APRIL 25, 2021

# BDE- AT1
# HANDOVER DOCUMENT

## NEW YORK TLC TRIP RECORD DATASET

ABHISEK GAUTAM (13679042)
UNIVERSITY OF TECHNOLOGY SYDNEY
abhisek.gautam@student.uts.edu.au

# SUMMARY

This report attempts to give an overview on the dataset of New York TLC Trip Record. The report gives a method of setting up the environment. It then explains about different steps undertaken during this analysis, including the nature of the dataset, its features, its shortcomings and different methods applied to clean it. Further, it explains why the data was loaded in parquet format and how it was done. It then talks about the converted data and queried with different business questions and their answers. Lastly, how a ML model model was trained, and how did it perform while testing it with 3 month's data. Finally, it gives some recommendations and future steps that can be done with the project.

## Project Setup

The project is setup in a Docker environment setup with Spark and Hadoop, with Jupyter service. Coding is done in Jupyter notebooks, each notebook for specific purpose. Firstly, this repo (link: [https://github.com/bde-uts/bde/tree/main/bde_lab_4](https://github.com/bde-uts/bde/tree/main/bde_lab_4)) needs to be cloned. Then, an access key and a secret need to be obtained from AWS console for accessing the source data from S3 bucket. These should be put in a .env file in the project's main folder. Now, the cloned docker environment can be started by using the docker-compose file. After the environment gets started, the process is straightforward by accessing Jupyter notebooks hosted on the docker environment. Then, the script labelled "ETL" needs to be extracted to perform ETL on the data, which will load it in a parquet file format in the local hard disk. There are separate scripts for SQL queries, EDA and to performing modelling which can be found on the notebooks folder.

## Project Overview

This project involves analysing a large dataset using Spark. It includes extracting the data, performing cleaning and transformation, and then querying to get answers to a few business questions, and lastly building a machine learning model with to see how its predictions. The decisions undertaken while cleaning, performing transformations, problems encountered while dealing with the massive dataset and its solutions are described here.

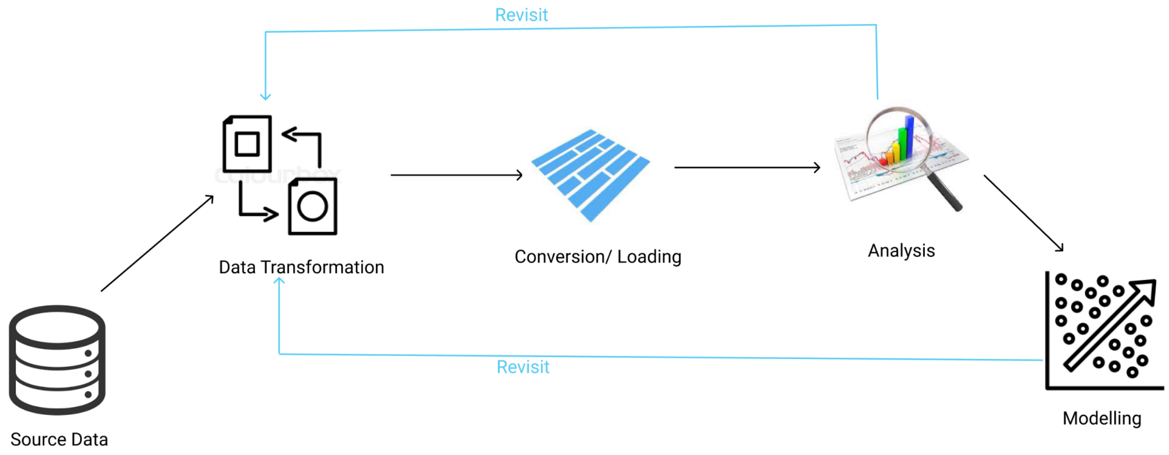


**FIGURE 1- AN OVERVIEW OF DATA FLOW**

This report will go through each step in the data flow and describe them.

## Source Data

The data is sourced from New York's Trip Record data. It contains the taxi trip records from 2009 till 2020. The data files are present in CSV format, separated by months. Initially, they only had Yellow taxis operating in the region. After 2013 August, there were Green taxis too. Green taxis were only allowed to have Dispatched rides and were not allowed to have street-hail rides. After 2013 August, the Green Taxis also could do street-hail rides.

The size of the CSV file varies between Yellow and Green cabs. Yellow Cabs have the files in the order of 2GB with around 10 million data points for each month. The file size of Green cabs ranges from 500MB to 900MB, with around 1 million data points each month.

The task of data analysis needed to be done for 2 years, 2015 and 2016. Approximately 250 million data points in total and a file size of around 60 GB was there. This was a massive amount of data. A simple aggregation query on one month's data would take more than 15 minutes. The data was in CSV format, thus taking a long time to read. They have a good readability, but perform very badly if data size is large.

## DATA TRANSFORMATION

A number of issues were found in the source data. The data is very big, so it was decided with my team that it was okay to drop faulty ones as dropping a couple of thousand would not have much impact. So, initial data cleaning was done by solely querying one month of the Green Cabs data.

Data cleaning and transformation was an ongoing task as faults were found in analysing it during the later stages. This process was done for four to five times to obtain a reasonably cleaner dataset. Filters applied on the data is given in Table 1.

| **Transformation/Filter Applied** | **Justification** |
|---|---|
| 1. Passenger count > 0 | There were a few data points where passenger count was 0, and also negative. If there is a taxi ride, then there must be 1 or more passengers. |
| 2. Trip Distance between 0 and 50 | If there is a trip, the distance must be greater than 0 miles. Also, since the greatest distance between two points in NYC is 35 miles, a trip cannot be more than 50 miles.<br>https://www.walksofnewyork.com/blog/nyc-by-the-numbers |
| 3. Total Amount between $2.5 and $100 | The total amount must be greater between $2.5 and $100. There were some instances where the total amount was less than 0, which was applied to cancel the previous transaction, but we decided to filter these corrections out.<br>Rate of taxi is: $2.50 initial charge, plus 50 cents per 1/5 mile when traveling above 12mph or per 60 seconds in slow traffic or when the vehicle is stopped, plus 50 cents MTA State Surcharge for all trips that end in certain places.<br>https://www1.nyc.gov/site/tlc/passengers/taxi-fare.page |
| 4. Payment type is not 6 | Payment type 6 refers to voided trip, hence were filtered out. |
| 5. Duration of the trip > 0 | The dataset has pickup and dropoff datetime. All the instances, the pickup time should be less than the dropoff time. |
| 6. Rate Code ID is not 99 | There were a few trips with non-available Rate Code ID, so they were omitted. |
| 7. Speed between 2mph and 70mph | Speed should be between 2mph (which is less than walking speed and max allowed speed, which is 65kmph according to the Wikipedia article- Speed limits in the United States (referred sources not available).<br>https://en.wikipedia.org/wiki/Speed_limits_in_the_United_States |

**TABLE 1- FILTERS APPLIED TO THE DATASET**

Some new features were engineered for later use in this step as well, which are shown in Table 2.

| **New Feature** | **Details (of each record)** |
|---|---|
| *taxi_colour* | Tells whether the trip was done by yellow or green cab |
| day | Day of the month |
| month | Month of the trip |
| year | Year of the trip |
| week | Week number in the year |
| *pickup_hour* | Hour of pickup in 24 hours |
| *duration_mins* | Trip duration in minutes |

| | |
|---|---|
| *cat_duration* | Category of duration |
| *speed_mph* | Speed of the vehicle (distance/duration) in mph |
| *trip_type* | Type of trip- 1 for yellow cabs; green cabs have this in the source |

**TABLE 2- NEW VARIABLES ADDED TO THE DATASET**

## Data Loading

Now that the data is transformed and filtered, it is required to load the data. Since the source files are massive and they are very heavy to load at once, the source data was read one file at a time, transformed and then loaded into parquet file format. Having data in parquet format offers the following advantages:

i. The data is kept in a column format. This helps to speed up time in fetching and filtering as the non-relevant data can be filtered quickly. It also quickens aggregations in the data.

ii. Parquet file format is built from ground up, hence is built to optimize queries on large volumes of data.

iii. The data is stored in binary format, so it is very compressed, hence saves space in the disk where it is stored.

iv. It is built in a manner that allows parallel queries to run simultaneously, thus our spark queries would also run in parallel.

The disadvantages of parquet file format are given below:

i. The data is not in a readable form. This is because data is stored in binary format, hence there is no option to read the data other than querying it.

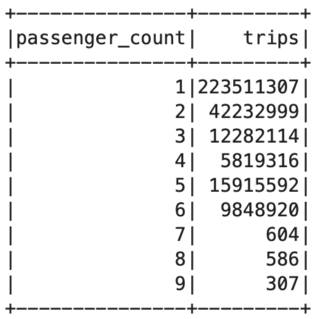
ii. Data transformations and joining operations may take longer or similar time as row-based formats since the data needs to be kept in the memory and referred to.

The transformed data was loaded into parquet file format and saved in local hard drive in a monthly manner. The size on disk after loading the full data was only 5.62GB, comparing to a staggering size 60GB of the CSV files.

## Findings

There were some interesting facts discovered while querying the entire filtered dataset. First of all, the number of trip records has actually decreased from 2015 to 2016; the first months of 2015 had close to 15 million trip records, which decreased to 11 million towards the end of 2016.  Number of trips was found to increase in the months of March and April, which can be attributed to the start of spring season. Moreover, it was observed that Saturday had the most trips. The busiest hours for the taxi were on 18th and 19th hours of the day, i.e., 6pm and 7pm.

The number of passengers in a trip was ~1.67 indicating that there were many passengers. Looking at the number of trips grouped by passengers, it was found that the highest number of passengers was 1 with 223 million trips, followed by 2 with 42 million.



```
+---------------+---------+
|passenger_count|    trips|
+---------------+---------+
|              1|223511307|
|              2| 42232999|
|              3| 12282114|
|              4|  5819316|
|              5| 15915592|
|              6|  9848920|
|              7|      604|
|              8|      586|
|              9|      307|
+---------------+---------+
```

**FIGURE 2- NUMBER OF TRIPS BY PASSENGER COUNT**

A trip in average earns the driver between $15 and $16, and the average spending of a passenger per trip is $13. It was seen that 41% of the green cab drivers and 62% of yellow cab passengers tip the driver. Among the tips provided for yellow cabs, 3.04% are more than $10. In the case of green cabs however, 1.71% of the tips are more than $10.

Given the trip durations, it was observed that the average speed in kmph was the largest for trips above 30minutes. This is true because trips more than 30 minutes long would include travelling on freeways. These trips also had the largest earnings close to $0.4 per km.

From the boxplot in Figure 3, it is seen that the highest amount of median earning was from RateCodeID 2, 3 and 4, which were the rate codes for JFK, Newark and Nassau or Westchester Airports. These trips also would be of long distances because the airports are away from residential places, thus resulting in drivers earning more.

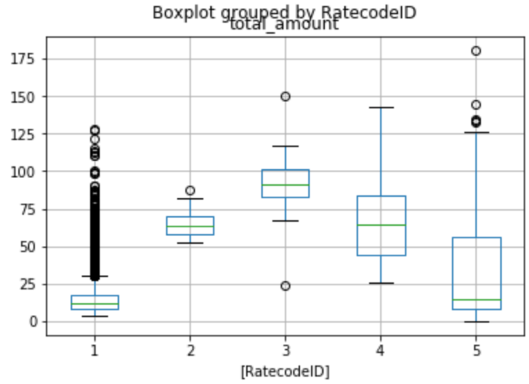


**FIGURE 3- VARIATION OF TOTAL AMOUNT BY RATE CODE ID**

## MODELLING AND EVALUATION

A predictive model needed to be built which gave a forecast of the trip amount for the last 3 months, i.e. 2016 October to 2016 December. 50,000 datapoints were randomly sampled and analyses were done. From Figure 4, a strong correlation was found on *total_amount* of *trip_distance* (0.94) and *duration_mins* (0.85). *Duration_mins* also has a good correlation with *trip_distance* (0.79). Thus, it is not wise to select both *trip_distance* and *duration_mins* for modelling due to high collinearity amongst themselves. So, models are created by choosing *trip_distance* only, and both *duration_mins* and *speed_mph*.

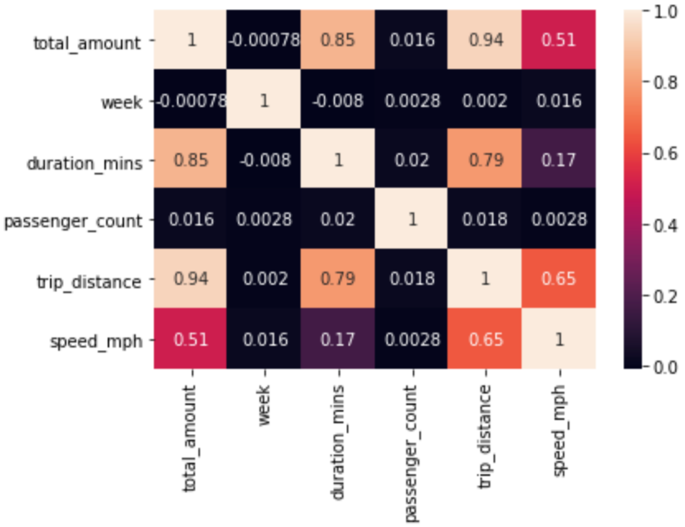


FIGURE 4- CORRELATION AMONG THE NUMERIC VALUES

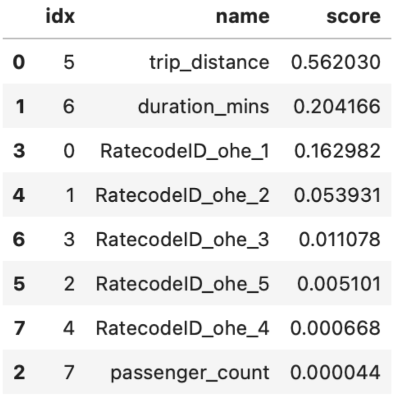
Among the categorical variables, only RatecodeID seemed to have some variation with the *total_amount*, and hence was chosen for the model (Figure 3).

Two ML algorihms were implemented- a general linear regression (glm) and a random forest model with a linear regressor. The Random Forest model gave the best RMSE score on the test data (Table 3).

| Model | RMSE Score | Training time (approx.) |
|---|---|---|
| Initial result of Random Forest Linear Regressor | 42.97 | 50 minutes |
| Final Random Forest Linear Regressor model | 4.06 | 50 minutes |
| GLM model I | 4.52 | 10 minutes |
| GLM model II | 4.33 | 10 minutes |

TABLE 3- MODELS, THEIR RMSE SCORES AND TRAINING TIME

The best model was found to be Random forest for the dataset. The variable importance given by the model is shown in Figure 5.



| | idx | name | score |
|---|---|---|---|
| **0** | 5 | trip_distance | 0.562030 |
| **1** | 6 | duration_mins | 0.204166 |
| **3** | 0 | RatecodeID_ohe_1 | 0.162982 |
| **4** | 1 | RatecodeID_ohe_2 | 0.053931 |
| **6** | 3 | RatecodeID_ohe_3 | 0.011078 |
| **5** | 2 | RatecodeID_ohe_5 | 0.005101 |
| **7** | 4 | RatecodeID_ohe_4 | 0.000668 |
| **2** | 7 | passenger_count | 0.000044 |

FIGURE 5- FEATURE IMPORTANCE OF RANDOM FOREST MODEL

The training time for GLM was much low compared to the Random Forest model, and if the RMSE score is acceptable, GLM model can be used in the place of Random Forest model for saving time.

## Encountered Issues and Solutions

A number of issues were faced during the course of the project. The main challenge was the size of the data with respect to its storage, time taken while querying it in the raw format, and time it takes for transformation and processing. Besides that, there were a lot of considerations taken, both by referring to online sources, and by intuition while cleaning the data. The data cleaning process had to be iterated a number of times to get a good dataset fit for modelling. These points are described in detail below:

1. Due to the enormous size of the data, it was very difficult to do a preliminary exploration of the data. A query would take 5 minutes to take the count of, and 15minutes to compute aggregations. This happened mostly while querying the yellow taxi trip data. So, most of the initial analysis which helped to know about the format of the data to make type conversions was done using only green taxi data and was used for yellow trip data.
2. The data dictionary has PULocationID and DOLoctionID, but while querying the first month's green cab data (2015 January), there were columns *pickup_longitude*, *pickup_latitude*, *dropoff_longitude*, *dropoff_latitude*. Every month was checked for the columns, and it was found that from 2016 June, the longitude and latitude columns were replaced by LocationIDs. This was a major fault found in the data, as it would mean that joining with the location lookup table would be impossible for the entire dataset. So, a decision was made to not use location columns completely.
3. Data cleaning using PySpark's functions was a good idea at the start. But as the conditions grew, the code started getting complicated. Also, from discussion with my team members, I came to know that the time to load a parquet file using the entire datatset took around 6 hours. I thought it might have been due to the numerous spark functions being executed line by line, and it would be good to write a spark SQL query to do the entire data transformation to speed up time. At the end however, ETL on my end also took around 6 hours, but the code was much more readable and easily editable.
4. Initially, data cleaning was done to remove the negative values in time, distance and amount. After ETL however, while running SQLs to answer the business questions on the whole dataset, most of the results looked absurd. These values needed to be removed. For this, discussion with the group was done, and later on online materials were referred, and filtering was done to keep these values within an acceptable range. This process of checking, refining and doing ETL was done at least 5 times.
5. Spark session terminated once with an error in the docker log saying out of memory. Memory shared with docker had to be increased from 6GB to 8GB. Also, docker was using a lot of HDD space due to numerous images. Thus, all the old containers had to be cleared so that space could be freed up for the environment.
6. During modelling random forest, at first, it took more than 2 hours to run. Next time, while running the model, all the kernels which were active were shut down, and the modelling time reduced to 50 minutes. Later, while using GLM, the training time reduced to 10 minutes with similar RMSE score.

## CONCLUSION AND RECOMMENDATIONS

The project included a full data pipeline, with ETL, data analysis and modelling and it was done successfully. A number of business questions were answered, and new facts were discovered of the dataset. The predictive model was also obtained with a low RMSE score.

For future, the filters used on the dataset can be adjusted because a lot of data is rejected in the current implementation, which would have provided more information to the model. An attempt to fix the voided records can also be done. A more drilling down on the variables can be done to find answers to questions such as:

i. Does VendorId has any impact on the final amount?
ii. Which part of the city has observed more trips at which times?

The data can also be joined with external dataset such as weather to find out the weather conditions where more people use taxi. Also, since airports have a good fare amount, flight arrival and departure data may also be used to extend the analyses. The number of trips is gradually decreasing, so it would also be a good idea to check Uber’s dataset to find out ways in which NYC trip counts can be increased.