Abstract

New York yellow taxis have significantly higher demand in the business areas of Manhattan compared to other boroughs of the city. This project explored the alternative transport patterns of EVs and E-bikes in terms of achieving the ultimate carbon neutrality goal by reducing the amount of the city's greenhouse emissions (GHG) which mainly result from the high demand of fossil fuels in traditional taxicabs on the road. We also conducted several models such as K-means clustering, Linear SVR and more in python on the perspective of predicting specific demands changing with various factors. Visualisations related to demand, distance and trips were developed for the taxis along with the visualisations for e-bike usage and electric vehicle sales. In the end, the study provided a series of possible suggestions for NYC's ongoing climate-changed initiatives.

Table of Contents

Table of Figures

1. Introduction

In recent years, the numerous challenges brought by climate change are currently becoming an extensive issue for human beings in all regions. In particular, New York City, one of the most popular global cities, is also facing problems with the constantly rising sea levels caused by the increasing temperature and precipitation as a coastal city on the world map. There's no doubt that resolving and fixing these urgent problems is a challenge for all New Yorkers. According to the Mayor's Office of Climate and Sustainability report [1] shows, over 1,000 deaths happen in NYC each year because of poor air quality, and nearly 30% of the Vehicle's main engines are still highly relying on fossil fuels that is the main contributor of the city's greenhouse emission (GHG), the city's second-largest source of GHG emissions, after buildings. On top of that, people believe electric vehicles (EVs) could be the alternative effective and direct ways of dealing with these climate issues and then achieving the carbon neutrality goal which is announced by The New York City Department of Transportation (NYC DOT) committing to reduce GHG emissions by 80% by 2050 [1][4]. A series of unexpected blocks are still on the way, even though the city's EVs movements are already marching on.

The one moving unique icon in NYC, Yellow Taxicabs has always been seen as a key role in New Yorkers' daily commuting due to its attribute that has the right to pick up street-hailing and prearranged passengers anywhere in New York City with the massive amount of 13,587 taxis [5][6]. However, when it comes to the environmental side, the average New York taxi emits more than 100,000 pounds of carbon dioxide each year as Environmental Protection Agency's carbon calculator declared [13]. More specifically, the total fleet emits nearly 580,000 metric tons which are nearly 800 pounds for each resident of Manhattan and the equivalent of what 500,000 acres of pine forests would store in carbon for one year [11]. The biggest reason why this certain type of taxis owing high demands is that they are the only way of transport where commuters are likely to be forced to travel using these taxis even for shorter distances. We believe that the demand for these taxis is affected by several factors such as pick-up and drop-off times, the precise location of the pick-ups and drop-offs, fares, weekends, holidays, time of the year, festivals, etc. Additionally, the weather could also be a crucial reason for a commuter to take a taxi. Hence, a demand prediction system which might be the remedy alternate for this pattern in terms of an eco-friendly transport chain or to encourage carpooling for increased utilization of the cabs and to minimize single passenger trips.

With that in mind, our project is aiming to find out those factors affecting the demand for yellow taxis in NYC by analysing and creating specific models based on the targeted datasets including each EV's brands, affordability, range, efficiency, batteries, and overall appeal, etc. thereby proposing more economical alternate transport ways for New Yorkers. This allows us to identify the best eco-friendly transportation system that fits the lifestyle of New Yorkers and allow them to minimize their carbon footprint.

2. Background

Regulatory reform team from Harvard has conducted a case study on the issue of New York Yellow Taxis [14]. In their case study, they inform that the majority of New Yorkers rely on public transit or limousine services; just 22% of Manhattan residents possess a car, compared to 91 percent of families nationwide that own at least one automobile. New York City's cab and livery system is the country's fourth-largest transportation provider. The New York City Taxi and Limousine Commission (TLC), which regulates yellow taxis, for-hire cars, commuter vans, para transit vehicles, and select limos, oversees the system. Despite the taxi and limousine network's size, the previous system of yellow cabs and for-hire cars did not effectively service all of New York City's boroughs.

Jen Roberton et al. in Emissions from the Taxi and For-Hire Vehicle Transportation Sector [13] in New York City state that despite an overall increase in MPG efficiency across the fleet and a decrease in per vehicle weekly mileage, the overall growth of the fleet, combined with the proliferation of vehicle registrations associated with HVFHSs, has resulted in a significant increase in GHG emissions across TLC-regulated industries. Between 2010 and 2018, total emissions from TLC-licensed vehicles climbed by 66%. For a variety of reasons, emissions climbed at a slower pace than total miles driven, which increased by 129 percent over the same time period, including an improvement in both overall fuel economy and the percentage of hybrid cars. They believe that Electric vehicles and Hybrid vehicles are a perfect replacement to reduce the emissions and to improve the air quality in New York City.

In their research of analysing the feasibility of battery powered vehicles in New York, Liang Hu et al. [14] have stated that only 8% taxis can run throughout the day without the need for charging with an average travel distance of 300 miles per day. They argue that switching to BEV can lead to a change in the operations patterns of taxis including trip distance, shift times etc. They conclude that the existing infrastructure cannot support electric vehicles, as there are

few electric charge stations for taxi drivers to use. Therefore, they suggest that it is necessary to build at least 300 fast charge stations in NYC to support the transition from traditional cars to battery powered electric vehicles.

While these studies assess the overall demand for cabs and their impact on the environment, they don't identify the factors that impact the demand or analyse the neighbourhoods that have high demand. Given the increasing demand for the yellow taxis, it is necessary to analyse the factors that contribute to the demand. In our research, we want to address the following questions:

Q1. What are the factors that impact the demand for yellow taxis in NYC?

Q1.1 Which boroughs have the highest demand for taxis in NYC?

Q1.2 Is there a significant change in demand between weekdays and weekends?

Q1.3 Does weather affect the demand for taxis?

Q2. What are the best sustainable alternatives for traditional cabs in NYC?

Q2.1 Are EVs a good alternative? What are the current sales of EVs in New York?

Q2.2 Is there any other service that can replace taxis in NYC?

This allows us to explore the taxi demand throughout the city to identify neighbourhoods with varying demands, to identify demand patterns for a time period, and to identify peak hours of demand of a day. Furthermore, it is equally important to understand the trends related to sustainable ways of transportation. These factors enable the business to target and transform the existing transportation with appropriate measure and to obtain successful results.

3. Analytical Methodology

3.1 Dataset Description

After exploring series of datasets on websites such as Kaggle, data.world and nyc.gov ,etc., we've identified that the following 4 high-related datasets are suitable for our further studies.

The description of each dataset is provided below:

1. **NYC Taxi dataset:**
   The dataset contains all the trip records from Jan 2016 to Jun 2016 of New York Yellow Cabs. This data was originally published by NYC Taxi and Limousine Commission (TLC).

The dataset contains approximately 1.45 million trip records spanning for over 6 months for the entire New York city.

Data field	Description
ID	Unique identifier of each trip
Vendor_id	A code indicating provider of the trip
Pickup_datetime	Date and time when the meter was engaged
Dropoff_datetime	Date and time when the meter was disengaged
Passanger_count	Number of passengers in the vehicle
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_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 seconds

Outlier Removal

The dataset was filtered to exclude all the rows that contain missing values. Finally, a box plot was plotted to inspect for outliers in trip durations and trip times.

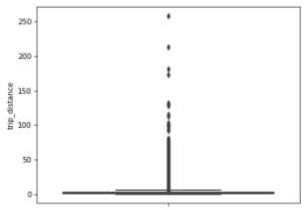

Figure 1: Box plot for trip duration

Feature Engineering

Additional data fields were engineered from the existing data fields to improve the analysis and model training

Data Field	Description
Day name	Name of the day when pickup happened
Weekday or weekend	whether the pickup day was a weekday or weekend
Pickup hour	hour of the pickup
Day period	based on the pickup hour, broadly classified as morning, evening, afternoon and night

Based on the longitude and latitude of the pickup and drop-off, with the help of Bing Map API we were able to get the address and zip code for all the pickups and drop-offs. The pickups and drop-offs were segregated into 5 boroughs of the New York City. The information was further used to extract the travel distance and duration between pickup location and drop-off location.

Data Field	Description
Pickup_zipcode	Zip code of the pickup location
Dropoff_zipcode	Zip code of the drop-off location
Trip Distance	Distance of the most efficient route from source to destination
Trip Duration	Duration from source to destination during traffic

Since Manhattan records nearly 75% of all the trips, we selected it as the primary region of focus for the analysis. Therefore, we filtered the dataset to include pickups only from Manhattan. We did this by writing a Python script that identifies all the zip codes that are part of Manhattan and filters the dataset based on these zip codes.

Clustering of Regions using location data with unsupervised learning:

The cleaned location data was introduced into K-means clustering to group locations based on the pick-up location. To make sure the clusters are of reasonable area, it was determined to form clusters based on a criterion in which the min inter cluster haversine distance between the cluster centers in 0.5 miles with maximum being 2.

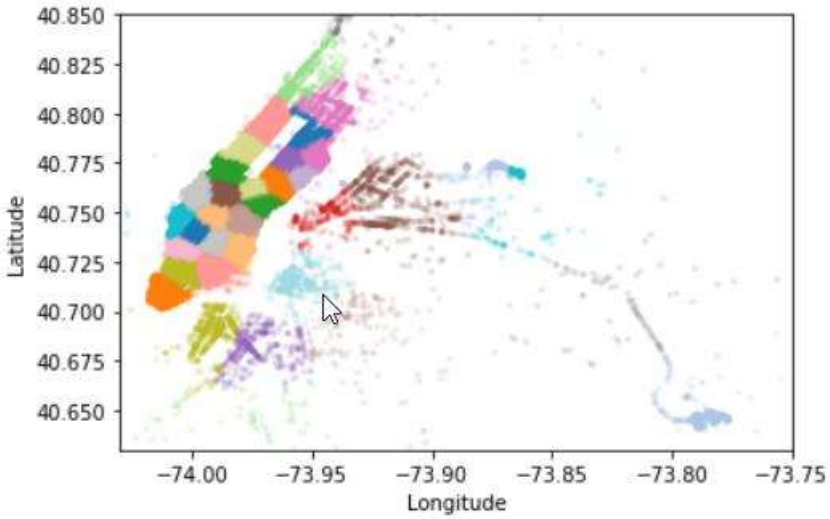

Figure 2: Clustering of locations based on the pickup latitude and longitude

Therefore, the clusters formed are intended to be used during the model building phase for better prediction of trip duration.

2. **NYC Weather dataset:**

Weather data for the New York city from Jan 2016 to Jun 2016 was download from New York weather service. This data can be used to analyse the weather patterns in NYC and to analyse the effects of weather on taxi demand. Below is the description of the data:

Data Field	**Description**
Time	Date and time of reading
Temp	Air temperature
Wdsp	Wind speed
Dewpt	Dew point temperature
Vism	Measured visibility
Pressure	Current pressure
Icon	Indicates the current weather. Ex: Sunny
Rain	Rain in millimeters
Snow	Snowfall in millimeters
Tornado	True if there's an active tornado

3. CitiBike Dataset

CitiBike is an e-bike sharing company in New York city that operates in Manhattan, Brooklyn, and Queens. We have extracted rides of each month from Jan 2016 to Dec 2020 from CitiBike's official website.

Data Field	Description
TripDuration	Duration of the trip
StartTime	Start time of the trip
StopTime	Trip end time
StartStationId	Unique Id of the start station
StartStationTime	Name of the start station
StartStationLatitude	Latitude of the start station
StartStationLongitude	Longitude of the start station
EndStationId	Unique Id of the end station
EndStationTime	Name of the end station
EndStationLatitude	Latitude of the end station
EndStationLongitude	Longitude of the end station
BikeId	Unique Id of the bike
UserType	Type of user (subscriber or non-subscriber)
BirthYear	Year of birth of the rider
Gender	Gender of the rider

The entire dataset contained around 50 million records. As this is a massive dataset, data was aggregated from each month and a new data file was generated to include the cumulative number of trips, average distance travelled each month etc. using Python.

4. NYC EV Registration Dataset

The electric vehicle registration data was obtained from Atlas EV Hub for the New York city from 2010 to 2021. The main objective of using this dataset is to analyze people's opinion towards sustainable means of transportation and to analyse the trends of EV sales for certain vehicles. Dataset was later filtered to include data from Manhattan region only using the registration zip code as filter criteria. The description of the dataset is as follows:

Data field	Description
Zip Code	Zip code of registration
Registration valid date	Date the vehicle was registered on
VIN Prefix	Vehicle Index prefix
DMV ID	DMV ID provided by the NYC
VIN Model Year	Specifies the vehicle model and year
Registration expiration date	Date of expiration of vehicle registration
State	State in which the vehicle was registered
Vehicle Name	Name of the vehicle
Technology	Type of technology used by the vehicle

3.2 Tools Used

Tableau: Tableau is a data visualization and business intelligence tool primarily used by analysts to generate interactive dashboards, visualize data, and obtain key insights from the data. We have mainly used tableau for exploratory data analysis to discover patterns in taxi demand, to identify the neighbourhoods that have the greatest number of pickups and more. Additionally, we created interactive dashboards that could help us in filtering the visualisations for a specific area or day.

Jupyter Notebook: It is a free software for interactive computing in about 40+ programming languages. This web application helps to create and share computational documents. We have used this tool to work on several machine learning models and to evaluate their performance and accuracy.

Visual Studio Code: VS Code is a code editing software designed to run code that can be debugged. To obtain address from the pickup location co-ordinates, we needed to run an iterative python script. VS Code was useful because of its debugging features that enabled us to look through the output JSON string received from the Map API and to properly parse the string to obtain necessary data.

Python: Python is essential for a data analysis. Python includes several libraries that are helpful for every stage of analytical lifecycle. We have used Pandas and NumPy libraries for data modelling and data cleaning, as they support large multi-dimensional arrays with mathematical functions. We have use Matplotlib library to visualise the data. Additionally, we used sklearn

for machine learning modelling in Python. Data wrangling in Python helped us combine the multiple data sets for data preparation and easy analysis.

Bing Maps API: This is a web mapping service by Microsoft. We can search topographically-shaded street maps for analysis of our data. It has open-source APIs that were used to get address information for pick-up and drop-offs.

3.3 Techniques

Linear SVR

This algorithm is used to solve regression problems. It uses the 'Linear Kernel Method' and produces good results with huge datasets. We were able to fit and predict our regression data using Sklearn class in Python and trained the model. We adopted this technique as it allows us the flexibility of defining error acceptance for our predictive model that finds a hyperplane to fit the dataset in hand.

K-means Clustering

This is method used to cluster the data into segments based on the observations where in each cluster, the observation is included based on the nearest mean or cluster centre. It is an unsupervised algorithm. Using the concept of cluster centre/centroids, it clusters data points to its nearest mean, post which the centroid is moved to the average of all data points associated until there are no data point changes to associated cluster centre.

4. Findings of the Analysis

4.1 Descriptive Analysis

A heatmap was generated based on the zip code of each pickup for Manhattan region. We can observe that most pickups have happened at the southern part of Manhattan, in particular, zip codes 10003 and 10011 with 40101 and 31200 pickups respectively. Zip codes 10021 and 10028 see fairly high number of pickups of, 34000 each, whereas most other southern zip codes see moderate pickups. The northern Manhattan sees minimal demand for taxis, with zip code 10034 being the lowest with 76 pickups.

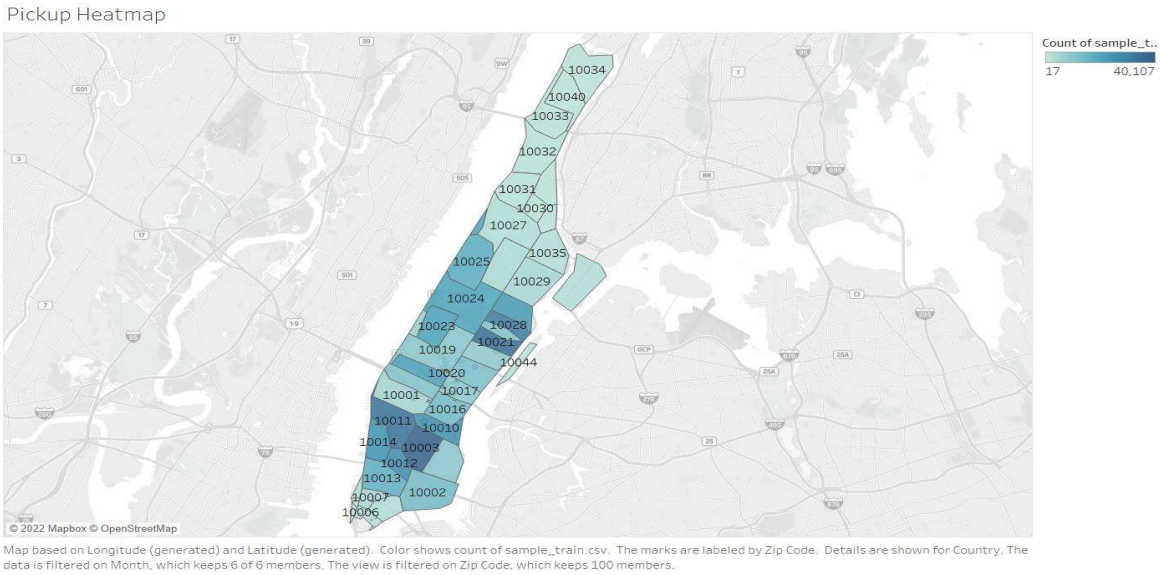

Figure 3: Pick-up heatmap in Manhattan grouped by zip codes

The neighborhood information was feature engineered to analyze pickup and drop-off patterns. The x-axis on the chart represents the pickup neighborhood, Y-axis represents the number of pickups, and the colors represent the destination neighborhood. Most pickups have originated from Upper Westside, Harlem, and East Harlem. It is surprising that these neighborhoods are also the most popular destinations along with Bedford and Kensington.

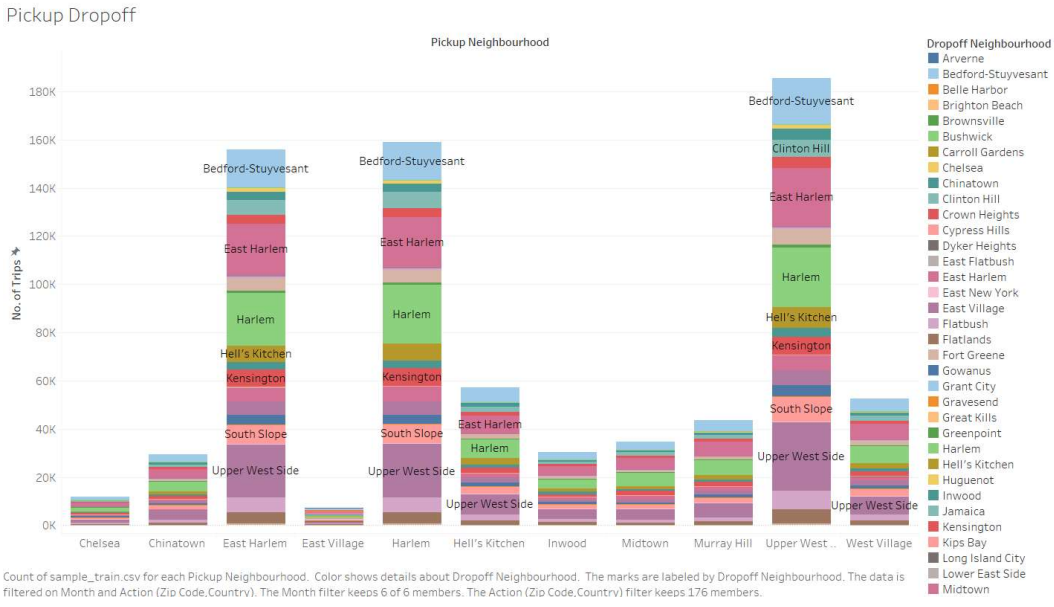

Figure 4: Pickup neighbourhood and respective Dropoff neighbourhood with number of trips

Time series analysis was performed on the entire dataset to analyze the demand of taxi in Manhattan every day for six months. From the time series chart, we understood that the taxi demand follows a specific pattern where Monday's see the least demand for taxis and the

demand gradually grows as the week progresses, hitting the peak demand usually on a Friday or Saturday and falling on Sunday and Monday.

Another interesting fact is that the overall demand for taxis grew from January to April, with highest number of pickups on April 9, 2016 and gently falling from April to June.

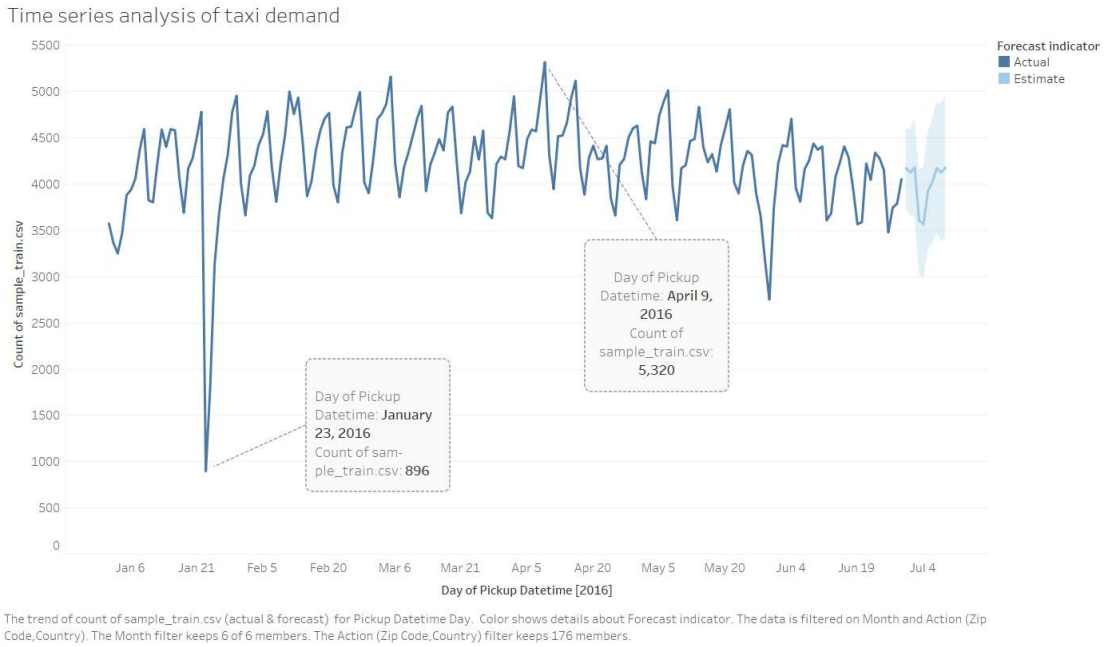

Figure 5: Time series analysis of the trips from Jan 2016 to Jun 2016. Prediction for Jul 2016

On Jan 23, 2016, significantly less pickups of just 896 were recorded. After analyzing the weather, it was identified that New York was buried in 3ft snow after a category 5 blizzard [19].

By using the 6-month data, time series prediction was performed to estimate the demand for the following 10 days. The parameters used for the prediction and the results are provided below.

Count of sample_train.csv

Model			Quality Metrics					Smoothing Coefficients		
Level	Trend	Season	RMSE	MAE	MASE	MAPE	AIC	Alpha	Beta	Gamma
Additive	None	Additive	215	159	0.57	3.9%	1,309	0.500	0.000	0.175

Count of sample_train.csv

Initial	Change From Initial	Seasonal Effect				Contribution		
June 30, 2016	June 30, 2016 – July 9, 2016	High		Low		Trend	Season	Quality
4,174 ± 422	6	July 9, 2016	209	July 4, 2016	-405	0.0%	100.0%	Ok

In order to understand passenger counts for the trips, all trips were grouped based on the recorded passenger. More than 70% of the trips had a passenger count of 1 person, 14% of the trips had 2 passengers, and the remaining trips consisted of 3 to 6 passengers.

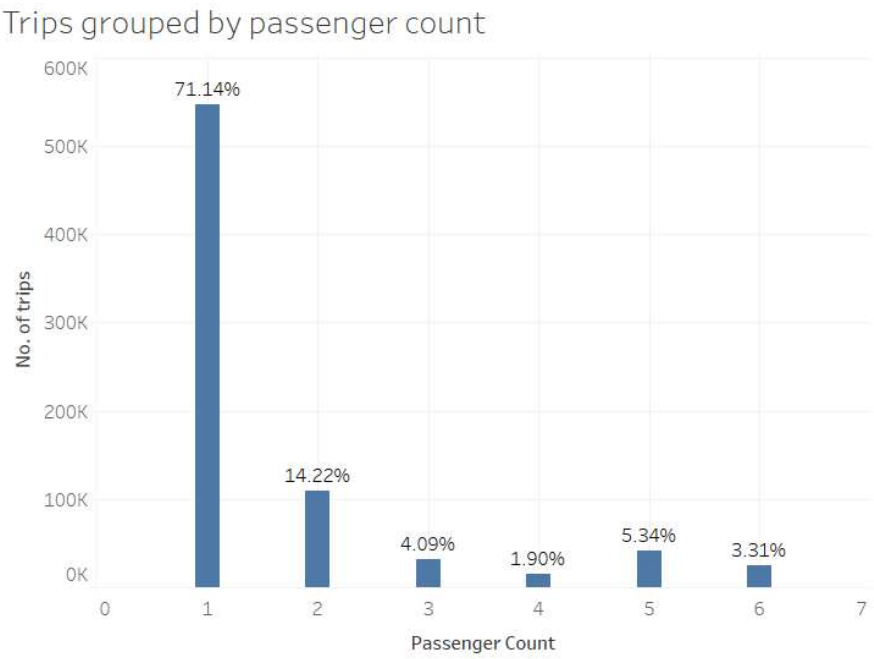

Figure 6: Total trips grouped by passenger count

Taxi demand based on the hour of the day for weekdays and weekends was also visualized. During the weekdays, the demand is at its lowest at 4am and rises sharply during the breakfast time, reaching the local peak at 8am. The pickup rate falls a little and stays consistent throughout the day, reaching the peak at 7pm in the evening and slowly falling as the night progresses.

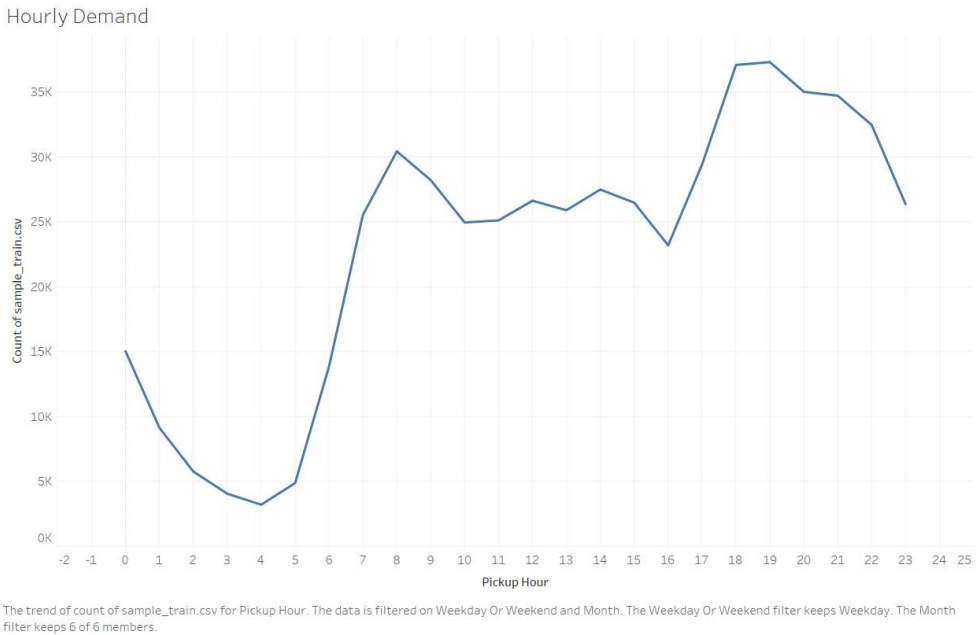

Figure 7: Demand analysis with respect to each hour of the day during weekdays

On the weekends, there is a high demand at 12AM which steeply falls by 5am to the minimum. However, the pickup rates increase in the morning till 11AM and stays consistent reaching the peak at 6pm in the evening.

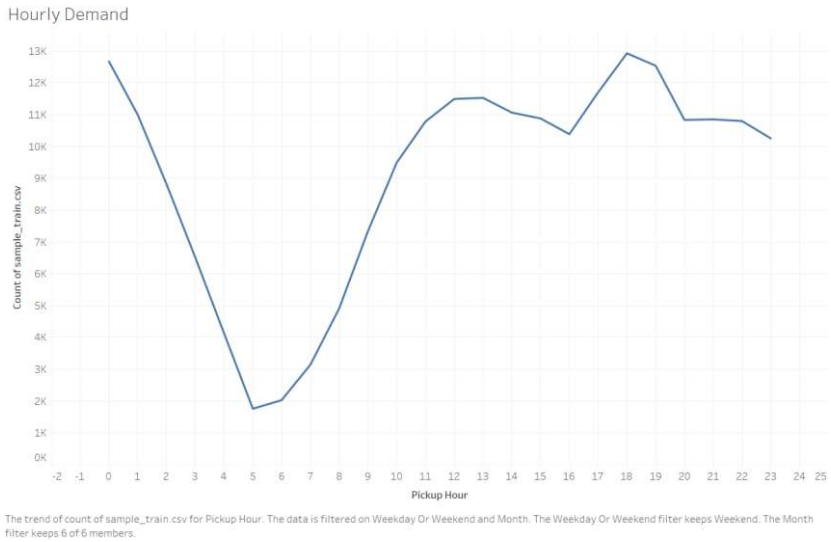

Figure 8: Demand analysis with respect to each hour of the day during weekends

A normal distribution curve was plotted for the actual trip duration and the trip duration for the route suggested by Bing Maps API. On average, the actual duration is significantly higher than the suggested duration. Actual duration has the mean of 6.3 which is around 9 min (EXP (6.3)) and the mean of suggested duration curve is 5.25 (4 mins).

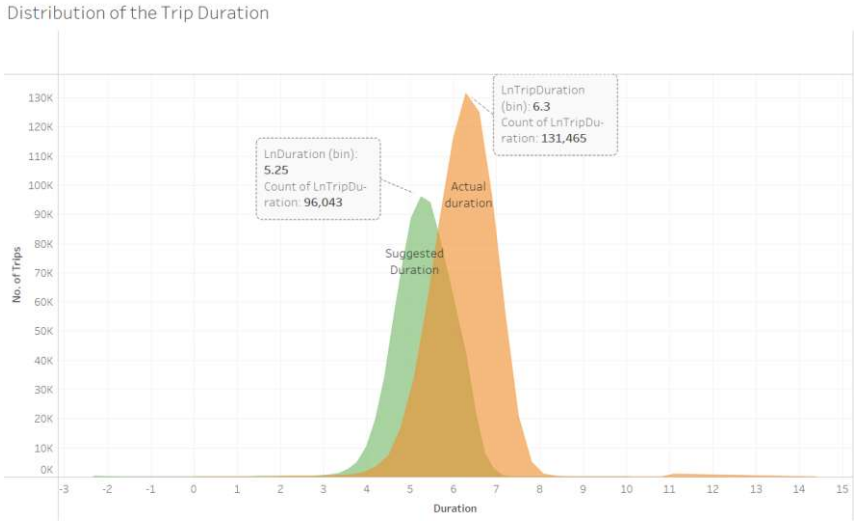

Figure 9: Distribution of actual distribution vs best suggested trip duration

Below bar chart indicates the weather of New York during various instances of measurement. Majority of the time the weather is clear and sunny, sometimes its partly cloudy. We can further

observe that New York sees very less snow or rainy days with very less instances of Heavy Rain or Snow.

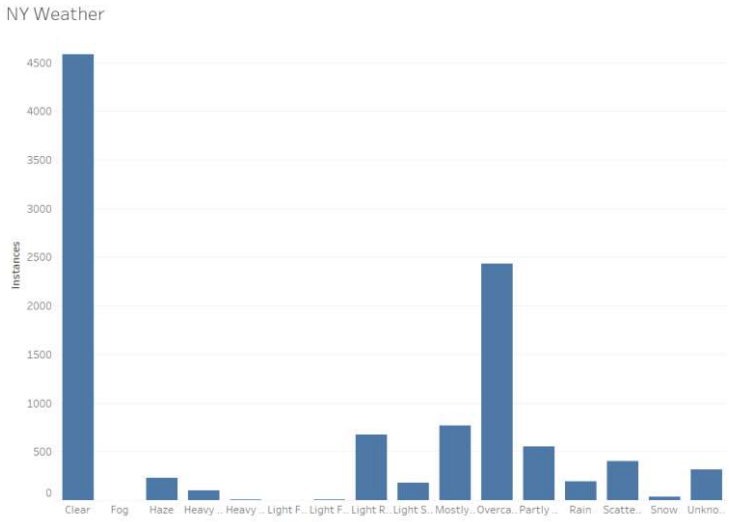

Figure 10: Bar chart depicting various weather events in NYC from Jan 2016 to Jun 2016

From the dashboard of CitiBike trips, heatmap indicates that Midtown and Southern Manhattan are the popular areas for bike riders. The time series analysis shows that the demand during spring and summer is significantly higher compared to winter. Users of age groups from 25 to 40 years contribute to 50% of the riders, with majority of them being males. This is further confirmed by the line graph. Additionally, there is an increased demand during 8am and 6pm.

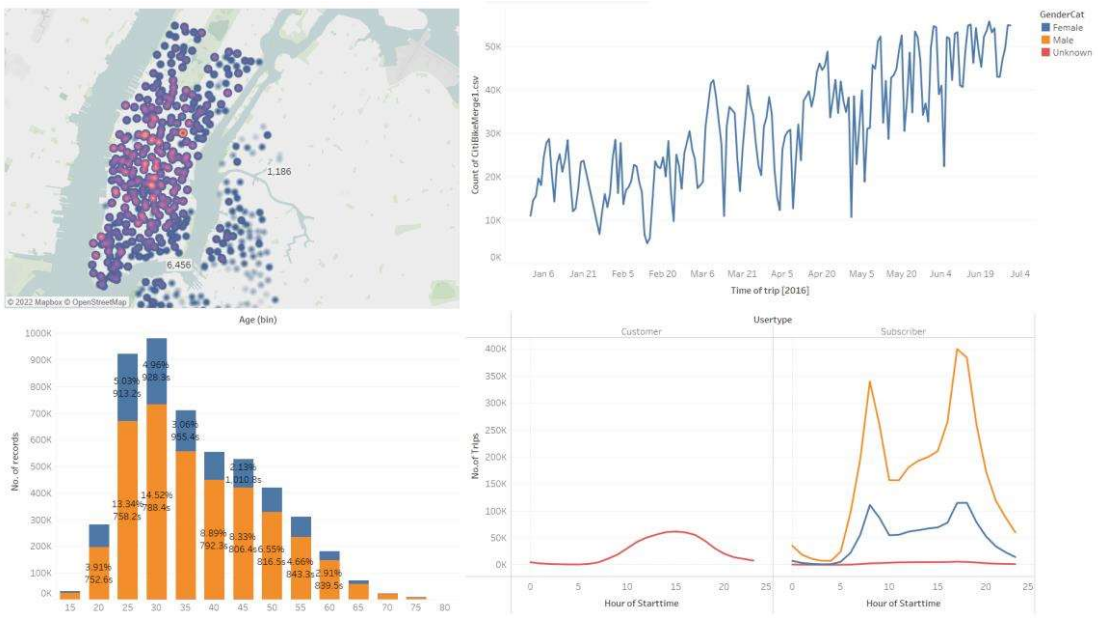

Figure 11: Dashboard of CitiBike trips from Jan 2016 to Jun 2016 (1. Pickup heatmap 2. Timeseries analysis of pickups 3. Distribution of age groups along with gender 4. Demand analysis by gender and subscription type)

Further, analyzing the CitiBike data from 2016 to 2021, it is evident that the demand for e-bikes and bike sharing has grown gradually every year. Additionally, average distance travelled during each year has increased, with customers travelling longer during summers than winters. The same pattern can be noticed in terms of active subscribers each month, where summer months see more subscriber than winter months and the growth has been constantly increasing each year.

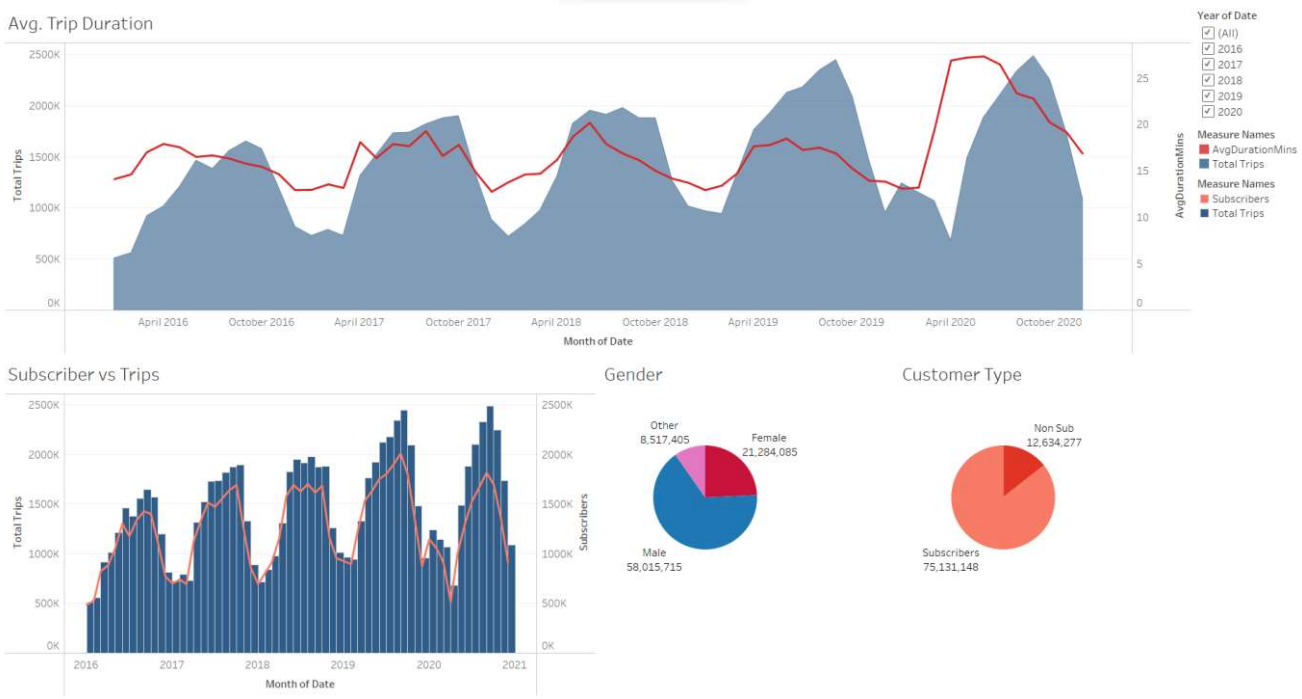

Figure 12: Dashboard of CitiBike trips for the span of 5 years (2016-2021) indicating monthly trips, average distance travelled, number of subscribers, distribution of gender and customer types

The registrations of Electric vehicles were analyzed to understand the trends of EV sales and customers opinion towards EVs and sustainability. The number registration from 2011 – 2013 stay low and see a small rise in 2014. From 2016 the registrations rise on steep curve, peaking at 2019 and falling off in the next couple of years.

The numbers have gone down in the last couple of years mainly because of the covid pandemic and the global chip shortage. Since most of the restrictions are being lifted and with the chip shortage coming to an end, the sales of EVs are predicted to go up.

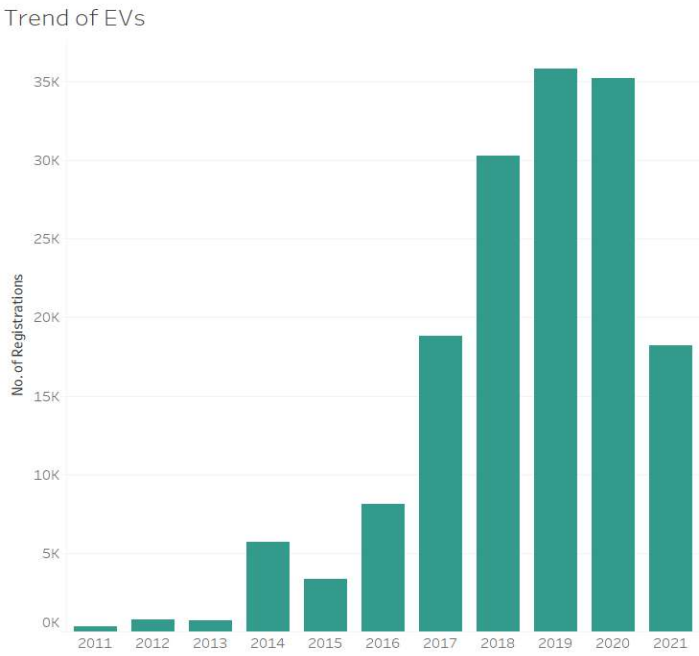

Figure 13: Bar chart of EV registrations in Manhattan, New York from 2011 to 2021

4.2 Predictive Analysis

As part of predictive analytics, Various supervised learning models were built to predict the trip durations given the attributes in the dataset along with the additional features that were engineered during feature engineering.

4.2.1 Feature Selection

Feature selection is the process of reducing the number of input variables when developing a predictive model. It is desirable to reduce the number of input variables to both reduce the computational cost of modelling and, in some cases, to improve the performance of the model.

In this case, the features that would benefit the prediction of trip duration were chosen to build different regression models.

4.2.2 Model building

The dataset split into train and test where 70% of data being train data and remaining 30% being the test data.

Feature Scaling

Feature Scaling is a technique to standardize the independent features present in the data in a fixed range. It is performed during the data pre-processing to handle highly varying magnitudes or values or units. The features obtained are further used to tune and train the models as shown below.

Support Vector Regressor

Support Vector Regression is a supervised learning algorithm that is used to predict discrete values. It uses the same principle as the SVMs. The basic idea behind SVR is to find the best fit line. In SVR, the best fit line is the hyperplane that has the maximum number of points.

In this case, Hyper parameter tuning was performed using RandomSearchCV with various values of C (penalty parameter) such as {0.1, 1, 10}

After the model execution we obtained the optimal root mean square log error of 0.785 for train data and 0.786 for test data.

Decision tree Regressor

Decision tree builds regression or classification models in the form of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed. The final result is a tree with decision nodes and leaf nodes.

In this case, Hyper parameter tuning was performed using GridSearchCV over a range of max_depth:[3,5,6]. The best model was obtained at max_depth=5 and estimators of 500.

After the model execution we obtained the optimal root mean square log error of 0.708 for train data and 0.709 for test data.

Random Forest Regressor

A random forest regressor. A random forest is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting. This model was executed as follows.

After the model execution we obtained the optimal root mean square log error of 0.518 for train data and 0.587 for test data.

Additionally, the timestamp of the pickup_location was binned to 10-minute intervals and Regression models were built to predict the number of likely pickups for a given cluster at certain 10 minute bin. As discussed earlier, Clusters were formed using the pickup_location co-ordinates. The clusters can be viewed on the map as follows.

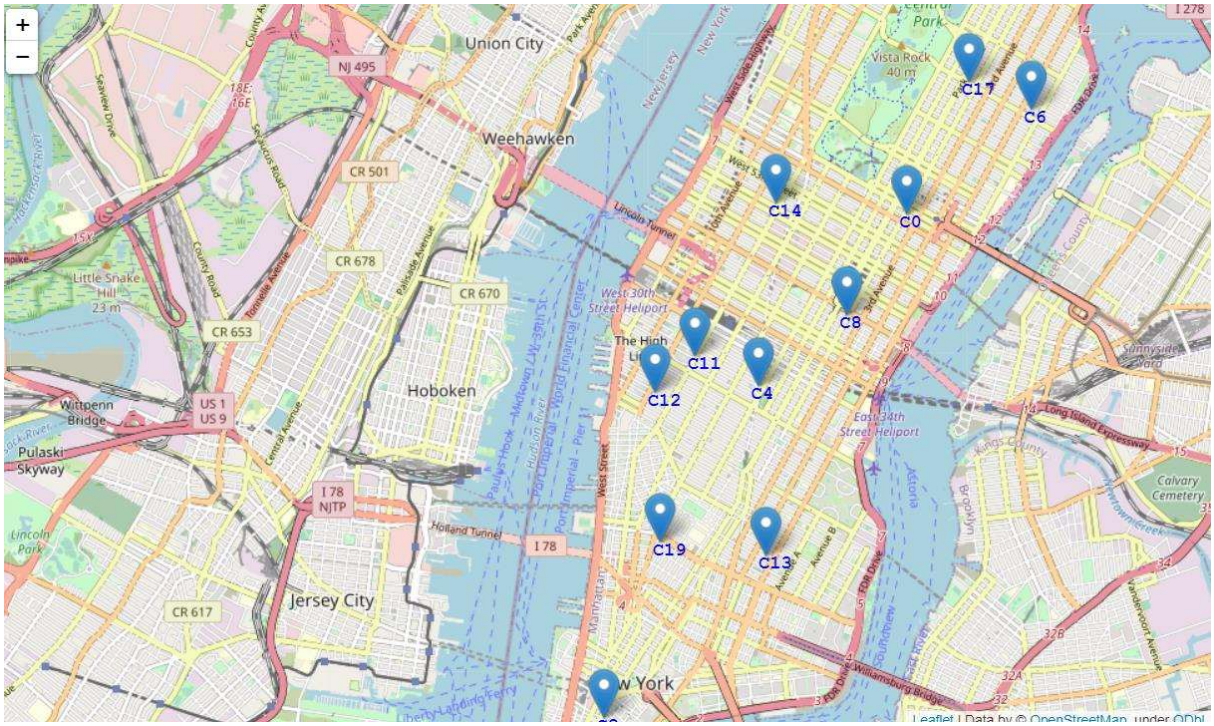

Figure 14: Location based clusters for pickups created using k-means clustering algorithm, visualised using Folium map library

Simple Moving Averages

A simple moving average (SMA) is an arithmetic moving average calculated by adding recent(previous) values and then dividing that figure by the number of time periods in the calculation average. The number of pickups in a cluster was calculated using SMA as follows:

Using these values, a linear regression model and Gradient boost Regressor model was built to predict the count of pickups with Mean absolute percentage error as the evaluation metric.

Linear Regression

Linear Regression is a machine learning algorithm based on supervised learning. It performs a regression task. Regression models a target prediction value based on independent variables. It used to predict the pickup times as follows:

Gradient boosting Regressor

Gradient Descent is an optimization algorithm for finding a local minimum of a differentiable function. Gradient descent is simply used in machine learning to find the values of a function's parameters (coefficients) that minimize a cost function as far as possible.

Summary of Results

After executing various regression models, the best model to predict the trip duration was found to be Random Forest Model with optimal root mean square log error of 0.51 for test data 0.58 for train data followed by Decision Tree Regressor and Linear Support Vector Regressor

Vectorizer	Model	HyperParameter	Test RMSLE	Train RMSLE
OneHot	SVR Linear	1.0	0.787	0.786
OneHot	Decison Trees Regressor	d=10,min-sample:100	0.7	0.71
OneHot	Random Forest Regressor	max_depth=6, max_features=0.5, n_estimators=100	0.51	0.58

Figure 15: RMSLE for each model

Similarly, the best model to predict the number of pickups given a cluster and timestamp was found to be Gradient boost Regressor(XgBoost) with train MAPE being 0.129 and test MAPE of 0.126 followed by Random Forest Regression and Linear Regression.

```
print ("Error Metric Matrix (Tree Based Regression Methods) -  MAPE")
print ("--------------------------------------------------------------------------------------------------------------")
print ("Linear Regression -                        Train: ",train_mape[4],"      Test: ",test_mape[4])
print ("Random Forest Regression -                  Train: ",train_mape[2],"     Test: ",test_mape[2])
print ("XgBoost Regression -                        Train: ",train_mape[3],"      Test: ",test_mape[3])
print ("--------------------------------------------------------------------------------------------------------------")
```

```
Error Metric Matrix (Tree Based Regression Methods) -  MAPE
--------------------------------------------------------------------------------------------------------------

Linear Regression -                        Train:  0.13331572016       Test:  0.129120299401
Random Forest Regression -                  Train:  0.0917619544199      Test:  0.127244647137
XgBoost Regression -                        Train:  0.129387355679       Test:  0.126861699078
--------------------------------------------------------------------------------------------------------------
```

Figure 16: MAPE for each model

5. Discussion

It is crucial for us to analyse various demands in terms of understanding the customer interests about a product or service in a target market. Demand analysis techniques not only can be used to determine when and where a service can be introduced and how it can generate expected results or profit, but also can be used to give a better understanding of the high-demand markets for the company's offerings, using which businesses can determine the viability of investing into such services.

From the analysis of taxi trip data, it is evident that 75% of all the taxi pickups in New York City happen in Manhattan. Additionally, from the pickup heatmap, it is confirmed that nearly 90% of these trips are concentrated to mid and southern areas of Manhattan. Further, looking into the pickup-drop off chart, it is observed that most of these pickups originate from Upper

Westside, Harlem, and East Harlem. Hence, we would suggest on focusing these neighborhoods as an initial target area for the solution to replace traditional cabs with EVs.

We can learn from the trends of EV registrations that people's interests toward electric vehicles is growing every year in NYC. This indicates that people are more concerned about climate change and the carbon footprint of their fossil fuel vehicles than before. In addition, according to US Energy Information Administration the gas prices are on the rise recently and there are no signs of these prices coming down soon. Therefore, by choosing electric vehicles, carbon footprint can be reduced and help fight the climate change. On top of that, our analysis of EV Sales shows that Tesla is the most popular automobile manufacturer. With all 4 models in the top 5, Tesla is undoubtedly the leader in EV market. Coming to specific models, Model X is at the top with overall sales followed by model S, 3 and Y at 2nd, 3rd, and 5th places respectively. By considering the launch dates, Model Y takes the lead in terms of sales as it is less expensive compared to other models. In conclusion, it is better to choose Tesla Model Y as taxis for the business.

From the analysis of hourly pickup rates during weekends and weekdays, we were able to identify the hours that see maximum demand (8am and 7pm for weekdays, 12am and 11am for weekends) and the hours with the least demand (4am on weekdays and 5am on weekends). By using this information along with the demand associated with each neighborhood, resources can be managed efficiently by appropriately directing the cabs during peak demand hours.

On the other hand, New York City has pleasant weather most of the days, with mostly sunny days. This could be seen as a perfect opportunity to introduce another alternative to taxis, which is bike sharing, a popular service helping travelers out while they are in a hurry and sufferable situation. Since these scooters can make use of the dedicated bike lanes, they are much better for users from the perspective of both safety, speed and eco-friendly, comparing with conventional taxis. NYC Department of Transportation has already started an initiative of shared scooters in few neighborhoods named as NYC DOT project, but it is still in development.

From the analysis of CitiBike data, it is evident that there is an increased growth in terms of users every year. In the span of 5 years, there has been an increase of 60% in terms of trips and subscriber count. It is important to note that the demand during winter months is significantly lower than other months. While the users span between the ages 15 to 75, 50% of the riders are aged between 25 and 40, males contribute to 75% of this population. Evidently, e-bikes are

used for shorter commutes by both men and women, with women travelling for longer durations than men.

Since the user base is not diverse, cabs cannot be replaced by bike sharing. However, more than 70% of the taxi trips consisted only 1 passenger. Such passengers can use bike sharing as a service that is available to customers for quick and short travels. Like cabs, bike sharing can be introduced in Mid-town, Manhattan during the initial phase which can be scaled gradually.

Furthermore, with help of predictive analytics using machine learning models, NYC taxi trip durations were predicted using the features such as pickup location, drop-off location, pickup timestamp, drop-off timestamp available in the dataset, along with these features additional features were engineered such as name of day, weekend or not, distance of the trip, speed etc. Additionally, to assist the predictive modelling, based on the location data available, clusters with criterion of having inter cluster distance of 2 miles and intra-cluster area of 0.5 miles were formed (Fig. 2) these clusters were further added as features to improve the predictive accuracy of trip durations. These predicted trip durations can further be used by EV bike platforms to encourage its customers to prefer E-bikes over conventional taxis, showing how delays can be curtailed for short trips with real time trip duration comparison. On the other hand, with an average New York taxi emitting more than 100,000 pounds of carbon dioxide each year and the great majority of commuters preferring to learn sustainable living methods, Customers can be provided with real time data about how each trip would reduce the emission of greenhouse gases, this can be calculated with the time saved and the amount of greenhouse gas emitted per unit time, encouraging each eco-conscious customer to prefer E-bikes and urging a regular customer to become eco-conscious by showing the significance of EV commutation. Consequently, this would also be a great initiative supporting the US Department of Energy's goals of reducing US petroleum imports by one-third by 2025.

With help of clusters created and by creating 10-minute time bins as features using the pickup timestamps available in the dataset along with simple moving averages computed for the number of pickups, the number of likely pickups for a particular cluster at a given time was predicted using various regression models with gradient boost regressor standing out as the best model. This information can be effectively leveraged to assess the demand of Taxis at a certain cluster (Fig 13). It can be crucial for an emerging EV-bike platform to understand the demand and manage the resources accordingly. For instance, if the number of pickups in cluster C1 (Fig13) is predicted to increase during a certain time period, its locality can be equipped

with higher number of EV bike stations allowing customers to easily switch their methods of commutation if it's feasible. Therefore, with the help of real time data analysis and predictive analytics, the certainty of EV-bikes usage and adaptation can be thrived to a great extent.

6. Conclusion

In this project, we've conducted some relevant research with the purpose of figuring out what elements contributed to the high demand for conventional yellow taxicabs and then generate several actionable suggestions to help New Yorkers achieve the ambitious goal of reducing greenhouse gas emissions (GHG) from transportation by rebuilding greener commuting way of EVs and E-bikes.

While we agree that replacing all the traditional taxis with EVs and e-bikes is a time-consuming process that involves many other factors that are not discussed in this paper, we believe sustainable transportation is the future. Introducing EVs and e-bikes in the areas that have high demand for taxis can be the first step towards sustainability. In order to motivate citizens to adapt eco-friendly travel into their lifestyle, we have come up with some actions:

- By analyzing each zip code of pickup area of Manhattan, most pickups occurred in the southern part of Manhattan where government should build more EV chargers in those concentrated areas for meeting consumer needs by having more continent places to plug in their vehicles.
- The current EVs registrations trends have been showing to increase in the prediction model showed that New York City should take more action to encourage people to buy EVs in daily life by providing some promoted financial incentives for electric cars.
- It's also critical for individuals to have the common concept of using eco-friendly way more often in their minds as well, the city needs to take the action to bring the principle of simplicity to the bike ride into daily life by enriching E-bike on-street targeted user group including but not limited to delivery workers with take-out orders, city's tourists, families or just people with their own bikes. The city can build more bike lanes across the city so that it allows riders to commute by e-bikes safely and swiftly. Dedicated bike lanes can motivate more people to use e-bikes during their daily routine.
- Like Ireland's "Bike to work" policy [16], NYC government can collaborate with corporations to motivate their employees to purchase e-bikes by offering financial incentives. Corporations can also include dedicates charge stations in the parking that allow bikes to charge when parked.

In conclusion, replacing traditional cars with electric vehicles and introducing shared bikes which could be rented by customers in neighborhoods with high demand (Upper Westside, Harlem, and East Harlem) can be helpful in evaluating the performance of these services during the initial phase. These results can be analyzed further to constantly improve and scale the service across the city.

7. Future plans and Improvements

In the discussion, we have mentioned introducing the new services in the 3 popular areas of Manhattan. Upon launching the services, usage patterns and demand for this new service can be recorded for a period. Along with that, dedicated user surveys can be conducted to understand customers' opinion about the newly introduced services. This data can be used to further improve the services to achieve maximum customer satisfaction.

The project can be extended to other areas of Manhattan and to the entire city subsequently. Once the project has been scaled, diverse data can be obtained through the services which could help the project to be scaled to other cities of the USA.

8. References

1. Www1.nyc.gov. 2022. nyc gov. [online] Available at: <https://www1.nyc.gov/assets/sustainability/downloads/pdf/publications/OneNYC-2050-Summary.pdf> [Accessed 9 June 2022].
2. Www1.nyc.gov. 2022. Electrifying NYC. [online] Available at: <https://www1.nyc.gov/html/dot/downloads/pdf/electrifying-new-york-report.pdf> [Accessed 9 June 2022].
3. Eia.gov. 2022. Gasoline and Diesel Fuel Update. [online] Available at: <https://www.eia.gov/petroleum/gasdiesel/> [Accessed 9 June 2022].
4. Nycdotscootershare.info. 2022. Shared E-Scooter Pilot | NYC. [online] Available at: <https://nycdotscootershare.info/> [Accessed 9 June 2022].
5. Www1.nyc.gov. 2022. TLC Trip Record Data - TLC. [online] Available at: <https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page> [Accessed 9 June 2022].
6. New York City Taxi and Limousine Commission (March 9, 2006). "The State of the NYC Taxi" (PDF). Retrieved February 18, 2007.

7. Sovacool, B., 2022. Early modes of transport in the United States. [online] Taylor & Francis. Available at: <https://www.tandfonline.com/doi/full/10.1016/j.polsoc.2009.01.006> [Accessed 9 June 2022].
8. Nytimes.com. 2022. Uber Partners With Yellow Taxi NYC. [online] Available at: <https://www.nytimes.com/2022/03/24/business/uber-new-york-taxis.html> [Accessed 9 June 2022].
9. Skelding, C., 2022. Uber, Lyft ridership falls 15% in NYC, taxi cab trips rise. [online] Nypost.com. Available at: <https://nypost.com/2021/10/30/uber-lyft-ridership-falls-15-in-nyc-taxi-cab-trips-rise/> [Accessed 9 June 2022].
10. Lim, S., 2022. Drivers are exposed to the highest levels of harmful air pollution – and taxi drivers are most at risk. [online] The Conversation. Available at: <https://theconversation.com/drivers-are-exposed-to-the-highest-levels-of-harmful-air-pollution-and-taxi-drivers-are-most-at-risk-124368> [Accessed 9 June 2022].
11. Rodrigue, Dr. Jean-Paul. "The Environmental Impacts of Transportation". people.hofstra.edu. Archived from the original on 2018-01-31. Retrieved 2016-04-14.
12. Jen Roberton et al. Emissions from the Taxi and For-Hire Vehicle Transportation Sector in New York City (2020), NYC Gov.
13. Liang Hu et al. Analyzing battery electric vehicle feasibility from taxi travel patterns: The case study of New York City, USA (2018), Transport Research Part C 81 (91-104)
14. Datasmart.ash.harvard.edu. 2022. Case Study: New York City Taxis. [online] Available at: <https://datasmart.ash.harvard.edu/news/article/case-study-new-york-city-taxis-596> [Accessed 9 June 2022].
15. Nytimes.com. 2022. Here's How Slowly New York City Is Moving on Electric Vehicles. [online] Available at: <https://www.nytimes.com/2022/04/05/nyregion/nyc-electric-vehicle-evs.html> [Accessed 9 June 2022].
16. 2022. Bike to work. [online] Available at: <https://www.biketowork.ie/> [Accessed 9 June 2022].
17. En.wikipedia.org. 2022. scikit-learn - Wikipedia. [online] Available at: <https://en.wikipedia.org/wiki/Scikit-learn> [Accessed 9 June 2022].
18. Towards Data Science. 2022. Towards Data Science. [online] Available at: <https://towardsdatascience.com/> [Accessed 9 June 2022].

19. 2016 Blizzard Was NYC's Biggest Snowstorm on Record, NOAA Report Finds. [online] Available at: <https://www.nbcnewyork.com/news/local/nyc-new-york-city-blizzard-biggest-ever-january-23-2016/831660//> [Accessed 9 June 2022]

Appendix A

All the datasets are obtained from the trusted/official sources. They can be found at the following links:

1. New York yellow taxi trips (2016) - TLC Trip Record Data - TLC (nyc.gov)
2. New York weather (2016) - NWS New York Significant Weather Events Archive
3. CitiBike bike trip records (2016-2021) - Index of bucket "tripdata"
4. New York Electric vehicles registrations (2011-2021) - State EV Registration Data – Atlas EV Hub

Appendix B

All the interactive dashboards are published in Tableau Public. These dashboards can be found here.

Appendix C

Python libraries Pandas, NumPy and Matplotlib was used for few phases of CRISP-DM methodologies (data understanding, data modelling, and data cleaning). The python code snippets for these processes are included below.

1. Python script to obtain zip code from longitude and latitude using GeoPy library

```
def get_zipcode(df, geolocator, lat_field, lon_field):
    location = geolocator.reverse((df[lat_field], df[lon_field]),timeout=200)
    try:
        df["zip code"] = location.raw['address']['postcode']

    except:
        df["zip code"] = np.NaN

    return df
```

2. Python script to obtain travel distance between two co-ordinates using Bing Map API

```
def calculate_distance(df, pickup_latitude, pickup_longitude, dropoff_latitude, dropoff_longitude):
    sleep(0.3)
    routeUrl = "http://dev.virtualearth.net/REST/V1/Routes/Driving?wp.0=" + str(df[pickup_latitude]) + ","
               + str(df[pickup_longitude]) + "&wp.1="+ str(df[dropoff_latitude]) + "," + str(df[dropoff_longitude])
               + "&key=" + bingMapsKey
    try:
        request = urllib.request.Request(routeUrl)
        response = urllib.request.urlopen(request)
        jsonResponse = json.load(response)
        df["travelDistance"] = jsonResponse["resourceSets"][0]["resources"][0]["travelDistance"]
        df["travelDuration"] = jsonResponse["resourceSets"][0]["resources"][0]["travelDuration"]
        df["travelDurationTraffic"] = jsonResponse["resourceSets"][0]["resources"][0]["travelDurationTraffic"]
    except:
        df["travelDistance"] = np.NaN
        df["travelDuration"] = np.NaN
        df["travelDurationTraffic"] = np.NaN
    return df
```

3. Python script to filter through zip codes of Manhattan, NYC.

```
import pandas as pd
import numpy as np

zipcode_df = pd.read_csv("manhattan_zipcodes.txt", sep="\t")
vehicle_df = pd.read_csv("ny_ev_registrations_public.csv")
sample_df = pd.read_csv("sample_train_withzip.csv")

print(type(zipcode_df["Zip Code"]))
zipcode_df["Zip Code"] = zipcode_df["Zip Code"].astype(int)
vehicle_df["Zip Code"] = vehicle_df["Zip Code"].astype(int)
sample_df["Zip Code"] = pd.to_numeric(sample_df["Zip Code"], errors='coerce')
sample_df = sample_df.dropna(subset=['Zip Code'])
sample_df["Zip Code"] = sample_df["Zip Code"].astype(int)

final_df1 = vehicle_df[vehicle_df["Zip Code"].isin(zipcode_df["Zip Code"])]
final_df2 = sample_df[sample_df["Zip Code"].isin(zipcode_df["Zip Code"])]
final_df1.to_csv("manhattan_evs.csv", index=False)
final_df2.to_csv("sample_train.csv", index=False)
```

4. Python script to create location-based clusters for k-means clustering algorithm.

```
def min_distance(cluster_len,cluster_centers):
    lessth2 = []
    moreth2 = []
    good_points = 0
    bad_points = 0
    min_dist=1000
    for i in range(0, cluster_len):
        good_points = 0
        bad_points = 0
        for j in range(0, cluster_len):
            if j!=i:
                distance = gpxpy.geo.haversine_distance(cluster_centers[i][0], cluster_centers[i][1],cluster_centers[j][0],
                min_dist = min(min_dist,distance/(1.60934*1000))
                if (distance/(1.60934*1000)) <= 2:
                    good_points +=1
                else:
                    bad_points += 1
        lessth2.append(good_points)
        moreth2.append(bad_points)
    neighbours.append(lessth2)
    print (" for K ",cluster_len,"\n Avg. Number of Clusters, inter cluster dist < 2):", np.ceil(sum(lessth2)/len(lessth2)),
```

```
def find_clusters(i):
    kmeans = MiniBatchKMeans(n_clusters=i, batch_size=10000,random_state=22).fit(coords)
    data['pickup_cluster'] = kmeans.predict(data[['pickup_latitude', 'pickup_longitude']])
    cluster_centers = kmeans.cluster_centers_
    cluster_len = len(cluster_centers)
    return cluster_centers, cluster_len
```

5. Python script to generate the month, average distance, number of trips, number of subscribers, number of males and females for each month.

```
import pandas as pd
import numpy as np
import os
from datetime import datetime

main_df = pd.read_csv('CitiBikeData.csv')

dir_name = 'D:\\Backup\\Datasets\\CitiBike\\zips'
os.chdir(dir_name) # change directory from working dir to dir with files

for item in os.listdir(dir_name): # check for ".zip" extension
    file_name = os.path.abspath(item) # get full path of files
    df = pd.read_csv(file_name)


    main_df.loc[len(main_df.index)]=[datetime.strptime(df.iloc[0,1],'%Y-%m-%d %H:%M:%S.%f').date(),
    df.iloc[:,0].mean().round(),
    df.iloc[:,0].count(),
    df.iloc[:,12].value_counts()['Subscriber'],
    df.iloc[:,12].value_counts()['Customer'],
    df.iloc[:,14].value_counts()[1],
    df.iloc[:,14].value_counts()[2],
    df.iloc[:,14].value_counts()[0]]


main_df.to_csv('CitiBikeData.csv', index=False)
```

Appendix D

Python libraries Pandas, NumPy, sci-kit learn, xgboost, gxpy, folium, metrics, was used for the remaining phases of CRISP-DM methodologies (data modelling, and evaluation). The python code snippets for these processes are included below.

1. Python code for data modelling with Unix time conversion, feature selection, test-train split and functions to calculate RMSLE (root mean-square log error)

```
import numpy as np
import pandas as pd
from matplotlib import pyplot
import xgboost
import pandas as pd
import gpxpy.geo #Get the haversine distance
import numpy as np
import matplotlib.pylab as plt
from sklearn.cluster import MiniBatchKMeans, KMeans
import folium
```

```
do_not_use_for_training = ['id', 'pickup_datetime', 'dropoff_datetime',
                           'trip_duration', 'check_trip_duration',
                           'pickup_date', 'avg_speed',
                           'pickup_lat_bin', 'pickup_long_bin',
                           'center_lat_bin', 'center_long_bin',
                           'pickup_dt_bin', 'pickup_datetime_group',
                           'store_and_fwd_flag']
feature_names = [f for f in train.columns if f not in do_not_use_for_training]
X, y = train[feature_names], train["trip_duration"]
Xtest = test[feature_names]
```

```
Index(['vendor_id', 'passenger_count', 'pickup_longitude', 'pickup_latitude',
       'dropoff_longitude', 'dropoff_latitude', 'distance', 'pickup_weekday',
       'pickup_weekofyear', 'pickup_hour', 'pickup_minute', 'pickup_dt',
       'pickup_week_hour'],
      dtype='object')
```

```
def add_pickup_bins(frame,month,year):
    unix_pickup_times=[i for i in frame['pickup_times'].values]
    unix_times = [[1420070400,1422748800,1425168000,1427846400,1430438400,1433116800],\
                  [1451606400,1454284800,1456790400,1459468800,1462060800,1464739200]]

    start_pickup_unix=unix_times[year-2015][month-1]
    # https://www.timeanddate.com/time/zones/est
    # (int((i-start_pickup_unix)/600)+33) : our unix time is in gmt to we are converting it to est
    tenminutewise_binned_unix_pickup_times=[(int((i-start_pickup_unix)/600)+33) for i in unix_pickup_times]
    frame['pickup_bins'] = np.array(tenminutewise_binned_unix_pickup_times)
    return frame
```

```
1 from sklearn.model_selection import train_test_split
2 X_train, X_test, y_train, y_test = train_test_split(X,y, test_size=0.3)
```

```
1 X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

```
((1021050, 14), (437594, 14), (1021050,), (437594,))
```

```
def rmsle(evaluator,X,real):
    sum = 0.0
    predicted = evaluator.predict(X)
    print("Number predicted less than 0: {}".format(np.where(predicted < 0)[0].shape))

    predicted[predicted < 0] = 0
    for x in range(len(predicted)):
        p = np.log(predicted[x]+1)
        r = np.log(real[x]+1)
        sum = sum + (p-r)**2
    return (sum/len(predicted))**0.5
```

```
def rmsl(X,real):
    sum = 0.0
    print("Number predicted less than 0: {}".format(np.where(predicted < 0)[0].shape))

    predicted[predicted < 0] = 0
    for x in range(len(predicted)):
        p = np.log(predicted[x]+1)
        r = np.log(real[x]+1)
        sum = sum + (p-r)**2
    return (sum/len(predicted))**0.5
```

2. Code for XGB Regressor model.

```
from sklearn.model_selection import ShuffleSplit
from sklearn.model_selection import cross_val_score
```

```
reg = xgboost.XGBRegressor(n_estimators=100, learning_rate=0.08, gamma=0, subsample=0.75,
                           colsample_bytree=1, max_depth=10)

#cv = ShuffleSplit(n_splits=2, test_size=0.2, random_state=0)
```

```
reg.fit(X_train,y_train)
```

```
Regressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
          colsample_bynode=1, colsample_bytree=1, gamma=0, gpu_id=-1,
          importance_type='gain', interaction_constraints='',
          learning_rate=0.08, max_delta_step=0, max_depth=10,
          min_child_weight=1, missing=nan, monotone_constraints='()',
          n_estimators=100, n_jobs=12, num_parallel_tree=1, random_state=0,
          reg_alpha=0, reg_lambda=1, scale_pos_weight=1, subsample=0.75,
          tree_method='exact', validate_parameters=1, verbosity=None)
```

```
x_model = xgb.XGBRegressor(
 learning_rate =0.1,
 n_estimators=1000,
 max_depth=3,
 min_child_weight=3,
 gamma=0,
 subsample=0.8,
 reg_alpha=200, reg_lambda=200,
 colsample_bytree=0.8,nthread=4)
x_model.fit(df_train, tsne_train_output)
```

```
GBRegressor(base_score=0.5, colsample_bylevel=1, colsample_bytree=0.8,
      gamma=0, learning_rate=0.1, max_delta_step=0, max_depth=3,
      min_child_weight=3, missing=None, n_estimators=1000, nthread=4,
      objective='reg:linear', reg_alpha=200, reg_lambda=200,
      scale_pos_weight=1, seed=0, silent=True, subsample=0.8)
```

```
#predicting with our trained Xg-Boost regressor
# the models x_model is already hyper parameter tuned
# the parameters that we got above are found using grid search

y_pred = x_model.predict(df_test)
xgb_test_predictions = [round(value) for value in y_pred]
y_pred = x_model.predict(df_train)
xgb_train_predictions = [round(value) for value in y_pred]
```

```
y_pred = reg.predict(X_train)
len(y_pred)
```

```
1021050
```

```
y_pred[1]
```

```
621.82935
```

```
rmsle(reg,X_train,np.array(y_train))
```

```
Number predicted less than 0: (193,)

0.5466871581540065
```

```
rmsle(reg,X_test,np.array(y_test))
```

```
Number predicted less than 0: (193,)

0.5943641245047101
```

3. Code for Random Forest regressor model

```
from sklearn.ensemble import RandomForestRegressor

 # create regressor object
regressor = RandomForestRegressor(n_estimators = 100, random_state = 0)

# fit the regressor with x and y data
regressor.fit(X_train,np.array(y_train) )
```

```
RandomForestRegressor(random_state=0)
```

```
rmsle(regressor,X_train,np.array(y_train))
```

```
Number predicted less than 0: (0,)

0.3162386821590284
```

```
rmsle(regressor,X_test,np.array(y_test))
```

```
Number predicted less than 0: (0,)

0.587123436546442
```

4. Code for Linear SVR model

```
from sklearn.svm import LinearSVR

 # create regressor object
regressor = LinearSVR(C=1.0, random_state = 0)

# fit the regressor with x and y data
regressor.fit(X_train,np.array(y_train) )
```

```
LinearSVR(random_state=0)
```

```
rmsle(regressor,X_train,np.array(y_train))
```

```
Number predicted less than 0: (0,)

0.7854509887412529
```

```
rmsle(regressor,X_test,np.array(y_test))
```

```
Number predicted less than 0: (0,)

0.7862231833538764
```

```
from sklearn.tree import DecisionTreeRegressor
regressor = DecisionTreeRegressor(max_depth=5)
regressor.fit(X, y)
```

```
DecisionTreeRegressor(max_depth=5)
```

```
rmsle(regressor,X_train,np.array(y_train))
```

```
Number predicted less than 0: (0,)

0.70836211514115
```

```
rmsle(regressor,X_test,np.array(y_test))
```

```
Number predicted less than 0: (0,)

0.7090681550258375
```

```
from tabulate import tabulate
```

5. Code for Simple Moving Average (SMA)

```
def MA_R_Predictions(ratios,month):
    predicted_ratio=(ratios['Ratios'].values)[0]
    error=[]
    predicted_values=[]
    window_size=3
    predicted_ratio_values=[]
    for i in range(0,4464*40):
        if i%4464==0:
            predicted_ratio_values.append(0)
            predicted_values.append(0)
            error.append(0)
            continue
        predicted_ratio_values.append(predicted_ratio)
        predicted_values.append(int(((ratios['Given'].values)[i])*predicted_ratio))
        error.append(abs((math.pow(int(((ratios['Given'].values)[i])*predicted_ratio)-(ratios['Prediction'].values)[i],1))))
        if i+1>=window_size:
            predicted_ratio=sum((ratios['Ratios'].values)[(i+1)-window_size:(i+1)])/window_size
        else:
            predicted_ratio=sum((ratios['Ratios'].values)[0:(i+1)])/(i+1)


    ratios['MA_R_Predicted'] = predicted_values
    ratios['MA_R_Error'] = error
    mape_err = (sum(error)/len(error))/(sum(ratios['Prediction'].values)/len(ratios['Prediction'].values))
    mse_err = sum([e**2 for e in error])/len(error)
    return ratios,mape_err,mse_err
```

6. Code Linear Regression model

```
from sklearn.linear_model import LinearRegression
lr_reg=LinearRegression().fit(df_train, tsne_train_output)

y_pred = lr_reg.predict(df_test)
lr_test_predictions = [round(value) for value in y_pred]
y_pred = lr_reg.predict(df_train)
lr_train_predictions = [round(value) for value in y_pred]
```