Wheel Analytics

Predicting New York Yellow Taxi demand and supply

Team Members:

- Narendra Bandi
- Pratik Chaudhari
- Santhosh Babu
- Mrugank Dave

Contents

1. Business Problem

There are approximately 210 million taxis rides every year in New York. Exploring and understanding taxi supply demand. Also, what could improve the efficiency of the supply chain for city's taxi system. No other city of United States has as much use of taxis as much it is used in the city of New York. New York taxi drivers usually tend to pick up their passengers from the street rather than being prebooked a few minutes prior or the previous day. Medallion (yellow) cabs are concentrated in the borough of Manhattan but can be hailed anywhere throughout the five boroughs of New York City.

Since the beginning of non – traditional taxi services era of Uber and Lyft, it has indeed become very difficult for the yellow taxis to compete with the way they use technology for pickups and drop-offs, which has led to the decrease of one of the oldest methods to commute in New York. Recently there is a rise in frustration of the customers due to the surge prices applied by these non - traditional cab companies which has made people expressing their willingness to turn to yellow taxi services. Even though customers who want to use yellow taxi mode of transport for their travel, the demand and supply of the taxis are not being contemplated. The main purpose of this project is to examine the data and find a way that would benefit both the Limousine and taxi commission and the customers.

The ability to predict demand of the taxi ridership in a particular area could present valuable insights to city planners and New York's Limousine and Taxi commission in answering questions such as how to position cabs where they are most needed, how many taxis to dispatch for a particular location at a particular time, and how the demand of taxi ridership changes over time. Our project focuses on predicting the number of taxi pickups given at a time and a location within New York City.

The solution here will help the taxis get profited by revenue earned from the rides and the customers by service. Customers can comparatively pay less amount for their rides and may not have to wait for the taxis to arrive with confusion with location pickups and the yellow taxis can know where exactly they are most likely to get a customer. Also, the drivers of the yellow taxi are facing financial problems with having to pay EMI's for their medallion as other non-traditional taxi companies are running the market. To help survive the yellow taxi tradition, this project will attempt to make their system more efficient.

The problem is formulated in terms of the following inputs and outputs.

In this project we have used 2 methods to predict demand of taxis in New York city. First method would consist of multiple linear regression, Deep neural network, Gradient boosted tree to predict the taxi's demand as per their location, time, average passenger count and average trip time. Second method is where we would be forecasting the demand of taxi's by using timeseries ARIMA model. We plan to run gradient boosted model to identify which features are relevant to make the model perform better.

2. Data Sourcing:

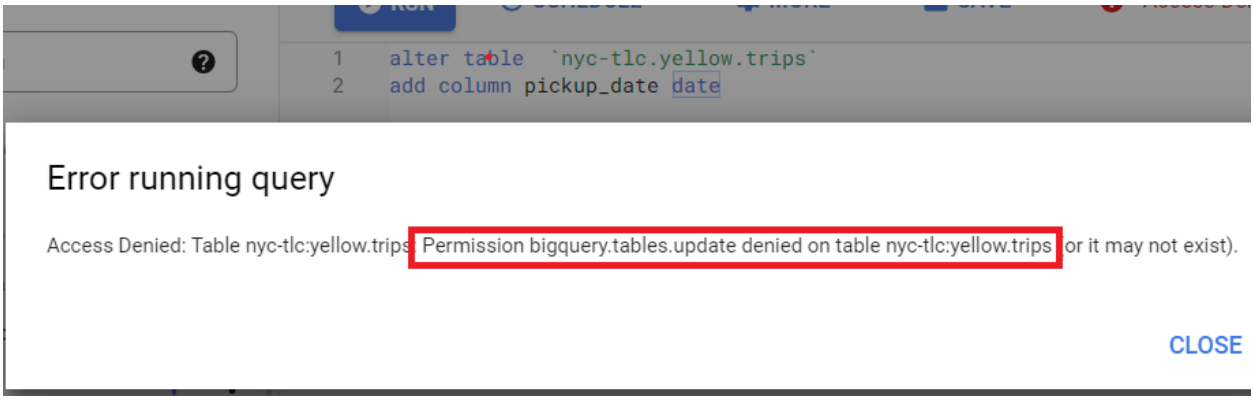
What is the data?

Our Primary dataset includes NYC taxi trips dataset which holds trip records from all trips completed in yellow taxis in NYC since 2009 to 2015. Records include fields capturing pick-up and drop-off date/time, pick-up and drop-off locations, trip distances, itemized fares, rate types, payment types, and driver-reported passenger counts. The data used in the datasets were collected and provided to the NYC Taxi and Limousine Commission (TLC) by technology providers authorized under the Taxicab Passenger Enhancement Program (TPEP).

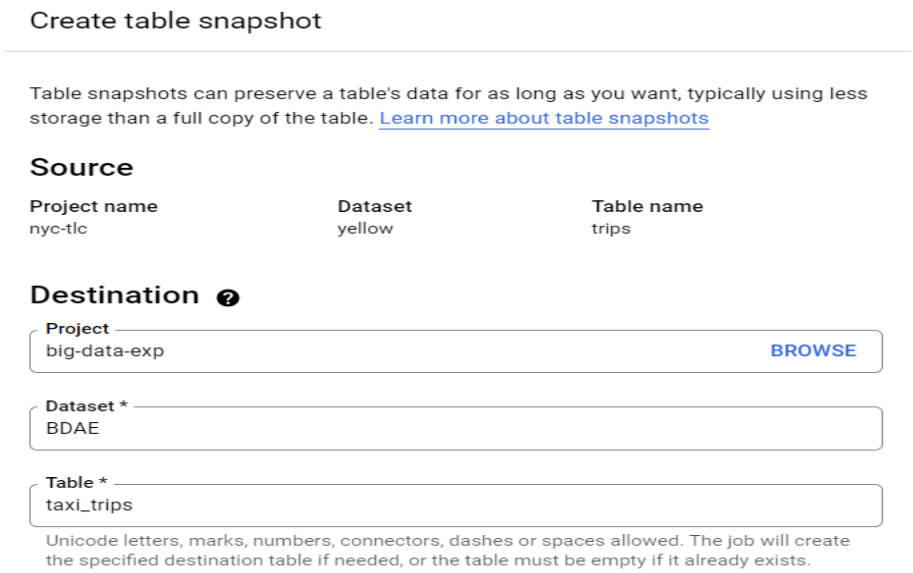
As our primary dataset holds the Latitude and Longitude of the Prick-up/Dropoff location.Wwe were facing difficulties in getting the exact location. Without the Location/Zone name we wouldn't have able to recognize the number of taxis required at a Location, at a given point of time. Hence, to get the location we found the Taxi_zone_geom table in another dataset which was a public dataset. The Taxi_zone_geom consist of LocationID, Zone, ZoneID, geom, etc.

2. Data Storage

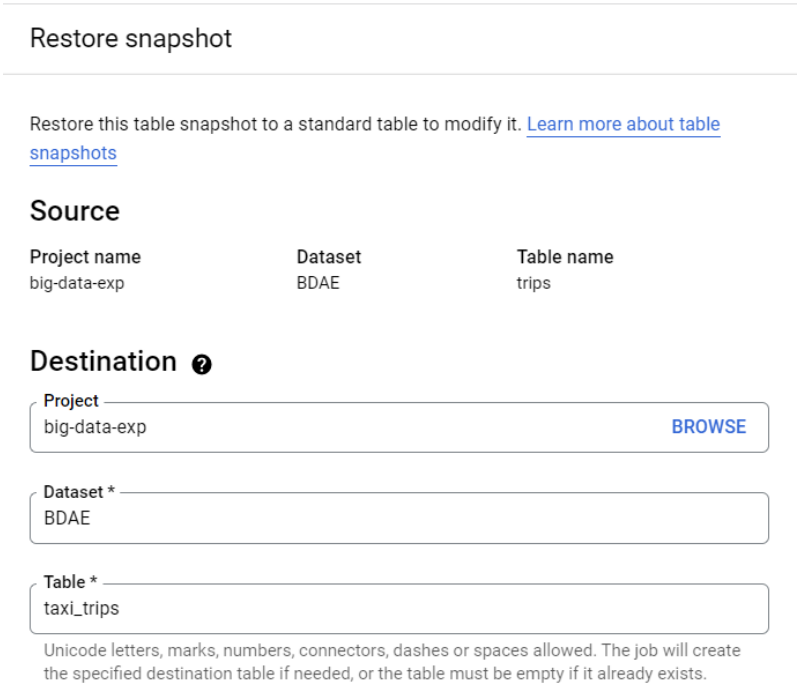
2.1 Challenges faced to use the data



The dataset which we used is on Google BigQuery was under public datasets. And as the data which is present is a raw dataset, we couldn't use it directly for our analysis. Hence, we tried performing some operations, but we couldn't perform any as Google don't allow the users execute any commands except the SELECT command as the dataset was present under public dataset. Hence, to perform actions the dataset was not supposed to be present under the public dataset. We tried copying the dataset, but we didn't have the required permissions as we were not the owner of the dataset.



Hence, we created a snapshot of the primary table which consist of all the trips details and holding 1.2 Bn records. Creating a snapshot is an easy task. We just have to select the table, hit the Snapshot button, select the destination Project and destination Dataset, name the Snapshot, and hit the create button. The snapshot must be created in the same dataset in which the table is currently present. Hence, we created the snapshot. Once the snapshot was created, we needed a destination dataset which should be owned by us. Hence, to host a dataset we need to create a project first. Once the project is created then we need to create a dataset which is a database in SQL.



After creating the Snapshot, we can now restore the snapshot. We can restore a snapshot into any dataset in our project. So, to restore the snapshot we have to select the snapshot, click on Restore button, select the destination project name, select the dataset and enter the table name and click on restore. Once the snapshot is restored, we can use it to perform all actions and can execute all the DDL and DML queries.

After creating the table, we thought of creating new columns for pickup date and time. Hence, we extracted the date and time from datetime format using the **CAST ()** function. Also, extracted

the Month and Day of Week using the **EXTRACT ()** function. The shared fields were required for analysis and to avoid iterative process or calling the functions iteratively we thought of getting them into separate columns.

```
--Query to get time from pickup_datetime and add the same to trip_start_time
update `big-data-exp.BDAE.trips_new`
set trip_start_time = cast(pickup_datetime as time)
where trip_start_time is null
```

```
--Query to get month and add the month into trip_start_month
update `big-data-exp.BDAE.trips_new`
set trip_start_month = extract(month from trip_start_date)
where trip_start_month is null
```

Now, the trips tables contain the pickup and drop-off latitude and longitudes, but we required to get the exact location, zone for forecasting the demand at a location. Hence, to get the respective details we found a table in another dataset which was present in one of the public datasets. So, to get the respective details we need to join the tables but before joining the dataset has to be present in the dataset owned by us. Hence, we followed the same steps for creating a snapshot and restoring the snapshot. The taxi_geom_zone table consist of the LocationID, the geom which was a range if latitudes and longitudes, but the datatype was string, the zone ID, the zone name. Hence, to get the desired fields we first tried to convert the latitudes and longitudes into Geography datatype from stings and then tried to figure out in which range it is falling. Once, the pickup drop-off latitude and longitudes were present in the range of Geom column we assigned the locationID and zone name in the trips table.

```
--find the boroughs and zone names for pickup locations
INNER JOIN `big-data-exp.BDAE.taxi_zone` tz_pu ON
(ST_DWithin(tz_pu.taxi_geom,ST_GeogPoint(pickup_longitude, pickup_latitude), 0))
```

2.2 Table Schema

Field name	Type	Mode	Description
pickup_datetime	TIMESTAMP	NULLABLE	The date and time when the meter was engaged.
dropoff_datetime	TIMESTAMP	NULLABLE	The date and time when the meter was disengaged.
pickup_longitude	FLOAT	NULLABLE	Longitude where the meter was engaged.
pickup_latitude	FLOAT	NULLABLE	Latitude where the meter was engaged.
dropoff_longitude	FLOAT	NULLABLE	Longitude where the meter was disengaged.
dropoff_latitude	FLOAT	NULLABLE	Latitude where the meter was disengaged.
rate_code	STRING	NULLABLE	The final rate code in effect at the end of the trip. 1= Standard rate 2=JFK 3=Newark 4=Nassau or Westchester 5=Negotiated fare 6=Group ride
passenger_count	INTEGER	NULLABLE	The number of passengers in the vehicle. This is a driver-entered value.
trip_distance	FLOAT	NULLABLE	The elapsed trip distance in miles reported by the taximeter.
payment_type	STRING	NULLABLE	A numeric code signifying how the passenger paid for the trip. CRD=

			Credit card CSH= Cash NOC= No charge DIS= Dispute UNK= Unknown
fare_amount	FLOAT	NULLABLE	The time-and-distance fare calculated by the meter.
extra	FLOAT	NULLABLE	Miscellaneous extras and surcharges. Currently, this only includes the $0.50 and $1 rush hour and overnight charges.
mta_tax	FLOAT	NULLABLE	$0.50 MTA tax that is automatically triggered based on the metered rate in use.
imp_surcharge	FLOAT	NULLABLE	$0.30 improvement surcharge assessed on trips at the flag drop. The improvement surcharge began being levied in 2015.
tip_amount	FLOAT	NULLABLE	Tip amount – This field is automatically populated for credit card tips. Cash tips are not included.
tolls_amount	FLOAT	NULLABLE	Total amount of all tolls paid in trip.
total_amount	FLOAT	NULLABLE	The total amount charged to passengers. Does not include cash tips.
store_and_fwd_flag	STRING	NULLABLE	This flag indicates whether the trip record was held in vehicle memory before sending to the vendor, aka “store and forward,” because the vehicle did not have a connection to

			the server. Y= store and forward trip N= not a store and forward trip
trip_start_date	DATE	NULLABLE	
trip_start_time	TIME	NULLABLE	
trip_start_month	INTEGER	NULLABLE	
trip_start_day	INTEGER	NULLABLE	
trip_start_hour	INTEGER	NULLABLE	

3. Data Cleaning

Data cleansing or data cleaning is the process of detecting and correcting (or removing) corrupt or inaccurate records from a record set, table, or database and refers to identifying incomplete, incorrect, inaccurate, or irrelevant parts of the data and then replacing, modifying, or deleting the dirty or coarse data.

The dataset which we used for forecasting the NYC taxi count is a raw dataset and we cannot use the dataset directly for our analysis or forecasting. Hence, we tried to check for the blank, NULL, outlier values. We found that most of the fields like **mta_tax, tip_amount, tolls_amount, imp_surcharges** where NULL. Also, the count of such records was also too high hence, to avoid loss of data we updated the records to 0 as the respective fields didn't have much impact on our analysis.

```
--- Convert NULL values to 0

 UPDATE `big-data-exp.BDAE.TaxiTrips_Geog`
SET tip_amount = 0
WHERE tip_amount IS NULL;

UPDATE `big-data-exp.BDAE.TaxiTrips_Geog`
SET tolls_amount = 0
WHERE tolls_amount IS NULL;

UPDATE `big-data-exp.BDAE.TaxiTrips_Geog`
SET mta_tax = 0
WHERE mta_tax IS NULL;

UPDATE `big-data-exp.BDAE.TaxiTrips_Geog`
SET imp_surcharge = 0
WHERE imp_surcharge IS NULL;
```

We also checked for the minimum and maximum latitude/longitude and figured out that some are out of range. The correct latitudes range from 0 to 90. Longitudes range from 0 to 180. Therefore, the latitude & longitude columns need to be filtered to the correct range. By doing this, null values will also be eliminated.

```
--- Drop records with invalid latitude and longitude

delete  from `big-data-exp.BDAE.trips_new` as trips WHERE
  ((trips.pickup_latitude not BETWEEN -90 AND 90) or
  (trips.pickup_longitude not BETWEEN -180 AND 180))
or
  ((trips.dropoff_latitude not BETWEEN -90 AND 90) or
  (trips.dropoff_longitude not BETWEEN -180 AND 180))
```

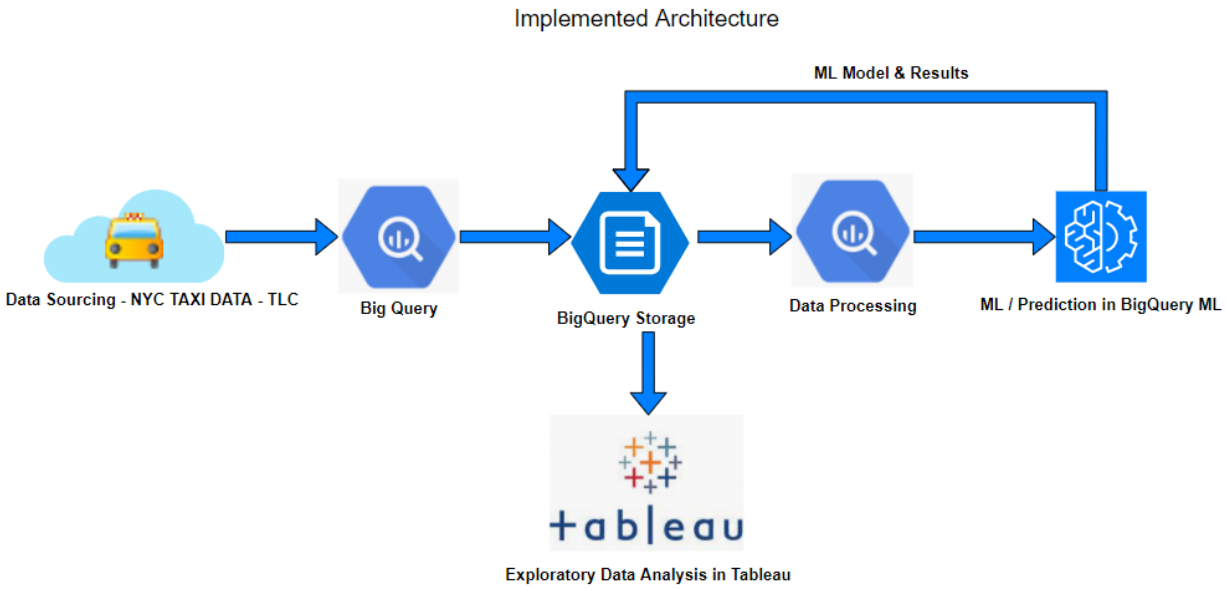
We further tried to get understanding of the passenger demographics, we found that, there were few trips which had trip amount greater than 0 i.e., postive fare amount but the passenger count was 0. Hence, to get rid off of such cases we dropped such records. Even further we discovered that few of the trips had positive distance travelled but fare_amount was 0, or the fare_amount was 0 but the trip_distance was 0. Also, in some cases the shared fields were NULL. Even the values in such fields were negative as well. Hence, after considering all such possibilities were dropped such records to get a clean dataset.

```
delete from `big-data-exp.BDAE.TaxiTrips_Geog` as T
where
T.trip_distance is null or
T.fare_amount is null or
T.passenger_count <= 0
```

Post executing the above query **9,400,294** i.e., **~9 Mn** records were deleted from **1.2 Bn**. The respective data was used for performing Exploratory Data Analysis and for Forecasting.

Before performing or while performing we had a risk of losing the dataset which would have led to iterating the steps from the start like importing the data, creating a new dataset, etc. Hence, to avoid such scenario we were taking timely backups and snapshots of our dataset to avoid re-working of following the same steps.

4. Architecture



Big Query has a vast collection of public datasets available to the users as we described earlier in the data sourcing section. The data which we are going to analyze is the New York City Yellow Taxi dataset. It has close to 1.1 billion rows. The Big Query platform makes it feasible to work with these huge volumes of data. The data is being stored in the Big Query database; we are utilizing two datasets here. One is the yellow taxi dataset and the other one being the

‘taxi_zone_geom’ dataset which contains all the pickup and drop-off location co-ordinates and the zone names.

We utilize Big Query as the storage medium as the data is already made available in the Big Query database. Also, transferring such huge volumes of data will take a considerable amount of time. Big Query obviates the need to transfer this dataset, and instead provides a platform where we can work with them within the console to perform our analysis.

Big Query with its serverless architecture allows it to operate at scale and speed over large datasets. We have utilized this processing power, also bigquery allows the use of standard SQL queries to work with and perform data processing and analysis. This platform is made available to the user with limited capabilities at free of cost, such as the data processing memory is limited to 1TB for free users. But we were able to accommodate all our data analysis and processing within this allotted memory. The cloud console within the Big Query platform mitigates the complexities involved in loading data and processing over them. There is no need to size up the clusters, or hardware resources or any other configuration or setup required to start with data processing.

Big Query has its own machine learning feature called the BigQuery ML. This feature allows us to create and execute machine learning models in BigQuery with the use of standard SQL queries. It also eliminates the need to move data and work with them within the same platform to perform machine learning modelling. We’ve made use of the google cloud console to run these SQL queries over the datasets. Also, BigQuery ML supports different models to work with. Since the data sourcing, storage and machine learning modelling were all implemented in the same Big Query platform, we were able to work with multiple datasets without the need to move data, as well as to create machine learning models with speed and efficiency. It also reduced the complexity because not many tools were required in this process.

To perform exploratory data analysis, we have used Tableau. It gives us the option to connect with our project in BigQuery and to work with the database that is available in that project. This setup didn’t consume more time as it was a straightforward process. This easy integration and

the interactive and easy to use interface that Tableau provides allowed us to analyze the massive volumes of data.

5. ETL

The dataset is made available in the BigQuery Storage under the public database. We created a new project within BigQuery to work with the dataset but to so, it demands the dataset to be available within the project that we created. So, we were looking for ways to overcome this, to move the dataset which is available in the public database to our project folder.

The BigQuery provides a feature to take snapshot of the table that preserves the content of the table at that time. The table snapshot can be configured to have an expiration date, once the time has passed, BigQuery deletes the table snapshot. We were able to create a snapshot and we were able to view the dataset in our project folder, but we were not able to modify or alter the table as table snapshots are read-only. We did find a way to overcome this issue by exploring the restore option within BigQuery. This function can be performed on a table snapshot to restore the snapshot to a standard table on which we will be able to modify the data.

As the data is already loaded in the BigQuery database, we had to move them into our project to work with them. Creating a snapshot from the public dataset and then restoring them in our project helped us to work with this data and to get started with our data pre-processing and analysis.

6.Analytics

Feature Engineering

The model performance is largely dependent on the data pre-processing techniques. The dataset though made available in the big query platform had a lot of missing values and other anomalies that we had to eliminate which were discussed in the data cleaning section. On top of that, to perform our modelling, we require a dataset that has to have the values which

correspond to the time, say for every hour, what could be the number of trips from a specific pickup to drop-off zone.

Binning is a technique used to transform numerical values and group them together into separate bins. This process clubs the data into discrete categorial variables. In our dataset, we had various derived fields such as the hour, month, day, and the location-ID which are all the different categorical variables. We created a new dataset that has the aggregated values of all these categorical features which also gives us information like the number of trips that took place at a specific time and at a specific location. This aggregated table is very much useful in the modelling process.

Aggregated by pickup location

```
select T.pickup_locationID as pickup_locationID, count(*) as Trips,
T.trip_start_month as Month,
T.trip_start_day as Day,
T.trip_start_hour as StartHour,
T.pickup_name as PickupZone,
T.pickup_borough as PickupBorough,
round(AVG(T.passenger_count),2) as PassengerCount,
round (AVG(T.tip_amount), 2) as TipAmount,
round (AVG(T.fare_amount),2) as FareAmount,
round (AVG(T.tolls_amount) ,2) as TollAmount,
round (AVG(T.mta_tax), 2) as MTA,
round (AVG(T.imp_surcharge), 2) as ImpSurcharge,
from `big-data-exp.BDAE.TaxiTrips_Geog` as T
group by T.pickup_locationID, T.trip_start_month, T.trip_start_day,
T.trip_start_hour,T.pickup_name, T.pickup_borough;
```

Row	pickup_locationID	Trips	Month	Day	StartHour	PickupZone	PickupBorough	PassengerCount	TipAmount	FareAmount	TollAmount	MTA	ImpSurcharge
1	256	283	1	3	17	Williamsburg (South Side)	Brooklyn	1.58	0.93	12.02	0.02	0.46	0.03
2	256	1448	1	1	21	Williamsburg (South Side)	Brooklyn	1.71	1.14	11.35	0.01	0.46	0.03
3	256	324	1	5	9	Williamsburg (South Side)	Brooklyn	1.68	1.83	14.95	0.06	0.47	0.07
4	256	225	1	4	11	Williamsburg (South Side)	Brooklyn	1.52	1.27	12.71	0.02	0.46	0.02
5	256	264	1	5	11	Williamsburg (South Side)	Brooklyn	1.56	1.3	13.63	0.1	0.44	0.04

Aggregated by drop-off location

```
select T.dropoff_locationID as dropoff_locationID, count(*) as Trips,
T.trip_start_month as Month,
T.trip_start_day as Day,
T.trip_start_hour as StartHour,
T.dropoff_zone as DropoffZone,
T.dropoff_borough as DropoffBorough,
round(AVG(T.passenger_count),2) as PassengerCount,
round (AVG(T.tip_amount), 2) as TipAmount,
round (AVG(T.fare_amount),2) as FareAmount,
round (AVG(T.tolls_amount) ,2) as TollAmount,
round (AVG(T.mta_tax), 2) as MTA,
round (AVG(T.imp_surcharge), 2) as ImpSurcharge,
from `big-data-exp.BDAE.TaxiTrips_Geog` as T
group by T.dropoff_locationID, T.trip_start_month, T.trip_start_day,
T.trip_start_hour,T.dropoff_zone, T.dropoff_borough;
```

Row	dropoff_locationID	Trips	Month	Day	StartHour	DropoffZone	DropoffBorough	PassengerCount	TipAmount	FareAmount	TollAmount	MTA	ImpSurcharge
1	256	418	1	6	7	Williamsburg (South Side)	Brooklyn	1.56	1.56	17.93	0.01	0.43	0.04
2	256	1226	1	4	17	Williamsburg (South Side)	Brooklyn	1.64	1.8	17.59	0.02	0.44	0.04
3	256	731	1	4	13	Williamsburg (South Side)	Brooklyn	1.71	1.34	17.56	0.01	0.45	0.04
4	256	1105	1	3	17	Williamsburg (South Side)	Brooklyn	1.73	1.76	17.23	0.03	0.44	0.04
5	256	542	1	3	11	Williamsburg (South Side)	Brooklyn	1.57	1.26	17.07	0.03	0.44	0.03

Sampling is the process of selecting observations or samples from the dataset with the intent of representing the dataset. There are various sampling techniques such as simple random sampling, stratified sampling etc. Big Query Machine Learning provided the option to use random sampling methods with the 'Data_Split_Method' option. We also have to specify the fraction to represent the training and test dataset. This process is well within the modelling parameters as Big Query allows us to provide the sampling method and fraction value with the SQL query parameters for model creation.

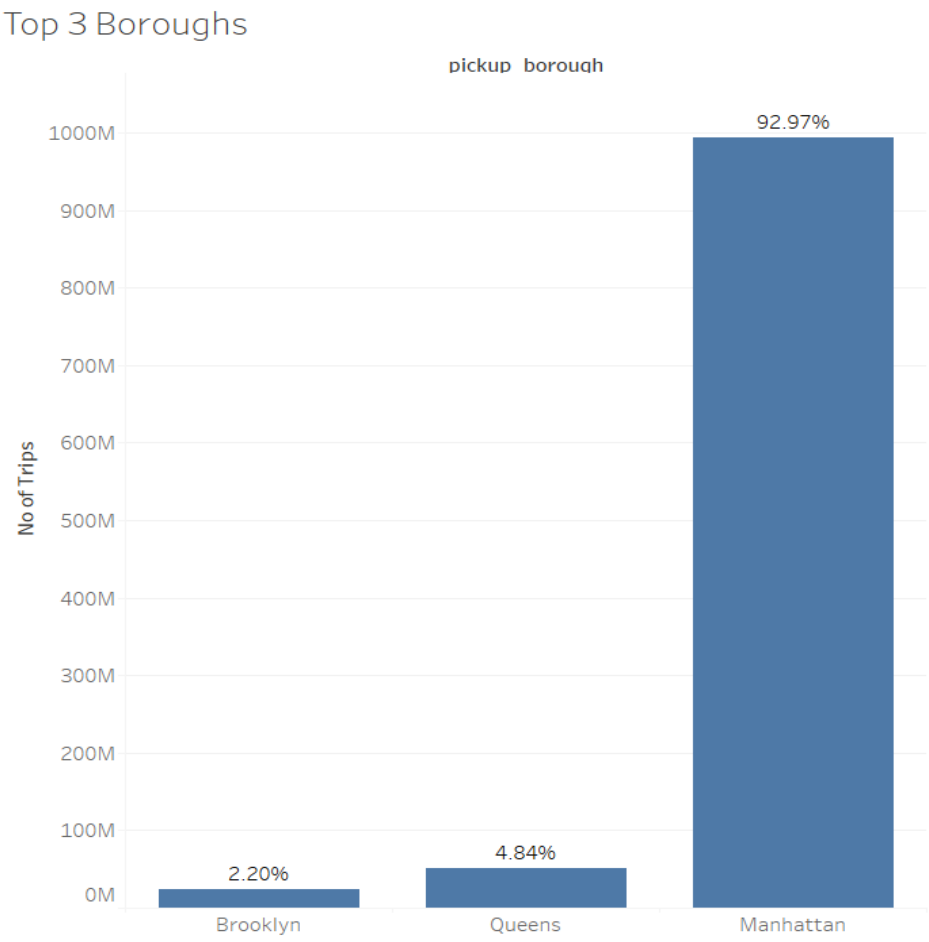
7. Visualizations

Data Visualization is the graphical representation of data. It helps to understand the underlying trends, pattern as well as to identify the outliers in the data. Often data visualization helps us to quickly get an understanding of data as we are plotting them and perceiving them visually which conveys a message that we will be able to internalize easily. It obviates the convoluted values of rows and columns in a spreadsheet to more visually highlight graphs and charts that

tell the story. Making sense of billions of rows is an especially important process as these data contain a lot of insights and trend patterns.

We have used Tableau Desktop server to plot the visualizations and to observe the patterns. Tableau eases the integration setup with Big Query, also it has a lot of added features and capabilities to plot the data in different visualization forms as per the need to better perceive the data and identity the patterns or the information that lies within them. We have created the following visualizations using Tableau.

Figure 1 Top Boroughs in NYC



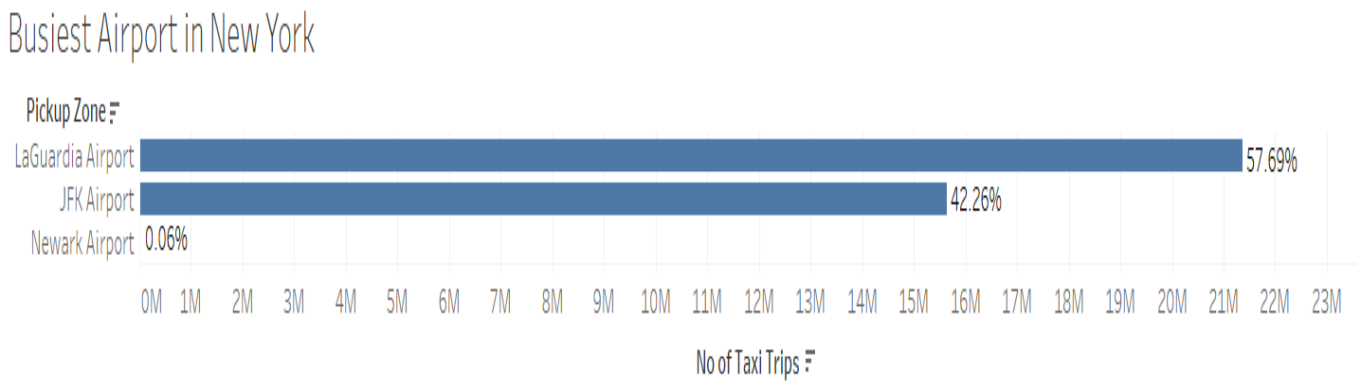
The whole of New York City is split into 6 Boroughs. They are listed as below,

- Manhattan,

- Brooklyn,
- Queens,
- Bronx,
- EWR,
- Staten Island

In our first visualization, we have looked at the popular boroughs in New York City. It was no surprise that Manhattan was at the top with over 92.97% of trips from the dataset. The dataset contains six years of data, starting from 2009 and till 2015. So, we were able to plot and decipher more than a billion trips within New York City. And among them, Manhattan came out to be the most popular borough. It is geographically smallest and densely populated borough. It is also home to many prominent landmarks, which include Times Square and Central Park, hence the results seen in the plot. Queens comes as the second borough with over 50 million trips in the six years period and then comes Brooklyn which has close to 20 million trips.

Figure 2 Busiest Airport in NYC

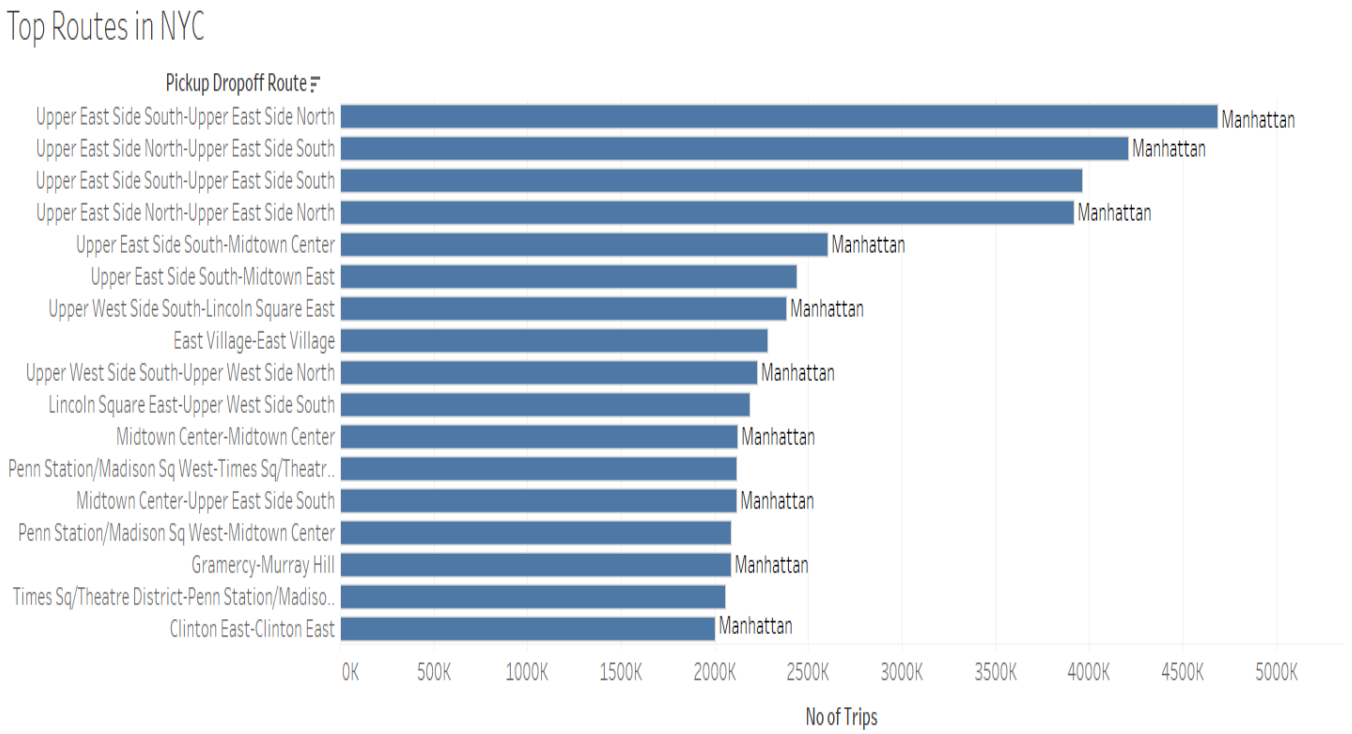


Next, we wanted to see how the airports look like in NYC with respect to the number of trips. This tells us about the busiest airport in New York City. We can see that LaGuardia Airport came at the top with more than 21 million trips. This can be attributed to the fact that this airport is

closest to Manhattan borough which is the most popular borough in New York City as we have seen in the earlier visualization.

People who are looking to explore Manhattan borough, which has the major attractions, are highly likely to choose this airport as their pickup and drop-off point due to its close proximity to Manhattan. Next to LaGuardia comes the John.F.Kennedy airport with over 15 million trips and then Newark airport which is in the EWR borough and has an exceptionally low percentage of trips from the whole data.

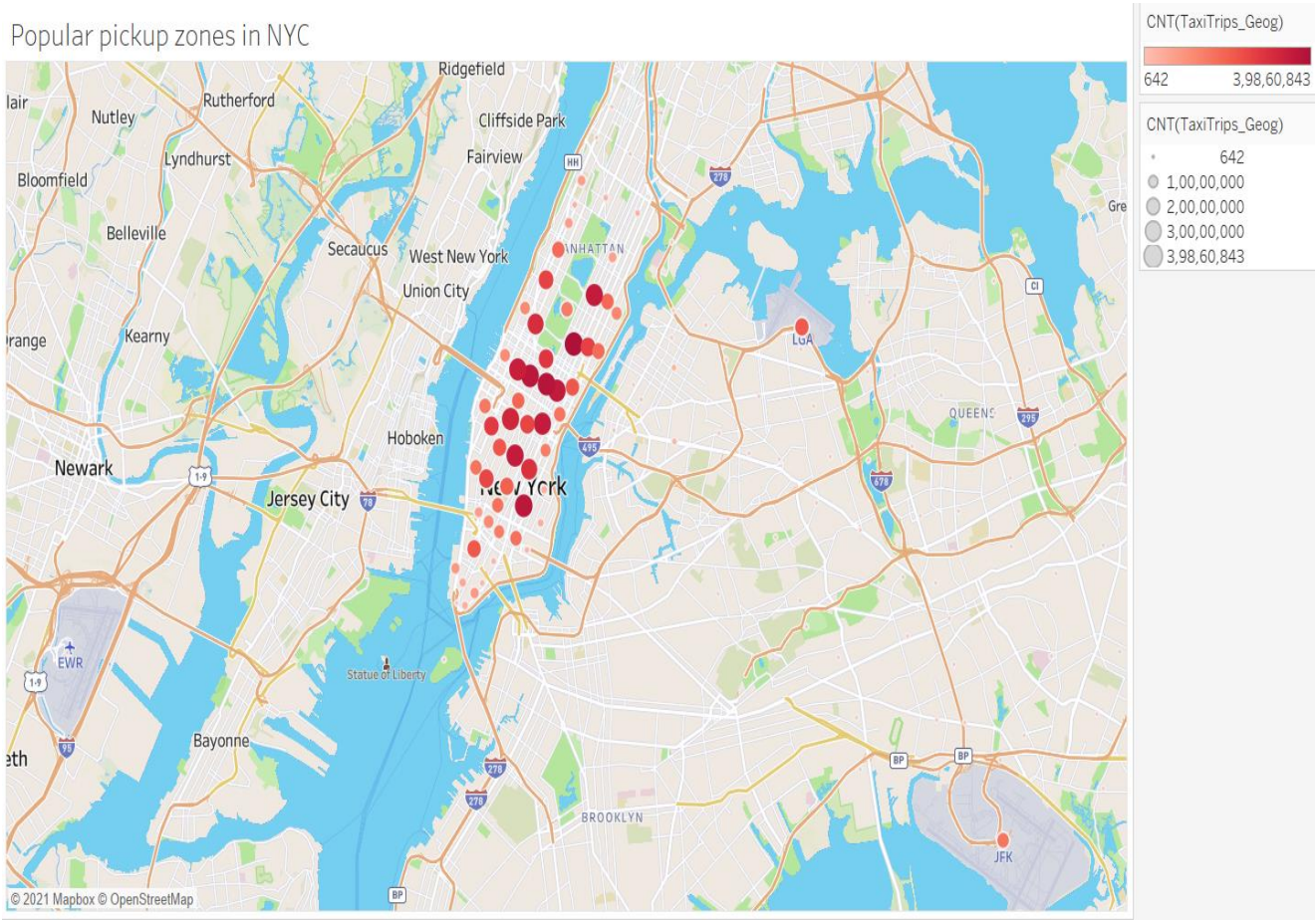
Figure 3 Popular Routes in NYC



In the above graph, we have created a new variable in Tableau that combines all the pickup zone and the drop off zones. It is then plotted against the number of trips variable to see which pickup and drop-off zone are more popular and to find the top routes in New York City. Again, the top 17 routes that we see in the visualization above are all within the Manhattan borough which can be seen on the right side of the horizontal bar plot. The top route being Upper East Side South - Upper East Side North, which has over 4 million trips and seems to be the famous

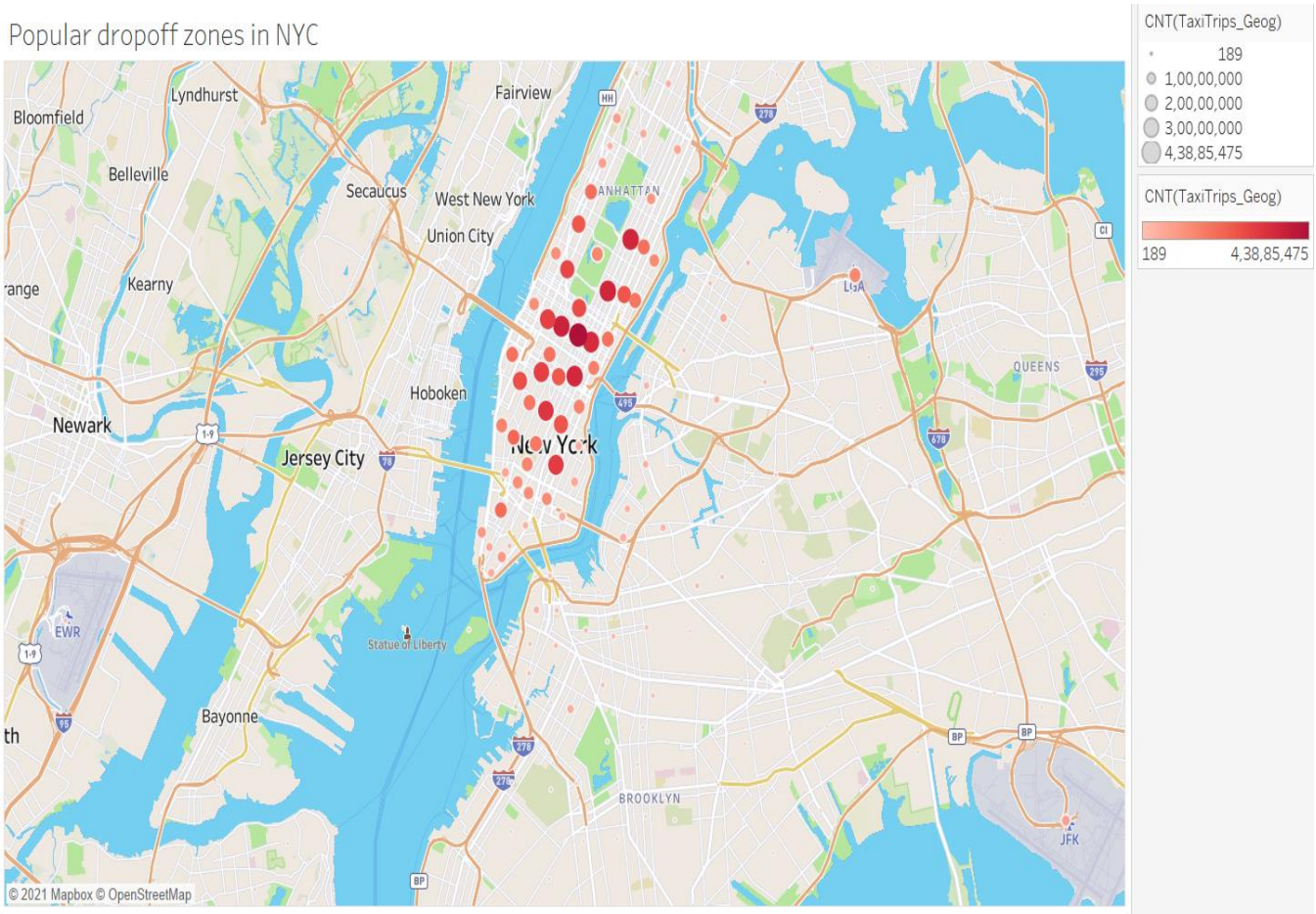
pickup and drop-off zone. This route is near the Central Park area which is one of the major attractions in New York City.

Figure 4 Popular Pickup Zones in NYC



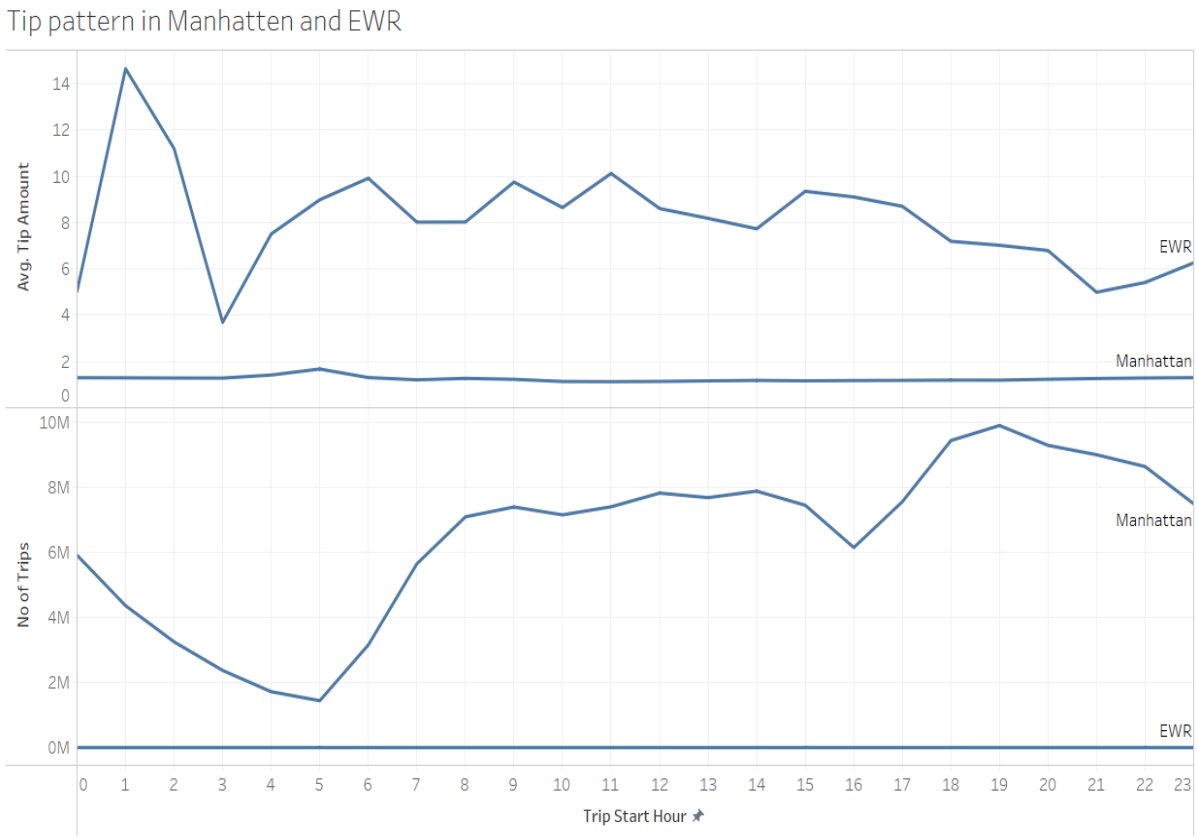
Here, we have visualized the popular pickup zones in New York City. The map feature that comes in Tableau enables the data points to be plotted over the geographic points with the latitude and longitude values. The circles show us the density of trips, the larger or darker the circle, the greater number of trips. As we can see in the above plot, majority of these circles are plotted against Manhattan borough which confirms our previous findings. The three airports in NYC are located at the top right, bottom right and bottom left in the plot. The top right is the is the LaGuardia Airport which is the closest one to Manhattan borough.

Figure 5 Popular Drop-Off Zones in NYC



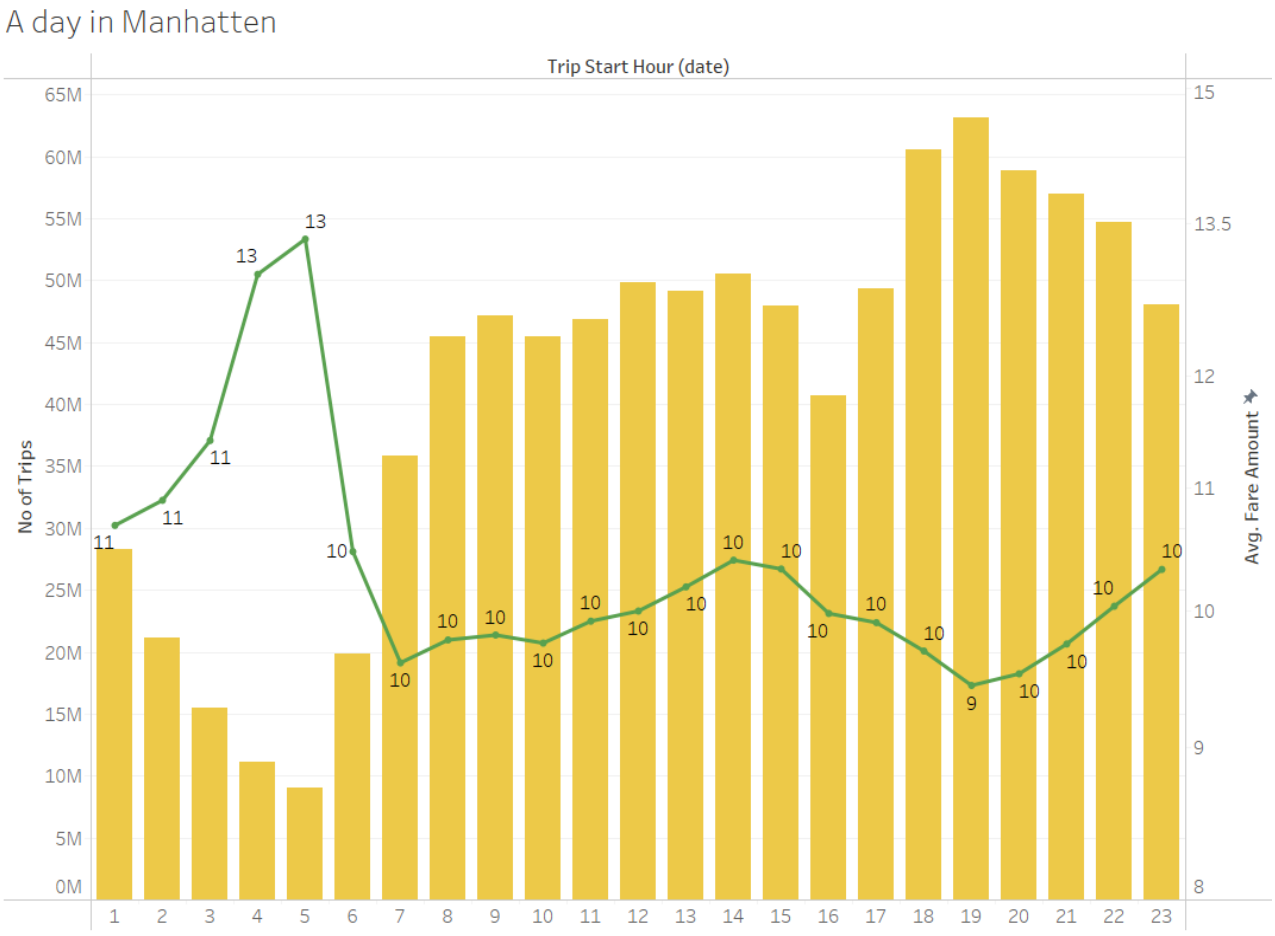
In our next visualization, we have looked at the popular drop-off zone in New York city. It can be seen that majority of points are plotted in Manhattan borough. In our earlier plot, where we looked at the popular pickup zone, too had Manhattan as the popular borough. Many people travel within Manhattan borough a lot as compared to other boroughs. It can also be confirmed from our earlier plot on the top routes in NYC, in which the top 17 routes were all within Manhattan borough.

Figure 6 Tip Pattern



In the above plot, we have visualized the tip pattern in two of the New York city boroughs. One being Manhattan, the busiest airport in NYC, and the other, EWR, the not so busiest borough in New York city. The top plot shows the Tip amount in both boroughs with respect to the time. In the bottom plot, it shows the no of trips in these two boroughs. As we can see from the bottom plot, the number of trips is considerably very low in EWR borough which lies in the lower bottom. On the other hand, we have Manhattan borough which has its highest peak at 7pm as we can see in the plot, and the lowest number at 5am which is when there are quite a few trips. But in comparison to this plot, we have a contrasting plot at the top which shows that EWR has the highest tip amount than Manhattan. We can see in the top plot, the Manhattan line lies at the bottom with the tip amount staying consistent over 2$ whereas the line plot for EWR, though has a lot of ups and downs, is very much ahead of Manhattan with respect to the tip amount. The highest being 14$ which is recorded at 1am midnight and the lowest being a little less than 4$ which is recorded at 3am as we can see in the plot.

Figure 7 A Day in Manhattan

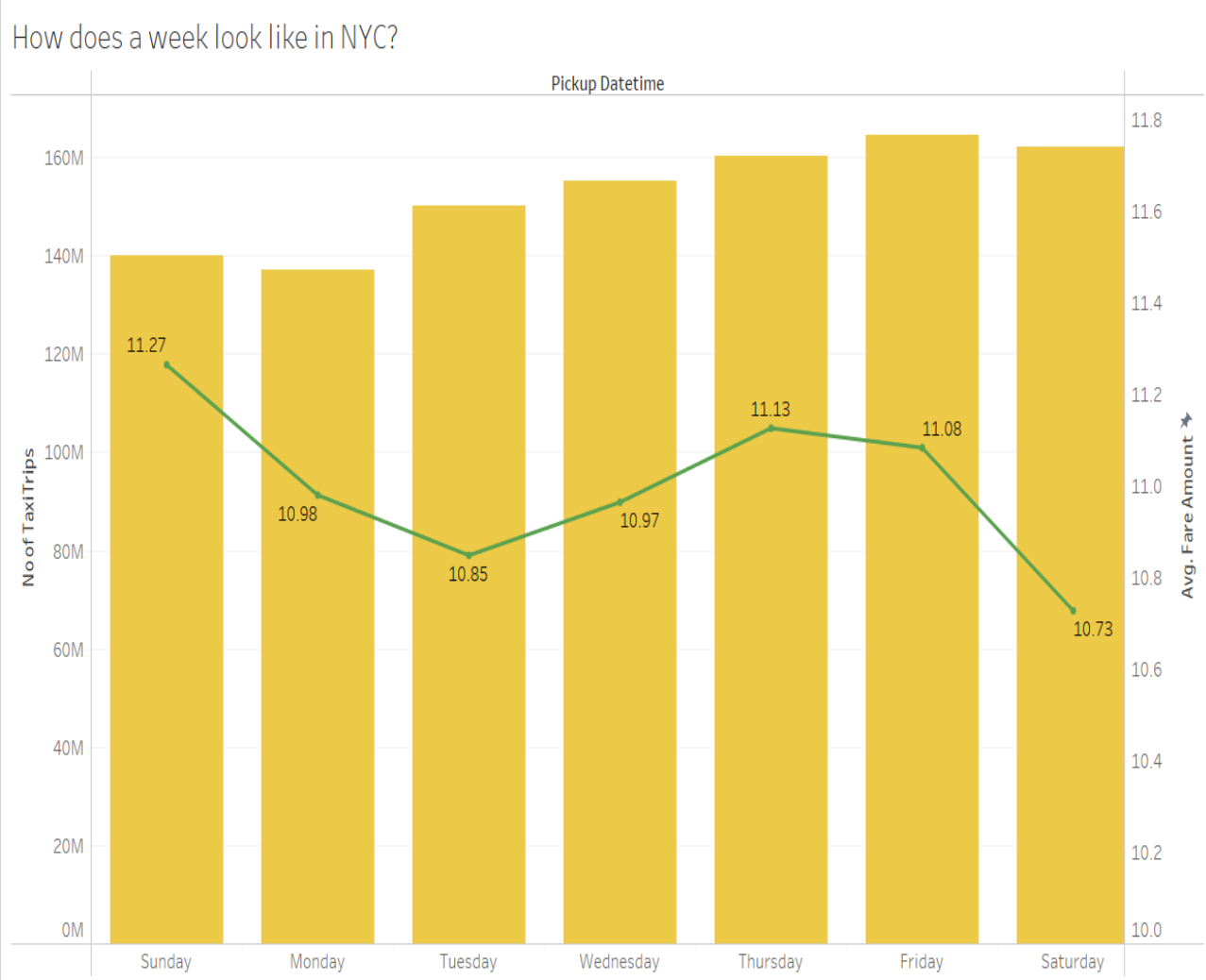


How does a day look like in Manhattan with respect to the number of trips?

The above bar plot shows the number of trips for every hour in a day. On the right, we have plotted the fare amount which is shown as a line chart on top of the bar plot. It can be seen there are very few trips in the morning at around 5 am, and the highest number of trips falls in the evening at 7pm. The more number of trips in the morning at around 9am, and in the evening at close to 7pm represents the working people who take taxis to commute to their work and home.

Also, the line plot tells us that the fare amounts are higher in the morning hours, around 3 to 5 am, while they are low towards the end of day at 6 to 8pm. Also, another insight from this plot, the fare amounts are very high during the quietist hour of the day, 5am and cheaper during peak hours, which is 7pm.

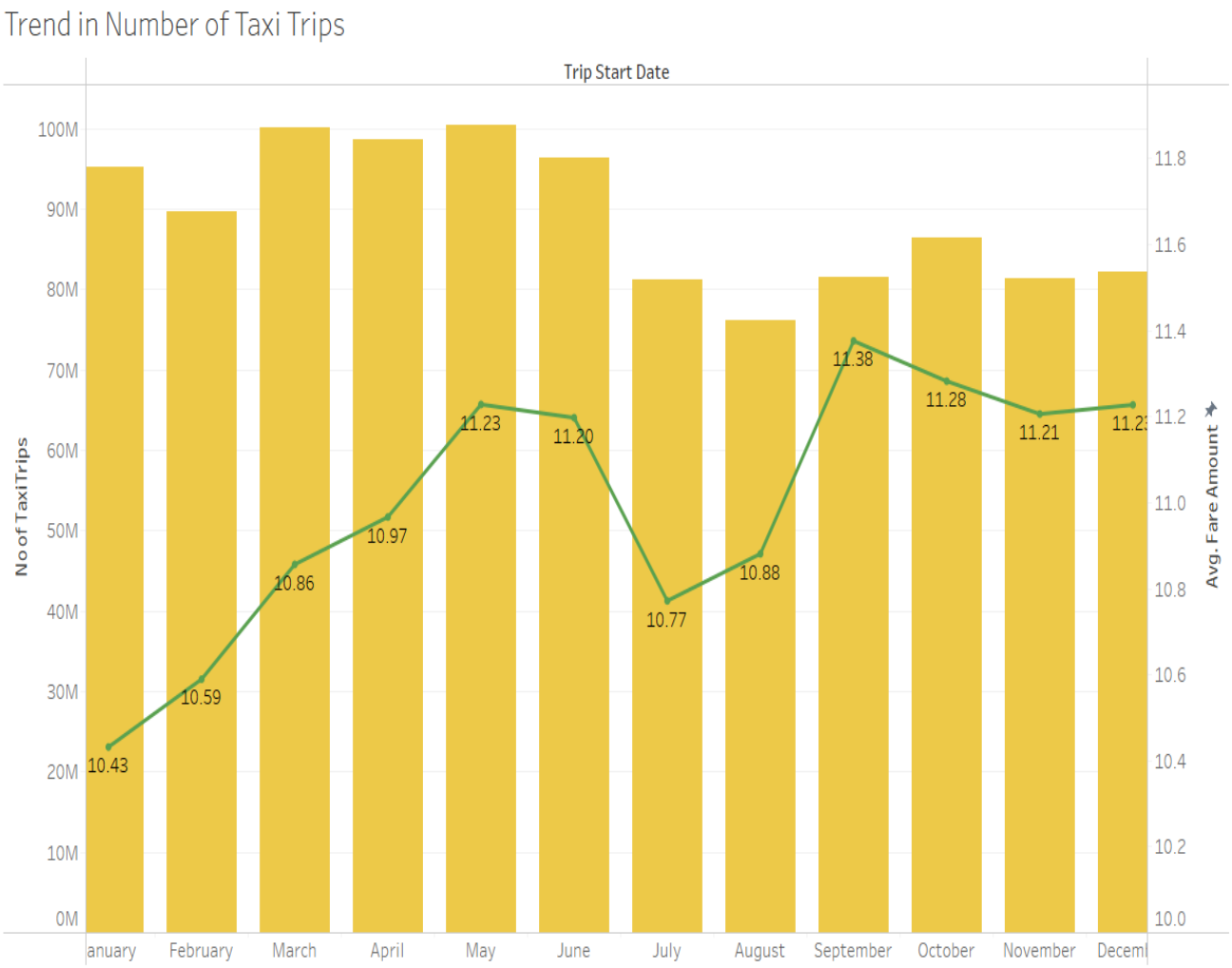
Figure 8 A Week Pattern



In the above plot, we have looked at how does a week looks like in New York City. The number of trips are plotted for every day of the week. We can see that Friday's are the busiest day, with over 160 million trips. And Monday the lowest among them with a little less than 140 million trips. The highest being Fridays and Saturdays, weekends, so the number of trips is comparatively higher than the weekdays.

Also, we have the fare amount which can be seen in the line plot. The lowest fare amount can be seen on both Saturdays and Tuesday and the highest on Sunday, Thursday, and Friday.

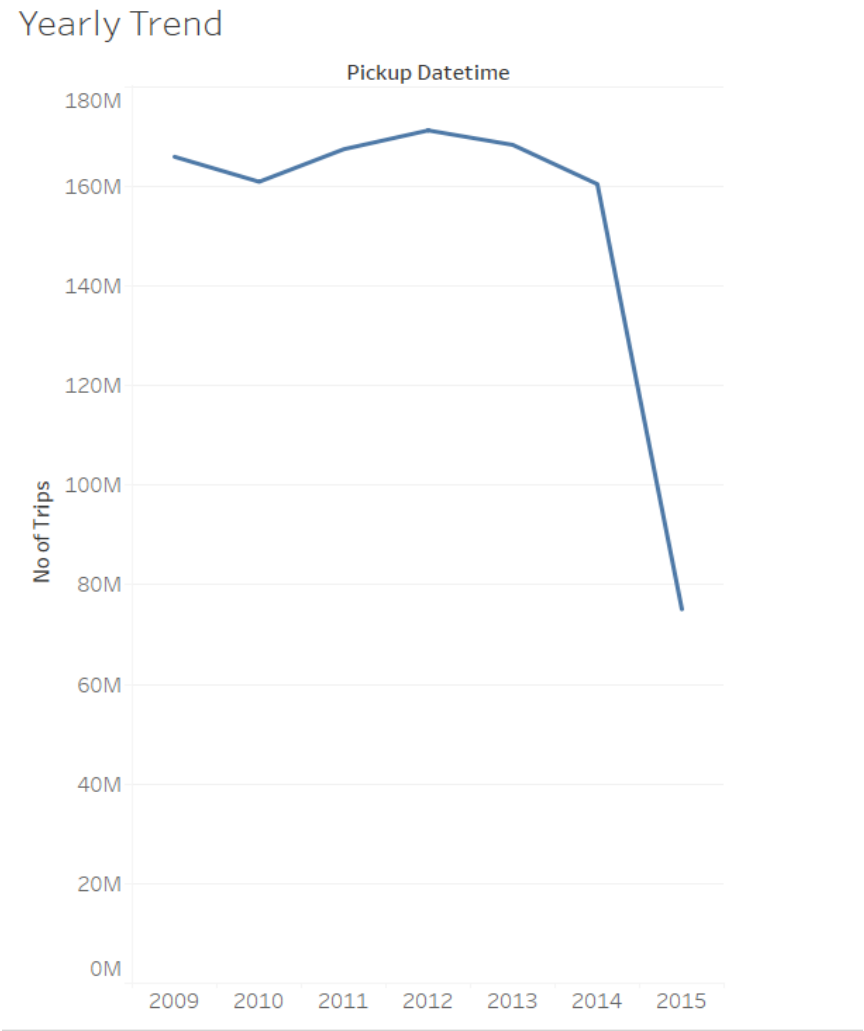
Figure 9 Yearly Pattern



Next, we have visualized the trend pattern in a year with respect to the number of taxi trips.

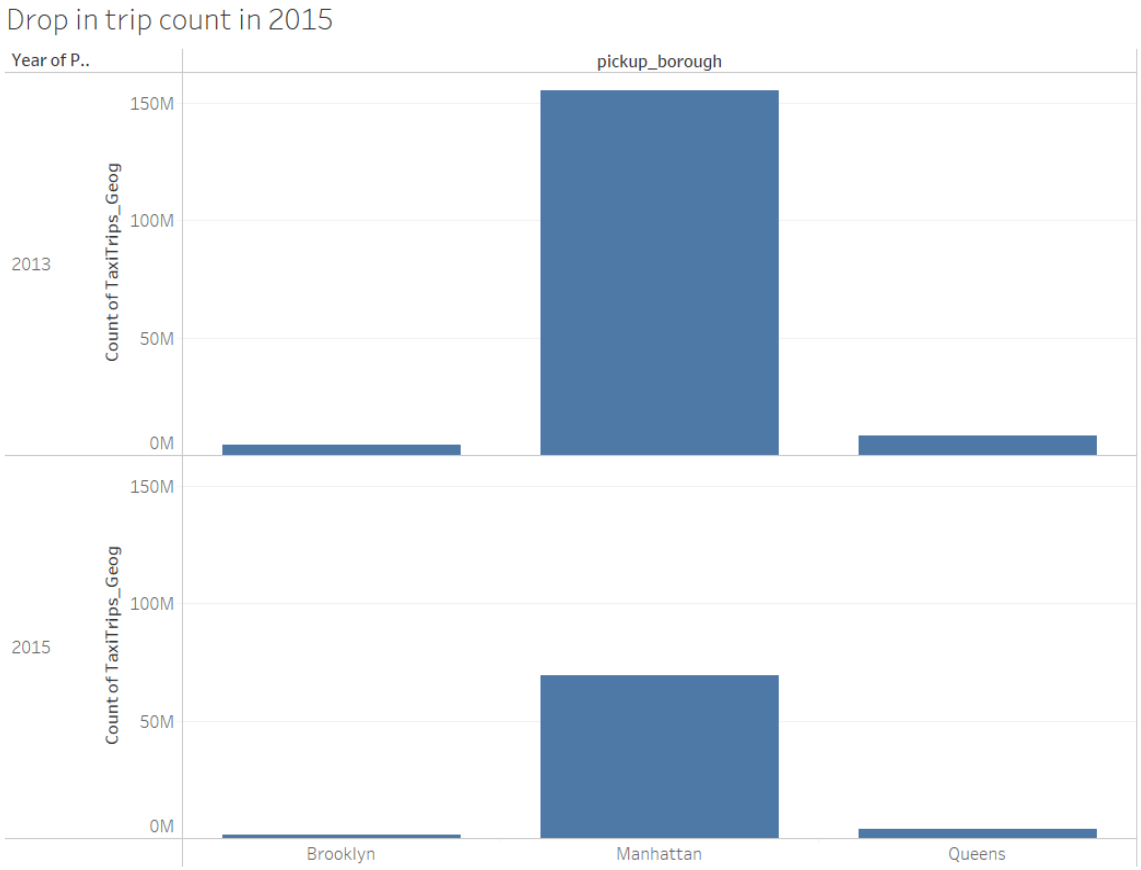
The fare amounts are plotted in the line chart. It can be seen that January has the lowest fare price with a price range of around 10.50$. The number of trips is higher in the months of March, April, May and even January. August has the lowest number of trips in a year, this can be very much related to the seasonal changes, the holiday seasons are expected to have more trips than the rest of the year.

Figure 10 Yearly Trend



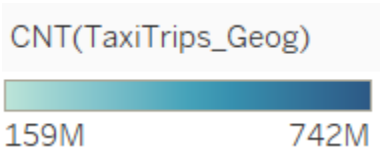
Another interesting story that we can infer from the above visualization, the Yearly trend pattern in the number of trips. Since we have the data from 2009 till 2015, we have plotted than against the number of trips to look at how the trend pattern vary. The number of trips hovered around 170 million from 2010 till 2013 and then a steady downhill after 2013. It is when UBER came into operation in New York City, in the month of July 2013. And we can see how UBER have shifted the yellow taxi operations in NYC, with over 160 million trips in 2013 to 70 million trips in 2015. The plot shows that there is a steady downhill in the number of trips for yellow taxis, ever since UBER started their operations in New York City. People prefer the UBER services more as they can book rides via their mobile phones and make it an easy and convenient process.

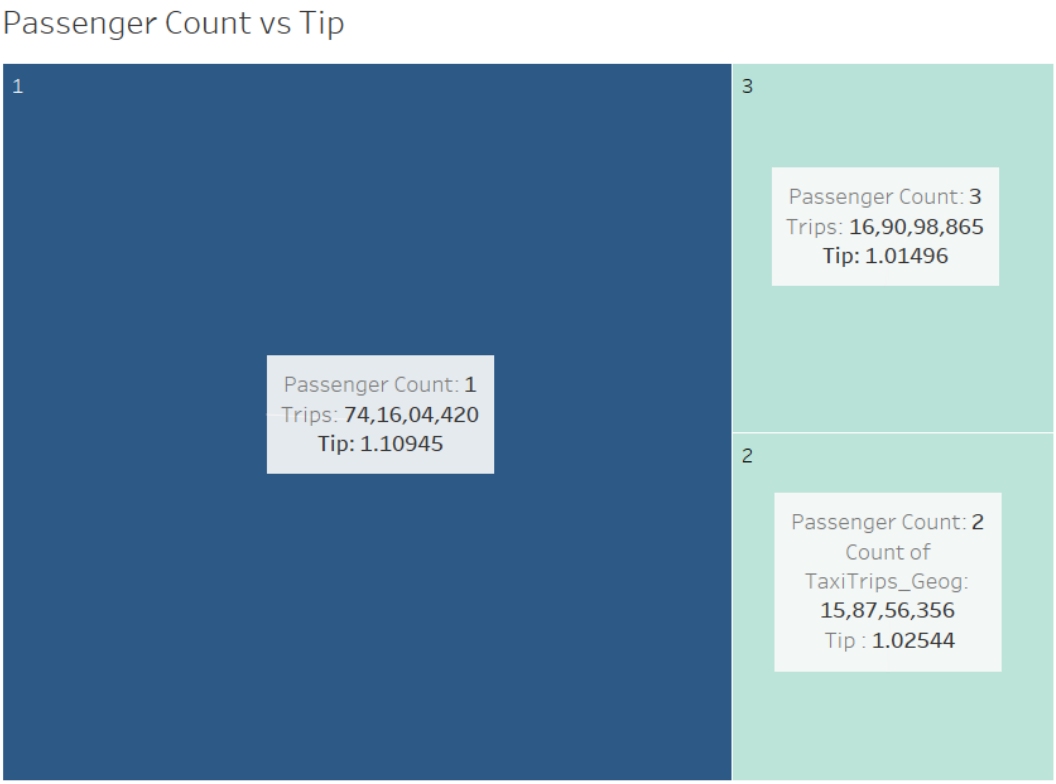
Figure 11 Trip Count 2013 vs 2015



The above visualization confirms the fact that we have seen in the earlier visualization. We've plotted the top 3 boroughs in New York City against the number of trips. The top plot represents the 2013 data and the bottom one represents the 2015 data. In the top plot, we can see that Manhattan borough has close to 150 million trips and the other two around one million trips. But in the 2015 plot, we can see from the bar plot, that the number of trips has considerably reduced. From over 150 million in 2015 to close to 55 million in 2015. The same pattern can be seen in all the three boroughs in the bottom plot.

Figure 12 Passenger Count vs Tip





In our last visualization, we have looked at how the passenger count affects the tip amount. We have plotted in the plot above with the darker the color, the greater number of trips it has. There are three different boxes with represents the different passenger counts among 1,2 and 3. As we can see, when the passenger count increases the tip amount is lower. Also, we can see that there are more trips with single passengers than two or three passengers.

8. Machine Learning Models

BigQuery ML(BQML) increases the speed of model development and innovation by removing the need to export data from the data warehouse, Instead, BigQuery ML brings ML to the data.

Categorical features of type *"BOOL", "STRING", "BYTES", "DATE", "DATETIME", "TIME"* are automatically encoded in BQML (It provides both One hot and Dummy variable encoding methods). Numerical features of type *"NUMERIC", "FLOAT" or "INT"* are standardized for both training data and future predictions.

"ML.FEATURE_INFO", "ML_TRAINING_INFO", "ML.EVALUATE", "ML.WEIGHTS" ,"ML.GLOBAL", "ML_GLOBAL_EXPLAIN" are the Support functions to query information from Machine Learning model.

Number of Trips in a location during a specific hour of the day in a month are predicted using Linear Regression, Decision Tree (Gradient Boosted Tree), Deep Neural Network and Auto ML machine learning models.

The Taxi Trips records are aggregated by grouping the trip records by the Pickup, Borough location, pick up Hour, Day of the week and Month, this aggregated data is used as the Training and Evaluation data for Machine learning model.
The aggregated data has **4,78,735** records.

Row	pickup_locationID	Trips	Month	Day	StartHour	PickupZone	PickupBorough	PassengerCount	TipAmount	FareAmount	TollAmount	MTA	ImpSurcharge
1	256	283	1	3	17	Williamsburg (South Side)	Brooklyn	1.58	0.93	12.02	0.02	0.46	0.03
2	256	1448	1	1	21	Williamsburg (South Side)	Brooklyn	1.71	1.14	11.35	0.01	0.46	0.03
3	256	324	1	5	9	Williamsburg (South Side)	Brooklyn	1.68	1.83	14.95	0.06	0.47	0.07
4	256	225	1	4	11	Williamsburg (South Side)	Brooklyn	1.52	1.27	12.71	0.02	0.46	0.02
5	256	264	1	5	11	Williamsburg (South Side)	Brooklyn	1.56	1.3	13.63	0.1	0.44	0.04

Figure 1 Aggregated data for modeling

Below table describes the features details of the model.

Feature	Data Type	Min	Max	Mean	Median	Std dev	Category Count	Null Count
Month	String						12	0
Day	String						7	0
Start Hour	String						24	0
Pickup Zone	String						260	0
Pickup Borough	String						6	0

Passenger Count	Float 64	1	7	1.74	1.67	0.61		0
Tip Amount	Float 64	0	300	1.17	0.98	1.5		0
Fare Amount	Float 64	0.01	457.5	13.7	11.9	7.69		0
Toll Amount	Float 64	0	185.5	0.35	0.07	0.81		0
MTA	Float 64	0	12.1	0.42	0.44	0.09		0
Imp Surcharge	Float 64	0	0.3	0.016	0	0.03		0

Trip Count Prediction using Generalized Linear Model (GLM):

Pickup Zone, Borough names and string values of Month, Day and Start Hour are used to force BQML to consider those features as categorical features.

Model is trained with 30% of input data as the evaluation data, the data is randomly split into Training and Evaluation. Below table shows the weights of the Categorical features in the model.

Features	Highest		Lowest	
Month	May	394.61	September	-136.70
Day	Friday	33.6	Sunday	-304.8
Start Hour	19	708.9	4	-2115.2
Pickup Zone	Upper East Side South	1851.57	Great Kills Park	-32260.92

Table 1 Weights of categorical features in GLM

The below table shows the how much influence different features have in the model; a higher value indicates a greater influence of that feature on the model.

Feature Name	Attribution
Pickup Zone	3,401.82
Pickup Borough	1,217.84
Start Hour	1,098.4
Toll Amount	894.23
Day	807.31
Passenger Count	497.8
Fare Amount	495.4
Month	239
Tip Amount	77.8
Imp Surcharge	75.7
MTA	19.1

Table 2 Feature significance for GLM

Below table shows the evaluation results of the GLM model, since this is regression model R^2, Mean Absolute Error, Median Absolute Error and Mean Squared Error metrics are used to evaluate the model performance.

Mean Absolute Error	1401.74
Mean Squared Error	7628947
Median Absolute Error	710.5
R^2	0.75

Trip Count Prediction using Gradient Boosted Tree:

Gradient boosted regression tree with a sub sample ration 50% and max iteration of 40 and learning rate of 0.3 trained. Below image shows how the training and evaluation errors were decreasing as the number of iterations went up.

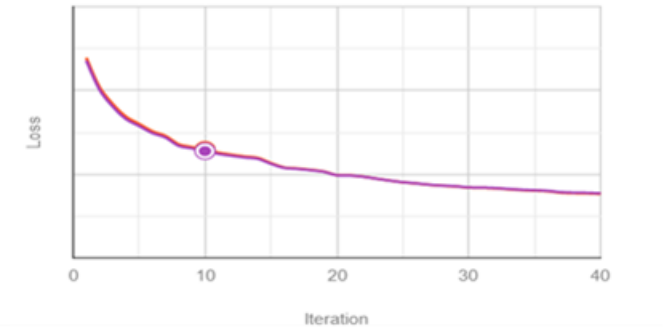


Table 3 Error during Training and Evaluation

Below tables show the evaluation metrics of the GBT model

Feature Name	**Attribution**
Fare Amount	1,411.97
Pickup Borough	1,372.44
Toll Amount	1,154.95
Pickup Zone	996.77

Tip Amount	284.8
Passenger Count	326.48
Imp Surcharge	250.58
Start Hour	214.9
Day	130.5
MTA	125.6
Month	64.89

Table 4 Feature significance in GBT model

Trip Count Prediction using Deep Neural Network:

Deep Neural Network regression model with default number of hidden layers (1 layer with no more than 128 units) is used to train the model.

Below table shows the options used for training the model:

Max allowed iterations	20
Actual iterations	9
Early stop	false
Min relative progress	0.01
Data split method	Random
Data split evaluation fraction	0.30

Below image shows how the Training and Evaluation loss have decreased with the iterations:

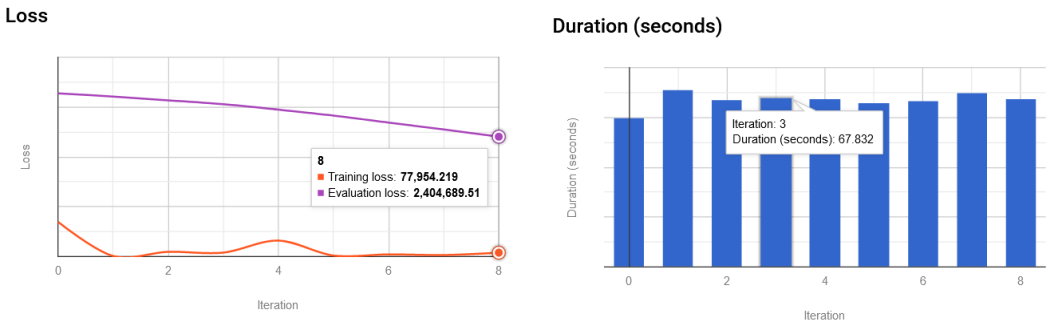


Table 5 Training and Evaluation loss in DNN

Comparison of Models:

Below table shows the comparison of evaluation results of the three machine learning models.

ML Model	Mean Absolute Error	Mean Squared Error	Median Absolute Error	R^2
DNN	632.11	2,258,572	279.52	0.92
GBT	739.4	2,379,907	247.85	0.92
GLM Regression	1401.74	7,628,947	710.52	0.75

Table 6 Comparison of three ML models

Below table lists the significance of different features in the three regression models.

Feature Name	**DNN Attribution**	**GBT Attribution**	**Linear Regression Attribution**
Fare Amount	495.4	1,476.51	24.4
Pickup Borough	1,217.8	1,456.21	5419.6
Toll Amount	894.2	1,170.1	49.1
Pick up Zone	3,401.8	809.7	17422.8
Tip Amount	77.8	284.8	1.33
Passenger Count	497.8	281.73	4.4
Imp Surcharge	75.7	253.2	66
Start Hour	1,098.4	214.9	627.2
Day	807.3	124.225	155.8

MTA	19.1	105.46	12.2
Month	239	50.84	177.2

Auto ML model:

BigQueryML supports AutoML Table model, AutoML automatically tries different machine learning models and returns the best performing model. A Regression type Auto ML model is trained with 1 Budget hour, optimizing for the loss function “Mean Squared Error”.

Below table shows the evaluation metrics of AutoML model:

Mean absolute error	188.6
Mean squared error	240,024
R^2	0.9926

9. Result Discussion

From the above modelling results, we could see that Deep Neural Network and Gradient Boosting Tree both had a similar value for R2 (0.92) while the Generalized Linear Regression model had a R2 value of 0.75. Though all three values are considered good, as anything above 0.70 in the training data is attributed to be a good model, but the values suggest that Deep Neural Network and Gradient Boosting Tree performed exceptionally well. The mean squared value in DNN is slightly lower than the Gradient Boosting tree but it doesn’t make much difference as the value is in million. So, both of these models can be considered to be good model.

Another aspect of it is the feature rank table, which we haven’t considered in the model building process as we had considerably low input features, we considered taking all of them

into the model building process. But the table actually gives a better understanding of the different features. It can be seen that pickup zone feature has got the 1st rank in Deep Neural Network. This essentially says that change in pickup zone has contributed the most in reaching the possible outcome. Though this feature is not the same when we look at all the three models. In Gradient Boosting Tree, we have Fare Amount feature as the most weighted variable holding the 1st rank and in Linear Regression mode, we have Pickup zone again as the most weighted variable in determining the possible outcome. Since we are predicting the values here, the number of trips could have been influenced by a lot of factors but considering the fact that, pickup zone holds the 1st spot in two of the models, we can very much attribute pickup zone as the most weighted variable in shaping the outcome.

The month feature has got the lowest rank in two of the three models, Deep Neural Network and in Gradient Boosting Tree, holding the 11th spot while in the Linear Regression model, we have Tip Amount holding the 11th spot. The tip won't be a deciding factor in majority of the case, as well as the month parameter which may tell us some story as we have seen earlier in the visualization plots, where we looked at how the trip count varies with respect to months, how the seasonal changes could affect the number of trips for each month, but this month feature has not been a significant feature in the modelling process as we can see from the Feature ranks. Also, the same can be inferred from the weighted feature values.

Finally, in out last model, where we have used the AutoML modelling technique that comes along as one of the model options in BigQuery. It enables to build state of the art machine learning model automatically with high speed and scale. BigQuery applies the default values for most of the model optimization parameters as well we it automatically splits data according to the dataset. We also have to specify the budget hours which determines the budget training time. And the results clearly tells us that AutoML model stood apart from all of the other three models that we discussed earlier, though Deep Neural Network and Gradient Boosting Tree models were able to achieve an R2 value of 92, AutoML achieved a score of 0.9926 as the R2 value. It is very high compared to the other models. With such a high R2 value, we conclude AutoML model to have the best model performance. Also, If we look at the mean squared error

value and mean absolute error value in AutoML model, it is very less compared to all the other three model. This again confirms the fact that AutoML stands out to be the best model.

Data drift, considered to be one of the top reasons for the model performance to degrade over time. It refers to the change in input data, due to which the model performance gets affected. In our dataset, there are variables which don't happen to change in the long run, but there may be some seasonal changes such as during pandemic we had a lot of people working virtually or some new features being added in the taxis that might affect the trained model, these effects will change the dynamics of the input parameter values and could lower the model performance.

We explored close to 1.1 billion rows in the data analysis part and visualized them using Tableau. We got to know a lot of hidden information and pattern / trends that we were able to look through by visualizing the data. One such visualization could be on the airports and how each airport differs with trip counts and what factors contribute to these drastic differences in the trip numbers. In the second visualization which we shown in the geographic map, it again confirmed the airport plot as the distance turns out to be a deciding criterion. Also, the values which we got from the machine learning models confirmed our analysis on the visualizations. Like in the day features, Friday has got the highest value, which matches the visualizations analysis on the weekdays, another feature, the upper east side south got the highest value in pickup zone which again confirms with the data analysis, where we had the upper east side south to upper east side north as the popular pickup and drop-off zone.

The BigQuery platfrom provided us the ways to work with these vast amounts of data, close to 1.1 billion rows and indeed the BigQuery ML platform to perform our modelling and analysis. A more recent data could have enabled us to visualize and see how the recent trends have changed the yellow taxi industry. The yellow taxis have also come up with E-Hail services, which could favour some target audience. But the impact that UBER has on the current market, is hard to overcome but nevertheless, the trend might change anytime.

10. References

1. https://cloud.google.com/bigquery-ml/docs/reference/standard-sql/bigqueryml-syntax-xai-overview
2. https://cloud.google.com/bigquery-ml/docs/reference/standard-sql/bigqueryml-syntax-create-dnn-models
3. https://cloud.google.com/bigquery-ml/docs/reference/standard-sql/bigqueryml-syntax-create-boosted-tree
4. https://cloud.google.com/bigquery-ml/docs/reference/standard-sql/bigqueryml-syntax-create-glm
5. https://cloud.google.com/bigquery-ml/docs/reference/standard-sql/bigqueryml-syntax-create-automl