

Project Report
on
Taxi Demand Prediction In New York City
Submitted as partial fulfillment for the award
BACHELOR OF TECHNOLOGY
DEGREE
Session 2019-20
in
Computer Science and Engineering

By
Ankush Kumar Singh
1603210036
Baibhav Tripathi
1603210054
Chitranshu Gupta
1603210056

Under the guidance of
Mrs. Rishu Gupta

ABES ENGINEERING COLLEGE, GHAZIABAD



AFFILIATED TO
DR. A.P.J. ABDUL KALAM TECHNICAL UNIVERSITY, U.P., LUCKNOW
(Formerly UPTU)

Project Report
on
Taxi Demand Prediction In New York City
 Submitted as partial fulfillment for the award
BACHELOR OF TECHNOLOGY
DEGREE
 Session 2019-20
 in
Computer Science and Engineering

By
Ankush Kumar Singh
1603210036
Baibhav Tripathi
1603210054
Chitranshu Gupta
1603210056

Under the guidance of
Mrs. Rishu Gupta

ABES ENGINEERING COLLEGE, GHAZIABAD



Estd. 2000



AFFILIATED TO
DR. A.P.J. ABDUL KALAM TECHNICAL UNIVERSITY, U.P., LUCKNOW
(Formerly UPTU)

STUDENT'S DECLARATION

We hereby declare that the work being presented in this report entitled "Taxi Demand Prediction In New York City" is an authentic record of our own work carried out under the supervision of Mrs. Rishu Gupta.

The matter embodied in this report has not been submitted by us for the award of any other degree.

Dated:

Signature of students(s)

Ankush Kumar Singh, Baibhav Tripathi, Chitranshu Gupta
(Computer Science and Engineering)

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Signature of HOD

(Prof. (Dr.) Pankaj Kumar Sharma)
(Computer Science & Engineering
Department)

Date.....

Signature of Supervisor

Name- Mrs. Rishu Gupta
Assistant professor
(Computer Science & Engineering
Department)

CERTIFICATE

This is to certify that Project Report entitled “Taxi Demand Prediction In New York City” which is submitted by Ankush Kumar Singh, Baibhav Tripathi, Chitranshu Gupta in partial fulfillment of the requirement for the award of degree B. Tech. in Department of Computer Science and Engineering of Dr. A.P.J. Abdul Kalam Technical University, formerly Uttar Pradesh Technical University is a record of the candidate own work carried out by him/them under my supervision. The matter embodied in this thesis is original and has not been submitted for the award of any other degree.

Supervisor: Mrs. Rishu Gupta

Date

ACKNOWLEDGEMENT

It gives us a great sense of pleasure to present the report of the B. Tech Project undertaken during B. Tech. Final Year. We owe special debt of gratitude to our guide Mrs. Rishu Gupta, Department of Computer Science & Engineering, ABESSEC Ghaziabad for his constant support and guidance throughout the course of our work. Her sincerity, thoroughness and perseverance have been a constant source of inspiration for us. It is only his cognizant efforts that our endeavors have seen light of the day.

We also take the opportunity to acknowledge the contribution of Prof. (Dr.) Pankaj Kumar Sharma, Head, Department of Computer Science & Engineering, ABESSEC Ghaziabad for his full support and assistance during the development of the project.

We also do not like to miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind assistance and cooperation during the development of our project. Last but not the least, we acknowledge our friends for their contribution in the completion of the project.

Signature:

Name: Ankush Kumar Singh

Roll No.: 1603210036

Date:

Signature:

Name: Baibhav Tripathi

Roll No.: 1603210054

Date:

Signature:

Name: Chitranshu Gupta

Roll No.: 1603210056

Date:

ABSTRACT

There are many solutions available for the riders and drivers connectivity and ways to predict the Taxi/Cabs demands. But they lack the view from the drivers front and results into sometime oversupply or lack in Taxi.

In this paper we develop a methodology for analyzing transportation data at different levels of temporal and geographic granularity, the taxis and its demands are crucial because millennials favour taxi on demand than ownership.

We used the NYC-TLC Trip Record Dataset, made publicly available by the NYC Taxi & Limousine Commission.

This data is represented by a set of trajectories, annotated with time and with additional information such as passenger count and cost. We analyze TLC data to identify and locate hotspots, which points to lack of convenient public transportation options, and popular routes, which motivate ride-sharing solutions or addition of a bus route.

Our methodology is based on using a collected data from New York City's taxi (both yellow and green) which includes trip records and also has fields capturing pick-up and drop-off dates, drop-off times, pick-up and drop-off locations, distances, fares, rate types, payment and payment types, and driver-reported passenger counts.

We have used the machine learning models to train and test on these data provided in the dataset of NYC-TLC. These models will help in developing a solution for taxi demands and their supplies as it's based on the city hotspot and hence It's obvious to be more accurate for predicting demands at some particular location and time interval throughout the day and in certain weather conditions.

It benefits both the riders and drivers, by facilitating the riders with more accessible taxi services and drivers with more rides.

TABLE OF CONTENTS

I.	STUDENT DECLARATION	3
II.	CERTIFICATE	4
III.	ACKNOWLEDGEMENTS	5
IV.	ABSTRACT	6
V.	LIST OF FIGURES	8
CHAPTER 1: INTRODUCTION		9-10
1.1.	PROJECT OBJECTIVE	9
1.2.	PROJECT DEFINITION	10
CHAPTER 2 : LITERATURE SURVEY		11
CHAPTER 3 : SYSTEM DESIGN AND METHODOLOGY		11
CHAPTER 4 : PROPOSED SYSTEM		12-14
4.1.	SYSTEM DESCRIPTION	12
4.2.	DESIGNING PROCEDURE	13-14
CHAPTER 5 : IMPLEMENTATION AND RESULTS		15-21
5.1.	SOFTWARE AND HARDWARE REQUIREMENTS	15
5.2.	IMPLEMENTATION DETAILS	16-20
5.2.	MODEL DETAILS	21-23
5.3.	MODEL ANALYSIS	24
CHAPTER 6 : CONCLUSIONS		25
6.1.	CONCLUSION	
APPENDIX		26-31
	• RESEARCH PAPER	26-29
	• PROOF OF SUBMISSION	30
REFERENCES		31

LIST OF FIGURES

Figure 1	System Flowchart
Figure 2	Data Flow Diagram
Figure 3	Process Diagram
Figure 4	Plotting the cluster centers
Figure 5	Result

CHAPTER 1

INTRODUCTION

We can't imagine New York City, without its iconic yellow taxis and there are over 120000 vehicles TLC licensed [1] with about two hundred million taxi rides each year in the City [2].

The knowledge of taxi supply and their demand could be helpful to operate taxis in the city efficiently. In New York City, yellow taxis are everywhere and people use these taxis more frequently than in other cities of USA.

These taxis pick up their passengers on the city roads, they are not booked using any mobile app or prior to the ride.[3]

The ability to predict taxi ridership at a given location and time in the city could present valuable insights for the taxi dispatchers and planners.[2]

And in providing the solution with problems, such as, where the taxis are most needed, what is the demand of taxis at some location and their dispatch, and how the demand changes with time and location.

So, this project is about the prediction of demand for taxis, that is the number of taxi pickups given the time and a location within New York City.

Project Motivation is to avail the facility to easily and accurately predict the requirement and to fulfill the needs in all conditions and adopt to the changes in demand and supply.

1.1. Project Objective

The Objective of the project is to predict the taxi ridership at a given location and time in the city could present valuable insights for the taxi dispatchers and planners.

1.2. Project Definition

The problem that this project is aim at is to facilitate the drivers, riders and organizers with a solution to effectively and economically use the available

resources of the city and provide a hassle-free transport experience to the riders and an effective business tool for the drivers.

CHAPTER 2

LITERATURE SURVEY

There had been many implementation of this concept but most of them lack the ability to locate the riders due to human errors and lack of knowledge of the attributes which affects the riders at any location and for some particular time interval. The closest and most accurate implementation is in Japan, there the mobile service provider had implemented this model for the drivers to get rides. NTT DOCOMO, a Japanese company, start using demand forecasting for taxi companies and local taxi drivers in February 2018. They collect real-time people density with the help of smart-phones and apply machine learning, data science, and advance deep learning models to predict the possible numbers of taxi riders in every block with the time period of every 30 minutes, with high accuracy of 93-95%.

CHAPTER 3

SYSTEM DESIGN AND METHODOLOGY

We are divided our dataset into three parts –

1. Training set -This set of data is around 60% of the data of our dataset. This dataset is used for training our models.
2. Cross-validation set- This set of data is around 20% of the data of our dataset. This data is used in hyper-parameter tuning.
3. Testing set- This set of data is around 20% of the data of our dataset. This data is used in order to determine the accuracy of the models. We are using Root Mean Square Error (RMSE) to evaluate our prediction. Root Mean Square Error provide consistency and good penalizes predictions with a high deviation. Any big mistake in gauging taxi demand for a particular block could be costly imagine sending 12 taxis to a block that only requires 9 taxis etc. RMSE prevent such big miss allocations.

CHAPTER 4

PROPOSED SYSTEM

4.1 System Description

4.1.1 System Flowchart

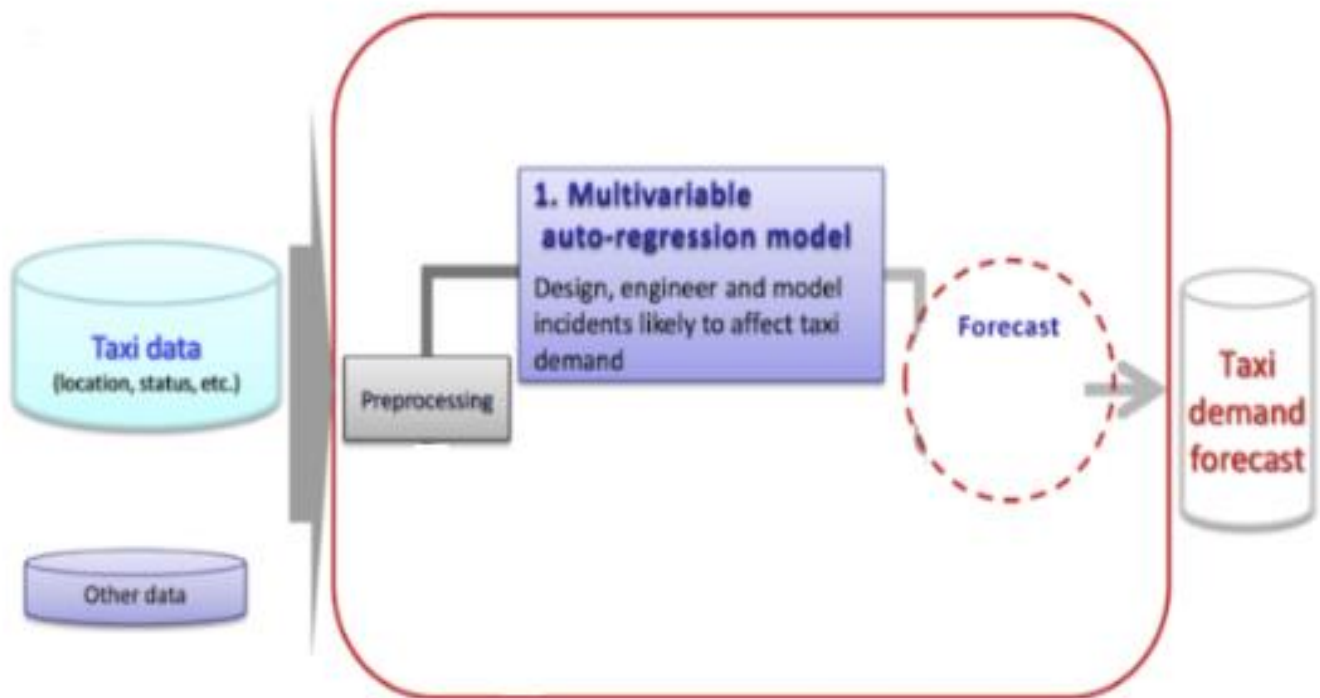


Figure1

DESIGNING PROCEDURE

4.2.1 Data Flow Diagram

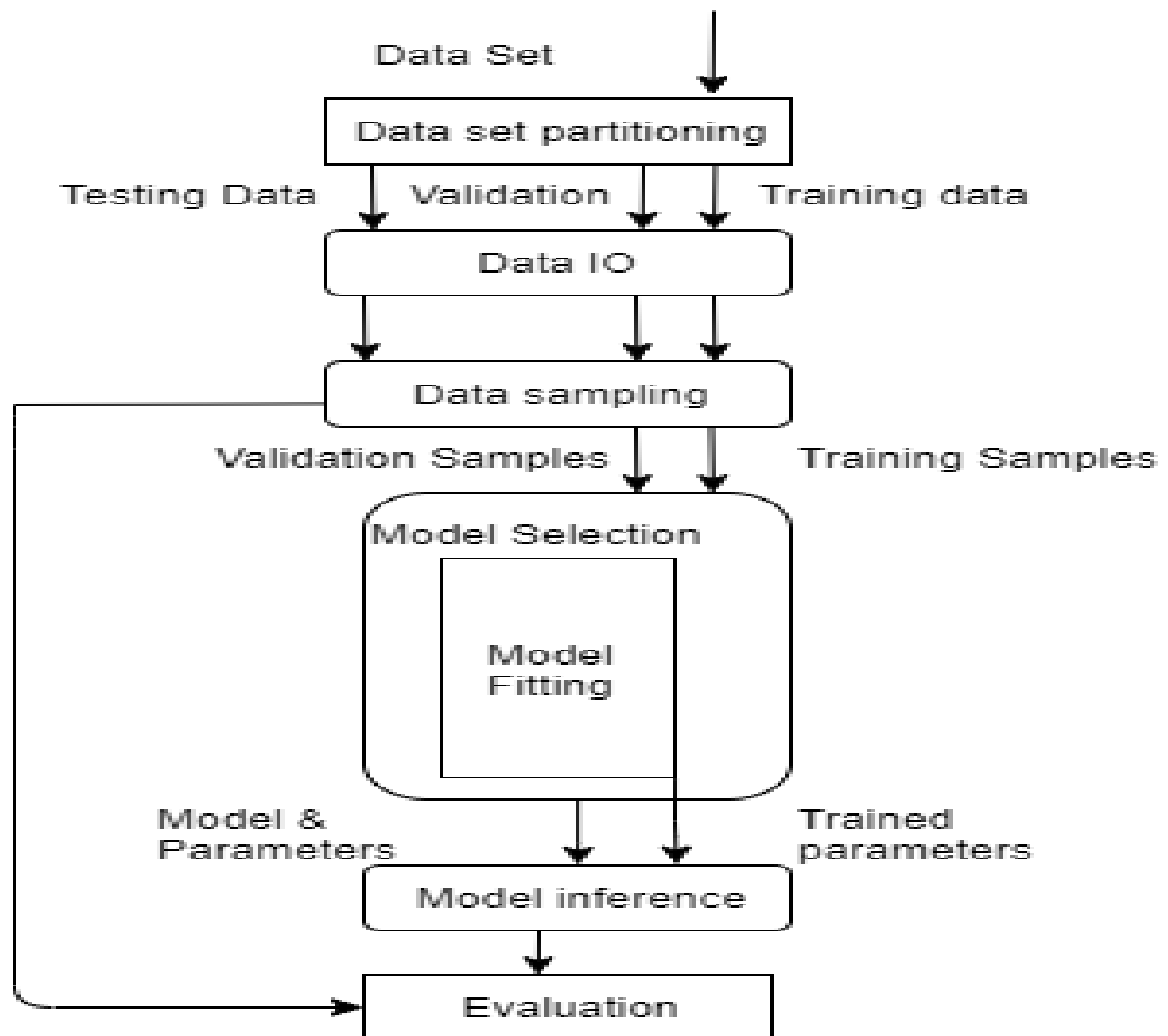


Figure 2

4.2.2 Process Diagram

By filtering the zip code within New York City, our data set has 71,000 observations.

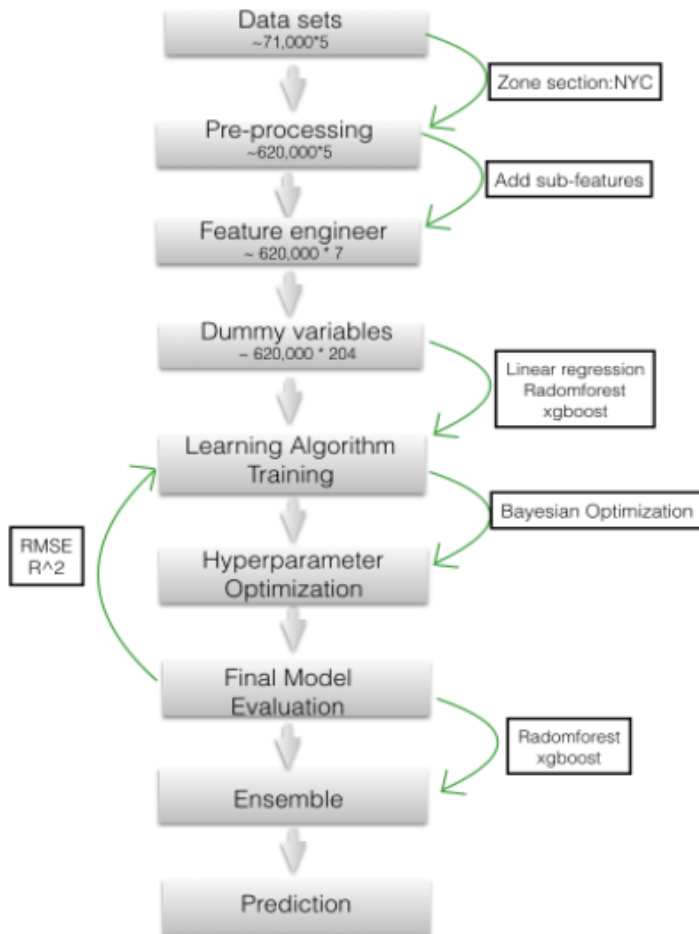


Figure 3

CHAPTER 5

IMPLEMENTATION AND RESULTS

5.1. Software and Hardware Requirements

Implementation Details

1. Python3 – code
2. Machine Learning
3. Data Analysis

System Specifications

1. CPU – Intel Core i7/i5 3.60 Ghz
2. RAM – 16 Gb GPU – Nvidia/AMD
3. Google Cloud

5.2 Implementation Details

1.Data Collection : We Have collected all yellow taxi trips data from jan-2015

Features in the dataset:

Field Name	Description
VendorID	A code indicating the TPEP provider that provided the record. 1. Creative Mobile Technologies 2. VeriFone Inc.
tpep_pickup_datetime	The date and time when the meter was engaged.
tpep_dropoff_datetime	The date and time when the meter was disengaged.
Passenger_count	The number of passengers in the vehicle. This is a driver-entered value.
Trip_distance	The elapsed trip distance in miles reported by the taximeter.
Pickup_longitude	Longitude where the meter was engaged.
Pickup_latitude	Latitude where the meter was engaged.
RateCodeID	The final rate code in effect at the end of the trip. 1. Standard rate 2. JFK 3. Newark 4. Nassau or Westchester 5. Negotiated fare 6. Group ride
Store_and_fwd_flag	This flag indicates whether the trip record was held in vehicle memory before sending to the vendor, aka "store and forward," because the vehicle did not have a connection to the server. Y= store and forward trip N= not a store and forward trip
Dropoff_longitude	Longitude where the meter was disengaged.
Dropoff_latitude	Latitude where the meter was disengaged.
Payment_type	A numeric code signifying how the passenger paid for the trip. 1. Credit card

	2. Cash 3. No charge 4. Dispute 5. Unknown 6. Voided trip
Fare_amount	The time-and-distance fare calculated by the meter.
Extra	Miscellaneous extras and surcharges. Currently, this only includes the 0.50 and 1 rush hour and overnight charges.
MTA_tax	0.50 MTA tax that is automatically triggered based on the metered rate in use.
Improvement_surcharge	0.30 improvement surcharge assessed trips at the flag drop. the improvement surcharge began being levied in 2015.
Tip_amount	Tip amount – This field is automatically populated for credit card tips. Cash tips are not included.
Tolls_amount	Total amount of all tolls paid in trip.
Total_amount	The total amount charged to passengers. Does not include cash tips.

Data Cleaning

In this section we will be doing univariate analysis and removing outlier/illegitimate values which may be caused due to some error

1. Pickup Latitude and Pickup Longitude

It is inferred from the source <https://www.flickr.com/places/info/2459115> that New York is bounded by the location coordinates(lat,long) - (40.5774, -74.15) & (40.9176,-73.7004) so hence any coordinates not within these coordinates are not considered by us as we are only concerned with pickups which originate within New York.

2. Dropoff Latitude & Dropoff Longitude

It is inferred from the source <https://www.flickr.com/places/info/2459115> that New York is bounded by the location coordinates(lat,long) - (40.5774, -74.15) & (40.9176,-73.7004) so hence any coordinates not within these coordinates are not considered by us as we are only concerned with dropoffs which are within New York.

Remove all outliers/erronous points.

```
#removing all outliers based on our univariate analysis above
def remove_outliers(new_frame):

    a = new_frame.shape[0]
    print ("Number of pickup records = ",a)
    temp_frame = new_frame[((new_frame.dropoff_longitude >= -74.15) & (
new_frame.dropoff_longitude <= -73.7004) & \
                            (new_frame.dropoff_latitude >= 40.5774) & (new_f
rame.dropoff_latitude <= 40.9176)) & \
                            ((new_frame.pickup_longitude >= -74.15) & (new_f
rame.pickup_latitude >= 40.5774)& \
                            (new_frame.pickup_longitude <= -73.7004) & (new_
frame.pickup_latitude <= 40.9176)))]
    b = temp_frame.shape[0]
    print ("Number of outlier coordinates lying outside NY boundaries:"
,(a-b))

    temp_frame = new_frame[(new_frame.trip_times > 0) & (new_frame.trip
_times < 720)]
    c = temp_frame.shape[0]
    print ("Number of outliers from trip times analysis:",(a-c))

    temp_frame = new_frame[(new_frame.trip_distance > 0) & (new_frame.t
rip_distance < 23)]
    d = temp_frame.shape[0]
    print ("Number of outliers from trip distance analysis:",(a-d))

    temp_frame = new_frame[(new_frame.Speed <= 65) & (new_frame.Speed >
```

```

= 0)]
    e = temp_frame.shape[0]
    print ("Number of outliers from speed analysis:",(a-e))

    temp_frame = new_frame[(new_frame.total_amount <1000) & (new_frame.
total_amount >0)]
    f = temp_frame.shape[0]
    print ("Number of outliers from fare analysis:",(a-f))

    new_frame = new_frame[((new_frame.dropoff_longitude >= -74.15) & (n
ew_frame.dropoff_longitude <= -73.7004) &\
                        (new_frame.dropoff_latitude >= 40.5774) & (new_f
rame.dropoff_latitude <= 40.9176)) & \
                        ((new_frame.pickup_longitude >= -74.15) & (new_f
rame.pickup_latitude >= 40.5774)& \
                        (new_frame.pickup_longitude <= -73.7004) & (new_
frame.pickup_latitude <= 40.9176)))]

    new_frame = new_frame[(new_frame.trip_times > 0) & (new_frame.trip_
times < 720)]
    new_frame = new_frame[(new_frame.trip_distance > 0) & (new_frame.tr
ip_distance < 23)]
    new_frame = new_frame[(new_frame.Speed < 45.31) & (new_frame.Speed
> 0)]
    new_frame = new_frame[(new_frame.total_amount <1000) & (new_frame.t
otal_amount >0)]

    print ("Total outliers removed",a - new_frame.shape[0])
    print ("---")
    return new_frame

```


Plotting the cluster centers:



Figure 4

5.3. Model Details

Regression Models

Train-Test Split

Before we start predictions using the tree based regression models we take 3 months of 2016 pickup data and split it such that for every region we have 70% data in train and 30% in test, ordered date-wise for every region

1 . Multiple Linear Regression.

```
# fit(X, y[, sample_weight])  Fit linear model.
# get_params([deep])  Get parameters for this estimator.
# predict(X)  Predict using the linear model
# score(X, y[, sample_weight])  Returns the coefficient of determination R^2 of the prediction.
# set_params(**params)  Set the parameters of this estimator

from sklearn.linear_model import LinearRegression
lr_reg=LinearRegression().fit(df_train, tsne_train_output)
y_pred = lr_reg.predict(df_test)
lr_test_predictions = [round(value) for value in y_pred]
y_pred = lr_reg.predict(df_train)
lr_train_predictions = [round(value) for value in y_pred]
```

2. Random Forest Regressor.

```
# Training a hyper-parameter tuned random forest regressor on our train data
# find more about LinearRegression function here http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html
# -----
# default paramters
# sklearn.ensemble.RandomForestRegressor(n_estimators=10, criterion='mse', max_depth=None, min_samples_split=2,
# min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0,
# min_impurity_split=None, bootstrap=True, oob_score=False, n_jobs=1, random_state=None, verbose=0, warm_start=False)

# some of methods of RandomForestRegressor()
# apply(X) Apply trees in the forest to X, return leaf indices.
# decision_path(X) Return the decision path in the forest
# fit(X, y[, sample_weight]) Build a forest of trees from the training set (X, y).
# get_params([deep]) Get parameters for this estimator.
# predict(X) Predict regression target for X.
# score(X, y[, sample_weight]) Returns the coefficient of determination R^2 of the prediction.

regl = RandomForestRegressor(max_features='sqrt',min_samples_leaf=4,min_samples_split=3,n_estimators=40, n_jobs=-1)
regl.fit(df_train, tsne_train_output)
# Predicting on test data using our trained random forest model

# the models regl is already hyper parameter tuned
# the parameters that we got above are found using grid search

y_pred = regl.predict(df_test)
rndf_test_predictions = [round(value) for value in y_pred]
y_pred = regl.predict(df_train)
rndf_train_predictions = [round(value) for value in y_pred]
```

3.XGBoost Regressor.

```
# Training a hyper-parameter tuned Xg-Boost regressor on our train data

# find more about XGBRegressor function here http://xgboost.readthedocs.io/en/latest/python/python\_api.html?#module-xgboost.sklearn
# -----
# default paramters
# xgboost.XGBRegressor(max_depth=3, learning_rate=0.1, n_estimators=100, silent=
True, objective='reg:linear',
# booster='gbtree', n_jobs=1, nthread=None, gamma=0, min_child_weight=1, max_del
ta_step=0, subsample=1, colsample_bytree=1,
# colsample_bylevel=1, reg_alpha=0, reg_lambda=1, scale_pos_weight=1, base_score
=0.5, random_state=0, seed=None,
# missing=None, **kwargs)

# some of methods of RandomForestRegressor()
# fit(X, y, sample_weight=None, eval_set=None, eval_metric=None, early_stopping_
rounds=None, verbose=True, xgb_model=None)
# get_params([deep]) Get parameters for this estimator.
# predict(data, output_margin=False, ntree_limit=0) : Predict with data. NOTE: T
his function is not thread safe.
# get_score(importance_type='weight') -> get the feature importance
x_model = xgb.XGBRegressor(
    learning_rate =0.1,
    n_estimators=1000,
    max_depth=3,
    min_child_weight=3,
    gamma=0,
    subsample=0.8,
    reg_alpha=200, reg_lambda=200,
    colsample_bytree=0.8,nthread=4)
x_model.fit(df_train, tsne_train_output)
#predicting with our trained Xg-Boost regressor
# the models x_model is already hyper parameter tuned
# the parameters that we got above are found using grid search

y_pred = x_model.predict(df_test)
xgb_test_predictions = [round(value) for value in y_pred]
y_pred = x_model.predict(df_train)
xgb_train_predictions = [round(value) for value in y_pred]
```

Model Analysis

Result:

```
Error Metric Matrix (Tree Based Regression Methods) - MAPE
-----
Baseline Model -                               Train: 0.1521991930346337
2      Test: 0.14527264149803146
Exponential Averages Forecasting -             Train: 0.1445324228877661
3      Test: 0.13742496628342404
Linear Regression -                           Train: 0.12331289993554674
      Test: 0.11103976486001399
Random Forest Regression -                   Train: 0.1372666441506003
3      Test: 0.12730480669750005
XgBoost Regression -                       Train: 0.1074881778262877
8      Test: 0.10221297417459224
-----
-----
```

Figure 5

F

CHAPTER 5

Conclusions :

1. XgBoost has the least MAPE of all the models built.
2. Linear Regression despite being simpler than RandomForest performed better than it but not by much.
3. Holts Winter Features with vales of alpha, beta and gamma - (0.2, 0.15, 0.2) helped to reduce the MAPE below 12 %.

Appendix

Research Paper

1

Taxi Demand Prediction In New York City

Baibhav Tripathi¹, Chitranshu Gupta², Ankush Kumar Singh³, Rishu Gupta⁴

^{1,2,3,4}Computer Science & Engineering Department
^{1,2,3,4}ABES Engineering College, Ghaziabad, Uttar Pradesh – 221009, India

¹baibhav.16bcs1106@abes.ac.in,

²chitranshu.16bcs1121@abes.ac.in,

³ankush.16bcs1049@abes.ac.in,

⁴rishu.gupta@abes.ac.in

Abstract— There are many solutions available for the riders and drivers connectivity and ways to predict the Taxi/Cabs demands. But they lack the view from the drivers front and results into sometime oversupply or lack in Taxi. In this paper we develop a methodology for analyzing transportation data at different levels of temporal and geographic granularity, the taxis and its demands are crucial because millennials favour taxi on demand than ownership. We used the NYC-TLC Trip Record Dataset, made publicly available by the NYC Taxi & Limousine Commission[1]. This data is represented by a set of trajectories, annotated with time and with additional information such as passenger count and cost.[2] We analyze TLC data to identify and locate hotspots, which points to lack of convenient public transportation options, and popular routes, which motivate ride-sharing solutions or addition of a bus route. Our methodology is based on using a collected data from New York City's taxi (both yellow and green) which includes trip records and also has fields capturing pick-up and drop-off dates, drop-off times, pick-up and drop-off locations, distances, fares, rate types, payment and payment types, and driver-reported passenger counts.

We have used the machine learning models to train and test on these data provided in the dataset of NYC-TLC. These models will help in developing a solution for taxi demands and their supplies as it's based on the city hotspot and hence It's obvious to be more accurate for predicting demands at some particular location and time interval throughout the day and in certain weather conditions. It benefits both the riders and drivers, by facilitating the riders with more accessible taxi services and drivers with more rides..

I. INTRODUCTION

We can't imagine New York City, without its iconic yellow taxis and there are over 120000 vehicles TLC licensed[1] with about two hundred million taxi rides each year in the City[2]. The knowledge of taxi supply and their demand could be helpful to operate taxis in the city efficiently. In New York City, yellow taxis are everywhere and people use these taxis more frequently than in other cities of USA. These taxis pick up their passengers on the city roads, they are not booked using any mobile app or prior to the ride.[3] The ability to predict taxi ridership at a given location and time in the city could present valuable insights for the taxi dispatchers and planners.[2] And in providing the solution with problems, such as, where the taxis are most needed, what is the demand of taxis at some location and their dispatch, and how the demand changes with time and location. So, this project is about the prediction of demand for taxis, that is the number of taxi pickups given the time and a location within New York City.

II. SCOPE FOR THE SOLUTION

The problem can be understand in the following inputs and outputs:[2]

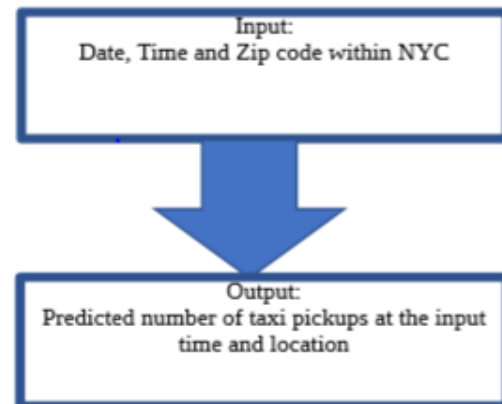


Fig 1

III. RELATED WORKS

There had been many implementation of this concept but most of them lack the ability to locate the riders due to human errors and lack of knowledge of the attributes which affects the riders at any location and for some particular time interval. The closest and most accurate implementation is in Japan, there the mobile service provider had implemented this model for the drivers to get rides.

NTT DOCOMO, a Japanese company, start using demand forecasting for taxi companies and local taxi

THEY COLLECT REAL-TIME PEOPLE DENSITY WITH THE HELP OF SMARTPHONES AND APPLY MACHINE LEARNING, DATA SCIENCE, AND ADVANCE DEEP LEARNING MODELS TO PREDICT THE POSSIBLE NUMBERS OF TAXI RIDERS IN EVERY BLOCK WITH THE TIME PERIOD OF EVERY 30 MINUTES, WITH HIGH ACCURACY OF 93-95%.[5]

In current, there are around 2,500 taxis in major Japanese cities that use this service.[5] Their system uses the following data as input for their models-

- Connection density statistics from their mobile network.
- Weather Data
- Taxi location and activity from each vehicle.

Based on statements from NTT DOCOMO[9] the accuracy of the model is increasing with the use of real world cases and the model is improving itself using the machine learning techniques. They are certain that within a short span of time model will be much accurate and reliable in the real world.

IV. PROPOSED SYSTEM

We are divided our dataset into three parts –

1. Training set -

This set of data is around 60% of the data of our dataset. This dataset is used for training our models.

2. Cross-validation set-

THIS SET OF DATA IS AROUND 20% OF THE DATA OF OUR DATASET. THIS DATA IS USED IN HYPER PARAMETER TUNING. [7]

3. Testing set-

This set of data is around 20% of the data of our dataset. This data is used in order to determine the accuracy of the models.

We are using Root Mean Square Error (RMSE) to evaluate our prediction. Root Mean Square Error provide consistency and good penalizes predictions with a high deviation. Any big mistake in gauging taxi demand for a particular block could be costly - imagine sending 12 taxis to a block that only requires 9 taxis etc. RMSE prevent such big miss allocations.[7]



Fig 2

Fig 3

Implementation and results

1. MODEL

Multiple-Linear Regression:-

The multiple-linear regression model allows us to exploit linear patterns in the data set. This model is an appealing first choice because feature weights are easily interpretable and because it runs efficiently on large datasets. The result is showed below.[10]

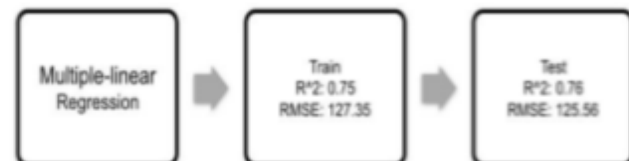


Fig 4

Ridge regression:-

To improve multiple-linear regression and introduce regularization, ridge model is applied and the hyper parameters alpha used for ridge model are determined using Bayesian Optimization[2] select parameter values.

observed to beat performance by human based on some good benchmark datasets.

Random Forest

The quality of the algorithm to preventing overfitting and being robust against outliers, the Random Forest Tree algorithm is the most preferred choice for the problem. Also, being a tree based regression model, it is capable of representing many complex decision boundaries, thus it based suited our chosen models.

And the hyperparameters of the algorithm are -- max features (number of splits at each tree) and n-estimators (number of trees) are determined using Bayesian Optimization algorithm and select the parameter values. [2]Of the values we swept, our model performed has the best performance with 14 as the max-features and 500 for n-estimators.[12]



Fig 5

Why did we use Bayesian Optimization instead of grid-search for hyperparameters tuning process?[2]

The uses a Gaussian Process *Bayesian Optimization* to model the surrogate, and typically optimizes the Expected Improvement, which is the expected probability that new trials will improve upon the current best observation. Gaussian Process is a distribution over functions.

A Gaussian process sample represents an entire function. Training the function involves fitting this distribution to the given data, such that it generates functions which is close to the observation. By the use of Gaussian process, we can calculate the Expected Improvement of any part that the user will feel that he is connecting with another point in the search space. Highest expected improved, function will be tried next.

XGBoost

XGBoost is an advanced gradient boosting framework based on decision-tree ensemble machine learning algorithm. It is a highly efficient, flexible and portable algorithm, and powerful enough to deal with all sorts of irregularities in the dataset. The algorithm is extremely flexible, it allow users to customize a wide range of hyper-parameters while training the mode, and at last to reach the optimal solution.[13]

For our model, we tuned some of the booster parameters which leads to the best performance. These parameters

gamma is minimum loss reduction for each split, min_child_weight is the minimum sum of weights of all divergence required at each split node, subsample is the fraction of observations randomly selected for each tree and colsample_bytree is the fraction of columns randomly sampled for each tree.[7]



Fig 6

Results are :-

Parameters	Range	Best (lowest MSE)
Max_depth	3 ~ 14	14
Learning_rate	0.01 ~ 0.2	0.1186
N_estimators	50, 1000	463
gamma	0.01 ~ 1.0	1.0
Min_child_weight	1 ~ 10	6.1929
Subsample	0.5 ~ 1	0.9675
Colsample_bytree	0.5 ~ 1	0.8544

Fig 7

Ensemble modelling

Ensemble modelling is the process of running multiple related but different analytical models and then strategically generate the results into a single score or spread in order to improve the accuracy of predictive analytics or in data mining applications. We combined our two most accurate models: random forest and xgboost for ensemble modelling and used linear regression.[14]

The results of all the models are compared in the below table, and ensemble modelling did not further improve the prediction as expected. Overall xgboost performs best.[2]

Model	R^2	RMSE
Multiple linear regression	0.76	125.56
Ridge	0.75	125.56
Randomforest	0.97	40.06
XGBoost	0.98	35.01
Ensemble	0.97	42.95

Fig 8

Model analysis

In order to visualize how well the models such as -- random forest, xgboost, ensemble performs, we plot the known versus predicted number of pickups for each data point in the test dataset represented in the Figure below. [2]

predictions lie close to the actual values. The data points straddle the unit slope line evenly, signifying that the models do not systematically underestimate or overestimate the number of taxi pickups. For three models, as expected, increase in absolute prediction error can be relate to increase in the actual number of pickups. This effect can be visualized in the form of a cone-shaped region extending outward from the origin within which the data points in dataset fall.

To better understand how absolute prediction error varies, dataset is divided into 3 subsets based on the actual number of pickups: subset 1 has the pickups greater and equal to 1000, subset 2 with pickups greater and equal to 100 and less than 1000, subset 3 with pickups less than 100. Using this table, we can conclude that the value of RMSE increases with increase in the actual number of pickups. The Xgboost algorithm performs best in 3 subsets.[2]

Subset	Pickups	Data Size	RFR RMSE	XGBR RMSE	Ensemble RMSE
Subset 1	≥ 1000	2479	150	123	156
Subset 2	100 ~ 999	24759	75	66	80
Subset 3	< 100	95472	8.7	7.2	9.5

Fig 10

The below figure represent the comparison between the prediction of all three models and the actual value, 10 values from each subset by randomly picking. At the each unique value, different models give different performance.

As we did the predictions for the upcoming week using three models and as the result is calculated, visualized, and compared using shiny interactive application.[2]

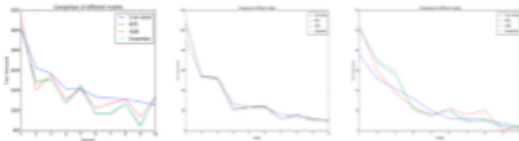


Fig 11

V. CONCLUSIONS


Overall, The XGBoost algorithm performs most accurate with R^2 value of 0.98 and RMSE 35.01 for our model. Our analysis for error and results supported our intuition and accurately supported our prediction and model usefulness. Using this model the taxi dispatchers and city planners can determine the location and timing according to the predicted ridership.

Future work includes the implementation of two more algorithms which are Neural network regression[2] due to its ability to select the useful features on its own, without any external tuning and the ability to learn and improve during the processes. Also, the implementation of K-mean clustering for finding non-obvious patterns in the dataset, using unsupervised learning we can create our clusters on

REFERENCES

- [1]. <https://www1.nyc.gov/site/tlc/about/data.page>
- [2]. <https://nycdatascience.com/blog/student-works/predict-new-york-city-taxi-demand/>
- [3]. https://en.wikipedia.org/wiki/Taxicabs_of_New_York_City
- [4]. http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml
- [5]. <http://www.ncdc.noaa.gov/cdo-web/>
- [6]. <https://github.com/fivethirtyeight/uber-tlc-foil-response>
- [7]. <https://www.appliedaicourse.com/course/11/Applied-Machine-learning-course>
- [8]. https://www.researchgate.net/publication/313451485_Predicting_taxi_demand_at_high_spatial_resolution_A_proaching_the_limit_of_predictability
- [9]. <https://cloud.google.com/blog/products/gcp/now-live-in-tokyo-using-tensorflow-to-predict-taxi-demand>
- [10]. https://scikit-learn.org/stable/modules/linear_model.html
- [11]. https://scikit-learn.org/stable/modules/linear_model.html#ridge-regression-and-classification
- [12]. <https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html#sklearn.ensemble.RandomForestRegressor>
- [13]. https://xgboost.readthedocs.io/en/latest/python/python_api.html#module-xgboost.sklearn
- [14]. <https://scikit-learn.org/stable/modules/ensemble.html#highlight=xgboost>

Proof of Submission :



International Journal of Engineering and Advanced Technology

ISSN: 2249-8958 (Online) | Exploring Innovation | A Key for Dedicated Services
 Published by: Blue Eyes Intelligence Engineering and Sciences Publication
 # G18-19-20, Block-B, Tirupati Abhinav Homes, Damkhada, Bhopal (Madhya Pradesh)-462037, India
 Website: www.ijeat.org Email: submit2@ijeat.org

Acceptance Letter

Dear Author(s): Baibhav Tripathi, Chitranshu Gupta, Ankush Kumar Singh, Rishu Gupta

Paper ID:	D6726049420
Paper Title:	Taxi Demand Prediction in New York City


This is to enlighten you that above manuscript appraised by the proficient and it is **accepted** by the Board of Referees (BoR) of 'Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)' for publication in the '**International Journal of Engineering and Advanced Technology (IJEAT)**' that will publish at **Volume-9 Issue-4, April 2020** in Regular Issue on **30 April 2020**. It will be available live at <http://www.ijeat.org/download/volume-9-issue-4/>

It is advised you to provide us **following supporting documents in a single email** before 11 April 2020 at submit2@ijeat.org


- Final Paper | Ms Word .doc | .docx file**
camera ready paper should be prepared as per journal template which is available at <http://www.ijeat.org/download/>
- Copyright Transfer Form | Submit Online only | do not send it through email**
<https://www.blueeyesintelligence.org/copyright/>
- Proof of Registration | Scanned | Online Received Email**
please visit in this URL. All details available at <https://www.blueeyesintelligence.org/registration/>

INFORMATION FOR AUTHOR(S)- Please read very carefully.











- Each author (s) profile of minimum of 100 words along with a photo should be available in the final paper. The final paper should be prepared as per the journal template. The Paper should have a minimum of 03 pages and a maximum of 10 pages. Maximum 05 authors can be seated in a paper. In the case of more than 05 authors, the paper (s) to be rejected. Final paper should not have more than 30% plagiarism including reference section.
- If the above three supporting documents (Final Paper, Copyright and Registration) does not submit to the journal by the author in the given date (s), then paper will automatically suspend from publication for particular volume/issue. During the final email, you have to attach Final Paper, Copyright and Proof of Registration in a single email. Final paper should be result oriented and should be prepared as per the reviewer (s) comment (s). In the case of failure, it to be suspended for correction. Please read review report carefully. It is compulsory to write the Paper ID of the paper in place of Subject Area in the email during the final paper submission. Header and footer of the paper template will be edited by journal staff only.
- Author (s) can make rectification/updation in the final paper but after the signing the copyright and final paper submission to the journal, any rectification/updation is not possible. Published paper to be available online from 30 April to 05 May 2020. Paper can not withdraw after submitting the copyright to the journal. Author(s) will receive publication certificate within 01 to 02 weeks after the date of publication of respective volume/issue
- All the scopus listed journals that were earlier in the scopus database (year 2018-2019) are under re-evaluation for year 2020 which result will come soon. Hence, paper (s) to be forwarded for inclusion in the scopus database after result of re-evaluation. The whole process for including any article (s) in the scopus database is done by scopus team only. Journal/ publication house does not have any involvement in the decision whether accept or reject a paper from the scopus database and can not influence the processing time of paper.
- The DOI can be checked and verified within 02 to 04 weeks after the date of publication of volume/issue: <https://www.doi.org/>
- This paper to be uploaded in the database of Web of Science (Publons), Elsevier (Mendeley), KUDOS and CrossRef within 04 to 06 weeks after publication.



Jitendra Kumar Sen
(Manager)



Dr. Shiv Kumar
(Editor-In-Chief)

References

- [1].<https://www1.nyc.gov/site/tlc/about/data.page>
- [2].<https://nycdatascience.com/blog/student-works/predict-new-york-city-taxi-demand/>
- [3].https://en.wikipedia.org/wiki/Taxicabs_of_New_York_City
- [4].http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml
- [5].<http://www.ncdc.noaa.gov/cdo-web/>
- [6].<https://github.com/fivethirtyeight/uber-tlc-foil-response>
- [7].<https://www.appliedaicourse.com/course/11/Applied-Machine-learning-course>
- [8].https://www.researchgate.net/publication/313451485_Predicting_taxi_demand_at_high_spatial_resolution_Approaching_the_limit_of_predictability
- [9]. <https://cloud.google.com/blog/products/gcp/now-live-in-tokyo-using-tensorflow-to-predict-taxi-demand>
- [10].https://scikit-learn.org/stable/modules/linear_model.html
- [11].https://scikit-learn.org/stable/modules/linear_model.html#ridge-regression-and-classification
- [12].<https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html?highlight=random%20forest#sklearn.ensemble.RandomForestRegressor>
- [13].https://xgboost.readthedocs.io/en/latest/python/python_api.html#module-xgboost.sklearn
- [14].<https://scikit-learn.org/stable/modules/ensemble.html?highlight=xgboost>