Flight Price Prediction

Ravishankar Yadav

B20EE048

Abstract – This paper reports my experience with building a Flight Price Prediction model. From a consumer's standpoint, determining the best time to buy plane tickets is difficult, mostly because consumers lack sufficient knowledge to make informed decisions regarding future price fluctuations. The main goal of this project was to use historical data to find underlying trends in flight prices in India, as well as to recommend the optimum time to buy a flight ticket.

I. Introduction

The airline sector is one of the most advanced in its use of dynamic pricing tactics, which are based on proprietary algorithms and hidden variables, to maximise revenue. As a result, customers find it difficult to forecast future price changes. With the availability of airfare information on the internet, customers are attempting to follow the price of a flight over a period of time and predict the future price changes. However, it turns out that estimating the price of a flight solely based on observation is challenging. This project aims to develop algorithms for predicting the cost of airline tickets. The elements that may influence the pricing, such as the weekday of departure and the number of stops on the route, are fed into this algorithm as input.

Datasets

ticket: The file ticket.csv is used as the training dataset.

The train dataset contains 10683 rows where each row represents flight details with 10 columns containing:

- Airline, Date of Journey, Source, Destination, Total stops
- Route, Departure time, Arrival time, Duration, Additional info
- The target column is 'Price'.

The dataset has been split into train and test with test size of 0.2

II. Methodology

Overview

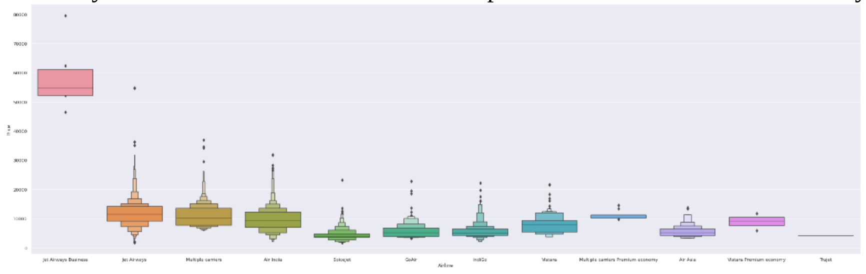
There are various regression algorithms present out of which we shall implement the following:

- *Random Forest Regressor*
- *XGB Regressor*
- *LightGBM Regressor*
- *ExtraTreesRegressor*
- *KNeighbors Regressor*
- *GradientBoosting Regressor*
- *HistGradientRegressor*
- *LazyRegressor*

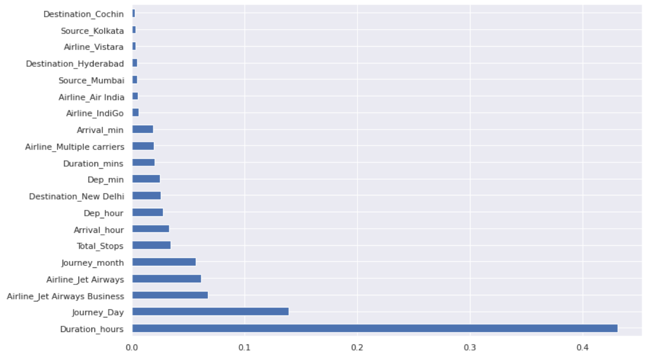
Exploring the dataset and pre-processing:

- On counting the number of NULL values in the train dataset, it was found that there are just 2 NULL values present. Those rows were dropped. It was further observed that Date_of_journey is an object datatype. Therefore, this datatype was converted to timestamp using to_datetime from pandas library.

- We then further analyse departure time column and extract the hours and minutes from this column before dropping it. The same procedure is repeated for arrival time column as well.
- In duration column, we observe that the format of time needs to converted to x hours y minutes. After conversion, the old column is dropped as it is of no use.
- To handle categorical data, we use encoders: OneHotEncoder in the case of nominal data and LabelEncoder in the case of ordinal data.
- We analyse the relation between airlines and prices and conclude that Jet Airways charges maximum.



- We also observe that there is no need for 'Route' because total stops are already given and 'Additional info' column because 78% of it is No-info. Hence, these are dropped.
- We then remove the string from Total stops column and convert strings like '1 stop' to numbers like 1.
- For feature selection, we use RandomForestModel.feature_importances and plot a graph.



Implementation of regression algorithms

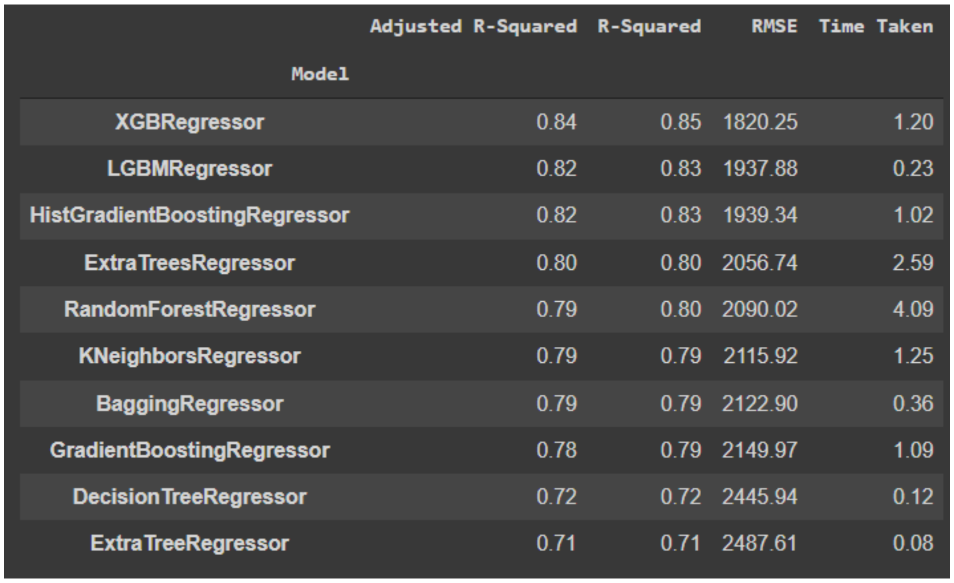
- *Random Forest Regressor :* Random Forest Regressors is a supervised learning algorithms that uses ensemble learning method for regression. It is a meta estimator that fits a number of classifying decision trees on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting.
    - The Random Forest Regressor made in this project gave us:
        - Training score of 0.952922189206
        - Testing score of 0.7966163873905
- *LightGBM Regressor:* It is a gradient boosting framework that makes use of tree based learning algorithms that is considered to be a very powerful algorithm when it comes to computation. It splits the tree leaf-wise opposed to other boosting algorithms that grow tree level wise. It chooses the leaf with maximum delta loss to grow.
    - The LightGBM Regressor made in this project gave us:
        - Training score of 0.9353790824718483
        - Testing score of 0.836332887966892

- *XGB Regressor:* XGBoost is a popular and efficient implementation of gradient boosted trees algorithm. XGBoost minimizes a regularized objective function that combines a convex loss function and penalty term for model complexity. In this project, this is the best model. This model is also tuned using randomsearchcv.
    - The XGB Regressor made in this project gave us:
        - Training score of 0.952493219207
        - Testing score of 0.846332117966892

IV. Evaluation of models

The models implemented were evaluated using techniques like – Adjusted R-squared, R-squared, RMSE, Time Taken accuracy score and overlapping plots, scatter plot.

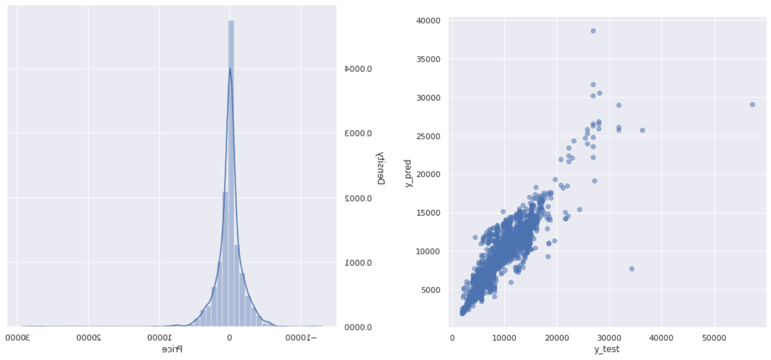
Table 1.1

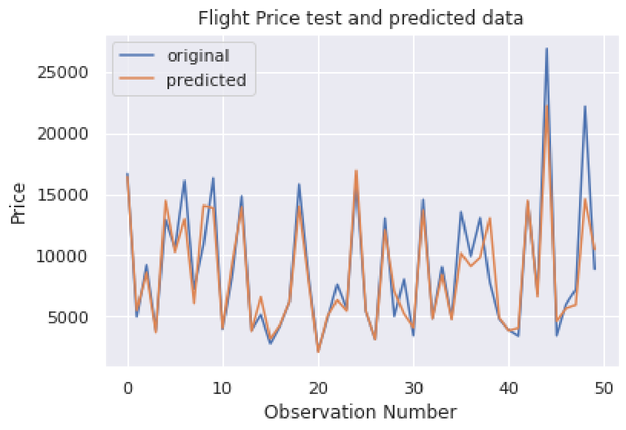


Model	Adjusted R-Squared	R-Squared	RMSE	Time Taken
XGBRegressor	0.84	0.85	1820.25	1.20
LGBMRegressor	0.82	0.83	1937.88	0.23
HistGradientBoostingRegressor	0.82	0.83	1939.34	1.02
ExtraTreesRegressor	0.80	0.80	2056.74	2.59
RandomForestRegressor	0.79	0.80	2090.02	4.09
KNeighborsRegressor	0.79	0.79	2115.92	1.25
BaggingRegressor	0.79	0.79	2122.90	0.36
GradientBoostingRegressor	0.78	0.79	2149.97	1.09
DecisionTreeRegressor	0.72	0.72	2445.94	0.12
ExtraTreeRegressor	0.71	0.71	2487.61	0.08

Plots

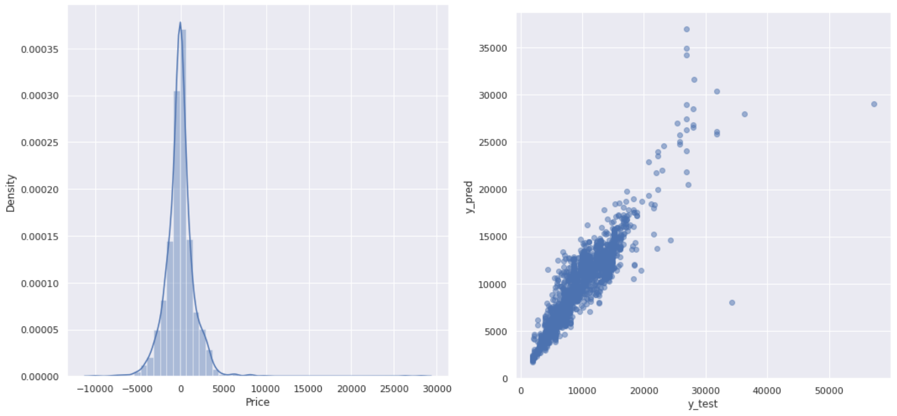
1. XGB Regressor:

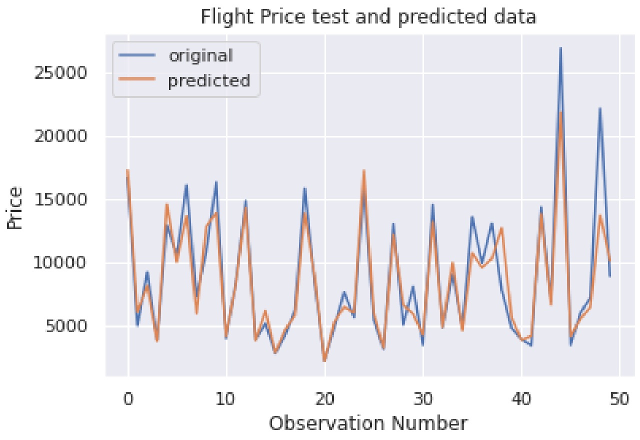




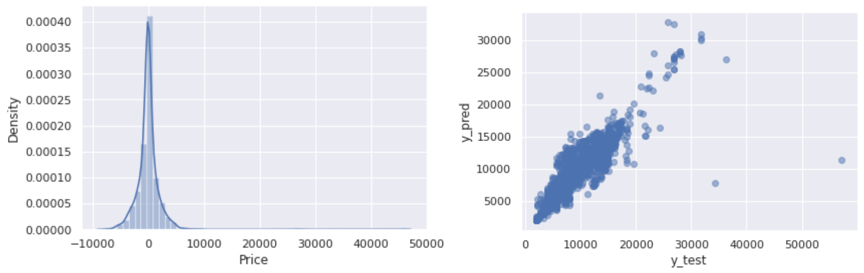
The model predictions and original prices are overlapping, this visual result confirms the high model score we obtained earlier.

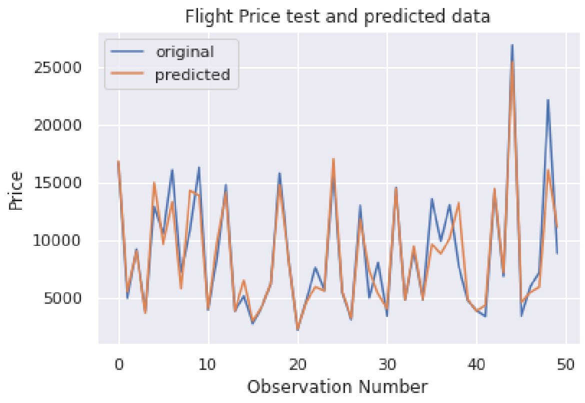
2. LightGBM Regressor:





3. RandomForest Regressor:





VI. Results

The model predictions and original prices are overlapping, this visual result confirms the high model score we obtained earlier.