Final Report on Flight Price Prediction

DATA1030-Fall2022: Hands-on Data Science

Chuning Xiao, 12/9/2022

Introduction

In this project, I'm using the dataset called "flight-price-prediction" which includes some extracted data from an online booking agency called "Ease My Trip" using a tool called "Octoparse" and contains information related to booking flights that were flying in between six major cities in India every day for 50 days, such as departure time/city, arrival time/city and ticket price, etc.… When selecting "Ticket Price" as my target variable, the problem that I'm trying to solve here is a regression one: given such information about a new airline, could we confidently predict the flight price?

This question is particularly interesting to me since by selecting the right model to predict prices we can find cheaper tickets within our tolerable standards and some of the correlation of the features and ticket prices turned out anti-intuitive.

To get an overview of this dataset, there are 300,153 data points and 10 features. This is a relatively clean dataset where the contributor on Kaggle already did a fair amount of engineering work, which I will specify below. The features are:

"Airline": categorical features including names of 6 unique airline companies; "Flight": categorical features that identify certain routes by certain airlines, which the contributor combined from raw data and generated over 3000 categories; "Source City": categorical features including names of 6 departure cities. "Departure Time": 6 categorical time features like "morning" and "afternoon" which are created by the contributor by binning the original time data into different time periods in the day. "Stops": A categorical feature with 3 distinct values that stores the number of stops between the source and destination cities. "Arrival Time": 6 categorical time features like "morning" and "afternoon" which are created by the contributor by binning the original time data into different time periods in the day. Destination City: categorical features which are the same 6 cities as departure cities. Class: categorical features having business class and economy class; "Duration": a continuous feature that displays the length of the flight's time span and transformed. "Days Left": the contributor combined this data in the raw dataset with derived record time, took the difference, and made this into a new feature.

The previous works which had the same machine-learning question are included below in references. In this particular work that I studied from, the author was trying to run a dummy variable regression model and then build a supervised ML Model with higher accuracy of prediction.

EDA

Moving on to exploratory analysis of this dataset, here are couple of interesting findings during my EDA process:

1. One significant cause of price difference is different carriers, which is caused by the dramatic price variance between business class and economic class, since only two of the six carriers have business class for booking. The Figure is illustrated below:

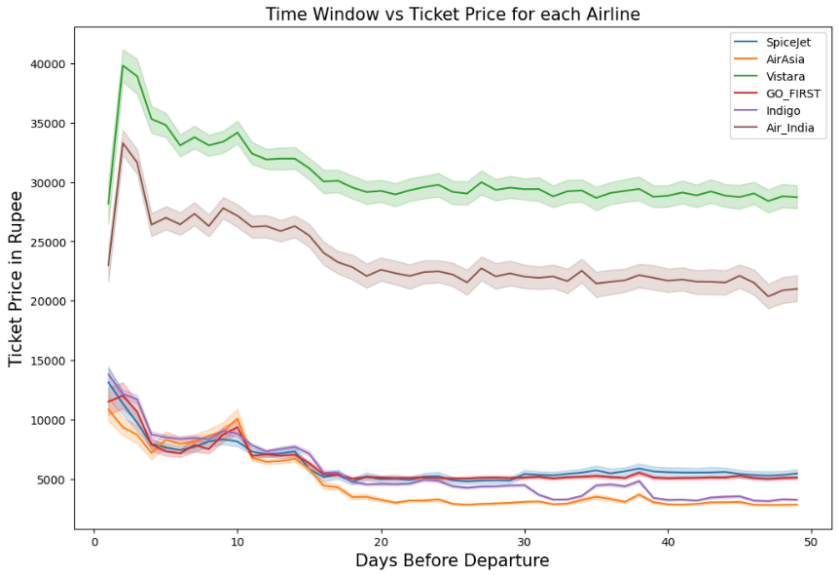

Figure1. Vistara and Air India have significantly more expensive tickets on average than the four other carriers.

From Figure 1 above we can also notice that with the time getting closer to departure, the prices of flight ticket are moving in a smooth fashion until 20 days prior, where they started to increase for all six carriers and for Vistara and Air India, i.e. The expensive airlines, their price dropped drastically since about 2 or 3 days prior to departure. This is understandable because when the flight is not fully sold out due to their lower cost performance, the company would attempt to drop the price to fill the seats to cut their sunk cost of oil and labor, etc.

2. What surprised me the most during my exploration of different features versus target variable is that the relationship between flight duration and ticket price is not linear.

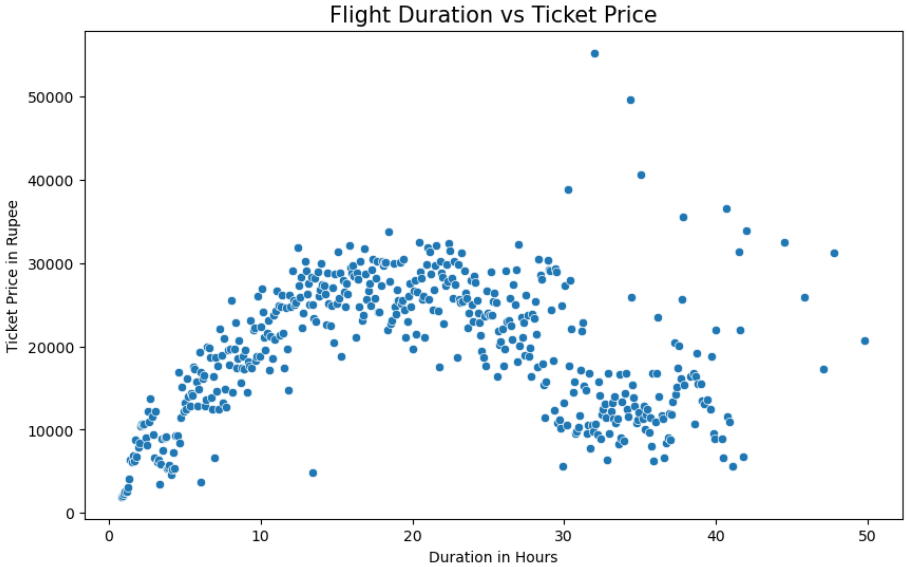

Figure 2. Non-linear relationship between flight duration and ticket price.

From Figure 2 we can find that as the duration of the flight increases, the average flight ticket price peaks at around 18 hours and then it starts to drop. This can be interpreted as the result of customers' willingness to trade extra stops and longer layovers with cheaper prices. What is also noticeable is that there are a significant number of outliers who are fairly expensive with very unrealistic duration and

should not be booked by any reasonable human being. This is a result of auto-generated flight information, where the online agency generates all the possible combination of available flights given certain departure and arrival conditions as a strategic distraction to make other choices seem more reasonable and even more attractive. I choose to keep these data points because they are, after all, real data points in the web-searching results and manually filtering them on a certain price and duration level might interfere with the predictive power of the model.

3. By observing the distribution of the target variable, we can find that most of the ticket are within the cheaper range, while it is a combination of two normal distributions due to two classes of tickets.

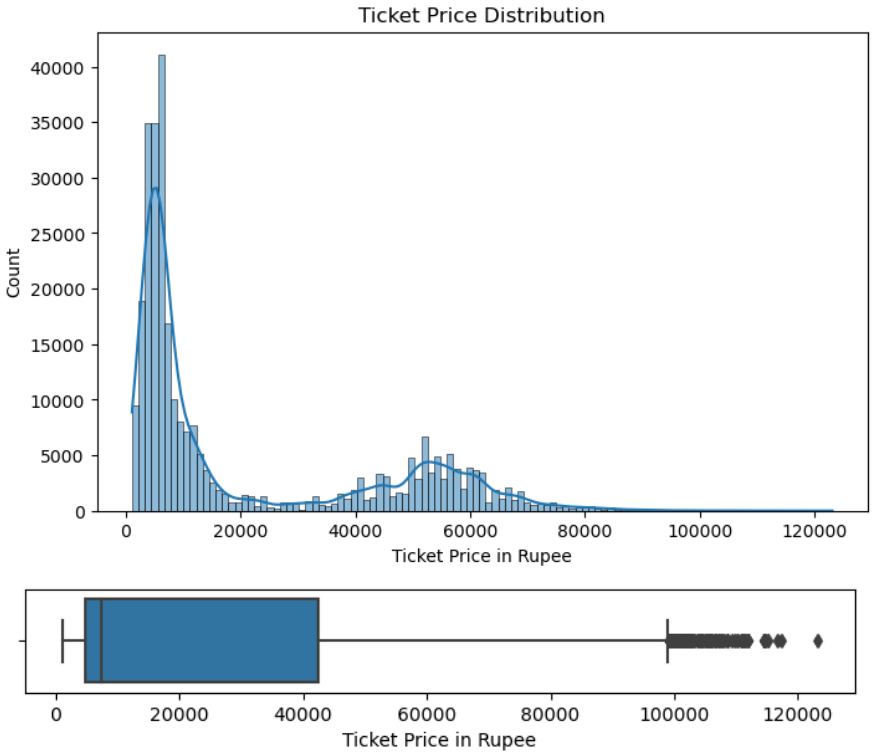

Figure 3.and 4. Ticket price has two peaks due to the separation of business and economic classes but mostly cheap.

We can also notice a significant number of outliers here in Figure 4 which could be explained with the same reason as in the last question.

Methods

Splitting:

For this dataset, I chose to do a basic train_test_split due to its IDD nature and a large amount of data included. While the target variable does not have a group structure, it does have some imbalanced distribution. It is not a time-series. I would split the data into 90-10-10 for train-test-split.

Pre-processing:

I chose MinMaxEncoder for all the continuous features since the duration of a flight and the time window are all reasonably bounded. I chose OneHotEncoder for all the categorical features since SimpleImputer helps the categorical features to be understandable by machine learning models. After applying transformers, I have 37 features.

Pipeline

For evaluation of models, I was determined to use adjusted R^2 score as my evaluation metric since it is a common metric for pricing regression problems based on my research, and almost all of the relative works on this dataset are pre-determined to use adjusted R^2 score. The reasoning behind this prevalence

might be adjusted R^2 score taking consideration of the number of independent variables which punishes the models with higher variance as a result.

The very first ML algorithm I tried on this dataset is linear regression and it worked poorly even after tunning hyperparameters on a wide range of scales. This is understandable due to the non-linear nature of my target variable's distribution. The accuracy was under 0.70 and I moved on to other models right away.

Then I chose Random Forest Regression as the next model to try, and it worked out significantly better than Linear Regression. After tuning hyperparameters including max_features on "auto" and "sqrt"; max_depth on an array of values between 10 and 120; n_estimators on [5,20,50,100,200], the validation score was maxed at around 0.85.

To increase the predictive power, I then decided to choose a similar but better performed ML algorithm, Extra Tree Regression. The reason that I chose this algorithm as a substitute to Random Forest Regression is that the process of splitting and building each stump is more randomized as compared to optimized in Random Forest, lowering the variance of the model. The hyperparameter I tried to tune here are also n_estimators and max_depth of the same values as in Random Forest.

Moving on to the next ML algorithm, I tried XGB Regressor and KNN Regressor and the KNN regressor performed the best. The hyperparameter I tried to tune with XGB Regressor are n_jobs on integers from 1 to 10; learning_rate from 0.1 to 0.9 with steps; max_depth on an array of values between 2 and 20. The hyperparameter I tries to tune with KNN Regressor were n_neighbors on an array of values between 1 and 30; p values on the integers from 1 to 10; leaf_size on integers from 1 to 50. The maximum validation score was around 0.90 and I'm satisfied with this value then stopped.

Comparing all the test scores of the various random states on of the algorithms above, KNN Regressor has the second highest mean and a reasonable standard deviation, while XGB Regressor obtained the highest mean test score while the standard deviation was relatively too high for me to accept. Therefore, I will proceed training with KNN Regressor.

Results

Here is a summary of adjusted R^2 score of the best of each algorithms:

Model_Name	Adj_R_Square	std of Adj_R2	Root_Mean_Squared_Error_RMSE	Mean_Squared_Error_MSE	R2_score
KNeighborsRegressor	0.984565	0.267551	2819.671401	7.950547e+06	0.984566
ExtraTreesRegressor	0.982547	0.282746	2998.379240	8.990278e+06	0.982548
RandomForestRegressor	0.975529	0.284355	3550.419168	1.260548e+07	0.975530
XGBRegressor	0.969915	0.356739	3936.676054	1.549742e+07	0.969916
LinearRegression	0.910380	0.273623	6794.493419	4.616514e+07	0.910383

Figure 5. Ranking of different ML models based on adjusted R2 score.

From Figure 5 we can find that KNN is the model with highest predictive power who performed the best on adjusted R2 score as well as the unadjusted R^2 score, causing my doubt about the purpose of adjustment at the first place. One possible reason for such nuance might be lack of independent variables in my features.

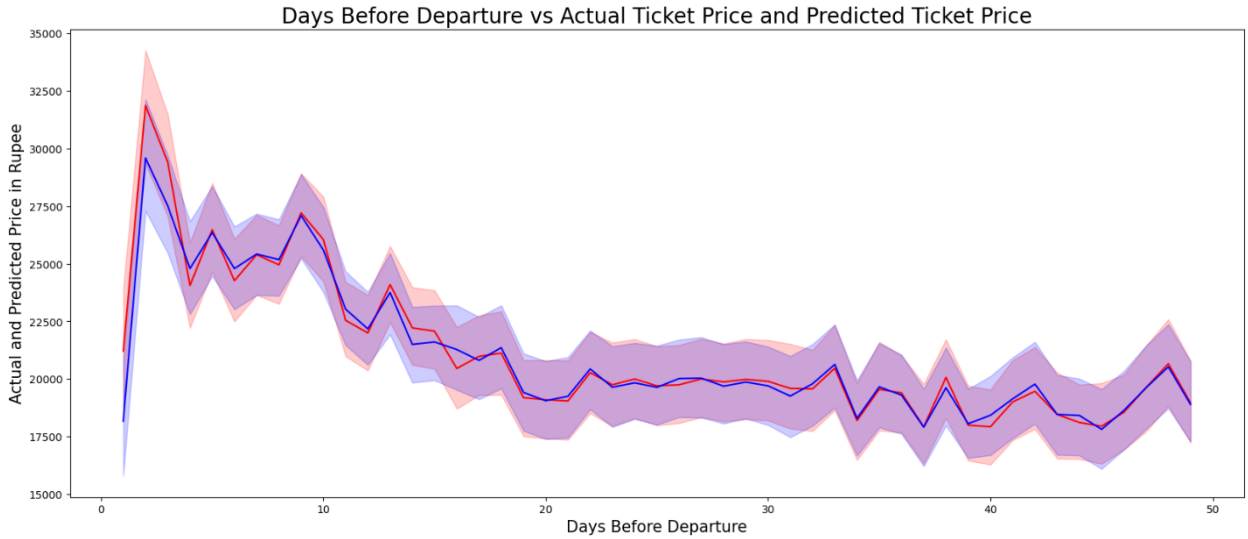

Figure 6. Predicted Price in red and Actual Price in blue versus days before departure

From Figure 6 we can find the predicted price seemingly overlapping well with the true price, with uncertainty around 3 days before departure becoming the highest.

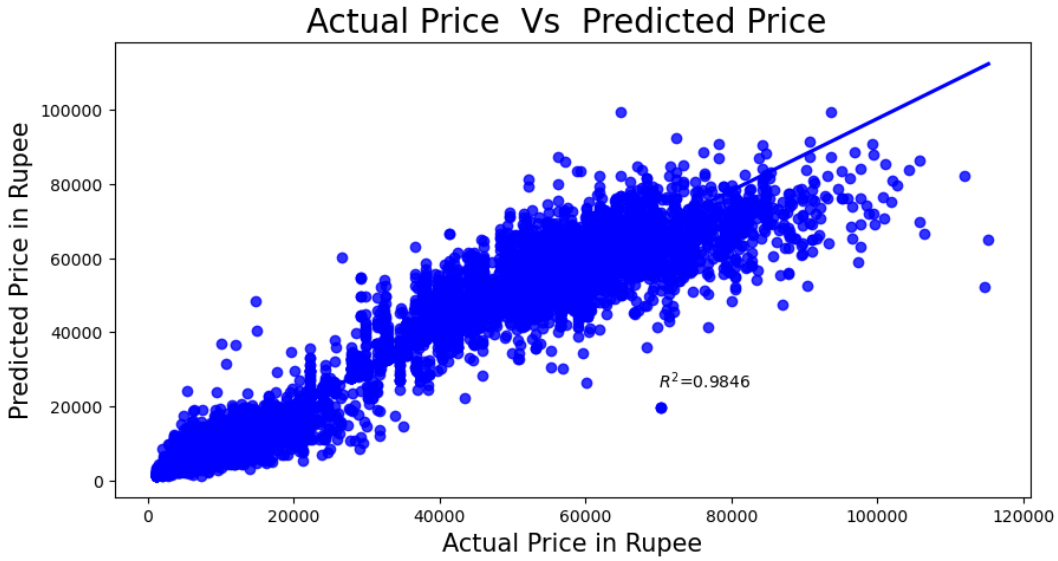

Figure 7. Scatter plot of actual price and predicted price using KNN Regressor

With some trace of the original two separate distributions, we can still tell that the prediction performed well with R^2 score of around 0.98.

Outlook

There are three methods I would consider trying in order to improve the predictive power and efficiency in the future:

- Randomized Search CV as compared to Grid Search CV for efficiency on tunning
- Hyperparameter tuning with folds for better performance
- Stacking and blending to improve R^2 score after final decision on the model

References

https://www.kaggle.com/code/kanuriviveknag/flight-prices-eda-tableau-dashboard

https://www.kaggle.com/code/saibhargav2810/flight-price-prediction-97-6score

GITHUB LINK:

https://github.com/urspuc/Flight-Prices