TAXI TRIP DURATION PREDICTION

1. Introduction

Yellow Taxicabs are very iconic to New York City and they are widely recognizable symbols of the city. These millions of rides that happen can give us insights into the city traffic patterns and users commute preferences. Our objective of the project is to predict the duration of taxi rides in New York City using Linear Regression model.

Google Maps provides a transit time estimator by leveraging the real-time traffic and historical data. Our model estimates the time taken using the non-real time taxi rides data and only data which would be available at the beginning of a ride was used eg: pickup and drop-off coordinates, pickup time, number of passengers etc. Predicting the duration of trips can help users plan their commute well, drivers can decide between potential rides better and Taxi Commision can use these predictions for allocating no. of taxis to a stand.

1.1. Data

The taxi trips dataset is made available by this Kaggle Competition based on the NYC Taxi and Limousine Commission (TLC) 2016 NYC yellow Cab data. The data comes in the shape of 1.5 million training observations and 630k test observations. Each row contains one taxi trip details.

Attributes are as follows: 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 disengage, 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. Data was processed to remove outliers and to build new features like the month, weekday, hour of the day from the pickup date and time of each ride to help us build the model.

Weather is known to affect the road conditions and has a severe effect on the time taken for a ride completion .To verify and include this effect; we used New York weather data for 2016 collected by the National Weather Service.

1.2. Loading data

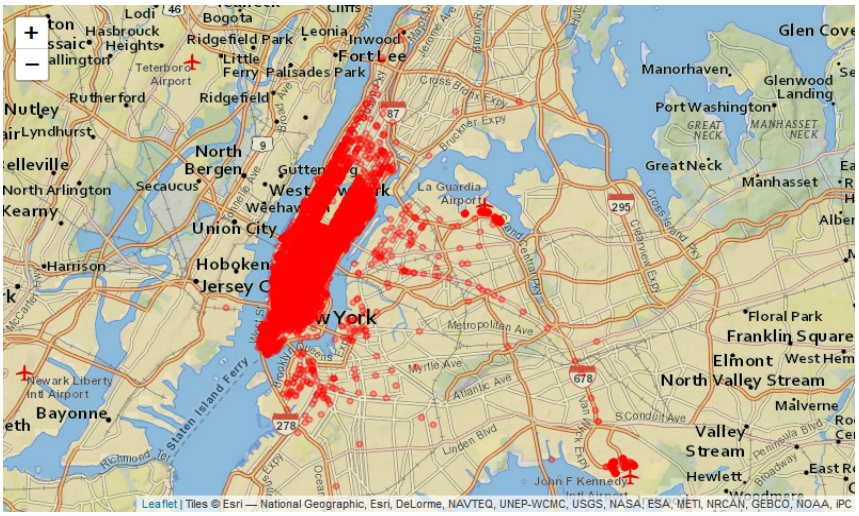
```
train <- as.tibble(fread('./train.csv'))
test <- as.tibble(fread('./test.csv'))
```

2. Individual Feature Analysis

In this section, we try to understand how each individual feature is related to the response variable through techniques such as partial regression and visualizations. Since most of the features we have in the data set are categorical we preferred visual analysis through relevant plots rather than statistical techniques like partial regression.

2.1. Visualizing pick up locations

```
set.seed(100)
foo <- sample_n(train, 8e3)
leaflet(data = foo) %>% addProviderTiles("Esri.NatGeoWorldMap") %>%
  addCircleMarkers(~ pickup_longitude, ~pickup_latitude, radius = 1,
                   color = "red", fillOpacity = 0.3)
```



Using the leaflet package we visualized where the trips originate. We see that most of the trips are in the Manhattan city area. One interesting aspect we found with much analysis was that most trips where the duration was high were airport trips i.e either trips ending up at JFK or LAGuardia or starting there.

2.2. Data Exploration

2.2.1. Plots – 1

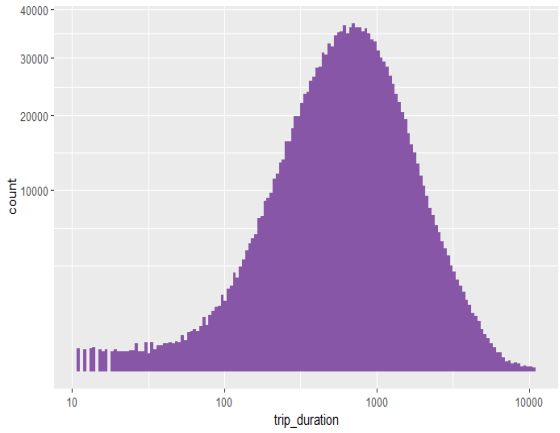


Fig 1. Histogram – Trip durations

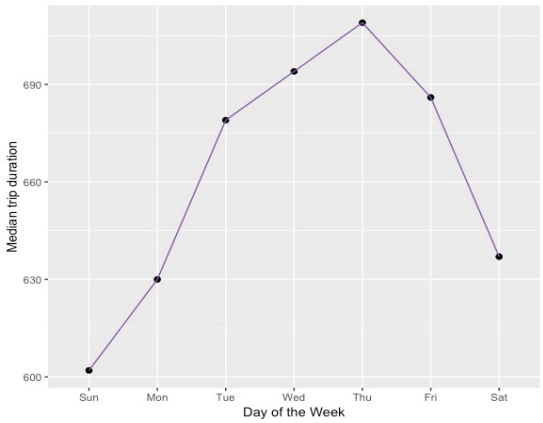


Fig 2. Duration vs Day of the week

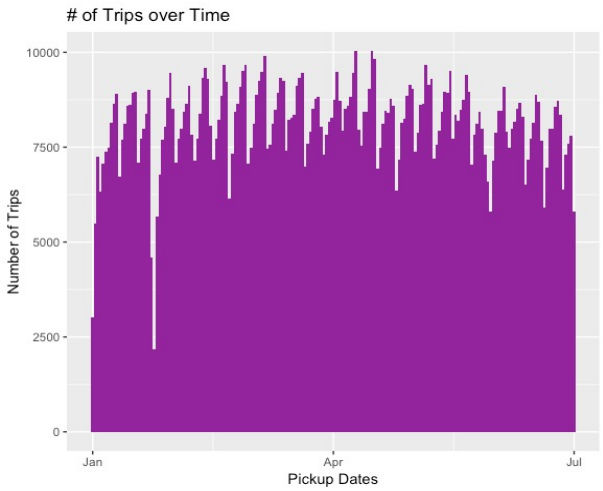


Fig 3. # Trips vs. Pickup dates

Inference

- Fig 1: From the trip durations histogram we see that the mean trip duration is 900 seconds
- Fig 2: We see that trip duration is higher on the week-days as opposed to that on the week-ends
- Fig 3: There is a sharp drop in the number of trips towards the end of Jan 2018. This motivated us to use weather data to check for influence of blizzards on trip durations

2.2.2. Plots – 2

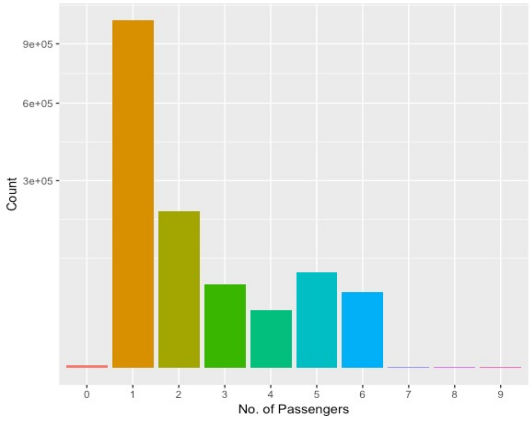


Fig 4. # Passengers per trip vs. # Trips

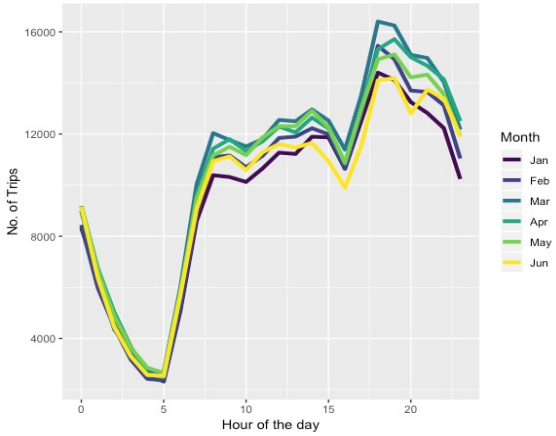


Fig 5. # Trips vs. Hour of the day (By month)

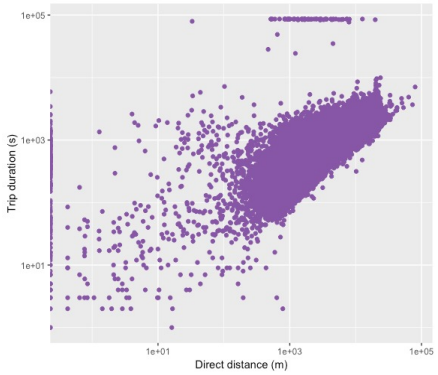


Fig 6. Distance traveled vs. Duration

Inference

- Fig 4: Lesser number > 6 passenger trips. Total pickups increases towards the later part of the week
- Fig 5: Number of trips increases towards the later part of the day. Trend is consistent across months
- Fig 6: Relationship between distance and time is noted

3. Feature Engineering

Direct distance of the trip: Existing features do not already include distance. We compute the approximate distance between a pair of coordinates using the geosphere package. . This helped us to find some outliers.

Work: If day is workday/weekend (Bool). Public Holidays were excluded from work days

Blizzard: If there is snow fall on a given day (Boolean)

4. Unusual Observations

4.1. Checking for inconsistencies in data

We will check to see if there are any outliers and will remove them from our analysis. We checked for all the following scenarios and the corresponding number of data points that satisfied that criteria.

1. Trips where the duration was greater than 3 hours – 2,108 data points
2. Trips which had very high average speeds (> 80mph) – 129 data points
3. Trips where distance traveled is zero – 4,210 data points
4. Trips which had 300 km long pickups and drops – 31 data points
5. Trips where the duration of the trip is less than 10 seconds – 3,406 data points

Code used to filter these data points is as follows

```
train <- train %>%
  filter(trip_duration <= 3*3600,
         dist > 0 | (near(dist, 0) & trip_duration < 60),
         jfk_dist_pick < 300000 & jfk_dist_drop < 300000,
         trip_duration >10,
         speed < 128)
```

A total of 1450082 records are left in the training set.

4.2. Leverages

First we build a base model and then we calculate the hatvalues associated with the model and then we plot these hatvalues against half-normal quantiles and check to see if there are major deviations. Code is as follows:

```
lmod=lm(trip_duration ~ vendor_id + passenger_count + store_and_fwd_flag +
dist + date + month + wday + hour + work + blizzard,data=train)

hatv <- hatvalues(lmod)
ids <- train$id
halfnorm(hatv, labs=ids, ylab = "Leverages")
abline(0,1)
```

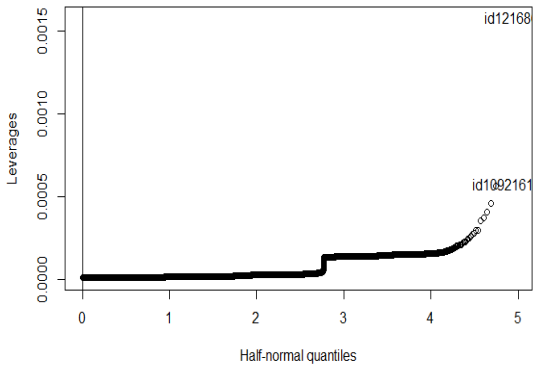


Fig 7. Leverage Points

This is the plot from code above

- Data points id1092161 and id1216866 are points with high leverages

4.3. Outliers

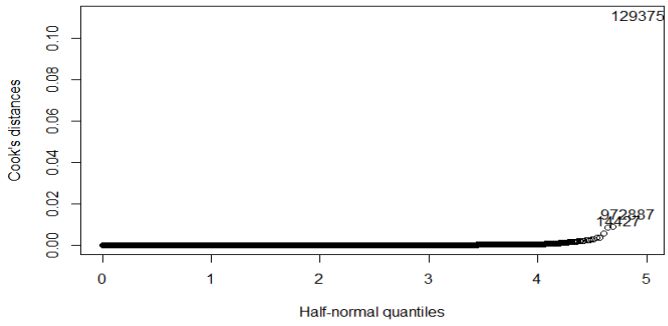
We used Bonferroni estimate to check for outliers. Bonferroni critical value is: -5.325801. Checking points above this estimate

```
stud <- rstudent(lmod)
length(stud[abs(stud)> 5.325801])
```

[1] 3480

4.4. Influential Observations

```
cook <- cooks.distance(lmod)
halfnorm(cook, 3,ylab="Cook's distances")
```



- We checked the cook's distance of the points and there is one point "id1489236" which showed up as an influential observation

Fig 8. Influential Points

For outliers and influential observations we checked our models by removing these data points from the model and seeing if they influence the model or not and based on that we took a call on retaining them or removing them.

5. Regression Analysis

5.1. Diagnostics

5.1.1. Checking Error Assumptions on base model

Base model – Normality check

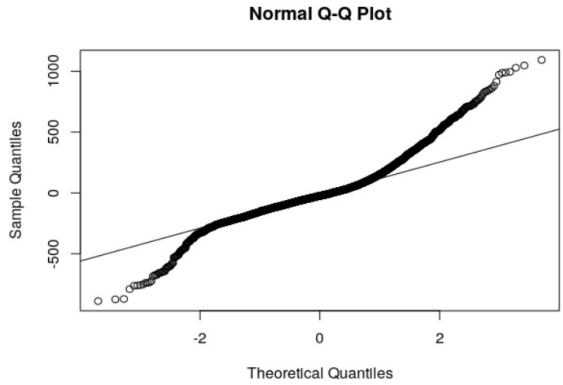


Fig 9. Q-Q plot on base model

Base model – Heteroscedasticity check

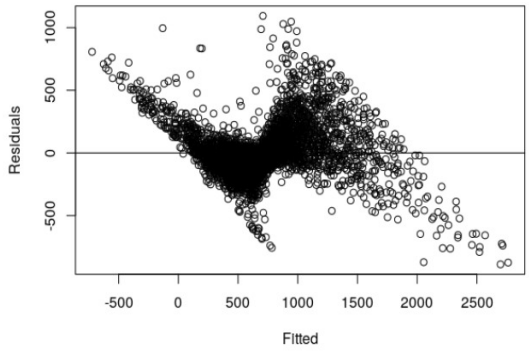


Fig 10. Fitted vs. Residuals for base model

We see that this is a distribution with long tails. For long-tailed distributions, the consequences of non-normality could imply a non-linearity in the structure.

5.1.2. Box-Cox Transformation

```
boxcox(lmod,plotit=T)
```

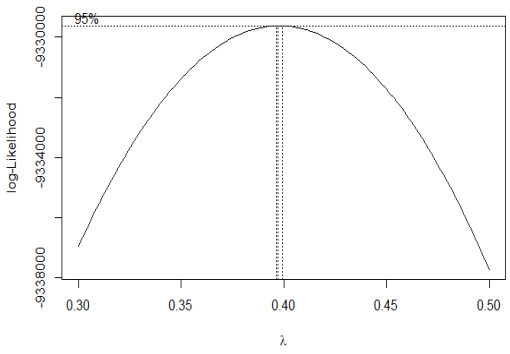


Fig 11. Box-cox plot (lambda vs. log-likelihood)

- From this we choose lambda = 0.397 for the transformation.

5.1.3. Transformed model

```
lmod=lm((trip_duration)^0.397~vendor_id + passenger_count +
    store_and_fwd_flag + dist + date + month + wday + hour +
    work + blizzard,data=train)
```

5.1.4. Checking error assumptions on transformed model

Transformed model – Normality check

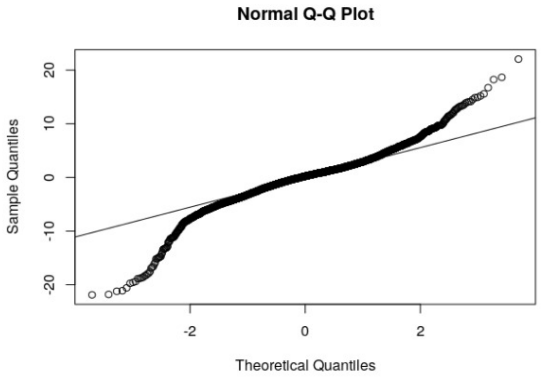


Fig 12. Q-Q plot on base model

Heteroscedasticity check

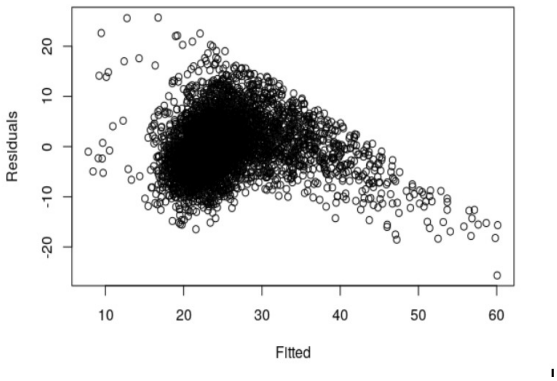


Fig 13. Fitted vs. Residuals for base model

After the transformation non-constant variance has reduced. But even after transformation non-normality is an issue. To confirm that the residuals are not normal, we used the Shapiro method to test. We obtain the following results.

```
Shapiro-Wilk normality test data:

residuals(lmod) W = 0.94633, p-value < 2.2e-16
```

Shapiro method also confirms what have found in the QQ plot. Therefore the structure is not at all linear. We will try to model the linear aspect using linear regression and try to capture non-linearity as much as possible using the addition non-linear features.

5.1.5. Correlated Errors

```
n <- length(residuals(lmod))
plot(tail(residuals(lmod),n-1) ~ head(residuals(lmod),n-1),
xlab=expression(hat(epsilon)[i]),ylab=expression(hat(epsilon)[i+1]))
```

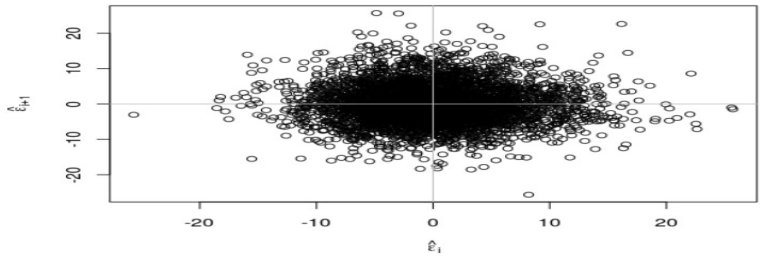


Fig 13. Correlated Errors

As we can see the successive errors are not correlated. We verify this by fitting a linear regression over the residuals i.e. (e_i+1 ~ e_i) and the R-Squared obtained is almost zero

```
summary(lm(tail(residuals(lmod),n-1) ~ head(residuals(lmod),n-1) -1))
Residual standard error: 5.554 on 4691 degrees of freedom
Multiple R-squared:  0.0004199,    Adjusted R-squared:  0.0002069
F-statistic: 1.971 on 1 and 4691 DF,  p-value: 0.1604
```

6. Model selection

After the required transformations and cleaning needed for the linear regression we now select the most important features using model selection techniques such as LASSO & step-wise linear regression.

6.1. Lasso Regression

```
require(lars)
lassomod <-
lars(as.matrix(train[,c('vendor_id','passenger_count','dist','hour','blizzard
','work')]),as.matrix(train$trip_duration^0.397))
```

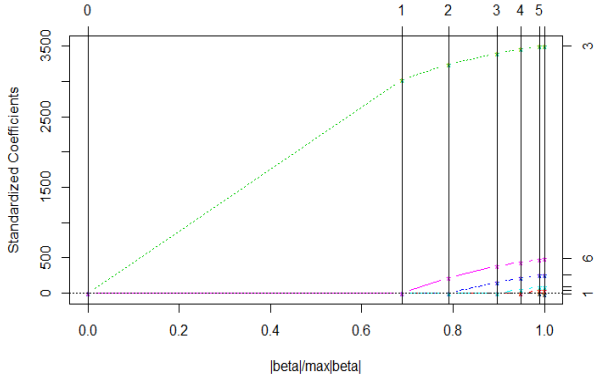


Fig 14. Lasso Regression

- From the lasso plot we see that only "vendor_id", "distance" and "work" are sufficient to explain the response.

We fit the model again with just these three predictors and obtain an R2 of 53.46%

6.2. Stepwise Regression

```
step(lmod)
```

Step-wise regression returns the best model with AIC=2851746. It selects 9 out of 10 predictors provided in the original model. Selected model is

(trip_duration)^0.397 ~ vendor_id + passenger_count + store_and_fwd_flag + dist + month + wday + hour + work + blizzard

6.3. Final selection

Since Lasso specifies a smaller set of predictors and the model has almost the same amount of explanation in terms of R^2, we proceed with predictors/model suggested by Lasso regression. Summary of different model performances is presented in Exhibit 1 at the end of the document.

7. Correlation Analysis

Analyzing the correlation matrix we find that trip duration is highly correlated with trip distance.

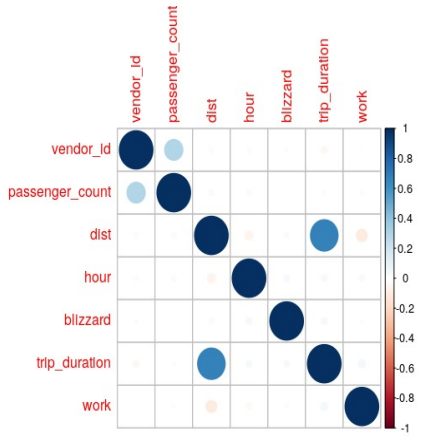


Fig 15. Correlation Plot

8. Interpretation

8.1. Explaining the predictors and their significance

We selected the model suggested by Lasso. The model is as follows:

```
summary(lm(trip_duration^0.397 ~ vendor_id + work + dist, train))
```

Vendor-id, work-True and distances are able to explain the variance in cab trip duration by 53.46%. All the predictors are significant at p-value of 0.05. Their interpretation is:

- Vendor ID is significant , taxis of vendor 1 takes longer time to complete the trip compared to vendor 2
- Trips on work days take longer time compared to that of week ends
- Extreme weather has impact on trip duration

8.2. Exploring interaction terms

We checked the interaction between the predictors using

```
summary(lm(trip_duration^0.397 ~ vendor_id + work + dist + work:vendor_id +
dist:work + vendor_id:dist, train))
```

We found no significant change in the R^2 value. Hence, we go by the model without the interaction terms.

9. Exhibits

9.1. Exhibit 1

Summary of Model Performances

Model Config	Model selection method	R2	RMSE (on test set)
(trip_duration)^0.397~ vendor_id + passenger_count + store_and_fwd_flag + dist + date + month + wday + hour + work + blizzard	After Box-Cox	55.17%	0.36730
trip_duration^0.397 ~ vendor_id + work + dist	Lasso	53.46%	0.31152
(trip_duration)^0.397 ~ vendor_id + passenger_count + store_and_fwd_flag + dist + month + wday + hour + work + blizzard	Stepwise	55.1%	0.36635
trip_duration^0.397 ~ vendor_id + work + dist + work:vendor_id + dist:work + vendor_id:dist	Interactions	53.46%	0.33683