NewYork City-Taxi Fare Price Prediction

Siddharth Chittoor, Saketh Nimmala, Manasa Alla May 5, 2019

Abstract

Taxi prices in New York can be notoriously unpredictable. There are many factors that might affect the price of the taxi ride. Considering all these factors, a model has been developed to predict the taxi fare prices.The data used for prediction has been taken from Kaggle.

Keywords: Pearson correlation, K-Nearest Neighbour Regression, Random Forest Regression, Gradient Boosting Regression.

1 Introduction

The New York City taxi rides has a vibrant role in the city. The rides recorded can provide insight into traffic patterns, road blockage or large-scale events. It is important for taxi to provide visibility of their estimated fare and ride duration, since the riders expect these metrics to be provided upfront to have a vivid view on the optimal time to start their commute. Furthermore, the visibility of the fare will attract the riders during the surge pricing.

The prediction is done by the data that is available at the beginning of the ride.Various attributes such as the trip distance, start time, pickup and drop off coordinates, number of passengers have been used to estimate the fare price. Once the data is collected, the noise data is extracted and the models are built. Various regression techniques like linear regression, K-nearest neighbour, gradient boost, random forest models have been used to predict the fare amount. These predictions are compared to give the appropriate output. Python libraries like pandas, seaborn, numpy and matplotlib have been used for generating the model.

2 Review of Related Works

The taxi fare is a function of mileage and the duration of the ride(sum of time charge, distance charge and drop charge).The distance can be calculated easily but estimating the duration is a trivial task as it involves the complex traffic processes that are non- linear.

The duration prediction is done by short time pre- diction with the real time data is one of the method that can be adopted. The author in [1] tackle the problem by using the data from GPS of buses. A similar approach is used in [2], real time data from smart mobile inside the vehicle is used.

The travel time for highways has a higher accuracy than in cities. The author [3] used a combination of traffic modelling, real time data and the history of traffic to predict the travel time in congested free- ways. There are several other papers that worked on freeways. In [4] the prediction is done using Neural network and in [5] using Support vector regression.

3 Data

Our dataset had 55 million rows but we considered 2 million. It has 8 columns such as pick up longitude, pick up latitude, drop-off longitude and latitude, time of the day and fare amount. The two datasets used are called test.csv[6] and train.csv[7].

3.1 Data Exploration and Cleaning

The following things which exploration and cleaning pertains are done:

1. Datatype checks : There are object, float64 and int64 values.

2. Shape checks : 2 million rows and 8 columns.

3. Check for Nan values : We checked for nan values that are present in our dataset and found that 14 values in drop-off longitude and latitude are missing. So we dropped them from our dataset. We used the isnull() function for that.

4. Check for outliers : We found that in passengers row one column had 208 and 9. This is not possible in a taxi so we dropped them and considered passenger count less than or equal to 6. In fare amount, we plotted a graph to show distribution of the fare amount and found mostly it was under [0,80] dollars . The maximum was 1200 dollars too but we did not drop it as it may have potential information.

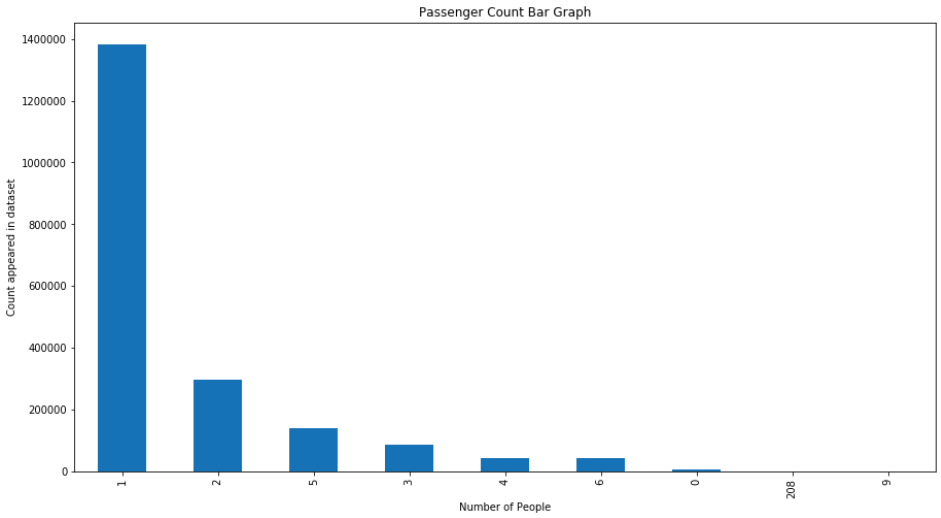

Fig 1. Passenger count bar chart

5. Cleaning the data pertaining to each attribute in the training set : In pickup longitude drop off longitude we found that few of the rows had values that were outside New York city so we kept the threshold values for those and trained our model accordingly.

6. Distance Calculation : We will calculate distance now as it is important in determining the fare of the ride. We try to use haversine distance which is a slight tweak to euclidean distance considering the cosine angle of the latitudes as the latitudes gap is more near equator and decreases as we approach north pole.The reason for this consideration is that, it might be simple to use euclidean on a high level but the distance between each longitude is about 69 miles which is a lot. As distance is very import for us, we try to be as precise as possible. There is a specific method in ' geopy ' library for distance calculation but takes too much of time.

3.2 Correlation

The Pearson correlation coefficient is used to calculate the strength of the relationship between the dependent or independent variables or attributes. Its value lies between -1 and +1. The 0 value indicates that there is no relationship between the variables. Negative value indicates that they are negatively cor- related to each other. Positive value indicates the strong correlation and similarity between the variables.

In this project, we calculated correlation among three variables. They are distance of the ride, taxi fare and the time of the day.

- Distance of the ride and the taxi fare.

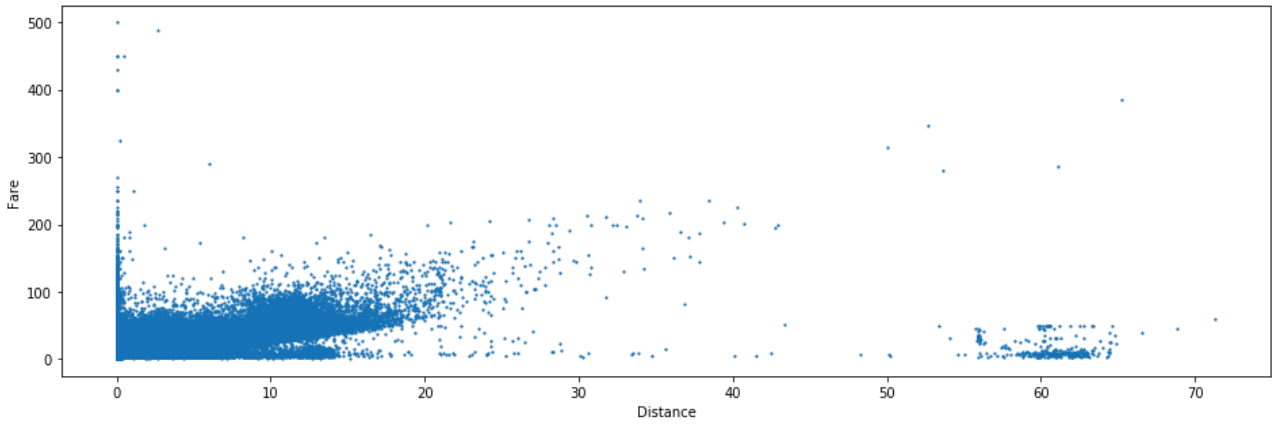

Fig 2. Correlation between distance and fare amount

- Time of day and distance traveled.

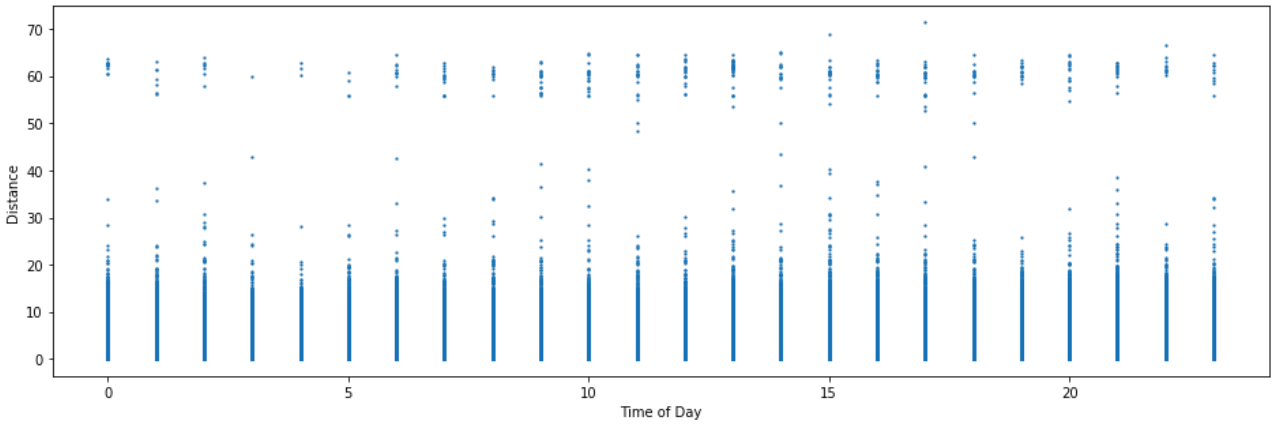

Fig 3. Correlation graph between time of the day and distance

- Time of day and the taxi fare.

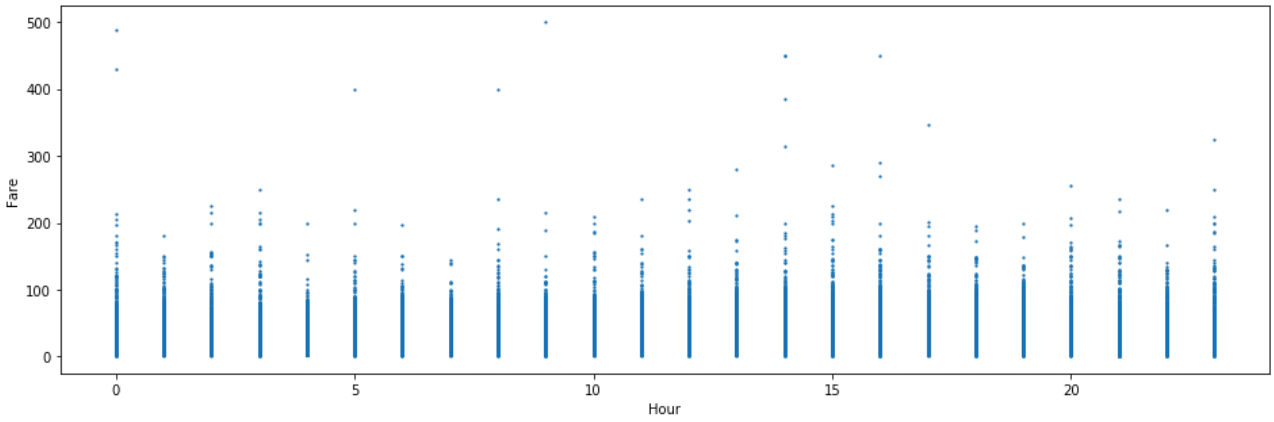

Fig 4. Correlation graph between time of the day and the fare

We have used the function stats.pearson to calculate the Pearson coefficient. This function will return a tuple which has the p-value and the coefficient. The null hypothesis is that the two variables are uncorrelated. The p-value is a number between zero and one that represents the probability that your data would have arisen if the null hypothesis were true.

The plot between the distance and fare is a linear plot which indicates that there is a correlation be- tween them. Moreover, the coefficient value is 0.8173. There are trips with zero distance but with a non- zero fare. How can a trip end at the same location of pickup? May be the drop location is updated as pickup location mistakenly or something else There are quite a few trips with distances of 60 miles but the fare is pretty less. There might be direct reasons like congestion', 'no tolls', 'rider applied discount coupon', 'outstation trips which cost less for more miles we travel as fare/mile ratio will be decreased as norms of cab company.' Overall, we generically say that there is a linear relationship be- tween distance and fare amount since pearsonr returned 0.82 which is close to 1 (directly related) and the plot evidently checks out as a linear relationship. Although there might be a vague discussion of the relationship of the right side fewer dots but generically we can call it a linear one.

Secondly, the plot between the distance and time of the day is non linear and thus the there is no particular trend in the data i.e the relation is pretty vague. Though, we can get some insights from the data. The people travel much distance between the office hours i.e 9 to 10 AM and 3 to 5PM.The people have travel less during 3 AM to 5 AM and there reason is pretty obvious that they might be sleeping. The coefficient value is -0.0290.

Thirdly, the plot between the time of the day and the fare is also non linear. We can observe that the fares at 12AM, 5AM to 10AM and 2PM to 5PM, 8PM are high. This can be the reason of surge in the area, like at 12AM it might be Friday or Saturday night people might go clubbing and wanted to return home but the cabs were at a surge due to unavailability and they book it anyway as they might be wasted!!! The coefficient value is -0.0181.

Highest Correlation: 'Distance' and 'Fare Amount' The correlation between distance and fare amount is the highest about 0.82 (Pearson Coefficient). This is clearly obvious as the fare amount is and will always be dependent on distance travelled. Basic formula for fare calculation is,

Fare = (Base)+(distance)∗(generalised rate/1mile)

4 Models and Methodology

Various regression models have been used for predicting the data. Initially the data is divided into training and testing data set. The training data is then considered to build a regression model.The estimation of accuracy of this models is done by calculating the RMSE and the R square values. Finally the test data is used to predict the taxi fare prices from the generated model.

We plot a heat map which gives correlation coefficients so we can select features through this.

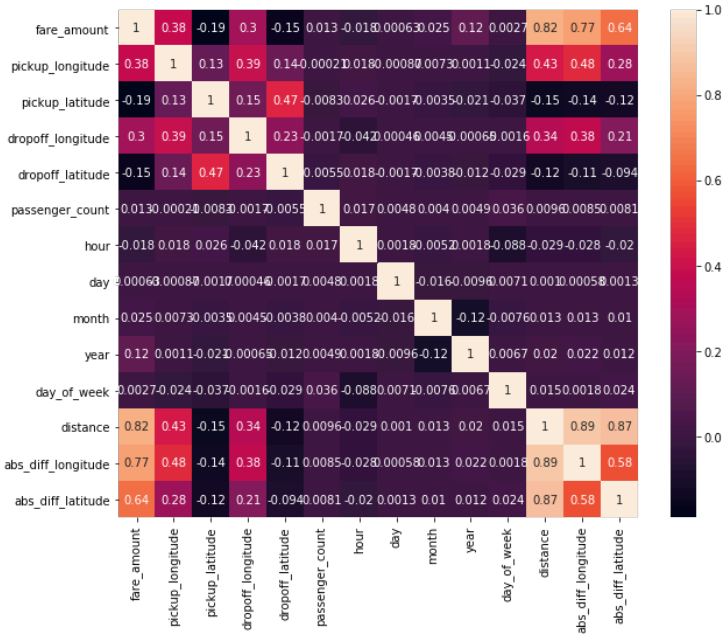

Fig 5. Heat map of the coefficients

4.1 Linear Regression

As a basic prediction duration of the ride and fare's values mean value is taken to determine the prediction set. Rather than selecting variables that don't have much effect on the result, forward selection is used to identify variable that select covariates that would be best. For simplicity smallest model was selected for liner regression.

In feature selection linear regression of all variables present at pickup time was used for prediction. Covariates alone cannot effect the non linear traffic, higher order and interaction was considered for the model to fit the data more precisely. But after plotting it, it is seen that no strong correlation exist, so they are excluded.

But higher oder terms make logical sense to be included as non linearities arise from traffic and can

affect for longer distance trips so squared trip distance to the model would increase the trip distance's importance in the duration and price prediction.

We calculated the r2 error and RMSE value. They are shown below.

```
In [79]: r2_score(Y_test , reg_predict)
Out[79]: 0.7278574157491385
```

Fig 6. R2 score

```
In [80]: from sklearn.metrics import mean_squared_error
         rmse = np.sqrt(mean_squared_error(Y_test , reg_predict))
         print("Root Mean Squared Error:",rmse)

         Root Mean Squared Error: 5.023003572657178
```

Fig 7. RMSE value for liner regression model

4.2 K-Nearest Neighbour Regression

K nearest neighbours is a simple algorithm that stores all available cases and predict the numerical target based on a similarity measure (e.g., distance functions). *k*-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. The *k*-NN algorithm is among the simplest of all machine learning algorithms.

```
In [90]: r2_score(Y_test , knr_predict)
Out[90]: 0.7784185124805778
```

Fig 8. R2 score for K- nearest regression

The neighbours are taken from a set of objects for which the class (for *k*-NN classification) or the object property value (for *k*-NN regression) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required.A peculiarity of the *k*-NN algorithm is that it is sensitive to the local structure of the data.

```
In [89]: rmse_knr = np.sqrt(mean_squared_error(Y_test , knr_predict))
         print("Root Mean Squared Error:",rmse_knr)

         Root Mean Squared Error: 4.532439039003952
```

Fig 9. RMSE value for k nearest regression

4.3 Random forest regression

The location of the trip also affects the fare because some places are more densely populated than others. The linear models don't consider this effect. Many decision trees are build to reduce the high covariance. Each tree deals with a set of covariances so that it is not clustered into one. The R2 and RMSE value can be see below.

```
In [108]: rmse_random_forest = np.sqrt(mean_squared_error(Y_test , random_fr_predict))
          print("Root Mean Squared Error:",rmse_random_forest)

          Root Mean Squared Error: 3.8588669372248727
```

Fig 10. RMSE value for random forest regression

```
In [96]: r2_score(Y_test , random_fr_predict)
Out[96]: 0.851488225936209
```

Fig 11. R2 value for random forest regression

4.4 Gradient boosting regression

A machine learning technique for classification and regression, which typically is a decision tree used to produce the ensemble of the weak prediction models. Boosting is used to build the model in a stage wise fashion and generalises the data by allowing optimisation. We calculated the R2 and RMSE values for gradient boosting regression.

```
In [103]: r2_score(Y_test , grad_predict)
Out[103]: 0.8444108301673305
```

Fig 12. R2 error value for gradient boosting

```
In [102]: from sklearn.metrics import mean_squared_error
          rmse = np.sqrt(mean_squared_error(Y_test , grad_predict))
          print("Root Mean Squared Error:",rmse)

          Root Mean Squared Error: 3.7979996168317043
```

Fig 13. RMSE value for gradient boosting

5 Result

The following table provides the evaluation of different regression models. We used testing dataset for all our models and the results summarised can be seen below.

Table 1. Results of the regression models

Type of Regression	R2 error value	RMSE value
Linear regression	0.727	5.02
K- nearest neghor regression	0.77	4.52
Random forest regression	0.85	3.58
Gradient boosting	0.844	3.79

As the results show Random forest and gradient boosting regressions yielded best results with the least errors. The predicted values we got using gradient boosting is shown in the below picture.

	key	fare_amount
9894	2013-09-25 22:00:00.000000153	11.461989
9895	2013-09-25 22:00:00.000000241	25.300802
9896	2013-09-25 22:00:00.000000127	11.299599
9897	2015-02-20 11:08:29.0000001	16.057318
9898	2015-01-12 15:36:37.0000002	5.395552
9899	2015-06-07 00:38:14.0000002	17.967066
9900	2015-04-12 21:56:22.0000005	8.539272
9901	2015-04-10 11:56:54.0000004	7.709704
9902	2015-06-25 01:01:46.0000002	15.635910
9903	2015-05-29 10:02:42.0000001	9.689080
9904	2015-06-30 20:03:50.0000002	48.656269
9905	2015-02-27 19:36:02.0000006	25.265407
9906	2015-06-15 01:00:06.0000002	5.361575
9907	2015-02-03 09:00:58.0000001	37.338546
9908	2015-05-19 13:58:11.0000001	8.003290
9909	2015-05-10 12:37:51.0000002	9.176932
9910	2015-01-12 17:05:51.0000001	10.991877
9911	2015-04-19 20:44:15.0000001	54.236879
9912	2015-01-31 01:05:19.0000005	23.164294
9913	2015-01-18 14:06:23.0000006	6.428535

Fig 14. Predicted fare values obtained from gradient boosting regression

6 Individual contribution

Siddharth Chittoor - Data Collection, data cleaning and distance calculation
Saketh Nimmala – Correlation analysis and linear regression

Manasa Alla – Random Forest, K nearest neighbour and gradient boosting

7 Conclusion and Future Scope

The data-driven approach for taxi destination and trip time prediction has been proposed based on trip matching. The experimental results shows that our models exhibit good performance for Gradient booster when compared to linear regression, random forest and K-nearest Neighbour.

For future work, The current approach will be generalised to automatically adapt to contexts and select a particular set of features that would improve the performance of prediction.

8 References

[1] Vanajakshi, L., S. C. Subramanian, and R. Sivanandan. ”Travel time prediction under hetero- geneous traffic conditions using global positioning system data from buses.” IET intelligent transport systems 3.1 (2009): 1-9.

[2] Biagioni, James, et al. ”Easytracker: auto- matic transit tracking, mapping, and arrival time prediction using smartphones.” Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems. ACM, 2011.

[3] Yildirimoglu, Mehmet, and Nikolas Geroli- minis. ”Experienced travel time prediction for congested freeways.” Transportation Research Part B: Methodological 53 (2013): 45-63.

[4] Wu, Chun-Hsin, Jan-Ming Ho, and Der-Tsai Lee. ”Travel-time prediction with support vector regression.” IEEE transactions on intelligent trans- portation systems.

4 (2004): 276-281. [5] Van Lint, J. W. C., S. P. Hoogendoorn, and Henk J. van Zuylen. ”Accurate freeway travel time prediction with state-space neu- ral networks under missing data.” Transportation Re- search Part C: Emerging Technologies 13.5 (2005): 347-369.

[6]. “Your Home for Data Science.” *Kaggle*, www.kaggle.com/c/new-york-city-taxi-fare-prediction/download/test.csv.

[7]. “Your Home for Data Science.” *Kaggle*, www.kaggle.com/c/new-york-city-taxi-fare-prediction/download/train.csv.