NYC Taxi Duration Prediction

December 14, 2023

Authors: Jeff Hansen, Dylan Skinner, Jason Vasquez, Dallin Stewart

1 Introduction

We aim to tackle the challenge of predicting taxi ride durations in New York City based on starting and stopping coordinates. A 'taxi ride duration' refers to how long, in seconds, a taxi ride takes. This research question is relevant for urban commuters and transportation systems, and it incorporates the myriad of factors influencing taxi ride times. These factors include traffic patterns, congestion, time of day, and external variables such as local events or weather conditions. The strengths of machine learning methods align well with the intricacies of this problem, and allow us to uncover more nuanced relationships within the data. Beyond the potential convenience for individual riders, the ability to predict taxi ride times carries practical implications for optimizing taxi fleet management, city resource allocation, and enhancing overall traffic flow in New York.

In preparation for our analysis, we conducted initial research about travel time prediction. We quickly learned about the importance of removing outlying data and unnecessary attributes [RR22] (see section `8. Data Cleaning`). Additionally, we learned which models typically perform best with travel time prediction [BLM18] and concluded that tree based ensembles are typically best for this type of task [HPMP20].

Kaggle published the dataset we are exploring for a coding competition, so over one thousand other groups have investigated this research question. Successful teams performed feature engineering to create fields such as month, day, hour, day of the week, and used models such as Random Forest Regression, Extra Trees Regression, PCA, XGBoost, linear regression, and Light GBM [RIS17].

The original dataset came fully complete, does not contain missing data, and includes fields such as: taxi driver, the number of passengers, and whether the trip time was recorded in real time. We added a second dataset that contains weather information with timestamps, temperature, precipitation, cloud cover, and wind information at every hour [AAD22]. The NYC taxi cab dataset was published by NYC Taxi and Limousine Commission (TLC) in Big Query on Google Cloud Platform and is well documented and densely populated with over one million data points [TLC23]. The weather dataset is much more sparse and has a much larger time range than needed to match the NYC Taxi data. These datasets provide a good source for addressing our research questions because they extensively cover NYC taxi travel for a significant time period. In short, the data allows us to effectively identify significant features impacting taxi trip duration and develop a robust predictive model for accurate estimations.

While we focus our primary investigation on determining taxi cab trip duration, we are also interested in several other questions. What days of the week and times of day are most busy? Where are the most popular destinations? Furthermore, the multifaceted exploration of temporal, spatial,

and environmental influences on travel durations in NYC presents a well-rounded analysis to reveal comprehensive insights into travel behaviors. This poses questions such as how do environmental factors such as rain impact taxi popularity? Our data is well suited for our research question in exploring and predicting taxi trip duration, and has a single, clear answer. A process of machine learning model development will help us go one step further and develop a robust predictive model to estimate and understand trip durations accurately. Our approach, seen below, is comprehensive and unique in several ways, for example we use KMeans clustering to group the pickup and dropoff locations together, and use a grid search to find optimal hyperparameters of each model.

2 Feature Engineering

In pursuit of a model with higher predictive power, we included several additional features in our dataset that fall in one of three distinct feature groups: datetime, distance, and weather.

2.1 Data Cleaning

The taxi cab duration dataset from Kaggle contained several outliers that required removal. For example, some trips lasted 1 second, and others lasted over 980 days. To prevent erroneous data, we removed all rows where `trip_duration` was in the .005 quantile, or less than 60 seconds. The dataset also contained outliers in the pickup and dropoff locations that fell far outside New York City. We fixed these rows by removing any point outside of city limits. For the full plot visualization code, see section `8. Data Cleaning` in the Appendix.

2.2 Datetime

One of the most important features in our dataset is passenger pickup time, originally represented as a string in the format `YYYY-MM-DD HH:MM:SS`. We created multiple time features from this column including `pickup_month`, one-hot encoded `pickup_day` columns, `pickup_hour`, and `pickup_minute`. We also added other versions of these data points including one-hot encoded `pickup_period`, `pickup_hour_sin`, `pickup_hour_cos`, and `pickup_datetime_norm`. See section `8. Data Cleaning.clean_data` in the Appendix for more details.

2.2.1 Pickup Period

The feature `pickup_period` captures the time of day when passengers were picked up in one of four periods: morning (6:00 AM to 12:00 PM), afternoon (12:00 PM to 6:00 PM), evening (6:00 PM to 12:00 AM), and night (12:00 AM to 6:00 AM). These divisions align intuitively with significant periods of the day for taxi services, such as morning rush hours and evening nightlife. See section `8. Data Cleaning.clean_data` in the Appendix for more details/

2.2.2 Pickup Period Sine/Cosine

We applied a circular encoding to time to account for the cyclical nature of the hours of the day. We created `pickup_hour_sin` and `pickup_hour_cos` features using sine and cosine transformations to avoid discontinuities such as the start and end of a day. See section `8. Data Cleaning.clean_data` in the Appendix for more details.

$$\text{pickup\_hour\_sin} = \sin\left(\frac{2\pi \cdot \text{pickup\_hour}}{24}\right) \quad \text{pickup\_hour\_cos} = \cos\left(\frac{2\pi \cdot \text{pickup\_hour}}{24}\right). \quad (1)$$

2.2.3 Pickup Datetime Norm

The final feature we created in the datetime feature grouping was `pickup_datetime_norm` to represent the normalized pickup datetime. This feature converts the pickup datetime from nanoseconds to seconds, then scales the value by the maximum to place all the values between 0 and 1. See section `8. Data Cleaning.clean_data` in the Appendix for more details.

2.3 Distance

We created two features that estimate distance between pickup and dropoff locations: the Manhattan distance and the average distances between local coordinate clusters.

2.3.1 Manhattan Distance

We include the Manhattan Distance feature because of its grid based metric. Many of the streets of New York are laid out in a grid-like fashion, so this metric can better approximate road distances than the Euclidean distance. The Manhattan distance is also more simple and interpretable, since it computes the sum of the absolute values of the differences between the x and y coordinates of the two points. We calculated the Manhattan distance by first converting the pickup and dropoff coordinate points into radians and using the following formula:

$$\text{Manhattan Distance} = R \cdot (|\text{pickup\_latitude} - \text{dropoff\_latitude}| + |\text{pickup\_longitude} - \text{dropoff\_longitude}|) \quad (2)$$

where $R = 6371$ km, the radius of the Earth in kilometers. See section `9. Feature Engineering.distance` in the Appendix for the full implementation.

2.3.2 KMeans Clustering Average Duration

Along with incorporating distance metrics and weather information, we also added a duration feature that acts as an initial estimate for the model's actual trip duration prediction. To do this, we fit a pickup and dropoff KMeans clustering model with 200 clusters as shown in section `4. Data Visualization`. We then labeled each pickup location and dropoff location in the data with its respective cluster label. By grouping the data by these cluster pairs, we computed the average trip_duration between each cluster pair and merged this onto the original dataframe. See section `6. KMeans Clustering` in the Appendix for the full code implementation. This feature helps by giving the model a baseline for what duration to expect based on location. It also allows us to segment the data in a way that can group outliers, or large distance locations, together and improve accuracy by grouping like locations together. See section `9. Feature Engineering.add_avg_cluster_duration` in the Appendix for more details.

2.4 Weather

When considering potential features to add to our dataset, accounting for the effect of local weather on taxi ride time was one of the most obvious additions to include. For example, if it is raining

or snowing, we may assume there will be more traffic on the roads, and more people who would normally walk would prefer a taxi. A dataset created by Kaggle user [AAD22] contains a myriad of weather data for New York City between 2016 and 2022 on an hourly basis. This dataset includes features such as temperature (in Celcius), precipition (in mm), cloud cover (low, mid, high, and total), wind speed (in km/h), and wind direction. We decided to use temperature, precipitation, and total cloud cover as features in our dataset with a simple join on the pickup datetime rounded to the nearest whole hour. See sections `8. Data Cleaning.get_clean_weather` and `9. Feature Engineering.add_weather_feature` in the Appendix for more information.

3 Feature Selection

As seen in section `2. Feature Enginnering`, we created several new features in addition to those already in the dataset. We also consider which features are unnecessary or unhelpful.

3.1 L^1 Regularization

To determine the most important features, we utilized L^1 regularization since it functions by setting unneeded feature coefficents to 0 and typically out-performs step-wise feature removal. We did not use L^2 regularization because the large number of features allows L^1 regularization to perform better.

[22]:
```
# lasso_feature_selection performs feature selection using the LassoLarsIC
↪method
lasso_feature_selection(X, y)
```

```
Optimal Alpha     : 1.0806
Optimal BIC       : 18056884.5229
Lasso Coefficients: [ 0.    , 0.     , 0.    , 0.    , 0.    ,
                      0.    , 0.     , 0.    , 0.    , 0.    ,
                      0.    , 0.     , 0.    , 0.0169, 0.    ,
                      0.    , 0.     , 0.    , 0.    , 0.    ,
                     -0.0345, -0.1089, 0.    , 0.0147,  0.0043]
Important Features: ['pickup_minute', 'distance_km',
                     'temperature_2m (°C)', 'cloudcover (%)',
                     'avg_cluster_duration']
```

4 Data Exploration and Visualization

4.1 Data Exploration

Before visualizing the data, we will first perform basic data exploration to show the effects of adding the features mentioned above.

We built a function `get_X_y()` that performs basic feature engineering and returns two Pandas dataframes: X and y. We also built the function `generate_features()` that performs more advanced feature engineering and returns one Pandas dataframe: X. See section `8. Data Cleaning` in the Appendix for the full function. The table below shows a single example of the data and features from the final `feature_X` dataframe.

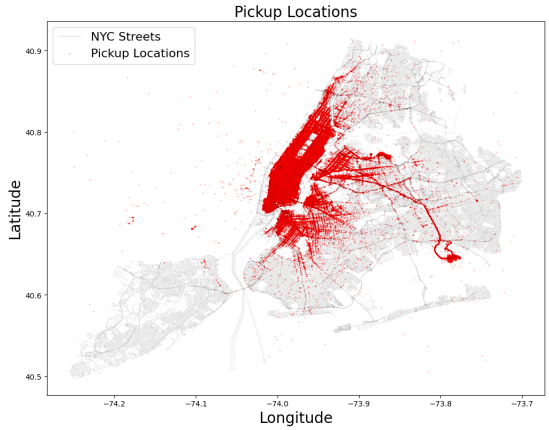
[3]:
```
X, y = get_X_y(force_clean=True)    # All feature engineering is done in the␣
 ↪function get_X_y, found in appendix
feature_X = generate_features(X, y) # generates features adds more features to␣
 ↪the data, including weather
(rows, cols), y_shape = feature_X.shape, y.shape
print("Data Rows:", rows)
print("Feature Cols:", cols)
```

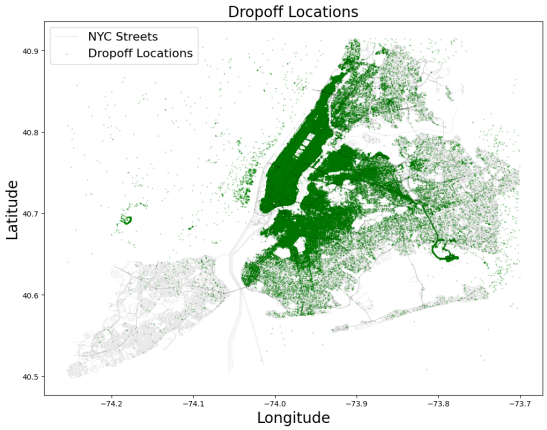
```
Data Rows: 1441615
Feature Cols: 27
```

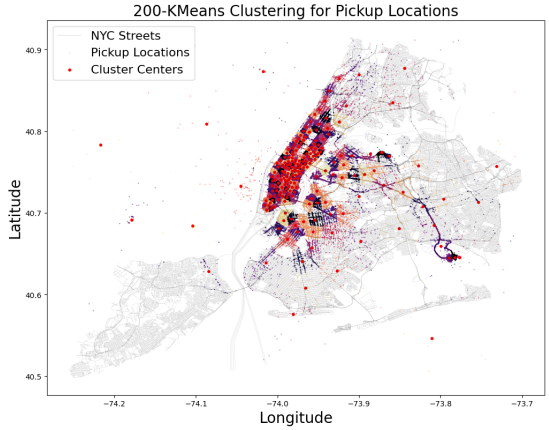
Feature	Value	Feature	Value	Feature	Value
vendor id	1	pickup datetime	2016-03-14 17:24:55	passenger count	1
pickup longitude	-73.98	pickup latitude	40.77	dropoff longitude	-73.96
dropoff latitude	40.77	pickup month	3	pickup day Monday	1
pickup day Saturday	0	pickup day Sunday	0	pickup day Thursday	0
pickup day Tuesday	0	pickup day Wednesday	0	pickup hour	17
pickup minute	24	pickup period morning	0	pickup period afternoon	1
pickup period evening	0	pickup hour sin	-0.97	pickup hour cos	-0.26
pickup datetime norm	0.41	distance km	2.21	temperature 2m (°C)	6.40
precipitation (mm)	0.20	cloudcover (%)	100.00	avg cluster duration	814.73

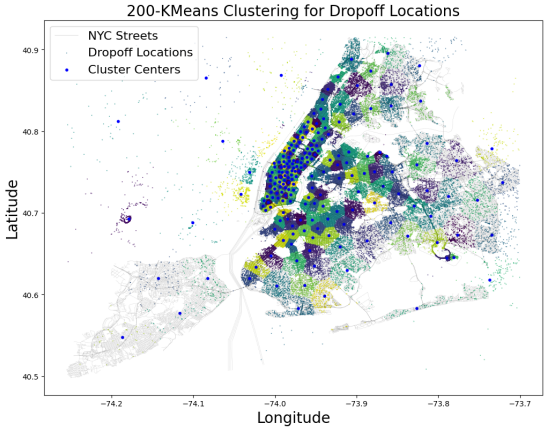
4.2 Data Visualization

In the following figure, the top two graphs visualize the pickup and dropoff locations overlaid over a map of NYC. The bottom two graphs display the pickup and dropoff locations clustered into groups using KMeans clustering. These charts reveal that pickup locations are more heavily clustered around downtown (Manhattan), while the dropoff locations are more evenly distributed throughout the city. This distribution indicates that Manhattan, where more people go to work, is the most popular place to hail a taxi. The most popular destination is still in Manhattan, but the destinations are much more distributed across New York. While it is difficult to definitively conclude the cause for this difference, one possible explanation is that people get rides home more frequently. For the full plot implementation, see section `7. Visualization` in the Appendix.

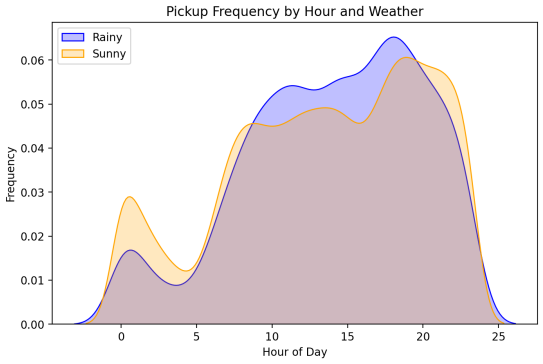


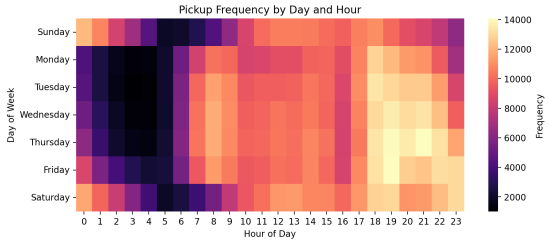






We can also learn about the factors that influence a New Yorker's decision to take a taxi from the data. For example, in the figure below, the graph on the left displays the most popular times of day to hail a taxi, which peeks around 6:00 PM and drops the lowest around 4:00 PM. We also see that poor weather encourages more people to take a taxi during the day when people are more likely to be returning from their daily activities, but less likely to choose to go out at night in the first place. The graph on the right shows the most popular days of the week for taxis, which peeks on Friday and Saturday and during the evenings of the weekdays. In the graph on the right, brighter colors indicate more taxi rides, and darker colors indicate fewer taxi rides. For the full plot implementation, see section `7. Visualization` in the Appendix.





5 Modeling and Results

Our primary research question is to predict the duration of a taxi ride in New York City. To achieve this goal, we implemented a variety of machine learning models including Lasso regression, Random Forest regression, XGBoost, and LightGBM. We explain our implementation, training, optimization, and results for each model below.

5.1 Model Selection

After performing feature engineering, we performed hyperparameter grid searches for the each of these models. The table below shows the training time for each model in the first row, and the optimal hyperparameter choices and the best Root Mean Squared Error (RMSE) for each model:

Parameter	Lasso Params	LightGBM Params	LightGBM_large Params	RF Params	XGBoost Params
Training Type	<1 min	1.5 hr	–	6 hr	9 hr
boosting_type	–	gbdt	gbdt	–	gbtree
learning_rate	–	0.01	0.01	–	0.01
max_depth	–	20	50	3	10
n_estimators	–	100	100000	200	100
num_leaves	–	30	500	–	–
reg_alpha	1.0806	0.1	0.1	–	0.1
reg_lambda	–	0.5	0.5	–	–
max_features	–	–	–	log2	–
min_samples_leaf	–	–	–	2	–
Best RMSE	606.9680	606.9699	622.4705	605.1684	603.7398

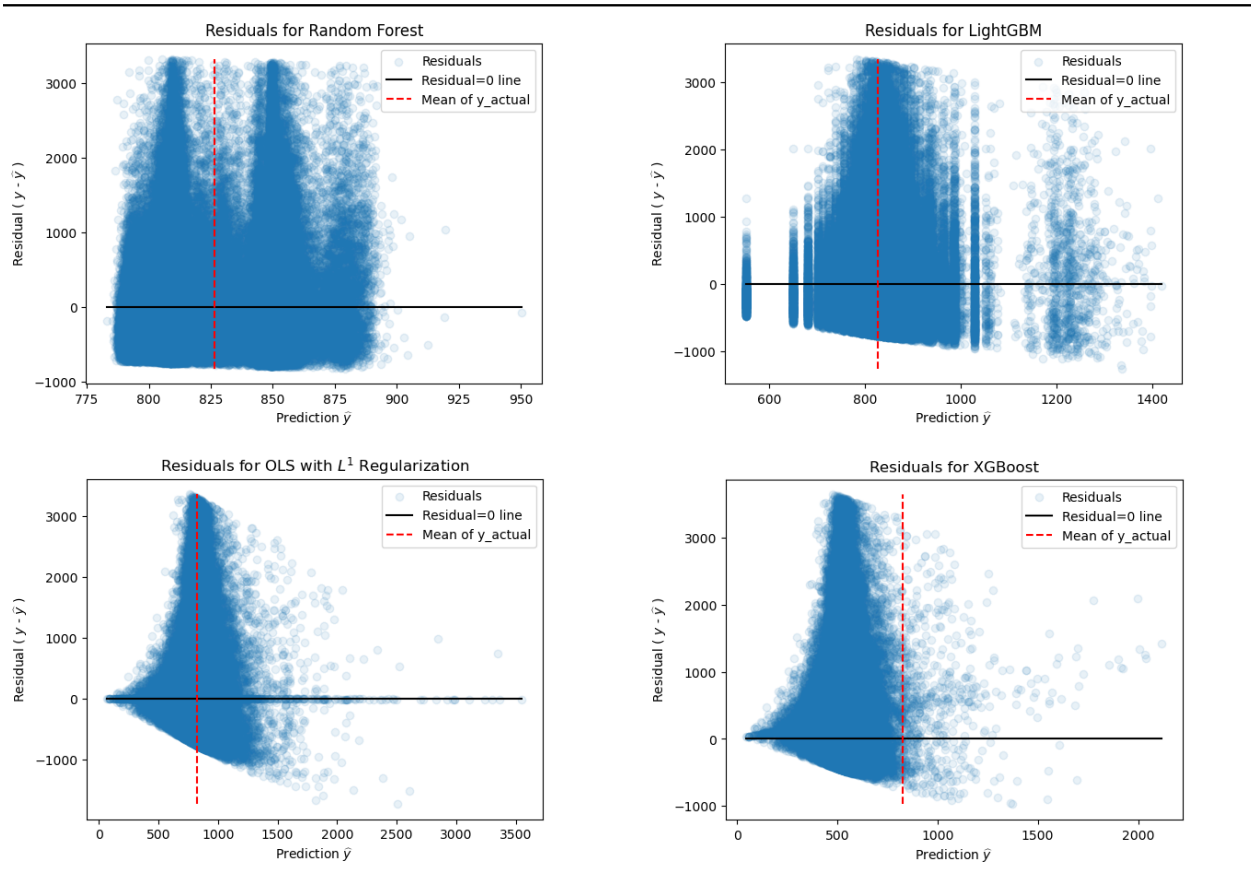
See sections `1. Lasso Feature Selection`, `2. LightGBM Hyperparameter Selection`, `XGBoost Hyperparameter Selection`, `4. XGBoost Model`, `5. Random Foreset Hyperparameter Selection` in the Appendix for the code implementations.

6 Interpretation

The residual plots below demonstrate how far away our models' predictions deviate from the actual value, and they provide insights into how each model makes predictions See section `7. Visualizations.plot_residual` in the Appendix for the plotting code. As seen across all of the residual plots, all of the models generally tend to predict the mean of the actual data. This behavior indicates our model is underfit.

The Random Forest residual plot reveals a multimodal-like distribution with three main hills. This layout implies that the Random Forest model identified several clusters with different means from the taxi data. Additionally, the LightGBM residuals appear bi-modal, suggesting that the model found an additional clustering. The bin-like prediction values for the LightGBM model is likely a result of LightGBM's use of histogram binning to speed up the training process.

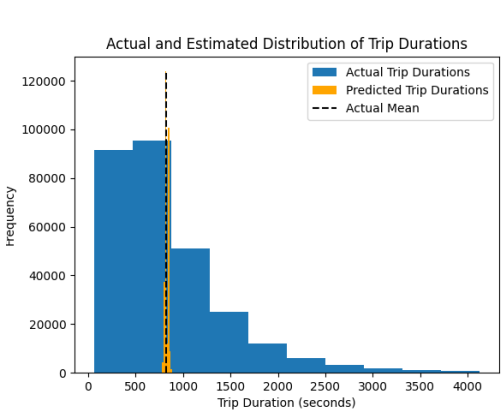
The Lasso model's slightly right-skewed normal distribution likely stems from its underfitting linear model. Despite its limitation, the Lasso model's residual plot aligns remarkably well with the actual trip duration distribution depicted in the histograms below. Furthermore, the XGBoost model appears very similar to the Lasso model, but with more skew to the right. This difference likely arrises because the dataset has many more shorter trips than longer ones, and the model is more likely to predict the mode over the mean.

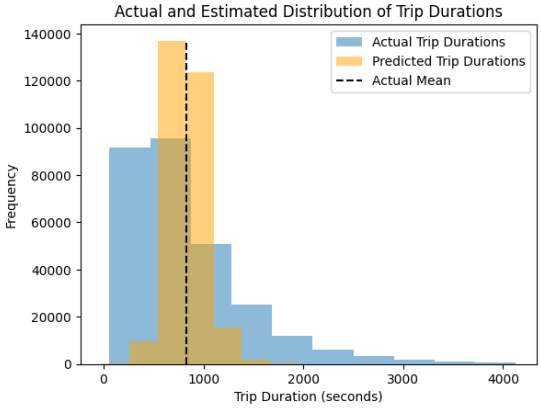


The figures below depict the actual distribution of taxi trip durations along with our model's predicted distributions See section `7.`

`Visualizations.plot_actual_vs_predicted_distributions` in the Appendix for the plotting code. The plot on the left depicts the distribution of outputs from a LightGBM model with optimal parameters (100 `estimators`, 30 `leaves`, and a `max_depth` of 50). This model is not very strong, as it predicts the mean of the actual distribution almost exclusively. This poor performance implies that the feature space is not large enough for the model to create a more accurate approximation of the distribution.

We generated the plot on the right with a different LightGBM model using much larger parameters (100,000 `estimators`, 500 `leaves` and a `max_depth` of 50). This model took a full hour to train and around 15 minutes to make predictions. As the prediction distribution demonstrates, this model makes predictions farther from the mean, indicating that the model fits the data better as we raise its complexity and compute capabilities. With enough compute resources and time, we could increase the model parameters to higher values and create more features for the data. This upgrade would allow the model to better approximate the data instead of simply predicting the mean.





The map in **Figure 1** at the end of this paper plots cluster pairs which have the highest travel time to distance traveled ratio on average. See section `7. Visualizations.duration_per_distance` in the Appendix for the code. Interestingly, some of the cluster paths cross major roadways, implying that there should be more support roads that cross under or over the major highways. Furthermore, a majority of the largest duration trips per distance occur in Manhattan. This demonstrates that the public transit system could be improved to optimize taxi traffic use.

7 Ethical Implications

Our research involves analyzing a large dataset created by tracking some of the life-style patterns of real people living in New York during 2016, raising concerns about privacy and responsible data usage for individual behaviors, locations, and travel patterns. Our dataset and model protect this data by excluding all personally identifiable information to ensure that only aggregate information can be meaningful, and the patterns of individuals remain indiscernible.

The risk of misinterpretation or misuse of our predictive models is very real—especially considering its weak predictive power. Users could misunderstand predictions, leading to inappropriate decision-making. Our predictive model may contain and inadvertently perpetuate biases that we

are unaware of, such as inappropriate associations with certain neighborhoods and taxis. Furthermore, users might misunderstand the predictive and uncertain nature of the model and treat its estimates as certainties. For instance, if taxi companies or transportation authorities were to make decisions solely based on the model's predictions without considering broader traffic management strategies, it could inadvertently lead to concentrated traffic, worsening congestion in certain areas. If our model were deployed in conjunction with algorithms influencing taxi availability, the system might inadvertently create self-fulfilling feedback loops, disproportionately affecting certain areas or demographics that cannot afford taxi's and therefore will not be seen in the data.

To address these issues, clear communication about the model's limitations, potential biases, and intended use is crucial. Providing educational resources, user-friendly interfaces, adequate documentation, and implementing fairness-aware algorithms can contribute to responsible and ethical deployment. Regular assessments, periodic audits, and interventions are necessary to avoid reinforcing existing biases. We have also considered ethical responsibilities such as the responsible disclosure of findings, ensuring the public benefits from the research, and avoiding any unintentional harm. Active engagement with potential stakeholders and the community can further help address concerns and foster ethical practices.

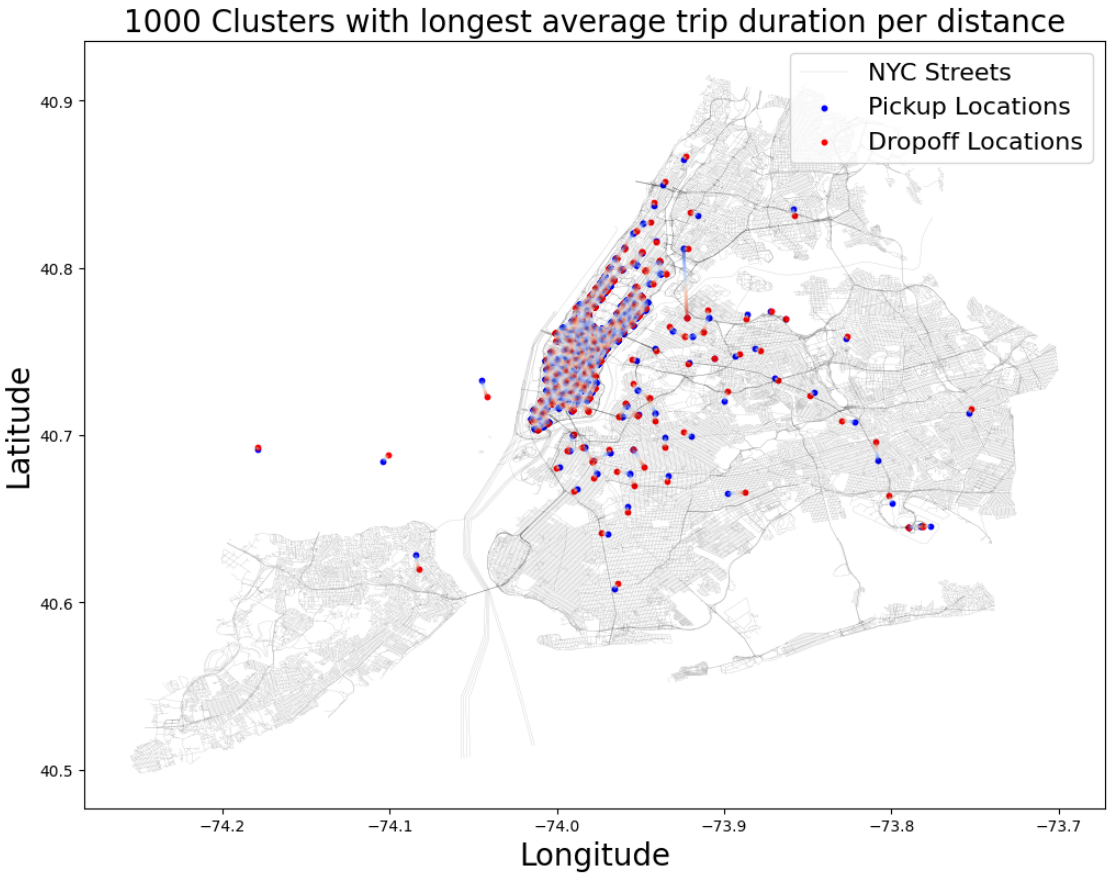


Figure 1: Cluster pairs with highest travel time to distance

8 References

[RR22] - Roy, B., Rout, D. (2022). Predicting Taxi Travel Time Using Machine Learning Techniques Considering Weekend and Holidays. In: Abraham, A., et al. Proceedings of the 13th International Conference on Soft Computing and Pattern Recognition (SoCPaR 2021). SoCPaR 2021. Lecture Notes in Networks and Systems, vol 417. Springer, Cham. https://doi.org/10.1007/978-3-030-96302-6_24

[HPMP20] - Huang, H., Pouls, M., Meyer, A., Pauly, M. (2020). Travel Time Prediction Using Tree-Based Ensembles. In: Lalla-Ruiz, E., Mes, M., Voß, S. (eds) Computational Logistics. ICCL 2020. Lecture Notes in Computer Science(), vol 12433. Springer, Cham. https://doi.org/10.1007/978-3-030-59747-4_27

[BLM18] - Bai, M., Lin, Y., Ma, M., Wang, P. (2018). Travel-Time Prediction Methods: A Review. In: Qiu, M. (eds) Smart Computing and Communication. SmartCom 2018. Lecture Notes in Computer Science(), vol 11344. Springer, Cham. https://doi.org/10.1007/978-3-030-05755-8_7

[RIS17] - Meg Risdal. (2017). New York City Taxi Trip Duration. Kaggle. https://kaggle.com/competitions/nyc-taxi-trip-duration

[TLC23] - TLC Trip Record Data. (2023). NYC Taxi and Limousine Commission. https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page

[AAD22] - Aadam. (2022). NYC Weather - 2016 to 2022. https://www.kaggle.com/datasets/aadimator/nyc-weather-2016-to-2022