Project report for final project
CS 6685

Team Members:
Kanak Tenguria, Namita Raghuvanshi, Chet Garlick

Group Number: 11

1 Introduction

Our goal is to predict duration of trip of taxi in New York City. This can be very useful and practical in real world scenario where the customer is shown Estimated Time Arrival(ETA) giving them insight of how much time it is going to take to reach from one place to another. This can also help cab companies in understanding when the cab will be free and how to efficiently assign it for another trip. Many factors influence this prediction including traffic, weather, speed, etc.
This problem is available as a Kaggle competition (`https://www.kaggle.com/c/nyc-taxi-trip-duration`) and a lot of work has already been done on this problem. But since the rewards for this competition were given on the basis of collective learning done rather than leaderboard, most of the work available on Kaggle generally focuses on teaching basic data analysis and some kind of regression. You can find all the work done on this problem which is available on Kaggle here: `https://www.kaggle.com/c/nyc-taxi-trip-duration/notebooks`

2 Data

The data is downloaded from one of the competitions from Kaggle website. It is based on the 2016 NYC Yellow Cab trip record data and was originally published by the NYC Taxi and Limousine Commission. This data represents logs of taxi trips from NYC including pickup location and drop off location along with many other things, mainly the duration of trip. The data fields are:

- id - a unique identifier for each trip
- vendor_id - a code indicating the provider associated with the trip record
- pickup_datetime - date and time when the meter was engaged
- dropoff_datetime - date and time when the meter was disengaged

- passenger_count - the number of passengers in the vehicle (driver entered value)
- pickup_longitude - the longitude where the meter was engaged
- pickup_latitude - the latitude where the meter was engaged
- dropoff_longitude - the longitude where the meter was disengaged
- dropoff_latitude - the latitude where the meter was disengaged
- store_and_fwd_flag - This flag indicates whether the trip record was held in vehicle memory before sending to the vendor because the vehicle did not have a connection to the server - Y=store and forward; N=not a store and forward trip
- trip_duration - duration of the trip in seconds

Here, the trip_duration field will be treated as label.

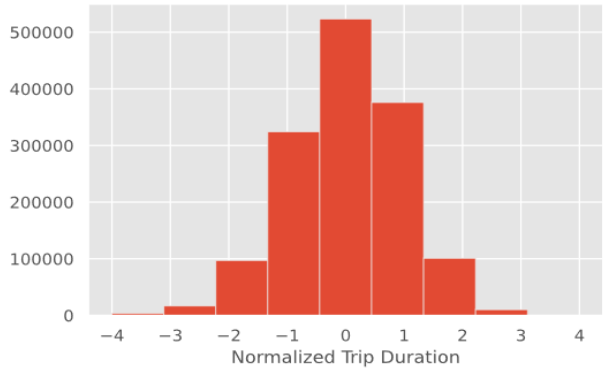
id	vendor_id	pickup_datetime	dropoff_datetime	passenger_count	pickup_longitude
id2875421	2	2016-03-14 17:24:55	2016-03-14 17:32:30	1	-73.982155
id2377394	1	2016-06-12 00:43:35	2016-06-12 00:54:38	1	-73.980415
id3858529	2	2016-01-19 11:35:24	2016-01-19 12:10:48	1	-73.979027
id3504673	2	2016-04-06 19:32:31	2016-04-06 19:39:40	1	-74.010040
id2181028	2	2016-03-26 13:30:55	2016-03-26 13:38:10	1	-73.973053

pickup_latitude	dropoff_longitude	dropoff_latitude	store_and_fwd_flag	trip_duration
40.767937	-73.964630	40.765602	N	455
40.738564	-73.999481	40.731152	N	663
40.763939	-74.005333	40.710087	N	2124
40.719971	-74.012268	40.706718	N	429
40.793209	-73.972923	40.782520	N	435

2.1 Data analysis and feature engineering

We have total 1.458 million records in our dataset. There were no null values in the data. We have dropped the following columns: id, vendor_id, passenger_count and store_and_fwd_flag as they does not seem useful in prediction of taxi trip duration.
pickup_datetime and dropoff_datetime columns are datetime object of dataframe

and are used to extract useful information like day of the week, minute of the day and day of the year. After this, we have dropped pickup_datetime and dropoff_datetime as well because they represent datetime object and cannot be used for learning.
Columns related to latitude and longitude of pickup and dropoff location are used to calculate distance of the trip. It helped in removing outliers like very long distance trips or trips with zero distance. trip_duration smaller than 1 second and greater than 4 hours are also considered as outliers. The distance was also used along with trip_duration in calculating speed. It was used to remove outliers like trip with very high speed (120 mph).
We also tried looking for weather information for the data because weather can be influencing factor in duration of trip but we were not able to find anything useful. Once the analysis and feature engineering was done, we applied necessary transformation and normalized the data and it was ready for learning.
Below is the bar graph of transformed trip duration as target feature.



2.2 Selected features

For learning purpose, only four fields from the original dataset was used and three fields are generated using pickup_datetime and dropoff_datetime fields. The data fields which we are going to use for learning are: pickup_longitude, pickup_latitude, dropoff_longitude, dropoff_latitude, minuteofday (Minute of the day), week_day (Day of the week) and year_day (Day of the year) and trip_duration is going to be our target feature.

pickup_longitude	pickup_latitude	dropoff_longitude	dropoff_latitude	trip_duration	minuteofday	week_day	year_day
-0.472735	-0.025804	0.083277	0.315378	1.078369	-0.455332	-0.025569	-0.694934
-0.544152	-0.933187	-0.224513	-0.958336	-0.071764	-2.141517	0.997960	1.651570
-0.048001	0.174063	0.084324	0.328771	-0.482517	-0.791008	0.486195	-1.218534
-0.852833	-0.610667	-0.504665	-0.078649	0.046229	0.072902	-1.560863	-0.617364
-0.349342	-0.409312	1.580350	-0.110016	1.363674	1.288100	-1.049099	0.352266

3 Evaluation Metrics

We are using Mean Squared Error as our evaluation metric. It measures the average squared difference between the estimated values and what is estimated.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Our goal is to minimize the MSE as much as possible. But first we need to consider a baseline to compare our results to.
We are using sklearn library for calculating MSE (`https://scikit-learn.org/stable/modules/generated/sklearn.metrics.mean_squared_error.html`).

4 Methods

4.1 Baseline: Mean as predicted value

First, we created a model using the mean value as the predicted value for each test data point and find MSE for it. This worked as our baseline for all other methods. MSE for this model is 1.000001609325409. We have to ensure that all the approaches we are using give us better result then this.

4.2 Regression

We are using 2 regression techniques, Linear Regression and Random Forest Regression because regression describes the relation between set of independent variables and dependent variable and it can be used to explain how the changes in each independent variable are related to changes in the dependent variable. Here trip duration will be dependent variable and things like pickup and dropoff location and date and time are all independent variables. Another reason for using regression is because we are dealing with continuous values.

4.2.1 Linear Regression

In linear regression, we try to model the relationship between all the independent variables and dependent variable by fitting a linear equation. We have also used 3-fold cross validation while evaluating the score.
We have used sklearn library to implement Linear regression and cross validation. (`https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html`)

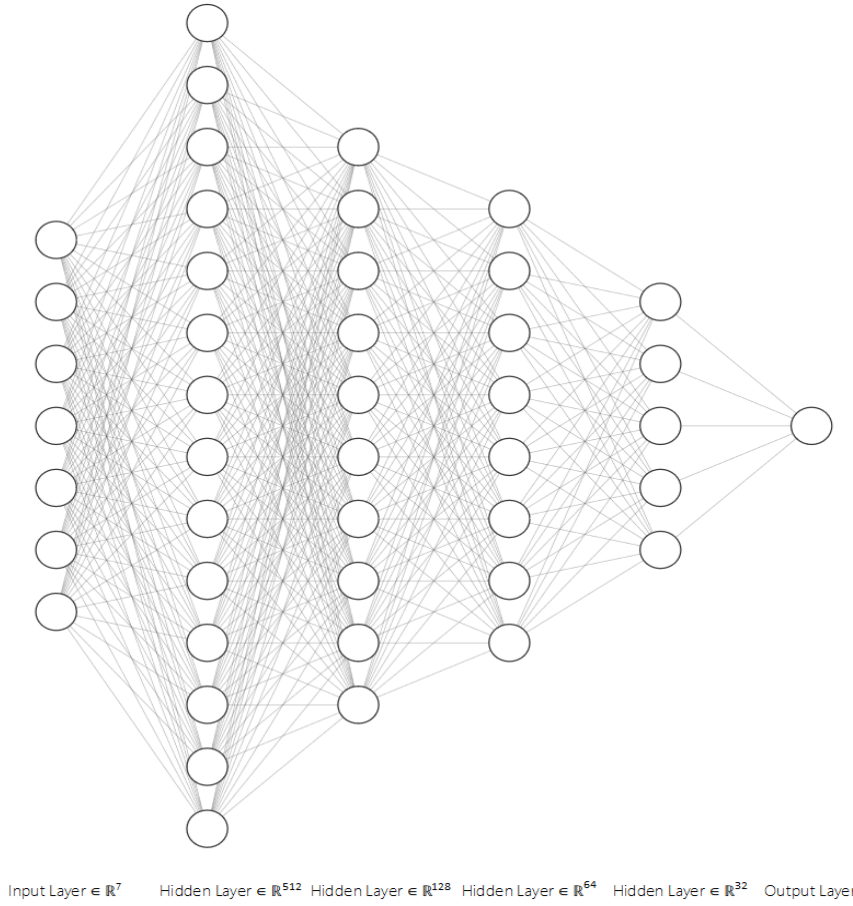
4.2.2 Random Forest Regression

We have used Random Forest regressor because it runs efficiently on large databases and it is very robust and accurate in terms of performance and results. It makes predictions by combining decisions from a sequence of base models.

Unlike linear regression, random forests are able to capture non-linear relationship between the features and the target. We are using 10 decision trees and "auto" as our max_features. We have used 3 fold CV here to evaluate score.

4.3 Neural Network

We have used Sequential model from keras library for implementation of Neural Network (`https://keras.io/models/sequential/`).
Our model has 7 inputs and one output where the output represents predicted trip duration. We have used 4 hidden layers which are fully connected. First layer consist of 512 nodes, second layer has 128 nodes, third layer has 64 nodes and fourth hidden layer has 32 nodes. Each layer is followed by batch normalization, ReLU activation function and Dropout of 10%. We have used ADAM optimizer here and Mean squared error as our loss. Learning rate for this model is 0.001. We have used batch size of 2048. We have trained the data for 100 epochs. We are using 33% data for testing.



5 Results

Results obtained using four methods are shown in table below.

Method	Parameter	MSE
Mean as predicted value	none	1.0000016
Linear Regression	default	0.9089858
Random Forest	Decision Tree = 10 max_features = auto	0.2238257
Neural Network	Learning Rate = 0.01	0.2351336
Neural Network	**Learning Rate = 0.001**	**0.2198500**
Neural Network	Learning Rate = 0.0001	0.2261821

As we can see, our goal was to minimize MSE and perform better then results from Mean as predicted value. Linear regression showed some improvement but not much with MSE of 0.9089858. Random forest and neural network performed well. With random forest, we get MSE of 0.2238257 whereas with neural network with best parameters, we get MSE of 0.2198500. Neural networks in this case outperforms Random Forest.

5.1 Parameter selection

While training random forest, we tried different number of decision trees and max_features and found best results with decision tree = 10 and max_feature = auto.
For neural networks, we started with two hidden layer with 200 nodes each and ReLU activation but it didn't worked and we were getting nan as our loss after every epoch. After some experimentations, we added more layers and decreased the nodes in later layers because 200 inputs to 1 output caused some problems. Initially we trained the model on 50000 datapoints. This increased our speed of experimentation and frequency of monitoring. For mini batch size on whole dataset, we set the size to 512 initially. After tuning other parameters, we increased it to 1024 then to 2048 to see the changes and finally we ended up setting it to 2048 after realizing that it is taking less time than 512 and 1024 epochs. We started with very small learning rate and increased it by multiple of 10 to see changes. We settled on 0.001 at the end. There were many more experiments done by all the three members of the group but only major changes are reported.

5.2 Previous work

Most relevant work we found on this problem was in this paper: `http://www.andrew.cmu.edu/user/zijingg/Taxi_duration_prediction_report.pdf`.
But we were not able to compare our results properly with them because of the way they transformed the data was different and the data itself is different (includes 60 million taxi rides recorded by New York City taxicabs from 2017). They were using Mean absolute error and Root mean squared error as evaluation

metric. We were surprised to find out that how the transformation of data affects the final results. In this paper, RMSE using Random forest is 400.39 whereas we are getting RMSE of 0.4785 using random forest. We didn't find it fit to compare results in this way.
Another source for comparison was Kaggle itself but challenge was evaluated using Root Mean Squared Logarithmic Error (RMSLE) and the test dataset was different where trip duration was not given which prevented us from comparing our results with the leaderboard. Leaderboard results can be seen here: (`https://www.kaggle.com/c/nyc-taxi-trip-duration/leaderboard`). Although we tried to calculate the RMSLE for our results but we were facing a problem. As you can see that after trasformation and normalization, some of the values of trip duration are negative. There were some difficulties in calculating RMSLE on negative values. We tried to find a solution but we had to normalize the whole data and transform it in the range of 0-1 (or something positive). It was not possible to apply different transformation and start everything over since we had to retune the model and evaluate and report the results all over again.

6 Conclusion

Although there was not much difference between results from Random forest and neural network but we can say that our neural network performed better than any regression techniques.

Biggest challenge which we faced was in understanding evaluation metrics and how to select them. What we generally used till now was percentage accuracy where we know that 100% is best and we need to reach as close to it as we can. We can say that it is not totally true in case of MSE. We found out that while it is good to reach closer to zero, it is not necessary that a value much bigger than zero is not a good answer. It pretty much depends on the data and how it is transformed. Clearly we lack some understanding to comprehend the results this way. Another important thing was the step of features selection for learning. We started with 46 features which we extracted from original data but soon after some research ended up selecting only 7.

During the development of the project, another major challenge where we hit the dead end was "nan" in our loss. After few days of research and some previous knowledge with the issue, we overcome it by changing optimizer from SGD to ADAM and reducing difference between number of nodes in later layer and output layer.

Another thing worth noting is transformation of data. Before normalization of the data, we were getting MSE in thousands but after normalization, we obtained our results in decimals. It was more easy to comprehend that way. If we had more time, we would definitely love to look more into different kind of evaluation metrics and understand how they work and how transformation of data effect the results. Also, we surely want to work on evaluating RMSLE and comparing results with leaderboard.

7 Contribution

All the team members analysed the data individually. This helped us in avoiding mistakes of each other and it also helped in looking at the data from different perspective. Once we were done with our analysis, we discussed and did the necessary transformations for final data which we used for baseline approach. After this, we simply applied random forest and changed the depth of tree and number of features to consider to analyze the results. Random forest was also applied before transformation of data. For the neural network, we started individually with the decision of having 2 hidden layers with 200 neurons. After many failed attempts, experiments, help of each other and constant communication we finalized our model and selected our final parameters. The report was shared on overleaf and necessary changes were done accordingly.