NYC Green Taxis with Neural Network

Brittany Nicholls
Department of Computer Science
and Electrical Engineering
University of Maryland, Baltimore County
Baltimore, Maryland 21250
Email: brn1@umbc.edu

***Abstract*—NYC Green TAxi. Use Neural Net. Kaggle Competition. RMSE vs RMSLE**

I. Hard Lesson Learned

Dr. Hamed,

I am going to begin this paper by stating a lesson learned the hard way, and unfortunately, learned too late to try to fully rectify the situation (though I'm trying). This project was about trying to predict the travel time for NYC Green Taxis at the beginning of the trip since they have different routes than Yellow Taxis that, I believe, would make them more difficult to model, but a worthwhile exercise. Also, I wanted to learn how to build a neural net and this seemed like a really good dataset to do that with. :)

Unfortunately, it is the afternoon December 20, 2017 and as I put what I thought would be the finishing touches on the paper (specifically the feature section), I realized that I made a pretty big mistake and I've decided that honesty is the best policy. To sum it up, I trained my model using information about the dropoff datetime, but the whole basis of my paper was going to be predicting the duration without using that information. Honestly, I think I developed "tunnel vision" for the past month or so after looking at the Kaggle competition that started me down this road, which includes the dropoff datetime as an available feature [5].

I have currently started another Tensorflow job to try to train a new neural net on the data without the dropoff datetime information; however, it takes about 8-12 hours for the job to finish the 10000 iterations. I have it saving every 500 iterations, so I will stop the job tonight and do a quick analysis of the iterations it has finished to include in the paper, even if the results are not as good as I hoped.

I sincerely apologize for any inconvenience this causes you. While I understand there is probably going to be some points taken off for such a blatant mistake in feature generation, I hope my honesty is appreciated and allows for the focus to be on my efforts instead of on the outcome.

Have a good winter break! Please let me know if there's anything I should do to fix this issue.

Brittany Nicholls

II. Introduction

This paper explores using a neural network in order to predict the duration, in minutes, of a taxi ride for Green Taxis in New York City. The model is built only using information that would be available at the start of the ride which means that the trip route is not known.

Fig. 1. Map of NYC with the green taxi pickup areas colored in green. [2]

Most people are familiar with the iconic New York Yellow Taxis; however, there are a second type of taxis in New York City called Green taxis which might also be referenced as Boro taxis. Green taxis were implemented in 2013 after the government realized that 95% of Yellow taxi pickups were in central Manhattan, LaGuardia Airport, and JFK Airport [1]. You can see a map of the green and yellow zones in Figure 1. Note that Green taxis cannot pick up passengers from the yellow zone.

Figure 1 shows us that the rides for Green taxis are very different than the rides of Yellow taxis. Yellow taxis for the most part stay around Manhattan, while the Green taxis are all over the city. This means that, for example, the way the drivers find passengers, the traffic the drivers deal with, the routes that drivers take and the number of stoplights the drivers deal with are different between Yellow taxis and Green taxis. In addition, the number of trips that Green taxis and Yellow taxis make in a

day are very different. Doing a basic comparison between the 2016 taxi datasets available at NYC Open Data, we see that there are 16.4 million records for Green taxis from January 1, 2016 to June 30, 2016 [3] while there are 133 million records for Yellow taxis for the whole year of 2016 [4].

There has been previous work on these datasets which has focused on both the datasets for Yellow and Green taxis; however, given the obvious differences in taxi trip details as well as data size, this report documents my attempt to model trip duration for only the Green taxi data. The best results were achieved using a basic feed-forward neural net to minmize the root mean squared error (RMSE) of the rounded trip duration in minutes; however, these results did use

A. Motivation

Originally, I was going to try to work on the problem of taxi trip duration prediction introduced in a Kaggle challenge [5]. However, as I explored existing work surrounding this problem, I realized that when most papers look at the NYC taxi data, they either combine Yellow and Green taxi data [6], or only focus on Yellow taxis [5][7]. Not to mention, most work available has been done by students or through Kaggle.

Predicting taxi trip duration is important for multiple reasons. First, trip duration can be used by a customer to schedule their ride or to estimate the fare for a trip. Second, it could allow taxi companies to start developing, or improving, their ability to chain together rides to increase profit for a driver. Lastly, having a baseline for what the expected trip duration would allow taxi companies to detect when drivers are taking their passengers on the "scenic route" in order to charge the passengers more. Note that in these cases, having an exact time prediction in seconds is not necessary as a general "it will take x minutes" will suffice. This is why the model is going to train on the duration of the trip in rounded minutes and not seconds.

III. Related Work

As I mentioned above, the work on this dataset has been done informally through school projects or Kaggle competitions; however, there is other work related to taxi data.

A. Existing Work on NYC Taxi Trip Duration Prediction

In July 2017, Kaggle issued a challenge to Kagglers to work on a pre-selected NYC taxi trip dataset in order to predict the trip duration; however, the focus of the competition was to encourage collaboration, so the top publically available kernels are focused on exploring the data in a clear way to benefit the group [5]. In addition, the few publically available codebases were not from top performers [8] [9] [10]. Since it was still a competition to predict the trip duration, Kaggle still provided the means for evaluating the models using RMSLE. After evaluating using RMSLE loss function, I decided to go with RMSE as my loss function instead. I will go into more detail in section VII-A.

Additionally, other students have used this dataset to predict trip duration and made their work publically available; however, none of them used neural nets, which at the moment are a "hot" thing to do because of their ability to fit to a lot of datasets fairly well. In one paper, they used linear regression and random forests in order to build their model to predict the number of minutes the trip will take and achieved a low RMSE of 5.24 [11]. Another project achieved an RMSE of 4.87 by using a gradient boosting regressor [7]. I used the second paper that achieved a RMSE of 4.87 as a baseline, which will be discussed in section VI.

B. Other Work on Taxi Data

Ferreira et. al. have also used the NYC Taxi data in order to discuss how to best visualize and query datasets such as the NYC taxi data [12]. While their work is interesting and does use the NYC taxi data, it does not attempt to make any predictions on the trip duration.

It should be noted that in 2015 there was another Kaggle competition to predict the trip duration for taxi data from Porto, Portugal [13]. One of the top ranking results were published by a group from IBM in which they discuss how they would predict the final destination based on the beginning trip trajectory and would use that to predict the trip duration [14]. However, they were predicting the trip duration based on the initial route of the taxi after picking up a customer, which contains very different data than is available for the NYC taxi data.

C. Other Work on Trip Duration Prediction

Work has been done to predict when a bus will arrive at its stop using an algorithm based on Kalman Filtering [15]; however, their work was focused on data collected in India and they needed to be able to analyze the data real time. First, traffic in India is probably different from traffic in NYC. Second, bus routes and stops are well defined while taxi routes and start/end locations are not pre-determined until a customer gets into a taxi.

More work on bus arrival prediction has been done by Biagoni et. al. in which they use a smartphone that is placed on a bus to predict when the bus will arrive at its next stop [16]. Once again, this is bus data and relies on real time data for a specific route with specific stops.

Additionally, there has been quite a bit of work on predicting travel time on freeways [17] [18]. However, driving in a city is very different from driving on a freeway.

IV. Data and Data Challenges

A. NYC Open Data

New York City made the data for January 2016 - June 2016 of the trip records for the Green taxi data publically available [3]. There are approximately 16 million rows. This data contains the following information:

- Vendor ID : One of two vendors
- Pickup and Dropoff Date time: Time and Date when customer was picked up and dropped off
- RateCode ID: Rate (and the code) change based on where the customer is going
- Trip Type: Was the taxi hailed or dispatched

- Pickup and Dropoff Location: Contains the lat/long for both of these locations
- Passenger count: Driver recorded number of passengers in the vehicle for the ride
- Information about the fare: Base rate, taxes, tips

I enriched this dataset with two other datasets: weather and sunrise/sunset times.

B. Weather

Since traffic and taxi use can be heavily influenced by weather, I pulled down some basic stats about the weather for the taxi data being analyzed. The weather for April and March was retrieved from a site run by NOAA [19]. There was one record per day containing:

- Minimum/Maximum/Average temperature
- Precipitation: The amount of rain that fell. Note that a value of T stands for trace
- New Snow: The amount of new snow that fell
- Snow depth: How deep the snow was

Note that both the New Snow and Precipitation columns contained a few values of "T", which stands for trace. This means that very little snow/rain fell and it was not able to be measured. Records that had a value of "T" were assigned the value of 0.00001 based on a recommendation in the Journal of Service Climatology [20].

C. Sunrise/Sunset

Lastly, the taxi data was enriched with sunrise and sunset information. Note that the sunrise and sunset can affect traffic because the sun may get into people's eyes, causing them to slow down. The sunset and sunrise times were retrieved from a site owned by the US Navy [21].

V. Data Cleaning and Features Used

The Green Taxi data used for training and testing was cleaned and filtered. Mostly, the data needed to be filtered because there was so much of it and the limited resources on the computer being used to build the model would not have supported training a neural net on millions of datapoints. Instead, about a million points were used for training, 100k were used for validation, and 300k were used for testing. All of the training/validation/testing data came from April of 2016. In addition, the set of filtered data for March of 2016 was also extracted to see how well the model might transfer to a different month of taxi rides.

One thing to be aware of is the definition of the Vincenty distance. Basically, the Vincenty distance is an approximation of the distance between two geo points "as the bird flies". Note that this distance isn't exact, but it is fairly close. This distance can be calculated at the beginning of the trip using the geo points, so it is one of the features of this dataset.

Some of the steps taken to filter the data were:

1) Remove bad geo points: Some of the pickup and dropoff lat/longs were null, one of the points would be 0, or the points would be in areas outside the area of interest (NYC). In fact, a polygon was loosely drawn around the area shown in Figure 1 and the geo for the pickup and dropoff points had to be in that polygon.
2) Rate Codes: The data was filtered for either negotiated rates or the standard rates. Since there was a small percentage of trips to LaGuardia or JFK, these were removed, as were records with an "unknown/bad" rate code.
3) NonZero Trip Distance: The original dataset provided the trip distance as recorded by the odometer. If this recorded trip distance was 0, the trip was removed
4) Trip Distance ¡ Vincenty Distance - 0.1: If the distance the odometer recorded is less than the Vincenty distance by more than 0.1 miles, the record was removed.
5) Low Trip Cost/High Fare per Miles: Records that had low low trip cost or high fare per miles were determined to be bad data and removed. On one hand, drivers are not going to take a trip that will cost them money and the threshhold set was $0.50/mile, but most riders are not going to pay a lot of money to go a small distance, which threshhold was determined to be $50/0.1 mile - about $50 a block.
6) Very Long/Short Durations: While it is possible that a taxi might only take a passenger for a trip of less than 30 seconds or that a customer might rent a taxi for more than 3 hours, these were considered outliers and records were removed. 3 hours was the upper end because traffic can significantly lengthen a trip.

In essence, the datasets are for green taxi trips that start and end in New York City (not including the airport). The data was filtered to try to catch primarily one way rides that are a reasonable cost/distance/duration for the trip itself, the driver and the passenger.

After filtering the data as described above, some features were generated based off of the assumption that the information would have to exist at the beginning of the trip. In the main model being discussed, the following features were used:

- Pickup/Dropoff Location: Lat/Long for pickup and dropoff location
- Minute of Day for Pickup/Dropoff: The minute of the day that the pickup and dropoff was made - This is the mistake that I mention in the beginning
- Day of the week for Pickup/Dropoff: A one-hot encoded value representing each day of the week
- Passenger Count: How many passengers were in the vehicle
- Vincenty Distance in miles
- Weather: Average temperature, amount of rainfall, amount of snowfall
- Inverse of the number of minutes after Sunrise/until Sunset: Only for a 1.5 hour window after/before sunrise/sunset. 0 for the rest of the times.
- Rate Code ID, Trip Type, Vendor ID:One hot encoding was performed for these features.

VI. Baseline Model

The baseline used was proposed by Jaiwal et. al. [7]. Basically, their best-performing approach was to use k-means to create 40 clusters of the geo points for the pickup/dropoff locations and then train a Gradient Boosting Regressor on a similar feature set to mine [7]. They also used RMSE to calculate their error and ended up achieving an RMSE of 4.87 for duration in minutes.

Because they worked over a different, but related dataset, I decided to try to implement their algorithm over my data. Unfortunately, they only had a significantly smaller training dataset and when trying to run sklearn's Gradient Boosting Regressor over the green taxi training dataset, the computer ran out of memory. Therefore, a smaller sample of the training dataset (50k samples) was used and after performing their algorithm over the smaller dataset, the calculated RMSE was 5.66.

Once again, I realized that my mistake in including dropoff time information in the feature set that was run through this algorithm probably affected the results. Given this realization, the calculated RMSE for this baseline can only be used for the neural network model that also used this information. However, I cannot compare the neural network model I attempted to train without the dropoff datetime features against this model.

VII. Predictive Model

A. Loss Function

I debated between using two loss functions, RMSLE and RMSE. Utlimately, I choose to use RMSE and will explore why in this section.

Kaggle decided to use RMSLE [5]:

$\epsilon = \sqrt{\frac{1}{n}\sum (log(p_i + 1) - log(a_i + 1))^2}$

with a little bit of manipulation of the logs, this transforms into:

$\epsilon = \sqrt{\frac{1}{n}\sum log(\frac{p_i+1}{a_i+1})^2}$

This shows that, in essence, the model is heavily punished when $p_i < a_i$ as compared to when $p_i > a_i$, even if the difference is the same. This means that this model will try to avoid predicting under the actual value, since the loss isn't as severe if it goes over. Thus, minimizing the RMSLE loss is better for finding a model that will predict the minimum trip duration. Furthermore, because of it's use of the log function, RMSLE would be good for numbers that are going to be large. For instance, if we were to predict the trip duration in milliseconds or seconds (as the Kaggle competition did), then RMSLE might be a better choice; however, I choose to focus on predicting the trip duration in minutes, not seconds.

RMSE is calculated as follows:

$\epsilon = \sqrt{\frac{1}{n}\sum (p_i - a_i)^2}$

which shows that the model will get equally punished for predicting too large of a value as it will too small of a value. Thus, I choose to minimize the RMSE when building my model.

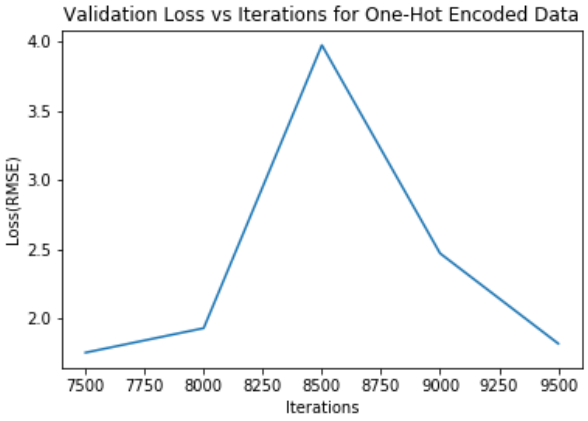

Fig. 2. Validation error (RMSE) for the best performing neural network

B. Neural Network Settings for Best Results

Utimately, the best results came from a feed-forward neural network with two hidden layers. 38 features were fed into the network and each hidden layer had 40 neurons. Note that these were the features discussed above. The activation function used for each neuron was the rectified linear unit. Instead of using the plain gradient descent optimizer available through Tensorflow, the Adam Optimizer was experimented with because of the larger number of features **??**.

Up to 9500 steps were taken, with each step training on 300 randomly selected datapoints. Every 500 steps, the model was saved off; however, Tensorflow only retains the 5 newest models, so the data was validated on the model after 7500, 8000, 8500, 9000, and 9500 iterations.

At the end, the validation data was used to determine which iteration would be the best model. Figure 3 shows that, interestingly enough, the model had the lowest error after 7500 iterations and was very closely followed by the model after 9500 iterations; however, there is a spike at 8500 iterations in which the error doubles. This probably indicates that the model had found a local minima after 7500 steps, but was moved out of it during the following steps. Thus, it was decided to use the model at 9500 to evaluate the RMSE of the test data. The RMSE of the test data for this model is 2.12, which is less than half of the baseline RMSE for this data.

The jump in error at 8000 iterations for the validation data could be due to the algorithm stepping out of a local minima to the function, but it is possible that the model was starting to overfit. The fact that the validation error became lower as more steps were taken points to the fact that this may not be the case.

Technically, the validation data came from data that might be very similar to the training data, so it may not have picked up the possibility of overfitting. For curiousity's sake, the graph of the iterations vs the test error has been provided in Figure **??**.

This indicates that the data is very likely not being overfit since the model didn't see any data that shares the same dates

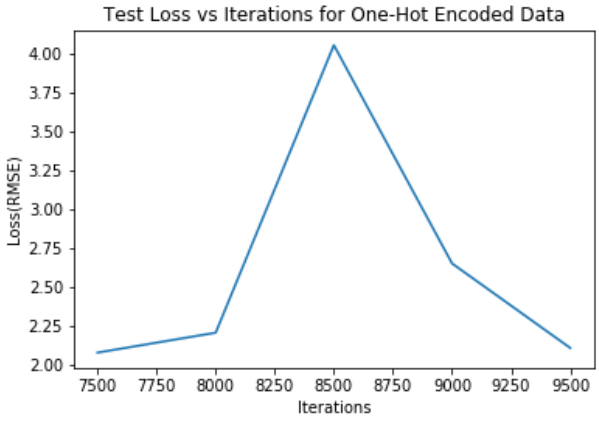

Fig. 3. Test error (RMSE) for the different iterations of best performing neural network

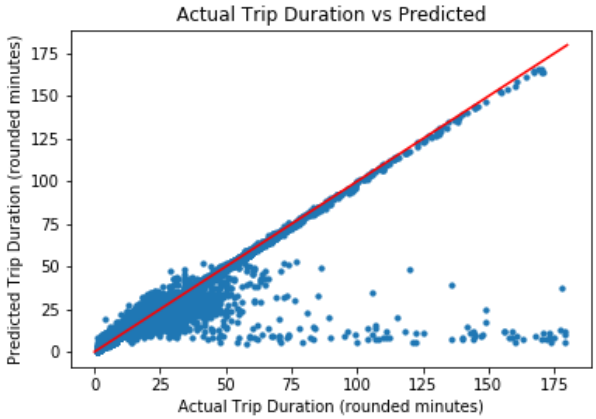

Fig. 4. Scatterplot of the actual durations vs the predicted durations

as the testing data. Furthermore, in section VII-C, we explore how well this model transfers to the green taxi data, but for March instead of April.

1) Visualizing Results: While it's great that the RMSE was minimized and is half of that of the baseline model, it is important to make sure that the actual results look right. Figure 4 visualizes the actual trip time (in rounded minutes) against the predicted trip time (in rounded minutes - done after the raw prediction). The red line in the figure represents the correct predictions. If predicted values are fairly close to the actual values, the points should be grouped around the red line, and as you can see, while not perfect, most of the predictions are near the red line.

C. Model Against March Data

In order to see how robust the model was, we tried the model built using the training data from April against the March dataset. The RMSE of the March data against this model is 2.24, which was significantly lower than might be expected.

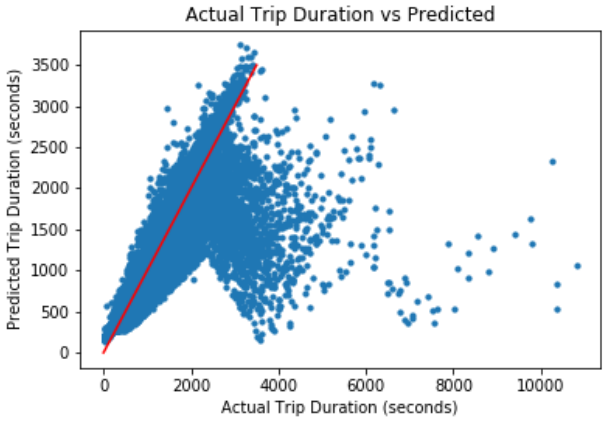

Fig. 5. Scatterplot of the actual duration (in seconds) vs the predicted durations (in seconds) for the model minimizing RMSLE

D. Alternative Settings Considered

Unfortunately, it is difficult to perform proper cross validation on a neural net, especially given the shorter timeframe and limited computing resources. Each neural net took 8-12 hours to train with 10000 iterations. I had tried training the neural net using tanh as the activation function, but the loss on the training data for that was poor enough that I switched to relu, which had much better/slower loss results. We tried training against the non one-hot encoded data to see if the neural net could accomodate the more complicated features, but the RMSE for that was above 10, which was even worse than the baseline; that neural net also used the basic gradient descent optimizer and each hidden layer only had 25 neurons, which both may have been a cause for the higher error.

1) RMSLE: I did spend some time trying to build a neural net for the Kaggle competition in the beginning to familiarize myself with the data and Tensorflow. I was actually able to achieve a RMSLE of 0.29184, which is fantastic given that the top ranked score was 0.28976, with second place achieving 0.30664. Unfortunately, we cannot directly compare the two scores because 1) I did not use the Kaggle data and 2) there is a possibility my data was a lot cleaner than the Kaggle datasets; however, this was a strong indication that building a neural net using these features was the route to go. Additionally, Kaggle did allow the model to have access to the dropoff datetime during training and testing, so fortunately, this work wasn't affected by that discovery.

Figure 5 shows the actual values vs the predicted values.

E. Predictive Model without Dropoff Time Information

As I mentioned above, it completely slipped my notice that I was training my models on training data that had the pickup time in it. In order to try to make up for this, I tried to run the model discussed above over the same feature set but without the dropoff time related information. The model had to be stopped early for time reasons. Unfortunately, the RMSE for both the validation and test sets were less than ideal. The

lowest RMSE of the validation set was 6.55 at iteration 6500 and the RMSE of the test data for the same model was 6.31. These numbers might have minorly improved had there been more steps taken, but most likely not enough to say that this model would beat the baseline.

In great news, this does prove that the neural net above was trained to heavily use the dropoff datetime related features. This is good, in a sense, because knowing the dropoff time does give you a distinct advantage and the neural net should have designed a function around relying on that data as it would be the most accurate way to predict the duration of the trip.

VIII. Future Work and Ideas

Ideally, the future work would have included ideas to potentially improve upon the model; however, after realizing that I had been training on data with the answer, in essence, in the feature set I would need to continue to improve upon this model. Ultimately, I would focus on constructing the neural net differently by changing the activation functions as well as the number of neurons in the hidden layers. I would also explore having different activation functions on different layers. Lastly, I would see if using a third hidden layer would improve the model.

In all honesty, I was expecting to spend a bit of time today to explore doing some sort of ensembling between the RMSLE predictions in seconds and the RMSE predictions in minutes to estimate the minutes. If you look at Figure 5, it seems that this approach was good for predicting trip duration for shorter trips; however, it was not good at predicting longer trips. Figure 4 shows that that approach could better predict the longer trips, although it's hard to see how well it worked on the shorter trips due to the cloud around the shorter trips. One of the ensembling methods I was going to look at was going to be using a gradient boosting regressor; however, I was also going to try just taking the RMSLE prediction in seconds, converting it to minutes, and taking the rounded average between that and the RMSE prediction that was doing so well.

Some other ideas for after creating a good model were to introduce small geoboxes covering NYC to allow for some geo error. This is similar to what the clustering did in the baseline algorithm; however, the clustering is going to be affected by the pickup and dropoff locations. In addition, I would explore using a neural net to predict the trip distance, as this was data that was filtered out. The predicted distance could potentially be fed into the neural net that predicts the duration, instead of using the Vincenty distance. Lastly, there has been work on predicting when and where taxis are going to be needed based on zones [6]. This information could be used in order to predict where taxis are going to be needed which can, in a way, predict traffic.

IX. Conclusion

Unfortunately, neural nets were not able to meet the goal of this report of predicting NYC green taxi trip durations based off of information available at the start of a taxi ride; however, it is important to note that this is because I made a mistake and did not realize early enough that the neural nets were being trained on data that had the features related to the dropoff datetime information. I made a beginners error, and I made it early enough that it was propogated through all aspects of this project starting with the baseline.

It is still valuable that the work that has been done is presented because 1) it exhibits the very painful lesson I have learned of needing to triple check the reason for including every feature, 2) the model that I did create could be used after a taxi trip as a potential baseline in which to try to predict anomalous taxi trips against and 3) it does show that neural nets will use the best features to their utmost advantage. In this case, I had used the pickup and dropoff time (minute in the day) to predict the duration. Any function trying to do this prediction should very definitely try to use as much information as it can from the pickup and dropoff times. As we saw, when the neural net lost the information about the dropoff times, the RMSE increased because the predictions were worse. While it is not good that the neural net without the the dropoff time features performed poorly, that is very likely due to not be configured properly. Based off of this report, I wouldn't stop trying to use neural nets in predicting taxi times, instead, I would focus on tuning the neural net for the correct feature set.

References

[1] The City of New York. (2017) Background on the boro taxi program. [Online]. Available: http://www.nyc.gov/html/tlc/html/passenger/shl_passenger_background.shtml

[2] ——. (2017) Your guide to boro taxis. [Online]. Available: http://www.nyc.gov/html/tlc/html/passenger/shl_passenger.shtml

[3] Taxi and Limousine Commission (TLC). (2017, Jun.) 2016 green taxi trip data. NYC OpenData. [Online]. Available: https://data.cityofnewyork.us/Transportation/2016-Green-Taxi-Trip-Data/hvrh-b6nb

[4] ——. (2017, Sep.) 2016 yellow taxi trip data. NYC OpenData. [Online]. Available: https://data.cityofnewyork.us/Transportation/2016-Yellow-Taxi-Trip-Data/k67s-dv2t

[5] Kaggle. (2017) New york city taxi trip duration. [Online]. Available: https://www.kaggle.com/c/nyc-taxi-trip-duration

[6] S. Z. J. Z. Yunrou Gong, Bin Fang. (2016, Sep.) Predict new york city taxi demand. NYC Data Science Academy. [Online]. Available: https://nycdatascience.com/blog/student-works/predict-new-york-city-taxi-demand/

[7] P. J. T. S. Himanshu Jaiwal, Tushar Bansal, "Nyc taxi rides: Fare and duration prediction," University of California, San Diego, Tech. Rep., 2017. [Online]. Available: https://cseweb.ucsd.edu/classes/wi17/cse258-a/reports/a077.pdf

[8] Currie32. (2017, Jul.) Nyc-taxi-trip-duration. [Online]. Available: https://github.com/Currie32/NYC-Taxi-Trip-Duration

[9] yukw777. (2017, Aug.) kaggle-nyc-taxi. [Online]. Available: https://github.com/yukw777/kaggle-nyc-taxi

[10] mk9440. (2017, Oct.) New-york-city-taxi-trip-duration. [Online]. Available: https://github.com/mk9440/New-York-City-Taxi-Trip-Duration

[11] A. F. A. J. Christophoros Antoniades, Delara Fadavi, "Fare and duration prediction: A study of new york city taxi rides," Stanford University, Tech. Rep., 2016. [Online]. Available: http://cs229.stanford.edu/proj2016/report/AntoniadesFadaviFobaAmonJuniorNewYorkC report.pdf

[12] H. T. V. J. F. C. T. S. Nivan Ferreira, Jorge Poco, "Visual exploration of big spatio-temporal urban data: A study of new york city taxi trips," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 12, Dec. 2013. [Online]. Available: https://vgc.poly.edu/ juliana/pub/taxivis-tvcg2013.pdf

[13] Kaggle. (2015) Ecml/pkdd 15: Taxi trip time prediction (ii). [Online]. Available: https://www.kaggle.com/c/pkdd-15-taxi-trip-time-prediction-ii

[14] A. P. Y. G. Hoang Thanh Lam, Ernesto Diaz-Aviles and B. Chen, "(blue) taxi destination and trip time prediction from partial trajectories," IBM Research - Ireland, Tech. Rep., 2015. [Online]. Available: https://arxiv.org/pdf/1509.05257.pdf

[15] R. S. L. Vanajakshi, S.C. Subramanian, "Travel time prediction under heterogeneous traffic conditions using global positioning system data from buses," *IET Intelligent Transport Systems*, vol. 3, no. 1, Mar. 2009.

[16] T. M. J. E. James Biagioni, Tomas Gerlich, "Easytracker: Automatic transit tracking, mapping, and arrival time prediction using smartphones," in *ACM Sensys*, 2011.

[17] N. G. Mehmet Yildirimogl, "Experienced travel time prediction for freeway systems," in *2012 15th International IEEE Conference on Intelligent Transportation Systems*, 2012.

[18] D.-C. S. M.-H. C. Chun-Hsin Wu, Chia-Chen Wei and J.-M. Ho, "Travel time prediction with support vector regression," *IEEE Transactions on Intelligent Transportation Systems*, vol. 5, no. 4, Dec. 2004. [Online]. Available: http://www.csie.nuk.edu.tw/ wuch/publications/2003-itsc-svr.pdf

[19] NOAA. Nowdata - noaa online weather data. National Weather Service Forecase Office. [Online]. Available: http://w2.weather.gov/climate/xmacis.php?wfo=okx

[20] M. A. Adnan Akyux, Karsten Shein, "Procedure for assigning a value for trace precipitation data without changing the climatic history," *Journal of Service Climatology*, vol. 6, no. 1, 2013.

[21] U. S. N. Observatory. [Online]. Available: http://aa.usno.navy.mil/data/docs/RS_OneYear.php