# Capstone Project – 2

## NYC Taxi Trip Time Prediction

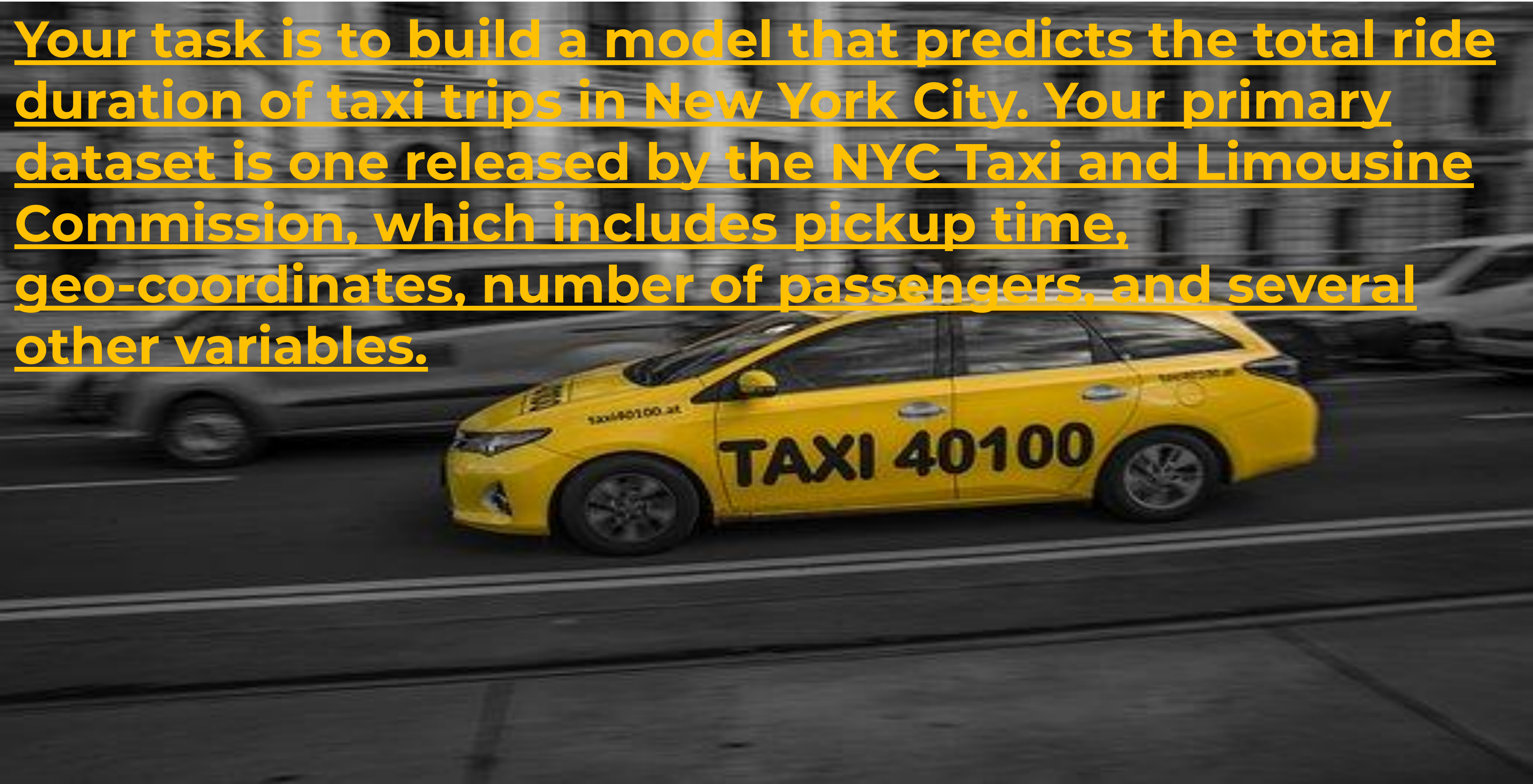
**By - Wasim Khan**

# Presentation Outline :

- Business Objective
- Exploring the Dataset
- Data Cleaning & Initial EDA
- Methodology
- EDA & Data Preprocessing
- Decomposition of Data : PCA
- Machine Learning Model Evaluation: Regression
- Model Evaluation Results
- Conclusion
- Recommendations

# Business Objective

Your task is 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.



# Exploring the Dataset

# Data Summary:

**Data Set Name** -- NYC Taxi Data.csv - the training set

**Statistics** --

- Rows - 1458644
- Features - 11 (Including Target)
- Target – Trip Duration

**Important Column** -- ‘id’, 'vendor_id', 'pickup_datetime', 'dropoff_datetime', 'passenger_count', 'pickup_longitude', 'pickup_latitude', 'dropoff_longitude', 'dropoff_latitude', 'store_and_fwd_flag', 'trip_duration’.

# Data Menu :

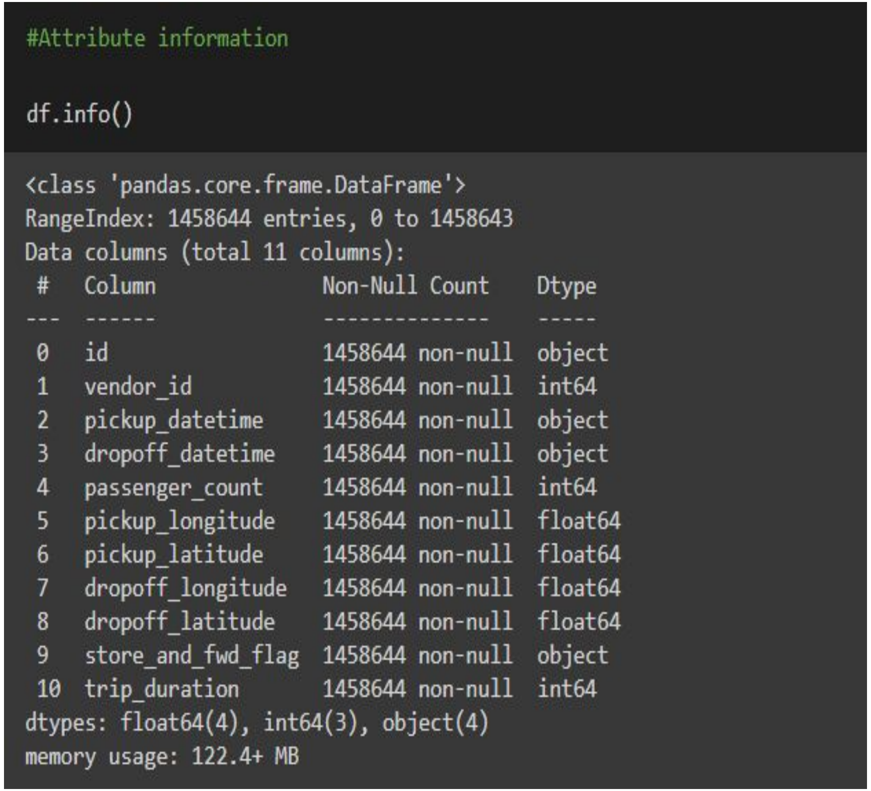
## Independent Variables --

- 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.

## Target Variable --

- trip_duration—duration of the trip in seconds

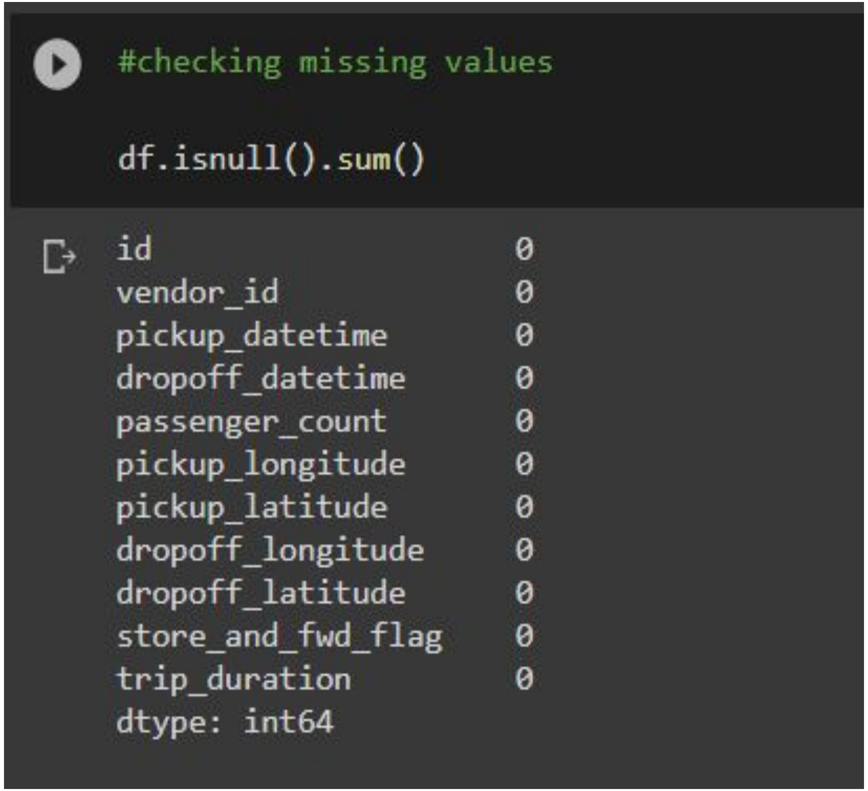
# Attribute Information : Dtype & Null values



```
#Attribute information

df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1458644 entries, 0 to 1458643
Data columns (total 11 columns):
 #   Column              Non-Null Count    Dtype
---  ------              --------------    -----
 0   id                  1458644 non-null  object
 1   vendor_id           1458644 non-null  int64
 2   pickup_datetime     1458644 non-null  object
 3   dropoff_datetime    1458644 non-null  object
 4   passenger_count     1458644 non-null  int64
 5   pickup_longitude    1458644 non-null  float64
 6   pickup_latitude     1458644 non-null  float64
 7   dropoff_longitude   1458644 non-null  float64
 8   dropoff_latitude    1458644 non-null  float64
 9   store_and_fwd_flag  1458644 non-null  object
 10  trip_duration       1458644 non-null  int64
dtypes: float64(4), int64(3), object(4)
memory usage: 122.4+ MB
```

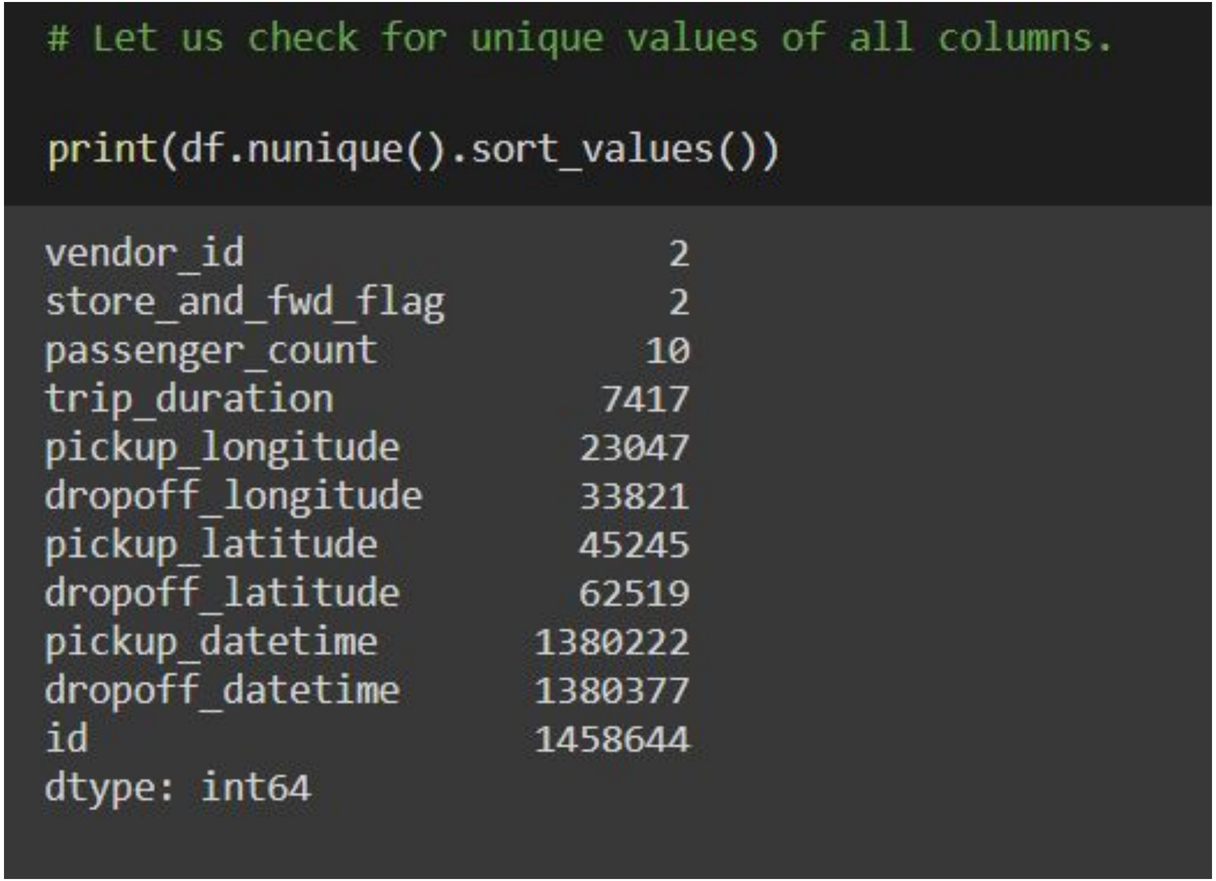


```
#checking missing values

df.isnull().sum()
```

```
id                    0
vendor_id             0
pickup_datetime       0
dropoff_datetime      0
passenger_count       0
pickup_longitude      0
pickup_latitude       0
dropoff_longitude     0
dropoff_latitude      0
store_and_fwd_flag    0
trip_duration         0
dtype: int64
```

# Attribute Information : Unique Values



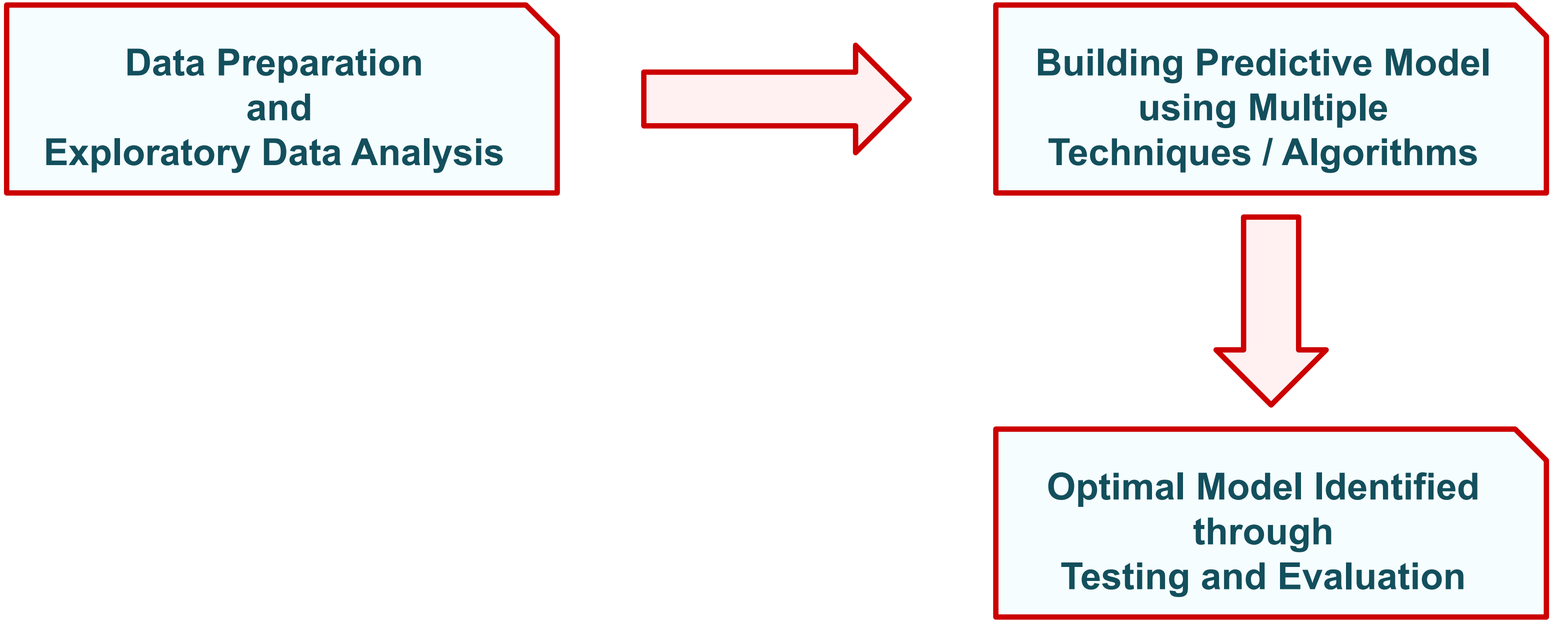
```
# Let us check for unique values of all columns.

print(df.nunique().sort_values())
```

```
vendor_id                   2
store_and_fwd_flag          2
passenger_count            10
trip_duration            7417
pickup_longitude        23047
dropoff_longitude       33821
pickup_latitude         45245
dropoff_latitude        62519
pickup_datetime       1380222
dropoff_datetime      1380377
id                    1458644
dtype: int64
```

# Methodology

# Approach : Architecture



Data Preparation
and
Exploratory Data Analysis

Building Predictive Model
using Multiple
Techniques / Algorithms

Optimal Model Identified
through
Testing and Evaluation

## Machine Learning Algorithm

❑ Decomposition : PCA

❑ Linear Regression

❑ Decision Tree

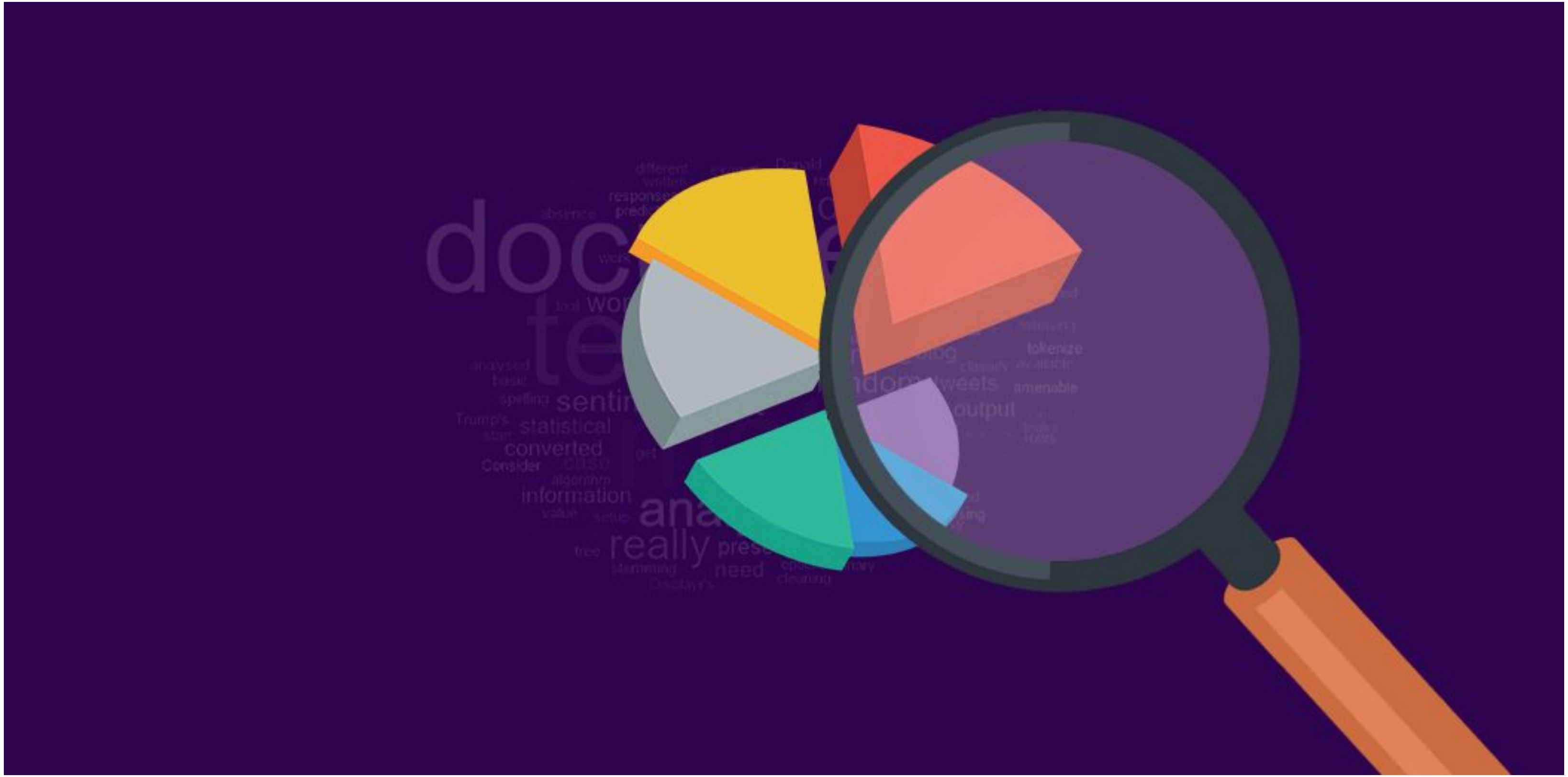
❑ Random Forest

## Tools Used

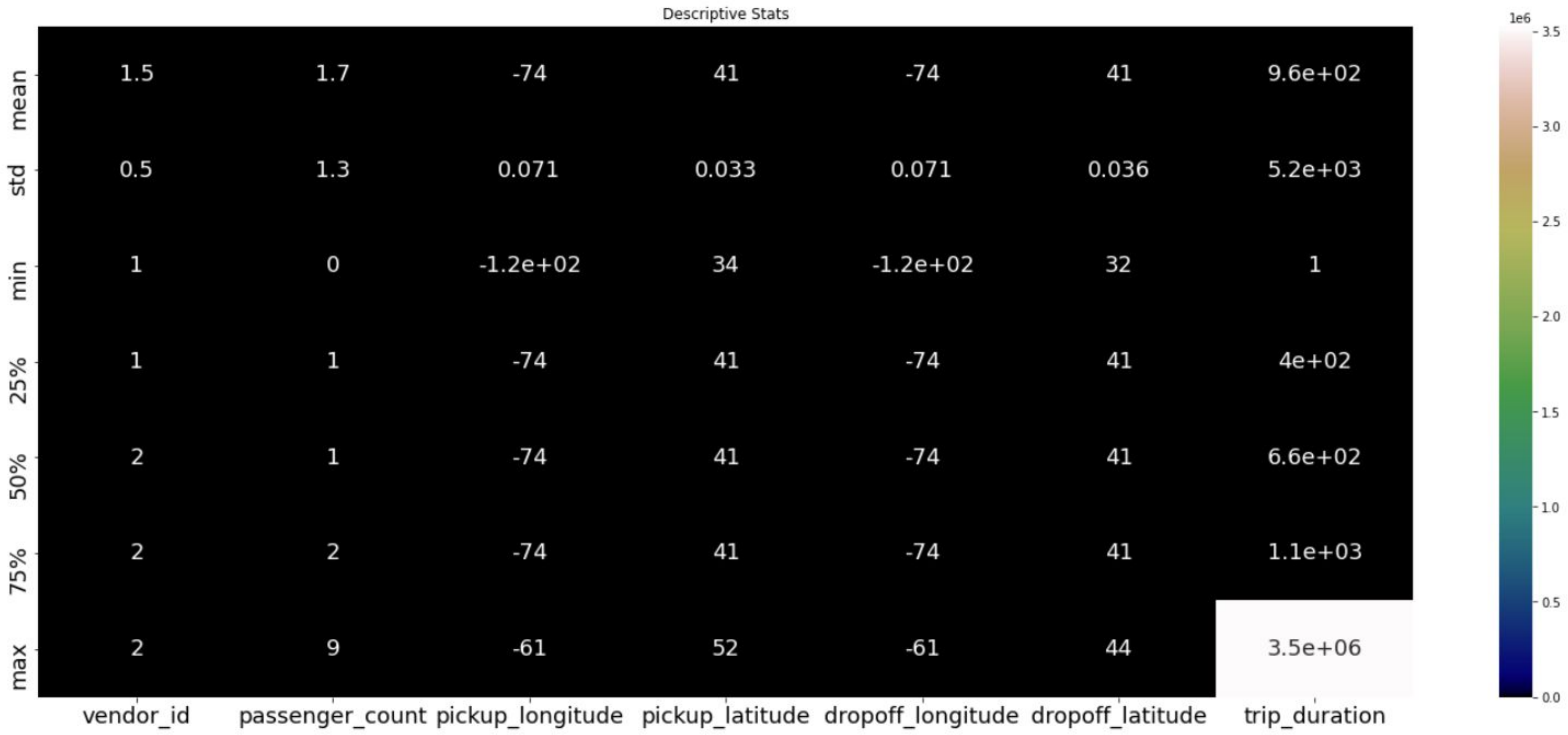
❑ Jupyter Notebook(Python)

❑ Google Collab Research

# EDA & Data Preprocessing



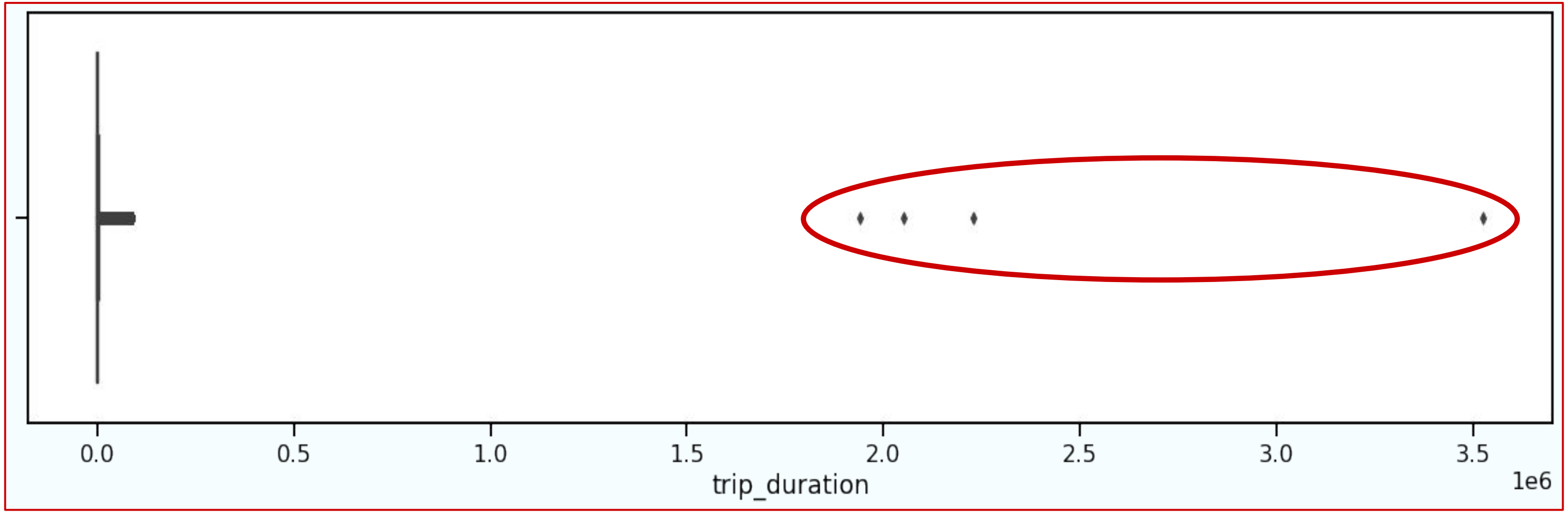
# Descriptive Stats in visual form :



Descriptive Stats

| | vendor_id | passenger_count | pickup_longitude | pickup_latitude | dropoff_longitude | dropoff_latitude | trip_duration |
|---|---|---|---|---|---|---|---|
| mean | 1.5 | 1.7 | -74 | 41 | -74 | 41 | 9.6e+02 |
| std | 0.5 | 1.3 | 0.071 | 0.033 | 0.071 | 0.036 | 5.2e+03 |
| min | 1 | 0 | -1.2e+02 | 34 | -1.2e+02 | 32 | 1 |
| 25% | 1 | 1 | -74 | 41 | -74 | 41 | 4e+02 |
| 50% | 2 | 1 | -74 | 41 | -74 | 41 | 6.6e+02 |
| 75% | 2 | 2 | -74 | 41 | -74 | 41 | 1.1e+03 |
| max | 2 | 9 | -61 | 52 | -61 | 44 | 3.5e+06 |

# Analysis on : Target Variable – Trip Duration

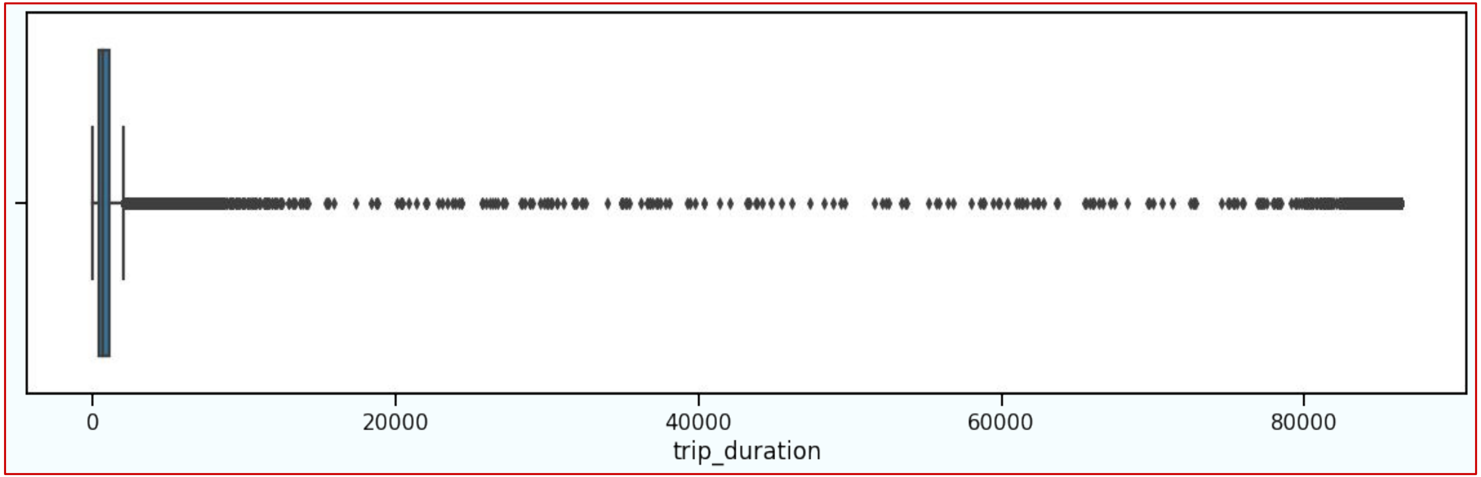


**Trip Duration Viz.**

**Probably in this visualization we can clearly see some outliers (marked in Red) , their trips are lasting between 1900000 seconds (528 Hours) to somewhere around 3500000 (972 hours) seconds which is impossible in case of taxi trips , How can a taxi trip be that long ? It's Quite suspicious. We'll have to get rid of those Outliers or else it'll affect our model's performance.**

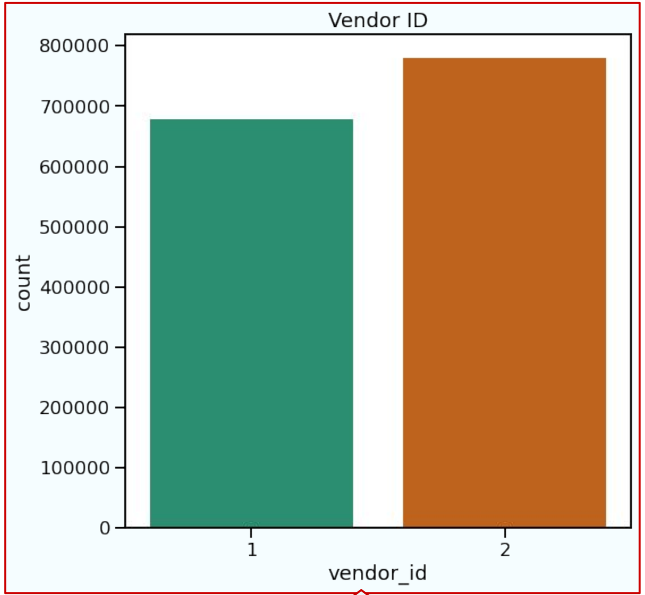
# Analysis on : Target Variable – Trip Duration (contd.)

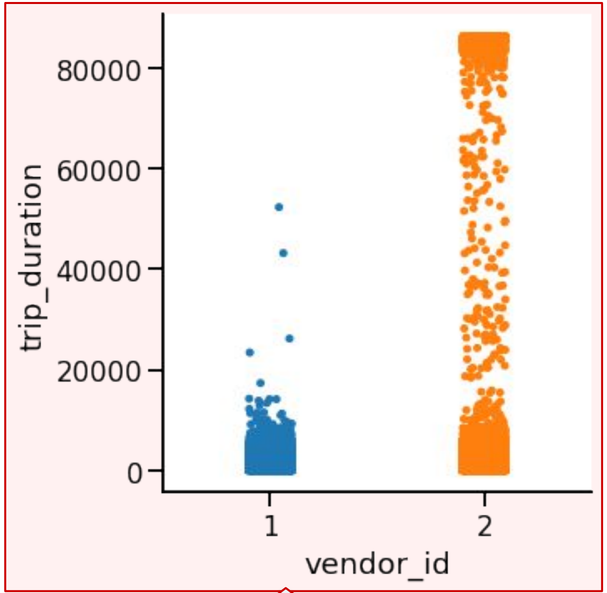
There were some entries in Trip duration which is significantly different from others. As there is this 4 rows only, we drop this rows. Then printed a plot of Trip Duration.



Now it looks Good.

# Analysis on : Vendor ID

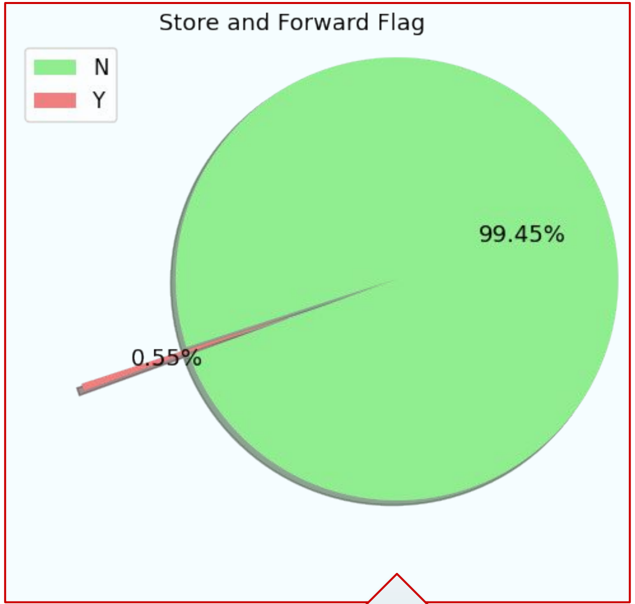


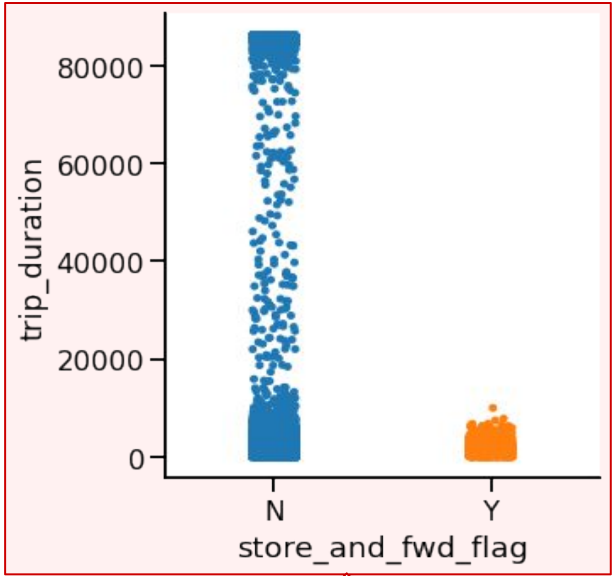


Vendor_ID stating the provider associated with trip, preferably 2 different taxi companies.
Analysis tells us that Second service provider has been most frequently opted by people over First service provider over the period of time.

Vendor id 2 takes longer trips as compared to vendor 1.

# Analysis on : Store and Forward Flag

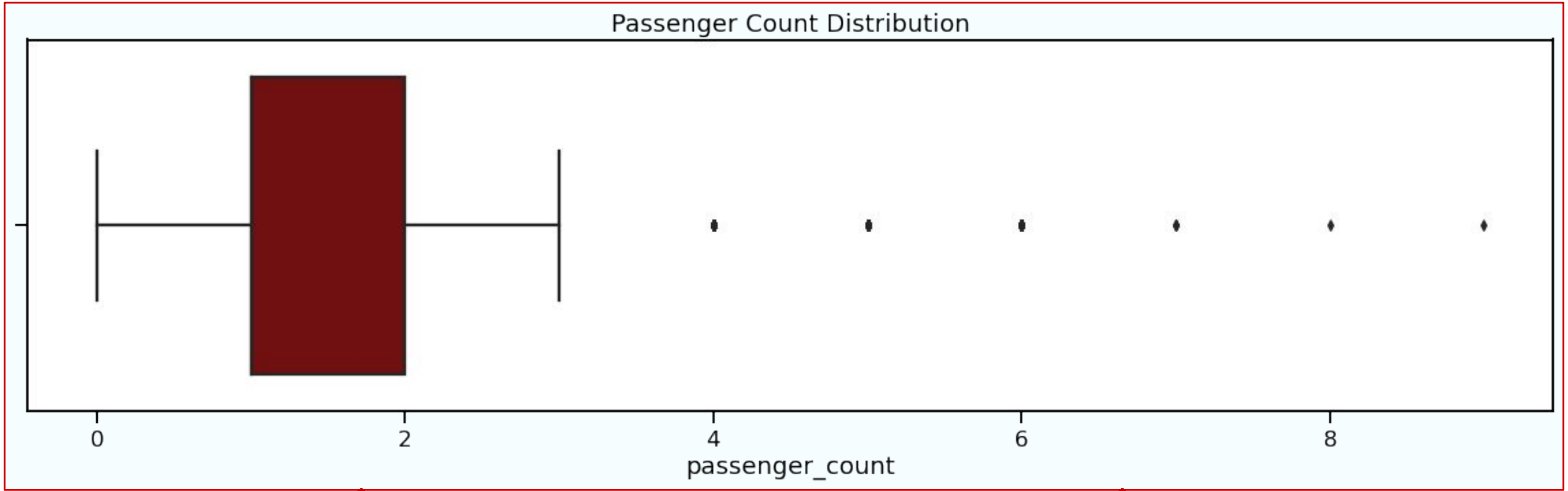




- We see there are less than 1% of trips that were stored before forwarding
- The number of N flag is much larger. We can later see whether they have any relation with the duration of the trip

Trip duration is generally longer for trips whose flag was not stored.

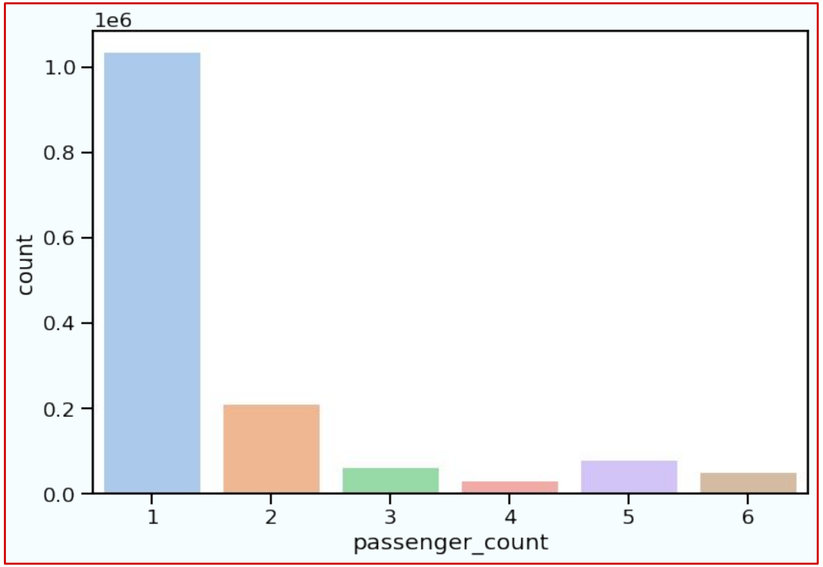
# Analysis on : Passenger Count

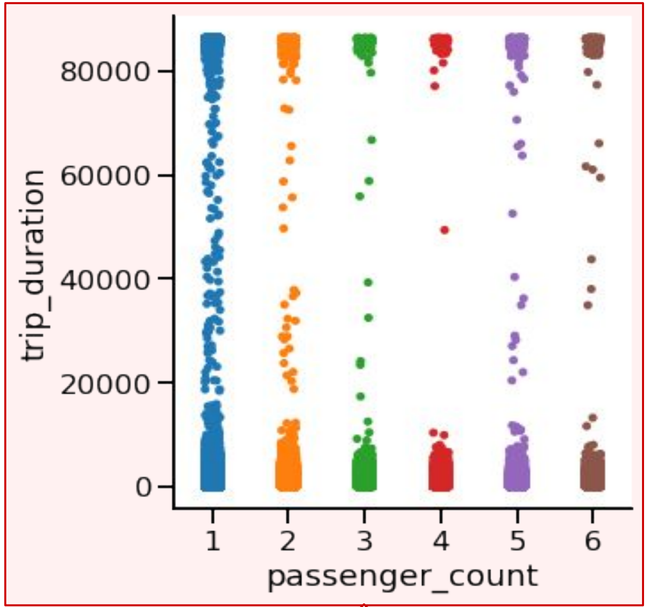


There are some trips with even 0 passenger count.
And 3 trips with 7 passenger. And there is only 1 trip each for 8 and 9 passengers.

Most number of trips are done by single or double passengers.
But one thing is Interesting to observe, there exist trip with ZERO passengers, was that a free ride ? Or just a False data recorded ?

# Analysis on : Passenger Count (contd.)



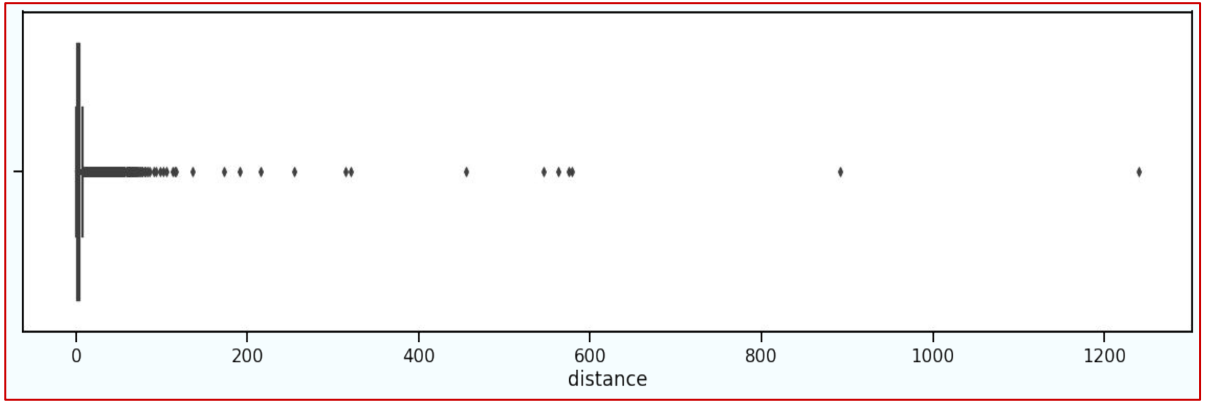


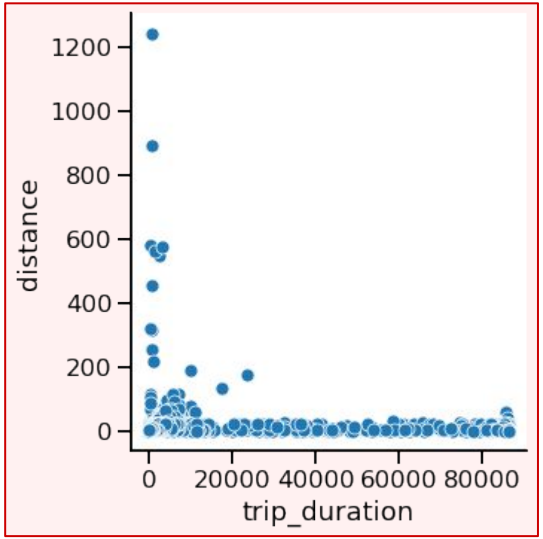
Now, that seems like a fair distribution.

We see the highest amount of trips are with 1 passenger.

There is no visible relation between trip duration and passenger count.

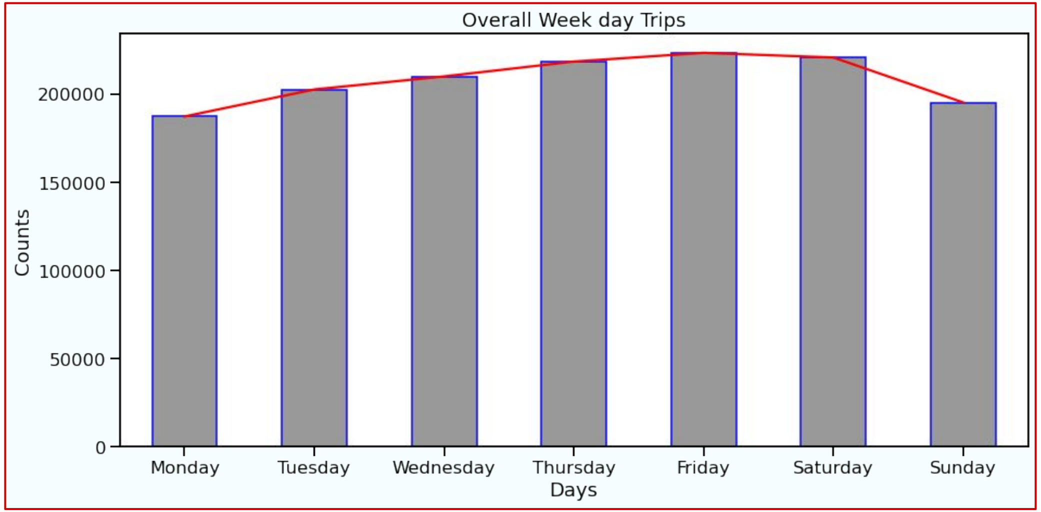
# Analysis on : Distance (pickup_longitude, pickup_latitude, dropoff_longitude, and dropoff_latitude)

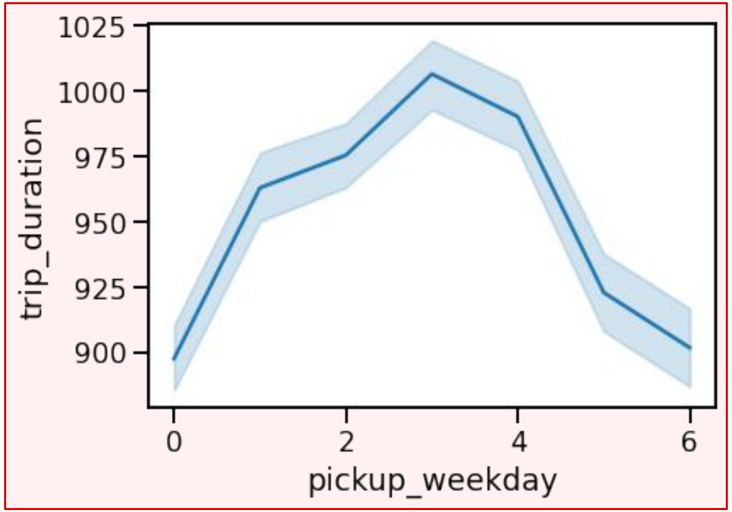




- We can see there are trips which trip duration as short as 0 seconds and yet covering a large distance. And, trips with 0 km distance and long trip durations.
- One reason can be that the dropoff coordinates weren't recorded.
- Another reason one can think is that for short trip durations, maybe the passenger changed their mind and cancelled the ride after some time.

# Analysis on : Trip Duration in Weekday

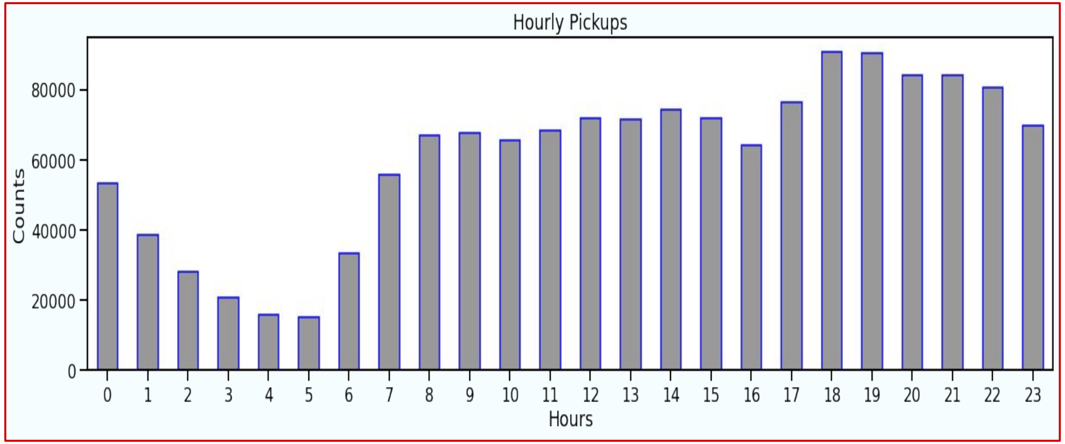


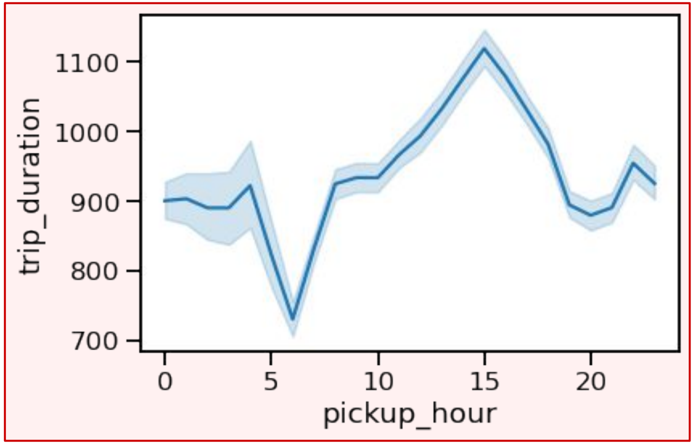


**Observations says that Fridays and Saturdays are those days in a week when New Yorkers prefer to rome in city. GREAT !!**

**Trip duration is the longest on Thursdays closely followed by Fridays.**

# Analysis on : Trip Duration Per Hour

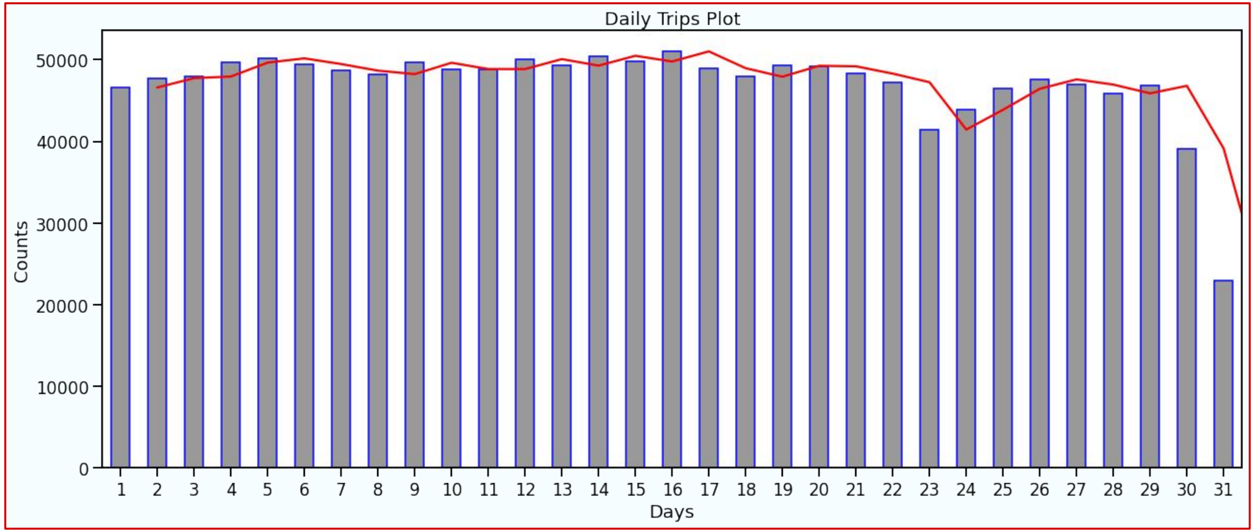




In which hour we get to see maximum trips ? - Rush hours (5pm to 10pm)

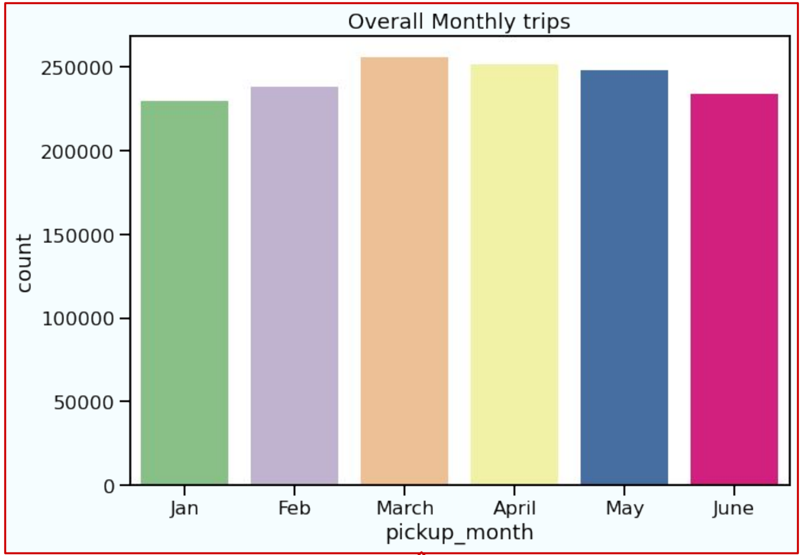
We see the trip duration is the maximum around 3 pm which may be because of traffic on the roads. Trip duration is the lowest around 6 am as streets may not be busy.

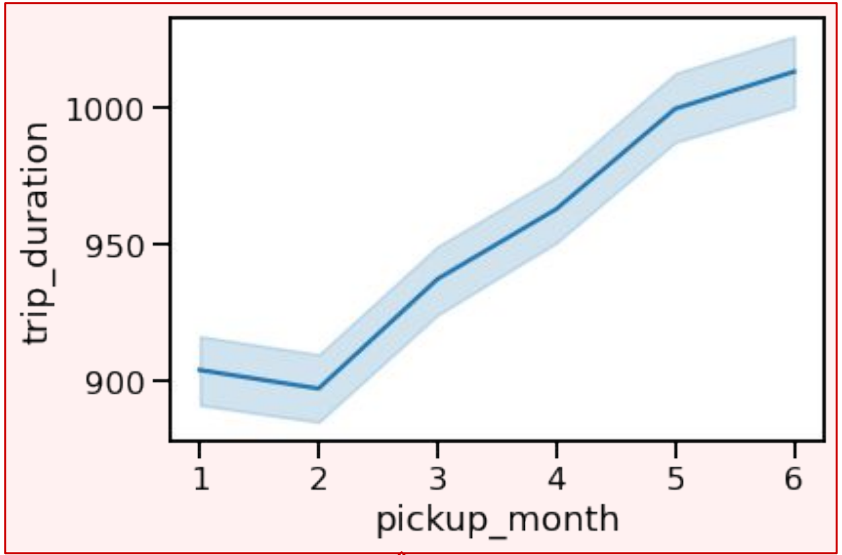
# Analysis on : Trip Duration in a Month



**Seem like New Yorker’s do not prefer to get a Taxi on Month end , there is a significant drop in the Taxi trip count as month end approach.**

# Analysis on : Trip Duration in 6 Months

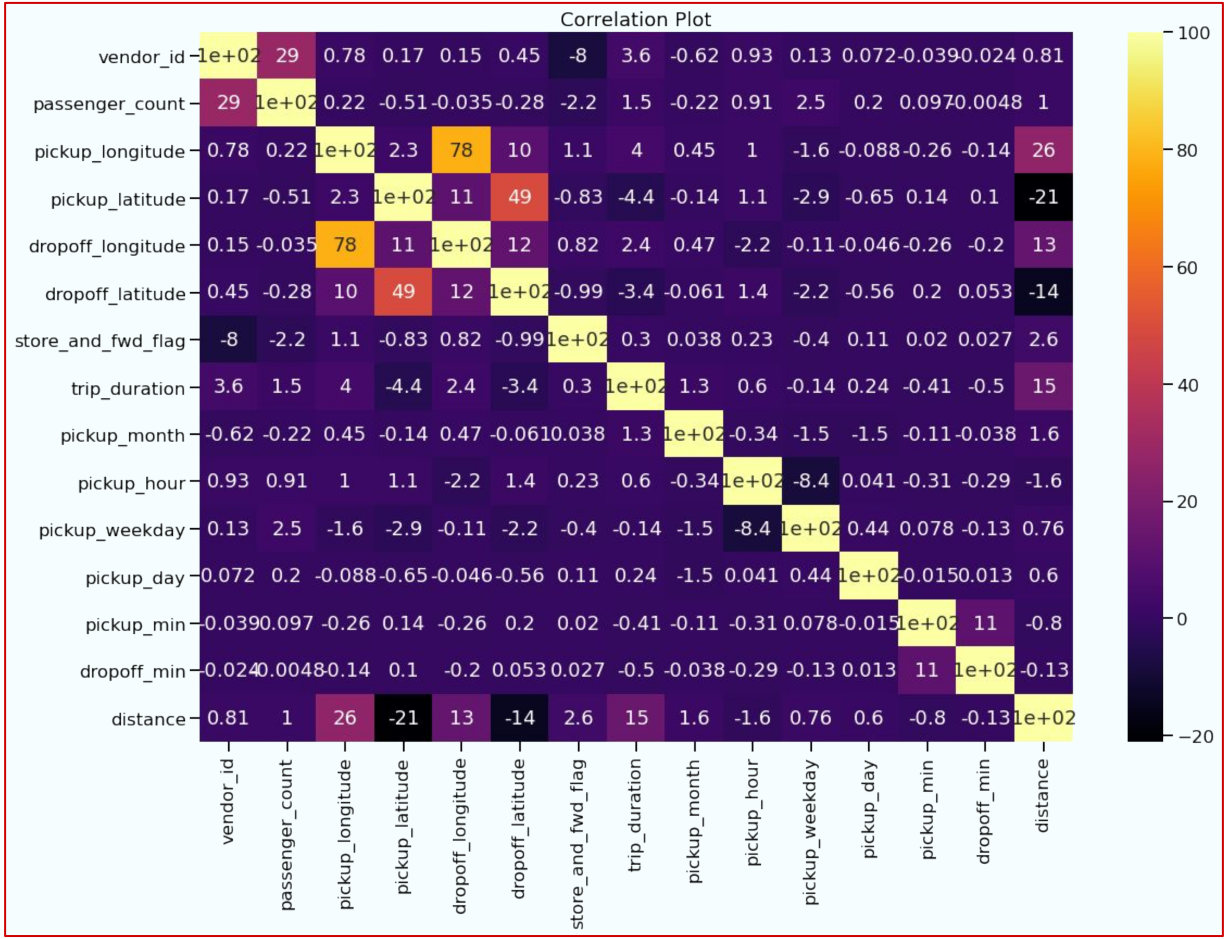




Number of trips in a particular month - March and April marking the highest. January being lowest maybe due to SnowFall.

From February, we can see trip duration rising every month.

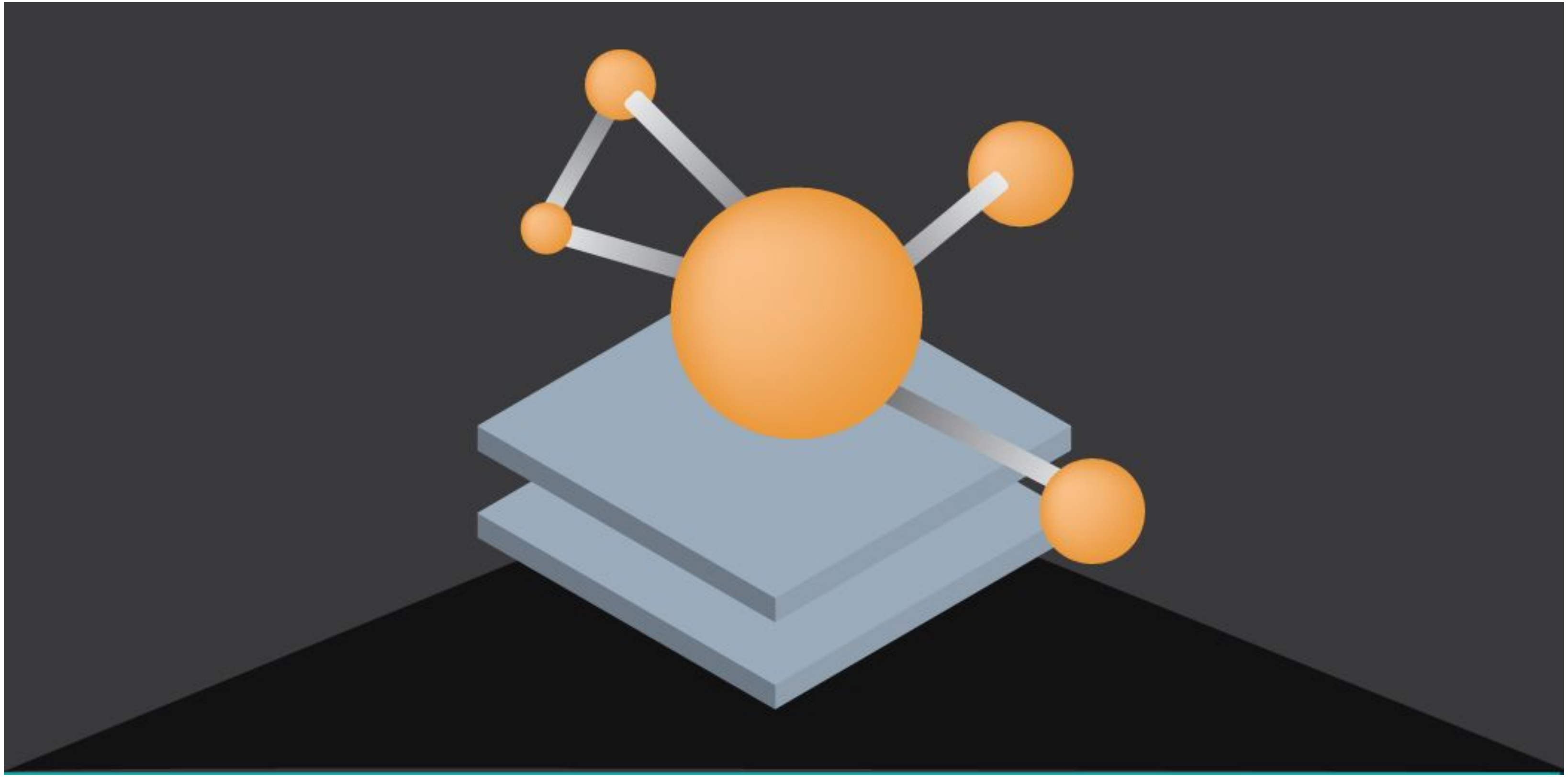
# Analysis on : Correlation Heatmap



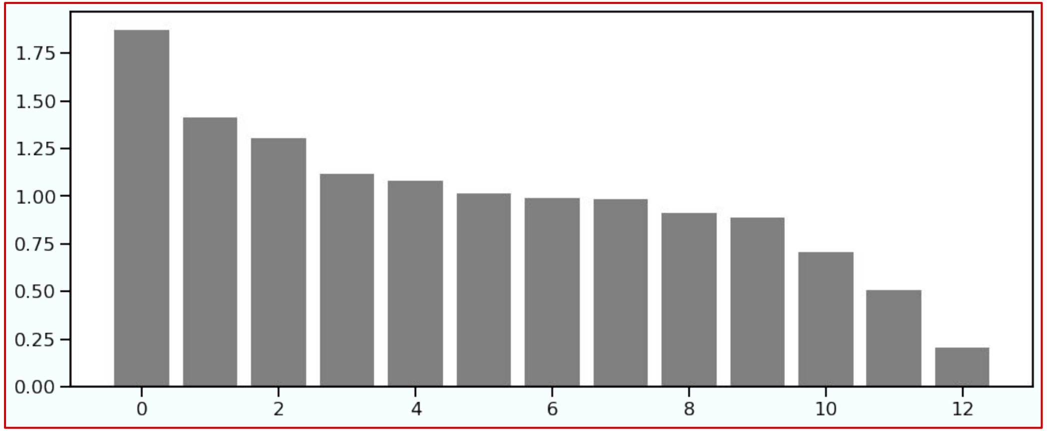
Correlation Plot

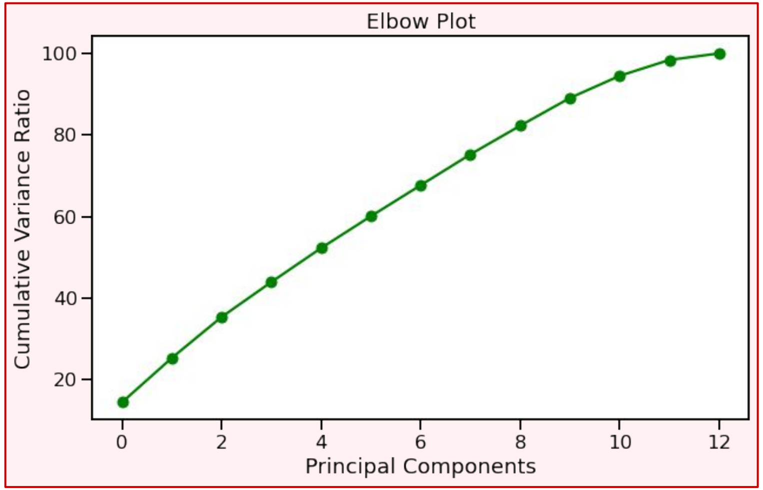
| | vendor_id | passenger_count | pickup_longitude | pickup_latitude | dropoff_longitude | dropoff_latitude | store_and_fwd_flag | trip_duration | pickup_month | pickup_hour | pickup_weekday | pickup_day | pickup_min | dropoff_min | distance |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| vendor_id | 1e+02 | 29 | 0.78 | 0.17 | 0.15 | 0.45 | -8 | 3.6 | -0.62 | 0.93 | 0.13 | 0.072 | -0.039 | -0.024 | 0.81 |
| passenger_count | 29 | 1e+02 | 0.22 | -0.51 | -0.035 | -0.28 | -2.2 | 1.5 | -0.22 | 0.91 | 2.5 | 0.2 | 0.097 | -0.0048 | 1 |
| pickup_longitude | 0.78 | 0.22 | 1e+02 | 2.3 | 78 | 10 | 1.1 | 4 | 0.45 | 1 | -1.6 | -0.088 | -0.26 | -0.14 | 26 |
| pickup_latitude | 0.17 | -0.51 | 2.3 | 1e+02 | 11 | 49 | -0.83 | -4.4 | -0.14 | 1.1 | -2.9 | -0.65 | 0.14 | 0.1 | -21 |
| dropoff_longitude | 0.15 | -0.035 | 78 | 11 | 1e+02 | 12 | 0.82 | 2.4 | 0.47 | -2.2 | -0.11 | -0.046 | -0.26 | -0.2 | 13 |
| dropoff_latitude | 0.45 | -0.28 | 10 | 49 | 12 | 1e+02 | -0.99 | -3.4 | -0.061 | 1.4 | -2.2 | -0.56 | 0.2 | 0.053 | -14 |
| store_and_fwd_flag | -8 | -2.2 | 1.1 | -0.83 | 0.82 | -0.99 | 1e+02 | 0.3 | 0.038 | 0.23 | -0.4 | 0.11 | 0.02 | 0.027 | 2.6 |
| trip_duration | 3.6 | 1.5 | 4 | -4.4 | 2.4 | -3.4 | 0.3 | 1e+02 | 1.3 | 0.6 | -0.14 | 0.24 | -0.41 | -0.5 | 15 |
| pickup_month | -0.62 | -0.22 | 0.45 | -0.14 | 0.47 | -0.061 | 0.038 | 1.3 | 1e+02 | -0.34 | -1.5 | -1.5 | -0.11 | -0.038 | 1.6 |
| pickup_hour | 0.93 | 0.91 | 1 | 1.1 | -2.2 | 1.4 | 0.23 | 0.6 | -0.34 | 1e+02 | -8.4 | 0.041 | -0.31 | -0.29 | -1.6 |
| pickup_weekday | 0.13 | 2.5 | -1.6 | -2.9 | -0.11 | -2.2 | -0.4 | -0.14 | -1.5 | -8.4 | 1e+02 | 0.44 | 0.078 | -0.13 | 0.76 |
| pickup_day | 0.072 | 0.2 | -0.088 | -0.65 | -0.046 | -0.56 | 0.11 | 0.24 | -1.5 | 0.041 | 0.44 | 1e+02 | -0.015 | 0.013 | 0.6 |
| pickup_min | -0.039 | 0.097 | -0.26 | 0.14 | -0.26 | 0.2 | 0.02 | -0.41 | -0.11 | -0.31 | 0.078 | -0.015 | 1e+02 | 11 | -0.8 |
| dropoff_min | -0.024 | -0.0048 | -0.14 | 0.1 | -0.2 | 0.053 | 0.027 | -0.5 | -0.038 | -0.29 | -0.13 | 0.013 | 11 | 1e+02 | -0.13 |
| distance | 0.81 | 1 | 26 | -21 | 13 | -14 | 2.6 | 15 | 1.6 | -1.6 | 0.76 | 0.6 | -0.8 | -0.13 | 1e+02 |

# Decomposition of Data : PCA



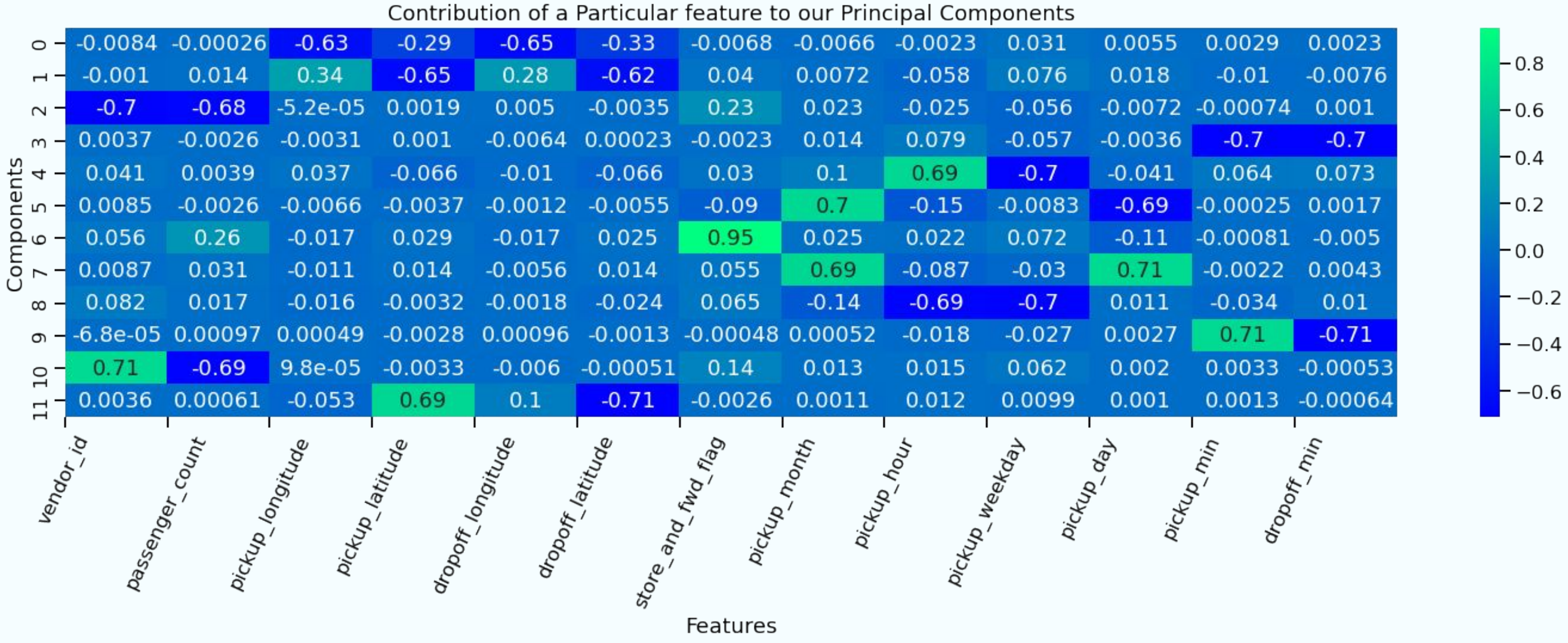
# Analysis on : Principal Component Analysis





**Decomposition - Now that we're done, we have to pass our Scaled Dataframe in PCA model and observe the elbow plot to get better idea of explained variance. At 12th component our PCA model seems to go Flat without explaining much of a Variance.**

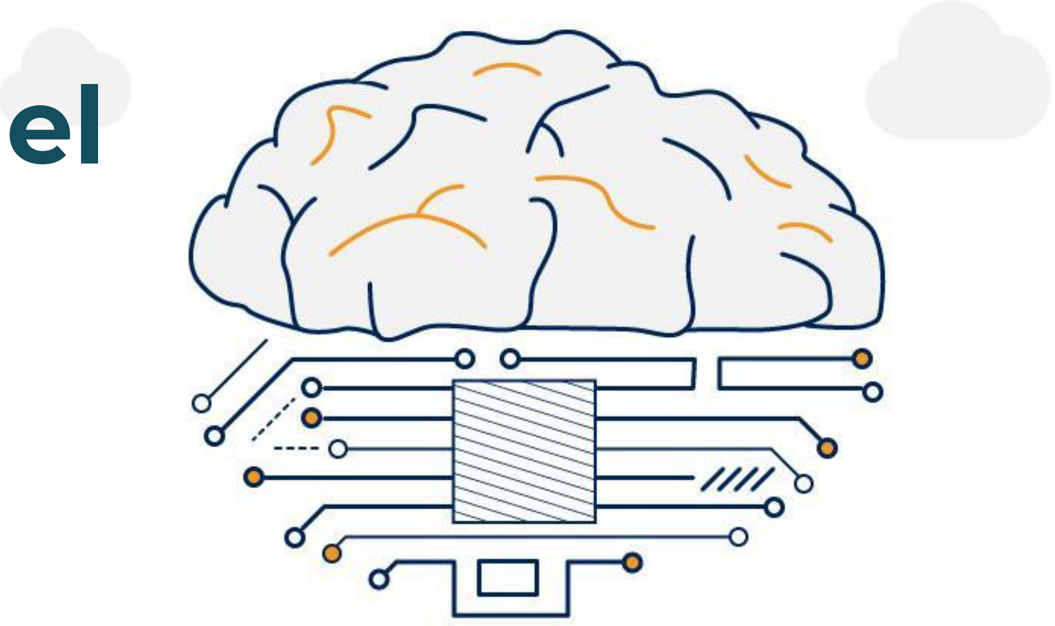
# Analysis on : Feature Contribution



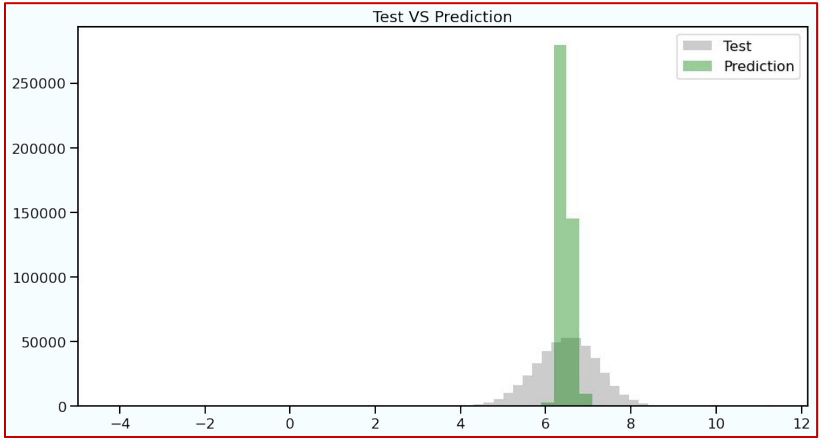
Contribution of a Particular feature to our Principal Components

| Components | vendor_id | passenger_count | pickup_longitude | pickup_latitude | dropoff_longitude | dropoff_latitude | store_and_fwd_flag | pickup_month | pickup_hour | pickup_weekday | pickup_day | pickup_min | dropoff_min |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | -0.0084 | -0.00026 | -0.63 | -0.29 | -0.65 | -0.33 | -0.0068 | -0.0066 | -0.0023 | 0.031 | 0.0055 | 0.0029 | 0.0023 |
| 1 | -0.001 | 0.014 | 0.34 | -0.65 | 0.28 | -0.62 | 0.04 | 0.0072 | -0.058 | 0.076 | 0.018 | -0.01 | -0.0076 |
| 2 | -0.7 | -0.68 | -5.2e-05 | 0.0019 | 0.005 | -0.0035 | 0.23 | 0.023 | -0.025 | -0.056 | -0.0072 | -0.00074 | 0.001 |
| 3 | 0.0037 | -0.0026 | -0.0031 | 0.001 | -0.0064 | 0.00023 | -0.0023 | 0.014 | 0.079 | -0.057 | -0.0036 | -0.7 | -0.7 |
| 4 | 0.041 | 0.0039 | 0.037 | -0.066 | -0.01 | -0.066 | 0.03 | 0.1 | 0.69 | -0.7 | -0.041 | 0.064 | 0.073 |
| 5 | 0.0085 | -0.0026 | -0.0066 | -0.0037 | -0.0012 | -0.0055 | -0.09 | 0.7 | -0.15 | -0.0083 | -0.69 | -0.00025 | 0.0017 |
| 6 | 0.056 | 0.26 | -0.017 | 0.029 | -0.017 | 0.025 | 0.95 | 0.025 | 0.022 | 0.072 | -0.11 | -0.00081 | -0.005 |
| 7 | 0.0087 | 0.031 | -0.011 | 0.014 | -0.0056 | 0.014 | 0.055 | 0.69 | -0.087 | -0.03 | 0.71 | -0.0022 | 0.0043 |
| 8 | 0.082 | 0.017 | -0.016 | -0.0032 | -0.0018 | -0.024 | 0.065 | -0.14 | -0.69 | -0.7 | 0.011 | -0.034 | 0.01 |
| 9 | -6.8e-05 | 0.00097 | 0.00049 | -0.0028 | 0.00096 | -0.0013 | -0.00048 | 0.00052 | -0.018 | -0.027 | 0.0027 | 0.71 | -0.71 |
| 10 | 0.71 | -0.69 | 9.8e-05 | -0.0033 | -0.006 | -0.00051 | 0.14 | 0.013 | 0.015 | 0.062 | 0.002 | 0.0033 | -0.00053 |
| 11 | 0.0036 | 0.00061 | -0.053 | 0.69 | 0.1 | -0.71 | -0.0026 | 0.0011 | 0.012 | 0.0099 | 0.001 | 0.0013 | -0.00064 |

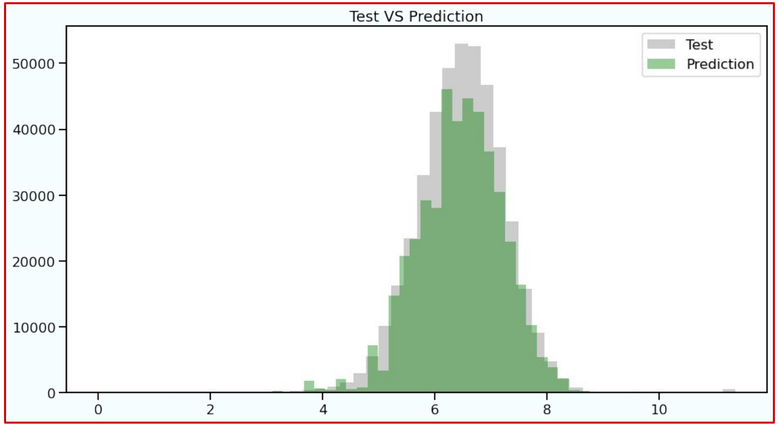
# Machine Learning Model Evaluation & Result : Regression



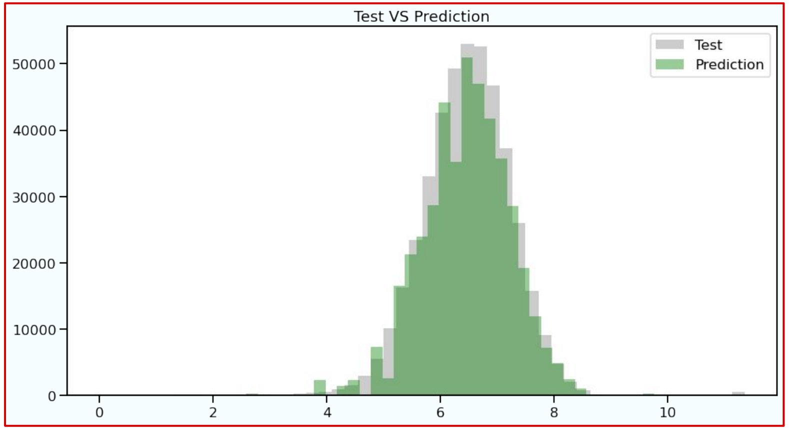
# Analysis on : ML Model Prediction with PCA



Linear Regression



Decision Tree



Random Forest

**Visualizations show us How our model's Predictions are close to Test Data. It is evident that Decision Tree and Random Forest are Performing well.**

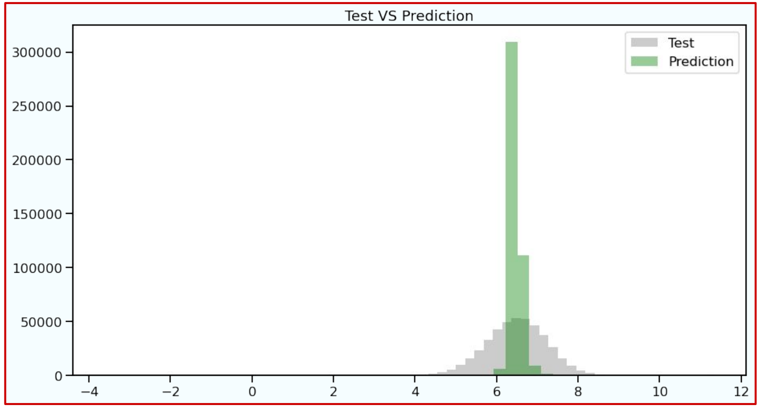
# Analysis on : Model Evaluation Result with PCA

- We can clearly observe that our Decision Tree model and Random Forest model are  good performers.
- As, Random Forest is providing us reduced RMSLE, we can say that it's a model to Opted for.
- We're getting good fit score for Decision Tree and Random Forest , i.e, close to 1.0

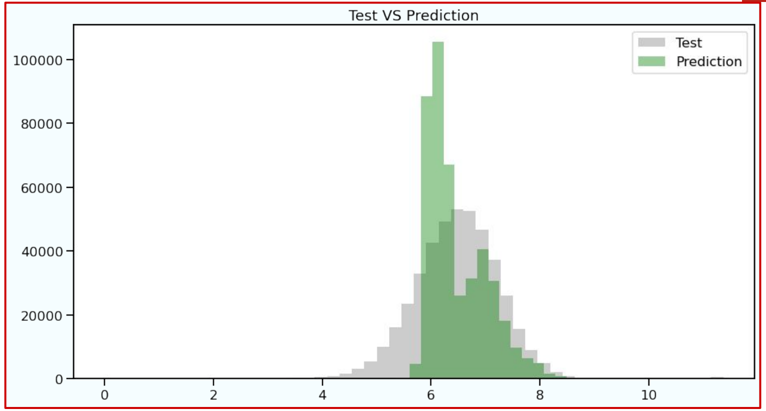
| Algorithms | Training Score | Validation Score | Cross Validation Score | R2-Score | RMSLE |
|---|---|---|---|---|---|
| Linear Regression | 0.0389 | 0.0509 | 0.0329 | -34.90 | - |
| Decision Tree | 0.9238 | 0.9149 | 0.9161 | 0.9076 | 0.038 |
| Random Forest | 0.9329 | 0.9260 | 0.9241 | 0.9191 | 0.036 |

- R2-score: Usually must be between 0 and 1, towards 1 considered as good fit.
- RMSLE: Lesser is Better

# Analysis on : ML Model Prediction without PCA

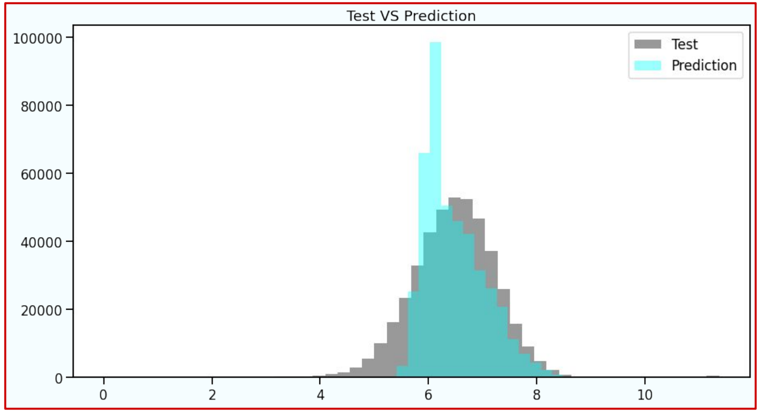


Linear Regression

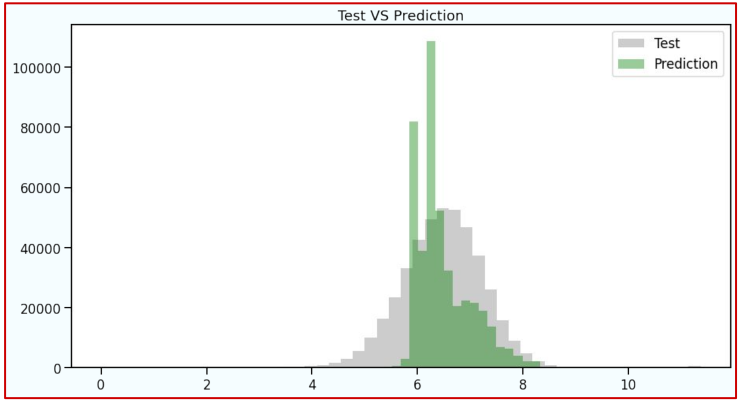


Decision Tree

Test VS Prediction, How close they are ?



Decision Tree with GridsearchCV



Random Forest

# Analysis on : Model Evaluation Result without PCA

- We can clearly observe that our **Decision Tree with GridsearchCV** model are good performers. As, It is providing us reduced RMSLE, we can say that it's a model to Opt for.
- We're getting good fit score for **Decision Tree with GridsearchCV** , i.e, close to 1.0

| Algorithms | Training Score | Validation Score | Cross Validation Score | R2-Score | RMSLE |
|---|---|---|---|---|---|
| Linear Regression | 0.0401 | 0.0517 | 0.0342 | -32.90 | - |
| Decision Tree | 0.4649 | 0.4555 | 0.4550 | -0.177 | 0.0885 |
| Decision Tree with GridsearchCV | 0.5135 | 0.4952 | - | 0.0149 | 0.0857 |
| Random Forest | 0.4803 | 0.4720 | 0.4692 | -0.231 | 0.0874 |

- **R2-score: Usually must be between 0 and 1, towards 1 considered as good fit.**
- **RMSLE: Lesser is Better**

# Conclusion :

- Trip Duration varies a lot ranging from few seconds to more than 20 hours also some are going from 528 Hours to 972 Hours, possibly Outliers.
- Observed Vendor 2 taxi service provider is most Frequently used by New Yorkers.
- Trip duration is generally longer for trips whose flag was not stored.
- There were few trips with Zero Passengers and few trips with 7,8 and 9 passengers and Most number of trips are done by single or double passengers.
- Few Trip duration has covered 0 Km distance.
- Trip duration is the maximum around 3 pm and the lowest around 6 am.
- Trip duration is the longest on Thursdays closely followed by Fridays.
- From February, we can see trip duration rising every month also significant drop in the Taxi trip count as month end approach.

- One problem that might occur with Decision Tree is that it can overfit.
- A decision tree model considers all the features which makes it memorize everything, it gets overfitted on training data which couldn't predict well on unseen data.
- A random forest chooses few number of rows at random and interprets results from all the Trees and combines it to get more accurate and stable final result.
- If not reducing dimension we can use Hyperparameter tuning but it take time and system resource.

# Recommendations :

## # Recommended Approach :

- Apply Standard Scaling on the Dataset to Normalize the values.
- Further, Apply PCA to reduce dimensions, as you'll extract features from our primary DateTime Feature. Those additional features might lead our model to suffer from Curse of dimensionality and could drastically affect performance.
- Pass the PCA Transformed data in our ML Regression Algorithms and Evaluate results.

## # Also we can try this Approach :

- We can use IQR outliers finding and removing technique and then apply hyperparameter tuning technique it will give us best result.
- We can perform hyper tuning on our Algorithm to get the most out of it but Hyper Tuning consume lot of time and resources of the system depending upon the how big the Data we have and what algorithm we're using. It will go through number of Iterations and try to come up with the best possible value for us.

# Q & A