Traffic Predictions

Data 621 Final

Tommy Jenkins, Violeta Stoyanova, Todd Weigel, Peter Kowalchuk, Eleanor R-Secoquian, Anthony Pagan

2019/12/04

Utility functions

From R Cookbooks to create multi-panel plots.

OVERVIEW

In this competition, Kaggle is challenging you to build a model that predicts the total ride duration of taxi trips in New York City. Your primary dataset is one released by the NYC Taxi and Limousine Commission, which includes pickup time, geo-coordinates, number of passengers, and several other variables.

The competition dataset is based on the 2016 NYC Yellow Cab trip record data made available in Big Query on Google Cloud Platform. The data was originally published by the NYC Taxi and Limousine Commission (TLC). The data was sampled and cleaned for the purposes of this playground competition. Based on individual trip attributes, participants should predict the duration of each trip in the test set.

File descriptions

- train.csv - the training set (contains 1458644 trip records)
- test.csv - the testing set (contains 625134 trip records)
- sample_submission.csv - a sample submission file in the correct format https://www.kaggle.com/c/6960/download-all

Data fields

Variable Name	Definition
id	a unique identifier for each trip
vendor_id	a code indicating the provider associated with the trip record
pickup_datetime	date and time when the meter was engaged
dropoff_datetime	date and time when the meter was disengaged
passenger_count	the number of passengers in the vehicle (driver entered value)
pickup_longitude	the longitude where the meter was engaged
pickup_latitude	the latitude where the meter was engaged
dropoff_longitude	the longitude where the meter was disengaged
dropoff_latitude	the latitude where the meter was disengaged
store_and_fwd_flag	This flag indicates whether the trip record was held in vehicle memory before sending to the vendor because the vehicle did not have a connection to the server: Y=store and forward; N=not a store and forward trip
trip_duration	duration of the trip in seconds

Objective:

The purpose of this project is to build various models in an attempt to predict the trip duration of yellow taxis in New York City.

Using the techniques we've learned in the class, like classification, model diagnostics and transformation, we will explore data to find new patterns. And just like what is required in the Kaggle contest, we will try to predict the duration of each trip in the test set. We will build multiple linear regression modeling and then summary to interpret the results. We'll further analyze the results by adding discrimination in the model and then assess the discrimination with ROC curve.

vvvv REWRITE vvv

DATA EXPLORATION

Load data

View Data

```
## Observations: 1,458,644
## Variables: 11
## $ id                 <chr> "id2875421", "id2377394", "id3858529", "id350467...
## $ vendor_id          <int> 2, 1, 2, 2, 2, 2, 1, 2, 1, 2, 2, 2, 2, 2, 2, 1, ...
## $ pickup_datetime    <chr> "2016-03-14 17:24:55", "2016-06-12 00:43:35", "2...
## $ dropoff_datetime   <chr> "2016-03-14 17:32:30", "2016-06-12 00:54:38", "2...
## $ passenger_count    <int> 1, 1, 1, 1, 1, 6, 4, 1, 1, 1, 1, 4, 2, 1, 1, 1, ...
## $ pickup_longitude   <dbl> -73.98215, -73.98042, -73.97903, -74.01004, -73....
## $ pickup_latitude    <dbl> 40.76794, 40.73856, 40.76394, 40.71997, 40.79321...
## $ dropoff_longitude  <dbl> -73.96463, -73.99948, -74.00533, -74.01227, -73....
## $ dropoff_latitude   <dbl> 40.76560, 40.73115, 40.71009, 40.70672, 40.78252...
## $ store_and_fwd_flag <chr> "N", "N", "N", "N", "N", "N", "N", "N", "N", "N"...
## $ trip_duration      <int> 455, 663, 2124, 429, 435, 443, 341, 1551, 255, 1...
```

Data Summary

```
##  [1] "id"                 "vendor_id"          "pickup_datetime"
##  [4] "dropoff_datetime"   "passenger_count"    "pickup_longitude"
##  [7] "pickup_latitude"    "dropoff_longitude"  "dropoff_latitude"
## [10] "store_and_fwd_flag" "trip_duration"
```

```
## [1] "id"                 "vendor_id"          "pickup_datetime"
## [4] "passenger_count"    "pickup_longitude"   "pickup_latitude"
## [7] "dropoff_longitude"  "dropoff_latitude"   "store_and_fwd_flag"
```

```
## Observations: 2,083,778
## Variables: 12
## $ id                 <chr> "id2875421", "id2377394", "id3858529", "id350467...
## $ vendor_id          <int> 2, 1, 2, 2, 2, 2, 1, 2, 1, 2, 2, 2, 2, 2, 2, 1, ...
## $ pickup_datetime    <chr> "2016-03-14 17:24:55", "2016-06-12 00:43:35", "2...
## $ dropoff_datetime   <chr> "2016-03-14 17:32:30", "2016-06-12 00:54:38", "2...
## $ passenger_count    <int> 1, 1, 1, 1, 1, 6, 4, 1, 1, 1, 1, 4, 2, 1, 1, 1, ...
## $ pickup_longitude   <dbl> -73.98215, -73.98042, -73.97903, -74.01004, -73....
## $ pickup_latitude    <dbl> 40.76794, 40.73856, 40.76394, 40.71997, 40.79321...
## $ dropoff_longitude  <dbl> -73.96463, -73.99948, -74.00533, -74.01227, -73....
## $ dropoff_latitude   <dbl> 40.76560, 40.73115, 40.71009, 40.70672, 40.78252...
## $ store_and_fwd_flag <chr> "N", "N", "N", "N", "N", "N", "N", "N", "N", "N"...
## $ trip_duration      <int> 455, 663, 2124, 429, 435, 443, 341, 1551, 255, 1...
## $ dset               <fct> train, train, train, train, train, train, train,...
```

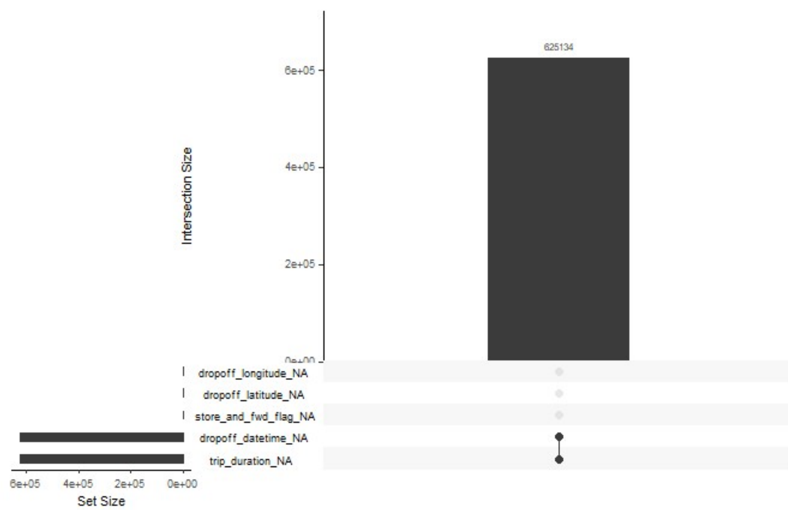
```
##       id              vendor_id     pickup_datetime    dropoff_datetime
##  Length:2083778     Min.   :1.000   Length:2083778     Length:2083778
##  Class :character   1st Qu.:1.000   Class :character   Class :character
##  Mode  :character   Median :2.000   Mode  :character   Mode  :character
##                     Mean   :1.535
##                     3rd Qu.:2.000
##                     Max.   :2.000
##
##  passenger_count pickup_longitude  pickup_latitude dropoff_longitude
##  Min.   :0.000   Min.   :-121.93   Min.   :34.36   Min.   :-121.93
##  1st Qu.:1.000   1st Qu.: -73.99   1st Qu.:40.74   1st Qu.: -73.99
##  Median :1.000   Median : -73.98   Median :40.75   Median : -73.98
##  Mean   :1.664   Mean   : -73.97   Mean   :40.75   Mean   : -73.97
##  3rd Qu.:2.000   3rd Qu.: -73.97   3rd Qu.:40.77   3rd Qu.: -73.96
##  Max.   :9.000   Max.   : -61.34   Max.   :51.88   Max.   : -61.34
##
##  dropoff_latitude store_and_fwd_flag trip_duration       dset
##  Min.   :32.18    Length:2083778     Min.   :      1   test : 625134
##  1st Qu.:40.74    Class :character   1st Qu.:    397   train:1458644
##  Median :40.75    Mode  :character   Median :    662
##  Mean   :40.75                       Mean   :    959
##  3rd Qu.:40.77                       3rd Qu.:   1075
##  Max.   :48.86                       Max.   :3526282
##                                      NA's   :625134
```

Missing values

```
##              columns na_count neg_count zero_count unique_count
## 1                 id        0         0         NA      2083778
```

```
## 2           vendor_id        0        0          0           2
## 3     pickup_datetime        0        0         NA     1926217
## 4    dropoff_datetime   625134       NA         NA     1380378
## 5     passenger_count        0        0         83          10
## 6    pickup_longitude        0  2083778          0       24960
## 7     pickup_latitude        0        0          0       48068
## 8   dropoff_longitude        0  2083778          0       36977
## 9    dropoff_latitude        0        0          0       67086
## 10 store_and_fwd_flag        0        0         NA           2
## 11      trip_duration   625134       NA         NA        7418
## 12               dset        0       NA          0           2
```

```
##                columns na_count neg_count zero_count unique_count
## 1                   id        0         0         NA      2083778
## 2            vendor_id        0         0          0            2
## 3      pickup_datetime        0         0         NA      1926217
## 4     dropoff_datetime   625134        NA         NA      1380378
## 5      passenger_count        0         0         83           10
## 6     pickup_longitude        0   2083778          0        24960
## 7      pickup_latitude        0         0          0        48068
## 8    dropoff_longitude        0   2083778          0        36977
## 9     dropoff_latitude        0         0          0        67086
## 10  store_and_fwd_flag        0         0         NA            2
## 11       trip_duration   625134        NA         NA         7418
## 12                dset        0        NA          0            2
```




```
##    Mode   FALSE    TRUE
## logical  625134 1458644
```

#Data Preparation

Reformating features

For our following analysis, we will turn the data and time from characters into *date* objects. We also recode *vendor_id* as a factor. This makes it easier to visualise relationships that involve these features.

Visualizations

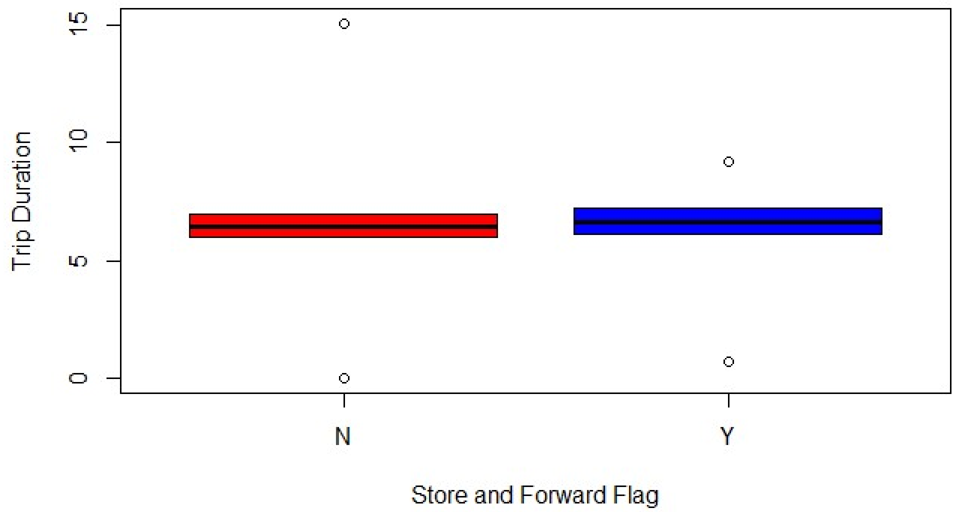


Fig. 2

```
## train$store_and_fwd_flag: N
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##   0.000   5.984   6.495   6.464   6.979  15.076 
## ------------------------------------------------------------ 
## train$store_and_fwd_flag: Y
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
##  0.6931  6.1203  6.6995  6.6274  7.2442  9.2087
```

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

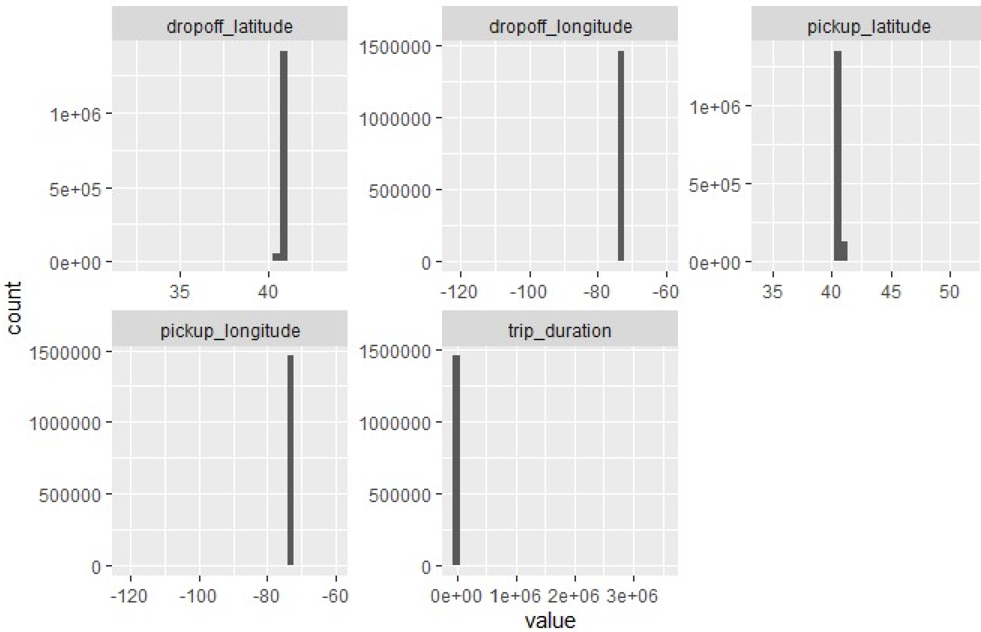
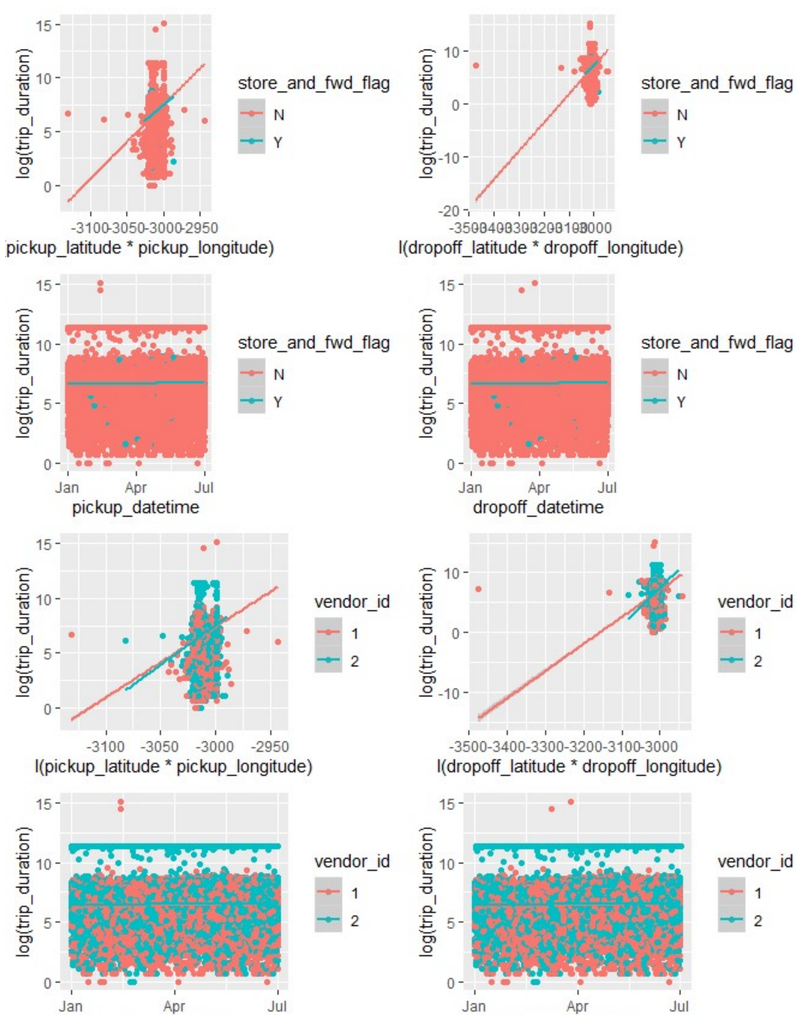
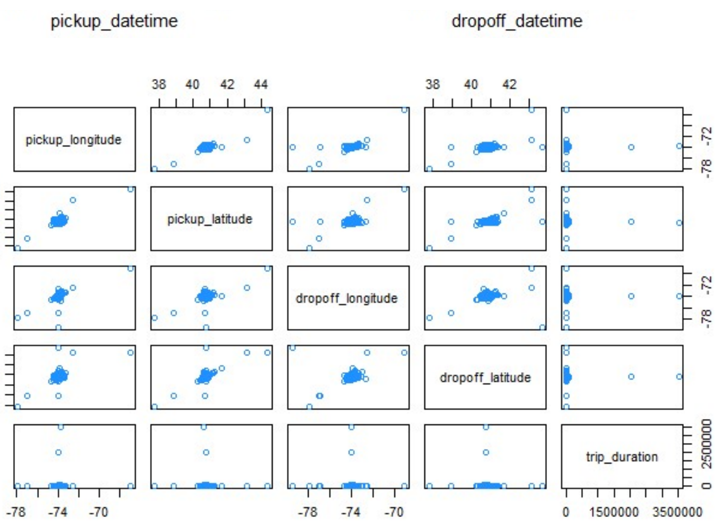


Fig. 2

Trip duration vs pickup and dropoff datetime and location using 20% of sampled data





More visualizations

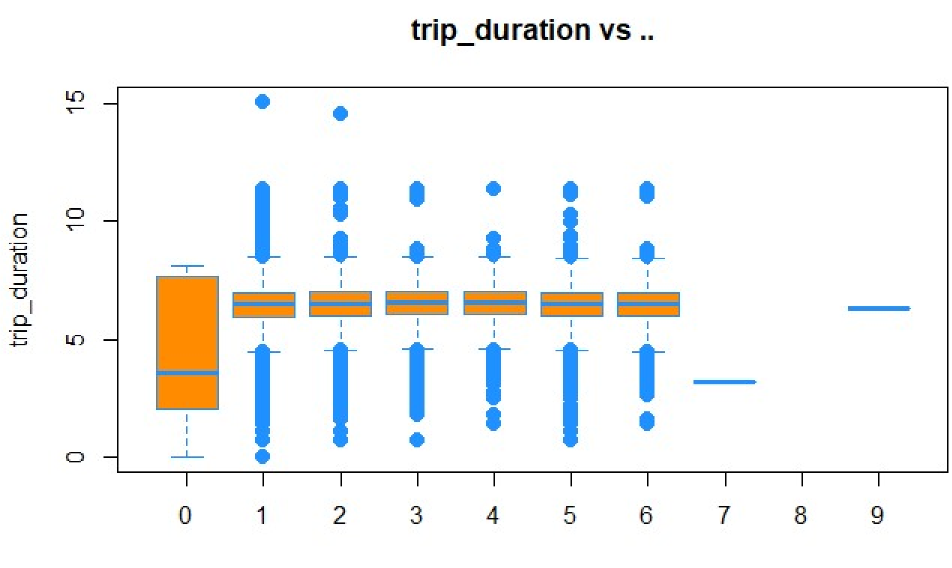


Fig. 2

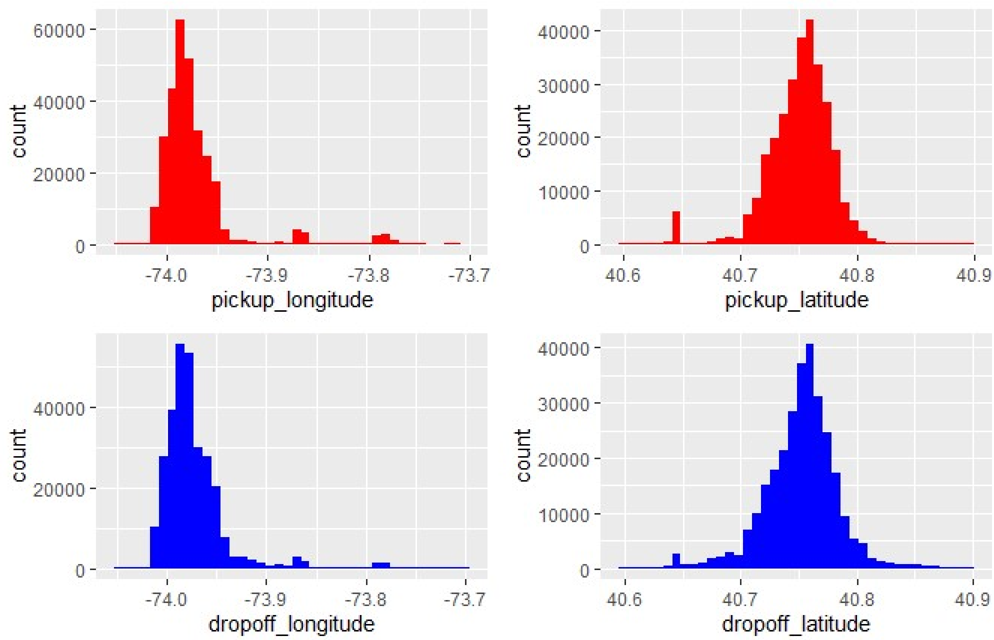


Fig. 6

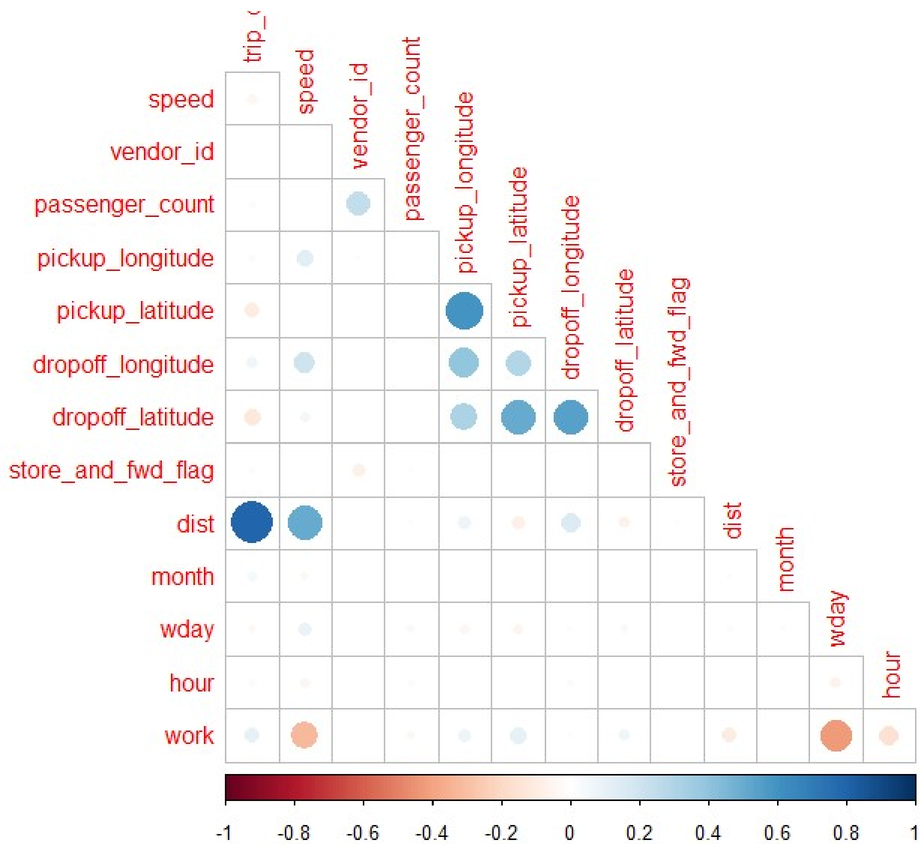


Fig. 30a

An excursion into classification

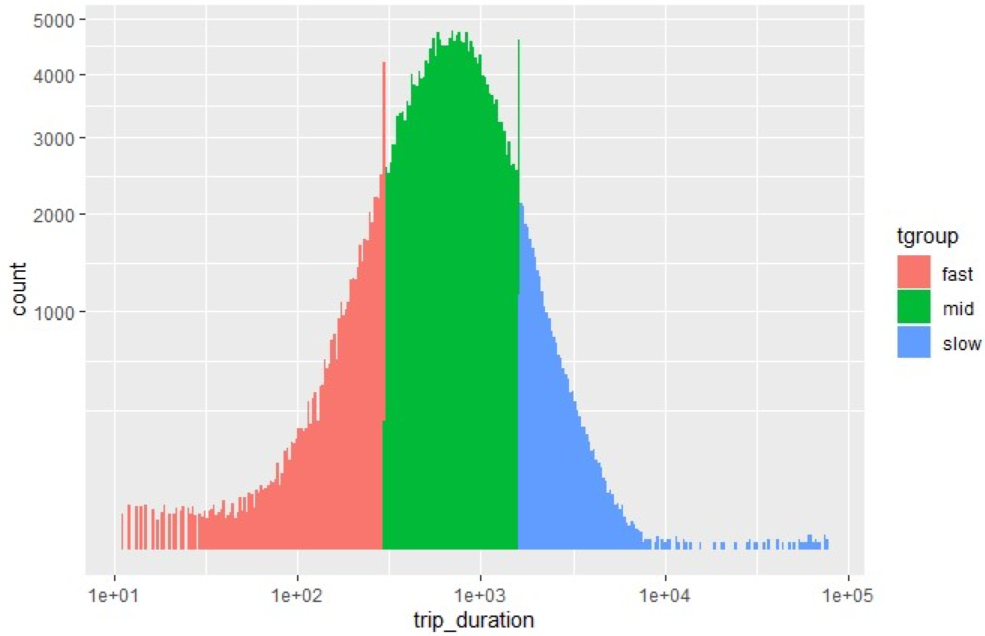


Fig. 31

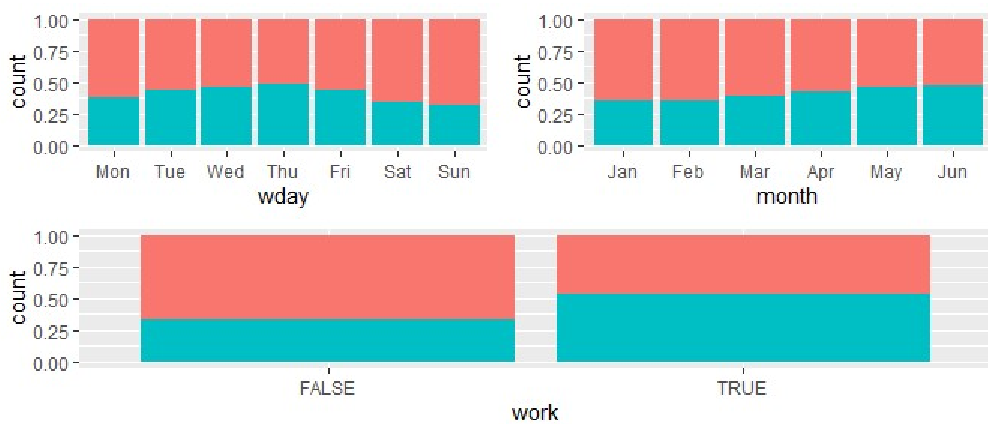


Fig. 32

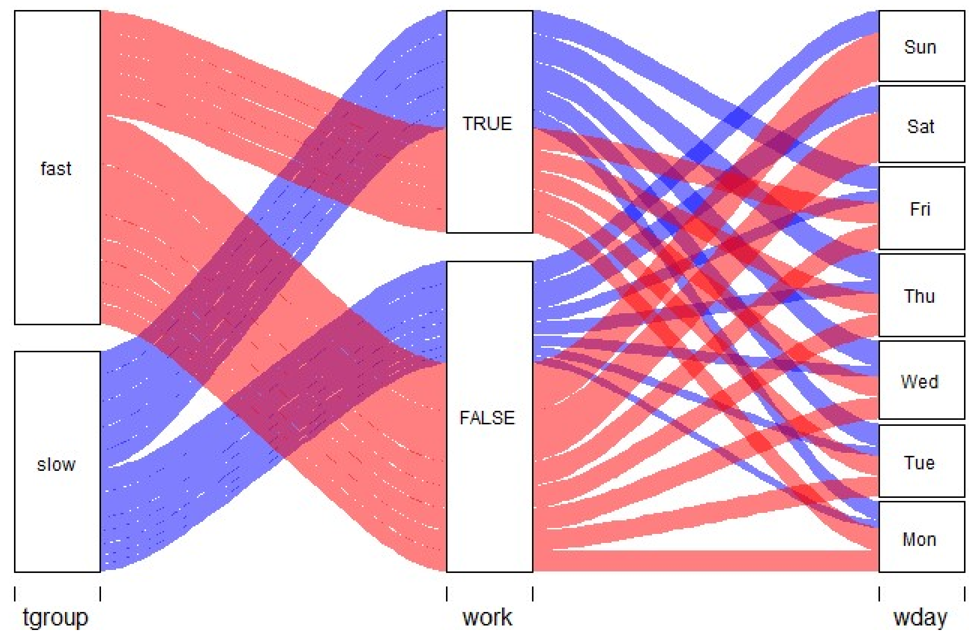


Fig. 34

Model, correlation

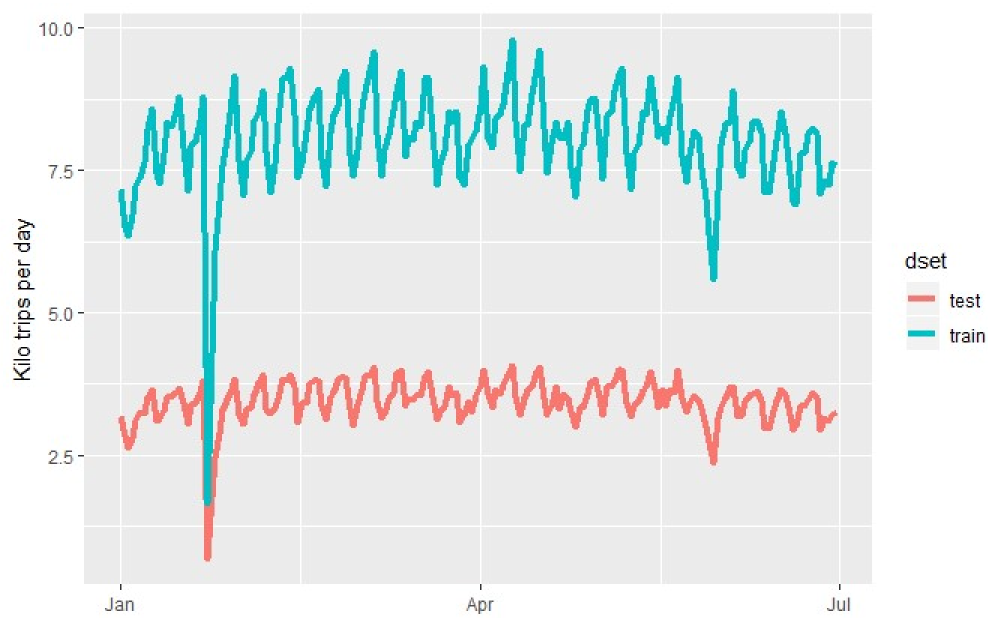


Fig. 35

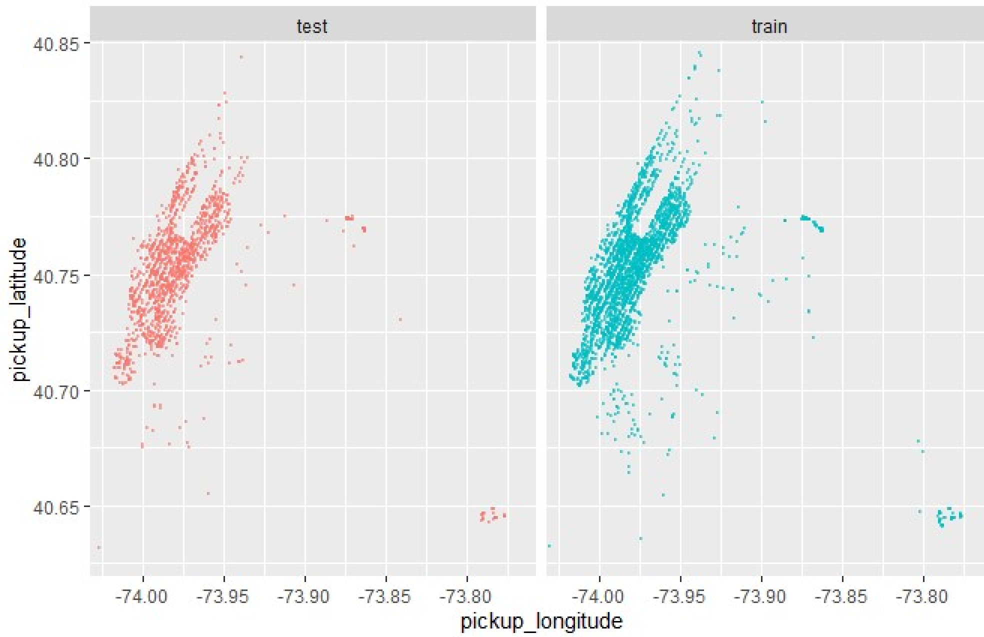


Fig. 36

Modeling

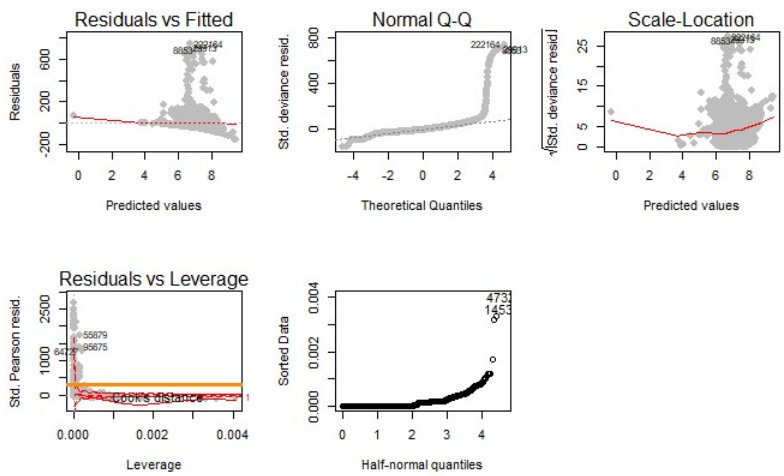
Sample Model 1 Poisson (No store_and_fwd_flag, id or vendor_id)

```
##
## Call:
## glm(formula = trip_duration ~ pickup_datetime:dropoff_datetime +
##     pickup_latitude:pickup_longitude + dropoff_latitude:dropoff_longitude,
##     family = poisson)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -156.81   -15.43    -4.92     8.35   740.39
##
## Coefficients:
##                                     Estimate Std. Error z value Pr(>|z|)
## (Intercept)                        3.149e+02  6.351e-02  4957.2   <2e-16 ***
## pickup_datetime:dropoff_datetime   3.183e-18  4.902e-21   649.3   <2e-16 ***
## pickup_latitude:pickup_longitude   6.069e-02  1.666e-05  3643.4   <2e-16 ***
## dropoff_latitude:dropoff_longitude 4.378e-02  2.015e-05  2172.3   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 136608788  on 290041  degrees of freedom
## Residual deviance: 114317748  on 290038  degrees of freedom
## AIC: 116727043
##
## Number of Fisher Scoring iterations: 5
```

```
## # A tibble: 4 x 5
##   term                                estimate std.error statistic p.value
##   <chr>                                  <dbl>     <dbl>     <dbl>   <dbl>
```

```
## 1 (Intercept)                          3.15e+ 2  6.35e- 2     4957.       0
## 2 pickup_datetime:dropoff_datetime    3.18e-18  4.90e-21      649.       0
## 3 pickup_latitude:pickup_longitude    6.07e- 2  1.67e- 5     3643.       0
## 4 dropoff_latitude:dropoff_longitude 4.38e- 2  2.02e- 5     2172.       0
```

```
## # A tibble: 1 x 7
##   null.deviance df.null     logLik        AIC        BIC   deviance df.residual
##           <dbl>   <int>      <dbl>      <dbl>      <dbl>      <dbl>       <int>
## 1    136608788.  290041 -58363517. 116727043. 116727085. 114317748.      290038
```



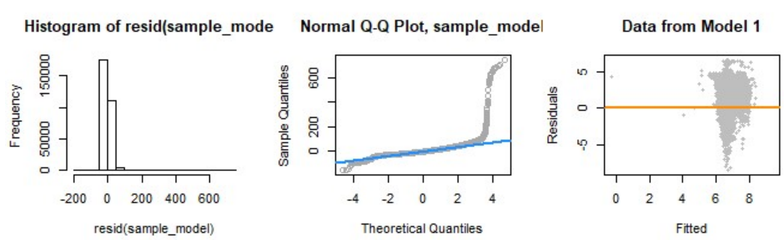
Breusch-Pagan Test.

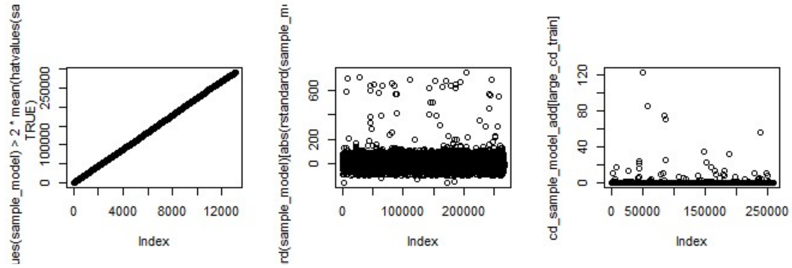
Cooks Distance H0: Homoscedasticity. The errors have constant variance about the true model. H1: Heteroscedasticity. The errors have non-constant variance about the true model. Leverage, Outliers, Influence, coef Change

```
##   11
## TRUE
```

```
##
##  studentized Breusch-Pagan test
##
## data:  sample_model
## BP = 123.87, df = 3, p-value < 2.2e-16
```

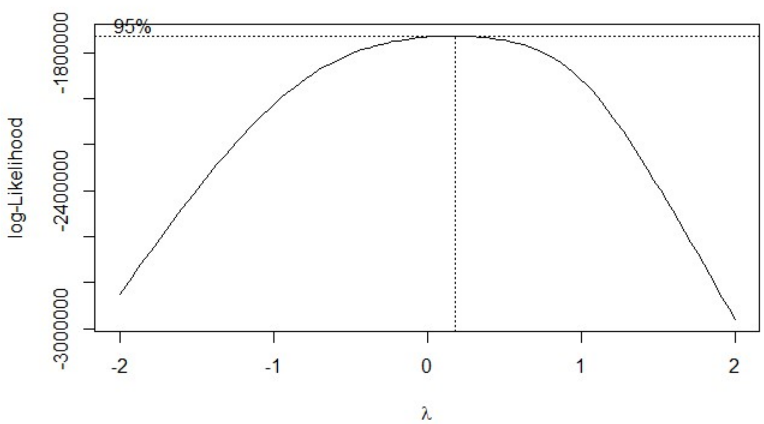
```
## [1] 257881
```






```
##                          (Intercept)   pickup_datetime:dropoff_datetime
##                         3.148518e+02                       3.182603e-18
##   pickup_latitude:pickup_longitude dropoff_latitude:dropoff_longitude
##                         6.069042e-02                       4.377879e-02
```

```
##       (Intercept) dropoff_longitude
##        182176.241          2449.298
```



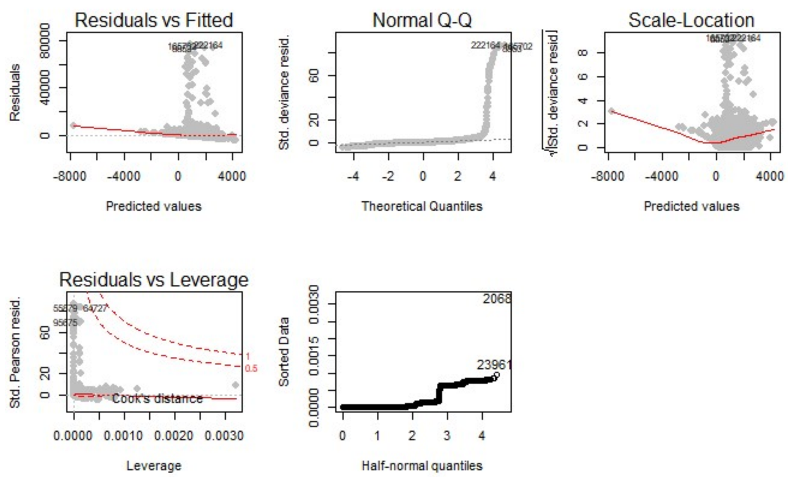
Sample Model 2 Gaussian(No id or vendor_id)

```
## 
## Call:
## glm(formula = trip_duration ~ pickup_datetime:dropoff_datetime + 
##     pickup_latitude:pickup_longitude + dropoff_latitude:dropoff_longitude + 
##     store_and_fwd_flag, family = gaussian)
## 
## Deviance Residuals: 
##    Min      1Q  Median      3Q     Max  
##  -4148    -392    -127     251   76335  
## 
## Coefficients:
##                                     Estimate Std. Error t value Pr(>|t|)    
## (Intercept)                        3.795e+05  2.263e+03 167.699  < 2e-16 ***
## store_and_fwd_flagY                1.810e+02  2.231e+01   8.114 4.93e-16 ***
## pickup_datetime:dropoff_datetime   2.710e-15  1.257e-16  21.560  < 2e-16 ***
## pickup_latitude:pickup_longitude   8.126e+01  6.250e-01 130.025  < 2e-16 ***
## dropoff_latitude:dropoff_longitude 4.628e+01  6.545e-01  70.702  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
```

```
## (Dispersion parameter for gaussian family taken to be 774496.8)
##
##     Null deviance: 2.4925e+11  on 290041  degrees of freedom
## Residual deviance: 2.2463e+11  on 290037  degrees of freedom
## AIC: 4756071
##
## Number of Fisher Scoring iterations: 2
```

```
## # A tibble: 5 x 5
##   term                              estimate std.error statistic   p.value
##   <chr>                                <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)                       3.80e+ 5  2.26e+ 3    168.   0.
## 2 store_and_fwd_flagY               1.81e+ 2  2.23e+ 1      8.11 4.93e- 16
## 3 pickup_datetime:dropoff_datetime  2.71e-15  1.26e-16     21.6  5.16e-103
## 4 pickup_latitude:pickup_longitude  8.13e+ 1  6.25e- 1    130.   0.
## 5 dropoff_latitude:dropoff_longitude 4.63e+ 1  6.55e- 1     70.7  0.
```

```
## # A tibble: 1 x 7
##   null.deviance df.null   logLik     AIC     BIC      deviance df.residual
##           <dbl>   <int>    <dbl>   <dbl>   <dbl>         <dbl>       <int>
## 1 249251879377.  290041 -2378030. 4756071. 4756134. 224632723577.      290037
```



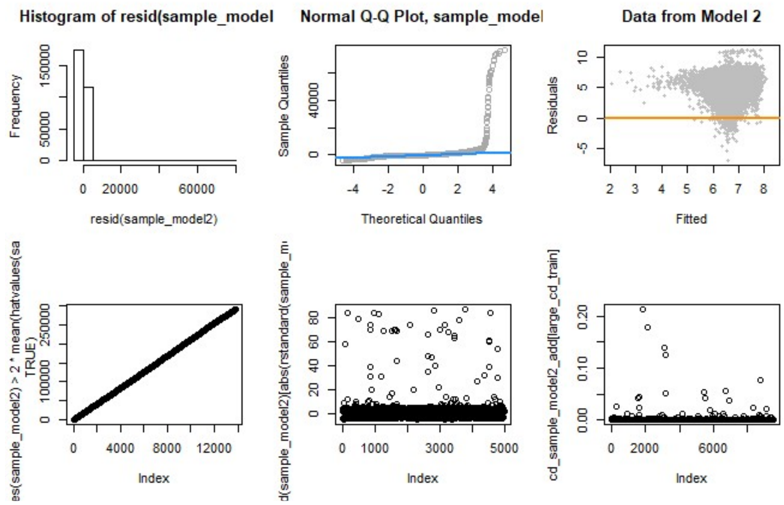
Breusch-Pagan Test.

Cooks Distance H0: Homoscedasticity. The errors have constant variance about the true model. H1: Heteroscedasticity. The errors have non-constant variance about the true model. Leverage, Outliers, Influence, coef Change

```
##    11
## FALSE
```

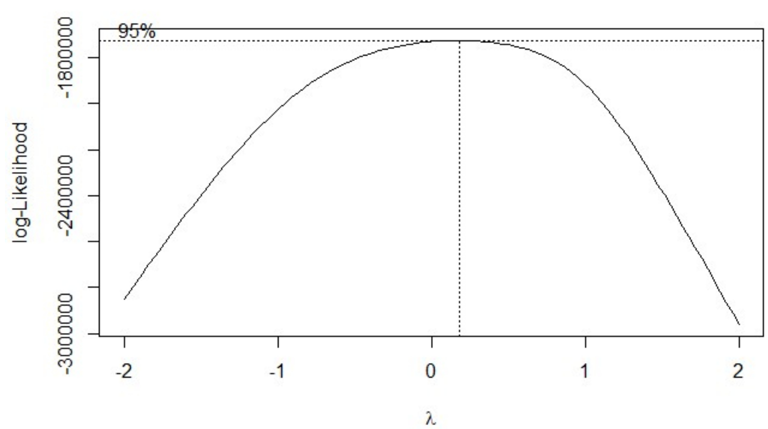
```
##
##  studentized Breusch-Pagan test
##
## data:  sample_model2
## BP = 123.86, df = 4, p-value < 2.2e-16
```

```
## [1] 9498
```




```
##                           (Intercept)                    store_and_fwd_flagY
##                          3.795352e+05                           1.810469e+02
##    pickup_datetime:dropoff_datetime   pickup_latitude:pickup_longitude
##                          2.709699e-15                           8.126130e+01
## dropoff_latitude:dropoff_longitude
##                          4.627524e+01
```

```
##       (Intercept) dropoff_longitude
##         79730.086          1064.841
```



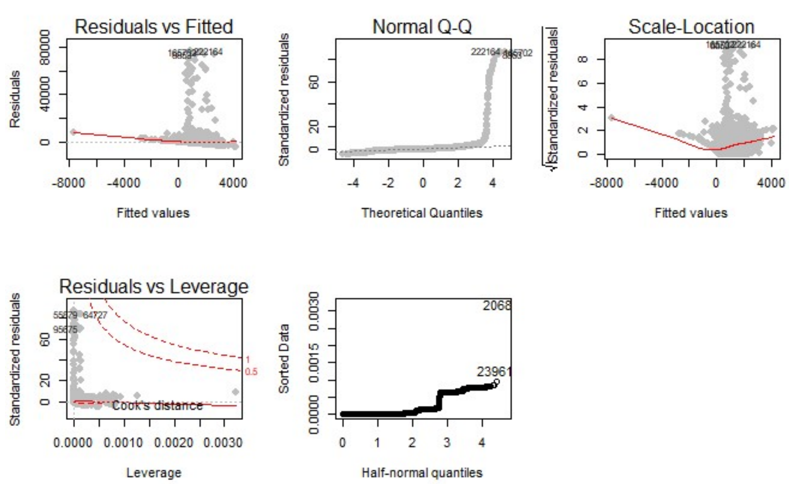
Sample Model 3 Negative Binomial (No id)

```
##
## Call:
## lm(formula = trip_duration ~ pickup_datetime:dropoff_datetime +
##     pickup_latitude:pickup_longitude + dropoff_latitude:dropoff_longitude +
##     store_and_fwd_flag + vendor_id, family = negative.binomial(1))
```

```
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -4133   -392   -127    251  76323
##
## Coefficients:
##                                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)                        3.794e+05  2.263e+03 167.672  < 2e-16 ***
## store_and_fwd_flagY                1.952e+02  2.238e+01   8.722  < 2e-16 ***
## vendor_id2                         2.629e+01  3.286e+00   7.998 1.27e-15 ***
## pickup_datetime:dropoff_datetime   2.717e-15  1.257e-16  21.619  < 2e-16 ***
## pickup_latitude:pickup_longitude   8.122e+01  6.249e-01 129.971  < 2e-16 ***
## dropoff_latitude:dropoff_longitude 4.629e+01  6.544e-01  70.735  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 880 on 290036 degrees of freedom
## Multiple R-squared:  0.09897,    Adjusted R-squared:  0.09896
## F-statistic:  6372 on 5 and 290036 DF,  p-value: < 2.2e-16
```

```
## # A tibble: 6 x 5
##   term                               estimate std.error statistic   p.value
##   <chr>                                 <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)                        3.79e+ 5  2.26e+ 3    168.   0.
## 2 store_and_fwd_flagY                1.95e+ 2  2.24e+ 1      8.72 2.75e- 18
## 3 vendor_id2                         2.63e+ 1  3.29e+ 0      8.00 1.27e- 15
## 4 pickup_datetime:dropoff_datetime   2.72e-15  1.26e-16     21.6  1.45e-103
## 5 pickup_latitude:pickup_longitude   8.12e+ 1  6.25e- 1    130.   0.
## 6 dropoff_latitude:dropoff_longitude 4.63e+ 1  6.54e- 1     70.7  0.
```

```
## # A tibble: 1 x 11
##   r.squared adj.r.squared sigma statistic p.value    df  logLik    AIC    BIC
##       <dbl>         <dbl> <dbl>     <dbl>   <dbl> <int>   <dbl>  <dbl>  <dbl>
## 1    0.0990        0.0990  880.     6372.       0     6 -2.38e6 4.76e6 4.76e6
## # ... with 2 more variables: deviance <dbl>, df.residual <int>
```



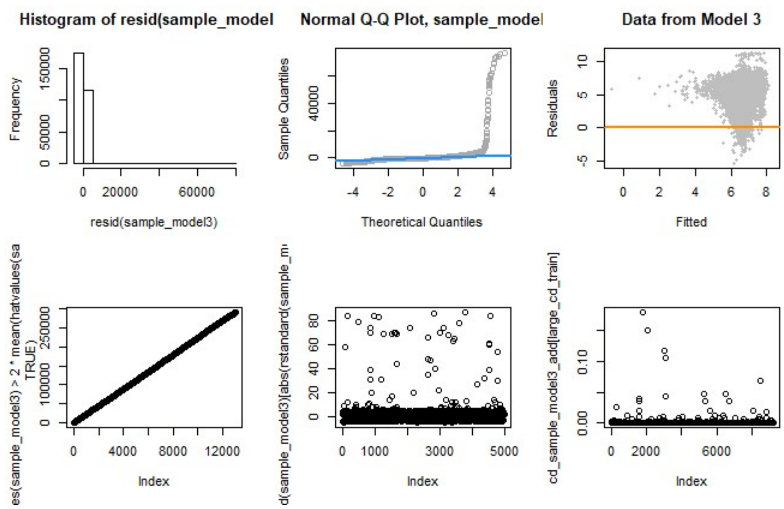
Breusch-Pagan Test.

Cooks Distance H0: Homoscedasticity. The errors have constant variance about the true model. H1: Heteroscedasticity. The errors have non-constant variance about the true model. Leverage, Outliers, Influence, coef Change

```
##    11
## FALSE
```

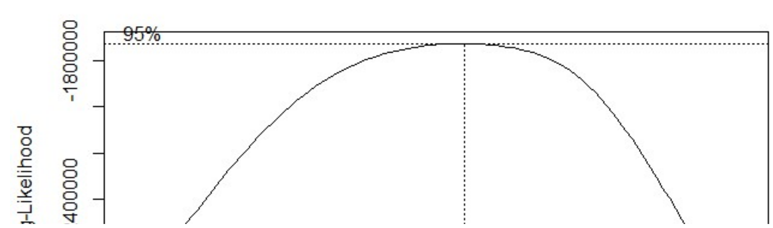
```
##
##  studentized Breusch-Pagan test
##
## data:  sample_model3
## BP = 150.33, df = 5, p-value < 2.2e-16
```

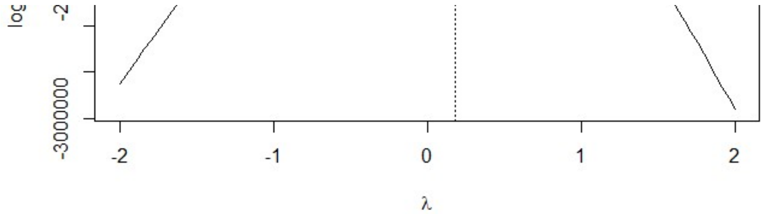
```
## [1] 9185
```




```
##                         (Intercept)                 store_and_fwd_flagY
##                        3.794375e+05                        1.952059e+02
##                          vendor_id2    pickup_datetime:dropoff_datetime
##                        2.628544e+01                        2.716875e-15
##   pickup_latitude:pickup_longitude dropoff_latitude:dropoff_longitude
##                        8.122143e+01                        4.629244e+01
```

```
##       (Intercept) dropoff_longitude
##          79811.38           1065.94
```



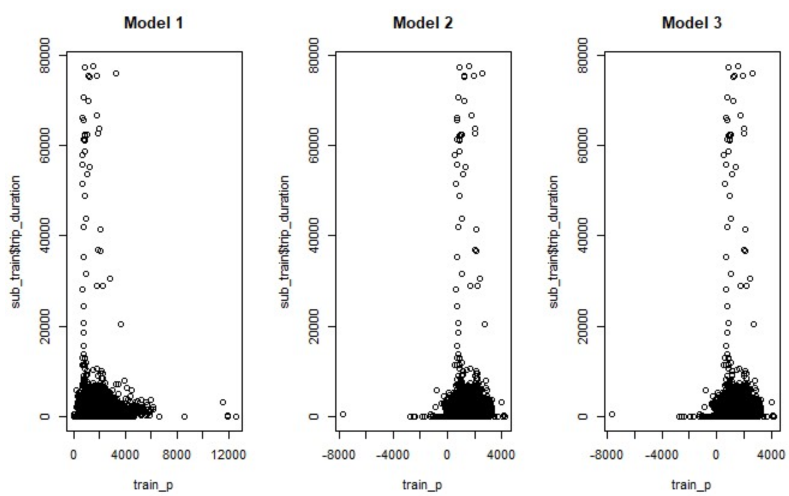


	Model 1	Model 2	Model 3
AIC	116727042.65499	4756071.0066072	4756009.03978159
BIC	116727084.966114	4756134.47329331	4756083.08424872

CONCLUSION

With 3 models computed, we select the model with the lowest combination of AIC and BIC. From the table, we can see the model to pick is model 1

Model 1 showed the best result. We can observe its performance by plotting the datasets Vendor_ID values against the predicted values.



APPENDIX

Code used in analysis

```
#list.of.packages <-
  c("alluvial","caret","caret","corrplot","corrplot","data.table","dplyr","faraway","forcats","geosph#ere","gg
#new.packages <- list.of.packages[!(list.of.packages %in% installed.packages()
  [,"Package"])]
#if(length(new.packages)) install.packages(new.packages)
require(knitr)
knitr::opts_chunk$set(echo = FALSE, warning = FALSE,fig.align='center')

library(faraway)
library(MASS)
library(psych)
library(pROC)
library(corrplot)
library(jtools)
library(mice)
library('corrr')
```

```r
library(kableExtra)
library(gridExtra)
library(pander)
library(zoo)
library(lmtest)
library(corrr)
library(broom)

#devtools::install_github("thomasp85/patchwork")
library(patchwork)
library(tidyverse)
library(ggplot2)
library(ggplot2)
library(reshape2)
library(leaps)
library(caret)
library(naniar)
library('ggplot2') # visualisation
library('scales') # visualisation
library('grid') # visualisation
library('RColorBrewer') # visualisation
library('corrplot') # visualisation
library('alluvial') # visualisation
library('dplyr') # data manipulation
library('readr') # input/output
library('data.table') # data manipulation
library('tibble') # data wrangling
library('tidyr') # data wrangling
library('stringr') # string manipulation
library('forcats') # factor manipulation
library('lubridate') # date and time
library('geosphere') # geospatial locations
library('leaflet') # maps
library('leaflet.extras') # maps
library('maps') # maps
library('xgboost') # modelling
library('caret') # modelling
library('widgetframe') #visualizaiton
library('grid')
library('gridExtra')
# Define multiple plot function
#
# ggplot objects can be passed in ..., or to plotlist (as a list of ggplot objects)
# - cols:   Number of columns in layout
# - layout: A matrix specifying the layout. If present, 'cols' is ignored.
#
# If the layout is something like matrix(c(1,2,3,3), nrow=2, byrow=TRUE),
# then plot 1 will go in the upper left, 2 will go in the upper right, and
# 3 will go all the way across the bottom.
#

multiplot <- function(..., plotlist=NULL, file, cols=1, layout=NULL) {

  # Make a list from the ... arguments and plotlist
  plots <- c(list(...), plotlist)

  numPlots = length(plots)

  # If layout is NULL, then use 'cols' to determine layout
  if (is.null(layout)) {
    # Make the panel
    # ncol: Number of columns of plots
    # nrow: Number of rows needed, calculated from # of cols
```

```
    layout <- matrix(seq(1, cols * ceiling(numPlots/cols)),
                    ncol = cols, nrow = ceiling(numPlots/cols))
  }

 if (numPlots==1) {
    print(plots[[1]])
  } else {
    # Set up the page
    grid.newpage()
    pushViewport(viewport(layout = grid.layout(nrow(layout), ncol(layout))))

    # Make each plot, in the correct location
    for (i in 1:numPlots) {
      # Get the i,j matrix positions of the regions that contain this subplot
      matchidx <- as.data.frame(which(layout == i, arr.ind = TRUE))

      print(plots[[i]], vp = viewport(layout.pos.row = matchidx$row,
                                      layout.pos.col = matchidx$col))
    }
  }
}
var_stats<- function(df){
  wt<-data.frame(columns=colnames(df))
  wt$na_count <- sapply(df, function(y) sum(is.na(y)))
  wt$neg_count <- sapply(df, function(y) sum(y<0))
  wt$zero_count <- sapply(df, function(y) sum(as.integer(y)==0))
  wt$unique_count <- sapply(df, function(y) sum(n_distinct(y)))
  print(wt)
  return(wt)
}
rows <-
  c("id","vendor_id","pickup_datetime","dropoff_datetime","passenger_count","pickup_longitude","pickup_latitud
"dropoff_longitude","dropoff_latitude","store_and_fwd_flag","trip_duration")

def <- c("a unique identifier for each trip",
"a code indicating the provider associated with the trip record",
"date and time when the meter was engaged",
"date and time when the meter was disengaged",
"the number of passengers in the vehicle (driver entered value)",
"the longitude where the meter was engaged",
"the latitude where the meter was engaged",
"the longitude where the meter was disengaged",
"the latitude where the meter was disengaged",
"This flag indicates whether the trip record was held in vehicle memory before sending to
  the vendor because the vehicle did not have a connection to the server: Y=store and
  forward; N=not a store and forward trip",
"duration of the trip in seconds")

kable(cbind(rows, def), col.names = c("Variable Name", "Definition")) %>% kable_styling()
train <- as_tibble(fread('data/train.csv'))
test <- as_tibble(fread('data/test.csv'))
sample_submit <- as_tibble(fread('data/sample_submission.csv'))
#str(train)
glimpse(train)
#summary(train)
#describe(train)
names(train)
names(test)
#glimpse(test)

#
vars_to_add <- train[!names(train) %in% names(test)]
```

```r
 #vvvvv
## Combining train and test

combine <- rbind(train %>% mutate(dset = "train"),
                     test %>% mutate(dset = "test",
                                        dropoff_datetime = NA,
                                        trip_duration = NA))
combine <- combine %>% mutate(dset = factor(dset))
glimpse(combine)
summary(combine)
var_stats(combine)
gg_miss_upset(combine)
summary(complete.cases(combine))
train <- train %>%
  mutate(pickup_datetime = ymd_hms(pickup_datetime),
         dropoff_datetime = ymd_hms(dropoff_datetime),
         vendor_id = factor(vendor_id),
         passenger_count = factor(passenger_count))
#ggplot(combine, aes(trip_duration)) +
#  geom__histogram(aes(y = ..density..)

attach(train)
boxplot(by(log(train$trip_duration),train$store_and_fwd_flag,summary),col=c("red","blue"),xlab="Store
  and Forward Flag", ylab="Trip Duration")
by(log(train$trip_duration),train$store_and_fwd_flag,summary)

#plot(trip_duration ~ dropoff_longitude,pch  = 20,cex  = 2,col  = "grey")

train[sapply(train, function(x) is.numeric(x) && !is.na(x))] %>%
  gather() %>%
  ggplot(aes(value), main="") +
  facet_wrap(~ key, scales = "free") +
  geom_histogram()

sub_train = train%>%sample_frac(.2)
attach(sub_train)
g1<-ggplot(sub_train, aes(x=I(pickup_latitude*pickup_longitude), y=log(trip_duration),
  color = store_and_fwd_flag)) +geom_point() +stat_smooth(method="glm", se=TRUE)
g2<-ggplot(sub_train, aes(x=I(dropoff_latitude*dropoff_longitude), y=log(trip_duration),
  color = store_and_fwd_flag)) +geom_point() +stat_smooth(method="glm", se=TRUE)
g3<-ggplot(sub_train, aes(x=pickup_datetime, y=log(trip_duration), color =
  store_and_fwd_flag)) +geom_point() +stat_smooth(method="glm", se=TRUE)
g4<-ggplot(sub_train, aes(x=dropoff_datetime, y=log(trip_duration), color =
  store_and_fwd_flag)) +geom_point() +stat_smooth(method="glm", se=TRUE)
grid.arrange(g1, g2, g3, g4, ncol = 2)

g1<-ggplot(sub_train, aes(x=I(pickup_latitude*pickup_longitude), y=log(trip_duration),
  color = vendor_id)) +geom_point() +stat_smooth(method="glm", se=TRUE)
g2<-ggplot(sub_train, aes(x=I(dropoff_latitude*dropoff_longitude), y=log(trip_duration),
  color = vendor_id)) +geom_point() +stat_smooth(method="glm", se=TRUE)
g3<-ggplot(sub_train, aes(x=pickup_datetime, y=log(trip_duration), color = vendor_id))
  +geom_point() +stat_smooth(method="glm", se=TRUE)
g4<-ggplot(sub_train, aes(x=dropoff_datetime, y=log(trip_duration), color = vendor_id))
  +geom_point() +stat_smooth(method="glm", se=TRUE)
grid.arrange(g1, g2, g3, g4, ncol = 2)


pairs(sub_train[sapply(sub_train, function(x) is.numeric(x))], col = "dodgerblue")


ssub_train<-sub_train[sapply(sub_train, function(x) is.numeric(x) && !is.na(x))]

ssub_train %>%
  correlate() %>%
  network_plot(min_cor = .2)
```

```r
#log(sub_train$trip_duration) %>% as.double() %>% boxplot()
#bins
#scale_x_log10() +
#scale_y_sqrt()
attach(sub_train)
boxplot(log(trip_duration) ~ as.factor(passenger_count),
     xlab   = "..",
     ylab   = "trip_duration",
     main   = "trip_duration vs ..",
     pch    = 20,
     cex    = 2,
     col    = "darkorange",
     border = "dodgerblue")

p1 <- sub_train %>%
  filter(pickup_longitude > -74.05 & pickup_longitude < -73.7) %>%
  ggplot(aes(pickup_longitude)) +
  geom_histogram(fill = "red", bins = 40)

p2 <- sub_train %>%
  filter(dropoff_longitude > -74.05 & dropoff_longitude < -73.7) %>%
  ggplot(aes(dropoff_longitude)) +
  geom_histogram(fill = "blue", bins = 40)

p3 <- sub_train %>%
  filter(pickup_latitude > 40.6 & pickup_latitude < 40.9) %>%
  ggplot(aes(pickup_latitude)) +
  geom_histogram(fill = "red", bins = 40)

p4 <- sub_train %>%
  filter(dropoff_latitude > 40.6 & dropoff_latitude < 40.9) %>%
  ggplot(aes(dropoff_latitude)) +
  geom_histogram(fill = "blue", bins = 40)

layout <- matrix(c(1,2,3,4),2,2,byrow=FALSE)
multiplot(p1, p2, p3, p4, layout=layout)
p1 <- 1; p2 <- 1; p3 <- 1; p4 <- 1
#jfk_coord <- tibble(lon = -73.778889, lat = 40.639722)

#la_guardia_coord <- tibble(lon = -73.872611, lat = 40.77725)


#train$jfk_dist_pick <- distCosine(pick_coord, jfk_coord)

#train$jfk_dist_drop <- distCosine(drop_coord, jfk_coord)

#train$lg_dist_pick <- distCosine(pick_coord, la_guardia_coord)

#train$lg_dist_drop <- distCosine(drop_coord, la_guardia_coord)



pick_coord <- sub_train %>% select(pickup_longitude, pickup_latitude)

drop_coord <- sub_train %>% select(dropoff_longitude, dropoff_latitude)

sub_train$dist <- distCosine(pick_coord, drop_coord)

#train$bearing = bearing(pick_coord, drop_coord)
```

```r
sub_train <- sub_train %>%
  mutate(speed = dist/trip_duration*3.6,
         date = date(pickup_datetime),
         month = month(pickup_datetime, label = TRUE),
         wday = wday(pickup_datetime, label = TRUE),
         wday = fct_relevel(wday, c("Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun")),
         hour = hour(pickup_datetime),
         work = (hour %in% seq(8,18)) & (wday %in% c("Mon","Tue","Wed","Thu","Fri")),
 #        jfk_trip = (jfk_dist_pick < 2e3) | (jfk_dist_drop < 2e3),
 #       lg_trip = (lg_dist_pick < 2e3) | (lg_dist_drop < 2e3),
 #        blizzard = !( (date < ymd("2016-01-22") | (date > ymd("2016-01-29"))) )
         )

sub_train <- sub_train %>%
  filter(trip_duration < 22*3600,
         dist > 0 | (near(dist, 0) & trip_duration < 60),
#         jfk_dist_pick < 3e5 & jfk_dist_drop < 3e5,
         trip_duration > 10,
         speed < 100)

sub_train %>%
  select(-id, -pickup_datetime, -dropoff_datetime, -date) %>% #-jfk_dist_pick,
 #        -jfk_dist_drop, -lg_dist_pick, -lg_dist_drop, -date) %>%
  mutate(passenger_count = as.integer(passenger_count),
         vendor_id = as.integer(vendor_id),
         store_and_fwd_flag = as.integer(as.factor(store_and_fwd_flag)),
 #        jfk_trip = as.integer(jfk_trip),
         wday = as.integer(wday),
         month = as.integer(month),
         work = as.integer(work))%>%
 #        lg_trip = as.integer(lg_trip),
```

```
 #        blizzard = as.integer(blizzard),

 #        has_snow = as.integer(has_snow),

  #       has_rain = as.integer(has_rain)) %>%
#
  select(trip_duration, speed, everything()) %>%

  cor(use="complete.obs", method = "spearman") %>%

  corrplot(type="lower", method="circle", diag=FALSE)


train_group <- sub_train %>%

  mutate(tgroup = case_when(trip_duration < 3e2 ~ "fast",

                            trip_duration >= 3e2 & trip_duration <= 1.6e3 ~ "mid",

                            trip_duration > 1.6e3 ~ "slow"))



train_group %>%

  ggplot(aes(trip_duration, fill = tgroup)) +

  geom_histogram(bins = 300) +

  scale_x_log10() +

  scale_y_sqrt()


train_group <- train_group %>%

  filter(tgroup != "mid")



p1 <- train_group %>%

  ggplot(aes(wday, fill = tgroup)) +

  geom_bar(position = "fill") +

  theme(legend.position = "none")



p2 <- train_group %>%

  ggplot(aes(month, fill = tgroup)) +

  geom_bar(position = "fill") +

  theme(legend.position = "none")
```

```
p3 <- train_group %>%
  ggplot(aes(hour, fill = tgroup)) +
  geom_bar(position = "fill")
p7 <- train_group %>%
  ggplot(aes(work, fill = tgroup)) +
  geom_bar(position = "fill") +
  theme(legend.position = "none")

layout <- matrix(c(1,1,2,2,3,3,3,3,4,5,6,7),3,4,byrow=TRUE)
multiplot(p1, p2,   p7, layout=layout)
p1 <- 1; p2 <- 1;   p7 <- 1

allu_train <- train_group %>%
  group_by(tgroup, work, wday) %>% # jfk_trip
  count() %>%
  ungroup


alluvial(allu_train %>% select(-n),
         freq=allu_train$n, border=NA,
         col=ifelse(allu_train$tgroup == "fast", "red", "blue"),
         cex=0.75,
         hide = allu_train$n < 150,
         ordering = list(
           order(allu_train$tgroup=="fast"),
       #    NULL,
           NULL,
           NULL))

foo <- combine %>%
  mutate(date = date(ymd_hms(pickup_datetime))) %>%
  group_by(date, dset) %>%
  count() %>%
```

```r
  ungroup()

foo %>%

  ggplot(aes(date, n/1e3, color = dset)) +

  geom_line(size = 1.5) +

  labs(x = "", y = "Kilo trips per day")


pick_good <- combine %>%

  filter(pickup_longitude > -75 & pickup_longitude < -73) %>%

  filter(pickup_latitude > 40 & pickup_latitude < 42)

pick_good <- sample_n(pick_good, 5e3)



pick_good %>%

  ggplot(aes(pickup_longitude, pickup_latitude, color = dset)) +

  geom_point(size=0.1, alpha = 0.5) +

  coord_cartesian(xlim = c(-74.02,-73.77), ylim = c(40.63,40.84)) +

  facet_wrap(~ dset) +

  #guides(color = guide_legend(override.aes = list(alpha = 1, size = 4))) +

  theme(legend.position = "none")

attach(sub_train)
sample_model = glm(trip_duration ~
  pickup_datetime:dropoff_datetime+pickup_latitude:pickup_longitude+dropoff_latitude:dropoff_longitude,
  family = poisson)

par(mfrow = c(2,3))
plot(sample_model,
     pch  = 20,
     cex  = 2,
     col  = "grey")
abline(sample_model, lwd = 3, col = "darkorange")
halfnorm(hatvalues(sample_model))

summary(sample_model)
#confint(sample_model, level = 0.99)
tidy(sample_model)
#augment(sample_model)
glance(sample_model)
cooks.distance(sample_model)[11] > 4 / length(cooks.distance(sample_model))
bptest(sample_model)
par(mfrow = c(2,3))
hist(resid(sample_model))

qqnorm(resid(sample_model), main = "Normal Q-Q Plot, sample_model", col = "darkgrey")
qqline(resid(sample_model), col = "dodgerblue", lwd = 2)
```

```
plot(log(fitted(sample_model)), log(resid(sample_model)), col = "grey", pch = 20,
     xlab = "Fitted", ylab = "Residuals", main = "Data from Model 1")
abline(h = 0, col = "darkorange", lwd = 2)

plot(which(hatvalues(sample_model) > 2 * mean(hatvalues(sample_model)), TRUE))

plot(rstandard(sample_model)[abs(rstandard(sample_model)) > 2])
cd_sample_model_add = cooks.distance(sample_model)
sum(cd_sample_model_add > 4 / length(cd_sample_model_add))


large_cd_train = cd_sample_model_add > 4 / length(cd_sample_model_add)
plot(cd_sample_model_add[large_cd_train])




coef(sample_model)
sample_model_add_fix = lm(trip_duration ~ dropoff_longitude,
                    data = train,
                    subset = cd_sample_model_add <= 4 / length(cd_sample_model_add))
coef(sample_model_add_fix)

#set.seed(42)
#shapiro.test(resid(sample_model))

boxcox(sample_model, plotit = TRUE)
attach(sub_train)
sample_model2 = glm(trip_duration ~
  pickup_datetime:dropoff_datetime+pickup_latitude:pickup_longitude+dropoff_latitude:dropoff_longitude+store_a
  family = gaussian)

par(mfrow = c(2,3))
plot(sample_model2,
     pch  = 20,
     cex  = 2,
     col  = "grey")
abline(sample_model2, lwd = 3, col = "darkorange")
halfnorm(hatvalues(sample_model2))

summary(sample_model2)
#confint(sample_model2, level = 0.99)
tidy(sample_model2)
#augment(sample_model)
glance(sample_model2)
cooks.distance(sample_model2)[11] > 4 / length(cooks.distance(sample_model2))
bptest(sample_model2)
par(mfrow = c(2,3))
hist(resid(sample_model2))

qqnorm(resid(sample_model2), main = "Normal Q-Q Plot, sample_model", col = "darkgrey")
qqline(resid(sample_model2), col = "dodgerblue", lwd = 2)

plot(log(fitted(sample_model2)), log(resid(sample_model2)), col = "grey", pch = 20,
     xlab = "Fitted", ylab = "Residuals", main = "Data from Model 2")
abline(h = 0, col = "darkorange", lwd = 2)

plot(which(hatvalues(sample_model2) > 2 * mean(hatvalues(sample_model2)), TRUE))

plot(rstandard(sample_model2)[abs(rstandard(sample_model2)) > 2])
cd_sample_model2_add = cooks.distance(sample_model2)
sum(cd_sample_model2_add > 4 / length(cd_sample_model2_add))
```

```r
large_cd_train = cd_sample_model2_add > 4 / length(cd_sample_model2_add)
plot(cd_sample_model2_add[large_cd_train])



coef(sample_model2)
sample_model2_add_fix = lm(trip_duration ~ dropoff_longitude,
                    data = train,
                    subset = cd_sample_model2_add <= 4 / length(cd_sample_model2_add))
coef(sample_model2_add_fix)

#set.seed(42)
#shapiro.test(resid(sample_model))

boxcox(sample_model2, plotit = TRUE)
attach(sub_train)
sample_model3 = lm(trip_duration ~
  pickup_datetime:dropoff_datetime+pickup_latitude:pickup_longitude+dropoff_latitude:dropoff_longitude+store_a
  family = negative.binomial(1))

par(mfrow = c(2,3))
plot(sample_model3,
     pch  = 20,
     cex  = 2,
     col  = "grey")
abline(sample_model3, lwd = 3, col = "darkorange")
halfnorm(hatvalues(sample_model3))

summary(sample_model3)
#confint(sample_model3, level = 0.99)
tidy(sample_model3)
#augment(sample_model)
glance(sample_model3)
cooks.distance(sample_model3)[11] > 4 / length(cooks.distance(sample_model3))
bptest(sample_model3)
par(mfrow = c(2,3))
hist(resid(sample_model3))

qqnorm(resid(sample_model3), main = "Normal Q-Q Plot, sample_model", col = "darkgrey")
qqline(resid(sample_model3), col = "dodgerblue", lwd = 2)

plot(log(fitted(sample_model3)), log(resid(sample_model3)), col = "grey", pch = 20,
     xlab = "Fitted", ylab = "Residuals", main = "Data from Model 3")
abline(h = 0, col = "darkorange", lwd = 2)

plot(which(hatvalues(sample_model3) > 2 * mean(hatvalues(sample_model3)), TRUE))

plot(rstandard(sample_model3)[abs(rstandard(sample_model3)) > 2])
cd_sample_model3_add = cooks.distance(sample_model3)
sum(cd_sample_model3_add > 4 / length(cd_sample_model3_add))


large_cd_train = cd_sample_model3_add > 4 / length(cd_sample_model3_add)
plot(cd_sample_model3_add[large_cd_train])



coef(sample_model3)
sample_model3_add_fix = lm(trip_duration ~ dropoff_longitude,
                    data = train,
                    subset = cd_sample_model3_add <= 4 / length(cd_sample_model3_add))
```

```r
coef(sample_model3_add_fix)

#set.seed(42)
#shapiro.test(resid(sample_model))

boxcox(sample_model3, plotit = TRUE)
m1AIC <- AIC(sample_model)
m1BIC <- BIC(sample_model)
m2AIC <- AIC(sample_model2)
m2BIC <- BIC(sample_model2)
m3AIC <- AIC(sample_model3)
m3BIC <- BIC(sample_model3)


AIC <- list(m1AIC, m2AIC, m3AIC)
BIC <- list(m1BIC, m2BIC, m3BIC)
kable(rbind(AIC, BIC), col.names = c("Model 1", "Model 2", "Model 3"))  %>%
  kable_styling(full_width = T)
eval_p<-predict(sample_model3,sub_train, type = "response")
write.csv(eval_p,"predicted_eval_values.csv")
par(mfrow = c(1,3))
train_p<-predict(sample_model,sub_train, type = "response")
plot(train_p,sub_train$trip_duration,main = "Model 1")
train_p<-predict(sample_model2,sub_train, type = "response")
plot(train_p,sub_train$trip_duration,main = "Model 2")
train_p<-predict(sample_model3,sub_train, type = "response")
plot(train_p,sub_train$trip_duration,main = "Model 3")
```

References:

1. Traffic congestion costs metro economy $20 billion a year: study. Crain's New York Business. https://www.crainsnewyork.com/article/20180118/POLITICS/180119895/traffic-congestion-costs-metro-economy-20-billion-a-year-study. Published January 18, 2018. Accessed November 13, 2019.

2. FDNY: Traffic — Not Bike Lanes — is to Blame for Increased Response Times. Streetsblog New York City. https://nyc.streetsblog.org/2019/09/19/fdny-traffic-not-bike-lanes-is-to-blame-for-increase d-response-times/. Published September 19, 2019. Accessed November 13, 2019.

3. Skandul, E. How New York City taxis can get ahead of Uber and Lyft. CSNY. https://www.cityandstateny.com/articles/opinion/commentary/how-new-york-city-taxis-can-get-ahead-of-uber-and-lyft.html. Published October 17, 2019. Accessed November 13, 2019.

4. Sanders A. De Blasio struggles to find new Taxi and Limousine commissioner after first pick's disastrous performance at Council Hearing. nydailynews.com. https://www.nydailynews.com/news/politics/ny-city-council-de-blasio-official-tlc-commissioner-jeff-lynch-20191003-cfx27yihafcebj4e62wytsprpu-story.html. Published October 3, 2019. Accessed November 13, 2019.

5. New York City Taxi Trip Duration | Kaggle. Kaggle.com. https://www.kaggle.com/c/nyc-taxi-trip-duration/data#. Published 2017. Accessed November 13, 2019.

6. About TLC - TLC. Nyc.gov. https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page. Published 2019. Accessed November 13, 2019.

7. Use holidays as a feature | Kaggle. Kaggle.com. https://www.kaggle.com/c/nyc-taxi-trip-duration/discussion/37192#latest-572412. Published 2017. Accessed November 13, 2019.