Digital Object Identifier

Individual Report

MICHAEL GLENNON
Group 4, Data Lead

I. INTRODUCTION

The aim of this project was full-stack development of a web application, with the purpose of providing a service to users with the expectation of utilizing NYC taxi data, along with some other sources of data, to generate a model of busyness. This model would then be used to inform user decisions in terms of route and itinerary planning when visiting Manhattan and surrounding areas. The user interface was to include tools for the user to generate a personal itinerary for travel, and to allow the user a clear set of options to choose from. Options for visiting places of interest, restaurants, cafes and events, among others, were to be presented to the user. The main presentation of busyness information to the user in the form of a busyness heatmap with secondary, more detailed views available on selection of more specific areas in Manhattan and its surroundings. The project is limited in its scope by certain design decisions and data availability. First is the expectation to use data available without a fee, which limits certain details of the modeling of busyness. Second is the set of decisions made to limit the project scope for the purpose of a concise and user-friendly experience in the finished project. All features were discussed in-depth to ensure they fit within, or around, the intended use case. This discussion process also included the restriction of certain development objectives to be completed after core features were already implemented. There were several features that are at a stage where they could still be considered ‘proof-of-concept’, relative to their potential in further development.

My personal aims in completing this project were to further develop my ability to work in a team and to be able to do so while gaining a sense of how to keep project work cohesive while independent portions of the work were to be completed in-between sessions of collaboration.

As I was assigned the role of data lead, I wanted to use this opportunity to gain more experience with working with large datasets and machine learning. I aimed to build on my experience working on the data end of my software engineering project from last semester.

II. GROUP/TEAM STRUCTURE

The team was split into three sub-teams; Front-end, Backend, and Data. The frontend team almost exclusively worked on the user-facing portion of the application, the backend developed an API for requests from the frontend to the backend, set up the database on the server, and dealt with the exchange of information between the Data portion and the frontend.

III. ROLE AND CONTRIBUTIONS

There were four sections of work I had to complete for the project. One was the selection and cleaning of data, second was the generation of data to be put into a machine learning model from the collected data, third was model creation and testing, and finally was the design of a database format to provide access to usable information to the frontend via an API written by the backend team.

A. OVERVIEW AND DESCRIPTION OF BUSYNESS DEFINITION

The source of the data used for the machine learning model creation was the data available on NYC OpenData. This included two datasets:

- 2015 NYC Open Data Yellow Taxi Records
- 2022 MTA Subway Ridership

These datasets were selected because both included features that allowed precise model generation - each taxi pickup/-dropoff had a location in the form of coordinates assigned to it, and an exact time for each. This was also the case with the subway ridership data, where the coordinates of the station were recorded, along with the time (hourly).

The following is an outline of the system for assigning busyness to an area for a given time:

1) There are two sets of zones. Each zone corresponds to measurement of either taxi or subway busyness.
2) A square grid of small zones is used in the measurement or prediction of taxi zones
3) A larger set of zones, based on the original taxi zones provided by the NYC Open Data website, are used in the measurement or prediction of subway busyness.
4) The busyness of a taxi zone is determined by the number of drop-offs that occur within it in a 30-minute period.
5) The busyness of a subway zone is determined by the number of passengers that go through a station within an hour period.
6) The data provided to the front end is given in terms of busyness of the smaller zones, where a small zone’s busyness is generated as a combination of the busyness

level predicted for itself, averaged with the busyness of the larger subway that contains its centroid.

It was determined that the two-zone system added to the accuracy of busyness categorization, as it could account for the different modes of transport and their differing effects on pedestrian activity in their surrounding areas.

B. ZONES

Creating the zones was a primary step in the processing of the data, as further analysis relied on a ’zone’ feature for the taxi trips and subway ridership data.

For creation of the grids and operations using them, the python geopandas and shapely libraries were used. Geopandas can be used to generate objects ’Polygons’, which are what make up the grid squares used for the smaller taxi zones. A polygon object was created at regular intervals, with side lengths of the intervals. Distances were informed by several literary sources [1], [2], [3], [4] which determined that pedestrians generally tend to be willing to travel a maximum distance of 400-450 metres from the location of their public transport drop-off, and around double this distance for high-capacity and high-speed rail such as underground metro systems, in dense metropolitan areas. The grid was positioned within a coordinate window which contains the areas of interest within NYC (Manhattan and the surrounding areas). Zone size and grid location can be easily configured by simple entering of new arguments into the grid creation function.

```python
def create_grid(min_x, min_y, max_x, max_y,
    grid_size):

    bounds = gpd.GeoSeries([Polygon([(min_x, min_y
        ), (max_x, min_y), (max_x, max_y), (min_x,
         max_y)])], crs='EPSG:4326')
    bounds = bounds.to_crs(epsg=32618)
    min_x, min_y, max_x, max_y = bounds.
        total_bounds

    x_coords = np.arange(min_x, max_x, grid_size)
    y_coords = np.arange(min_y, max_y, grid_size)

    grid_cells = []
    for x in x_coords:
        for y in y_coords:
            grid_cells.append(Polygon([
                (x, y),
                (x + grid_size, y),
                (x + grid_size, y + grid_size),
                (x, y + grid_size),
                (x, y)
            ]))

    grid_gdf = gpd.GeoDataFrame({'geometry':
        grid_cells}, crs='EPSG:32618')
    return grid_gdf.to_crs('EPSG:4326')

grid_size = 850
grid_gdf = create_grid(min_x, min_y, max_x, max_y,
     grid_size)
```

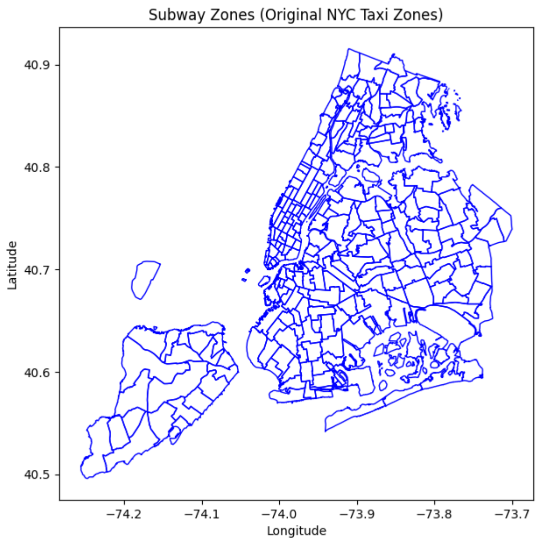


FIGURE 1: These zones are the taxi zones available via the NYC Open Data website. In data processing for this application, they are repurposed as subway zones.

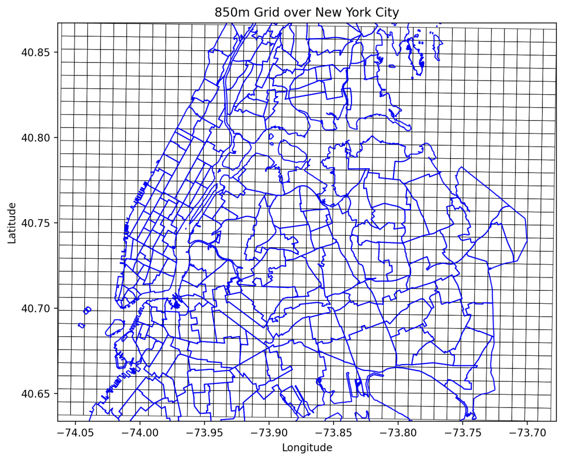


FIGURE 2: The smaller taxi zone grid overlayed on the subway zones.

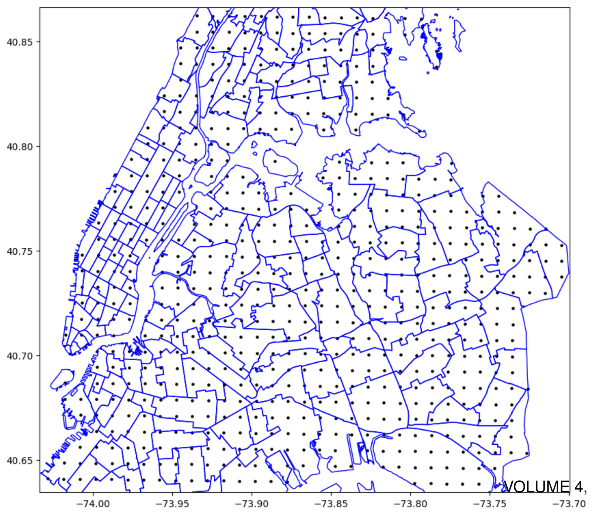


FIGURE 3: Centroids of the taxi zone grid.

C. DATASET ANALYSIS

The taxi and subway datasets were 16M and 146M rows, respectively. This required batch processing of data. Batch processing was completed by conversion of the original data files (bulk download, csv format) into parquet format files, separated by the month of the data - pandas was used to load the csv in chunks and a condition was applied to rows to select the month information form the string corresponding to the date-time column for each row. This data was then cleaned - a small number of rows needed to be removed, never reaching more than 2.5% of the total data.

Note: Attempts to process 2016 taxi data provided unusable rows at a much higher proportion, and so a switch was made to 2015 data.

The most common data quality issues were taxi date-times that were inconsistent (drop-offs recorded as occurring before pick-ups), and trips recorded as having less than 1 passenger. These rows were simply removed from the dataset before any subsequent processing.

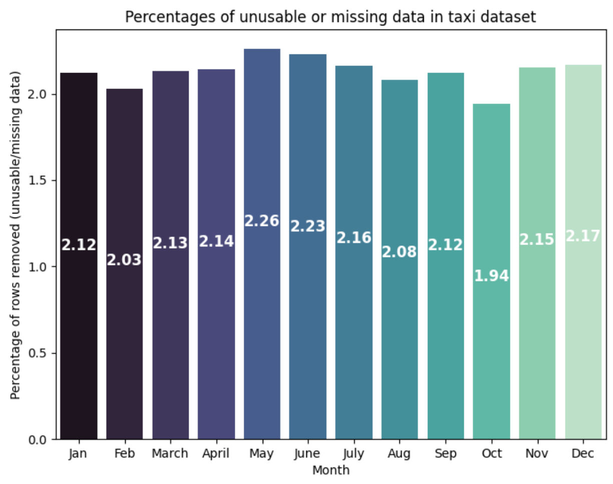


FIGURE 4: Percentage of data removed from the dataset, by month, for the taxi data. The subway data contained even lower proportions of unusable entries.

The features that were available for use in the model generation and predictions were limited by the amount of information that could be provided to the model from the frontend/user, and whatever information could be collected with associated date-time information from APIs. Information such as passenger count, fare, tips, trip length, etc. were not usable for the prediction of the target feature, which was the number of the drop-offs in an area, as these could not be provided to the model.

Some trends in the data were noted, but the transformation of the data into a form that could represent trip counts within a set time interval was required to generate a dataset that was relevant to the application's planned features.

D. PREPARATION OF DATA

The two tasks of processing the cleaned dataset were:

1. Group taxi trips/subway ridership rows by a set time interval. 2. Allocate the zone of a taxi trip/record of subway ridership by their coordinates in the dataset.

The creation of the 'zone' feature needed to be completed first. This was the most time-consuming of the data tasks, and required several hours of processing to complete. In order to facilitate this task in a reasonable amount of time, multi-processing/parallel row operations were used via the python library pandarallel. This allowed multi-core processing of the rows.

The geopandas dataframe of the zones for the taxi data first had to be labelled with the operations:

```
small_zones['centroid'] = small_zones['geometry'].centroid
small_zones['centroid_x'] = small_zones['centroid'].x
small_zones['centroid_y'] = small_zones['centroid'].y
small_zones = small_zones.sort_values(by=['centroid_x',
'centroid_y'], ascending=[True, True])
small_zones['zone_label'] = range(1, len(small_zones) + 1)
small_zones = small_zones.drop(columns=['centroid'])
```

The function applied to the dataset:

```
def zone_finder(lon, lat, zones):
    point = Point(lon, lat)
    point_gdf = gpd.GeoDataFrame([{'geometry':
        point}], crs=small_zones.crs)
    result = gpd.sjoin(point_gdf, zones, how='left
        ', predicate='within')
    if not result.empty:
        return result.iloc[0]['zone_label']
    else:
        return None
```

This function used the geopandas and shapely libraries. A point was created from the longitude and latitude values in a row, added to a new geopandas dataframe, and a spatial join operation was performed with the 'within' predicate. This finds which of the zones in the 'zones' geojson that each of these points is contained by, and returns the 'zone_label' column value from the resulting dataframe.

p_count	dropoff_longitude	dropoff_latitude	dropoff_datetime	dropoff_zone
1	-73.991791	40.746620	08/21/2015 11:30:20 PM	195
2	-73.981514	40.762455	08/01/2015 11:08:12 AM	227
1	-73.804558	40.649757	08/17/2015 03:34:57 PM	752
1	-73.979805	40.7617237	08/08/2015 03:01:14 AM	257
1	-73.786385	40.644257	08/30/2015 08:34:47 AM	812

TABLE 1: Dataframe after removal of unnecessary columns, and addition of 'dropoff_zone' column *('passenger_count' column label shortened for display)*

dropoff_zone	month	day	time_interval_float	passenger_count
6	8	0	14.5	2
6	8	0	17.0	2
8	8	2	21.0	1
8	8	4	9.5	1
9	8	1	14.0	5

TABLE 2: Dataframe after taxi trips were grouped into date-time intervals, with datetime values separated into days, months, and a float value representing hours

E. MODEL TESTING

A number of models were reviewed for their usefulnes
this application. The two models chosen for compariso
tests run with the taxi and subway data were the rand
forest regressor and xgboost regressor models.

Across several tests of the performance of the models,
trends in time taken to produce predictions and the accur
of the predictions were the key targets for analysis. It
assumed that prediction time was a more significant conc
for this application, as it limited the resolution and accur
of the predicted dataset for the frontend. Predictions wer
be made for 660 zones, and so small differences in test ti
would imply noticeable differences for the deployed mod
performance.

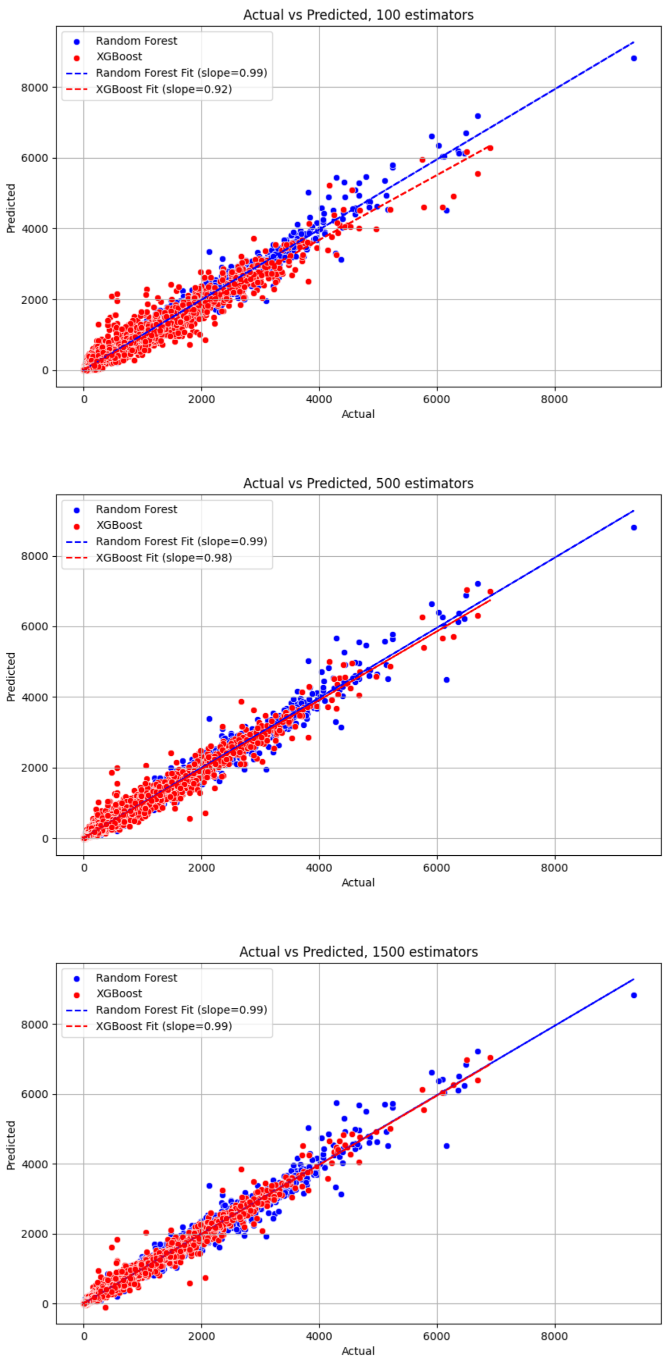


FIGURE 5: Tests of the random forest and xgboost models, with 100, 500 and 1500 estimators each. By 500 estimators, the accuracy of xgboost was similar to the random forest model.

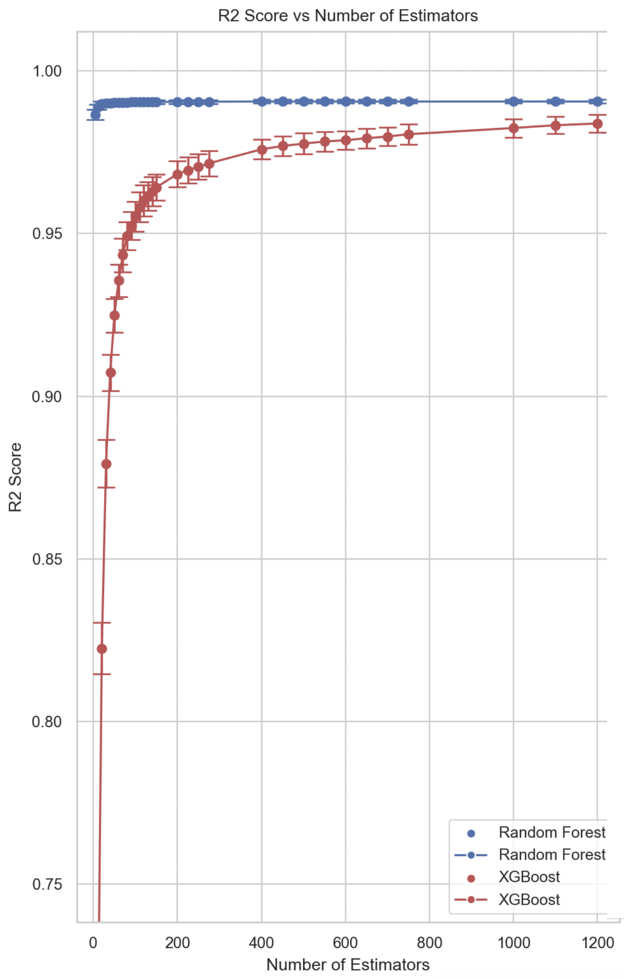


FIGURE 6: R^2 score vs Number of estimators for the random forest and xgboost models.

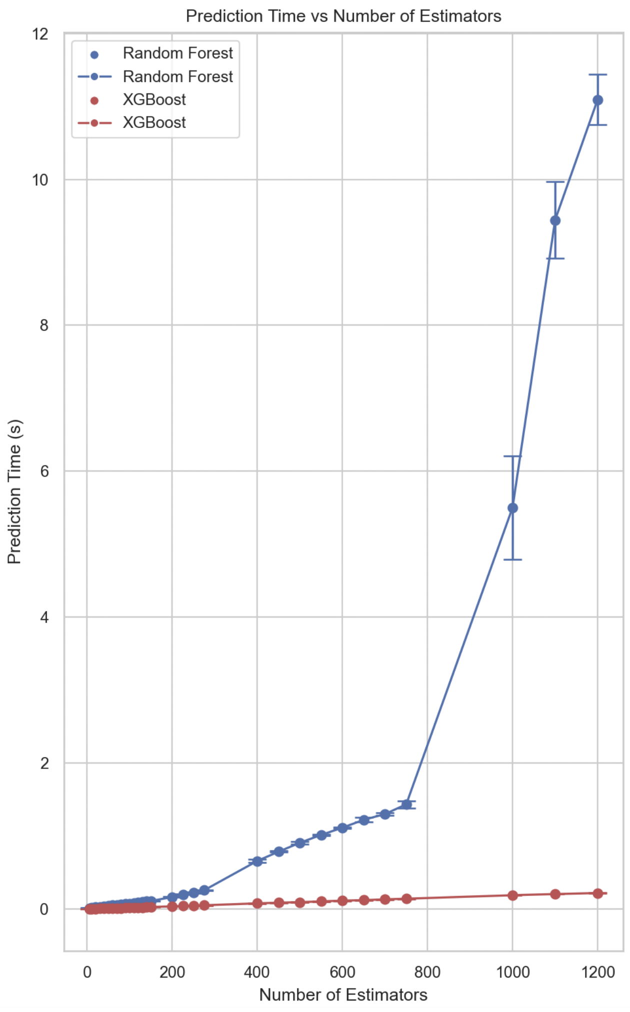


FIGURE 7: Prediction time vs Number of estimators both models. An approximately exponential relationship is seen with the random forest model, while a much shallower and linear relationship is visible with the xgboost model.

It was determined that the prediction time advantage of the xgboost model was noticeable enough for it to be chosen over the random forest, as it demonstrated similar levels of accuracy with much less wait time.

The use of the xgboost model necessitated that the data be converted via one-hot encoding to allow it to function properly with categorical data (weekdays, months). An encoder is generated along with the creation of the model, so new data can be encoded as it is provided to the model for predictions.

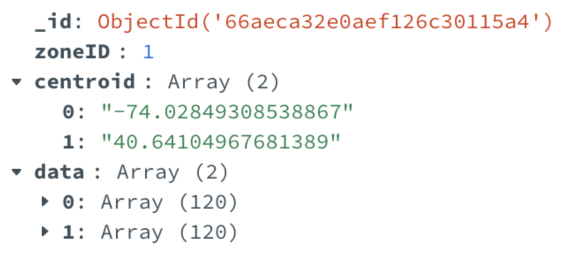
F. DATABASE DESIGN

The design of the database was an important topic of collaboration with other team members. The frontend team in particular provided feedback and suggestions based on what features they wanted to implement. The database had to provide information that made the display of data easy on the frontend and the features needed for the application as simple as possible to create.

The structure of the data was changed several times during development. In its initial format for an MVP, database entries (hash table) contained a set of coordinates as one tuple, and then a tuple containing two lists of values; one was a list of times in half-hour intervals (hour fraction floats, e.g. 3.5 for 3 and a half hours) and the second was the predicted busyness levels for those times. This general format of having two lists for the time-busyness relationship was maintained. The initial format contained no zone labels and only spanned a constant 72 hours into the future.

Due to feedback from the frontend team, it was decided that the predictive data should cover the previous 24 hours and provide predictions into the future for the number of hours required to reach midnight in 3 days' time, from the time of the prediction.

This data structure allowed the frontend to only easily use the coordinates of the centroids of the zones. Therefore, for a tile-based heatmap feature in the frontend, modifications were made to the way the database's entries were labelled. In the deployment version of the model and database, the predictions are labelled by zone. The frontend uses a zone-lookup file (geoJSON) to find the coordinates of the zone vertices for the UI graphics.



```
  _id: ObjectId('66aeca32e0aef126c30115a4')
  zoneID : 1
▾ centroid : Array (2)
    0: "-74.02849308538867"
    1: "40.64104967681389"
▾ data : Array (2)
  ▸ 0: Array (120)
  ▸ 1: Array (120)
```

FIGURE 8: Deployment database structure for a single zone. 'ZoneID' is the number associated with the grid square, and the number which the frontend uses to find the locations and vertices of the grid squares when generating them on the UI, graphically. '_id' is an identifier applied by MongoDB. MongoDB modifies certain datastructure types upon upload, seen here where the tuples are labelled as 'arrays'.

G. POTENTIAL IMPROVEMENTS AND COMMENTS

1) Database

One improvement to the database update system that would increase performance would be to only update the hour interval predictions that are required rather than running the model for all times each time the database is updated. This was overlooked in favor of testing the updating functions (some testing required re-forming the database).

2) Dataset and zones

The zone size that is used for the current deployment is still quite large. It is possible to change the zone size, but in doing so, the availability of data per zone needs to be considered carefully. With the inclusion of multiple years of data, it would be possible to get a higher resolution for the frontend's heatmap, but the processing time would be substantial to set this up. There is also the issue of adding wait time to the prediction and update process, which is directly related to the amount of grid squares that are being used.

3) Model parameters

Additional parameters for input into the model were planned but not implemented, such as the inclusion of weather prediction data from OpenWeather API, where the models would be trained on past data from NYC Open Data weather datasets. This would require matching taxi trip intervals to the weather data, training the models, and querying the OpenWeather API for a forecast for the next 3 days (current deployment prediction scope).

H. ADDITIONAL INFORMATION

TABLE 3: NYC OpenData Yellow Taxi dataset example entry

Feature	Data
passenger_count	*2*
trip_distance	*1*.69
pickup_longitude	-73.972847
pickup_latitude	*4*0.793552
RateCodeID	*1*.0
store_and_fwd_flag	*N*
dropoff_longitude	-73.988297
dropoff_latitude	*4*0.774899
fare_amount	*8.5*
extra	*0.5*
mta_tax	*0.5*
tip_amount	*0.0*
tolls_amount	*0.0*
total_amount	*9.8*
vendor_id	*2*
pickup_datetime	*0*8/31/2015 11:52:07 PM
dropoff_datetime	*0*9/01/2015 12:00:23 AM
rate_code	*N*aN
payment_type	*2*
imp_surcharge	*0.3*

TABLE 4: NYC OpenData MTA Subway Ridership dataset example entry

Feature	Data
transit_timestamp	2022-06-24 04:00:00
station_complex_id	*H*007
station_complex	*1* Av (L)
borough	*M*
routes	*L*
payment_method	*o*mny
ridership	*19*
transfers	*0*
latitude	*4*0.730953
longitude	*7*3.981628
Georeference	*P*OINT (-73.98162841796875 40.730953216552734)
itsuid	2022-06-24T04:00:00H0071 Av (L)MLomny

IV. LESSONS LEARNED

Technical skills: My role as data lead allowed me to gain further experience in dealing with large datasets and some feature generation for machine learning models. I have a general awareness of the tools available via python libraries specifically designed to deal with these larger datasets and process them. I gained some experience in using and modifying non-SQL databases (MongoDB) and an understanding of the python tools used to do so.

I plan to use this more in-depth experience as a starting point to learn more about the libraries and methods available for both data processing and, importantly, machine learning/deep learning. I am interested in developing this part of my own skill set to match the fast-developing field of AI.

Professional skills: The level of collaboration between team members was more substantial than the other team projects I have been a part of so far. There was a consistent checking-in between teams to ensure compatibility, or at least to ensure changes were independent.

The single most important aspect of working in this sort of team structure is allowing the team's aims and collective approach to take precedent over your own. Progress made independently must be verified consistently, and notification of issues must be made in good time.

In practice, and due to how I chose to design the database and prediction system with the other teams, my work was more independent than the work of the other members. Most changes made to data processing such as those made to increase efficiency, or utilise different libraries had no effect on the backend and frontend teams. The separation of the model prediction from the database query API made it so that changes related to the predictions didn't need to affect anything else in the application development. I believe task delegation in a team of this size and for a project of this scope is difficult.

REFERENCES

[1] R. Daniels and C. Mulley, "Explaining walking distance to public transport: The dominance of public transport supply," Journal of Transport and Land Use, vol. 6, p. 10, 2011.

[2] M. Burke and A. Brown, "Distances people walk for transport," Road & Transport Research, vol. 16, no. 3, pp. 16–29, 2007.

[3] A. Tennøy, M. Knapskog, and F. Wolday, "Walking distances to public transport in smaller and larger norwegian cities," Transportation Research Part D: Transport and Environment, vol. 103, p. 103169, 2022.

[4] D. van Soest, M. R. Tight, and C. D. F. Rogers, "Exploring the distances people walk to access public transport," Transport Reviews, 2019. Advance online publication.