Cab Fare Prediction

- Project report

By Divakar Sunkara

Contents:

Chapter 1

Introduction

1.1 Problem Statement

The objective of this project is to predict Cab Fare amount. You are a cab rental start-up company. You have successfully run the pilot project and now want to launch your cab service across the country. You have collected the historical data from your pilot project and now have a requirement to apply analytics for fare prediction. You need to design a system that predicts the fare amount for a cab ride in the city.

1.2 Data

Attributes: ·
• pickup_datetime - timestamp value indicating when the cab ride started.
• pickup_longitude - float for longitude coordinate of where the cab ride started.
• pickup_latitude - float for latitude coordinate of where the cab ride started.
• dropoff_longitude - float for longitude coordinate of where the cab ride ended.
• dropoff_latitude - float for latitude coordinate of where the cab ride ended.
• passenger_count - an integer indicating the number of passengers in the cab ride.

Summary on the initial level of understanding on data:

1. We have both input variable and target variable. SO it is a supervised machine learning model.
2. We need to find the fare amount for the cab services which is a continuous variable. So it is a regression problem.
3. For regression problems we need to use regression supervised machine learning algorithms.
    1. KNN
    2. Linear regression
    3. Decision tree
    4. Ensemble methods.
4. For regression problems we should follow regression metrics
    1. RSME (Root mean square error)
    2. MSE (Mean square error)
    3. R square
    4. Adjusted R square.

Chapter 2
2. Pre-Processing

Data preprocessing is the first stage of any type of project. In this stage we get the feel of the data. We do this by looking at plots of independent variables vs target variables. If the data is messy, we try to improve it by sorting deleting extra rows and columns. This stage is called as Exploratory Data Analysis. This stage generally involves data cleaning, merging, sorting, looking for outlier analysis, looking for missing values in the data, imputing missing values if found by various methods such as mean, median, mode, KNN imputation, etc.
Further we will look into what Pre-Processing steps do this project was involved in.

Summary after the second level of understanding the data:

In the passenger count column we have many outliers and null values. As per the above results, We can consider the maximum number of people who can sit in a cab is 6 persons. So other than passenger count (1,2,3,4,5,6) remaining values we should clean.

P_Count	(Count of passenger_count)
1.00	18173
2.00	3796
5.00	1741
3.00	1123
4.00	535
6.00	479
0.00	57
53.00	2
43.00	2
554.00	1
0.12	1
531.20	1
456.00	1
354.00	1
55.00	1
557.00	1
5345.00	1
236.00	1
1.30	1
345.00	1
58.00	1

```
35.00       1
535.00      1
536.00      1
537.00      1
5334.00     1
87.00       1
Name: passenger_count, dtype: int64
```

Summary:

1. We have a lot of columns as numeric or float. So we don't have categorical variables in the data.
2. Preprocessing techniques vary for both numerical and categorical data.

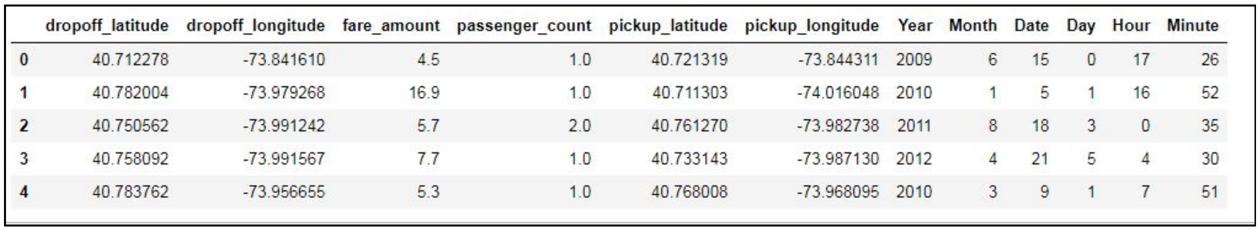
```
Categorical :  ['fare_amount', 'pickup_datetime']

Numerical :  []

float_data :  ['dropoff_latitude', 'dropoff_longitude', 'passenger_count', 'pickup_latitude',
'pickup_longitude']
```

Datatypes of the columns:

```
dropoff_latitude     float64

dropoff_longitude    float64

fare_amount          object

passenger_count      float64

pickup_datetime      object

pickup_latitude      float64

pickup_longitude     float64
dtype: object
```

Summary:

1. We have a date object in the dataframe, We cannot perform with the data object for training the machine learning model. So either we need to drop the variable or we need to perform feature engineering for that column by converting the date into separate columns like year, month, date,time etc..



	dropoff_latitude	dropoff_longitude	fare_amount	passenger_count	pickup_latitude	pickup_longitude	Year	Month	Date	Day	Hour	Minute
0	40.712278	-73.841610	4.5	1.0	40.721319	-73.844311	2009	6	15	0	17	26
1	40.782004	-73.979268	16.9	1.0	40.711303	-74.016048	2010	1	5	1	16	52
2	40.750562	-73.991242	5.7	2.0	40.761270	-73.982738	2011	8	18	3	0	35
3	40.758092	-73.991567	7.7	1.0	40.733143	-73.987130	2012	4	21	5	4	30
4	40.783762	-73.956655	5.3	1.0	40.768008	-73.968095	2010	3	9	1	7	51

Removing values which are not within desired range(outlier) depending upon basic understanding of dataset.

In this step we will remove values in each variable which are not within desired range and we will consider them as outliers depending upon basic understanding of all the variables. You would think why haven't made those values NA instead of removing them well I did make them NA but it turned out to be a lot of missing values(NA's) in the dataset. Missing values percentage becomes very much high and then there will be no point of using that imputed data. Take a look at the 3 scenarios. If everything beyond range is made nan also except latitudes and longitudes then:

So after performing this. We have many null values.

```
          Variables  Missing_percentage
0        fare_amount          38.257824
1     pickup_latitude           1.212518
2    pickup_longitude           1.212518
3   dropoff_longitude           1.208669
4    dropoff_latitude           1.200970
5     passenger_count           0.434967
6              Year           0.000000
7             Month           0.000000
8              Date           0.000000
9               Day           0.000000
10             Hour           0.000000
11           Minute           0.000000
```

2.1 Statistics:

We need to perform many statistical techniques like mean, median, mode, skew etcc.. on data to understand it better.

Mean of each column:

```
dropoff_latitude       40.223677

dropoff_longitude     -73.038957

passenger_count         2.260439

pickup_latitude        40.233812

pickup_longitude      -73.039649

Year                 2011.763193

Month                   6.488433

Date                   15.869510

Day                     2.964125

Hour                   13.486200

Minute                 29.603680
dtype: float64
```

From this statistical data we can get some business insights which help the company to understand the pain areas or other related information.

2.2 Missing value Analysis

In this step we look for missing values in the dataset like empty row column cell which was left after removing special characters and punctuation marks. Some missing values are in form of NA. missing values left behind after outlier analysis; missing values can be in any form. Unfortunately, in this dataset we have found some missing values. Therefore, we will do some missing value analysis. Before imputed we selected random row no. and made it NA, so that we will compare original value with imputed value and choose best method which will impute value closer to actual value.

If missing value percentage is very less (<30%) in the data, either we can remove those or can perform missing value imputation but the best option is to remove that data.

After removing the missing data which is less than 30% from the entire data.

```
Variables  Missing_percentage
0         fare_amount          38.904549
1    dropoff_latitude           0.000000
2   dropoff_longitude           0.000000
3     passenger_count           0.000000
4     pickup_latitude           0.000000
5    pickup_longitude           0.000000
6                Year           0.000000
7               Month           0.000000
8                Date           0.000000
9                 Day           0.000000
10               Hour           0.000000
11             Minute           0.000000
```

Fare amount columns alone missing value count greater than 30% . So we need to impute the missing values.

There are many missing value imputation techniques like mean, median, mode, KNN etc..

We need to try all the models on the sample of the data and confirm which can work better.

Framework

1. Create a small subset of data with complete observations
2. Delete some values manually
3. Use multiple methods to fill
4. See where they are failing
5. Choose the best method

After missing value imputation:

```
Missing values count :
 dropoff_latitude     0
dropoff_longitude    0
fare_amount          0
passenger_count      0
pickup_latitude      0
pickup_longitude     0
Year                 0
Month                0
Date                 0
Day                  0
Hour                 0
Minute               0
dtype: int64
```

Median imputation works perfectly well for the data. So we followed median imputation technique.

2.3 Outlier Analysis

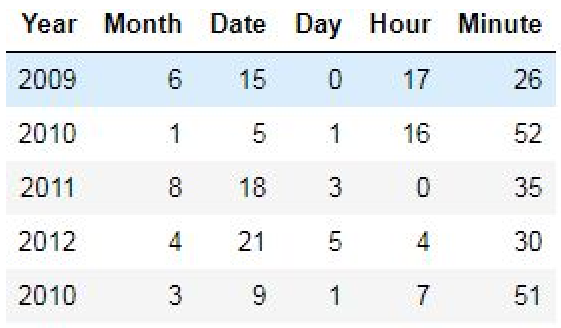
We look for outliers in the dataset by plotting Boxplots. There are outliers present in the data. we have removed these outliers. This is how we done,
I. We replaced them with Nan values or we can say created missing values.
II. Then we imputed those missing values with the median method.
• We Will do Outlier Analysis only on Fare_amount just for now and we will do outlier analysis after feature engineering latitudes and longitudes.
• Univariate Boxplots: Boxplots for target variable.

Univariate Boxplots: Boxplots for all Numerical Variables also for target variable
Bivariate Boxplots: Boxplots for all fare_amount Variables Vs all passenger_count variable.

We need to remove outliers from the data otherwise. Models will not give good results.

2.4 Feature Engineering

Feature Engineering is used to drive new features from existing features.
1. For 'pickup_datetime' variable: We will use this timestamp variable to create new variables. New features will be year, month, day_of_week, hour. 'year' will contain only years from pickup_datetime. For ex. 2009, 2010, 2011, etc. 'month' will contain only months from pickup_datetime. For ex. 1 for January, 2 for February, etc. 'day_of_week' will contain only a week from pickup_datetime. For ex. 1 which is for Monday,2 for Tuesday,etc. 'hour' will contain only hours from pickup_datetime. For ex. 1, 2, 3, etc.



Year	Month	Date	Day	Hour	Minute
2009	6	15	0	17	26
2010	1	5	1	16	52
2011	8	18	3	0	35
2012	4	21	5	4	30
2010	3	9	1	7	51

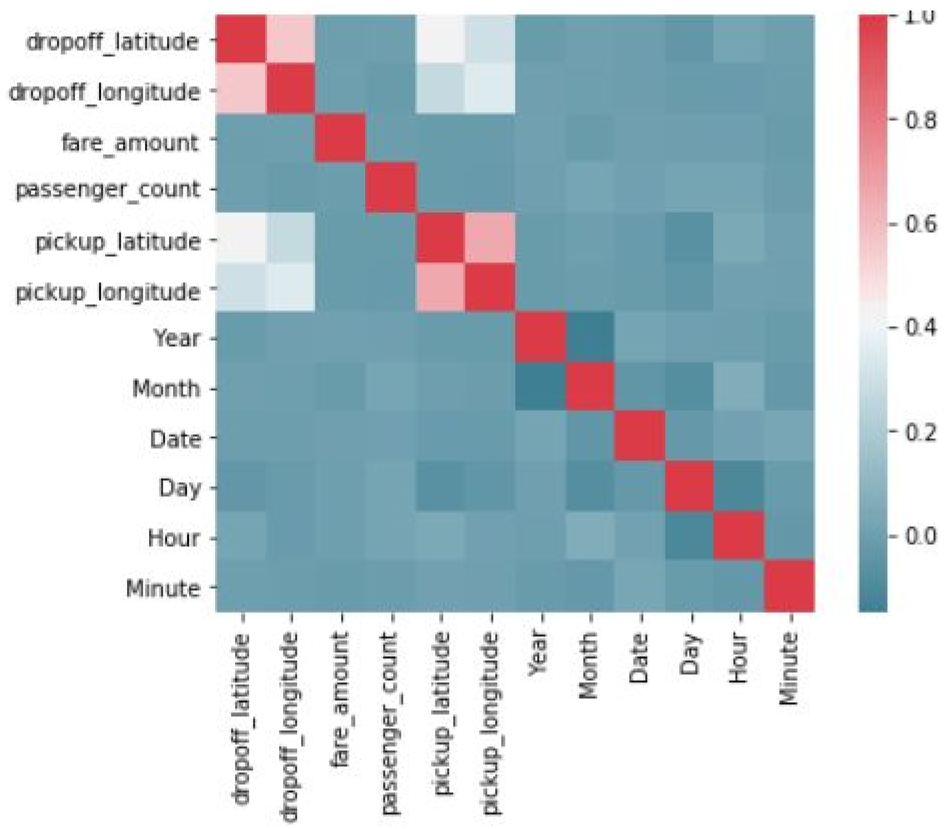
It will helps to train the models well.

2.5 Feature Selection

In this step we would allow only to pass relevant features to further steps. We remove irrelevant features from the dataset. We do this by some statistical techniques, like we look for features which will not be helpful in predicting the target variables. In this dataset we have to predict the fare_amount. Further below are some types of test involved for feature selection:

Correlation analysis –

1. This requires only numerical variables. Therefore, we will filter
out only numerical variables and feed it to correlation analysis. We do this by plotting correlation plots for all numerical variables. There should be no correlation between independent variables but there should be high correlation between independent variable and dependent variable. So, we plot the correlation plot. we can see that in correlation plot faded colour like skin colour indicates that
2. variables are highly correlated with each other. As the colour fades correlation values increases. From below correlation plot we see that:



From this we can observe that pickup latitude and pickup longitude are positively correlated to each other. We can remove one column because these are positively correlated with each which not give any useful information.

2.6 Feature Scaling

Data Scaling methods are used when we want our variables in data to scaled on common ground. It is performed only on continuous variables.

• Normalization: Normalization refers to the dividing of a vector by its length. normalization normalizes the data in the range of 0 to 1. It is generally used when we are planning to use distance method for our model development purpose such as KNN. Normalizing the data improves convergence of such algorithms.

Chapter 3

Splitting train and test Dataset.

We need to split the entire data in to train and test based on certain sampling techniques like random sampling, stratified sampling etc..

I have followed random sampling techniques with 30% test data and 70% train data.

Once we have splitted the data we need to build the machine learning model on top of train data and with the test data we will find the performance of the model. If any performance needs to improve we need to go for the cross validation where we will tune the hyperparameter values of the machine learning models.

After splitting the data:

Train shape: (17085, 11) (17085,)
Test Shape: (8416, 11) (8416,)

Chapter 4

4.1 Model Development

Our problem statement wants us to predict the fare_amount. This is a Regression problem. So, we are going to build regression models on training data and predict it on test data. In this project I have built models using 5 Regression Algorithms:
I. Linear Regression
II. Ridge Regression
III. Lasso Regression
IV. Decision Tree
V. Random Forest
VI. Xgboost Regression

We will evaluate performance on validation dataset which was generated using Sampling. We will deal with specific error metrics like – Regression metrics for our Models: -
- r square
- Adjusted r square
- MAPE (Mean Absolute Percentage Error)
- MSE(Mean square Error)
- RMSE(Root Mean Square Error)

4.2 Model Performance

Here, we will evaluate the performance of different Regression models based on different Error Metrics

Linear regression:

```
==================Building the model... ==========================
Model LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False) ran
successfully..


====================== Score's ==================================
r square :  -0.062267888135728144
Adjusted r square : -0.06365829113067023
MAPE : 115.3805765311416
```

```
MSE : 2478.8495793492157
RMSE : 49.788046550846076
```

Ridge:

```
==================Building the model... ==========================
Model Ridge(alpha=1.0, copy_X=True, fit_intercept=True, max_iter=None,
   normalize=False, random_state=None, solver='auto', tol=0.001) ran successfully..


====================== Score's ==================================
r square :  -0.06212661249933005
Adjusted r square : -0.06351683057851765
MAPE : 115.30725541539431
MSE : 2478.519906339451
RMSE : 49.78473567610308
```

Lasso:

```
==================Building the model... ==========================
Model Lasso(alpha=1.0, copy_X=True, fit_intercept=True, max_iter=1000,
   normalize=False, positive=False, precompute=False, random_state=None,
   selection='cyclic', tol=0.0001, warm_start=False) ran successfully..


====================== Score's ==================================
r square :  -0.006552050987458813
Adjusted r square : -0.007869527493986839
MAPE : 77.13450954847055
MSE : 2348.8341839667373
RMSE : 48.46477260822274
```

KNN:

```
==================Building the model... ==========================
Model KNeighborsRegressor(algorithm='auto', leaf_size=30, metric='minkowski',
          metric_params=None, n_jobs=1, n_neighbors=5, p=2,
          weights='uniform') ran successfully..


====================== Score's ==================================
r square :  -12.030828994841643
Adjusted r square : -12.047885053735415
MAPE : 73.08036373193538
```

```
MSE : 30408.02168003364
RMSE : 174.3789599694689
```

DecisionTreeRegressor:

```
===================Building the model... ===========================
Model DecisionTreeRegressor(criterion='mse', max_depth=None, max_features=None,
           max_leaf_nodes=None, min_impurity_decrease=0.0,
           min_impurity_split=None, min_samples_leaf=1,
           min_samples_split=2, min_weight_fraction_leaf=0.0,
           presort=False, random_state=None, splitter='best') ran successfully..


======================= Score's ===================================
r square :  -0.004573787642432547
Adjusted r square : -0.005888674799032545
MAPE : 47.11650414546063
MSE : 2344.217818061837
RMSE : 48.417123190683654
```

RandomForestRegressor :

```
===================Building the model... ===========================
Model RandomForestRegressor(bootstrap=True, criterion='mse', max_depth=None,
           max_features='auto', max_leaf_nodes=None,
           min_impurity_decrease=0.0, min_impurity_split=None,
           min_samples_leaf=1, min_samples_split=2,
           min_weight_fraction_leaf=0.0, n_estimators=10, n_jobs=1,
           oob_score=False, random_state=None, verbose=0, warm_start=False) ran successfully..


======================= Score's ===================================
r square :  -10.494758978513847
Adjusted r square : -10.509804474559022
MAPE : 67.41123237476117
MSE : 26823.533664955292
RMSE : 163.77891703438294
```

XGBRegressor:

```
===================Building the model... ===========================
[02:37:23] WARNING:
C:/Jenkins/workspace/xgboost-win64_release_0.90/src/objective/regression_obj.cu:152: reg:linear is
now deprecated in favor of reg:squarederror.
Model XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
       colsample_bynode=1, colsample_bytree=1, gamma=0,
       importance_type='gain', learning_rate=0.1, max_delta_step=0,
       max_depth=3, min_child_weight=1, missing=None, n_estimators=100,
       n_jobs=1, nthread=None, objective='reg:linear', random_state=0,
       reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=None,
       silent=None, subsample=1, verbosity=1) ran successfully..


======================= Score's ===================================
r square :  -1.661918479403801
Adjusted r square : -1.6654026658951673
MAPE : 66.80702936695616
MSE : 6211.705706846061
RMSE : 78.81437500130329
```

4.3 Final best model selection:

From all the different models. Decision tree regression gave best results. The RMSE, MAPE, MSE values of decision tree is less than the other models. So we consider the decision tree.

```
===================Building the model... ==========================
Model DecisionTreeRegressor(criterion='mse', max_depth=None, max_features=None,
          max_leaf_nodes=None, min_impurity_decrease=0.0,
          min_impurity_split=None, min_samples_leaf=1,
          min_samples_split=2, min_weight_fraction_leaf=0.0,
          presort=False, random_state=None, splitter='best') ran successfully..


======================= Score's ===================================
r square :  -0.004573787642432547
Adjusted r square : -0.005888674799032545
MAPE : 47.11650414546063
MSE : 2344.217818061837
RMSE : 48.417123190683654
```

Root mean square error is 48%. The lower the RMSE value the better the model performs.

Chapter 5

Storing the model for deployment:

We can store the model in many different ways like json, pickle, csv etc in python.. But here I use pickle way of sterilizing the model.

Python pickle module is used for serializing and de-serializing a Python object structure. Any object in Python can be pickled so that it can be saved on disk. What pickle does is that it “serializes” the object first before writing it to file. Pickling is a way to convert a python object (list, dict, etc.) into a character stream. The idea is that this character stream contains all the information necessary to reconstruct the object in another python script.

When our model is trained , we need to dump or store the model. To do so we use a dump method in pickle to store the model data. When we want to predict the on test data we load the pickle object which we stored for the best model and will use predict method to predict for the test data..