Project Report

Problem statement: (Regression)

To design a machine learning model that predicts the fare amount for a cab ride in the city.

Approach in Python:

Exploratory data analysis

- Train dataset was stored in **cab_df** dataframe using the pd.read_csv() function.
- datatypes and non-null entries of the features checked using the **.info()** method.
- fare_amount and passenger_count had null entries.
- pickup_datetime and fare_amount had object datatype.
- summary of the features in dataset were checked using **.describe()** method.
- minimum fare_amount was negative.
- maximum of pickup_latitude was found to be 401.08, instead of being in range from -90 to +90.
- maxium passenger_count of 5345 does not seem to be realistic for a cab with holds at max 5-6 people.
- While converting datatype of fare_amount ,an **invalid amount='430-'** was observed.

Cleaning Data

- intersection of latitude and longitude of zero is somewhere in Atlantic Ocean, so we drop it from dataset.
- same pickup and dropoff coordinates were observed, which suggest some sort of cancellation fee was charged.
- Records with cancellation fee greater than $50 were dropped.
- Maximum passenger count of 6 was considered (i.e in case of a SUV).
- For fare_amount greater than 200, 4 outliers were observed and dropped.
- Geopy module was imported to calculate distance between the source and destination coordinates.
- Datatype of pickup_datetime changed from object to datetime.
- Extracted hour, day and year from the pickup_datetime column.
- A box plot was plotted on total_distance to check for any outliers.
- Distances greater than 3000km were dropped.
- Dropped false values of passenger count (i.e. 1.3, 0.12)

Visualisation

- From the count plot and scatter plot, we can see single passengers were the most frequent travellers, and the highest fare also come from cabs which carry just 1 passenger.
- From the count plot, the frequency of cab rides seems to be the **lowest at 5AM** and the **highest at 6PM**.
- However, fare seems to be high between 4-5AM and 2-4PM time slot.
- As mostly people leave for work early to avoid traffic, and frequency of cabs available in early morning is less.
- Also the cab fares are high during rush hours/ traffic.
- The total fare_amount is more during **friday and saturday** as people love to travel or go home during weekends.
- From the **heatmap**, we can see total_distance is highly correlated with fare_amount.
- From the scatter plot between fare_amount and total_distance we can see few outliers with distances greater than 80km.
- Imputing those records with Average of 1 $USD/km and basefare of $2.5.

Training data

- Features and labels were separated into numpy arrays X and y.
- Pipeline was used with to run the flow in steps sequentially.
- StandardScaler() was used, so that the data is normally distributed with mean of zero and standard deviation of 1.
- Simple **LinearRegression** was used first for our Regression model.
- Train_test_split function splits the data into 70% train and 30% test.
- After training the model with training data and testing it on test data, following results were observed, **R^2: 0.70 and RMSE: 5.535**
- For regression, both R2 and RMSE indicate the **goodness of the fit**.
- R-squared is scaled between 0 and 1, whereas RMSE is not scaled to any particular values.
- Even though R-squared can be more easily interpreted, but with RMSE we explicitly know how much our predictions deviate, on average, from the actual values in the dataset. Hence, we choose **RMSE as our error metric.**
- Here, *R^2 tells us 70% of the variability in the fare_amount is explained by our model*
- Next, **Random Forest** algorithm was used to train our model.
- **Randomized Search CV** was used to **tune the hyperparameters** in the random_grid.
- After predicting on the best_estimator from grid, we got **RMSE =3.699 and R^2=0.867**
- As we can see random forest model performed way better than linear regression here.

- Finally we will use **XGBoost,** a gradient boosting algorithm to further improve our RMSE.
- By tuning the parameters we achieved a **RMSE = 3.390.**
- Out of all the 3 models we trained so far, **XGBoost** performed well with least RMSE.

Approach in R:

- Explored the cab_data dataset using **dim()** and **summary()** command to check for dimensions and spread of the features.
- fare_amount and pickup_datetime columns were of type factor.
- **Geosphere** package was used to calculate distance from pickup and dropoff coordinates.
- Removed invalid latitude coordinate using **filter** function of **dplyr** package.
- converted datatype of fare amount from factor to numeric.
- considered max 6 passengers for cab in case of SUV.
- dropped distances> 1000Km and fare amount >200 from dataset as outliers.
- **POSIXlt** function was used to seperate hour,weekday,month and year from pickup_datetime.
- Splitted the dataset into 70% train and 30% test.
- A simple **linear regression** model was used to train on training set.
- Prediction was made on the testing set, resulting in an **RMSE=5.56**
- Next, a **random forest** model was used, resulting in an **RMSE=3.63**
- Finally, a Support Vector Machine(SVM) was used, resulting in an **RMSE=3.65**
- Out of all the 3 models we trained so far, **random forest** performed well with least RMSE.