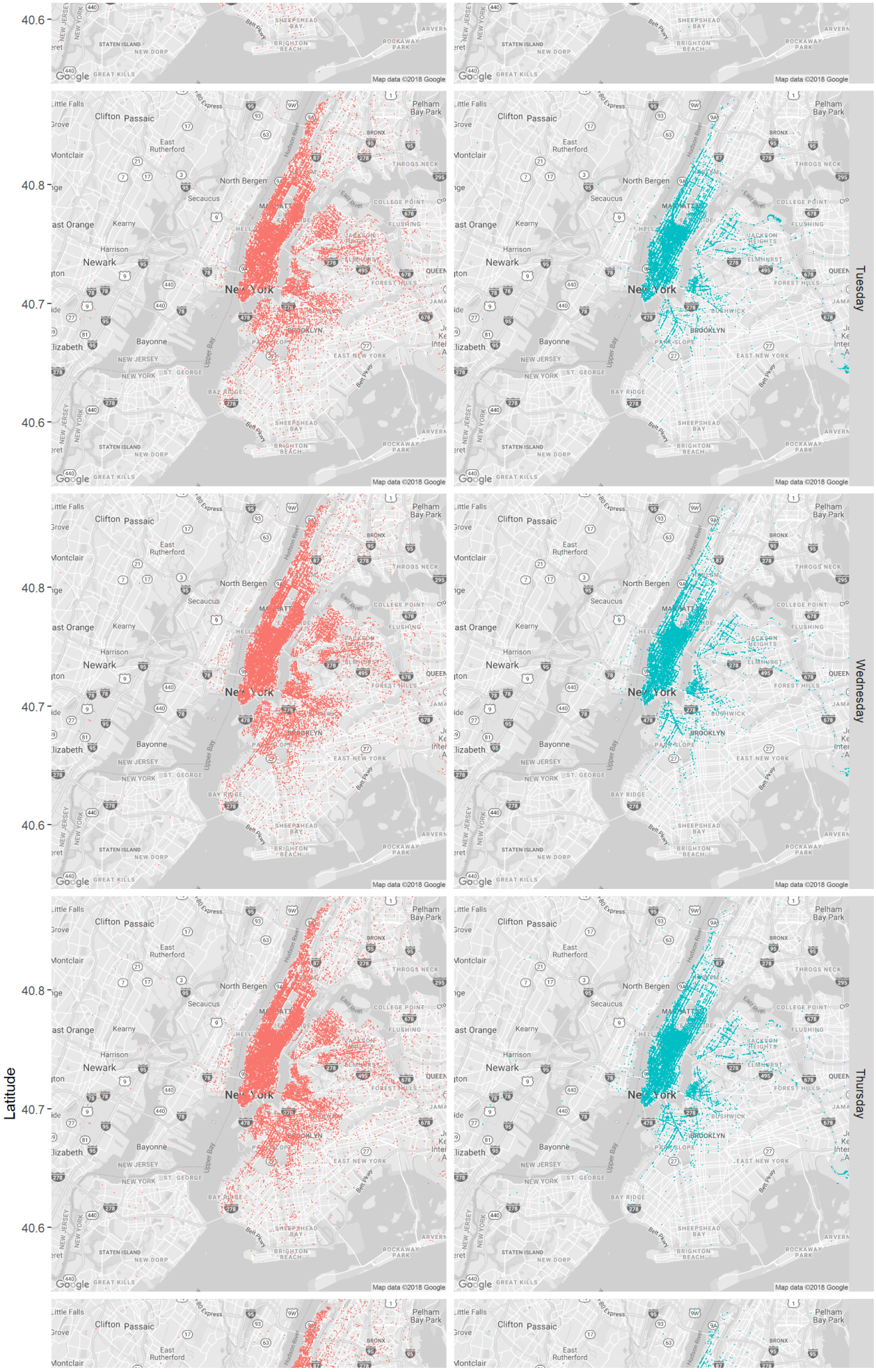
```
#Animation of trips based on hour of day and day of week
temp1 = temp[temp$fWeekday == "Monday",]
p = ggmap(nyc_map) + geom_point(aes(x = longitude, y = latitude, colour = Pickup_Dropoff), data
 = temp1, size=.001, alpha=0.9) +
  scale_fill_manual(values = c(pickup = "green4", dropoff = "tomato3")) + labs(x = "Longitude",
 y = "Latitude", title = "NYC Taxi Pickup/Dropoff for September 2013", subtitle  = 'Hour: {frame
_time}') +
  facet_grid(fWeekday~Pickup_Dropoff) + theme(legend.position="bottom") + transition_time(fHour)


animate(p)
```

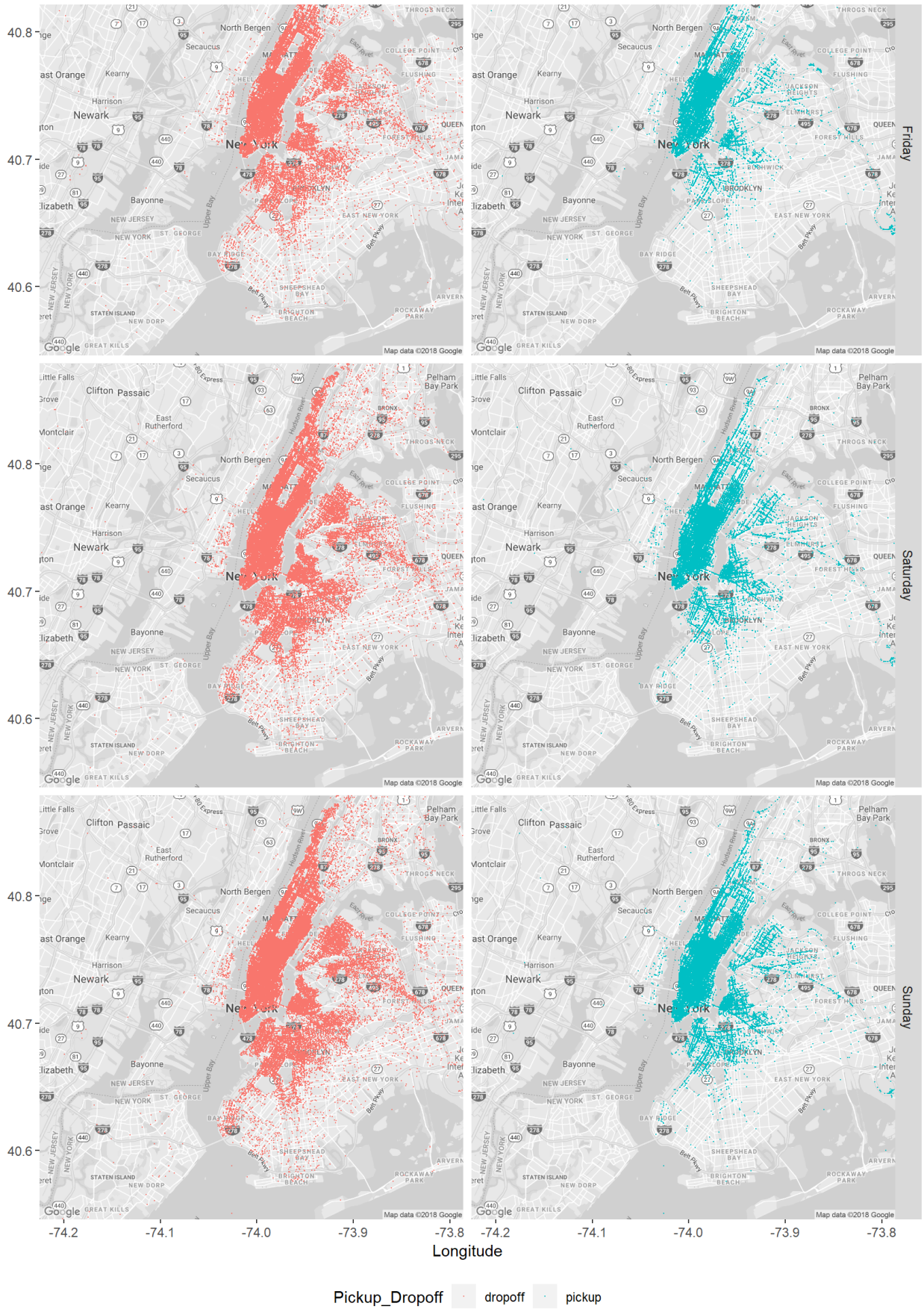



```
anim_save("sample_anim")
```

```
# Plot for Hour = 1
temp1 = temp[temp$fHour == 1,]
p = ggmap(nyc_map) + geom_point(aes(x = longitude, y = latitude, colour = Pickup_Dropoff), data
 = temp1, size=.001, alpha=0.9) +
  scale_fill_manual(values = c(pickup = "green4", dropoff = "tomato3")) + labs(x = "Longitude",
 y = "Latitude", title = "NYC Taxi Pickup/Dropoff for September 2013 from 1:00 - 2:00am") + face
t_grid(fWeekday~Pickup_Dropoff) + theme(legend.position="bottom")
p
```




Latitude
40.6
40.7
40.8
Tuesday
Wednesday
Thursday
Map data ©2018 Google

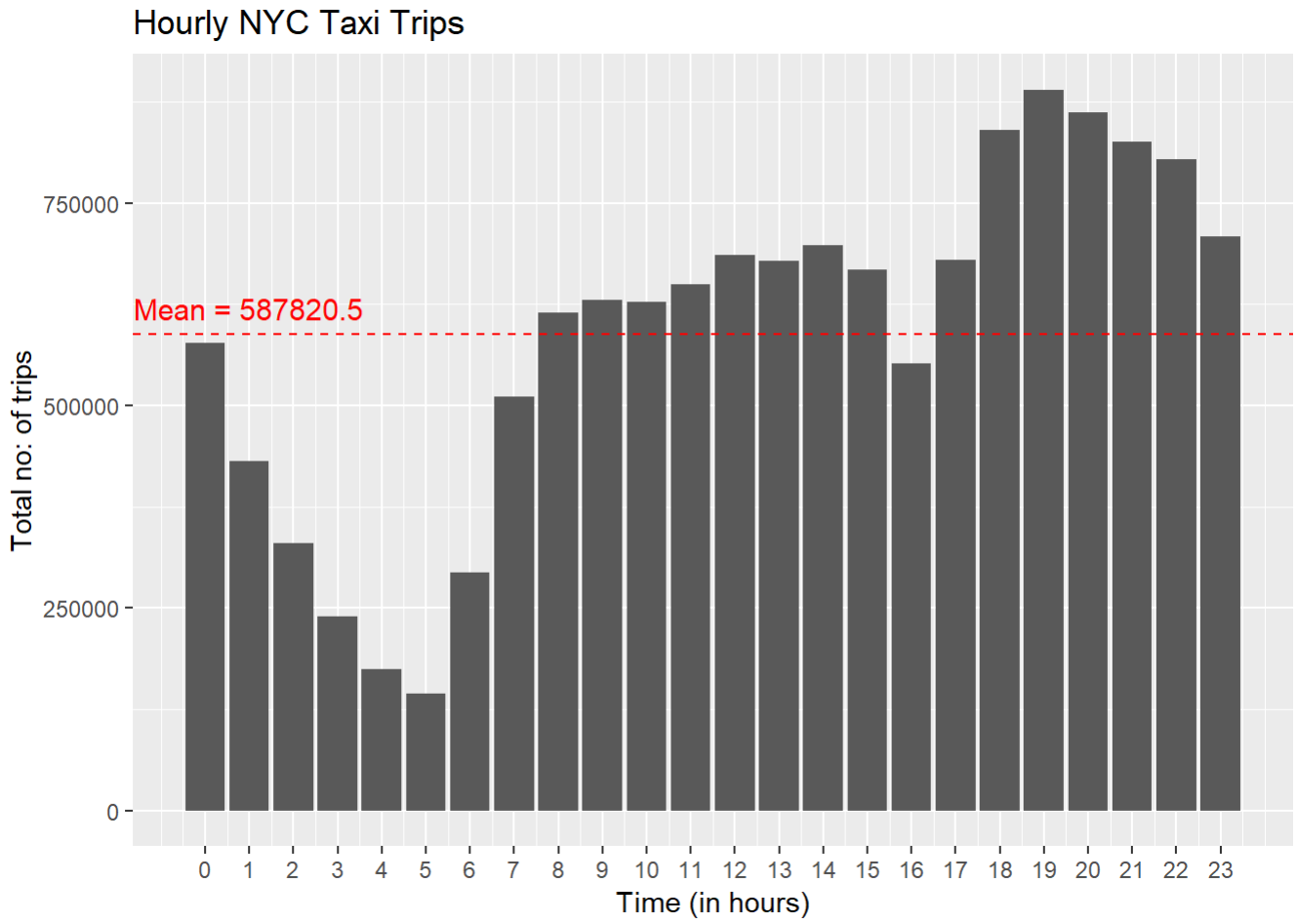


The above figures and animation shows that the whole of Manhattan is always busy. Any day, any time - a taxi driver in the Manhattan area will always have trips. The northern parts of Brooklyn and Queens are also busy, especially around the LaGuardia Airport. There are also steady traffic to and from JFK Airport in the south of

Brooklyn. Staten Island has the least amount of trips. Western part of Bronx, adjacent to Manhattan also has trips. There are also some taxi trips in the Newark area (especially Jersey City).

Trips by Hour of Day

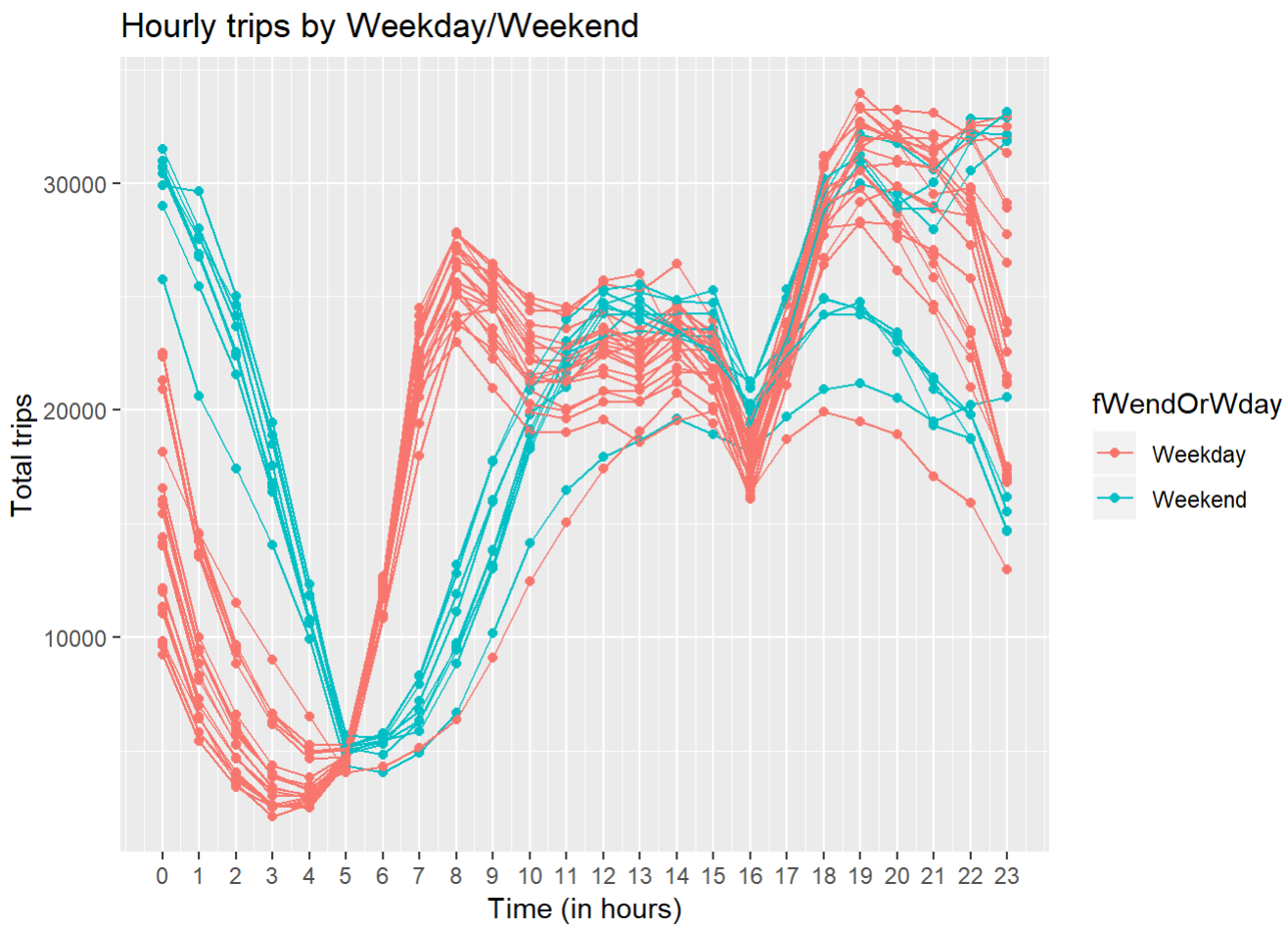
```
temp_hour = taxi_data %>% group_by(fHour) %>% summarise("Total_trips" = n())

ggplot(temp_hour, aes(x = fHour, y = Total_trips)) + geom_bar(stat = 'identity') + labs(title="H
ourly NYC Taxi Trips", x="Time (in hours)",y = "Total no: of trips") + geom_hline(aes(yintercept
 = mean(Total_trips)), colour = 'red', linetype = "dashed")+annotate(geom="text", x=1, y=620000,
 label="Mean = 587820.5", color="red", size = 4)+ scale_x_continuous(breaks = c(0:23))
```



The above plots shows that the morning rush hour is spread out and continues to afternoon lunch rush hour. There are sudden drops in trip numbers from 5:00-6:00 in the morning and from 4:00 - 5:00 in the evening. This looks like it corresponds to driver shift changes in the day. Assuming the 4:00pm trips are similar in numbers to 3:00pm and 5:00pm, there are a lot of unmet demand during this time. The evening rush hour is shorter in duration and peaks around 7:00pm in the night.

Hourly Trips groups as Weekday or Weekend

```
temp_day = taxi_data %>% group_by(fDay, fHour, fWendOrWday) %>% summarise("Total_trips" = n())

ggplot(temp_day, aes(x = fHour, y = Total_trips, group = fDay, colour = fWendOrWday)) + geom_poi
nt() + geom_line()+ labs(title="Hourly trips by Weekday/Weekend", x="Time (in hours)",y = "Total
 trips") + scale_x_continuous(breaks = c(0:23))
```

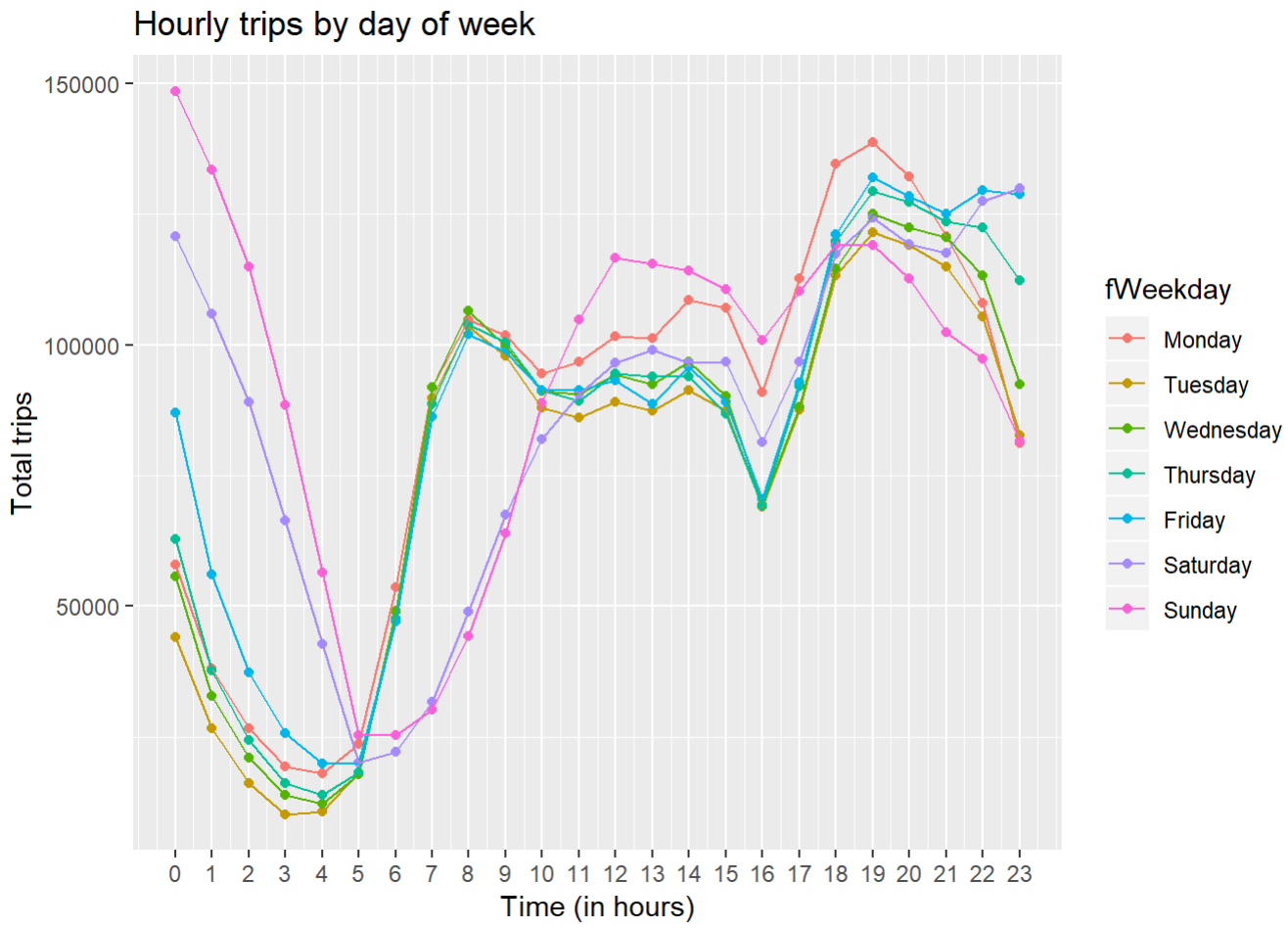


There are clearly different travel patterns during the weekday and the weekend. This is very useful for any taxi company to schedule and optimise their driver shifts. Some of the useful observations are:
* On a weekday, work trips start early peaking around 8:00am. In contrast weekends show a slow morning and trips peaking around noon - 1:00pm.
* Night trips are very high for Weekdays. My guess is that more people might be taking public transport in the morning than in the night.
* Weekend night trips show a completely different pattern. One patterns indicates trips as high as any weekday trips and increasing (I am guessing it is for Saturdays). Another pattern shows people retiring early and trips dropping drastically after 7:00pm.
* Early morning has higher number of trips over the weekends (possibly people getting back home after partying).

Hourly Trips groups as Weekday or Weekend

For more clarity, I divided the above plot again based on which day of the week it is.

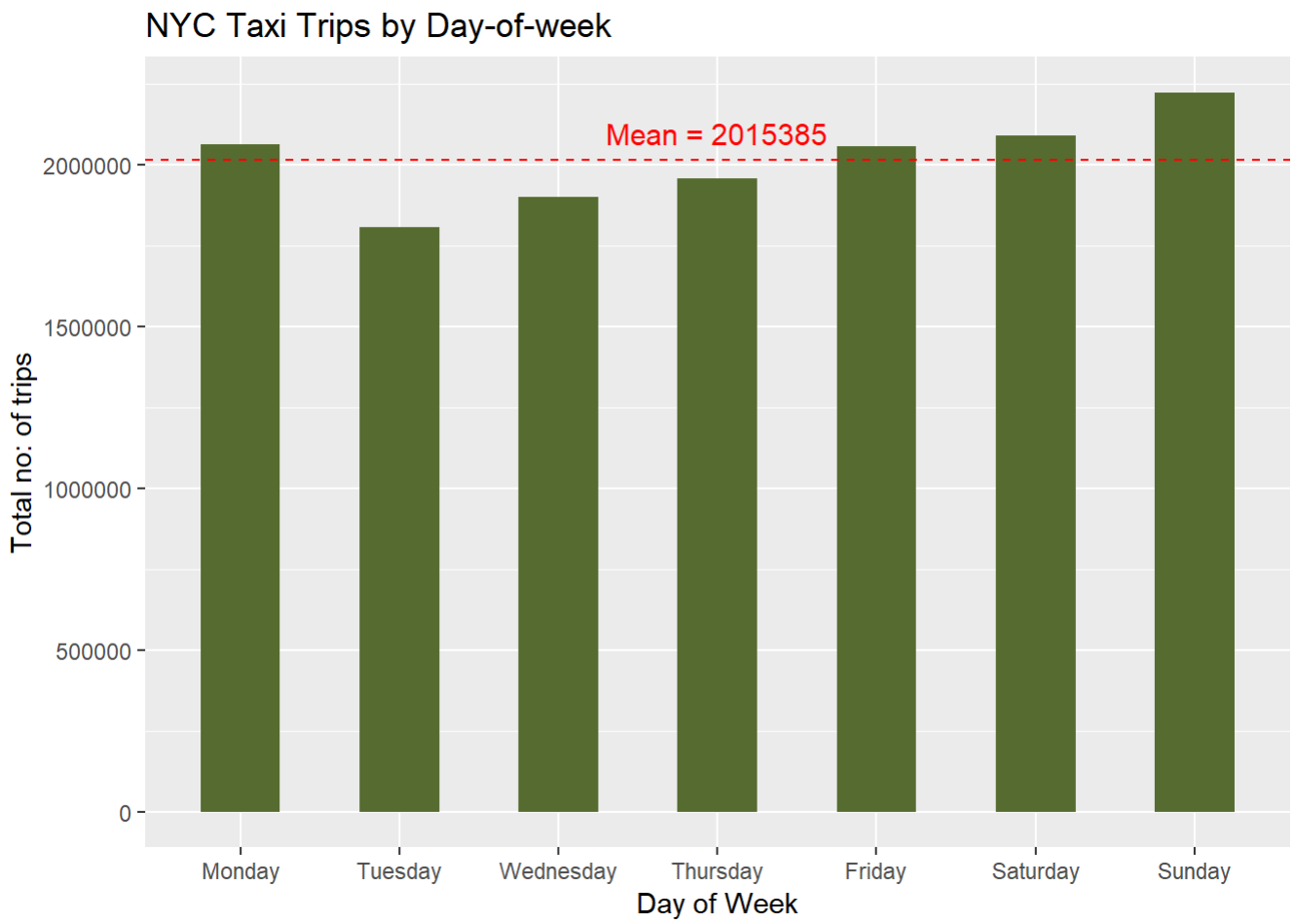
```
temp_day = taxi_data %>% group_by(fWeekday, fHour) %>% summarise("Total_trips" = n())

ggplot(temp_day, aes(x = fHour, y = Total_trips, group = fWeekday, colour = fWeekday)) + geom_po
int() + geom_line()+ labs(title="Hourly trips by day of week", x="Time (in hours)",y = "Total tr
ips") + scale_x_continuous(breaks = c(0:23))
```



The above plot confirms that Saturdays have high night trips. Saturday early mornings trips are also high denoting an active Friday night life in NYC. Surpirsingly, Mondays have a higher than normal afternoon and night trips (compared to other weekdays).

Trips by Day of Week

```
temp_day = taxi_data %>% group_by(fWeekday) %>% summarise("Total_trips" = n())

ggplot(temp_day, aes(x = fWeekday, y = Total_trips)) + geom_bar(stat = 'identity', width = 0.5,
 fill = "darkolivegreen") + labs(title="NYC Taxi Trips by Day-of-week", x="Day of Week",y = "Tot
al no: of trips")+ geom_hline(aes(yintercept = mean(Total_trips)), colour = 'red', linetype = "d
ashed")+annotate(geom="text", x=4, y=2100000, label="Mean = 2015385", color="red", size = 4)
```



Trips are the highest on Sundays with higher than average number of trips over the weekend. Fridays and Mondays also show a higher number. Higher Friday trips maybe due to Friday night trips as confirmed from the previous plot. It would be interesting to look at why Mondays have a higher than normal number of afternoon - night trips.
Volume of trips is highest over the weekends and any taxi company in the NYC taxi market should have taxis runnings during this time to maximize revenue.

** EDA Q2: Distribution of passengers per trip, payment type, fare and tip amounts**

Distribution of passengers per trip

Looking at the frequency table for the passenger counts, most the NYC yellow cab trips are single passenger trips. There are some zero passenger trips which might be data entry errors by the taxi driver.

```
table(taxi_data$passenger_count)
```

```
##
##       0       1       2       3       4       5       6       7       8       9
##     167 9918157 1940043  595639  280608  801801  571268       5       1       2
##     208
##       2
```

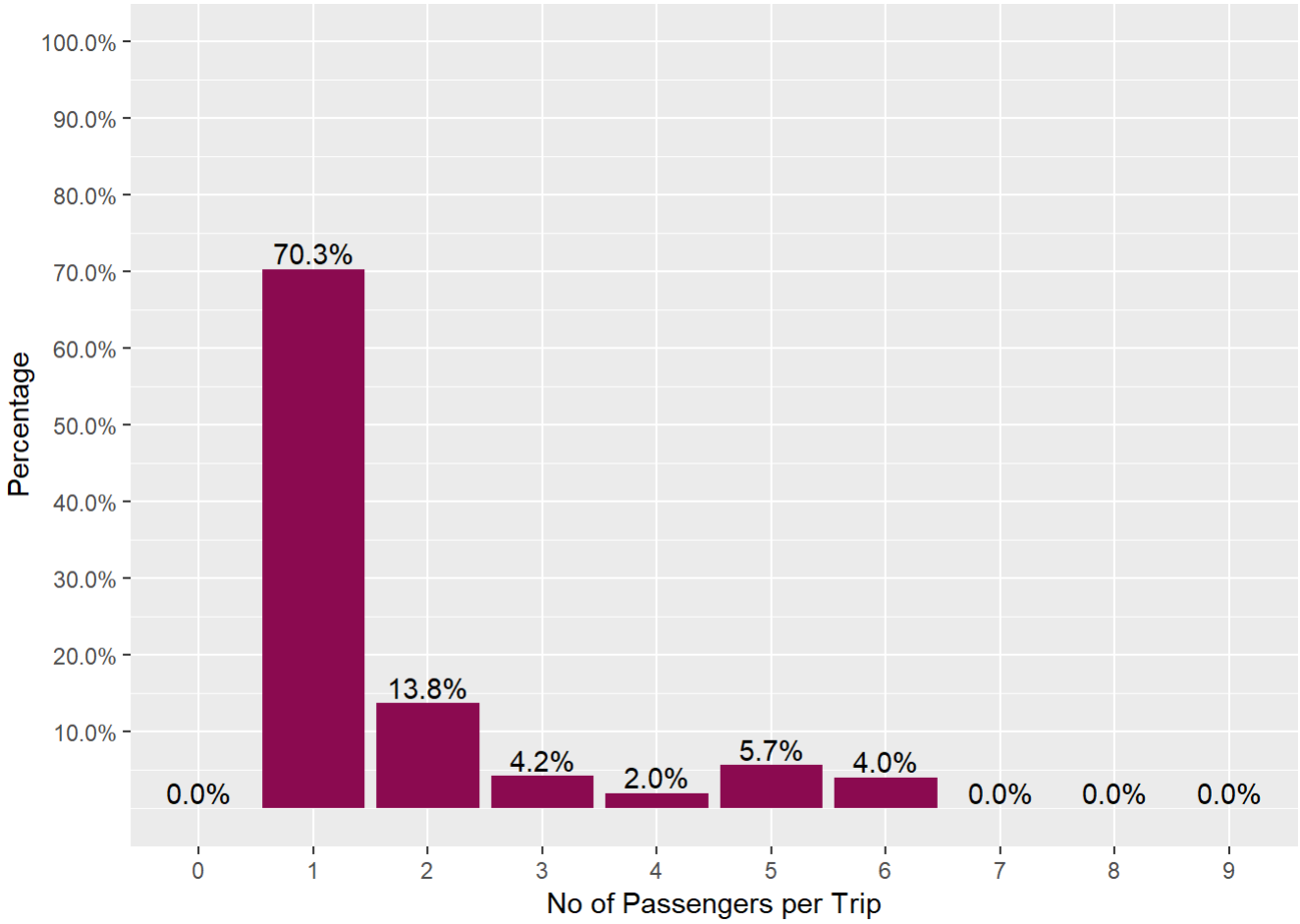
Passenger counts greater than 6 are rare. passenger count of 208 is definitely a data entry error.

```
##                              medallion                    hack_license
## 58154  771A920034826064643A2D215E7AE6E1 CFCD208495D565EF66E7DFF9F98764DA
## 369842 771A920034826064643A2D215E7AE6E1 CFCD208495D565EF66E7DFF9F98764DA
##        vendor_id rate_code store_and_fwd_flag     pickup_datetime
## 58154        VTS         1                    2013-09-14 09:36:00
## 369842       VTS         1                    2013-09-13 13:07:00
##           dropoff_datetime passenger_count trip_time_in_secs trip_distance
## 58154  2013-09-14 09:36:00             208                 0             0
## 369842 2013-09-13 13:07:00             208                 0             0
##        pickup_longitude pickup_latitude dropoff_longitude dropoff_latitude
## 58154                NA              NA                NA               NA
## 369842               NA              NA                NA               NA
##        payment_type fare_amount surcharge mta_tax tip_amount tolls_amount
## 58154           CSH         3.3         0       0          0            0
## 369842          CSH         3.3         0       0          0            0
##        total_amount fWeekday fDay fHour fWendOrWday
## 58154           3.3 Saturday   14     9     Weekend
## 369842          3.3   Friday   13    13     Weekday
```

Looking closely at the two records with 208 passenger_counts, we can find that it is by the same driver and the other attributes corresponding to these records look erronous as well. Hence, dropping these two records from the dataset.

```
taxi_data = taxi_data[-which(taxi_data$passenger_count ==208), ]
```

```
temp = taxi_data %>% group_by(passenger_count) %>% summarise(count = n()) %>% mutate(percentage=
count/sum(count))

colnames(temp) = c("No_of_Passengers", "Count_of_Passengers", "Percentage")

brks = seq(.10,1,.10)
ggplot(temp, aes(x = as.factor(as.character(No_of_Passengers)), y = Percentage*100)) +
  geom_bar(stat = "identity",fill = I("deeppink4"))+
  scale_y_continuous(breaks = brks*100, labels = scales::percent(brks), limits = c(0,100)) + xla
b(label = "No of Passengers per Trip") +
  ylab(label = "Percentage")+geom_text(aes(label=scales::percent(Percentage)), position=position
_dodge(), vjust=-0.25)
```


100.0%
90.0%
80.0%
70.0%
60.0%
50.0%
40.0%
30.0%
20.0%
10.0%
Percentage
70.3%
13.8%
4.2%
2.0%
5.7%
4.0%
0.0%
0.0%
0.0%
0.0%
0
1
2
3
4
5
6
7
8
9
No of Passengers per Trip

```
##                                 medallion                       hack_license
## 38134    8601A1E7EECD81D665152229A7350E2E 4BDF44E18B48DC534CC4077B534A500B
## 126139   FE19AC771E09280C5FB30EFD02F6F6E7 E7D72B9E6E7835B7129614DDECA1952B
## 673262   80A77FB91A5CB2E8A051186138C7CA59 8597E9E2422EC0BB669F673A07596AC6
## 987582   15D77D27D2F807E8D30E82BBB8046CBD 91D63B97DD8027FBE72087B6DD59FADD
## 1068387  41AF0EB6FDFCBF0A0C08F3241411C6C1 D7770D1E7438E4C52F1419940DF6FD1B
## 1096146  BF9D136E71CE1E93F3779316EA978AC7 2A763C9CE34866EE2155186CAD87C349
## 1197346  41AF0EB6FDFCBF0A0C08F3241411C6C1 D7770D1E7438E4C52F1419940DF6FD1B
## 1476721  41AF0EB6FDFCBF0A0C08F3241411C6C1 D7770D1E7438E4C52F1419940DF6FD1B
## 1758222  647F72E0DB27329BEDE30230F6556EEF A5BFADA8A2232F80DF1099696BDC44E4
## 1776226  294134BC853AC59B821B01C602249488 86EB50C0E1F74EEA23266FC5D0D80276
## 1821255  5A022528CD1174DF609DC92FF877EED2 0DD4A4383927098D2CB5A90C049005DE
## 1886968  25B987AB9F5FF5C4D634F386138792E3 882790FD2DD2A0E4C26631567B7F6FB7
## 1959896  33DF85D1E42AFEDE9F41AF25BD5AA8DE CFAF596405815111456E3BF8FB894CB9
## 1976825  BBA88D9A32E1D89EB561D916178C2973 7834BF9CB1A4560DE1CC735D080F3506
## 2037020  CD04870BCC7E31B793B50395DAB94A67 03173DD93C1171DA1788E6E7D733C5A9
## 2043804  496EDB4C31E23FF53B8A4043DB1D4935 DC97D6F5C0C997E0A86CDE92DDEA4E79
## 2111755  B2D8C705FD8A40CA6AC672CDB8750554 42FFD4341467F49AF3EA22E49403A07A
## 2255996  12DB9C4237F50D77F4D529C308E404DE DB84F30B78CD4B7E6C83F596696524C1
## 2323592  2770D5C78890742C0B7D7D07B0AA2628 025E47DB1154B924D0B627018527A492
## 2361763  6C1440837F31B7E48A0BFEDA11BF7A7A EBF6C54ECBE5375D667928DA98FB51D8
## 2412387  37E89818EBA67B4F6D2F8BA6F10E9130 5A2889B971B2406922F79FE3B0DD858F
## 2499887  4A527F0CDB3DE35336DE7E8BF6A41597 7A87D7C44DCB249E39BB9CCEEE518DC8
## 3318210  CCDD7C317BBF35D4585CB9BC4F4299A5 A80601BDCC989F498D0EEDF9700EEA92
## 3589536  4256332648D4C386D2556EA5F320E0F6 127522C70D9ED9EC0837D13DE60C35B5
## 3624757  79AA245159446C0B73BD5CE5DB6228A5 22FF4478DA7859709D3C741D32B84527
## 3631463  E10567BBAD2900F3DA14BDBA064359F6 229E397B13A730B9868261F445CE8CC2
## 3866556  41AF0EB6FDFCBF0A0C08F3241411C6C1 D7770D1E7438E4C52F1419940DF6FD1B
## 3946581  19B4B9F2AA7D248A61DAE4B8EA9B5D32 16398F0B2173E801E79AF8B73603539E
## 3968957  41AF0EB6FDFCBF0A0C08F3241411C6C1 D7770D1E7438E4C52F1419940DF6FD1B
## 4027179  41AF0EB6FDFCBF0A0C08F3241411C6C1 D7770D1E7438E4C52F1419940DF6FD1B
## 4117579  BF9CE67F2D9782421BD8F39B4C0B15EA E2BC338B48D98696D83076CFA9952B48
## 4142923  FC33D28148829383E2501520F7FDBF00 98DA73BD447E5AA6BB517079E81C2EB3
## 4296178  BF9CE67F2D9782421BD8F39B4C0B15EA E2BC338B48D98696D83076CFA9952B48
## 4385585  BF9CE67F2D9782421BD8F39B4C0B15EA E2BC338B48D98696D83076CFA9952B48
## 5043904  817030069CADBD7EC6F46F4BE098C5C0 7D12F47C8143410209F5A7C1FAE696F1
## 5107650  BF9D136E71CE1E93F3779316EA978AC7 2A763C9CE34866EE2155186CAD87C349
## 5165134  3DF0F9565A7E26AA877CDB83DD64F686 F994D1A3A4FBC5C43D812392165643B9
## 5173960  BDAB957909D0FA11D183578780C9F952 56DA2C27D9FEA8FCF6E0BCDF9B0A95D8
## 5220276  91FB7A9AF909FBB98BC3B2599E48EE45 A85049CDA20A4FCF08328FE941583F78
## 5973528  493E7E2C8EFBF7BA962F9BE5793596AE 8EEF8462A6AF95FBBB4C2B36530D8230
## 6000128  D86D7A22BADAEE364C83543982CF291A 181677C1AF2A8057A970E51F57B22D9D
## 6032392  5514F9B0D68216BDF8BAABBF45947408 1751BF76DC0F3CD5F7445A31F2FD0D38
## 6041668  8F160024B5E4CE5B7859AE20A4CC26D5 217622C8CAF6C45CB026B087DECC76F9
## 6053094  47AF1F2AEEF911FD2C5372AD3789062F 5F2DD1D6ABBDE2A5F09ADEBEAEBE2D97
## 6053095  47AF1F2AEEF911FD2C5372AD3789062F 5F2DD1D6ABBDE2A5F09ADEBEAEBE2D97
## 6053131  47AF1F2AEEF911FD2C5372AD3789062F 5F2DD1D6ABBDE2A5F09ADEBEAEBE2D97
## 6076281  BD3A6AD773AE2424DD1C7B1B7953E1B2 2B0BC139A0CB31D7B4994134C72D24DE
## 6110615  79226DFE9FE8214391EE99E9D6DFBE48 4426CD8259AD07358D508A326D3166B0
## 6172247  F9B3A00E6DDCA4F8BF2560DFF36B9E91 4EC477CA44F452A8134CE29860D4943F
## 6184196  E8950C1BF2CE598E95DE0CA5C89FA4BA 1A430147032A13B52BBF2E2F32CC2601
## 6202182  F06792C341B430BB69D3E02652389D6F 53E246419A3C88306FE9345E4CF8B758
## 6212986  58B3E6D45824A064F73A59B518F7DEBD CB5E2BA61B22934D3A7734C04C6EC4C2
```

```
## 6316632  647F72E0DB27329BEDE30230F6556EEF A5BFADA8A2232F80DF1099696BDC44E4
## 6326159  F16C1D7D394AF6A92163374ACB2F9AA8 649E658FC98CECF35E3F6BDBC5588042
## 6402677  D86D7A22BADAEE364C83543982CF291A D055A3A8EB9EE2DA18F4042909DA86CF
## 6431708  85EAFA124BA0D84BF918C722F0F5A23F 5E10E86C01ED37A484DFC3B299E0FA00
## 6462860  5D67A37D45DFCF93CB88B967C895C2AC 8BE7F26C050D95A3E46C6D88820B0A2E
## 6490673  47FA502B7576D751026A1A768A8FC43F 7ECF227031E6FC22CD582A9B91072D20
## 6511647  7997435A067169AB55EF7D77860D1024 91E42C14BCB350AD35A7B66951A88214
## 6543783  2718D0E7E0ED73571A2871C1AB9EA8DB 42D0A8359FEE9F6E84A93FBF5EFC283E
## 6545662  664D72E849CB0CB88BEE21BC9B3AF670 D4A63AD0396C3384D3B4EA4DA19613BE
## 6555412  B085C7CC04E3A80D137318997A8B494E 882790FD2DD2A0E4C26631567B7F6FB7
## 6578716  493E7E2C8EFBF7BA962F9BE5793596AE 8EEF8462A6AF95FBBB4C2B36530D8230
## 6591273  D2868A11E1599F3B4EC71DC681D7CDAF 42D0A8359FEE9F6E84A93FBF5EFC283E
## 6621967  63B83A1BE40DA227590EADEDFA1C7055 6AE05F6230868B6485FB84160DEC7BA3
## 6671012  53DDE4727836BA6785779499FF125E11 42059097E969E530D841BAE562FA488F
## 6678727  FBB7CB930BA1070F2E0613747440BA47 0BF34EE886B8B0AF0021D85CA959B0CC
## 6705485  B7F13019A79344334763CEF1BAE6D999 0B98E663F56F1700252319B6F0C175B1
## 6755663  580ECB971AF96627B0599CD5625AFA70 3B188C4784F670E10A905727ED8A9605
## 6779704  33DBCE3EEFEFD4052E0D14EA5D2C80FB 42D0A8359FEE9F6E84A93FBF5EFC283E
## 6791350  6486AC7204FA949CFA6D9F3BD738462E DEF77D27DAD5B20544477BCC68B5CEB7
## 7309896  A6E564506ED79015979D93114307B3CF B9B46AD5D6A3BDDE5066F7654C43658A
## 7393297  6106B9F2DF3A09E558B562E93D00B859 2DC689F7F2CCA8C13A6E808C99866E88
## 7519105  0B917B832C0116C40FEB0348A7B886E1 1D045728215EA3DE337833B85FC8379D
## 7727428  09DB0A922923E5C0EBDF09CF09C015B9 2ABD558A2427DEA5687C90050676260F
## 7732917  A28E1DA7789874A0F2533FA3981520FF 494E05CF7E12FCC7DAEDEFA6E1491BF5
## 7771581  747F37FEBB61C962235EC4486F096AD8 3210751878F1A59576FE3E2E0310F62D
## 7783128  A2B53A64004799B6FFC7F0FF2785AB73 0AD4E678D08C3BF876391DFB0BDA793F
## 7892259  D83874011C6CE4073DD028506956A4A0 74E3D1DABE438F55F7B945FC4ABAFBB6
## 8004992  A2B53A64004799B6FFC7F0FF2785AB73 0AD4E678D08C3BF876391DFB0BDA793F
## 8035307  CD098C57F2BC9DC180FCD92C4B37F08B 095851F44293BE8E5530FF64CC695CF9
## 8136232  CFC5047A760925F89DAF0326C9B2B89B 7930AF3429B99C59346390635AFDFFCF
## 8230991  0B917B832C0116C40FEB0348A7B886E1 1D045728215EA3DE337833B85FC8379D
## 8260660  8CCC1CF4D81808ECCACC59F5E3A449CC 9EC95F057C945A435B185F0B2D488F5D
## 8267089  554E4199E8B26EA94C3B1B158735A647 5C043E26BEF08C253F344691FC490C4D
## 8301396  E184081732527012A9227BE0D55F8DC4 53F43ED5DC3764322F2FFE91C7DC2841
## 8317533  6489FAF61426EED0D4B7467AF48B39F5 422997EBE4887BF1BF841BC50FE804C1
## 8333352  84883E3A8A6A7F0040E63BB1406F2BA7 45D260A074103E2853D4D5584678E191
## 8333817  A89281B0556AFE517F0295705E562805 4E0D757525B061890E4F31F80B3DF59A
## 8363338  CD4F6FCC28F932E66CD09DF340FFF16F 589B5C64B0A6D185CE8765C9DEE70D36
## 8364680  704A0EBF3CDB90215A25EB1E9A081640 C47C674CF4802073C12A624D6ABC79C7
## 8373109  7E9E4350AF2B8E606CACE1E05C75140B 00CFD01E9CE6530D39E5B5AFA4A55B6B
## 8414310  D02BD2AF3E81679ADE9AB2924BF5A306 AD692AB4CB07544B5C293E5E650CA334
## 8491188  F2D53D0905AE6A5E6957A5CC26D451AC D9D63902B78166A0CB6F33CA43C5D18F
## 8511023  8C1EED3FC560AC6AEEB3BD9E7C3753B5 A736D58B40AB94DB8346A56F6620F672
## 8523100  1FD758B30E864D2536032187A7F4A026 8BD304B834A571C38BF71DDB212ECA17
## 9071745  901F05620BEF6DB12454CF90190B0BE7 532724A7044919D63CA4A988AC091610
## 9155416  F2D53D0905AE6A5E6957A5CC26D451AC D9D63902B78166A0CB6F33CA43C5D18F
## 9372866  EB22735E26CA9FDF107D0D74EF6F0643 3C2CF3B7DAB74355D978B9E30E2EA3C8
## 9420861  CECFCD256DD02517B8D0BF64DAAEFDB8 1B683B2D81D38DA94C6B790E7F9CA5F9
## 9448168  23CB2BF8FABAB1F17948C209601C0B60 085BEB2B0AF57D98B1BA95E02ED9F9B2
## 9454371  0B917B832C0116C40FEB0348A7B886E1 1D045728215EA3DE337833B85FC8379D
## 9471426  71FFE6A4E97ECD83C4745123E3F563FB B51EA07359563F4E7507CBEC78A353A4
## 9520390  593114F98009023220132CA3BF4069BB 56A51E86701D0E8DF63A363065A3D465
## 9543066  6246DEC99C239C71FC0D79EA2C62C132 C1700AD63FFD7B5FD2C05B700D121135
## 9545551  E671FA2487A9A0F9DFA40A98690F57CE 354E938408B48C04B6B0317A18861284
```

```
## 9589183  6C1440837F31B7E48A0BFEDA11BF7A7A DD4F68C21143945779D5A771E233664A
## 9593164  830A001492E941D81089704FD9282265 B66933B8C7ED9BAC1486B568F013B80B
## 9603160  BCAA38AFBF6EAD8AA2F0007265F94540 C41FA9E5FB6DE35B2BFF9861E6D00A4D
## 9655875  E16D7A2452A7FD76F4EF522D94A5F8B7 0BA36B789DC617274903E8CCFDED72C2
## 9677302  9F6A867DBAA5F6AA71517F7C8EB25283 015E223FE62E0471D3945190F1D8DA59
## 9708469  0A5CC570753110C2E2B47FEF161CD4A0 34CB9D84E88CD581213EC49DAB2E7FAD
## 9726085  6A5C0BA17A78FCB1DC63932F8FBB92B7 1CC7F94BCC57F41DFBDE680D6984773C
## 9734388  82B08A622CF088AB06230D9913110FD8 11B8A9D20468A46323A7036C349063F9
## 9753385  8FC3B146F15F8B523688FF95C110B64A C1059E614E05082428C20540004D03CE
## 9817732  4CF38C0C5DE00C68B5AFFF44857FE359 DCBED2797C00AE1910E42B8F49532EB1
## 9864542  DA9A52DC3CFC80918A0F23EF7FB88AC3 361F8B8863C2CDD3D96435BBC45E19C0
## 9897393  B69326F5D825D6C350BF3E78F0FF2820 08122A3AF5DFA6B924B7BD9D0BFA65DB
## 9921326  934A402A229A191B3E4925837C2417DB 1BE935BB969FAD0F0CD471654397E19C
## 9927058  01A2F4366180AEB433600BAEA196BFC7 4BB683B0158059ED28493DCC4B601D1E
## 9935950  6E246E6EC58F295A0F140FFBF61A7997 BE5474B46DD3C703CC667F181A8FCB56
## 9970768  670501134C86B2DA67D4BD7D05C9444E 998B8560BB5685B4B50E264472B35F63
## 9983254  3EF3B2DD3AFBC8AABDBF2BD9F37AD66E 293BA90E87942967432152E409048257
## 9995334  C612423A09BF7298B817EBB674E7DC11 F3E5CC66E8544AF5FF67DFB70B6A7481
## 10042845 A9509245EB43F84F2F4DAD81A118E118 C15ABAE2763EDB16B34B177FD6173D1A
## 10076214 CD9DEF073BAB75B8B36015D85FD3F777 6F96BDD686CD937AB72C9446457D8B2D
## 10119407 7FFA9CB04343F2B0FEA0D081AB446D14 F50F314036309F93696AB2036B152B18
## 10128477 26F2F63E547B394377D5CF5EFDC3DEB0 4A0EE7BCFB7F975528ECA39D1F78F23A
## 10128702 08BAA2264C1C39B35299406F58E14598 F664D1573DECA9A5882310DA6E717337
## 10146842 738A62EEE9EC371689751A864C5EF811 9D8B31E5584117961016EF690F708721
## 10147505 830A001492E941D81089704FD9282265 B66933B8C7ED9BAC1486B568F013B80B
## 10156003 5C7FDA34C90CE9041A94AFEFC8E2E4A3 76265148A2D6C543C888EEA99CA3B0A8
## 10157521 5C7FDA34C90CE9041A94AFEFC8E2E4A3 76265148A2D6C543C888EEA99CA3B0A8
## 10225137 312281B9FC2886045B4CBD195134AD1E 16530F590955B12BEE2E0B49CC243944
## 10392990 0B917B832C0116C40FEB0348A7B886E1 1D045728215EA3DE337833B85FC8379D
## 10577947 A153217EB3C7B1E6B725C8C23637D148 B59C67BF196A4758191E42F76670CEBA
## 10620862 AE124FC3210EE986C339A4C4FB225F4C FEF547284023C46F0FC4C68589F22D20
## 10927728 0E852B1B3AA2D9CFD7485FC6851670E4 35BB1F95ECB0410F12C2149762EA0065
## 10933709 F6D6BAD72F4E5BAE1E2BD2A366ADAAED 5B6720867330E44817E65D7763C8DBF7
## 10937080 B509C8D65CBFE7F3CCED1FCA8149AE5F 2918196F31AC533D06001C497D4BBE8F
## 10952862 841C9AC0D0A40C9EE661BEE9BDA5E3EA 642F39616115312483D3B60D3D08FE11
## 10961297 4FE5B5954B70BB622DBE3BBE9B1DE80E 4ECBF8D37AB32F8370B25F2371E58A31
## 10963022 4E578E32AE8490FACF9C1B921C73D885 2EEE74881D1A0286EE1E1C2C948DBB35
## 10968127 E419F2DC059F6BF62988FFD9EAD5E50C 106B3A0124118401F4795EB8AAA1372E
## 10970205 B976C2299905CE417E1B19AD6C903DD6 3603DB426B6C9596DACF709F9D82F65E
## 10980635 7449F646469E47646D001F3BDED30E87 124D4D0FB3133DA14B96AD270E5CE090
## 11009831 83D8E776A05EEF731E4617F3CC9A1572 4F9E0E34F1A156EB3656254BE374A706
## 11013500 1171F2BCE81980A0E5B3C1612FFCF751 FD1CEF2DFA90840094EA57C06EE8224F
## 11023069 D2A54CCEA77527E69E6C65125CACD01D 53BC985CB63CB82BEDEE11FF2646E3BC
## 11029601 214DFEA924058D63E6F5B3F34E2B8162 BE2A54E64846E15CEB203434384EBDE7
## 11034800 312281B9FC2886045B4CBD195134AD1E 16530F590955B12BEE2E0B49CC243944
## 11044741 214DFEA924058D63E6F5B3F34E2B8162 BE2A54E64846E15CEB203434384EBDE7
## 11168642 8C1EED3FC560AC6AEEB3BD9E7C3753B5 A736D58B40AB94DB8346A56F6620F672
## 11784692 2A206801A42E7E225C917753FC0DD27D 32DDCDE76F12CFC1EB8E4B900E85CD13
## 12006895 2DA12E2776B48AB3C8EBE6B57B2A0730 C22E9D0CDCA33673631C1A1BF5A164AA
## 12114093 CEA00F3B61EF2424003312811E45B2FC 5C2B640585C5E49CE49BBA52AFB74AD9
## 12837882 41AF0EB6FDFCBF0A0C08F3241411C6C1 D7770D1E7438E4C52F1419940DF6FD1B
## 13021653 9E28C3598E3575951F8AD16241E5B874 0AD9418CD28ADD6707DD4854178722B5
## 13298838 8F0A1E787329C1C08F8F9120EE68E056 38AE6BD31887EC5F00446498413C14D9
## 13368364 922EFFD83A639FEB38CC41E0BB716DB5 C92B2C2D1E81AB10FC227010FF02433A
```

```
## 13369056 E5B70E847C120AC30EDA169887E95B61 4B846DFC0A35F4FEEC77F702866B4F3F
## 13383913 41AF0EB6FDFCBF0A0C08F3241411C6C1 D7770D1E7438E4C52F1419940DF6FD1B
## 13437875 8F0A1E787329C1C08F8F9120EE68E056 38AE6BD31887EC5F00446498413C14D9
## 13451302 8F0A1E787329C1C08F8F9120EE68E056 38AE6BD31887EC5F00446498413C14D9
## 13456241 39EFE76F7CFC78A43920607D812D9E40 9BEE4F56F4BB7226181A2AC52196A0F3
## 13698146 79AA245159446C0B73BD5CE5DB6228A5 22FF4478DA7859709D3C741D32B84527
## 13862322 BF9D136E71CE1E93F3779316EA978AC7 2A763C9CE34866EE2155186CAD87C349
##          vendor_id rate_code store_and_fwd_flag     pickup_datetime
## 38134          VTS         5                    2013-09-09 00:33:00
## 126139         CMT         0                    2013-09-07 11:37:45
## 673262         VTS         5                    2013-09-14 01:08:00
## 987582         VTS         5                    2013-09-11 15:10:00
## 1068387        VTS         5                    2013-09-14 18:14:00
## 1096146        VTS         5                    2013-09-11 19:06:00
## 1197346        VTS         5                    2013-09-14 20:23:00
## 1476721        VTS         5                    2013-09-15 03:35:00
## 1758222        CMT         5                  N 2013-09-05 12:47:45
## 1776226        CMT         0                    2013-09-02 17:15:47
## 1821255        CMT         5                  N 2013-09-02 18:04:53
## 1886968        CMT         5                  N 2013-09-03 03:07:24
## 1959896        CMT         0                    2013-09-05 00:02:14
## 1976825        CMT         5                  N 2013-09-05 16:40:45
## 2037020        CMT         5                  N 2013-09-03 00:10:25
## 2043804        CMT         1                  N 2013-09-03 20:39:15
## 2111755        CMT         0                    2013-09-04 09:55:16
## 2255996        CMT         5                  N 2013-09-02 08:02:13
## 2323592        CMT         0                    2013-09-08 02:19:26
## 2361763        CMT         0                    2013-09-08 10:43:24
## 2412387        CMT         5                  N 2013-09-03 13:53:47
## 2499887        CMT         5                  N 2013-09-05 00:00:56
## 3318210        VTS         5                    2013-09-20 15:02:00
## 3589536        VTS         5                    2013-09-16 12:26:00
## 3624757        VTS         5                    2013-09-19 01:24:00
## 3631463        VTS         5                    2013-09-21 00:33:00
## 3866556        VTS         5                    2013-09-17 18:07:00
## 3946581        VTS         5                    2013-09-20 15:13:00
## 3968957        VTS         5                    2013-09-17 21:36:00
## 4027179        VTS         5                    2013-09-18 18:14:00
## 4117579        VTS         5                    2013-09-16 12:26:00
## 4142923        VTS         5                    2013-09-19 02:57:00
## 4296178        VTS         5                    2013-09-17 18:31:00
## 4385585        VTS         5                    2013-09-18 14:46:00
## 5043904        VTS         5                    2013-09-25 18:51:00
## 5107650        VTS         5                    2013-09-25 19:22:00
## 5165134        VTS         5                    2013-09-23 14:18:00
## 5173960        VTS         5                    2013-09-25 23:46:00
## 5220276        VTS         5                    2013-09-26 02:36:00
## 5973528        CMT         0                    2013-09-14 18:50:20
## 6000128        CMT         0                    2013-09-10 18:44:47
## 6032392        CMT         0                    2013-09-11 21:01:59
## 6041668        CMT         0                    2013-09-09 19:19:41
## 6053094        CMT         1                  N 2013-09-09 01:39:59
## 6053095        CMT         1                  N 2013-09-09 01:12:01
## 6053131        CMT         1                  N 2013-09-09 01:58:03
```

```
## 6076281        CMT         0                   2013-09-13 21:11:44
## 6110615        CMT         0                   2013-09-13 22:24:07
## 6172247        CMT         0                   2013-09-12 18:18:10
## 6184196        CMT         0                   2013-09-09 06:58:43
## 6202182        CMT         0                   2013-09-11 08:44:23
## 6212986        CMT         0                   2013-09-13 11:24:32
## 6316632        CMT         5                 N 2013-09-15 15:45:27
## 6326159        CMT         0                   2013-09-13 06:26:50
## 6402677        CMT         0                   2013-09-10 21:19:30
## 6431708        CMT         0                   2013-09-11 08:50:06
## 6462860        CMT         0                   2013-09-11 21:43:56
## 6490673        CMT         0                   2013-09-10 07:45:00
## 6511647        CMT         5                 N 2013-09-12 22:48:42
## 6543783        CMT         5                 N 2013-09-09 05:17:54
## 6545662        CMT         0                   2013-09-13 11:12:43
## 6555412        CMT         5                 N 2013-09-14 04:52:00
## 6578716        CMT         0                   2013-09-14 20:20:48
## 6591273        CMT         5                 N 2013-09-14 06:52:22
## 6621967        CMT         0                   2013-09-10 08:22:25
## 6671012        CMT         0                   2013-09-13 00:45:47
## 6678727        CMT         0                   2013-09-13 06:32:08
## 6705485        CMT         5                 N 2013-09-14 00:44:58
## 6755663        CMT         5                 N 2013-09-11 10:08:48
## 6779704        CMT         5                 N 2013-09-15 07:05:58
## 6791350        CMT         5                 N 2013-09-09 07:34:39
## 7309896        CMT         1                 N 2013-09-14 11:55:45
## 7393297        CMT         1                 N 2013-09-13 22:11:25
## 7519105        CMT         5                 N 2013-09-14 05:06:43
## 7727428        CMT         5                 N 2013-09-20 10:13:00
## 7732917        CMT         5                 N 2013-09-21 05:25:56
## 7771581        CMT         5                 N 2013-09-18 15:52:04
## 7783128        CMT         5                 N 2013-09-21 00:25:26
## 7892259        CMT         0                   2013-09-16 14:14:54
## 8004992        CMT         5                 N 2013-09-22 17:13:32
## 8035307        CMT         0                   2013-09-20 18:08:19
## 8136232        CMT         0                   2013-09-19 14:40:39
## 8230991        CMT         5                 N 2013-09-16 17:10:16
## 8260660        CMT         5                 N 2013-09-17 23:16:31
## 8267089        CMT         0                   2013-09-18 17:54:25
## 8301396        CMT         0                   2013-09-21 09:27:51
## 8317533        CMT         5                 N 2013-09-19 03:10:37
## 8333352        CMT         0                   2013-09-19 10:39:12
## 8333817        CMT         0                   2013-09-18 08:28:32
## 8363338        CMT         0                   2013-09-19 17:23:39
## 8364680        CMT         5                 N 2013-09-21 20:39:30
## 8373109        CMT         0                   2013-09-22 00:31:29
## 8414310        CMT         0                   2013-09-19 23:27:16
## 8491188        CMT         1                 N 2013-09-21 20:06:11
## 8511023        CMT         5                 N 2013-09-22 23:46:03
## 8523100        CMT         0                   2013-09-20 08:29:48
## 9071745        CMT         1                 N 2013-09-21 13:13:27
## 9155416        CMT         1                 N 2013-09-21 19:46:38
## 9372866        CMT         0                   2013-09-26 00:53:54
## 9420861        CMT         0                   2013-09-24 15:30:53
```

```
## 9448168       CMT         0                   2013-09-23 11:50:28
## 9454371       CMT         5                 N 2013-09-28 21:00:26
## 9471426       CMT         0                   2013-09-26 11:19:24
## 9520390       CMT         0                   2013-09-26 00:28:37
## 9543066       CMT         0                   2013-09-27 22:52:43
## 9545551       CMT         5                 N 2013-09-27 00:02:55
## 9589183       CMT         5                 N 2013-09-28 10:16:31
## 9593164       CMT         0                   2013-09-23 13:50:01
## 9603160       CMT         0                   2013-09-23 07:50:26
## 9655875       CMT         5                 N 2013-09-26 00:05:55
## 9677302       CMT         0                   2013-09-25 15:49:35
## 9708469       CMT         0                   2013-09-25 16:36:44
## 9726085       CMT         0                   2013-09-27 07:58:32
## 9734388       CMT         0                   2013-09-28 02:30:07
## 9753385       CMT         0                   2013-09-24 05:25:45
## 9817732       CMT         0                   2013-09-28 06:38:34
## 9864542       CMT         5                 N 2013-09-29 07:36:27
## 9897393       CMT         0                   2013-09-26 19:56:48
## 9921326       CMT         5                 N 2013-09-27 04:25:19
## 9927058       CMT         0                   2013-09-25 09:30:38
## 9935950       CMT         5                 N 2013-09-29 15:33:52
## 9970768       CMT         0                   2013-09-26 05:57:08
## 9983254       CMT         0                   2013-09-28 15:32:23
## 9995334       CMT         0                   2013-09-25 21:40:18
## 10042845      CMT         0                   2013-09-29 01:41:01
## 10076214      CMT         0                   2013-09-26 05:54:20
## 10119407      CMT         0                   2013-09-28 04:55:38
## 10128477      CMT         0                   2013-09-26 05:55:04
## 10128702      CMT         5                 N 2013-09-25 07:38:38
## 10146842      CMT         0                   2013-09-27 11:04:21
## 10147505      CMT         0                   2013-09-26 06:39:30
## 10156003      CMT         0                   2013-09-24 20:45:10
## 10157521      CMT         0                   2013-09-26 17:49:54
## 10225137      CMT         1                 N 2013-09-24 17:58:42
## 10392990      CMT         5                 N 2013-09-23 14:42:21
## 10577947      CMT         0                   2013-09-27 16:16:03
## 10620862      CMT         3                 N 2013-09-25 14:20:55
## 10927728      CMT         0                   2013-09-30 08:48:12
## 10933709      CMT         0                   2013-09-30 18:55:37
## 10937080      CMT         0                   2013-09-30 02:04:36
## 10952862      CMT         0                   2013-09-30 02:05:01
## 10961297      CMT         0                   2013-09-30 20:34:40
## 10963022      CMT         0                   2013-09-30 02:04:35
## 10968127      CMT         0                   2013-09-30 08:59:01
## 10970205      CMT         0                   2013-09-30 02:04:57
## 10980635      CMT         0                   2013-09-30 04:50:39
## 11009831      CMT         0                   2013-09-30 06:59:54
## 11013500      CMT         5                 N 2013-09-30 21:03:59
## 11023069      CMT         0                   2013-09-30 02:04:48
## 11029601      CMT         0                   2013-09-30 02:04:56
## 11034800      CMT         1                 N 2013-09-30 16:41:24
## 11044741      CMT         0                   2013-09-30 01:22:40
## 11168642      CMT         5                 N 2013-09-01 03:26:20
## 11784692      VTS         5                   2013-09-04 07:48:00
```

```
## 12006895       VTS         5                   2013-09-02 02:48:00
## 12114093       VTS         5                   2013-09-04 02:46:00
## 12837882       VTS         5                   2013-09-06 20:44:00
## 13021653       VTS         1                   2013-09-02 16:09:00
## 13298838       VTS         5                   2013-09-27 21:40:00
## 13368364       VTS         5                   2013-09-28 01:36:00
## 13369056       VTS         5                   2013-09-28 01:09:00
## 13383913       VTS         5                   2013-09-28 02:18:00
## 13437875       VTS         5                   2013-09-28 04:23:00
## 13451302       VTS         5                   2013-09-28 03:58:00
## 13456241       VTS         5                   2013-09-28 04:06:00
## 13698146       VTS         5                   2013-09-29 02:29:00
## 13862322       VTS         5                   2013-09-29 20:05:00
##             dropoff_datetime passenger_count trip_time_in_secs trip_distance
## 38134    2013-09-09 00:34:00               0                60          0.00
## 126139   2013-09-07 15:37:45               0                 0          0.00
## 673262   2013-09-14 01:09:00               0                60          0.00
## 987582   2013-09-11 15:10:00               0                 0          0.00
## 1068387  2013-09-14 18:14:00               0                 0          0.00
## 1096146  2013-09-11 19:06:00               0                 0          0.00
## 1197346  2013-09-14 20:35:00               0               720          3.50
## 1476721  2013-09-15 03:35:00               0                 0          0.00
## 1758222  2013-09-05 12:48:01               0                16          0.00
## 1776226  2013-09-02 21:15:47               0                 0          0.00
## 1821255  2013-09-02 18:05:08               0                15          8.50
## 1886968  2013-09-03 03:07:45               0                20          0.00
## 1959896  2013-09-05 04:02:14               0                 0          0.00
## 1976825  2013-09-05 16:42:17               0                92          0.00
## 2037020  2013-09-03 00:12:20               0               115         14.40
## 2043804  2013-09-03 20:51:46               0               750          2.90
## 2111755  2013-09-04 13:55:16               0                 0          0.00
## 2255996  2013-09-02 08:47:02               0              2689         15.20
## 2323592  2013-09-08 06:19:26               0                 0          0.00
## 2361763  2013-09-08 14:43:24               0                 0          0.00
## 2412387  2013-09-03 13:53:49               0                 2         11.60
## 2499887  2013-09-05 00:00:58               0                 2          0.00
## 3318210  2013-09-20 15:02:00               0                 0          0.00
## 3589536  2013-09-16 12:26:00               0                 0          0.00
## 3624757  2013-09-19 01:24:00               0                 0          0.00
## 3631463  2013-09-21 00:33:00               0                 0          0.00
## 3866556  2013-09-17 18:08:00               0                60          0.00
## 3946581  2013-09-20 15:38:00               0              1500         12.70
## 3968957  2013-09-17 21:36:00               0                 0          0.00
## 4027179  2013-09-18 18:14:00               0                 0          0.00
## 4117579  2013-09-16 12:26:00               0                 0          0.00
## 4142923  2013-09-19 02:57:00               0                 0          0.00
## 4296178  2013-09-17 18:31:00               0                 0          0.00
## 4385585  2013-09-18 14:46:00               0                 0          0.00
## 5043904  2013-09-25 18:51:00               0                 0          0.00
## 5107650  2013-09-25 19:22:00               0                 0          0.00
## 5165134  2013-09-23 14:18:00               0                 0          0.00
## 5173960  2013-09-25 23:46:00               0                 0          0.00
## 5220276  2013-09-26 02:36:00               0                 0          0.00
## 5973528  2013-09-14 22:50:20               0                 0          0.00
```

```
## 6000128  2013-09-10 22:44:47              0                0          0.00
## 6032392  2013-09-12 01:01:59              0                0          0.00
## 6041668  2013-09-09 23:19:41              0                0          0.00
## 6053094  2013-09-09 01:47:24              0              445          1.40
## 6053095  2013-09-09 01:26:28              0              867          3.00
## 6053131  2013-09-09 02:27:32              0             1768         11.80
## 6076281  2013-09-14 01:11:44              0                0          0.00
## 6110615  2013-09-14 02:24:07              0                0          0.00
## 6172247  2013-09-12 22:18:10              0                0          0.00
## 6184196  2013-09-09 10:58:43              0                0          0.00
## 6202182  2013-09-11 12:44:23              0                0          0.00
## 6212986  2013-09-13 15:24:32              0                0          0.00
## 6316632  2013-09-15 16:09:38              0             1450         12.70
## 6326159  2013-09-13 10:26:50              0                0          0.00
## 6402677  2013-09-10 01:19:30              0                0          0.00
## 6431708  2013-09-11 12:50:06              0                0          0.00
## 6462860  2013-09-12 01:43:56              0                0          0.00
## 6490673  2013-09-10 11:45:00              0                0          0.00
## 6511647  2013-09-12 22:48:55              0               13          1.00
## 6543783  2013-09-09 05:22:19              0              264          0.30
## 6545662  2013-09-13 15:12:43              0                0          0.00
## 6555412  2013-09-14 04:52:16              0               16          0.00
## 6578716  2013-09-14 00:20:48              0                0          0.00
## 6591273  2013-09-14 06:52:30              0                7          0.00
## 6621967  2013-09-10 12:22:25              0                0          0.00
## 6671012  2013-09-13 04:45:47              0                0          0.00
## 6678727  2013-09-13 10:32:08              0                0          0.00
## 6705485  2013-09-14 00:45:05              0                7          5.30
## 6755663  2013-09-11 10:08:52              0                3          3.30
## 6779704  2013-09-15 07:06:13              0               14          0.00
## 6791350  2013-09-09 07:35:31              0               52          0.30
## 7309896  2013-09-14 12:00:16              0              270          0.70
## 7393297  2013-09-13 22:11:25              0                0          0.00
## 7519105  2013-09-14 05:18:27              0              704          4.70
## 7727428  2013-09-20 10:41:59              0             1738         13.50
## 7732917  2013-09-21 05:25:57              0                0          1.20
## 7771581  2013-09-18 17:15:50              0             5026         17.80
## 7783128  2013-09-21 00:25:34              0                7          0.00
## 7892259  2013-09-16 18:14:54              0                0          0.00
## 8004992  2013-09-22 17:13:41              0                8          0.00
## 8035307  2013-09-20 22:08:19              0                0          0.00
## 8136232  2013-09-19 18:40:39              0                0          0.00
## 8230991  2013-09-16 18:09:06              0             3529         38.20
## 8260660  2013-09-17 23:38:47              0             1336          8.10
## 8267089  2013-09-18 21:54:25              0                0          0.00
## 8301396  2013-09-21 13:27:51              0                0          0.00
## 8317533  2013-09-19 03:10:40              0                3          1.70
## 8333352  2013-09-19 14:39:12              0                0          0.00
## 8333817  2013-09-18 12:28:32              0                0          0.00
## 8363338  2013-09-19 21:23:39              0                0          0.00
## 8364680  2013-09-21 20:39:49              0               18          0.00
## 8373109  2013-09-22 04:31:29              0                0          0.00
## 8414310  2013-09-20 03:27:16              0                0          0.00
## 8491188  2013-09-21 20:27:20              0             1269          4.20
```

```
## 8511023  2013-09-22 23:46:23              0               19          2.10
## 8523100  2013-09-20 12:29:48              0                0          0.00
## 9071745  2013-09-21 13:57:53              0             2665         28.10
## 9155416  2013-09-21 19:46:38              0                0          0.00
## 9372866  2013-09-26 04:53:54              0                0          0.00
## 9420861  2013-09-24 19:30:53              0                0          0.00
## 9448168  2013-09-23 15:50:28              0                0          0.00
## 9454371  2013-09-28 21:21:55              0             1288          5.70
## 9471426  2013-09-26 15:19:24              0                0          0.00
## 9520390  2013-09-26 04:28:37              0                0          0.00
## 9543066  2013-09-28 02:52:43              0                0          0.00
## 9545551  2013-09-27 00:02:58              0                3          0.00
## 9589183  2013-09-28 10:16:53              0               22         16.90
## 9593164  2013-09-23 17:50:01              0                0          0.00
## 9603160  2013-09-23 11:50:26              0                0          0.00
## 9655875  2013-09-26 00:06:52              0               56          0.00
## 9677302  2013-09-25 19:49:35              0                0          0.00
## 9708469  2013-09-25 20:36:44              0                0          0.00
## 9726085  2013-09-27 11:58:32              0                0          0.00
## 9734388  2013-09-28 06:30:07              0                0          0.00
## 9753385  2013-09-24 09:25:45              0                0          0.00
## 9817732  2013-09-28 10:38:34              0                0          0.00
## 9864542  2013-09-29 07:36:56              0               29          0.00
## 9897393  2013-09-26 23:56:48              0                0          0.00
## 9921326  2013-09-27 04:25:39              0               19          0.00
## 9927058  2013-09-25 13:30:38              0                0          0.00
## 9935950  2013-09-29 15:34:15              0               23         22.80
## 9970768  2013-09-26 09:57:08              0                0          0.00
## 9983254  2013-09-28 19:32:23              0                0          0.00
## 9995334  2013-09-26 01:40:18              0                0          0.00
## 10042845 2013-09-29 05:41:01              0                0          0.00
## 10076214 2013-09-26 09:54:20              0                0          0.00
## 10119407 2013-09-28 08:55:38              0                0          0.00
## 10128477 2013-09-26 09:55:04              0                0          0.00
## 10128702 2013-09-25 07:38:47              0                9         27.30
## 10146842 2013-09-27 15:04:21              0                0          0.00
## 10147505 2013-09-26 10:39:30              0                0          0.00
## 10156003 2013-09-25 00:45:10              0                0          0.00
## 10157521 2013-09-26 21:49:54              0                0          0.00
## 10225137 2013-09-24 18:07:35              0              532          1.70
## 10392990 2013-09-23 15:40:19              0             3477         18.70
## 10577947 2013-09-27 20:16:03              0                0          0.00
## 10620862 2013-09-25 14:59:47              0             2331         17.70
## 10927728 2013-09-30 12:48:12              0                0          0.00
## 10933709 2013-09-30 22:55:37              0                0          0.00
## 10937080 2013-09-30 06:04:36              0                0          0.00
## 10952862 2013-09-30 06:05:01              0                0          0.00
## 10961297 2013-10-01 00:34:40              0                0          0.00
## 10963022 2013-09-30 06:04:35              0                0          0.00
## 10968127 2013-09-30 12:59:01              0                0          0.00
## 10970205 2013-09-30 06:04:57              0                0          0.00
## 10980635 2013-09-30 08:50:39              0                0          0.00
## 11009831 2013-09-30 10:59:54              0                0          0.00
## 11013500 2013-09-30 21:05:39              0               99          0.10
```

```
## 11023069 2013-09-30 06:04:48              0                0          0.00
## 11029601 2013-09-30 06:04:56              0                0          0.00
## 11034800 2013-09-30 16:44:23              0              178          0.60
## 11044741 2013-09-30 05:22:40              0                0          0.00
## 11168642 2013-09-01 03:26:47              0               26         10.30
## 11784692 2013-09-04 07:48:00              0                0          0.00
## 12006895 2013-09-02 03:15:00              0             1620          0.91
## 12114093 2013-09-04 02:46:00              0                0          0.00
## 12837882 2013-09-06 21:26:00              0             2520         23.07
## 13021653 2013-09-02 16:16:00              0              420          1.81
## 13298838 2013-09-27 21:40:00              0                0          0.00
## 13368364 2013-09-28 01:36:00              0                0          0.00
## 13369056 2013-09-28 01:09:00              0                0          0.00
## 13383913 2013-09-28 02:18:00              0                0          0.00
## 13437875 2013-09-28 04:23:00              0                0          0.22
## 13451302 2013-09-28 03:58:00              0                0          0.00
## 13456241 2013-09-28 04:07:00              0               60          0.37
## 13698146 2013-09-29 02:29:00              0                0          0.00
## 13862322 2013-09-29 20:05:00              0                0          0.00
##          pickup_longitude pickup_latitude dropoff_longitude dropoff_latitude
## 38134           -74.03187        40.74352         -74.03183         40.74350
## 126139          -73.99086        40.76609                NA                NA
## 673262          -74.04594        40.71960                NA                NA
## 987582                NA              NA                NA                NA
## 1068387               NA              NA                NA                NA
## 1096146               NA              NA                NA                NA
## 1197346         -74.00933        40.71305         -74.03723         40.73888
## 1476721               NA              NA                NA                NA
## 1758222         -74.17741        40.69076         -74.17741         40.69076
## 1776226         -74.00221        40.74692                NA                NA
## 1821255         -73.85777        40.73747         -73.85776         40.73748
## 1886968         -73.94124        40.72216         -73.94124         40.72216
## 1959896         -73.99048        40.75566                NA                NA
## 1976825         -73.54739        41.03518         -73.54739         41.03518
## 2037020         -74.05086        40.71801         -74.05090         40.71803
## 2043804         -73.98865        40.74595         -73.97931         40.71378
## 2111755         -73.98093        40.75314                NA                NA
## 2255996         -74.00562        40.74044         -74.17722         40.69493
## 2323592               NA              NA                NA                NA
## 2361763         -74.00173        40.74585                NA                NA
## 2412387         -74.06503        40.73641         -74.06503         40.73641
## 2499887         -74.04329        40.71908         -74.04329         40.71908
## 3318210               NA              NA                NA                NA
## 3589536               NA              NA                NA                NA
## 3624757               NA              NA                NA                NA
## 3631463               NA              NA                NA                NA
## 3866556               NA              NA                NA                NA
## 3946581         -74.00706        40.72505         -74.17748         40.69529
## 3968957               NA              NA                NA                NA
## 4027179               NA              NA                NA                NA
## 4117579               NA              NA                NA                NA
## 4142923               NA              NA                NA                NA
## 4296178               NA              NA                NA                NA
## 4385585               NA              NA                NA                NA
```

```
## 5043904               NA             NA               NA              NA
## 5107650               NA             NA               NA              NA
## 5165134               NA             NA               NA              NA
## 5173960               NA             NA               NA              NA
## 5220276               NA             NA               NA              NA
## 5973528         -73.96689       40.75672               NA              NA
## 6000128               NA             NA               NA              NA
## 6032392         -74.00629       40.72851               NA              NA
## 6041668         -73.97582       40.76017               NA              NA
## 6053094         -73.98109       40.76449         -73.98622       40.75178
## 6053095         -73.99115       40.74977         -73.98233       40.78267
## 6053131         -73.98441       40.75326         -73.89454       40.85988
## 6076281         -74.00558       40.74581               NA              NA
## 6110615         -73.99939       40.73160               NA              NA
## 6172247         -73.97839       40.74844               NA              NA
## 6184196         -73.78994       40.64668               NA              NA
## 6202182         -74.00736       40.70390               NA              NA
## 6212986         -73.98399       40.74635               NA              NA
## 6316632         -74.00581       40.72507         -74.17676       40.69421
## 6326159         -73.96853       40.79914               NA              NA
## 6402677         -73.98524       40.74558               NA              NA
## 6431708         -73.98333       40.76611               NA              NA
## 6462860         -73.99244       40.74341               NA              NA
## 6490673         -73.99503       40.76950               NA              NA
## 6511647         -73.72347       40.78620         -73.72348       40.78620
## 6543783         -73.86189       40.72901         -73.86413       40.73351
## 6545662         -73.98799       40.74373               NA              NA
## 6555412         -73.98782       40.80616         -73.98781       40.80615
## 6578716         -73.97406       40.79128               NA              NA
## 6591273         -74.00343       40.72401         -74.00343       40.72401
## 6621967         -73.95159       40.76983               NA              NA
## 6671012         -73.98440       40.75547               NA              NA
## 6678727         -73.98460       40.73637               NA              NA
## 6705485         -73.67447       40.70602         -73.67448       40.70602
## 6755663         -73.98530       40.77333         -73.98530       40.77333
## 6779704         -73.86175       40.72890         -73.86175       40.72890
## 6791350         -73.86706       40.77177         -73.87007       40.76996
## 7309896         -73.95558       40.78791         -73.95998       40.77977
## 7393297         -73.99067       40.73992               NA              NA
## 7519105         -74.00806       40.74699         -74.04121       40.72002
## 7727428         -74.00615       40.74038         -74.18265       40.68785
## 7732917         -74.07758       40.71982         -74.07758       40.71982
## 7771581         -73.99894       40.72641         -73.78230       40.64875
## 7783128         -73.98299       40.75600         -73.98299       40.75602
## 7892259               NA             NA               NA              NA
## 8004992         -74.15407       40.73827         -74.15408       40.73827
## 8035307               NA             NA               NA              NA
## 8136232         -73.86360       40.76989               NA              NA
## 8230991         -73.78956       40.64727         -74.04395       40.72327
## 8260660         -73.94754       40.79921         -74.02010       40.77776
## 8267089         -73.95332       40.79118               NA              NA
## 8301396         -73.95148       40.76985               NA              NA
## 8317533         -74.07368       40.60508         -74.07368       40.60508
## 8333352         -73.87189       40.77480               NA              NA
```

```
## 8333817        -73.95586       40.76420              NA             NA
## 8363338        -73.99013       40.73865              NA             NA
## 8364680        -73.94962       40.77269        -73.94964       40.77268
## 8373109        -74.28433       40.81750              NA             NA
## 8414310        -74.00717       40.72855              NA             NA
## 8491188        -73.98624       40.75499        -74.01471       40.71109
## 8511023        -74.03562       40.74099        -74.03563       40.74099
## 8523100        -73.94637       40.78091              NA             NA
## 9071745        -73.77682       40.64536        -73.89861       40.89737
## 9155416        -73.96217       40.77902              NA             NA
## 9372866        -74.00198       40.74032              NA             NA
## 9420861               NA             NA              NA             NA
## 9448168        -73.99963       40.72487              NA             NA
## 9454371        -73.97385       40.75423        -74.02988       40.74086
## 9471426        -73.99393       40.75087              NA             NA
## 9520390               NA             NA              NA             NA
## 9543066        -73.98289       40.73073              NA             NA
## 9545551        -73.96098       40.76875        -73.96098       40.76874
## 9589183        -73.98275       40.58398        -73.98274       40.58399
## 9593164        -74.01148       40.70275              NA             NA
## 9603160        -73.99386       40.74622              NA             NA
## 9655875        -74.03560       40.72483        -74.03558       40.72479
## 9677302        -73.97254       40.75003              NA             NA
## 9708469        -73.99112       40.73909              NA             NA
## 9726085        -73.98050       40.74312              NA             NA
## 9734388        -73.78230       40.64460              NA             NA
## 9753385        -73.93394       40.75241              NA             NA
## 9817732        -74.00221       40.73494              NA             NA
## 9864542        -73.79071       40.64661        -73.79071       40.64661
## 9897393        -73.97675       40.68215              NA             NA
## 9921326        -73.87058       40.77235        -73.87036       40.77355
## 9927058        -73.86379       40.76960              NA             NA
## 9935950        -74.07948       41.04322        -74.07946       41.04322
## 9970768        -73.97308       40.75637              NA             NA
## 9983254        -74.00423       40.70732              NA             NA
## 9995334        -73.99571       40.72316              NA             NA
## 10042845       -74.00262       40.73877              NA             NA
## 10076214       -74.00862       40.71999              NA             NA
## 10119407       -73.78889       40.64718              NA             NA
## 10128477       -73.98372       40.78082              NA             NA
## 10128702       -73.80274       40.67672        -73.80285       40.67693
## 10146842       -73.94898       40.74483              NA             NA
## 10147505       -74.00059       40.73249              NA             NA
## 10156003              NA             NA              NA             NA
## 10157521       -73.98367       40.72601              NA             NA
## 10225137       -73.99747       40.72442        -74.00346       40.72684
## 10392990       -74.17206       40.65912        -73.98178       40.75250
## 10577947       -73.94882       40.74492              NA             NA
## 10620862       -73.98625       40.74733        -74.17781       40.69558
## 10927728       -74.00652       40.73249              NA             NA
## 10933709       -73.98996       40.70236              NA             NA
## 10937080              NA             NA              NA             NA
## 10952862              NA             NA              NA             NA
## 10961297       -73.96179       40.76371              NA             NA
```

```
## 10963022              NA             NA               NA              NA
## 10968127        -73.96369       40.80850               NA              NA
## 10970205              NA             NA               NA              NA
## 10980635        -73.96742       40.75863               NA              NA
## 11009831        -73.96595       40.79036               NA              NA
## 11013500        -74.03181       40.75068        -74.03523       40.75111
## 11023069              NA             NA               NA              NA
## 11029601              NA             NA               NA              NA
## 11034800        -73.98144       40.75316        -73.98683       40.74538
## 11044741        -74.00234       40.72646               NA              NA
## 11168642        -74.00987       40.77980        -74.00985       40.77983
## 11784692              NA             NA               NA              NA
## 12006895        -73.99874       40.72055        -73.99868       40.72044
## 12114093              NA             NA               NA              NA
## 12837882        -74.01014       40.71050        -74.29768       40.76981
## 13021653        -73.97201       40.76115        -73.95191       40.77331
## 13298838        -74.18175       40.68776        -74.18175       40.68775
## 13368364        -73.99883       40.73567               NA              NA
## 13369056              NA             NA               NA              NA
## 13383913              NA             NA               NA              NA
## 13437875        -73.98793       40.73204        -73.98588       40.73523
## 13451302        -74.01708       40.64201        -74.01711       40.64202
## 13456241        -74.00211       40.73466        -74.00375       40.73186
## 13698146        -73.97839       40.74457               NA              NA
## 13862322              NA             NA               NA              NA
##          payment_type fare_amount surcharge mta_tax tip_amount tolls_amount
## 38134             CRD       50.00       0.0     0.0      10.00         0.00
## 126139            CRD       12.00       0.0     0.0       0.00         0.00
## 673262            CRD       50.00       0.0     0.0      10.00        10.25
## 987582            CSH       65.00       0.0     0.0       0.00         0.00
## 1068387           CSH       85.00       0.0     0.0       0.00         0.00
## 1096146           CRD      133.33       0.0     0.0      26.67         0.00
## 1197346           CRD       60.00       0.0     0.0      15.00         0.00
## 1476721           CRD       53.00       0.0     0.0      10.60         0.00
## 1758222           CRD       80.00       0.0     0.0      10.00         0.00
## 1776226           CRD        8.40       0.0     0.0       0.00         0.00
## 1821255           CRD       26.00       0.0     0.0       2.00         5.33
## 1886968           CRD       13.50       0.0     0.0       1.50         0.00
## 1959896           CRD       80.00       0.0     0.0       0.00         0.00
## 1976825           CRD      130.00       0.0     0.0       0.00         0.00
## 2037020           CRD       97.00       0.0     0.0      29.10         0.00
## 2043804           CRD       12.50       0.5     0.5       2.70         0.00
## 2111755           CRD       11.40       0.0     0.0       0.00         0.00
## 2255996           CRD       90.00       0.0     0.0       0.00         0.00
## 2323592           CRD       19.50       0.0     0.0       0.00         0.00
## 2361763           CRD       12.00       0.0     0.0       0.00         0.00
## 2412387           CRD       80.00       0.0     0.0      10.00         0.00
## 2499887           CRD       40.00       0.0     0.0       0.00         0.00
## 3318210           CRD      120.00       0.0     0.0      24.00         0.00
## 3589536           CRD       59.50       0.0     0.0      11.90         0.00
## 3624757           CRD       13.00       0.0     0.0       3.90         0.00
## 3631463           CSH       75.00       0.0     0.0       0.00         0.00
## 3866556           CRD       12.00       0.0     0.0       2.40         0.00
## 3946581           CRD       75.00       0.0     0.0      14.00        10.25
```

```
## 3968957          CRD       58.00       0.0     0.0      14.50         0.00
## 4027179          CRD       10.00       0.0     0.0       3.00         0.00
## 4117579          CSH       15.00       0.0     0.0       0.00         0.00
## 4142923          CRD        6.00       0.0     0.0       0.00         0.00
## 4296178          CRD       65.00       0.0     0.0       0.00         0.00
## 4385585          CRD       21.00       0.0     0.0       4.20         0.00
## 5043904          CRD       59.00       0.0     0.0      11.80         0.00
## 5107650          CRD      184.00       0.0     0.0       0.00         0.00
## 5165134          CRD       50.00       0.0     0.0       0.00         0.00
## 5173960          CRD        7.00       0.0     0.0       1.40         0.00
## 5220276          CRD        6.00       0.0     0.0       0.00         0.00
## 5973528          CRD       14.50       0.0     0.0       0.00         0.00
## 6000128          CRD       19.50       0.0     0.0       0.00         0.00
## 6032392          CRD       22.50       0.0     0.0       0.00         0.00
## 6041668          CRD       12.00       0.0     0.0       0.00         0.00
## 6053094          CRD        7.00       0.5     0.5       1.00         0.00
## 6053095          CRD       12.50       0.5     0.5       2.70         0.00
## 6053131          CRD       36.00       0.5     0.5       0.00         0.00
## 6076281          CRD       39.38       0.0     0.0       0.00         0.00
## 6110615          CRD       29.25       0.0     0.0       0.00         0.00
## 6172247          CRD       16.20       0.0     0.0       0.00         0.00
## 6184196          CRD       62.83       0.0     0.0       0.00         0.00
## 6202182          CRD       25.80       0.0     0.0       0.00         0.00
## 6212986          CRD       10.00       0.0     0.0       0.00         0.00
## 6316632          CRD       80.00       0.0     0.0      18.05        10.25
## 6326159          CRD       15.50       0.0     0.0       0.00         0.00
## 6402677          CRD        6.87       0.0     0.0       0.00         0.00
## 6431708          CRD       16.25       0.0     0.0       0.00         0.00
## 6462860          CRD       16.80       0.0     0.0       0.00         0.00
## 6490673          CRD       13.00       0.0     0.0       0.00         0.00
## 6511647          CRD       80.00       0.0     0.0      16.00         0.00
## 6543783          CRD       50.00       0.0     0.0      12.38         0.00
## 6545662          CRD       17.40       0.0     0.0       0.00         0.00
## 6555412          CRD       75.75       0.0     0.0      10.00         0.00
## 6578716          CRD        9.00       0.0     0.0       0.00         0.00
## 6591273          CRD       48.00       0.0     0.0       0.00         0.00
## 6621967          CRD       16.25       0.0     0.0       0.00         0.00
## 6671012          CRD       25.00       0.0     0.0       0.00         0.00
## 6678727          CRD        9.00       0.0     0.0       0.00         0.00
## 6705485          CRD       70.00       0.0     0.0       5.00         0.00
## 6755663          CRD       22.50       0.0     0.0       5.50         0.00
## 6779704          CRD       47.00       0.0     0.0       1.48         0.00
## 6791350          CRD       75.00       0.0     0.0      15.00         0.00
## 7309896          CSH        5.00       0.0     0.5       0.00         5.33
## 7393297          CSH        2.50       0.5     0.5       0.00         5.33
## 7519105          CSH       30.00       0.0     0.0       0.00         0.00
## 7727428          CRD       65.00       0.0     0.0      14.65         8.25
## 7732917          CRD       50.00       0.0     0.0       0.00         0.00
## 7771581          CRD       52.00       0.0     0.0       8.00         0.00
## 7783128          CRD       53.00       0.0     0.0       5.00         0.00
## 7892259          CRD       10.80       0.0     0.0       0.00         0.00
## 8004992          CRD       78.00       0.0     0.0       5.00         0.00
## 8035307          CRD        4.00       0.0     0.0       0.00         0.00
## 8136232          CRD       38.33       0.0     0.0       0.00         0.00
```

```
## 8230991          CRD      100.00       0.0     0.0      15.00         0.00
## 8260660          CRD       38.00       0.0     0.0       2.00         8.25
## 8267089          CRD       18.50       0.0     0.0       0.00         0.00
## 8301396          CRD        6.60       0.0     0.0       0.00         0.00
## 8317533          CRD       65.00       0.0     0.0       5.00         0.00
## 8333352          CRD       53.72       0.0     0.0       0.00         0.00
## 8333817          CRD       21.00       0.0     0.0       0.00         0.00
## 8363338          CRD        9.75       0.0     0.0       0.00         0.00
## 8364680          CRD        5.50       0.0     0.0       1.00         0.00
## 8373109          CRD      180.00       0.0     0.0       0.00         0.00
## 8414310          CRD       14.00       0.0     0.0       0.00         0.00
## 8491188          CRD       17.50       0.5     0.5       3.70         0.00
## 8511023          CRD       55.00       0.0     0.0       5.00         0.00
## 8523100          CRD       13.80       0.0     0.0       0.00         0.00
## 9071745          CSH       75.50       0.0     0.5       0.00         5.33
## 9155416          CSH        2.50       0.5     0.5       0.00         0.00
## 9372866          CRD       55.62       0.0     0.0       0.00         0.00
## 9420861          CRD        6.00       0.0     0.0       0.00         0.00
## 9448168          CRD        9.50       0.0     0.0       0.00         0.00
## 9454371          CRD       40.00       0.0     0.0       9.65         8.25
## 9471426          CRD       14.50       0.0     0.0       0.00         0.00
## 9520390          CRD       17.50       0.0     0.0       0.00         0.00
## 9543066          CRD       14.00       0.0     0.0       0.00         0.00
## 9545551          CRD       10.00       0.0     0.0       2.00         0.00
## 9589183          CRD       44.00       0.0     0.0       1.00         0.00
## 9593164          CRD       49.33       0.0     0.0       0.00         0.00
## 9603160          CRD       58.33       0.0     0.0       0.00         0.00
## 9655875          CRD       60.00       0.0     0.0       0.00         0.00
## 9677302          CRD       21.00       0.0     0.0       0.00         0.00
## 9708469          CRD       13.20       0.0     0.0       0.00         0.00
## 9726085          CRD       11.40       0.0     0.0       0.00         0.00
## 9734388          CRD       41.50       0.0     0.0       0.00         0.00
## 9753385          CRD        3.50       0.0     0.0       0.00         0.00
## 9817732          CRD       63.00       0.0     0.0       0.00         0.00
## 9864542          CRD       65.00       0.0     0.0       0.00         0.00
## 9897393          CRD       11.40       0.0     0.0       0.00         0.00
## 9921326          CRD       45.00       0.0     0.0       8.00         0.00
## 9927058          CRD       52.00       0.0     0.0       0.00         0.00
## 9935950          CRD      150.00       0.0     0.0      20.00         0.00
## 9970768          CRD       12.00       0.0     0.0       0.00         0.00
## 9983254          CRD       17.40       0.0     0.0       0.00         0.00
## 9995334          CRD       20.40       0.0     0.0       0.00         0.00
## 10042845         CRD       25.00       0.0     0.0       0.00         0.00
## 10076214         CRD       13.20       0.0     0.0       0.00         0.00
## 10119407         CRD       65.83       0.0     0.0       0.00         0.00
## 10128477         CRD       10.80       0.0     0.0       0.00         0.00
## 10128702         CRD      113.00       0.0     0.0      22.60         0.00
## 10146842         CRD        7.50       0.0     0.0       0.00         0.00
## 10147505         CRD       11.40       0.0     0.0       0.00         0.00
## 10156003         CRD       15.60       0.0     0.0       0.00         0.00
## 10157521         CRD       27.00       0.0     0.0       0.00         0.00
## 10225137         CSH        7.00       1.0     0.5       0.00         0.00
## 10392990         CSH       50.00       0.0     0.0       0.00         8.25
## 10577947         NOC        5.00       0.0     0.0       0.00         0.00
```

```
## 10620862          CSH       69.00       0.0     0.0       0.00         8.25
## 10927728          CRD       22.20       0.0     0.0       0.00         0.00
## 10933709          CRD       19.80       0.0     0.0       0.00         0.00
## 10937080          CRD        8.50       0.0     0.0       0.00         0.00
## 10952862          CRD       42.94       0.0     0.0       0.00         0.00
## 10961297          CRD        9.00       0.0     0.0       0.00         0.00
## 10963022          CRD       69.39       0.0     0.0       0.00         0.00
## 10968127          CRD       20.40       0.0     0.0       0.00         0.00
## 10970205          CRD       27.50       0.0     0.0       0.00         0.00
## 10980635          CRD       35.83       0.0     0.0       0.00         0.00
## 11009831          CRD       13.20       0.0     0.0       0.00         0.00
## 11013500          CRD       55.70       0.0     0.0       0.00         0.00
## 11023069          CRD        8.00       0.0     0.0       0.00         0.00
## 11029601          CRD       11.50       0.0     0.0       0.00         0.00
## 11034800          CSH        4.50       1.0     0.5       0.00         0.00
## 11044741          CRD       11.50       0.0     0.0       0.00         0.00
## 11168642          CRD       58.50       0.0     0.0      11.70         0.00
## 11784692          CRD       89.00       0.0     0.0      22.25         0.00
## 12006895          CRD        6.50       0.0     0.0       1.00         0.00
## 12114093          CRD        7.00       0.0     0.0       1.40         0.00
## 12837882          CRD      124.00       0.0     0.0      31.00         0.00
## 13021653          CRD        8.50       0.0     0.0       1.00         0.00
## 13298838          CRD       75.00       0.0     0.5       0.00         0.00
## 13368364          CRD        8.00       0.0     0.0       0.00         0.00
## 13369056          CRD       70.00       0.0     0.0      10.00         0.00
## 13383913          CRD       70.00       0.0     0.0      14.00         0.00
## 13437875          CRD       50.00       0.0     0.5       0.00         0.00
## 13451302          CRD        9.00       0.0     0.5       0.00         0.00
## 13456241          CRD       50.00       0.0     0.0       0.00        10.25
## 13698146          CRD       18.00       0.0     0.0       4.50         0.00
## 13862322          CRD        4.00       0.0     0.0       0.00         0.00
##          total_amount  fWeekday fDay fHour fWendOrWday
## 38134           60.00    Monday    9     0     Weekday
## 126139          12.00  Saturday    7    11     Weekend
## 673262          70.25  Saturday   14     1     Weekend
## 987582          65.00 Wednesday   11    15     Weekday
## 1068387         85.00  Saturday   14    18     Weekend
## 1096146        160.00 Wednesday   11    19     Weekday
## 1197346         75.00  Saturday   14    20     Weekend
## 1476721         63.60    Sunday   15     3     Weekend
## 1758222         90.00  Thursday    5    12     Weekday
## 1776226          8.40    Monday    2    17     Weekday
## 1821255         33.33    Monday    2    18     Weekday
## 1886968         15.00   Tuesday    3     3     Weekday
## 1959896         80.00  Thursday    5     0     Weekday
## 1976825        130.00  Thursday    5    16     Weekday
## 2037020        126.10   Tuesday    3     0     Weekday
## 2043804         16.20   Tuesday    3    20     Weekday
## 2111755         11.40 Wednesday    4     9     Weekday
## 2255996         90.00    Monday    2     8     Weekday
## 2323592         19.50    Sunday    8     2     Weekend
## 2361763         12.00    Sunday    8    10     Weekend
## 2412387         90.00   Tuesday    3    13     Weekday
## 2499887         40.00  Thursday    5     0     Weekday
```

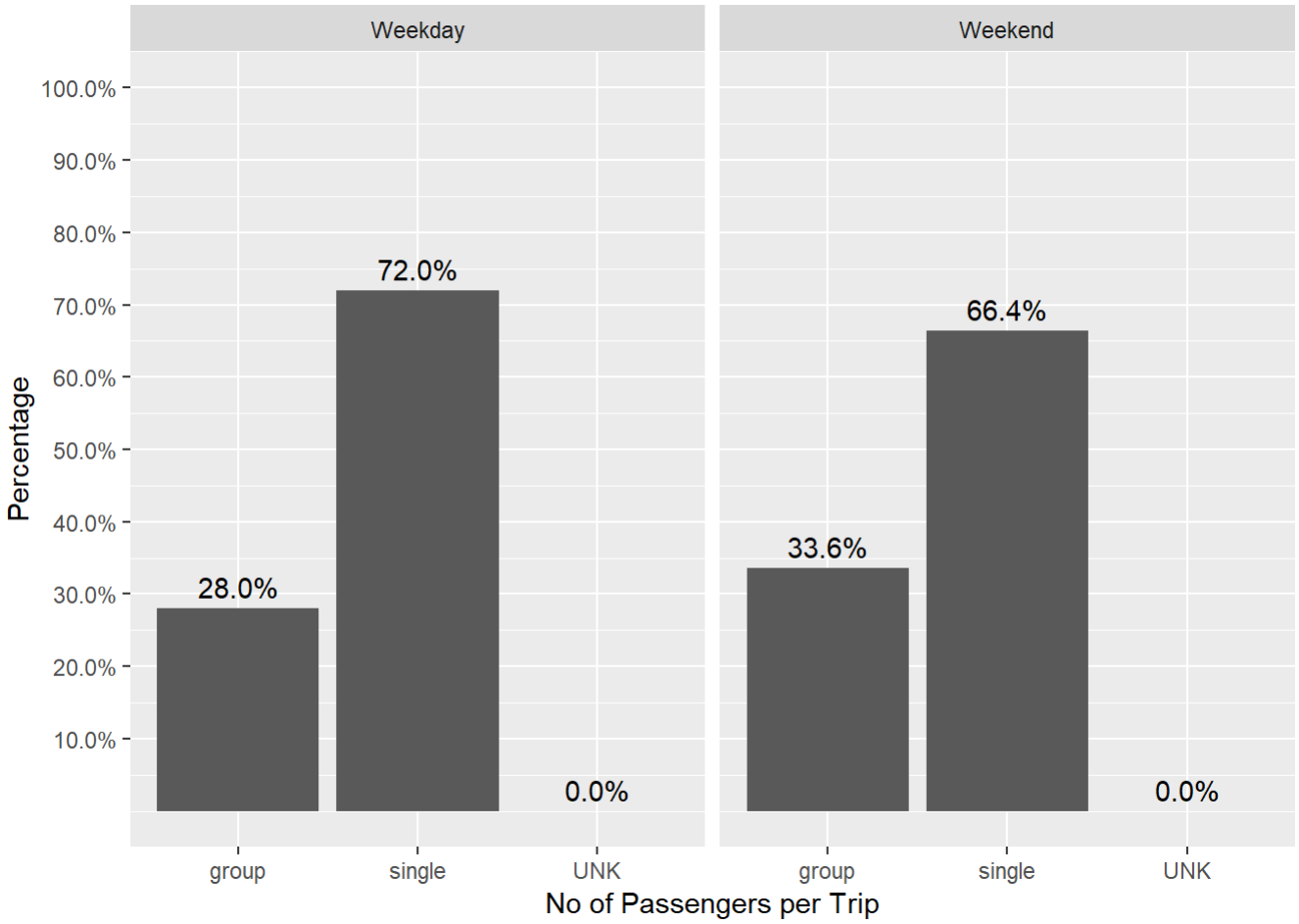
```
## 3318210        144.00    Friday   20    15     Weekday
## 3589536         71.40    Monday   16    12     Weekday
## 3624757         16.90  Thursday   19     1     Weekday
## 3631463         75.00  Saturday   21     0     Weekend
## 3866556         14.40   Tuesday   17    18     Weekday
## 3946581         99.25    Friday   20    15     Weekday
## 3968957         72.50   Tuesday   17    21     Weekday
## 4027179         13.00 Wednesday   18    18     Weekday
## 4117579         15.00    Monday   16    12     Weekday
## 4142923          6.00  Thursday   19     2     Weekday
## 4296178         65.00   Tuesday   17    18     Weekday
## 4385585         25.20 Wednesday   18    14     Weekday
## 5043904         70.80 Wednesday   25    18     Weekday
## 5107650        184.00 Wednesday   25    19     Weekday
## 5165134         50.00    Monday   23    14     Weekday
## 5173960          8.40 Wednesday   25    23     Weekday
## 5220276          6.00  Thursday   26     2     Weekday
## 5973528         14.50  Saturday   14    18     Weekend
## 6000128         19.50   Tuesday   10    18     Weekday
## 6032392         22.50 Wednesday   11    21     Weekday
## 6041668         12.00    Monday    9    19     Weekday
## 6053094          9.00    Monday    9     1     Weekday
## 6053095         16.20    Monday    9     1     Weekday
## 6053131         37.00    Monday    9     1     Weekday
## 6076281         39.38    Friday   13    21     Weekday
## 6110615         29.25    Friday   13    22     Weekday
## 6172247         16.20  Thursday   12    18     Weekday
## 6184196         62.83    Monday    9     6     Weekday
## 6202182         25.80 Wednesday   11     8     Weekday
## 6212986         10.00    Friday   13    11     Weekday
## 6316632        108.30    Sunday   15    15     Weekend
## 6326159         15.50    Friday   13     6     Weekday
## 6402677          6.87   Tuesday   10    21     Weekday
## 6431708         16.25 Wednesday   11     8     Weekday
## 6462860         16.80 Wednesday   11    21     Weekday
## 6490673         13.00   Tuesday   10     7     Weekday
## 6511647         96.00  Thursday   12    22     Weekday
## 6543783         62.38    Monday    9     5     Weekday
## 6545662         17.40    Friday   13    11     Weekday
## 6555412         85.75  Saturday   14     4     Weekend
## 6578716          9.00  Saturday   14    20     Weekend
## 6591273         48.00  Saturday   14     6     Weekend
## 6621967         16.25   Tuesday   10     8     Weekday
## 6671012         25.00    Friday   13     0     Weekday
## 6678727          9.00    Friday   13     6     Weekday
## 6705485         75.00  Saturday   14     0     Weekend
## 6755663         28.00 Wednesday   11    10     Weekday
## 6779704         48.48    Sunday   15     7     Weekend
## 6791350         90.00    Monday    9     7     Weekday
## 7309896         10.83  Saturday   14    11     Weekend
## 7393297          8.83    Friday   13    22     Weekday
## 7519105         30.00  Saturday   14     5     Weekend
## 7727428         87.90    Friday   20    10     Weekday
## 7732917         50.00  Saturday   21     5     Weekend
```

```
## 7771581         60.00 Wednesday   18    15     Weekday
## 7783128         58.00  Saturday   21     0     Weekend
## 7892259         10.80    Monday   16    14     Weekday
## 8004992         83.00    Sunday   22    17     Weekend
## 8035307          4.00    Friday   20    18     Weekday
## 8136232         38.33  Thursday   19    14     Weekday
## 8230991        115.00    Monday   16    17     Weekday
## 8260660         48.25   Tuesday   17    23     Weekday
## 8267089         18.50 Wednesday   18    17     Weekday
## 8301396          6.60  Saturday   21     9     Weekend
## 8317533         70.00  Thursday   19     3     Weekday
## 8333352         53.72  Thursday   19    10     Weekday
## 8333817         21.00 Wednesday   18     8     Weekday
## 8363338          9.75  Thursday   19    17     Weekday
## 8364680          6.50  Saturday   21    20     Weekend
## 8373109        180.00    Sunday   22     0     Weekend
## 8414310         14.00  Thursday   19    23     Weekday
## 8491188         22.20  Saturday   21    20     Weekend
## 8511023         60.00    Sunday   22    23     Weekend
## 8523100         13.80    Friday   20     8     Weekday
## 9071745         81.33  Saturday   21    13     Weekend
## 9155416          3.50  Saturday   21    19     Weekend
## 9372866         55.62  Thursday   26     0     Weekday
## 9420861          6.00   Tuesday   24    15     Weekday
## 9448168          9.50    Monday   23    11     Weekday
## 9454371         57.90  Saturday   28    21     Weekend
## 9471426         14.50  Thursday   26    11     Weekday
## 9520390         17.50  Thursday   26     0     Weekday
## 9543066         14.00    Friday   27    22     Weekday
## 9545551         12.00    Friday   27     0     Weekday
## 9589183         45.00  Saturday   28    10     Weekend
## 9593164         49.33    Monday   23    13     Weekday
## 9603160         58.33    Monday   23     7     Weekday
## 9655875         60.00  Thursday   26     0     Weekday
## 9677302         21.00 Wednesday   25    15     Weekday
## 9708469         13.20 Wednesday   25    16     Weekday
## 9726085         11.40    Friday   27     7     Weekday
## 9734388         41.50  Saturday   28     2     Weekend
## 9753385          3.50   Tuesday   24     5     Weekday
## 9817732         63.00  Saturday   28     6     Weekend
## 9864542         65.00    Sunday   29     7     Weekend
## 9897393         11.40  Thursday   26    19     Weekday
## 9921326         53.00    Friday   27     4     Weekday
## 9927058         52.00 Wednesday   25     9     Weekday
## 9935950        170.00    Sunday   29    15     Weekend
## 9970768         12.00  Thursday   26     5     Weekday
## 9983254         17.40  Saturday   28    15     Weekend
## 9995334         20.40 Wednesday   25    21     Weekday
## 10042845        25.00    Sunday   29     1     Weekend
## 10076214        13.20  Thursday   26     5     Weekday
## 10119407        65.83  Saturday   28     4     Weekend
## 10128477        10.80  Thursday   26     5     Weekday
## 10128702       135.60 Wednesday   25     7     Weekday
## 10146842         7.50    Friday   27    11     Weekday
```

```
## 10147505        11.40  Thursday   26     6     Weekday
## 10156003        15.60   Tuesday   24    20     Weekday
## 10157521        27.00  Thursday   26    17     Weekday
## 10225137         8.50   Tuesday   24    17     Weekday
## 10392990        58.25    Monday   23    14     Weekday
## 10577947         5.00    Friday   27    16     Weekday
## 10620862        77.25 Wednesday   25    14     Weekday
## 10927728        22.20    Monday   30     8     Weekday
## 10933709        19.80    Monday   30    18     Weekday
## 10937080         8.50    Monday   30     2     Weekday
## 10952862        42.94    Monday   30     2     Weekday
## 10961297         9.00    Monday   30    20     Weekday
## 10963022        69.39    Monday   30     2     Weekday
## 10968127        20.40    Monday   30     8     Weekday
## 10970205        27.50    Monday   30     2     Weekday
## 10980635        35.83    Monday   30     4     Weekday
## 11009831        13.20    Monday   30     6     Weekday
## 11013500        55.70    Monday   30    21     Weekday
## 11023069         8.00    Monday   30     2     Weekday
## 11029601        11.50    Monday   30     2     Weekday
## 11034800         6.00    Monday   30    16     Weekday
## 11044741        11.50    Monday   30     1     Weekday
## 11168642        70.20    Sunday    1     3     Weekend
## 11784692       111.25 Wednesday    4     7     Weekday
## 12006895         7.50    Monday    2     2     Weekday
## 12114093         8.40 Wednesday    4     2     Weekday
## 12837882       155.00    Friday    6    20     Weekday
## 13021653         9.50    Monday    2    16     Weekday
## 13298838        75.50    Friday   27    21     Weekday
## 13368364         8.00  Saturday   28     1     Weekend
## 13369056        80.00  Saturday   28     1     Weekend
## 13383913        84.00  Saturday   28     2     Weekend
## 13437875        50.50  Saturday   28     4     Weekend
## 13451302         9.50  Saturday   28     3     Weekend
## 13456241        60.25  Saturday   28     4     Weekend
## 13698146        22.50    Sunday   29     2     Weekend
## 13862322         4.00    Sunday   29    20     Weekend
```

Looking at the data for passenger_count = 0, it could be a data entry error or it could be erroneous records altogether. More detailed study is required to figure out which records are wrong. For example, some of the records with passenger count = 0 looks real data but some others have a lot of unusal values also like trip_time = 0 and trip_distance = 0 etc. I would need to look at trip_distance and trip_time variables more closely later on.

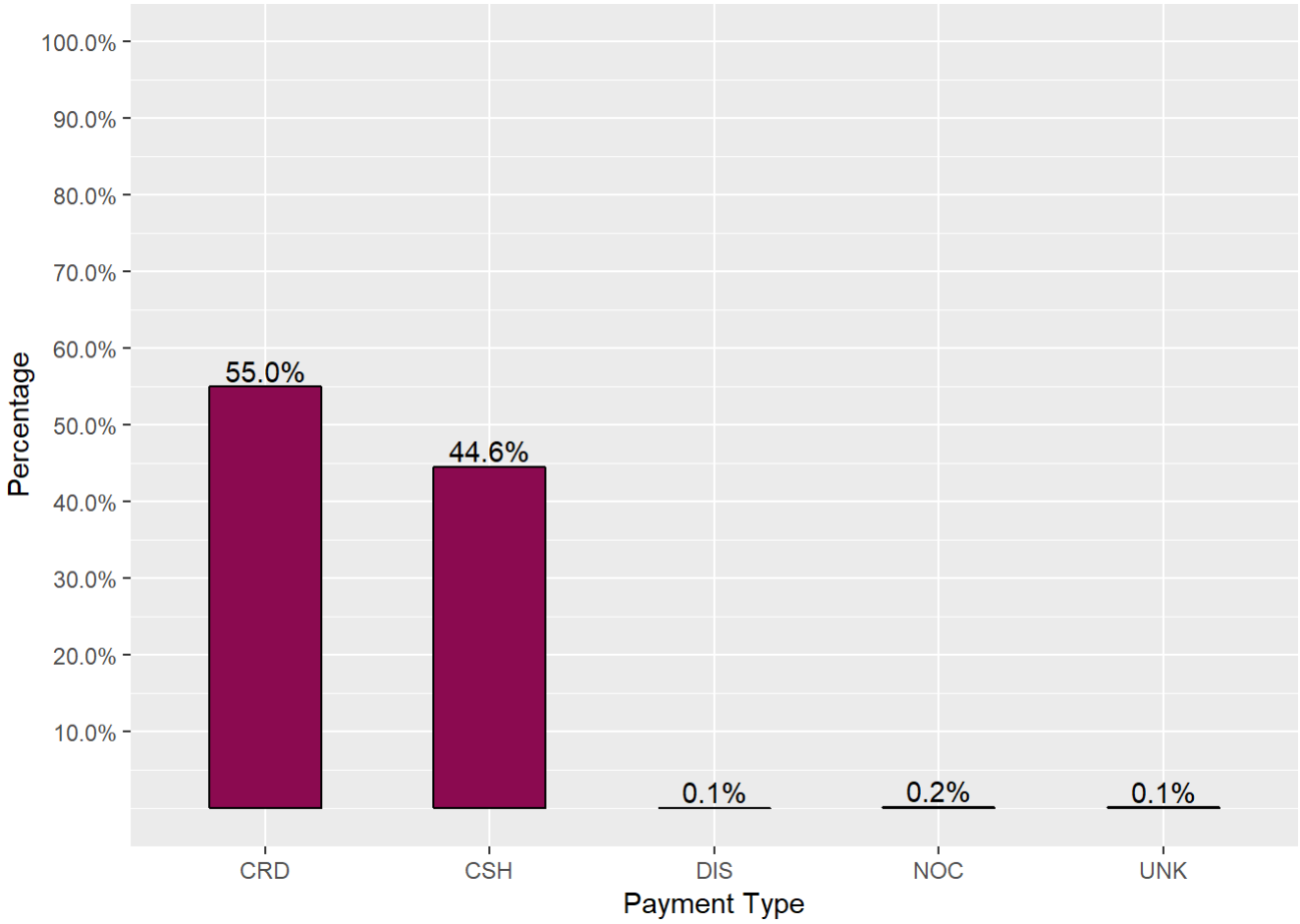
Next, I wanted to look at the distribution of passenger_counts based on Weekday/Weekend. I groups passenger_counts into three groups - single, group(2+), UNK (0 passenger_counts).



Group trips over the weekend goes up a bit but since the trip numbers are in millions, it is definitely significant. This shows that people like to go in groups over the weekend (with family, friends etc). This is a significant point for taxi companies trying to increase their revenue. They can schedule more fleet over the weekends and optimise the driver shift times according to the peak trip volumes seen in the previous section.

Distribution of payment type

Payment type shows the general payment preference of people.



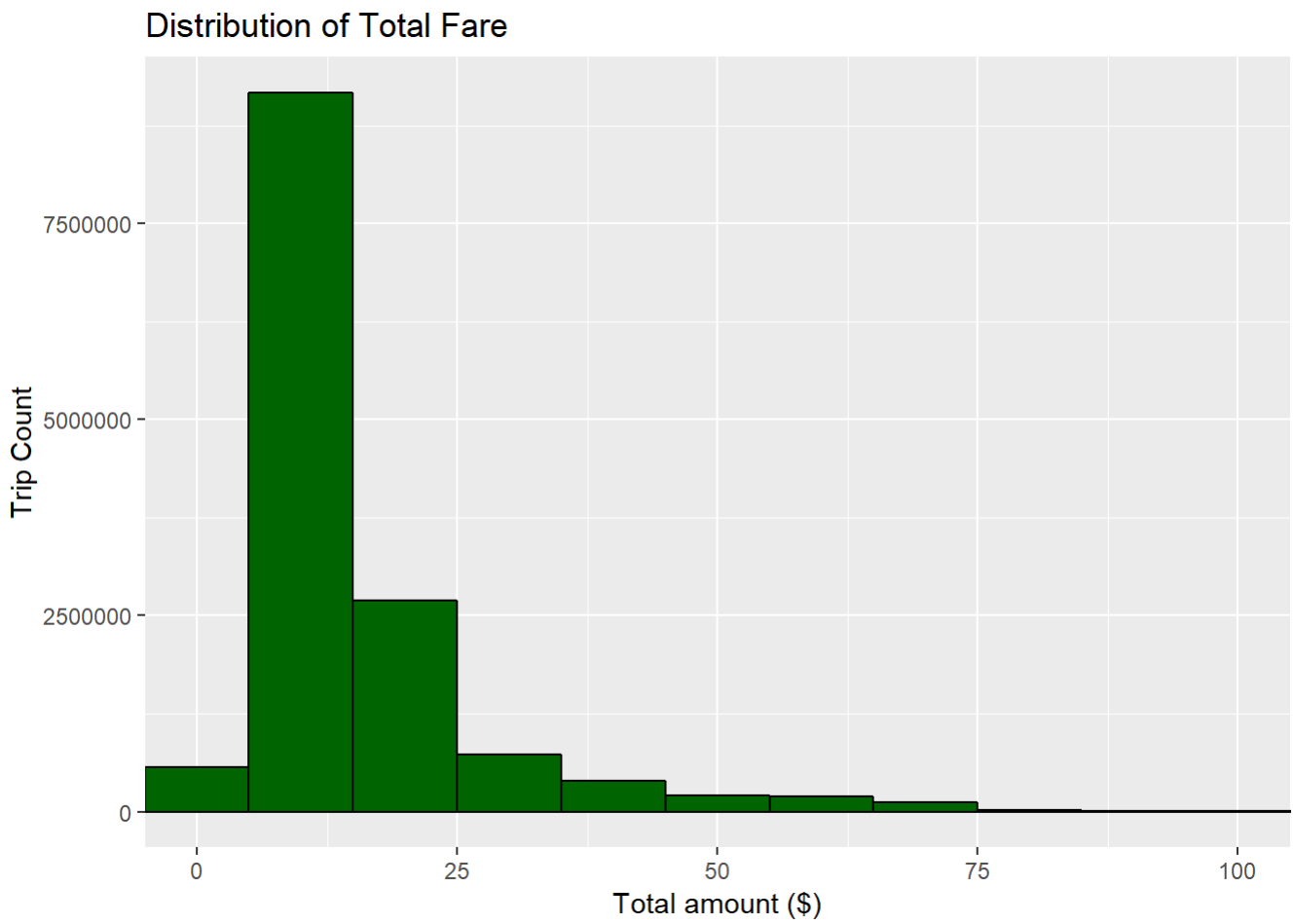
About 55% of people prefer card payments closely followed by card payments. I am expecting the actual card payment preference to be a little more higher than this as I believe people are forced to pay by cash in times of network connection issues. There are about 0.2% no charge payment type. Considering the actual trip numbers, I expected this number to be lower than it currently is. So many NOC trips are really suspicious. Assuming that the drivers need to pay some persentage amount to the taxi companies, it could be that drivers record the payment type as NOC for cash payments during network errors.

Distribution of Total Fare

```
# Sanity check for data errors
# taxi_data[taxi_data$total_amount<0,]  #0 records with negative amounts

cat("Mean amount per trip:", mean(taxi_data$total_amount))
```

Mean amount per trip: 15.23923

```
# 15.23923

#plot
ggplot(taxi_data, aes(x = total_amount)) + coord_cartesian(xlim = c(0, 100)) +
  geom_histogram(binwidth = 10, aes(fill = I("darkgreen"), colour = I("black"))) + labs(x = "Tot
al amount ($)", y = "Trip Count", title = "Distribution of Total Fare")
```

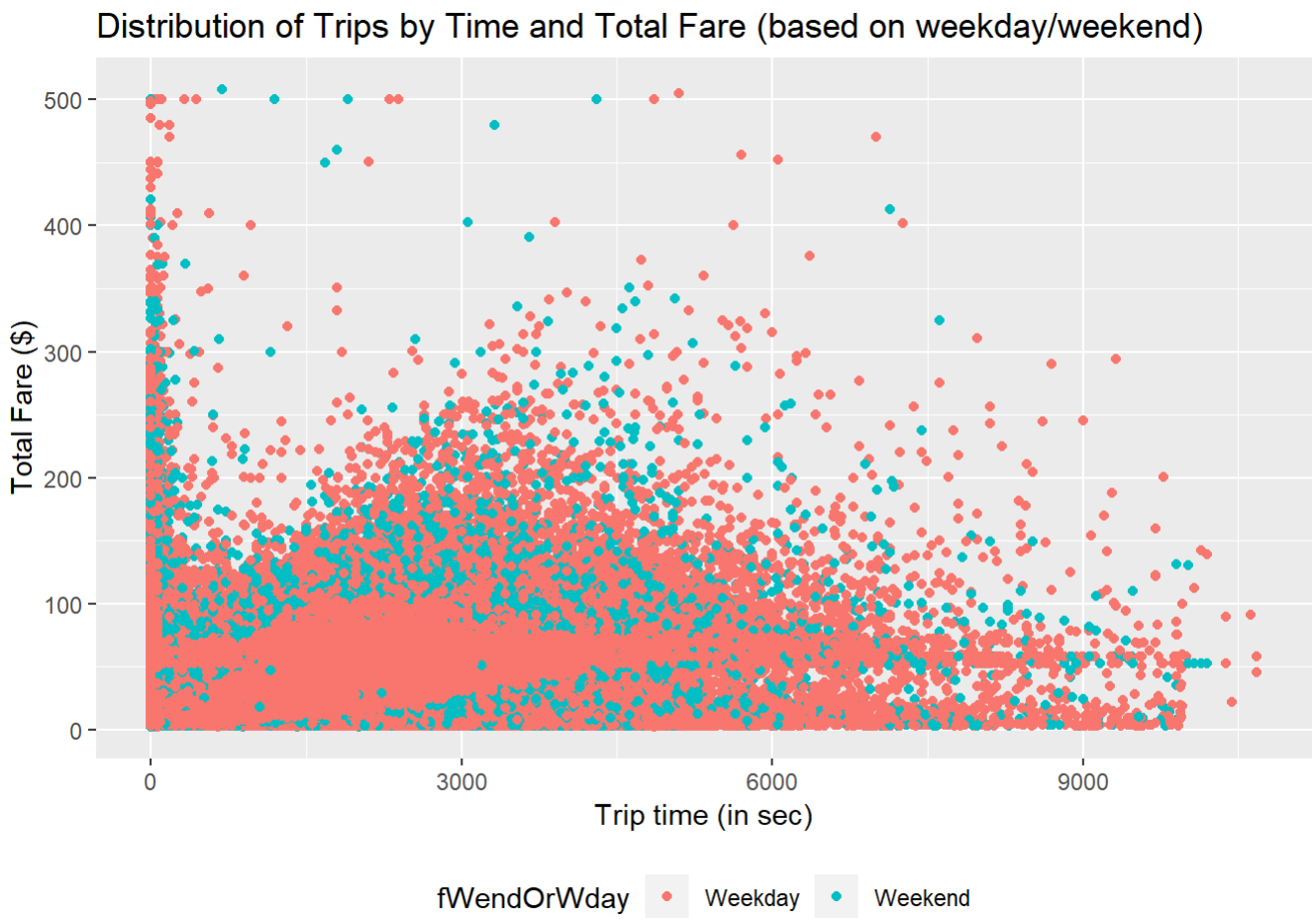


The mean amount earned per trip is around $15. The histogram above shows the distribution of the total fare. There are some high high earning trips which I guess are either airport (JFK) or Jersey City trips where there are flat rates + surcharges. It could also be just longer distance trips.

Total fares based on time

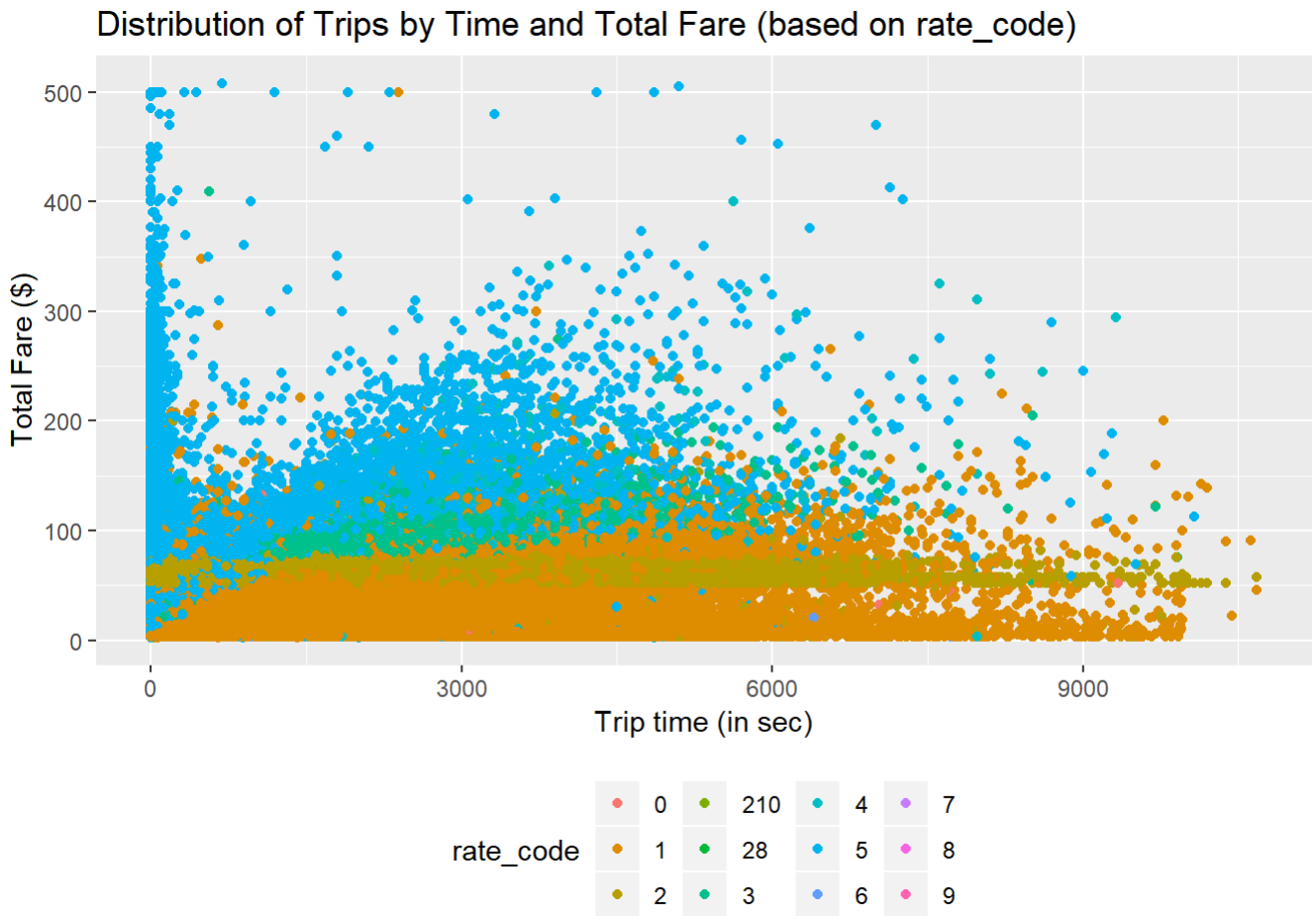
(EDA Q3: Distribution of Trips by Time and Fare)

```
ggplot(taxi_data, aes(x = trip_time_in_secs, y = total_amount, colour = fWendOrWday)) + geom_poi
nt() +theme(legend.position = "bottom") + labs(x = "Trip time (in sec)", y = "Total Fare ($)", t
itle = "Distribution of Trips by Time and Total Fare (based on weekday/weekend)")
```



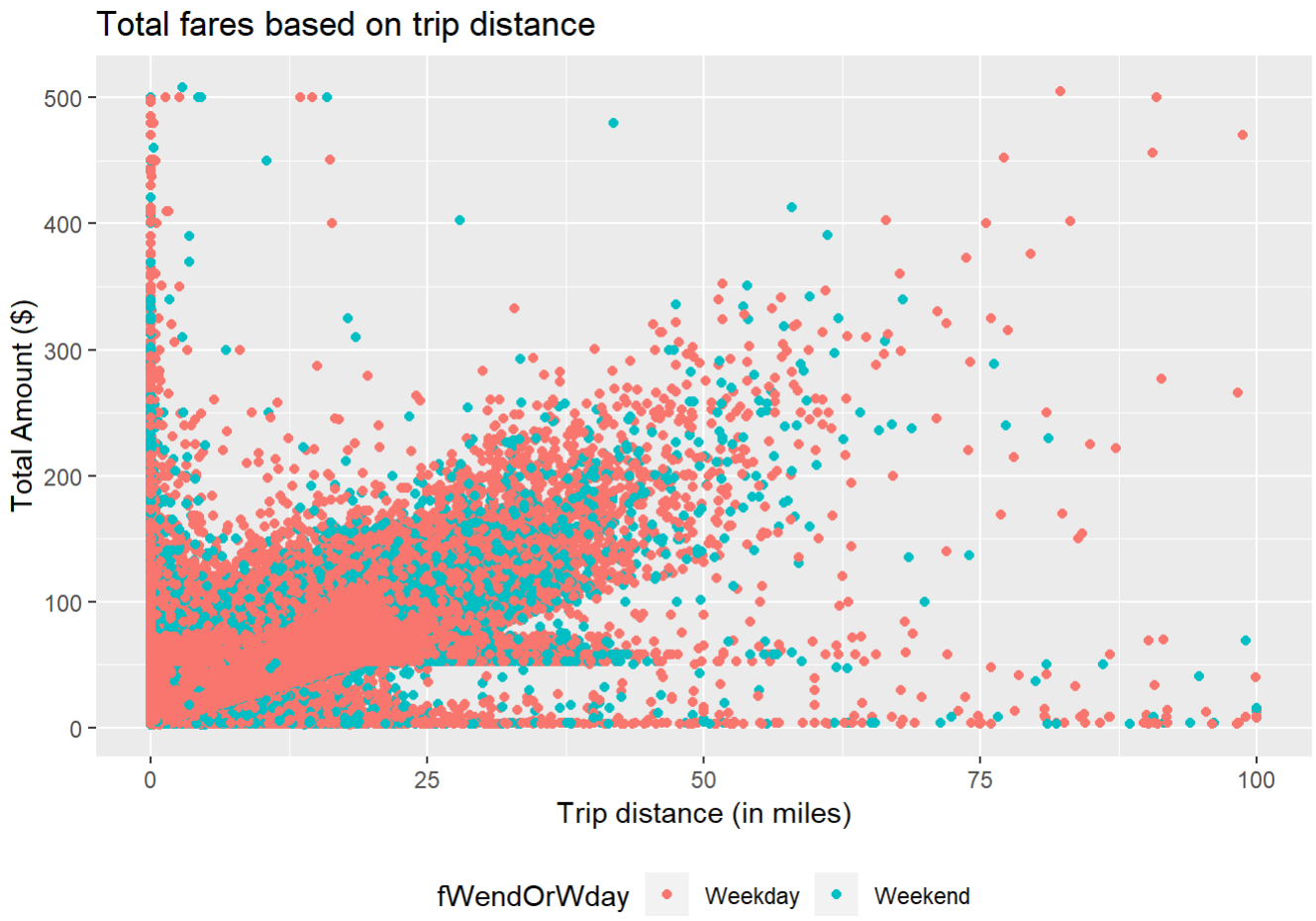
This is an interesting graph because it shows that the price does not increase with time. This means, the fares for NYC taxi trips are calculated based on distance and not time spent in transit. This can be especially disadvantageous in cities like NYC where traffic volume determines the trip time. There are also some outlier values like very high trip amount for insignificant trip time. There also seems to be a second line around the $50 mark. This suggests trips with certain rate_codes having flat charges. Also, people seems to be willing to pay more for similar time trips over the weekdays than over the weekends. But overall, there does not seem huge variations between weekdays and weekends. Maybe colouring with repect to rate_code might be useful.

```
ggplot(taxi_data, aes(x = trip_time_in_secs, y = total_amount, colour = rate_code)) + geom_point
() +theme(legend.position = "bottom") + labs(x = "Trip time (in sec)", y = "Total Fare ($)", tit
le = "Distribution of Trips by Time and Total Fare (based on rate_code)")
```



Total fares based on trip time

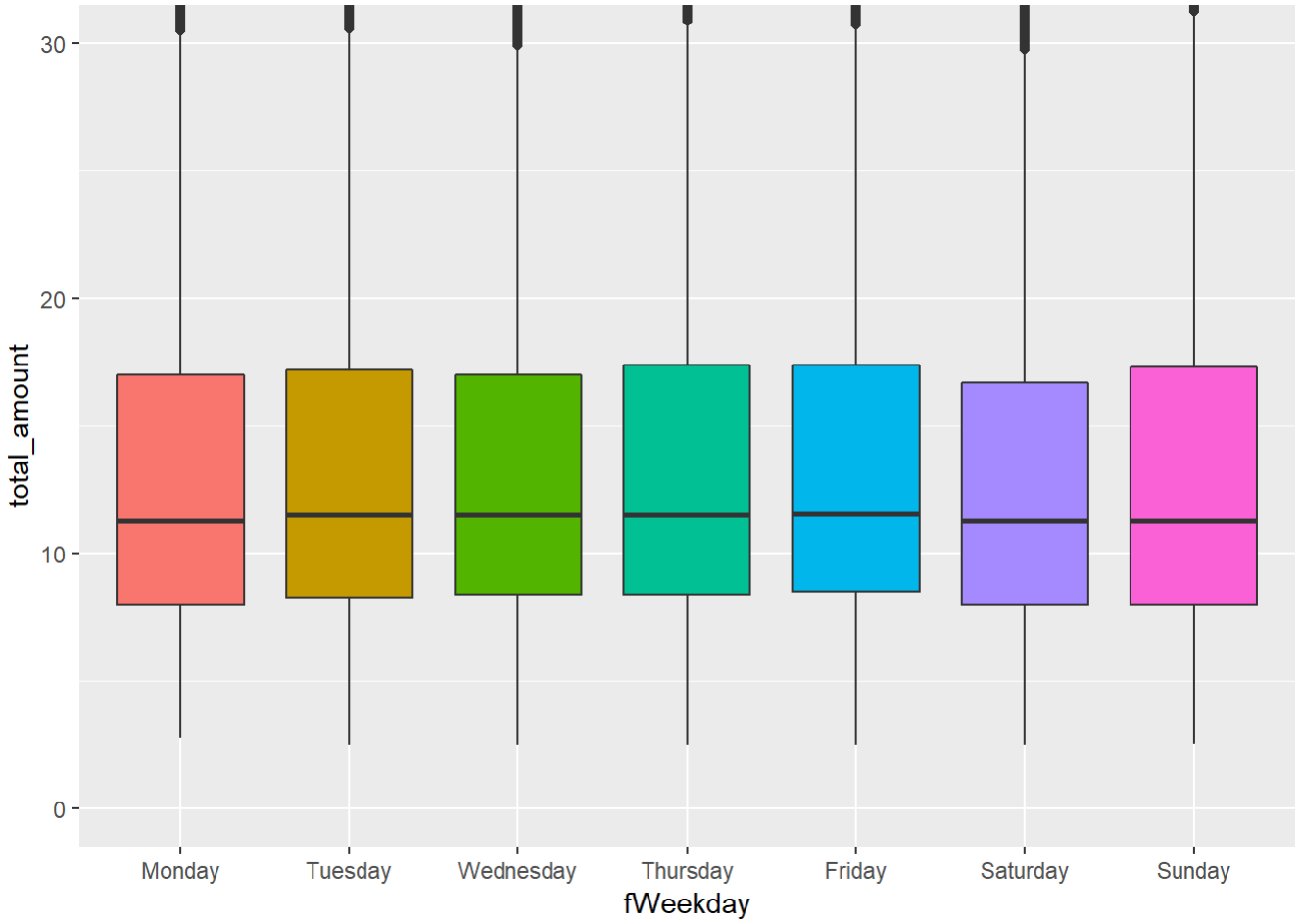
```
ggplot(taxi_data, aes(x = trip_distance, y = total_amount, colour = fWendOrWday)) + geom_point()
+theme(legend.position = "bottom") + labs(x = "Trip distance (in miles)", y = "Total Amount ($)"
, title = "Total fares based on trip distance")
```



This graph confirms the assumption in the previous graph that NYC taxi trips are charged based on distance only and not time. This also shows the trips with flat charges close to $50 (could be airport trips). This also seems to have a lot of possible outlier values which would need to be removed before modelling for fare (huge trip amounts for insignificant trip distance, insignificant trip amounts for high distance trips). This also shows the pattern that people are willing to pay more on the weekdays than over the weekends. It would be inetresting to see the tipping patterns over weekdays/weekends too.

Distribution of Total amount based on day of the week

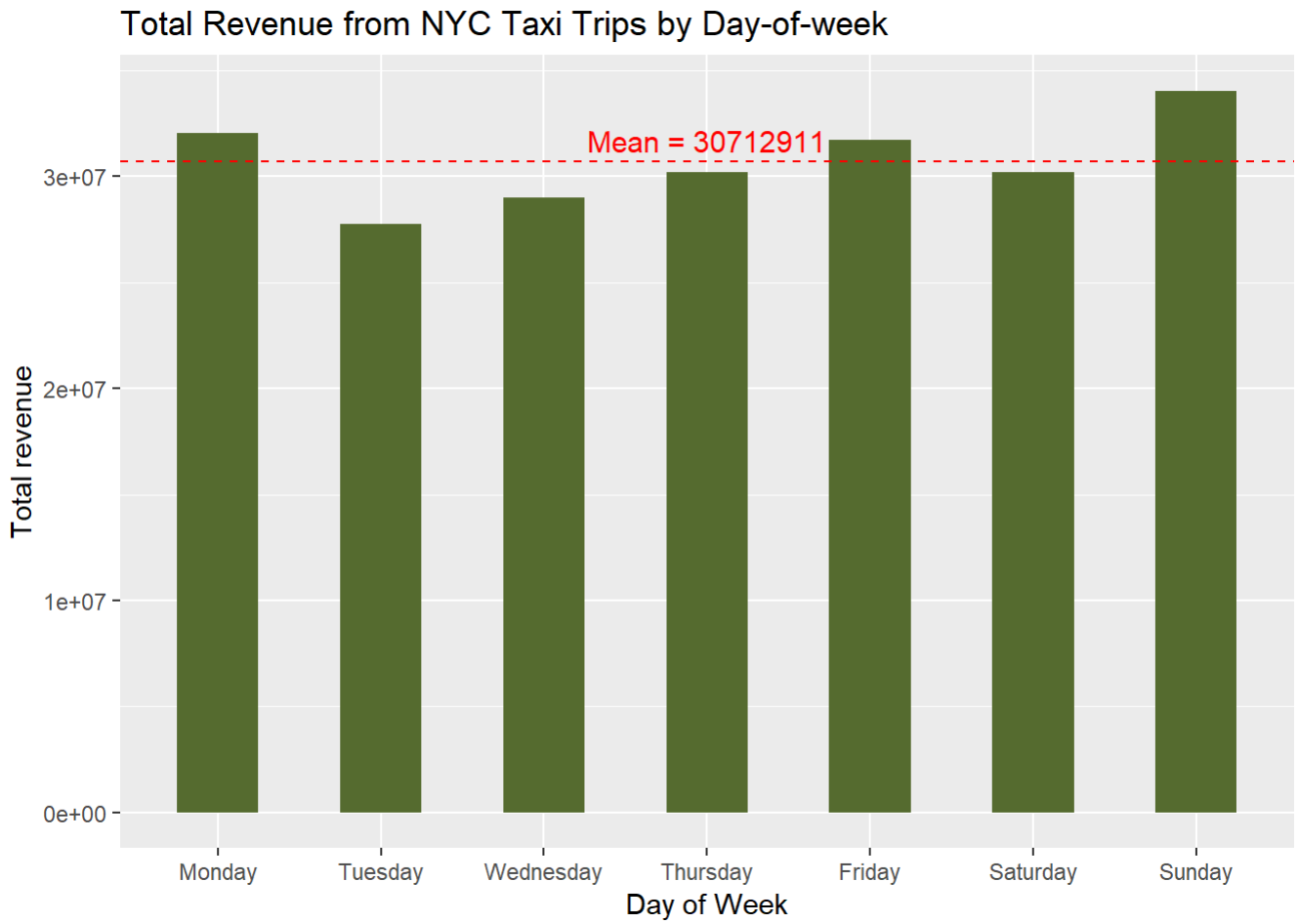
```
ggplot(taxi_data, aes(x = fWeekday, y = total_amount)) + geom_boxplot(aes(fill = fWeekday)) +the
me(legend.position = "none") + coord_cartesian(ylim = c(0, 30))
```



There does not seem to be a significant different between total amounts based on day of the week. This shows that although weekend trips are higher than weekday trips, people's willingness to spent differs. There are also a lot of 'high' amounts trips which extends way beyond the usual range. The y axis limits are adjusted to clearly visualise the boxplots.

Total Revenue by day of week

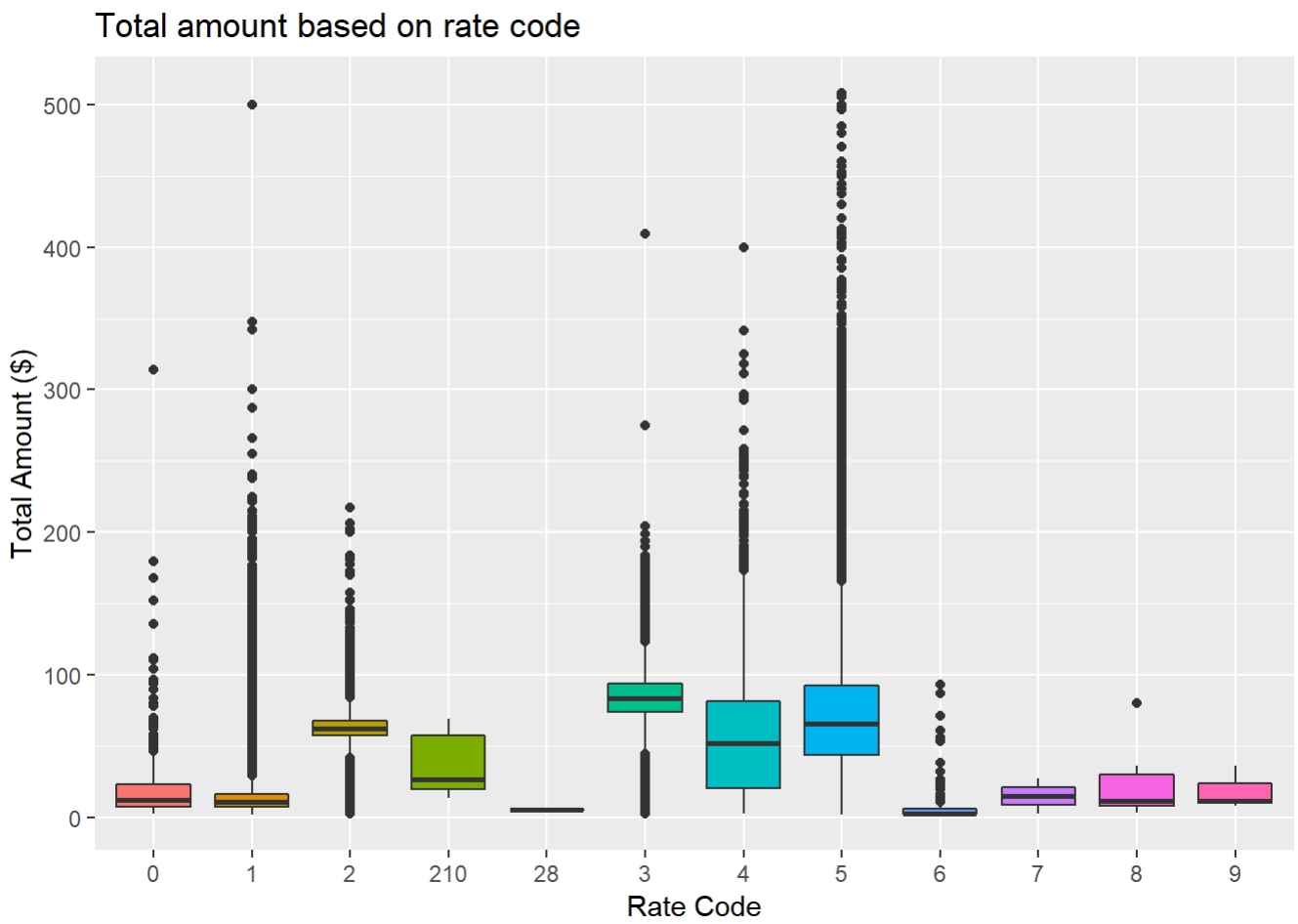
```
temp = taxi_data %>% group_by(fWeekday) %>% summarise("Total_amount" = sum(total_amount))

ggplot(temp, aes(x = fWeekday, y = Total_amount)) + geom_bar(stat = 'identity', width = 0.5, fil
l = "darkolivegreen") + labs(title="Total Revenue from NYC Taxi Trips by Day-of-week", x="Day of
 Week",y = "Total revenue")+ geom_hline(aes(yintercept = mean(Total_amount)), colour = 'red', li
netype = "dashed")+annotate(geom="text", x=4, y=31712911, label="Mean = 30712911", color="red",
 size = 4)
```



The total number of trips are higher than mean for Monday, Friday, Saturday and Sunday (previous section on trip volumes). The total revenue for Monday, Friday and Sunday are higher than the mean as expected but ironically the trip revenue is a bit lower than mean for Saturdays!

Total amount vs rate_code

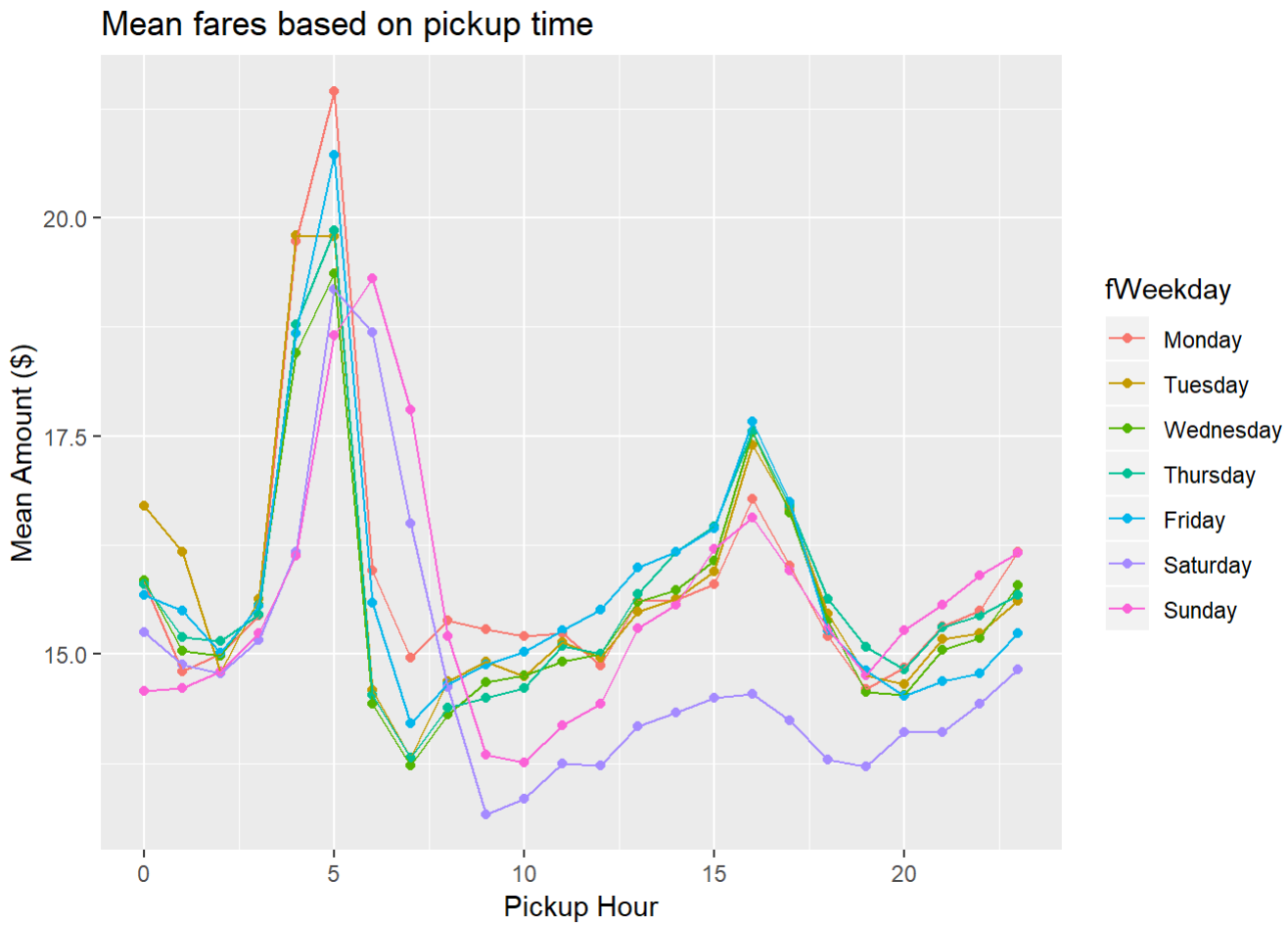
```
ggplot(taxi_data, aes(x = rate_code, y = total_amount)) + geom_boxplot(aes(fill = rate_code)) +t
heme(legend.position = "none")+ labs(x = "Rate Code", y = "Total Amount ($)", title = "Total amo
unt based on rate code")
```



Rate code = 1 is the standard rate based on distance and the revenue is one of the least. Also rate code = 6 which denotes group rides have lower revenues. This shows that group ride pricing does not favour taxi companies and drivers. Rate code = 2 denoting JFK airport has high revenue. It also has some lower than expected revenues. I am assuming that trips towards JFK are higher priced as it becomes the rider's necessisity. But once the driver is in JFK, he needs a return trip and then it becomes the driver's necessisity driving the prices down. Rate code = 5 denotes negotiated fare and this has very high variability.

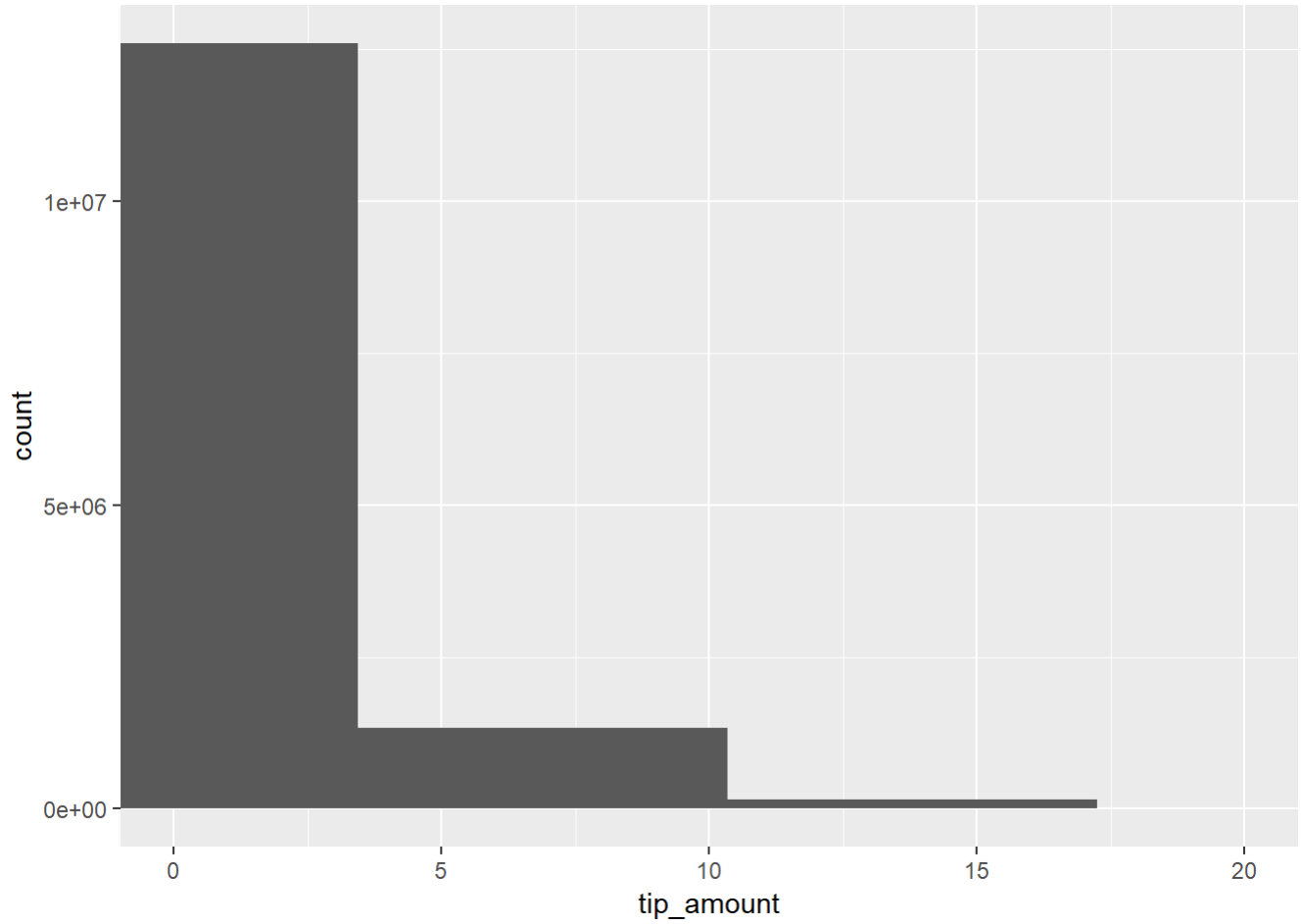
Total amount vs hour and day

```
temp =  taxi_data %>% group_by(fWeekday, fHour) %>% summarise("mean_earning" = mean(total_amoun
t))

ggplot(temp, aes(x = fHour, y = mean_earning, colour = fWeekday, group = fWeekday)) + geom_point
() + geom_line()+ labs(x = "Pickup Hour", y = "Mean Amount ($)", title = "Mean fares based on pi
ckup time")
```



There revenue at 5:00am and 4pm are unusually high and this corresponds to possible driver shift changes leading to higher demand than car availability. This also shows the unusually lower revenue for Saturday trips. It would be interesting to explore the data further for a reason.

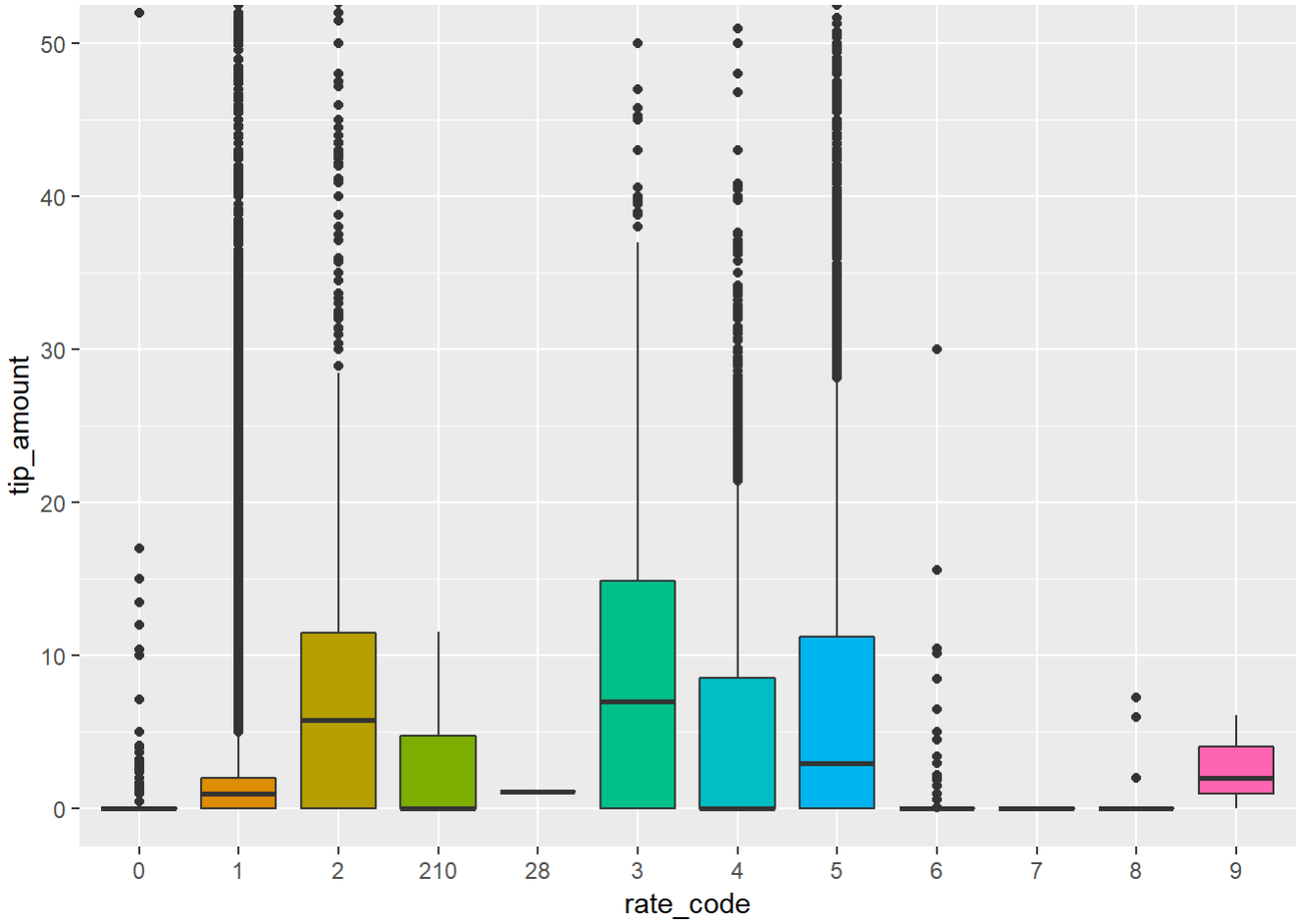
Distribution of Tip Amount



Majority of the tips are observed to be around $1-3.

Tip vs rate_code

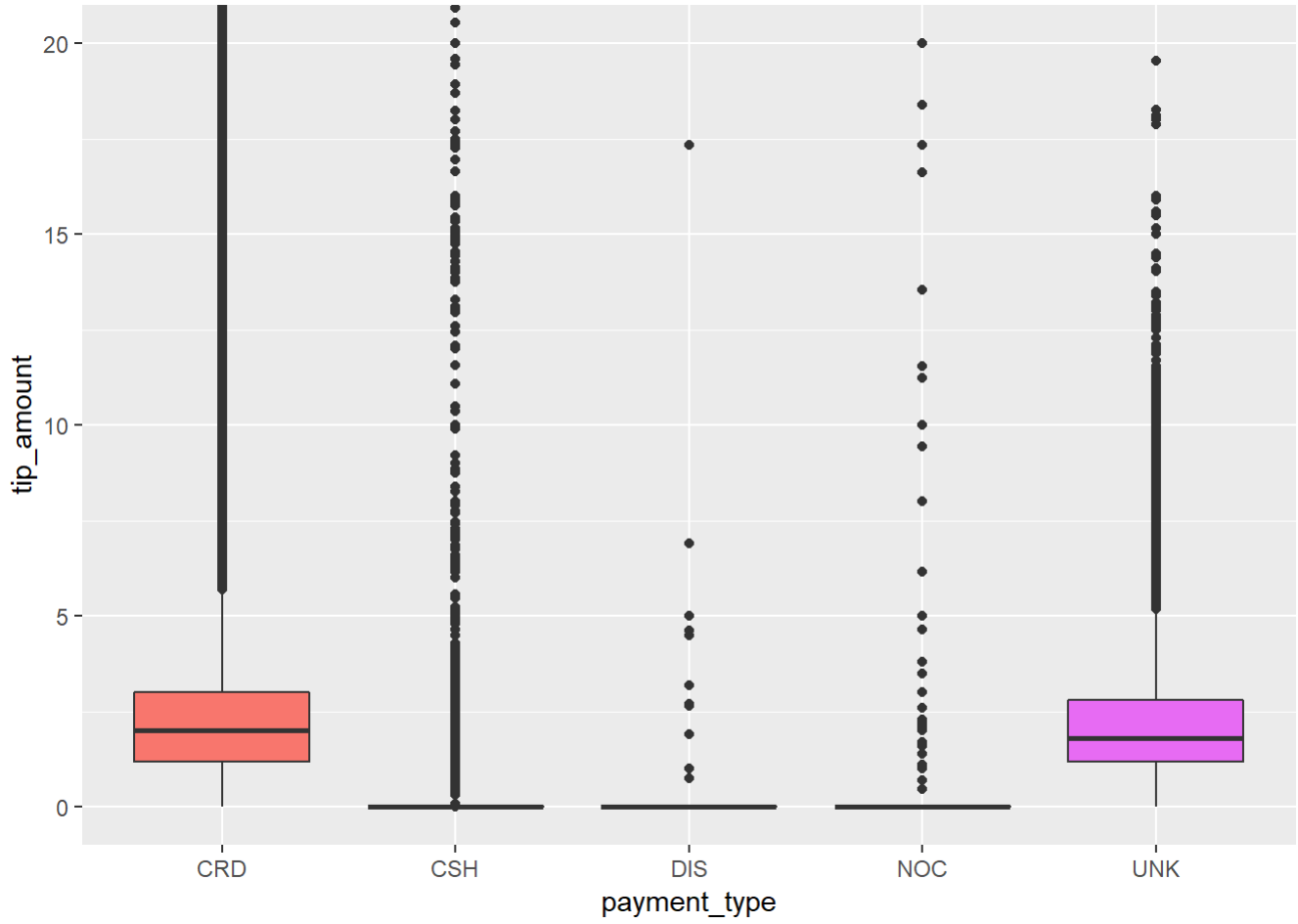
```
ggplot(taxi_data, aes(x = rate_code, y = tip_amount)) + geom_boxplot(aes(fill = rate_code)) +the
me(legend.position = "none")+coord_cartesian(ylim = c(0, 50))
```



Tips are higher for JFK (rate_code = 2), Newark (rate_code = 3)and Nassau and Westchester (rate_code = 4) possibly due to distance (and hence higher total amounts). JFK also has some unusally high tips! Interestingly, tip is also higher for negotiated rates (rate_code = 5). This implies, most of the negotiated charge trips are also longer distance trips. Tips for group rides (rate_code = 6) are the lowest.

Tip vs payment type

```
ggplot(taxi_data, aes(x = payment_type, y = tip_amount)) + geom_boxplot(aes(fill = payment_typ
e)) +theme(legend.position = "none")+coord_cartesian(ylim = c(0, 20))
```



This plot reinforces my assumption that tips are usually not recorded for cash payments by the drivers. As expected, disputed and no charge rides (possibly unrecorded cash payments) have close to zero tips. Most of UNK trips also display the same pattern as card payment trip tips.

(EDA Q4: Characterisation of drivers by working hours and earnings)

The dataset includes some sensitive information like medallion number (identifies taxis) and hack_license number (identifies driver). Eventhough these might not be actual numbers, they still represent unique identities. Hence, it is possible to track taxis and drivers in combination with the geo-spatial longitude and latitude information given. Also, since trip fare details are given, we can even figure out approximately how much each driver/taxi company earns per day/month.

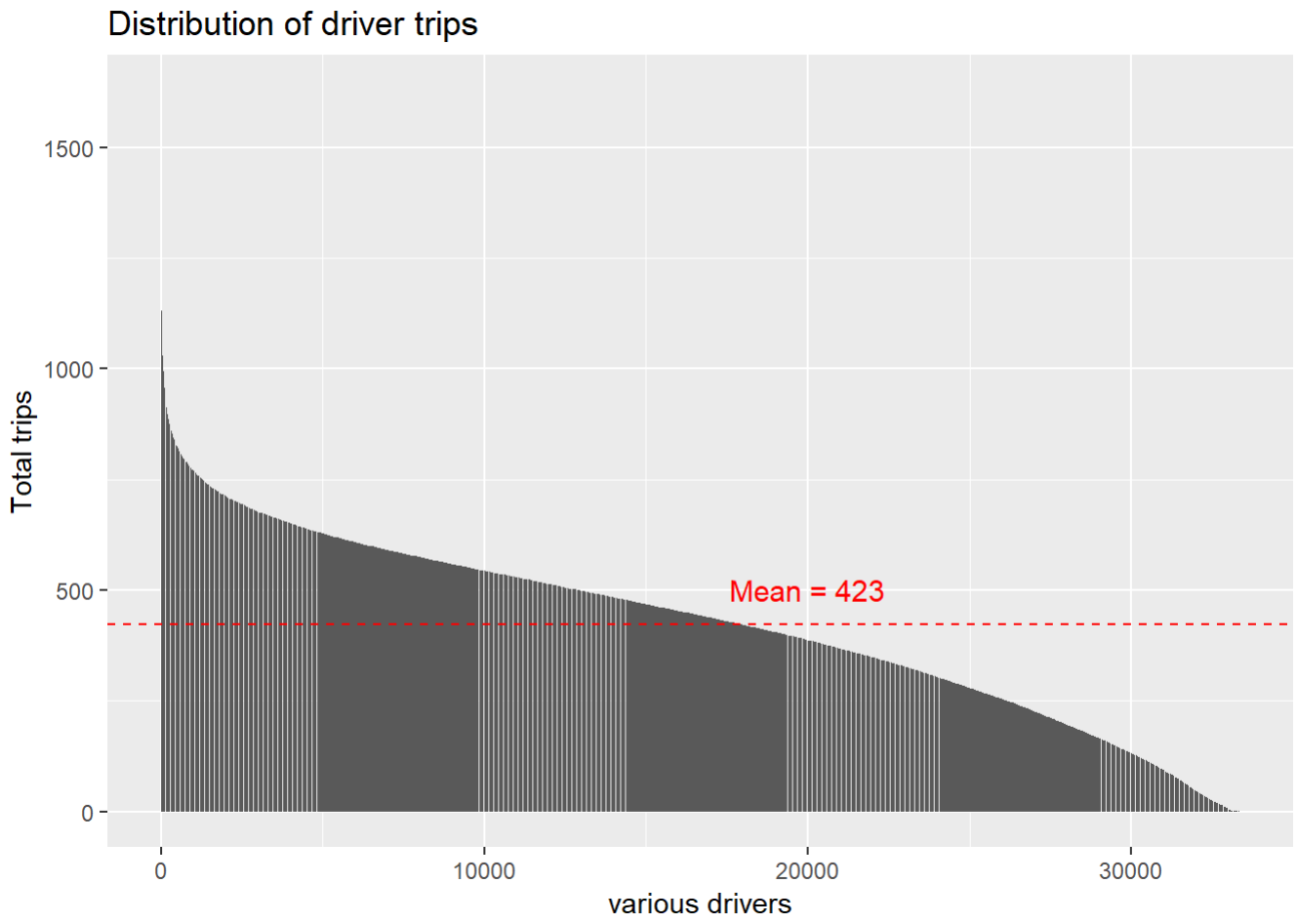
```
cat("Total unique taxi driver in Spetember 2013: ",length(unique(taxi_data$hack_license))) #3333
8
```

```
## Total unique taxi driver in Spetember 2013:  33338
```

```
cat("Total unique taxis in Spetember 2013: ",length(unique(taxi_data$medallion))) #13437
```

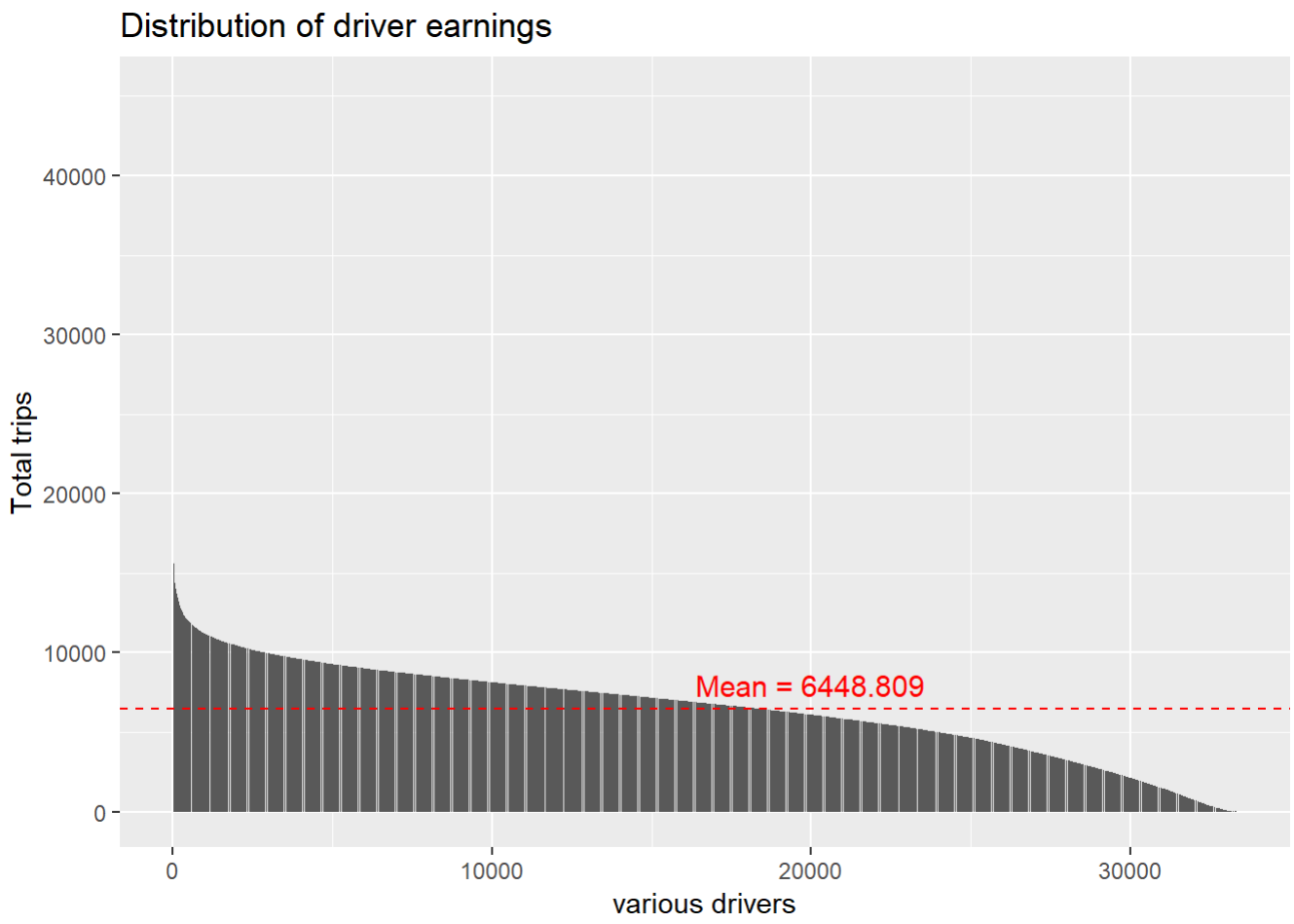
```
## Total unique taxis in Spetember 2013:  13437
```

```
#Summary of hack_licenses (driver)
#summary(taxi_data$hack_license)
temp = taxi_data %>% group_by(hack_license) %>% summarise("total_trips" = n()) %>% arrange(desc
(total_trips))

ggplot(temp, aes(x = 1:nrow(temp), y = total_trips)) + geom_bar(stat = 'identity') + labs(title=
"Distribution of driver trips", x="various drivers",y = "Total trips")+ geom_hline(aes(yintercep
t = mean(total_trips)), colour = 'red', linetype = "dashed")+annotate(geom="text", x=20000, y=50
0, label="Mean = 423", color="red", size = 4)
```



The mean trips in a month for the taxi drivers is around 432 trips. Most of the drivers seems to work full-time with total trips way more than the mean trips. I think the graph suggests that there is more demand for taxis than taxis on the ground.

```
temp = taxi_data %>% group_by(hack_license) %>% summarise("total_revenue" = sum(total_amount)) %
>% arrange(desc(total_revenue))

ggplot(temp, aes(x = 1:nrow(temp), y = total_revenue)) + geom_bar(stat = 'identity') + labs(titl
e="Distribution of driver earnings", x="various drivers",y = "Total trips")+ geom_hline(aes(yint
ercept = mean(total_revenue)), colour = 'red', linetype = "dashed")+annotate(geom="text", x=2000
0, y=8000, label="Mean = 6448.809", color="red", size = 4)
```



This plot follows the same pattern as total trips per driver. This is also logical that the driver earn based on their number of trips taken.

```
# Choosing one random driver for further analysis
# 6D3F364C51D18E1D4A077F53B9096166
driver_data = taxi_data[taxi_data$hack_license == "6D3F364C51D18E1D4A077F53B9096166",]

cat("Driver Characterisation")
```

```
## Driver Characterisation
```

```
cat("\n========================")
```

```
##
## ========================
```

```
cat("\nDriver License Number: 6D3F364C51D18E1D4A077F53B9096166")
```

```
##
## Driver License Number: 6D3F364C51D18E1D4A077F53B9096166
```

```
cat("\n----------------------------------")
```

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

```
cat("\nTotal trips in September 2013: ", nrow(driver_data))
```

```
##
## Total trips in September 2013:  1010
```

```
cat("\n----------------------------------")
```

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

```
cat("\nTotal taxis used for trips: ",length(unique(driver_data$medallion)))
```

```
##
## Total taxis used for trips:  9
```

```
cat("\n----------------------------------")
```

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

```
cat("\nTotal earnings for September 2013: ", sum(driver_data$total_amount))
```

```
##
## Total earnings for September 2013:  13329.17
```

```
cat("\n----------------------------------")
```

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

```
cat("\nTotal amount earned in tips for September 2013: ", sum(driver_data$tip_amount))
```

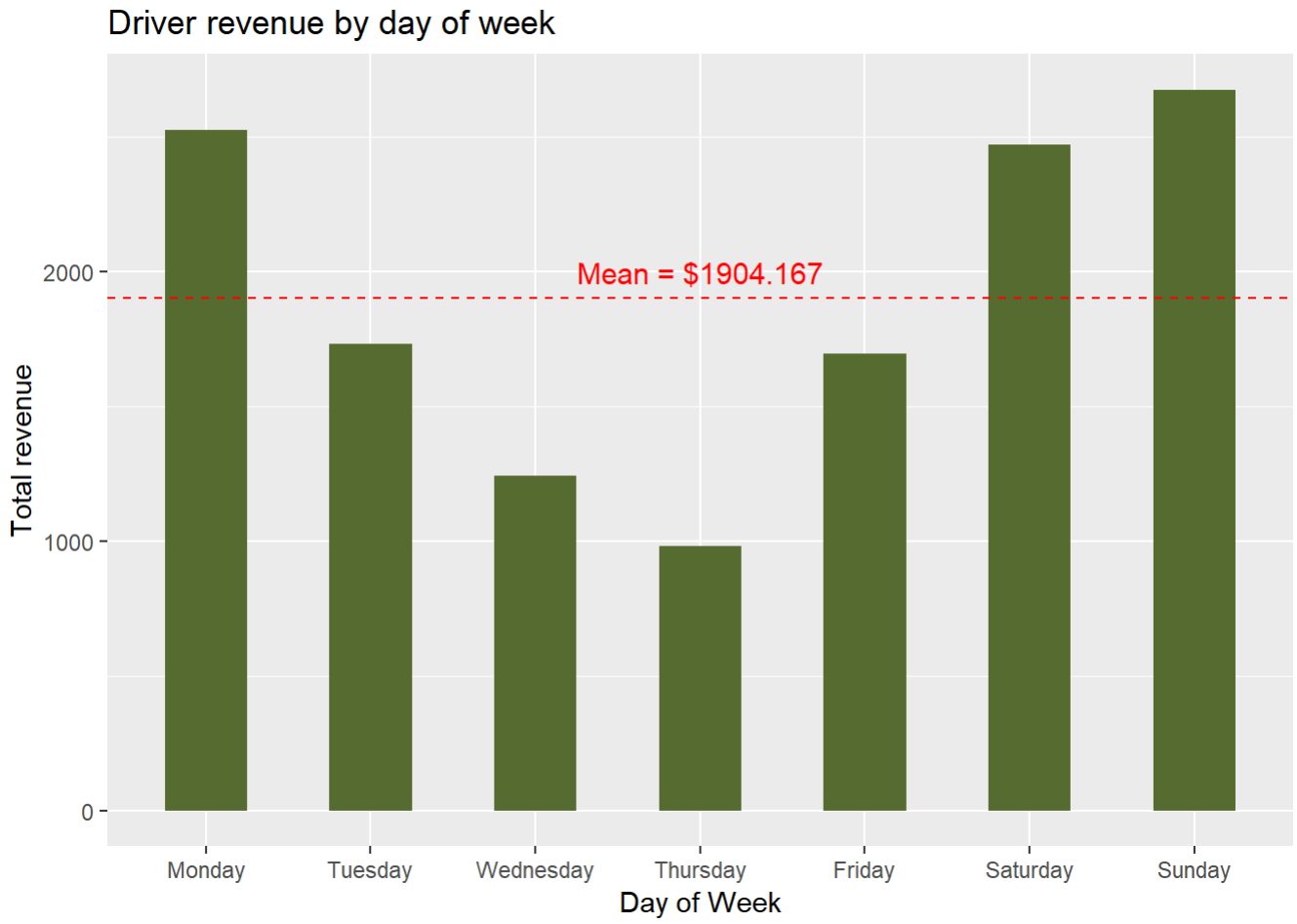
```
##
## Total amount earned in tips for September 2013:  1120.56
```

```
cat("\n----------------------------------")
```

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

Which days does this driver work more

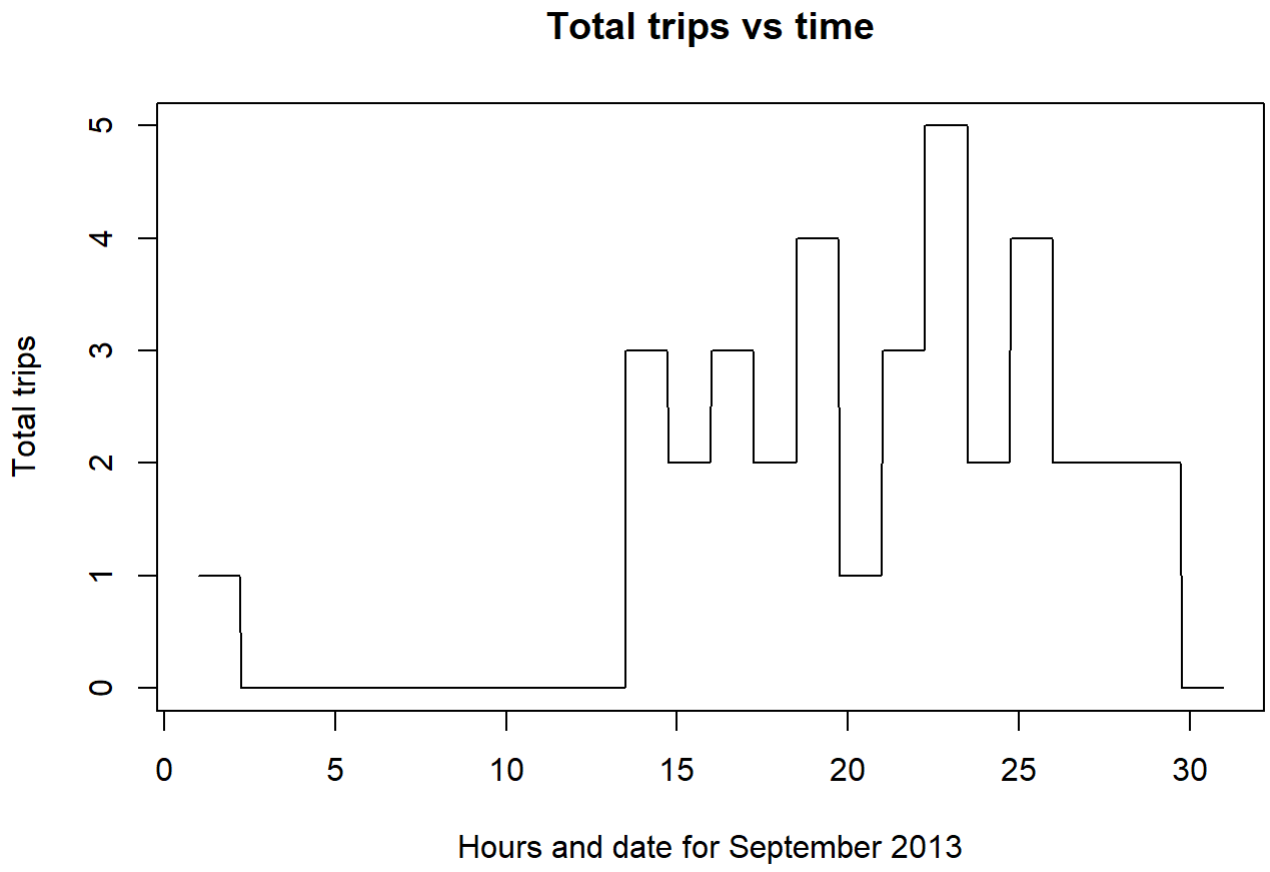
```
temp = driver_data %>% group_by(fWeekday) %>% summarise("Total_amount" = sum(total_amount))

ggplot(temp, aes(x = fWeekday, y = Total_amount)) + geom_bar(stat = 'identity', width = 0.5, fil
l = "darkolivegreen") + labs(title="Driver revenue by day of week", x="Day of Week",y = "Total r
evenue")+ geom_hline(aes(yintercept = mean(Total_amount)), colour = 'red', linetype = "dashed")+
annotate(geom="text", x=4, y=2000, label="Mean = $1904.167", color="red", size = 4)
```



The figure above shows that this driver is a well-seasoned driver who knows the NYC demand well. This driver seems to work everyday but more over the weekends and less towards midweek. At the same time, it is sadenning to know that he does not take any day off during the week (possibly the story of many other NYC drivers).

Driving pattern for September 2013

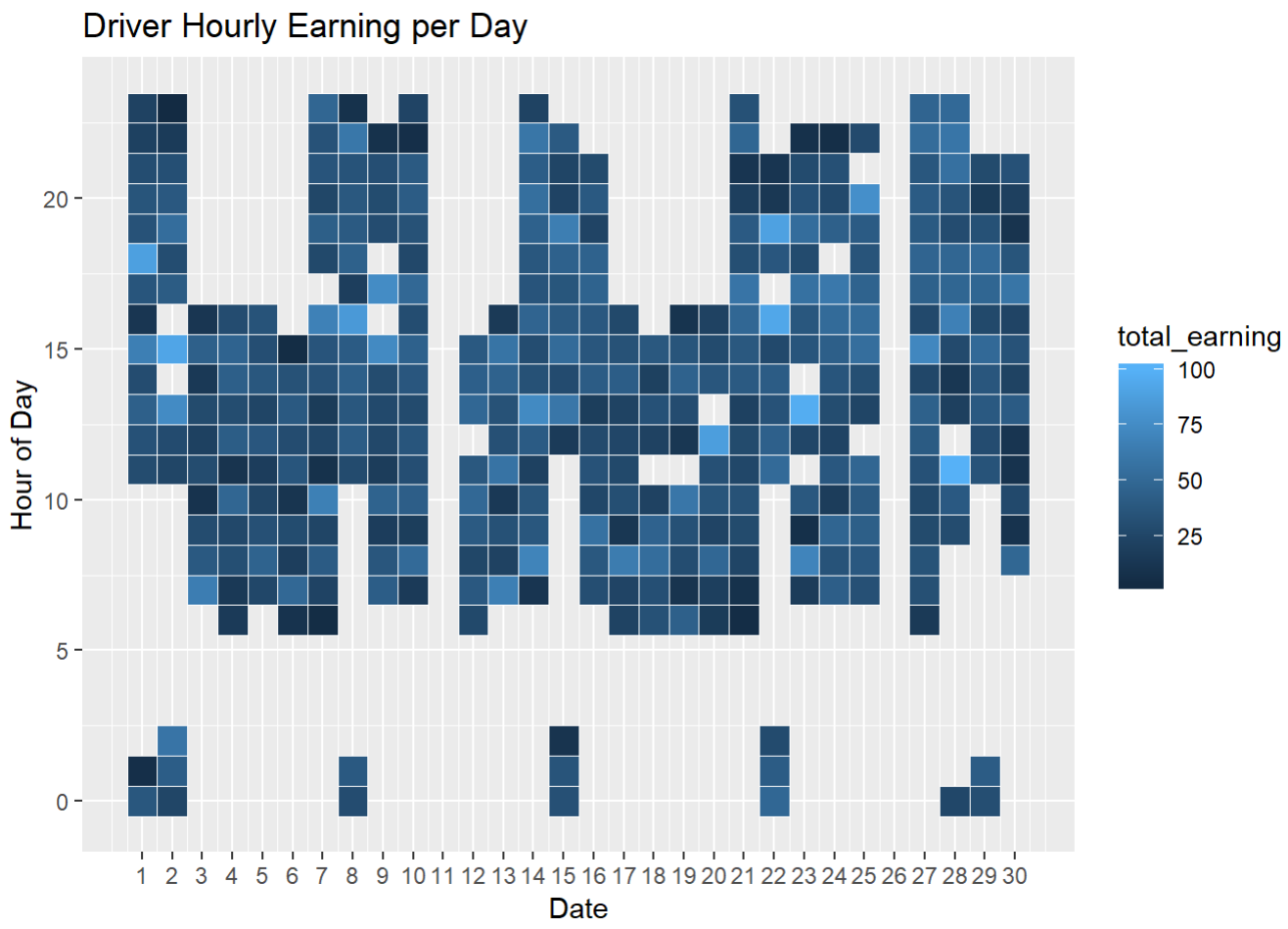
```
temp = driver_data %>% group_by(fDay, fHour) %>% summarise("total_trips" = n())

# making a time series suitable data frame
dateSeq = data.frame("fDay"=rep(1,24), "fHour" = rep(seq(1,24,1),30))
temp_merged = merge(dateSeq, temp, by = c("fDay", "fHour"), all.x=T)
# replacing all non-working hour trips as 0
temp_merged[which(is.na(temp_merged$total_trips)), "total_trips"] = 0
# time series data format
temp_ts = ts(temp_merged$total_trips, frequency = 24)
#plot
plot(temp_ts,xlab="Hours and date for September 2013",ylab="Total trips", main="Total trips vs t
ime")
```



Analysing the above graph, this driver works more on alternate days. This driver alternates between 2trips/hour and 3trips/hour on alternate days. He also took a break from driving for around 10 days.

Total earnings based on hour of day and day of month

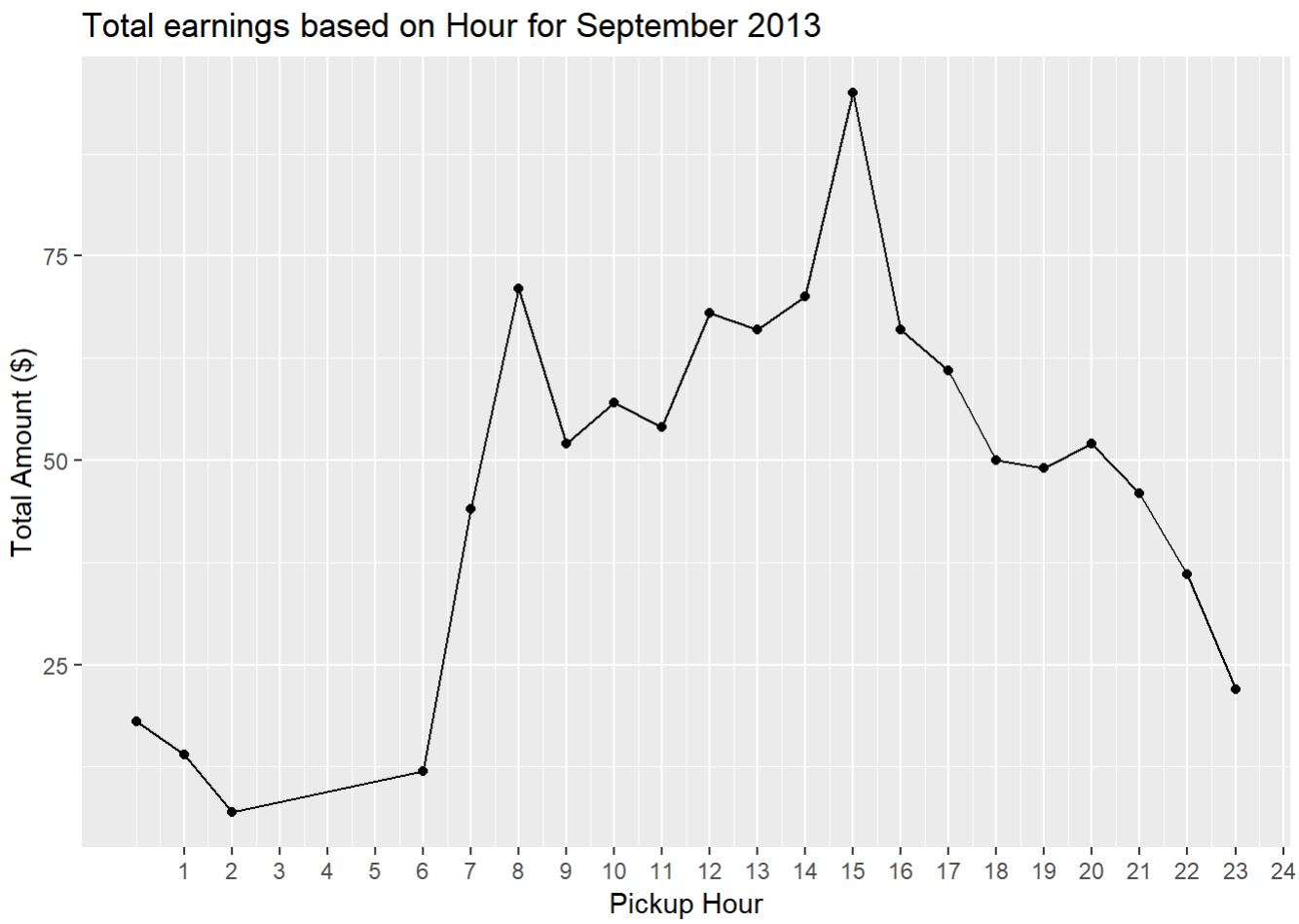
```
temp =  driver_data %>% group_by(fDay, fHour, medallion) %>% summarise("total_earning" = sum(tot
al_amount))

ggplot(temp, aes(fDay, fHour, fill = total_earning)) +
  geom_tile(colour = "white") +
  labs(x="Date",y="Hour of Day",title = "Driver Hourly Earning per Day") +scale_x_continuous(bre
aks = c(1:30))
```



The above plot gives a clear picture of the earning pattern of the driver per hour per day. This also shows that he moslty takes the morning shifts. It also shows that atleast 1 day a week, he workds extra during the night. This driver usually takes (or gets) shifts of approximately 10 hours. On the day he works extra, he takes around 7 hours break for rest. Hence, this is a responsible driver and takes adequate rests. It might be that the driving pattern of this driver is heavily influenced by the taxi he gets to rent for the day.

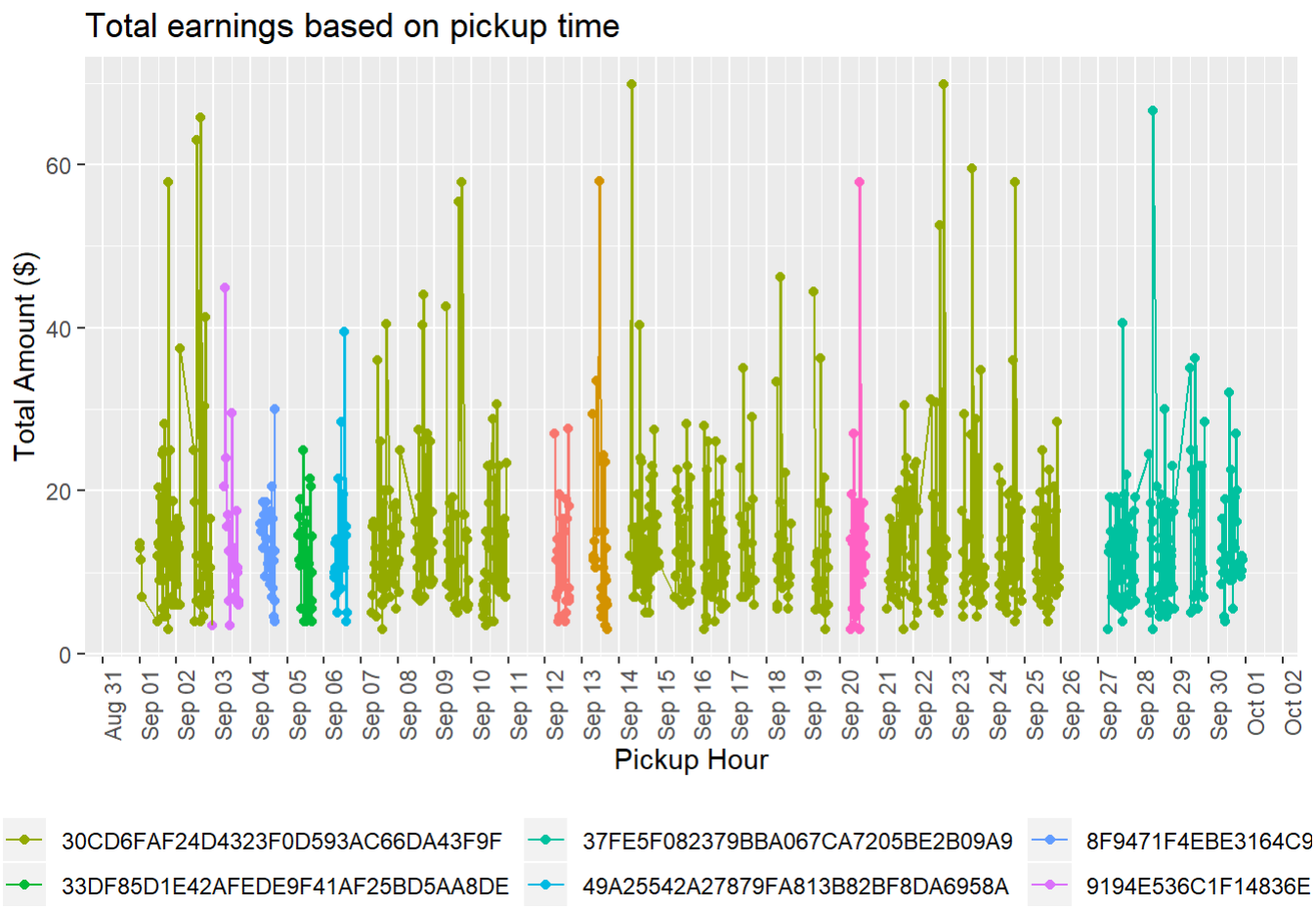
Does the driver work days or nights more

```
temp =  driver_data %>% group_by(fHour) %>% summarise("total_trips" = n())

ggplot(temp, aes(x = fHour, y = total_trips)) + geom_point() + geom_line()+ labs(x = "Pickup Hou
r", y = "Total Amount ($)", title = "Total earnings based on Hour for September 2013")+ theme(le
gend.position="bottom") +scale_x_continuous(breaks = c(1:24))
```

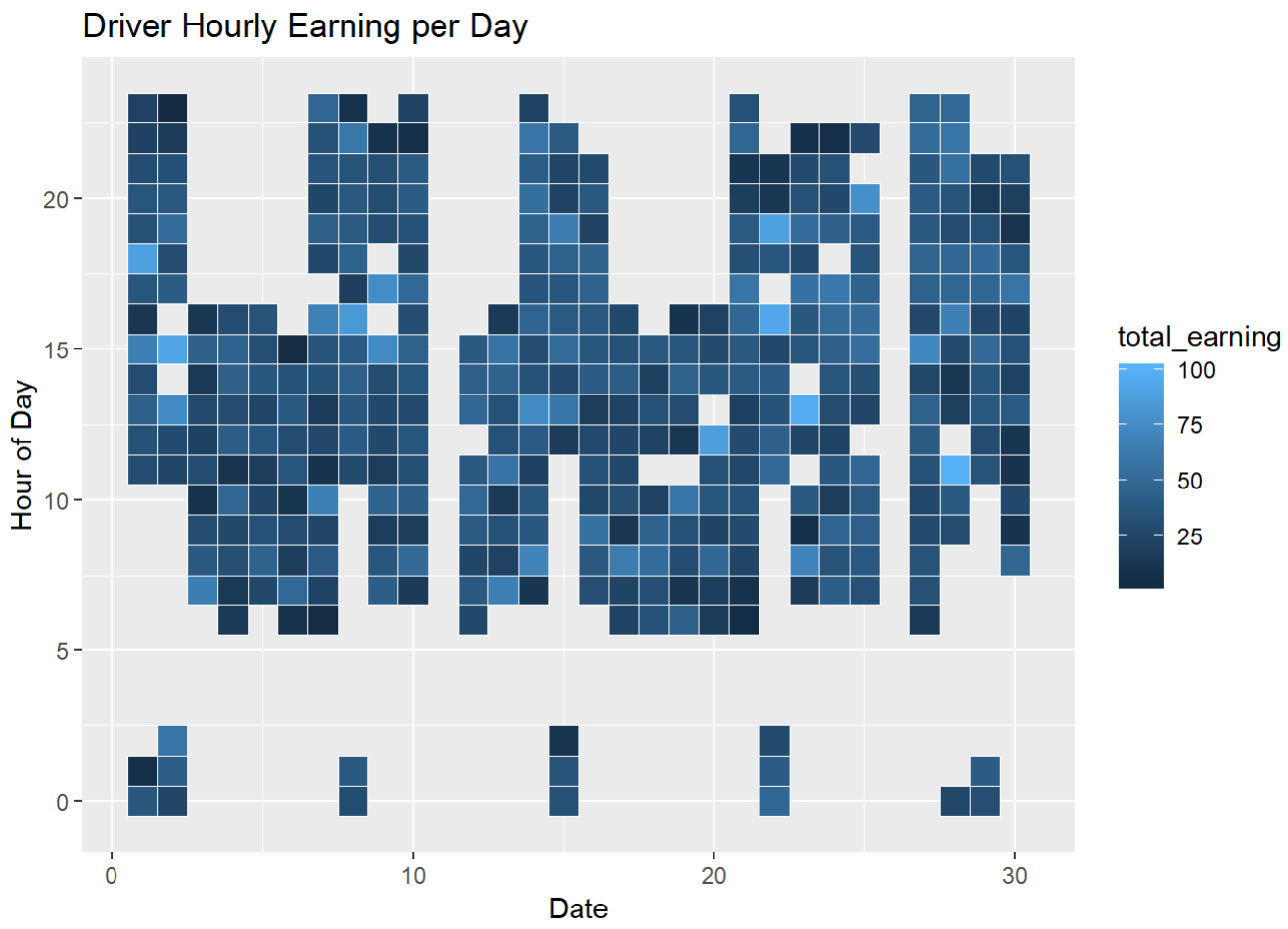


Which car did the driver use and when?

```
temp =  driver_data %>% group_by(fDay, fHour, medallion) %>% summarise("total_earning" = sum(tot
al_amount))
dateSeq = data.frame("fDay"=rep(1,24), "fHour" = rep(seq(1,24,1),30))
temp_merged = merge(dateSeq, temp, by = c("fDay", "fHour"), all.x=T)
temp_merged[which(is.na(temp_merged$total_earning)), "total_earning"] = 0

ggplot(driver_data, aes(x = pickup_datetime, y = total_amount, colour = medallion, group = fDa
y)) + geom_point() + geom_line()+ labs(x = "Pickup Hour", y = "Total Amount ($)", title = "Total
 earnings based on pickup time")+ theme(legend.position="bottom",axis.text.x = element_text(angl
e = 90, hjust = 1))+scale_x_datetime(date_breaks = "1 day", labels = date_format("%b %d"))
```



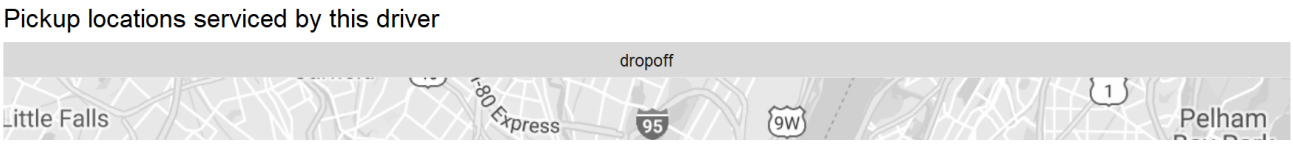

```
ggplot(temp, aes(fDay, fHour, fill = total_earning)) +
  geom_tile(colour = "white") +
  labs(x="Date",y="Hour of Day",title = "Driver Hourly Earning per Day")
```

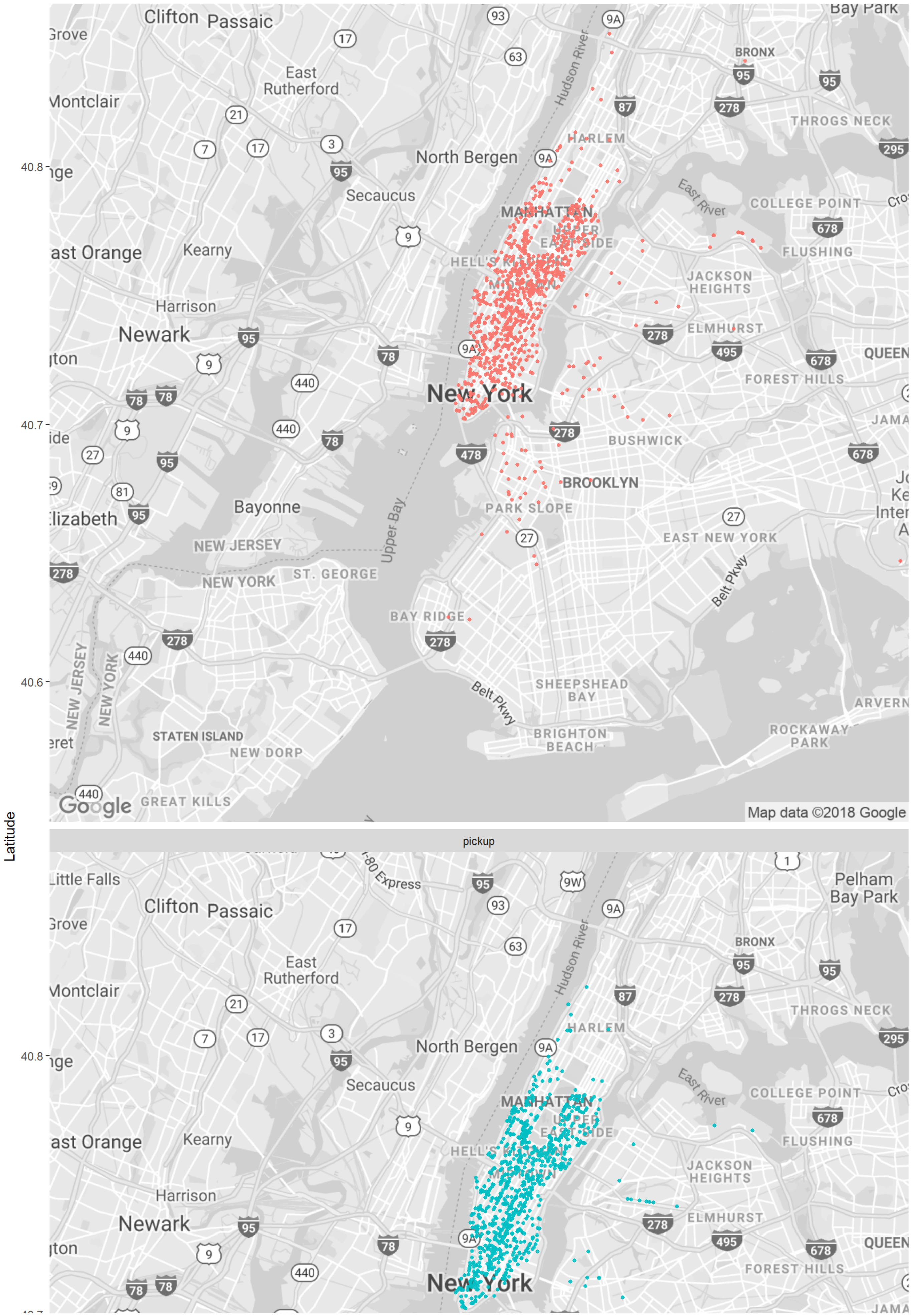


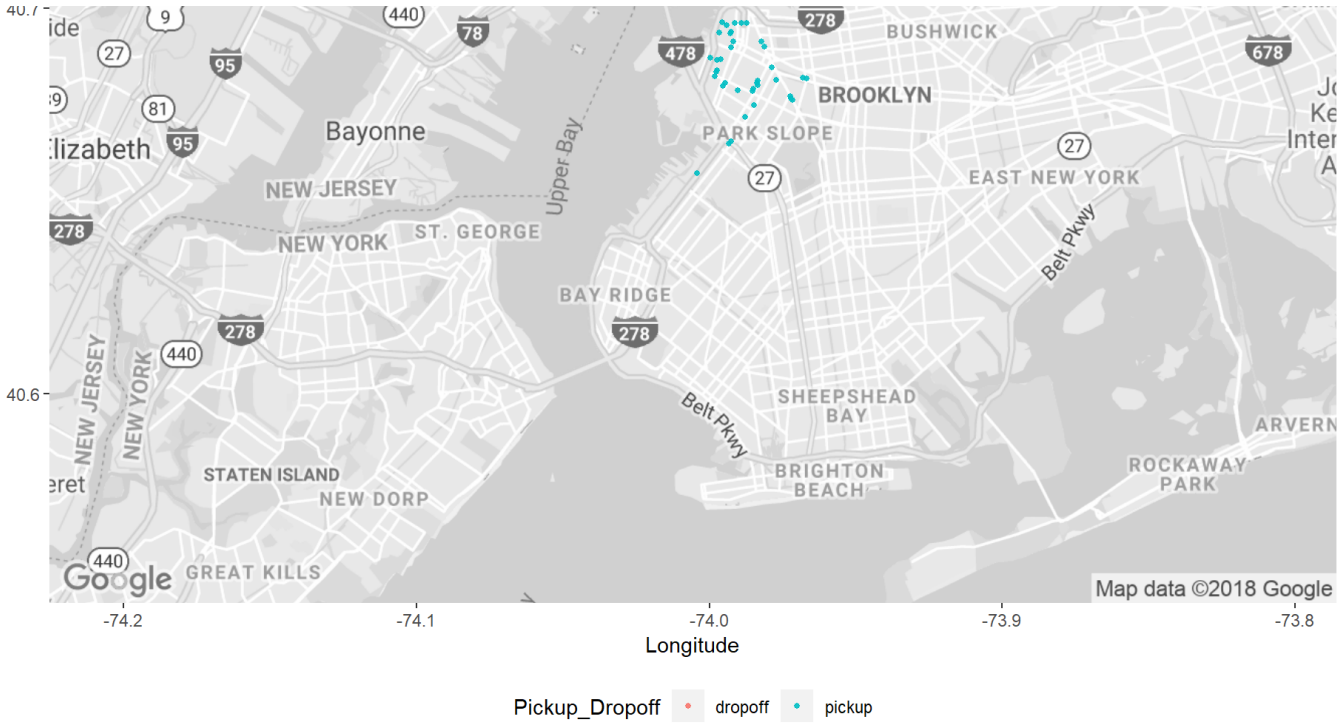
This driver seems to drive 9 different taxis probably based on which taxi is available for renting. This leads me to believe that her/his break days might not be by choice. Also, this driver seems to drive more during the daytime and less to nill during the nights.

Locations serviced by this driver

```
# combine pickup and dropoff latitudes and longitudes into long format data for plotting
temp_pickup = driver_data %>% select(pickup_longitude, pickup_latitude, fHour, fWeekday) %>% ren
ame(longitude = pickup_longitude , latitude = pickup_latitude) %>% mutate("Pickup_Dropoff" = "pi
ckup")
temp_dropoff = driver_data %>% select(dropoff_longitude, dropoff_latitude, fHour, fWeekday)  %>%
 rename(longitude = dropoff_longitude , latitude = dropoff_latitude) %>% mutate("Pickup_Dropoff"
 = "dropoff")

temp = rbind(temp_pickup, temp_dropoff)
rm(temp_dropoff, temp_pickup)

p = ggmap(nyc_map) + geom_point(aes(x = longitude, y = latitude, colour = Pickup_Dropoff), data
 = temp, size=1, alpha=0.9) + labs(x = "Longitude", y = "Latitude", title = "Pickup locations se
rviced by this driver") +
  facet_wrap(~Pickup_Dropoff, ncol = 1) + theme(legend.position="bottom")
p
```




40.8
40.7
40.6
Latitude
Map data ©2018 Google
pickup
40.8



Like most NYC taxi drivers, he drives mainly in the Manhattan area. He has some JFK airport pickups but no dropoffs in JFK for the whole month. There are also some pickup points in the Newark side which leads me to believe that he might be living (or he picks up vehicles to rent) near Newark.

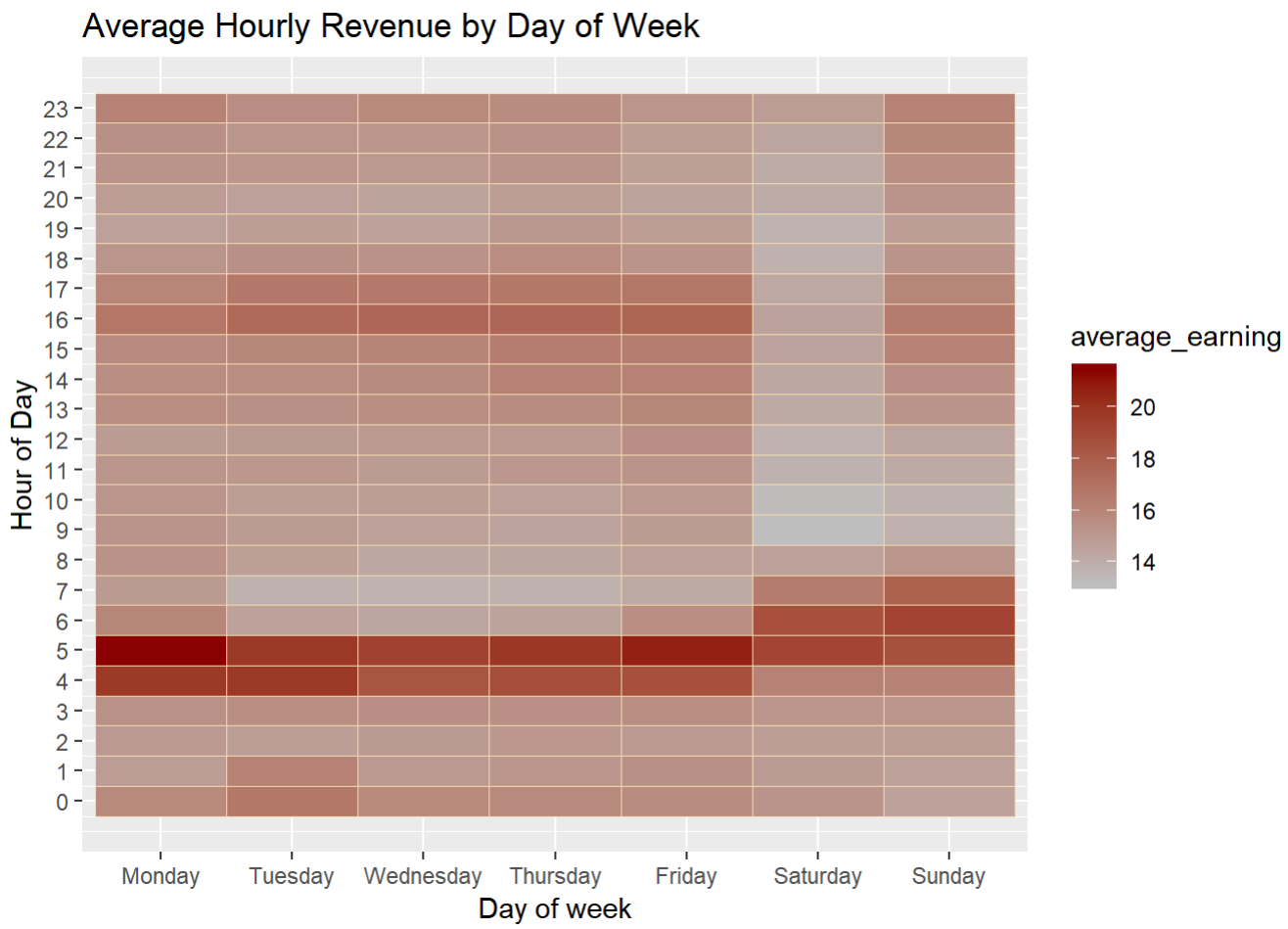
Q7. If you were a taxi owner, how would you maximize your earnings in a day?

Average income for a taxi driver can be defined as:

$$Average\ Income = Average\ revenue\ per\ hour\ \ *\ \ total\ hours\ worked$$

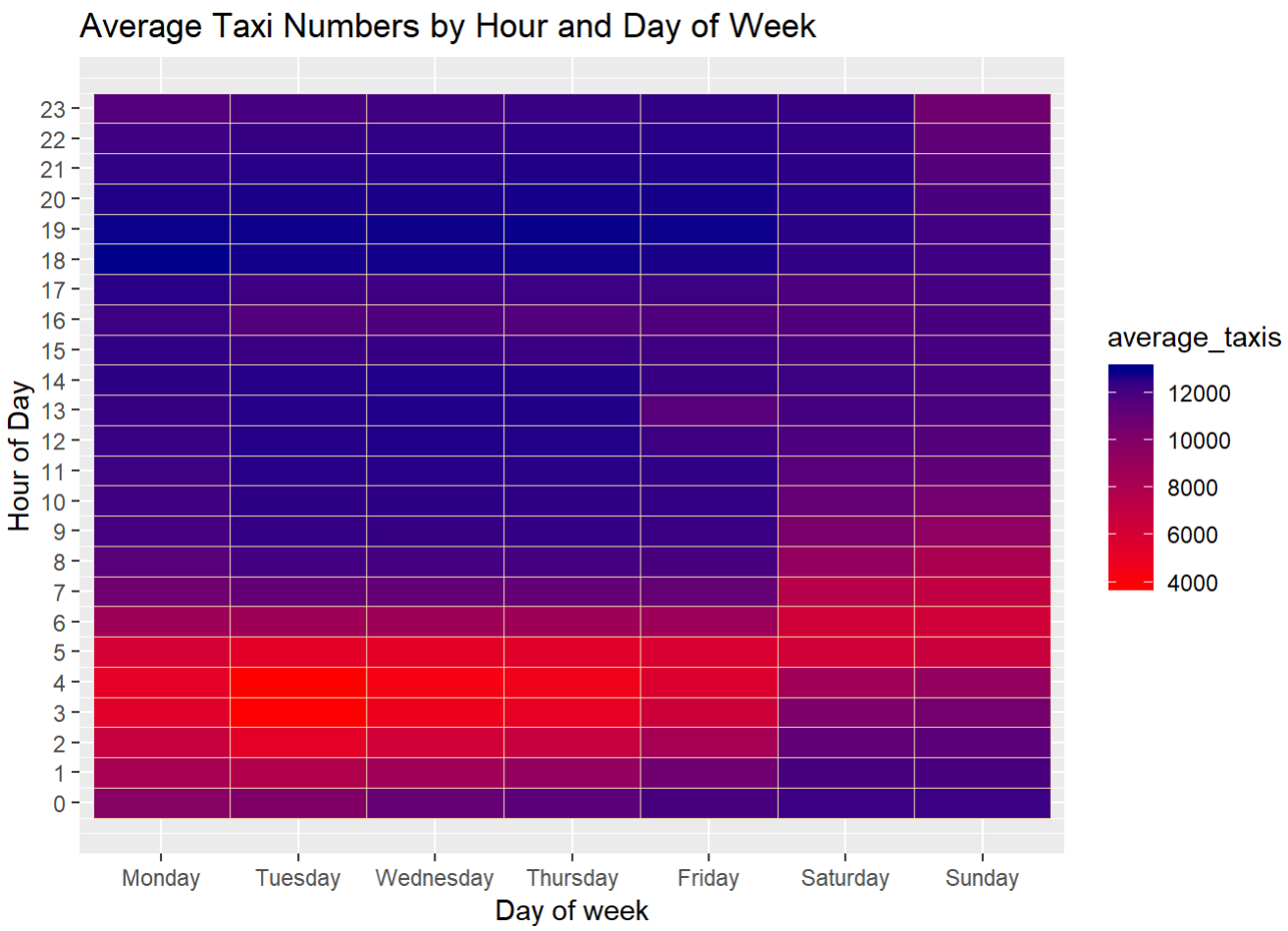
To find average revenue per hour, let us look group the data based on day of week and hour of day to understand the earning patterns per hour throughout the week.

```
temp =  taxi_data %>% group_by(fWeekday, fHour) %>% summarise("average_earning" = mean(total_amo
unt))

ggplot(temp, aes(fWeekday, fHour, fill = average_earning)) + scale_fill_gradient(low="gray", hig
h="darkred") + geom_tile(colour = "wheat") +
  labs(x="Day of week",y="Hour of Day",title = "Average Hourly Revenue by Day of Week") +scale_y
_continuous(breaks = c(0:23))
```



The plot shows that the weekends have as a slightly different pattern than the weekedays

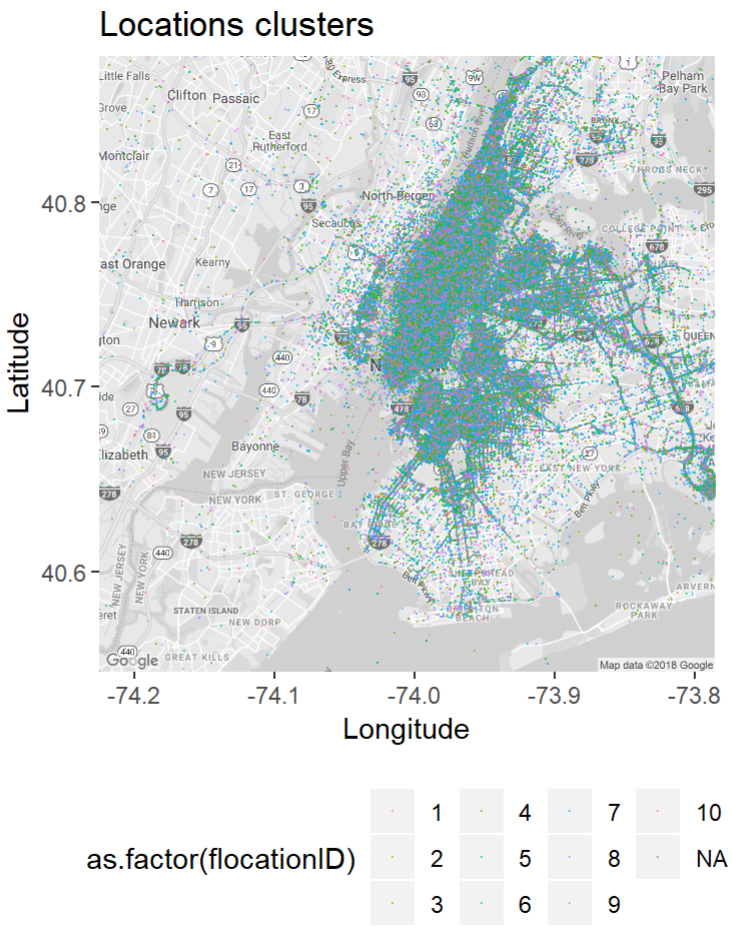
```
temp =  taxi_data %>% group_by(fWeekday, fHour) %>% summarise("average_taxis" = mean(length(uniq
ue(medallion))))

ggplot(temp, aes(fWeekday, fHour, fill = average_taxis)) + scale_fill_gradient(low="red", high=
"darkblue") + geom_tile(colour = "wheat") +
  labs(x="Day of week",y="Hour of Day",title = "Average Taxi Numbers by Hour and Day of Week") +
scale_y_continuous(breaks = c(0:23))
```



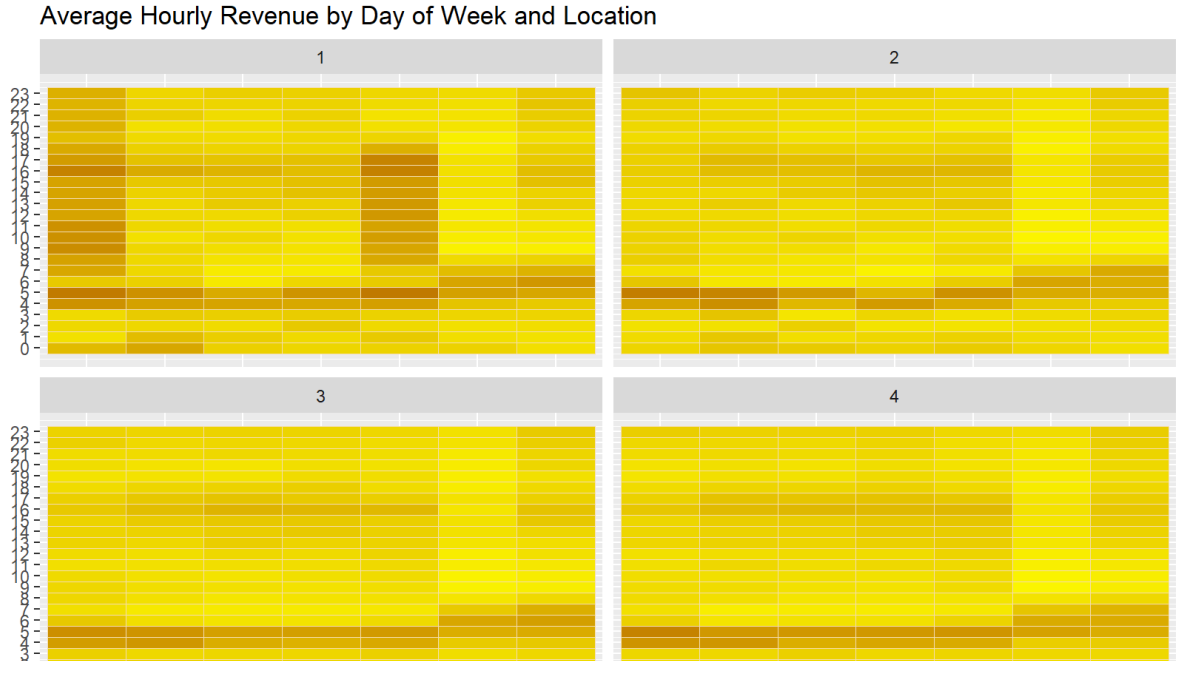
This plot confirms that price is driven by supply of taxis (comapring with the previous chart). There is a huge decrese in total taxi numbers from 2am to 6am. Consequntly, the average revenue is highest during 4am to 6am. Average revenue per hour changes with locations and neighbourhoods and type of trip taken (long-distance, short-distance). Since location is not given, we can cluster the pickup points based on average earnings and hence use cluster numbers to represent location. Pickup longitude and latitude is used for clustering as dropoff latitudes and longitudes might not be accurate as seen in earlier sections.

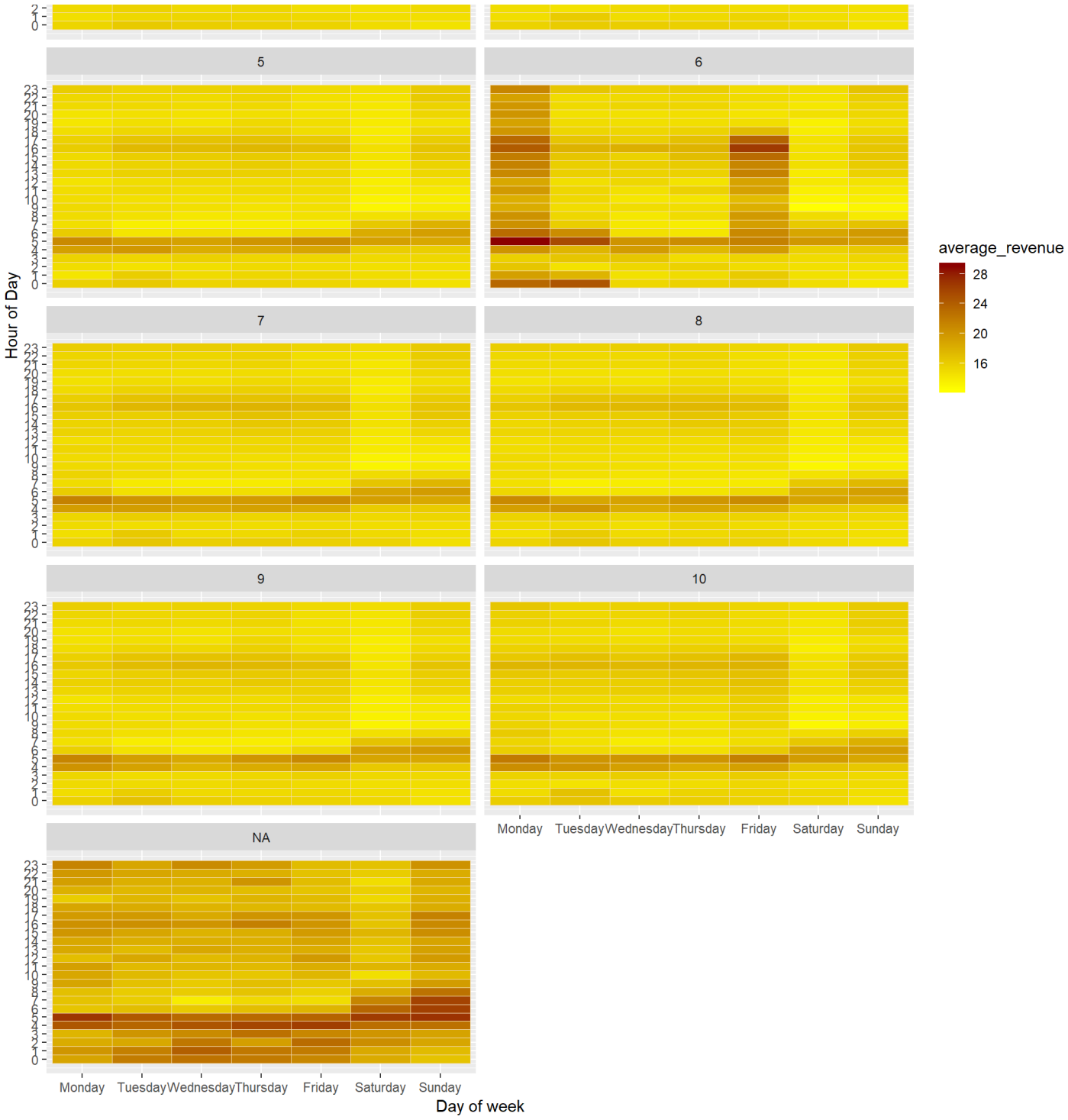
```
set.seed(847)
temp = na.omit(taxi_data[, c("pickup_longitude", "pickup_latitude")])
borough_cluster = kmeans(temp[, c("pickup_longitude", "pickup_latitude")], 10, nstart = 5)

temp$flocationID = borough_cluster$cluster

taxi_data$flocationID[which(!is.na(taxi_data$pickup_longitude))] = borough_cluster$cluster
```

```
ggmap(nyc_map) + geom_point(data = taxi_data, aes(pickup_longitude, pickup_latitude, color = as.
factor(flocationID)), size = 0.001, alpha = 0.5)+labs(x = "Longitude", y = "Latitude", title =
"Locations clusters")+ theme(legend.position="bottom")
```




```
temp =  taxi_data %>% group_by(flocationID, fWeekday, fHour) %>% summarise("average_revenue" = m
ean(total_amount))

ggplot(temp, aes(fWeekday, fHour, fill = average_revenue)) + scale_fill_gradient(low="yellow", h
igh="darkred") + geom_tile(colour = "wheat") + facet_wrap(~flocationID, ncol = 2)+
  labs(x="Day of week",y="Hour of Day",title = "Average Hourly Revenue by Day of Week and Locati
on") +scale_y_continuous(breaks = c(0:23))
```





The above plots shows that although the total number of trips are highest in Manhattan area, the revenue is low.

```
temp =  taxi_data %>% group_by(flocationID, fWeekday) %>% summarise("average_revenue" = mean(tot
al_amount), "average_trips" = mean(n()))

p1 = ggplot(temp, aes(flocationID, average_revenue, fill = fWeekday, group = fWeekday)) + geom_b
ar(stat = "identity")+ labs(x="LocationID",y="Average Revenue",title = "Average Revenue by Day o
f Week and Location") +scale_x_continuous(breaks = c(0:23))+ theme(legend.position="bottom")

p2 = ggplot(temp, aes(flocationID, average_trips, fill = fWeekday, group = fWeekday)) + geom_bar
(stat = "identity")+ labs(x="LocationID",y="Average Trips",title = "Average Trips by Day of Week
 and Location") +scale_x_continuous(breaks = c(0:23))+ theme(legend.position="bottom")

require(gridExtra)
```

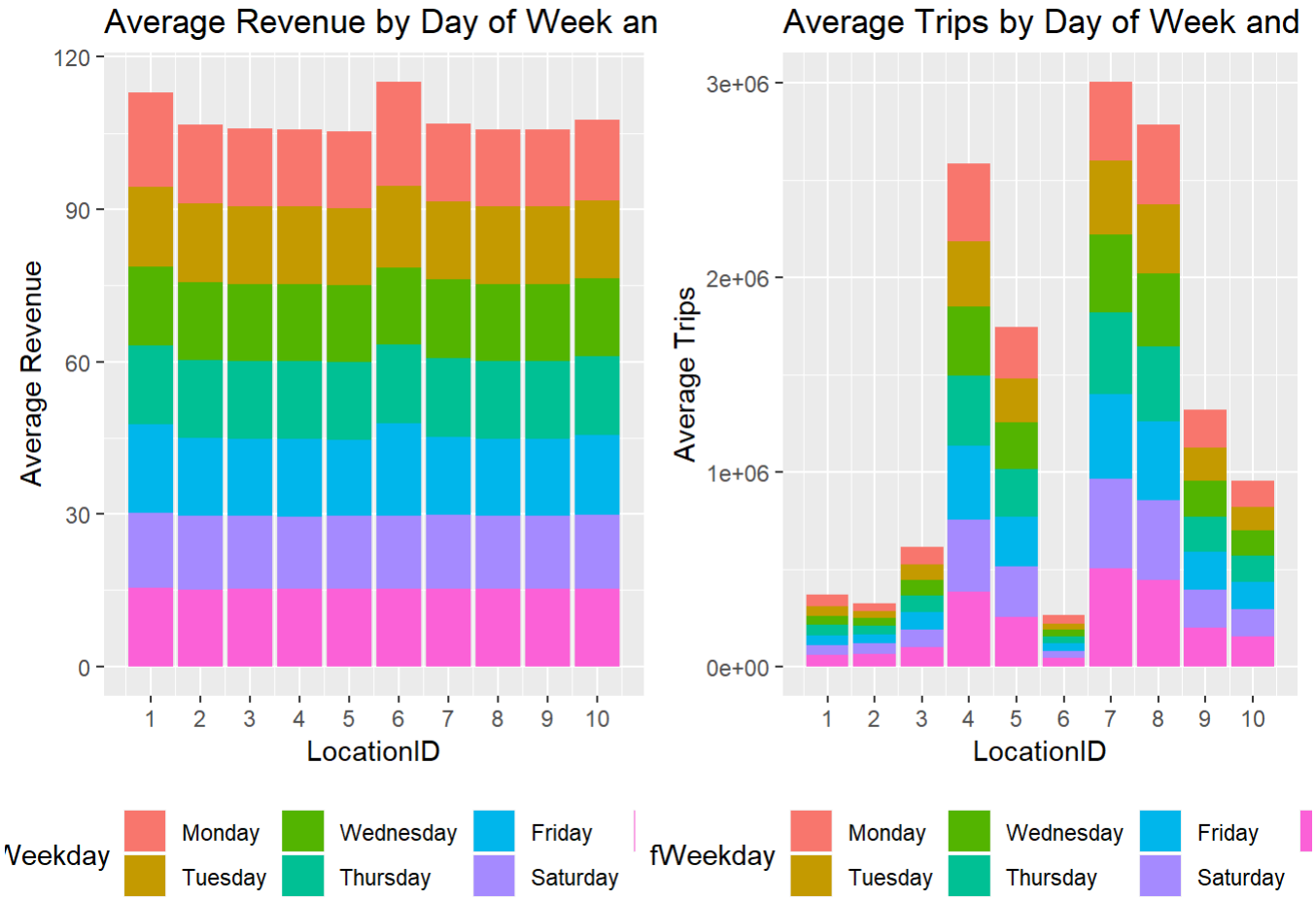
```
## Loading required package: gridExtra
```

```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
grid.arrange(p1, p2, ncol=2)
```

```
## Warning: Removed 7 rows containing missing values (position_stack).

## Warning: Removed 7 rows containing missing values (position_stack).
```



4. Data Modelling

Train Test Division

The data is split into train and test datasets randomly based on stratified sampling (70:30 split).

Medallion numbers and hack_licenses are unique codes which are not useful for predicting fare fare and tip amounts. Hence, they are not included in the modelling part. Also, variables like geo-cordinates, surcharges, mta_tax etc are excluded as they wouldn't help with the modelling and prediction.

```
# selecting only the required columns
selected_columns = c("rate_code", "passenger_count", "trip_time_in_secs", "trip_distance", "fHou
r", "fWendOrWday", "fare_amount", "tip_amount", "payment_type", "flocationID")
train_data = train_data[, selected_columns]
test_data = test_data[, selected_columns]
```

Missing Values

```
sum(is.na(train_data))
```

```
## [1] 104803
```

```
sum(is.na(test_data))
```

```
## [1] 44734
```

There are no missing values in the data.

Final Data Cleaning

In order to simulate real test data, the data cleaning is performed after train-test split. Since a taxi trip record depends a lot on human behaviour, the possible outliers would need to be closely studied before deciding to drop it. Some of the data analysis caried out to detect the outliers and study the corretness of the records are as follows:
1. trip_time - low or 0 & trip_distance - high 2. trip_distance - high & fare_amount - very low 3. trip_time - high & fare_amount - very low 4. trip_time - high & trip_distance - very low 5. trip_time - low, fare_amount - high

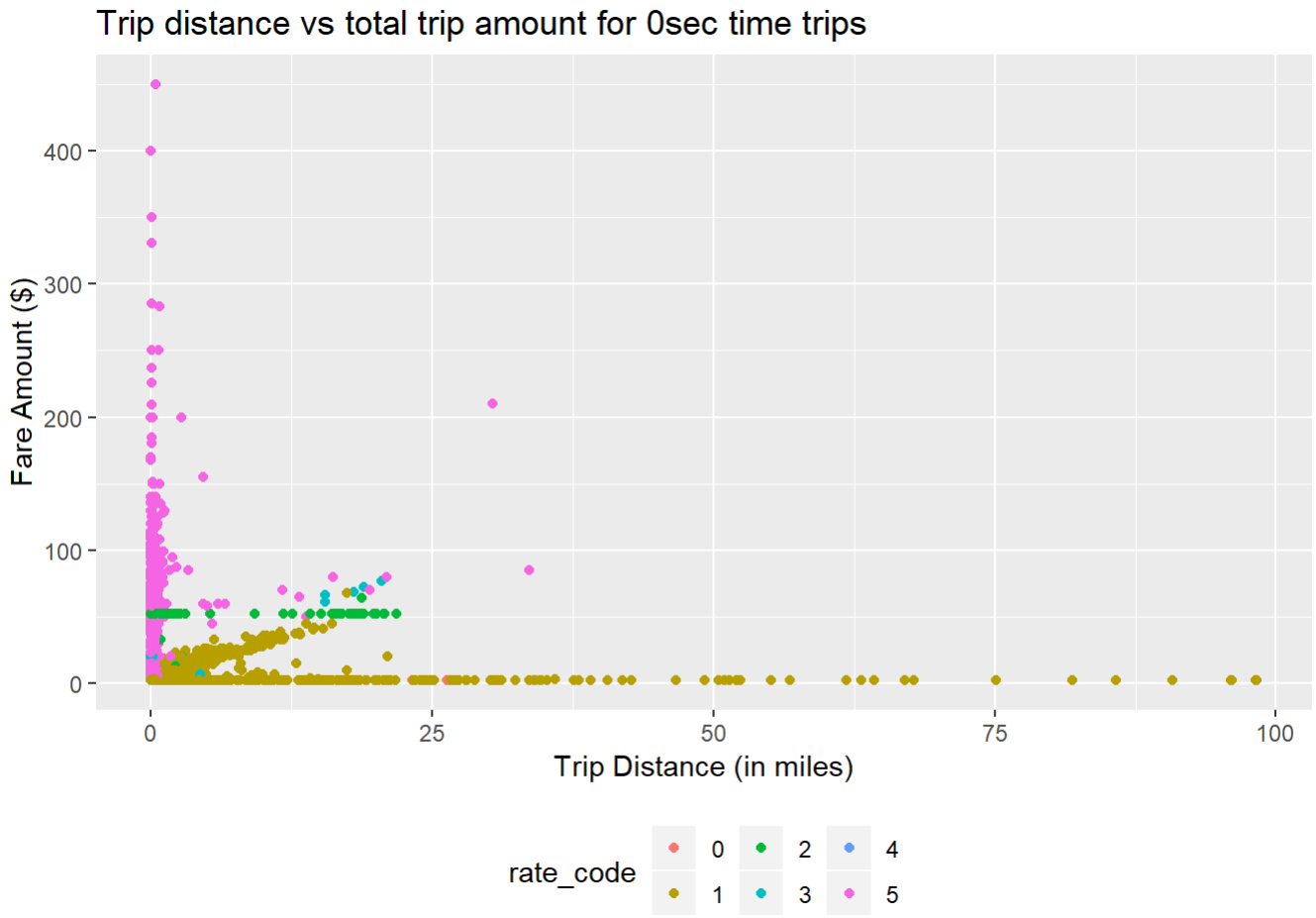
Trip distance & fare amount for trip_distance = 0

```
temp = subset(train_data, (trip_distance>0 & trip_time_in_secs == 0))

# extract where fare_amount is significant (esp. flat fares)
temp_52 = subset(temp, (fare_amount >= 52))
head(temp_52)
```

```
##       rate_code passenger_count trip_time_in_secs trip_distance fHour
## 1876          2               1                 0          0.05    10
## 8855          2               1                 0          0.07    11
## 9816          2               1                 0          0.01     0
## 10877         2               1                 0          0.05    18
## 11080         5               2                 0          0.59    16
## 12343         2               1                 0          0.01    21
##       fWendOrWday fare_amount tip_amount payment_type flocationID
## 1876      Weekday          52        0.0          CSH           7
## 8855      Weekend          52        0.0          CSH           9
## 9816      Weekday          52       10.4          CRD        <NA>
## 10877     Weekday          52       10.4          CRD           8
## 11080     Weekend         100        0.0          CRD        <NA>
## 12343     Weekend          52        0.0          CSH           1
```

Looking closely at the data where the trip time was 0 but trip distance >0, I found many of them having huge fare_amount and looks like real records (sometimes even with tips). This is really interesting! The NYC govt. website states that the taxi fares are calculated based on mainly trip_distance and a small component of trip_time. There are also flat_fare trips. (http://home.nyc.gov/html/tlc/html/industry/taxicab_rate_yellow.shtml (http://home.nyc.gov/html/tlc/html/industry/taxicab_rate_yellow.shtml))
JFK trips have a flat rate of $52 irrespective of the distance. Here is what I think is really happening. The trip time, geo-coordinate locations etc are recorded automatically when the driver presses end trip. During airport trips like JFK, people are in a hurry. Since it is a flat rate fee, they might be wanting to finish transactions before reaching their destination. This is an interesting find as this means, there are a lot more airport trips than shown by dropoff points. This also means the data can be erronous due to such usage methods and I have to do more creative processing.

```
# Trip patterns for 0sec travels with >0 fare amount
temp = subset(temp, (fare_amount >=0))

ggplot(temp, aes(x = trip_distance, y = fare_amount, colour= rate_code)) + geom_point() + labs(x
 = "Trip Distance (in miles)", y = "Fare Amount ($)", title = "Trip distance vs total trip amoun
t for 0sec time trips")+ theme(legend.position="bottom")
```



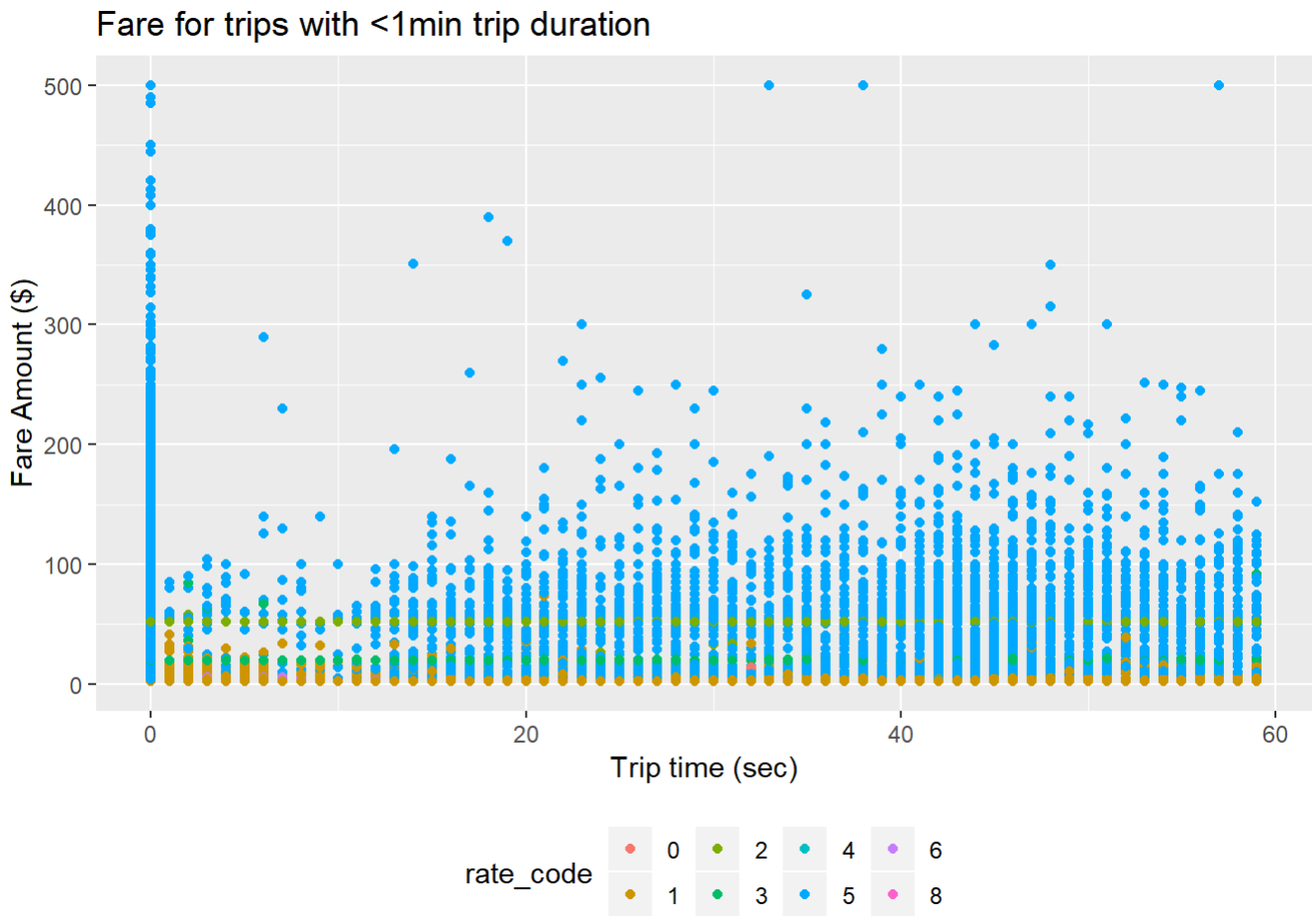
These records although genuine would lead to erronous predictions as trip time and trip distance are important attributes for modelling fare and tip. Hence, dropping the records with trip time = 0 but trip distance >0 and trip fare >0

```
train_data = subset(train_data, !(trip_time_in_secs ==0 & trip_distance >0 & fare_amount >0))
test_data = subset(test_data, !(trip_time_in_secs ==0 & trip_distance >0 & fare_amount >0))
```

Fare amount for trip times < 1minute

Also, looking at trips where the trip time is less than a minute:

```
temp = subset(train_data, (trip_time_in_secs < 60))

ggplot(temp, aes(x = trip_time_in_secs, y = fare_amount, colour= rate_code)) + geom_point() + la
bs(x = "Trip time (sec)", y = "Fare Amount ($)", title = "Fare for trips with <1min trip duratio
n")+ theme(legend.position="bottom")
```



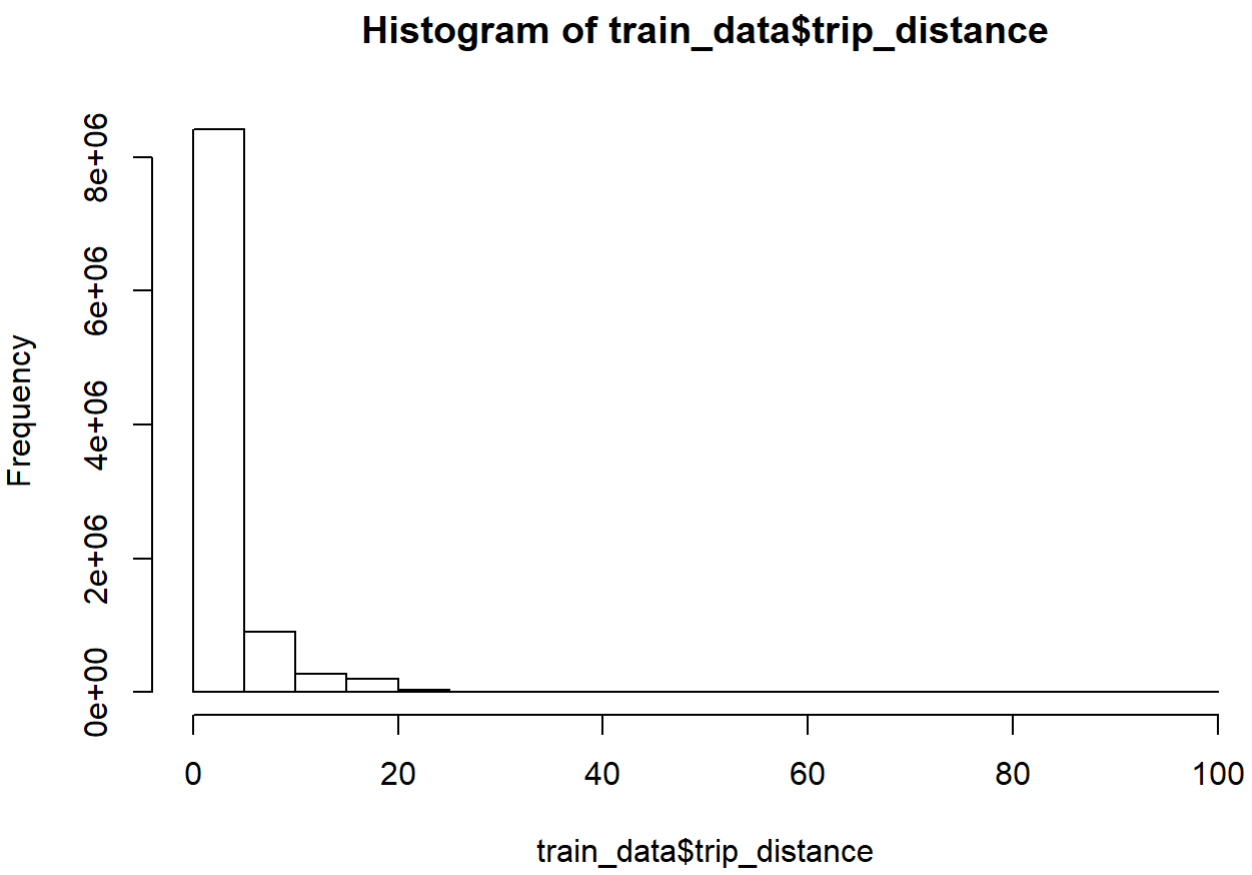
Studying the above data and plots closely, the following are the discoveries:
* The trip_time and geo coordinates are recorded when the driver processes the payment. * The trip_distance is only recorded at the end of the trip.
* There are also flat_fare trips (E.g., rate_code = 2 for JFK Airport). * Some of the minimum distance ($2.5 for first 1/5 mile) trips also look like flat_fare trips with the driver processing the payment before trip_end.
* Fare for Rate_code = 5 (negotiated fares) are entered at the start of the trip itself. Most drivers mistakenly process the payment too.
* Most drivers forget to stop the trip (an meter) if the payment was processed earlier (like flat_fare, min_charge and rate_code = 5 trips) leading to enormously high trip_distance records.
* It also looks like different rate_codes have different trip patterns.

```
# removing all records with trip time == 0
train_data = subset(train_data, !(trip_time_in_secs == 0))
test_data = subset(test_data, !(trip_time_in_secs == 0))
```

```
#removing data with trip distance == 0
train_data = subset(train_data, !(trip_distance == 0))
test_data = subset(test_data, !(trip_distance == 0))
```

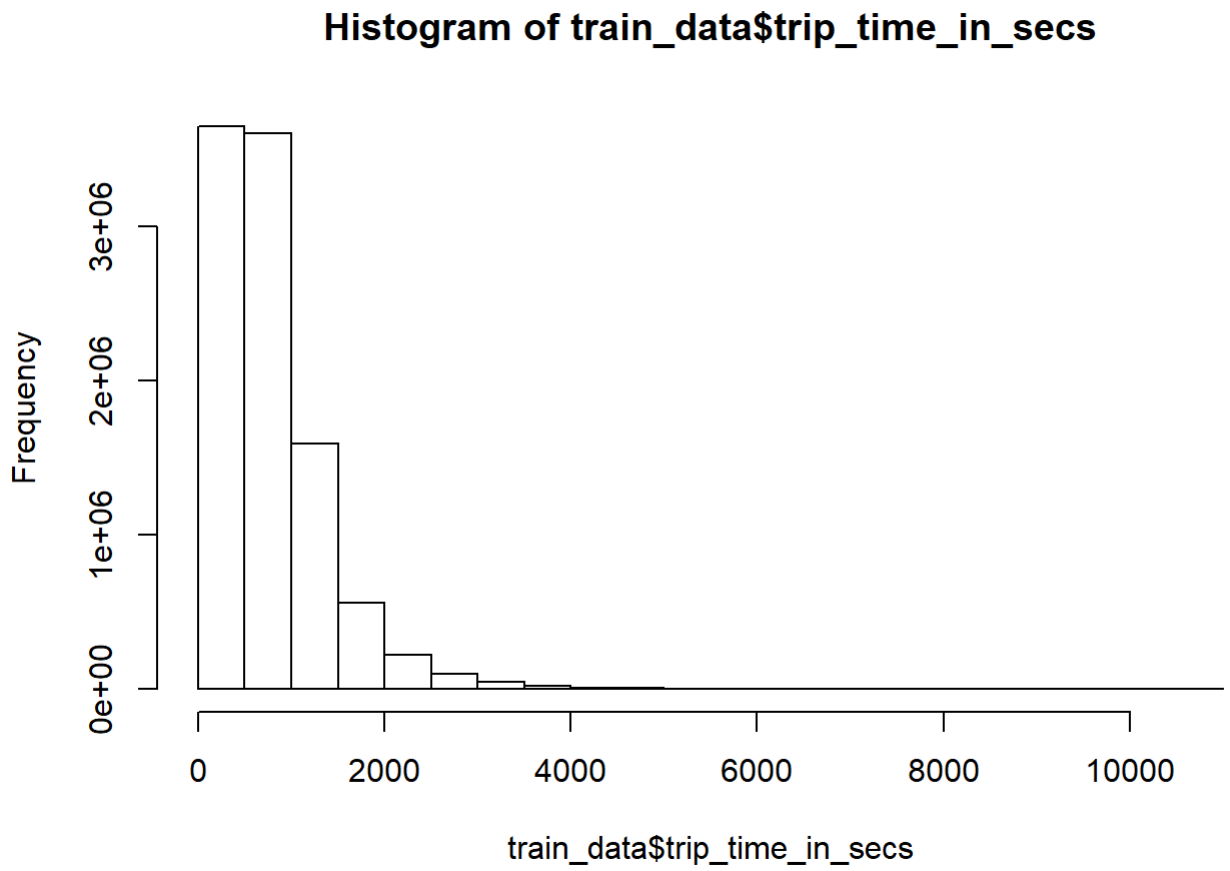
A lot of data especially those repsenting flat fares and negotiated flat fares would be lost. But since outliers would affect the predictionability of models, omitting them. Even now a lot of incorrect records are present like when the driver forgets to stop a trip after dropoff leading to high trip_distances for minimum fare and flat fares. Ideally more detailes study would be required but in the interest of time, dropping all extreme values detected based on the below histogram.

```
hist(train_data$trip_distance)
```



It looks like 30 miles is a descent assumption as it would covere trips till Newark. But this would need to be studied further in combination with fare amount and trip_time to better develop outlier ranges.

```
hist(train_data$trip_time_in_secs)
```



These data represents behavioural patterns in drivers as well as passengers. Ideally trip distance and trip time should be more than enough to model the fare_amount but due to all the data entry errors and driver/passenger behaviour, linear regression models might not be able to model the fare and tip amount accurately. I am expecting a non-linear ML algorithm like decision trees to model this data better.

Q5. Features influencing fare and tip

Total trip amount can be calculated is given below:

$$Total\ amount = Fare\ amount + surcharge + mta\ tax + tip\ amount + tolls\ amount$$

Although NYC government websites states that fare_amount is dependent on trip_distance and trip_time, actual fare_amount depends a lot on other variables like rate_code and a combination of the various variables (from the trip behaviour analysis from the previous section).

Strength of correlation between two numeric variables also shows their dependence on each other. Fare_amount definitely depends on trip_time and trip_distance.

```
cat("Correlation between Fare and trip_time: ", cor(train_data$fare_amount, train_data$trip_time
_in_secs))
```

```
## Correlation between Fare and trip_time:  0.8411311
```

```
cat("Correlation between Fare and trip_distance: ", cor(train_data$fare_amount, train_data$trip_
distance))
```

```
## Correlation between Fare and trip_distance:  0.943212
```

As found earlier, the tip amount depends on the payment_type and various other factors.

```
cat("Correlation between Tip and trip_time: ", cor(train_data$trip_time_in_secs, train_data$tip_
amount))
```

```
## Correlation between Tip and trip_time:  0.4382922
```

```
cat("Correlation between Tip and trip_time: ", cor(train_data$trip_distance, train_data$tip_amou
nt))
```

```
## Correlation between Tip and trip_time:  0.5050622
```

Usefulness of each variable can be tested using various metrics like gini index, entropy etc. Since I plan to use CART decision trees to do the base model, CART output gives gini index to represent variable importance.

Data Standradization

```
std_model = preProcess(train_data[, c("trip_time_in_secs", "trip_distance")], method = c("cente
r", "scale"))

# The predict() function is used to standardize any other unseen data
train_data[, c("trip_time_in_secs", "trip_distance")] = predict(object = std_model, newdata = tr
ain_data[, c("trip_time_in_secs", "trip_distance")])

test_data[, c("trip_time_in_secs", "trip_distance")] = predict(object = std_model, newdata = tes
t_data[, c("trip_time_in_secs", "trip_distance")])
```

Data standardization is important so that variables with higher scales do not unduly influcence the coefficients.

Data Modelling for fare_amount

Variable Importance and decision tree modelling for fare_amount

Tip is given after the trip and hence excluded from analysis for prediting fare.

Linear regression modelling

Although linear regression is not expcted to do a good job modelling this data, I attempted it just to see variable correlations and interactions.

```
lm_model = lm(formula = fare_amount~. , data = train_data[, !(colnames(train_data) %in% c("tip_a
mount", "flocationID"))])

summary(lm_model)
```

```
##
## Call:
## lm(formula = fare_amount ~ ., data = train_data[, !(colnames(train_data) %in%
##     c("tip_amount", "flocationID"))])
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -216.67   -0.40   -0.12    0.27  458.31
##
## Coefficients:
##                       Estimate Std. Error  t value Pr(>|t|)
## (Intercept)         1.278e+01  1.386e-01   92.248  < 2e-16 ***
## rate_code1         -2.120e-01  1.385e-01   -1.530 0.125926
## rate_code2          1.694e+00  1.387e-01   12.213  < 2e-16 ***
## rate_code210        2.566e+00  9.795e-01    2.620 0.008802 **
## rate_code28        -1.437e+01  2.173e+00   -6.612 3.80e-11 ***
## rate_code3          1.971e+01  1.397e-01  141.038  < 2e-16 ***
## rate_code4          1.321e+01  1.433e-01   92.159  < 2e-16 ***
## rate_code5          3.871e+01  1.398e-01  276.997  < 2e-16 ***
## rate_code6         -7.148e+00  3.484e-01  -20.514  < 2e-16 ***
## rate_code7         -1.304e+01  1.540e+00   -8.471  < 2e-16 ***
## rate_code9          1.970e-02  1.260e+00    0.016 0.987523
## passenger_count    -7.113e-03  5.023e-04  -14.162  < 2e-16 ***
## trip_time_in_secs   3.103e+00  1.119e-03 2773.325  < 2e-16 ***
## trip_distance       6.699e+00  1.282e-03 5224.544  < 2e-16 ***
## fHour1             -2.864e-02  5.243e-03   -5.463 4.68e-08 ***
## fHour10             1.812e-01  4.766e-03   38.011  < 2e-16 ***
## fHour11             1.770e-01  4.727e-03   37.444  < 2e-16 ***
## fHour12             2.066e-01  4.669e-03   44.260  < 2e-16 ***
## fHour13             1.604e-01  4.680e-03   34.274  < 2e-16 ***
## fHour14             1.290e-01  4.656e-03   27.695  < 2e-16 ***
## fHour15             4.898e-02  4.704e-03   10.413  < 2e-16 ***
## fHour16             1.895e-02  4.924e-03    3.847 0.000119 ***
## fHour17             5.905e-02  4.680e-03   12.617  < 2e-16 ***
## fHour18             2.026e-02  4.469e-03    4.533 5.82e-06 ***
## fHour19            -4.983e-02  4.413e-03  -11.291  < 2e-16 ***
## fHour2             -5.294e-02  5.692e-03   -9.300  < 2e-16 ***
## fHour20            -8.406e-02  4.436e-03  -18.949  < 2e-16 ***
## fHour21            -7.748e-02  4.473e-03  -17.322  < 2e-16 ***
## fHour22            -5.189e-02  4.495e-03  -11.543  < 2e-16 ***
## fHour23            -1.158e-02  4.617e-03   -2.509 0.012124 *
## fHour3             -2.846e-02  6.332e-03   -4.496 6.93e-06 ***
## fHour4              1.560e-01  7.140e-03   21.843  < 2e-16 ***
## fHour5              3.701e-01  7.716e-03   47.966  < 2e-16 ***
## fHour6              1.541e-02  5.933e-03    2.597 0.009391 **
## fHour7             -1.110e-01  5.026e-03  -22.092  < 2e-16 ***
## fHour8              8.884e-02  4.806e-03   18.484  < 2e-16 ***
## fHour9              2.071e-01  4.771e-03   43.398  < 2e-16 ***
## fWendOrWdayWeekend -2.729e-02  1.551e-03  -17.593  < 2e-16 ***
## payment_typeCSH    -1.132e-01  1.405e-03  -80.537  < 2e-16 ***
## payment_typeDIS    -1.010e+00  2.867e-02  -35.224  < 2e-16 ***
## payment_typeNOC    -1.526e+00  1.628e-02  -93.734  < 2e-16 ***
## payment_typeUNK    -1.823e-02  1.830e-02   -0.996 0.319305
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.168 on 9811185 degrees of freedom
## Multiple R-squared:  0.9533, Adjusted R-squared:  0.9533
## F-statistic: 4.89e+06 on 41 and 9811185 DF,  p-value: < 2.2e-16
```

```
#
# library(MASS)
#
# lm_model_aic = stepAIC(lm_model, direction = "both")
#
# summary(lm_model_aic)
#
# #par(mfrow = c(2,2))
# #plot(lm_model_aic)
```

Multi-colinearity Detection

A VIF > 4 means that there are signs of multi-collinearity and anything greater than 10 means that an explanatory variable should be dropped.

```
library(car)
```

```
## Loading required package: carData
```

```
##
## Attaching package: 'car'
```

```
## The following object is masked from 'package:dplyr':
##
##     recode
```

```
vif(lm_model)
```

```
##                       GVIF Df GVIF^(1/(2*Df))
## rate_code         1.645668 10        1.025220
## passenger_count   1.003062  1        1.001530
## trip_time_in_secs 2.611725  1        1.616083
## trip_distance     3.431320  1        1.852382
## fHour             1.170922 23        1.003436
## fWendOrWday       1.066114  1        1.032528
## payment_type      1.016221  4        1.002013
```

```
# prediction on test dat
preds_lm = predict(lm_model, test_data)

regr.eval(test_data$fare_amount, preds_lm)
```

```
##        mae        mse       rmse       mape
## 0.64747816 4.63255665 2.15233749 0.05483231
```

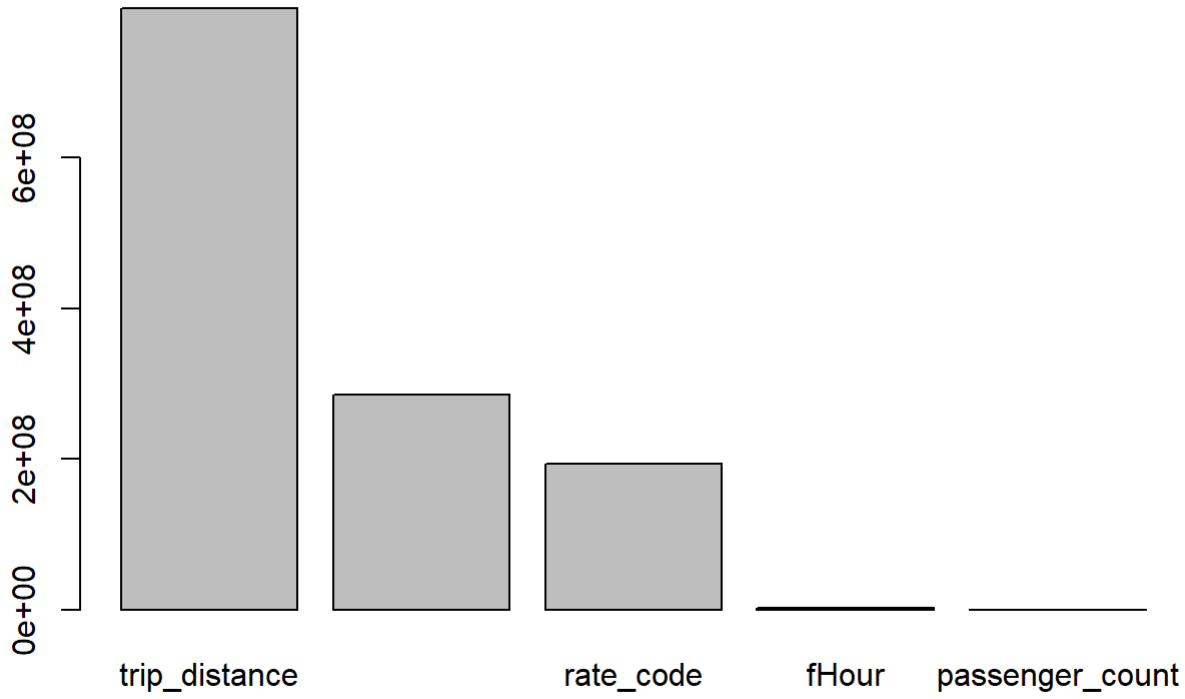
Surprisingly linear regression performs good!

Decision Tree modelling

```
library(rpart)

cart_model = rpart(fare_amount ~ ., data = train_data[, !(colnames(train_data) == "tip_amount"
)])

cart_model$variable.importance
```

```
##     trip_distance trip_time_in_secs         rate_code             fHour
##      7.982529e+08      2.852193e+08      1.935582e+08      2.755635e+06
##   passenger_count
##      7.051221e+03
```

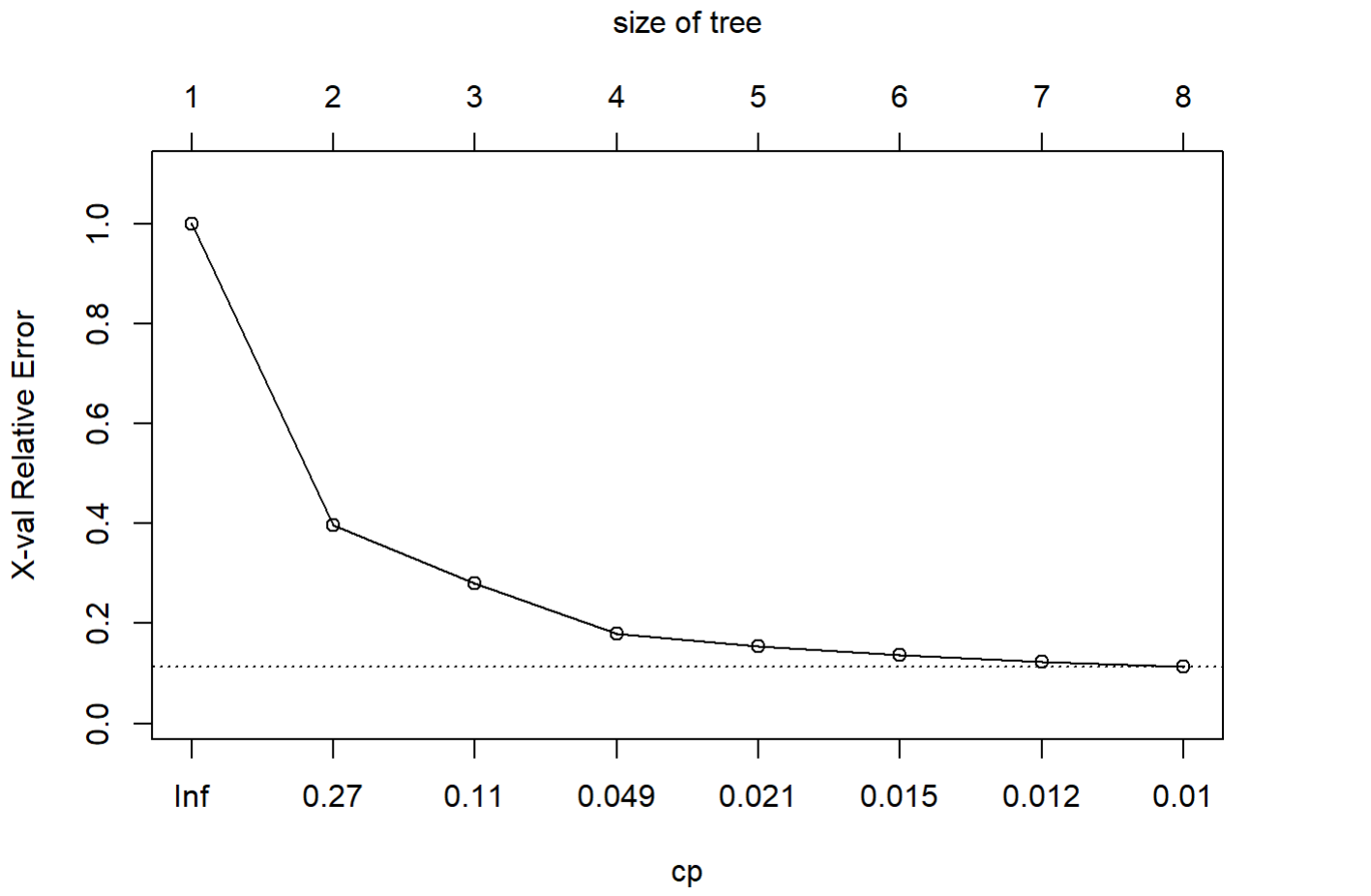
```
barplot(cart_model$variable.importance)
```




```
printcp(cart_model)
```

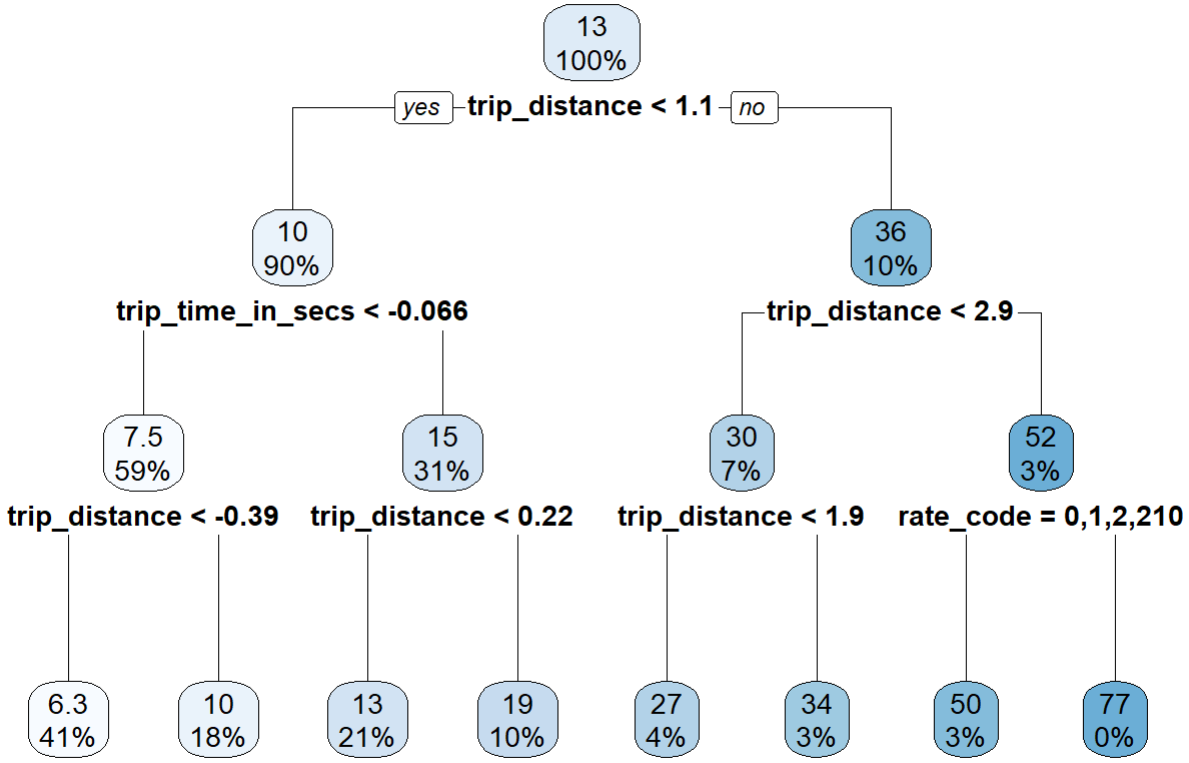
```
##
## Regression tree:
## rpart(formula = fare_amount ~ ., data = train_data[, !(colnames(train_data) ==
##     "tip_amount")])
##
## Variables actually used in tree construction:
## [1] rate_code         trip_distance     trip_time_in_secs
##
## Root node error: 988773301/9811227 = 100.78
##
## n= 9811227
##
##         CP nsplit rel error  xerror      xstd
## 1 0.602352      0   1.00000 1.00000 0.0016977
## 2 0.117643      1   0.39765 0.39765 0.0012579
## 3 0.101268      2   0.28001 0.27998 0.0012576
## 4 0.024102      3   0.17874 0.17876 0.0011508
## 5 0.017915      4   0.15464 0.15465 0.0011480
## 6 0.012984      5   0.13672 0.13673 0.0011482
## 7 0.010241      6   0.12374 0.12375 0.0010523
## 8 0.010000      7   0.11350 0.11353 0.0010503
```

```
plotcp(cart_model)
```



```
# Hence the default cp of 0.01 is appropriate.
```

```
library(rpart.plot)

rpart.plot(cart_model)
```




```
library(rattle)
```

```
## Rattle: A free graphical interface for data science with R.
## Version 5.2.0 Copyright (c) 2006-2018 Togaware Pty Ltd.
## Type 'rattle()' to shake, rattle, and roll your data.
```

```
asRules(cart_model)
```

```
##
##  Rule number: 8 [fare_amount=6.28195180715012 cover=3975403 (41%)]
##    trip_distance< 1.059
##    trip_time_in_secs< -0.0662
##    trip_distance< -0.3855
##
##  Rule number: 10 [fare_amount=13.133929253795 cover=2073666 (21%)]
##    trip_distance< 1.059
##    trip_time_in_secs>=-0.0662
##    trip_distance< 0.2172
##
##  Rule number: 9 [fare_amount=10.0622681247446 cover=1801184 (18%)]
##    trip_distance< 1.059
##    trip_time_in_secs< -0.0662
##    trip_distance>=-0.3855
##
##  Rule number: 11 [fare_amount=19.0667785473909 cover=1005267 (10%)]
##    trip_distance< 1.059
##    trip_time_in_secs>=-0.0662
##    trip_distance>=0.2172
##
##  Rule number: 12 [fare_amount=26.564753649793 cover=404201 (4%)]
##    trip_distance>=1.059
##    trip_distance< 2.922
##    trip_distance< 1.918
##
##  Rule number: 13 [fare_amount=34.4987231144062 cover=267197 (3%)]
##    trip_distance>=1.059
##    trip_distance< 2.922
##    trip_distance>=1.918
##
##  Rule number: 14 [fare_amount=50.272150686848 cover=264469 (3%)]
##    trip_distance>=1.059
##    trip_distance>=2.922
##    rate_code=0,1,2,210
##
##  Rule number: 15 [fare_amount=76.6466436491936 cover=19840 (0%)]
##    trip_distance>=1.059
##    trip_distance>=2.922
##    rate_code=3,4,5,6
```

```
# prediction on test data
preds_reg = predict(cart_model, test_data)

library(DMwR)

regr.eval(test_data$fare_amount, preds_reg)
```

```
##        mae        mse       rmse       mape
##  1.8753570 11.6003952  3.4059353  0.1781663
```

```
library(h2o)
```

```
##
## ----------------------------------------------------------------------
##
## Your next step is to start H2O:
##     > h2o.init()
##
## For H2O package documentation, ask for help:
##     > ??h2o
##
## After starting H2O, you can use the Web UI at http://localhost:54321
## For more information visit http://docs.h2o.ai
##
## ----------------------------------------------------------------------
```

```
##
## Attaching package: 'h2o'
```

```
## The following objects are masked from 'package:lubridate':
##
##     day, hour, month, week, year
```

```
## The following objects are masked from 'package:stats':
##
##     cor, sd, var
```

```
## The following objects are masked from 'package:base':
##
##     %*%, %in%, &&, ||, apply, as.factor, as.numeric, colnames,
##     colnames<-, ifelse, is.character, is.factor, is.numeric, log,
##     log10, log1p, log2, round, signif, trunc
```

```
h2o.init(nthreads = -1)
```

```
##  Connection successful!
##
## R is connected to the H2O cluster:
##     H2O cluster uptime:         5 hours 45 minutes
##     H2O cluster timezone:       Australia/Sydney
##     H2O data parsing timezone:  UTC
##     H2O cluster version:        3.20.0.2
##     H2O cluster version age:    2 months and 22 days
##     H2O cluster name:           H2O_started_from_R_Lakshmi_srr767
##     H2O cluster total nodes:    1
##     H2O cluster total memory:   3.12 GB
##     H2O cluster total cores:    4
##     H2O cluster allowed cores:  4
##     H2O cluster healthy:        TRUE
##     H2O Connection ip:          localhost
##     H2O Connection port:        54321
##     H2O Connection proxy:       NA
##     H2O Internal Security:      FALSE
##     H2O API Extensions:         Algos, AutoML, Core V3, Core V4
##     R Version:                  R version 3.5.1 (2018-07-02)
```

```
train_data.hex = as.h2o(train_data,destination_frame = "train_data.hex")
```

```
##
  |
  |                                                                      |   0%
  |
  |======================================================================| 100%
```

```
train_data.gbm = h2o.gbm(y = "fare_amount", x = setdiff(colnames(train_data), c("fare_amount",
"tip_amount")), training_frame = train_data.hex, ntrees = 10, max_depth = 3,min_rows = 2, learn_
rate = 0.2, distribution= "gaussian")
```

```
##
  |
  |                                                                      |   0%
  |
  |=======                                                               |  10%
  |
  |==============                                                        |  20%
  |
  |=====================                                                 |  30%
  |
  |============================                                          |  40%
  |
  |===================================                                   |  50%
  |
  |==========================================                            |  60%
  |
  |================================================                      |  70%
  |
  |=========================================================             |  80%
  |
  |================================================================      |  90%
  |
  |======================================================================| 100%
```

```
train_data.gbm@model$scoring_history
```

```
## Scoring History:
##             timestamp   duration number_of_trees training_rmse training_mae
## 1 2018-09-07 04:22:06  0.000 sec               0      10.03891      6.58841
## 2 2018-09-07 04:22:09  3.034 sec               1       8.28383      5.40706
## 3 2018-09-07 04:22:15  9.405 sec               4       4.90506      3.06108
## 4 2018-09-07 04:22:21 15.676 sec               7       3.19128      1.79862
## 5 2018-09-07 04:22:28 21.945 sec              10       2.34240      1.10722
##   training_deviance
## 1         100.77978
## 2          68.62181
## 3          24.05960
## 4          10.18426
## 5           5.48683
```

```
train_data.gbm
```

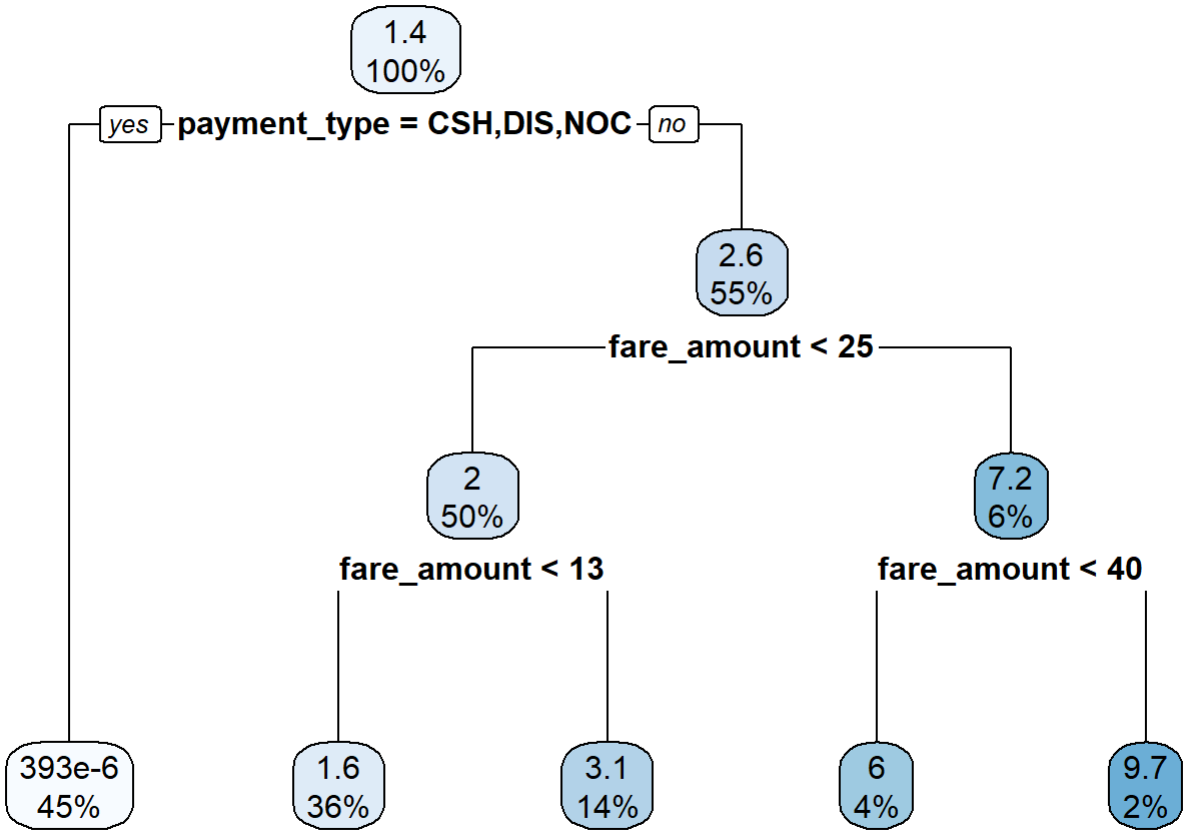
```
## Model Details:
## ==============
##
## H2ORegressionModel: gbm
## Model ID:  GBM_model_R_1536237223941_2
## Model Summary:
##   number_of_trees number_of_internal_trees model_size_in_bytes min_depth
## 1              10                       10                1620         3
##   max_depth mean_depth min_leaves max_leaves mean_leaves
## 1         3    3.00000          8          8     8.00000
##
##
## H2ORegressionMetrics: gbm
## ** Reported on training data. **
##
## MSE:  5.486825
## RMSE:  2.342397
## MAE:  1.107221
## RMSLE:  0.1425794
## Mean Residual Deviance :  5.486825
```

modelling for tip amount

```
cart_model_tip = rpart(tip_amount ~ ., data = train_data)
library(rpart.plot)
cart_model_tip$variable.importance
```

```
##       fare_amount      payment_type     trip_distance trip_time_in_secs
##      1.785681e+07      1.599544e+07      1.434640e+07      7.401125e+06
##         rate_code             fHour   passenger_count
##      4.227087e+06      1.138149e+04      3.571135e+02
```

```
rpart.plot(cart_model_tip)
```




```
# prediction on test data
preds_reg_tip = predict(cart_model_tip, test_data)

library(DMwR)

regr.eval(test_data$tip_amount, preds_reg_tip)
```

```
##       mae       mse      rmse      mape
## 0.4901647 1.4591417 1.2079494       Inf
```

5. Conclusions

6. References:

1. https://en.wikipedia.org/wiki/Taxicabs_of_the_United_States (https://en.wikipedia.org/wiki/Taxicabs_of_the_United_States) (last accessed August 30, 2018)
2. http://home.nyc.gov/html/tlc/html/industry/taxicab_rate_yellow.shtml (http://home.nyc.gov/html/tlc/html/industry/taxicab_rate_yellow.shtml)
3. www.nyc.gov (last accessed September 3, 2018)
4. ggmap - https://rpubs.com/jhofman/nycmaps (https://rpubs.com/jhofman/nycmaps) (last accessed August 31, 2018)