

INFO I 535 Final Project Report

Sasidev Mahendran

Introduction

The focus of this project is to make use of the Big Data concepts taught in the course to handle, process, visualize and model a data set which fits under the descriptions of a Big Data. I have used a public dataset on Google Cloud Platform called the 'Chicago Taxi dataset'. It consists of 199,984,305 rows and 22 columns of data. It is a structured, tabular data.

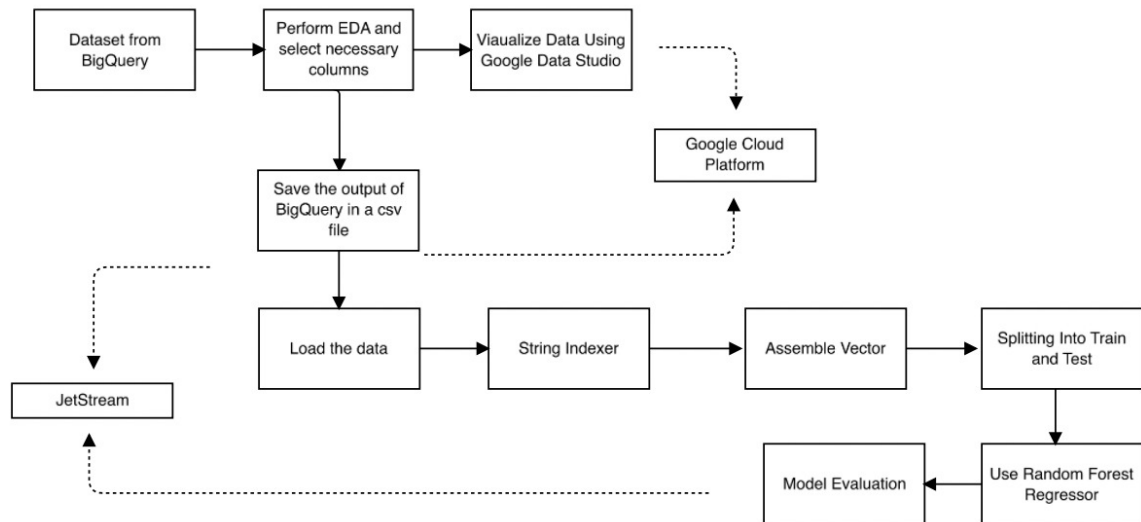
The objective is to predict how the output - 'total fare' for a trip varies based on several other inputs like pickup area, taxi company, trip distance, time. Regression based Machine Learning model has been used to predict the total fare. BigQuery has been used to filter and group data and Google Data studio has been used to visualise. Finally, the cleaning and modelling stages have been done using PySpark.

Background

I have previously worked with New York taxi dataset, and I found it interesting. Though the objective is simple - to estimate the taxi fare for a trip, there are a number of underlying factors that contribute to the fare. This makes it interesting to analyse and gather insights from the data. Another reason for selecting this dataset is that it is very huge and so allows a clear visualisation of time-bound effect on taxi trips. The data also has scope for applying many of the big data concepts. Also, the dataset allows utilisation of several functions in PySpark. Another advantage is that the dataset is structured and can be easily accessed through SQL queries.

This analysis can help taxi drivers to make optimised decision on the trips they take. It will provide them with information on which area will be the best to get a trip, which area will get more tips, which time of the day will be the best to get the best fare. Also, the taxi company can also use these data to allocate resources accordingly based on time and place. The passengers can also use this to make informed decisions on the taxi trips they take.

Methodology



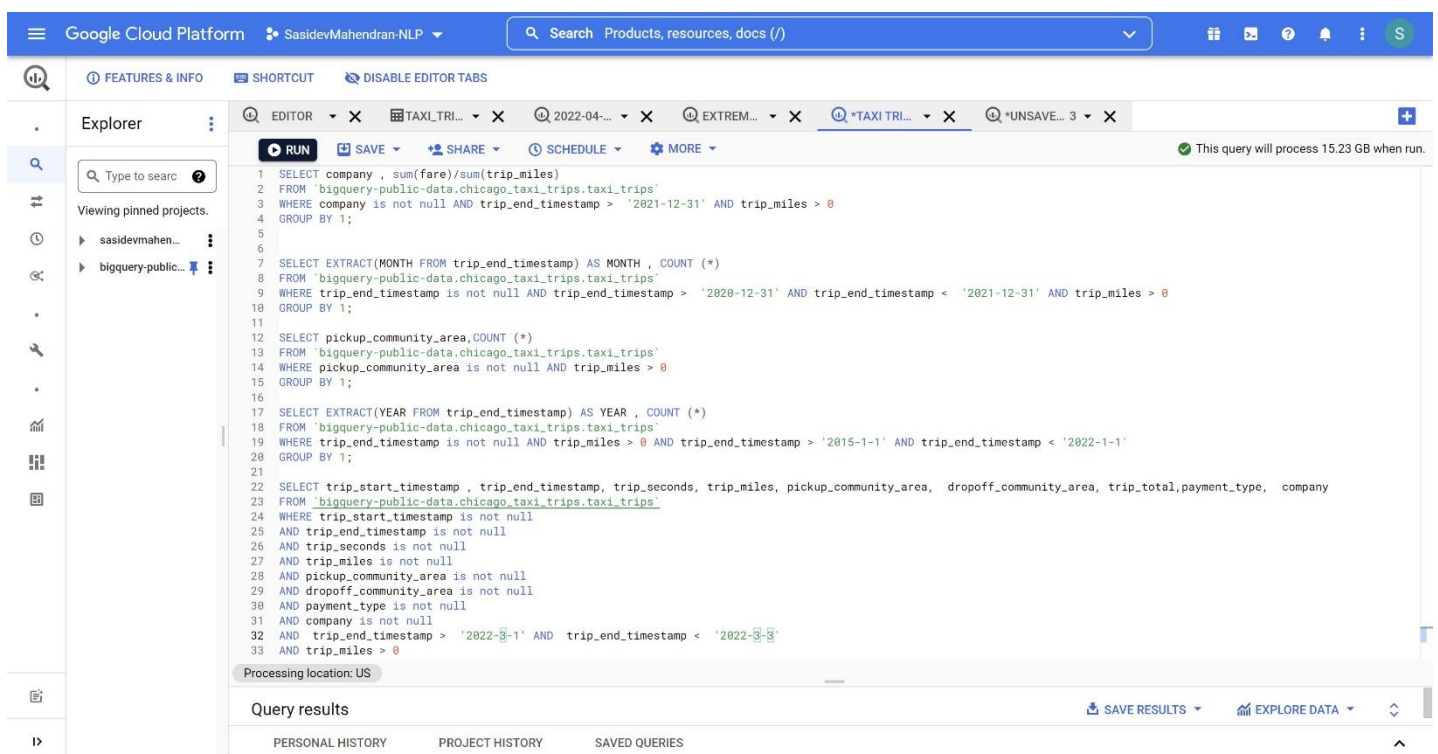
Have made use of two main resources - Google Cloud Platform and Jetstream. In GCP, I took the data set from the public dataset repository. I chose GCP because it allowed easy access to data, easy querying and visualizations and quick processing. In Jetstream, I used VM instance to create an environment to use PySpark in Jupyter notebook.

1. After creating the project in GCP, I loaded the dataset in BigQuery. This allowed easy preview of the data and the metadata.
2. Used the query editor to extract specific information that I wanted for the data. For each information extracted, I used Google data studio to visualise.
3. Implemented a query to select only the meaningful columns of data and which do not include null values. This step cut down the size of the data but still it was huge. So finally selected only the taxi trip data for 3 days - 2022-3-1 to 2022-3-3. The final data size was 22168 rows and 9 columns. Exported the final data as a csv

#	Column	Non-Null Count	Dtype
0	trip_start_timestamp	22168 non-null	datetime64[ns]
1	trip_end_timestamp	22168 non-null	datetime64[ns]
2	trip_seconds	22168 non-null	int32
3	trip_miles	22168 non-null	float64
4	pickup_community_area	22168 non-null	int32
5	dropoff_community_area	22168 non-null	int32
6	trip_total	22168 non-null	float64
7	payment_type	22168 non-null	object
8	company	22168 non-null	object

- Next, I created a VM instance on Jetstream and created a PySpark session in Jupyter notebook. Imported the csv and performed furthermore cleaning. Dropped the columns that weren't adding value to the prediction, and there were some more null values present whose corresponding rows were dropped.
- Implemented a Map Reduce function out of a dictionary that I created from the dataset. Performed String Indexing to convert categorical columns to Numerical, which is essential for implementing regression models.
- Split the data into 75% training and 25% testing subsets.
- ML model in PySpark requires only a single vector column of the input. So used an assemble vector to combine the columns.
- Finally evaluated the model using RMSE score and the value was 6.103. The model was performing well.

Performed the below queries to understand the relationship between various columns.



The screenshot shows the Google Cloud Platform BigQuery console. The interface includes a top navigation bar with the Google Cloud Platform logo, user name 'SasidevMahendran-NLP', and a search bar. Below the navigation bar, there's a sidebar with 'Explorer' and 'SHORTCUT' tabs. The main area is the 'EDITOR' tab, which contains three SQL queries. The first query calculates the average fare per mile for different taxi companies. The second query filters data for the month of December 2021. The third query filters data for the year 2022. The console also shows a status bar at the bottom with 'Query results' and buttons for 'SAVE RESULTS' and 'EXPLORE DATA'.

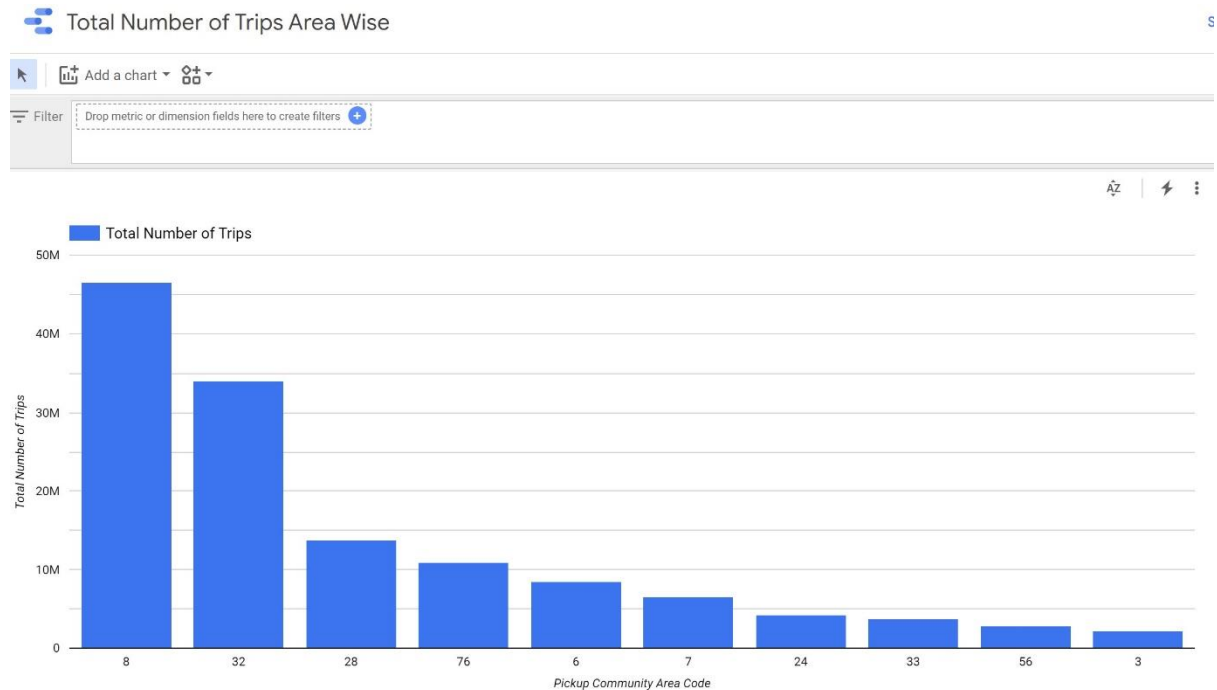
```
1 SELECT company , sum(fare)/sum(trip_miles)
2 FROM `bigquery-public-data.chicago_taxi_trips.taxi_trips`
3 WHERE company is not null AND trip_end_timestamp > '2021-12-31' AND trip_miles > 0
4 GROUP BY 1;
5
6
7 SELECT EXTRACT(MONTH FROM trip_end_timestamp) AS MONTH , COUNT (*)
8 FROM `bigquery-public-data.chicago_taxi_trips.taxi_trips`
9 WHERE trip_end_timestamp is not null AND trip_end_timestamp > '2020-12-31' AND trip_end_timestamp < '2021-12-31' AND trip_miles > 0
10 GROUP BY 1;
11
12 SELECT pickup_community_area,COUNT (*)
13 FROM `bigquery-public-data.chicago_taxi_trips.taxi_trips`
14 WHERE pickup_community_area is not null AND trip_miles > 0
15 GROUP BY 1;
16
17 SELECT EXTRACT(YEAR FROM trip_end_timestamp) AS YEAR , COUNT (*)
18 FROM `bigquery-public-data.chicago_taxi_trips.taxi_trips`
19 WHERE trip_end_timestamp is not null AND trip_miles > 0 AND trip_end_timestamp > '2015-1-1' AND trip_end_timestamp < '2022-1-1'
20 GROUP BY 1;
21
22 SELECT trip_start_timestamp , trip_end_timestamp, trip_seconds, trip_miles, pickup_community_area, dropoff_community_area, trip_total,payment_type, company
23 FROM `bigquery-public-data.chicago_taxi_trips.taxi_trips`
24 WHERE trip_start_timestamp is not null
25 AND trip_end_timestamp is not null
26 AND trip_seconds is not null
27 AND trip_miles is not null
28 AND pickup_community_area is not null
29 AND dropoff_community_area is not null
30 AND payment_type is not null
31 AND company is not null
32 AND trip_end_timestamp > '2022-3-1' AND trip_end_timestamp < '2022-3-3'
33 AND trip_miles > 0
```

- The first query was used to retrieve the various taxi companies and their average fare per mile
- The second query was used to filter the month wise data for the year 2021.
- The third query was used to get the total number of trips based on pick up community area

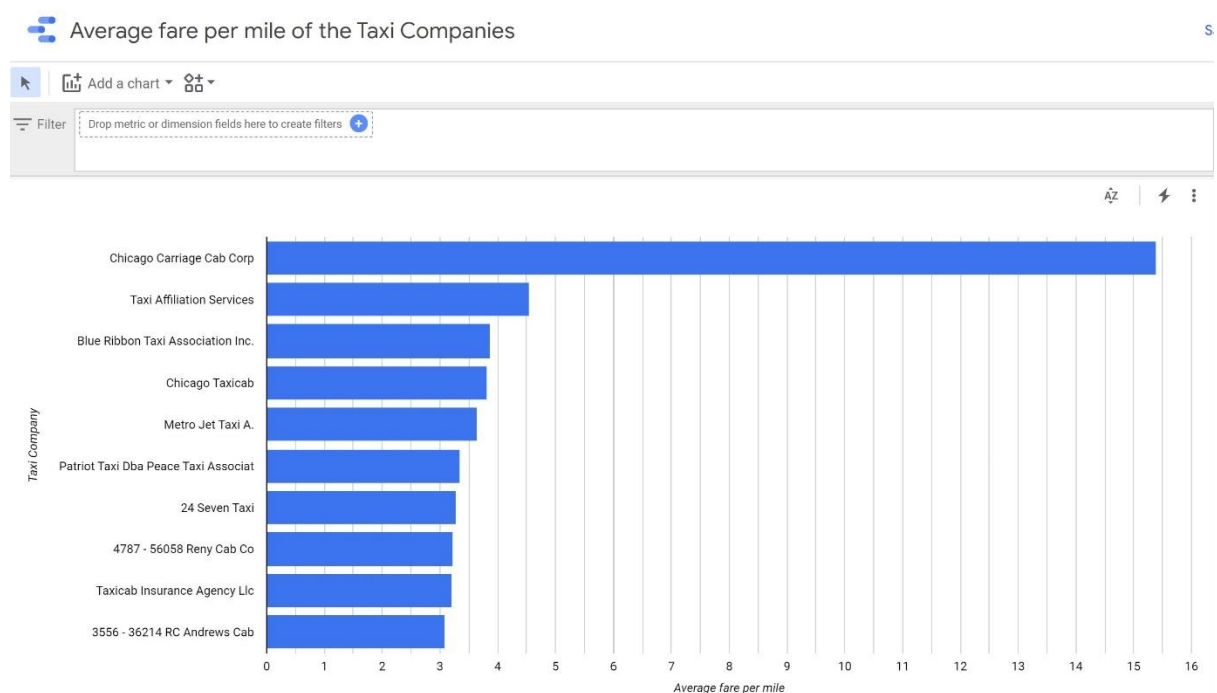
- Fourth query was used to select year wise taxi trip details.
- Final query was used to select required columns without any null values, for the required time period of three days.

Results

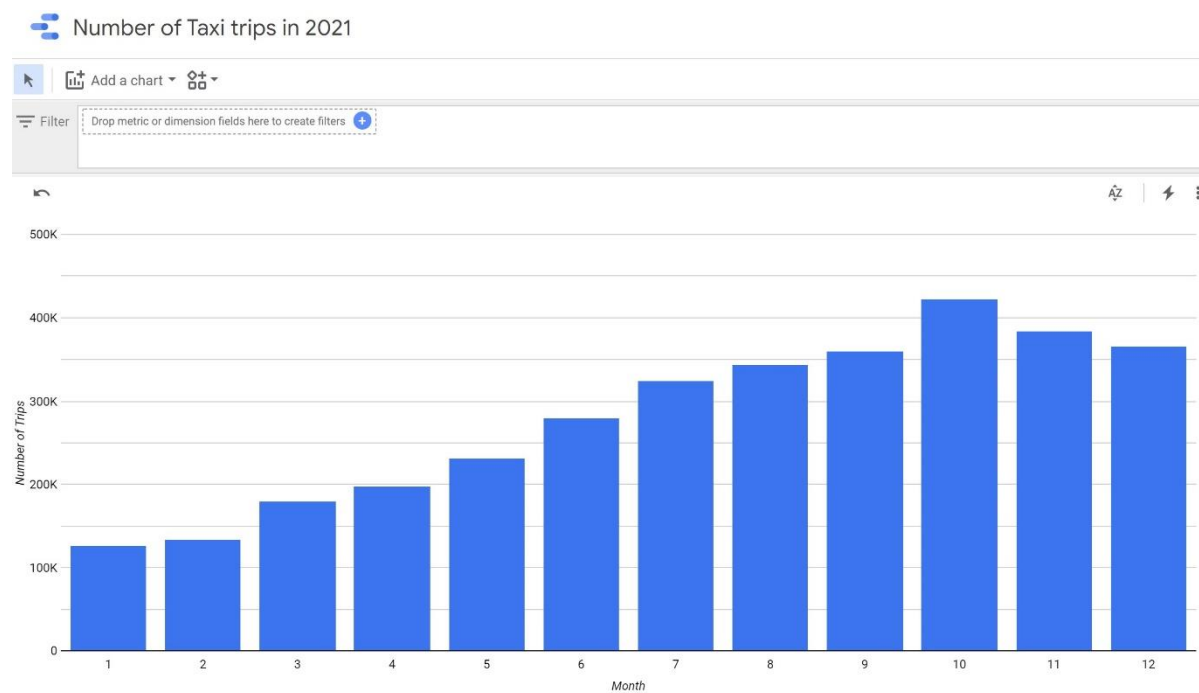
The number of trips is highest in the pickup community area 8 and 32



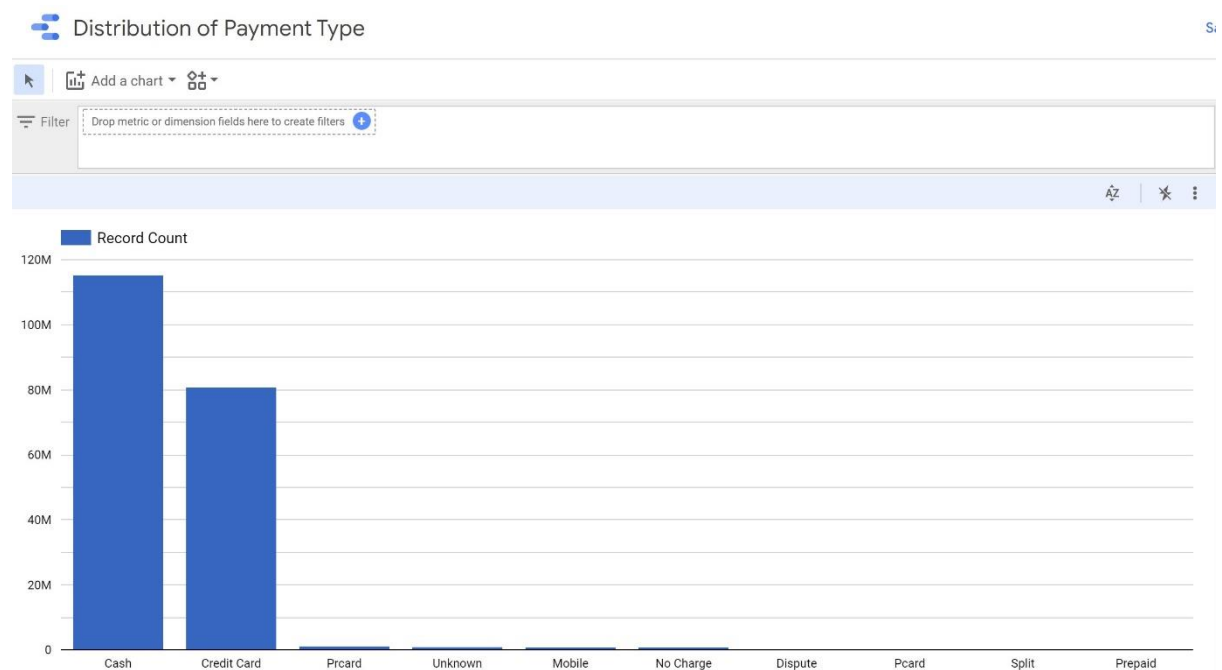
The average fare per mile is the highest for Chicago Carriage Cab Corp. It is close to 15.5 dollar/mile



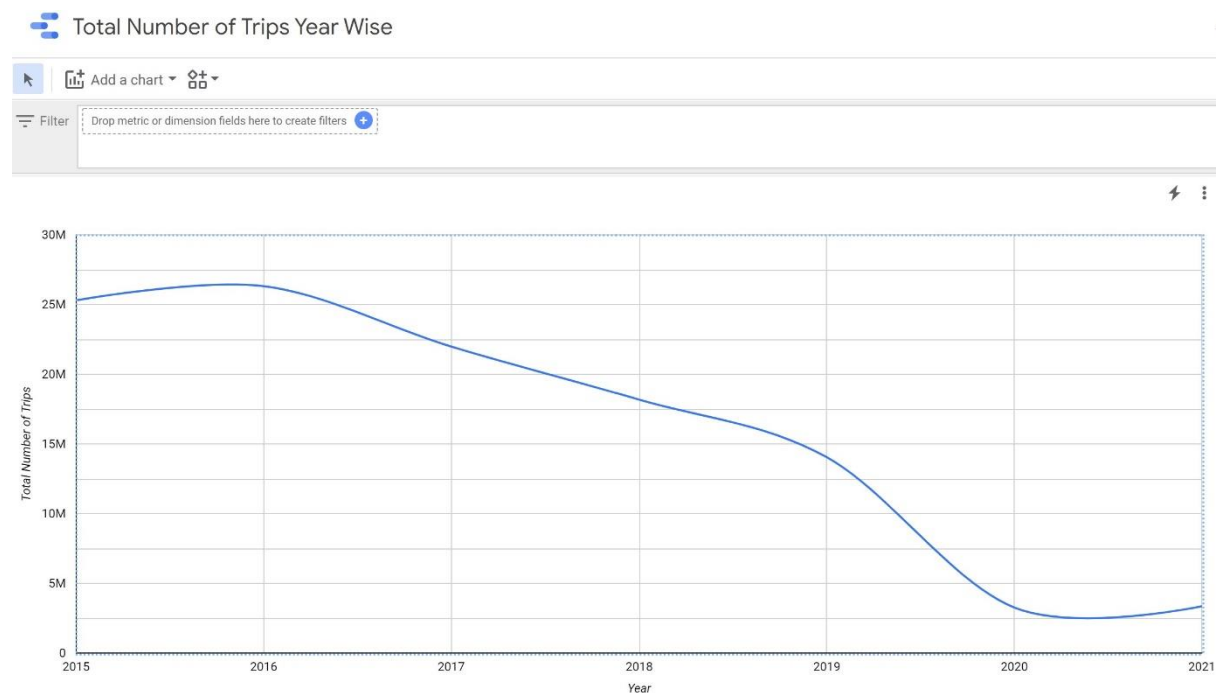
The number of trips is highest in the month of October for the year 2021.



The most common form of payment type is cash, followed by Credit Card



Between 2015 - 2021, the number of trips is highest for the year 2016 and lowest in 2020, 2021.



Final RMSE score of the model

```
pred.select("prediction","trip_total","features")
DataFrame[prediction: double, trip_total: double, features: vector]

evaluator = RegressionEvaluator(labelCol="trip_total", predictionCol="prediction",

rmse = evaluator.evaluate(pred)

rmse

6.10380290737564
```

Discussion

Interpretation of the visualization results:

1. To get a trip, it would be best for a taxi driver to target pickup areas that come under the code 8 and 32. Majority number of trips are from these areas.
2. If the passengers are looking for a cheaper ride, then it is better to avoid the Chicago Carriage Cab Corp as it has the highest fare per mile.

3. The taxi trips get affected by seasonality. Drivers and companies can anticipate lower number of trips in the winter, especially in January and February (This particular data might be biased because of COVID)
4. Cash and Credit Card are the most commonly used payment methods and there can be some methods employed, like giving offers for these two payment methods to attract more customers.
5. Taxi Trips are the lowest in 2020 and 2021. This might be because of the effect of COVID.

Interpretation of Model results:

1. The Random Regressor model performs well and has a low Root Mean Squared Error of 6.1. Thus, the model can be used with confidence to predict the total fare.

Employing the Technologies/skills from the course:

Connecting different aspects of the project to the course modules

1. From the 3Vs of big data, this dataset can be related to Volume. The size of the data is very huge (199,984,305 rows) and indirectly implies the high at which the data must be collected.
2. The dataset is in the CSV file format, in the row-based structure.
3. Have used a Virtual Machine Instance using Jetstream
4. Implemented parallel processing by means of MapReduce. Created a key-Value pair from the dataset and used Map Reduce to map the input to several nodes and finally combine and reduce the results. This is basically within the Hadoop framework and accesses the data stored in HDFS

```
dict = {row['payment_type']:row['trip_total'] for row in df.collect()}
print(dict)

{'Prcard': 15.25, 'Unknown': 42.25, 'Credit Card': 50.75, 'Cash': 42.25, 'Mobile': 15.52, 'Dispute': 11.25, 'No Charge': 21.5}
```

```
p = sc.parallelize(dict)
```

```
p.mapValues(lambda x: (x, 1)).reduceByKey(lambda x, y: (x[0] + y[0], x[1] + y[1]))
```

```
PythonRDD[111] at RDD at PythonRDD.scala:53
```

5. Have followed the below pipeline to implement the model:
 - **Plan:** Planned on the type of data, the scope of the project and the resources to be used.

- **Acquire:** Data was acquired from the Public dataset repository in Google Cloud Platform.
 - **Process:** The Data set was processed initially in BigQuery using SQL. After cleaning and transforming, it was further processed using PySpark in Jetstream.
 - **Analyze:** The data was analysed using Google Studio and PySpark
 - **Preserve/Publish:** The model will be preserved and Published in Github
6. Since the data is structured Relational database, have used SQL to retrieve data.
 7. Have used the following functions in PySpark - StringIndexer, VectorAssembler, randomSplit, evaluator, Pipeline.

Difficulties Faced:

1. Was interested in finding if there was any bias based on the area of trip pick up (pickup_community_area) using the AI Fairness 360. But it was not feasible to implement using this dataset.
2. Faced Permission issues with Google Cloud Platform. Had to change the roles for each section.
3. Tried to use a PySpark in a GCP cluster. But was not able to because of some error. So had to switch to Jetstream. This consumed a lot of time
4. Jetstream was slow. Had to reopen the session and restart the jupyter kernel multiple times.

Conclusion

The project enabled use of multiple big data concepts and tools. As seen in the discussion, a number of modules and topics covered in the course were used to work on the Chicago taxi dataset. It was truly a big data problem in the sense, it wouldn't have been possible to analyse the dataset without any of the concepts covered in the class. The use of cloud platform and Virtual Machines allowed to process the dataset in a very efficient way.

The Regression model was trained for the dataset and now can be used to make prediction of the total price based on the inputs. Though the dataset was limited to 3 days because of project limitation, the scope can be increased by including the entire data.

References

1. <https://sparkbyexamples.com/pyspark/pyspark-maptypes-dict-examples/>
2. <https://databricks.com/glossary/mapreducehttps://databricks.com/glossary/mapreduce>
3. <https://www.silect.is/blog/random-forest-models-in-spark-ml/>
4. <https://stackoverflow.com/questions/69293182/map-reducing-key-value-tuples-where-value-is-a-dictionary-in-spark>
5. <https://www.analyticsvidhya.com/blog/2021/05/9-most-useful-functions-for-pyspark-dataframe/>