Report

Preprocessing and EDA Steps:

1. The training data given was having 55M rows and it is not really feasible to train on all that data, so I have considered 2M rows for training the models.

2. The fare_amount feature had some negative values so got rid of them since they are definitely wrong values.

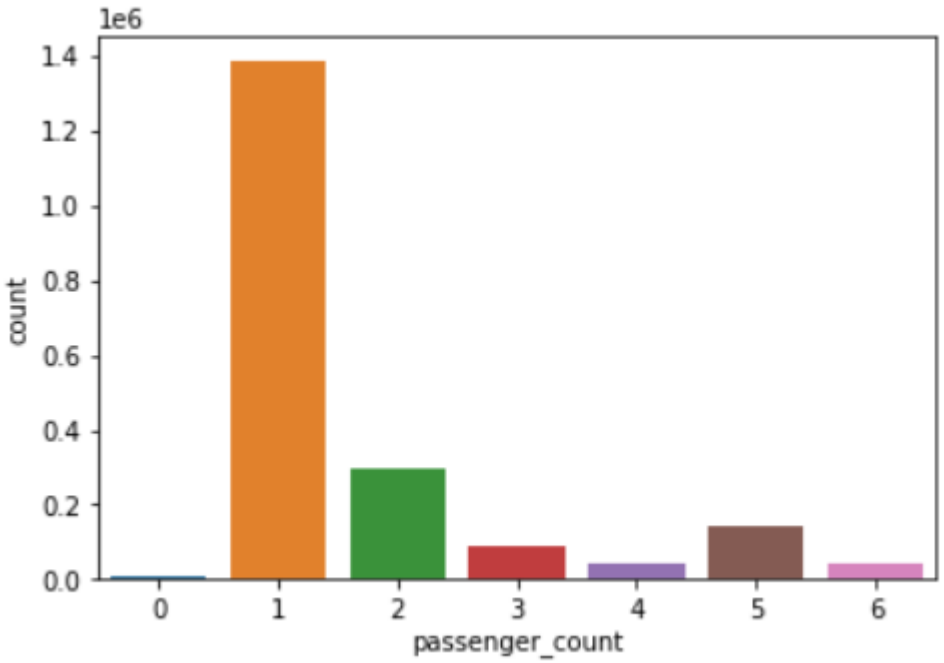
3. Got rid of some other outliers in features such as pickup_latitude, pickup_longitude, dropoff_latitude, dropoff_longitude etc by carefully examining the data.(latitude values should lie between -90 to 90 and longitude values -180 to 180. Anything outside of this is labelled as an outlier and removed)

4. Extracted Year,Month,Day,Hour and Day of Week out of pickup_datetime and created separate features for all of them.

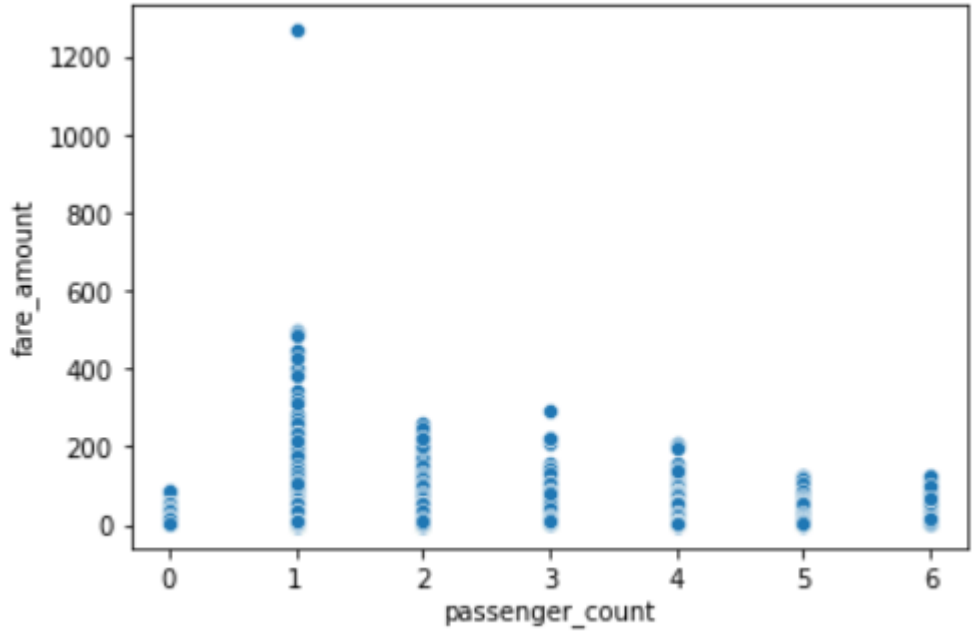
5. Created a seperate field called distance that will store the distance covered between pickup and the drop using Haversine formula and got rid of some NAN values.

6. Made some plots to get a better understanding.

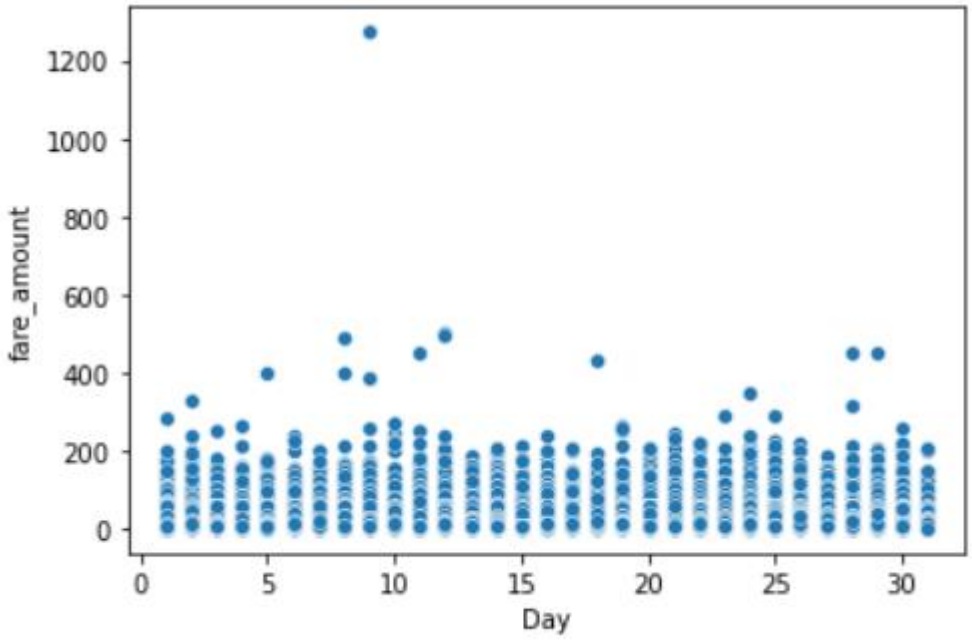
   Countplot of passenger_count feature



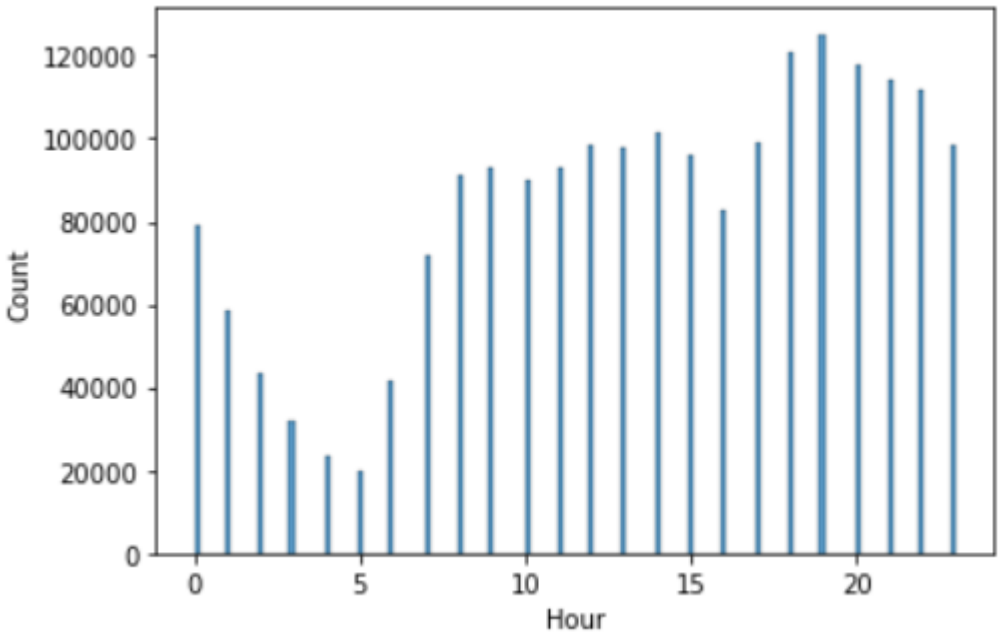
Scatterplot of passenger_count Vs fare_amount



Scatterplot of Day Vs fare_amount

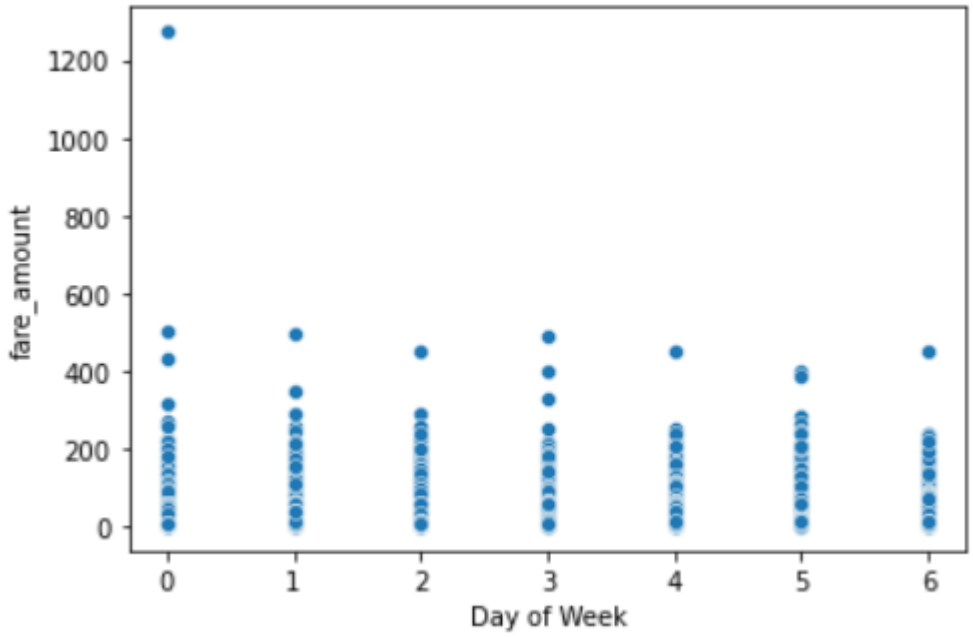


Histplot of Hour Vs Count



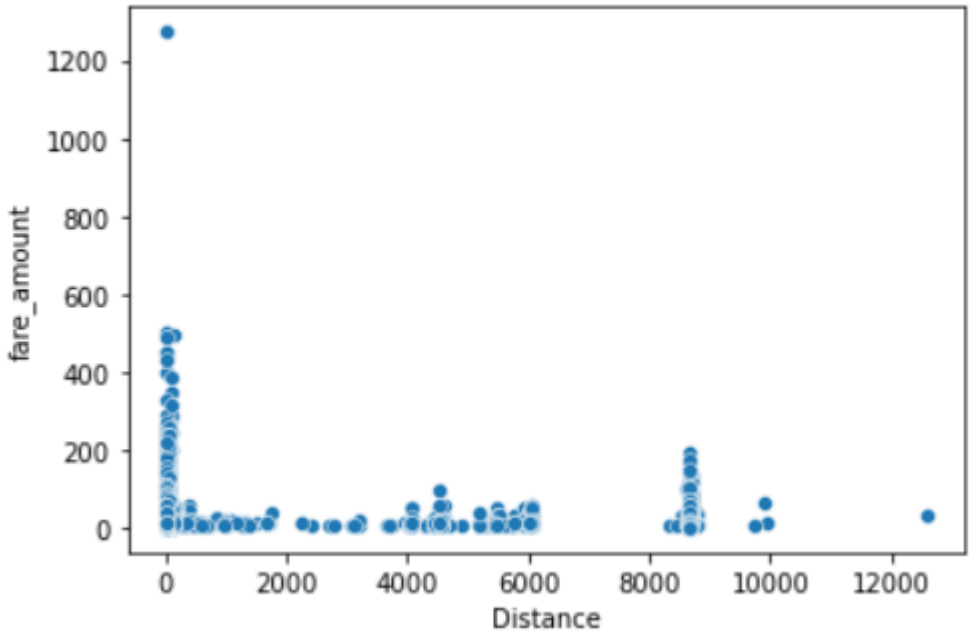
From this it looks like the time of the day plays a mojor role. The frequency of rides is lowest at 5am and highest at 7pm.

Scatterplot of Day of Week Vs fare_amount



The peak fare seem to be on sunday and monday and lowest on wednesday and Friday.

Scatterplot of Distance Vs fare_amount



7. Normalized the abnormally high distances using the .apply method where 2.5 is the base fare and 1.56 is the fare/km in NYC.

8. Adjusted the base fair as per day of week and time.

9. Repeated all the above steps for the test data as well.

Modelling and Top 2 Scores(using 2 Million data rows):

1st Top Score(RMSE): 3.08537

Using LightGBM with params : {'objective': 'regression', 'boosting': 'gbdt', 'reg_sqrt': True, 'learning_rate': 0.03, 'num_leaves': 1200, 'max_depth': -1, 'max_bin': 500, 'num_rounds': 500, 'early_stopping_round': 10, 'metric': 'rmse', 'force_row_wise': True}

Reason why it performed well compared to other models:

LightGBM is a histogram based algorithm that buckets the continuous feature values into discrete bins which helps reduce the training time. It provides better accuracy than any other

boosting method and goes neck to neck with XGBoost but is way faster than XGBoost, that's why I decided to go with it. It does leaf wise split approach rather than level-wise which is one of the factors in getting good accuracy. But, it could lead to overfitting which could be avoided by enabling the max_depth parameter. Another reason of picking it being it works really well with large datasets which was the case here.

2nd Top Score(RMSE): 3.13135

Using XGBoost Regressor with params: {n_estimators = 100, max_depth = 9}

Submission and Description	Private Score	Public Score	Use for Final Score
submission_xgboost.csv 39 minutes ago by Ayan Kumar Pahari add submission details	3.13135	3.13135	☐

Reason why it performed well compared to other models:

Reason for such a good score being it is proven to push the limits of computing power of boosted tree algorithms and often used for winning various Kaggle Competitions. It is build for the sole purpose of best performance and speed. This comes under ensemble methods where we create a strong classifier based on weak classifiers. It has many regularization parameters which also helps to prevent overfitting.

Other Models(which didn't work as well): In addition to the above two mentioned, I have also used Decision Trees, Random Forests, Gradient Boost etc. But they didn't give such good results, reasons are as follows:

1. **Decision Tree**: Not performance oriented. Gives a really low score and could overfit if max_depth is not properly tuned.

2. **Random Forests**: Better accuracy than decision trees, but for such a large dataset it is not recommended since it takes forever to train and the score is also not upto the mark compared to LGBM and XGBoost.

3. **Gradient Boost**: Better accuracy than both mentioned above and speed is also good but better techniques are available that can leverage the potential of Gradient Boost to the next level like Extreme Gradient Boost or XGBoost.

Note: I have also tried LGBM with 5 Million data rows which gave me a RMSE of **2.99722**. Reason for not including it on the top score being it takes a lot of time to fully train.