Flight Delay Prediction

Authors:

Abhishek Kushary, Aju John Thomas, Taru Kaushik, Yuvasree Pamujula

Email Address:

abhishekkush@iisc.ac.in, ajuthomas@iisc.ac.in, tarukaushik@iisc.ac.in, yuvasreep@iisc.ac.in

Table of Contents

[Problem Statement 2](#_Toc184332505)

[Motivation 2](#_Toc184332506)

[Data Source and Description of Data 2](#_Toc184332507)

[Flight Price Dataset 2](#_Toc184332508)

[Airport Dataset 2](#_Toc184332509)

[Tools/Technologies used 2](#_Toc184332510)

[Approach and Methods 2](#_Toc184332511)

[Data Collection and Preprocessing 2](#_Toc184332512)

[Exploratory Data Analysis (EDA) 4](#_Toc184332513)

[Feature Engineering 5](#_Toc184332514)

[Modeling and Evaluation 6](#_Toc184332515)

[Model 1: Random Forest Regressor 6](#_Toc184332516)

[Model 2: XG Boost Regressor 6](#_Toc184332517)

[Model Improvement 7](#_Toc184332518)

[Handling Outliers / Skews 7](#_Toc184332519)

[XGB retraining and cross-validation 7](#_Toc184332520)

[Comparison of Model 1 and Model 2 7](#_Toc184332521)

[Residuals Analysis 8](#_Toc184332522)

[Summary 8](#_Toc184332523)

# Problem Statement

The goal of this project is to predict flight ticket prices based on various factors such as departure and arrival locations, airlines, flight duration and other relevant features.

# Motivation

Accurately predicting flight prices will help to understand and predict various factors (like distance, no. of stops, travel duration etc.) that can impact the fares of flight tickets and thus can help to plan accordingly.

# Data Source and Description of Data

## Flight Price Dataset

* Data Source Link: [Flight Dataset [Kaggle]](https://www.kaggle.com/datasets/dilwong/flightprices)
* The original dataset is in CSV format and contains 27 columns which has ~6M unique values.
* The selected DataFrame includes 1M sample and 13 columns, which are used for further data preprocessing, EDA, training and prediction for implementation.

## Airport Dataset

* Data Source Link: [Airports Dataset [Kaggle]](https://www.kaggle.com/code/fabiendaniel/predicting-flight-delays-tutorial/input?select=airports.csv)
* This dataset is used to convert the IATA code in the flight price dataset to get airport name and city name. This output is then concatenated in the main data for further processing.

# Tools/Technologies used

* **Storage:** drive.google.com
* **Platform:** https://colab.research.google.com
* **Coding Languages:** Python
* **Libraries:** Pandas, NumPy, Matplotlib/Seaborn, Scikit-learn, xgboost, datetime, warnings

# Approach and Methods

## Data Collection and Preprocessing

The initial code is used to display, summarize and show descriptive statistics of the dataframe.

Then the missing and duplicate values are counted and the shape of the dataframe is checked. This helps in quickly understanding the data’s structure and quality.

A computer screen shot of a number

Description automatically generated

A screenshot of a computer

Description automatically generated

From the initial dataset relevant columns were selected for further analysis.

**'flightDate'**: Date of the flight

**'segmentsAirlineName'**: Concat of Multiple airline name for more than one stops

**'segmentsArrivalTimeEpochSeconds'**: Concat of Multiple arrival time in secs

**'segmentsDepartureTimeEpochSeconds'**: Concat of Multiple departure time in secs

**'startingAirport'**: initial airport

**'destinationAirport'**: destination airport

**travelDuration'**: Total time taken for completing the journey

**'isBasicEconomy'**: Type of ticket (Economy Yes or Not)

**'isRefundable'**: Ticket refundable or not

**'isNonStop'**: Direct flight or not

**'totalFare'**: Total price of ticket including taxes and additional fees

**'seatsRemaining'**: Number of available tickets

**'totalTravelDistance'**: Total flight travel distance

The code involves data cleaning to remove duplicate rows from the DataFrame and handle missing values by filling numerical columns with their median values and categorical columns with their mode values. This ensures the data is clean and ready for analysis.

The following data preprocessing steps were then undertaken:

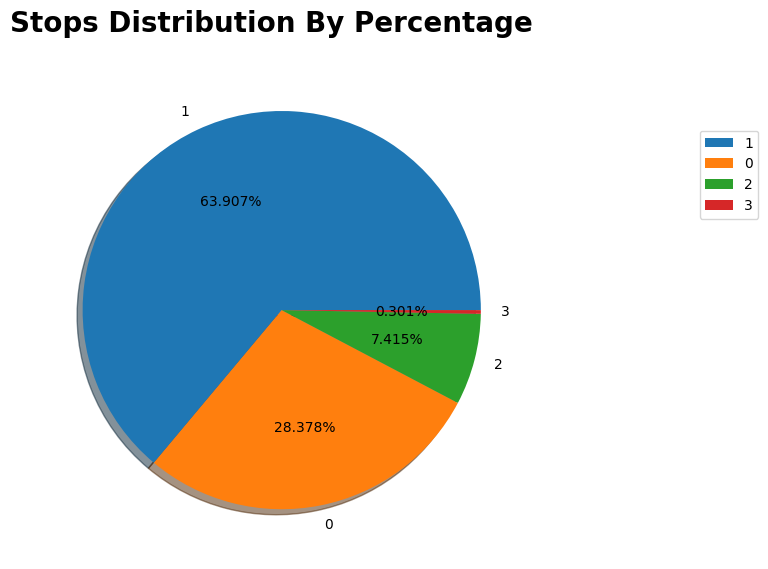
* Converted the flight date to datetime format.
* Transformed travel duration to total minutes.
* Extracted the source airline name.
* Created a new column ‘numberOfStops‘ for the number of stops.
* Converted Boolean columns like isBasicEconomy, isRefundable, and isNonStop to integers.
* Split flightDate into date, month, and year columns.
* Mapped IATA codes to full airport names and cities using a dictionary from US airports data set.
* Generated arrival and departure times from epoch seconds and converted them to detailed UTC format in individual columns for arrival and departure separately.

## Exploratory Data Analysis (EDA)

EDA is utilized to examine the dataframe and correlation between various features using visual graphs and plots.

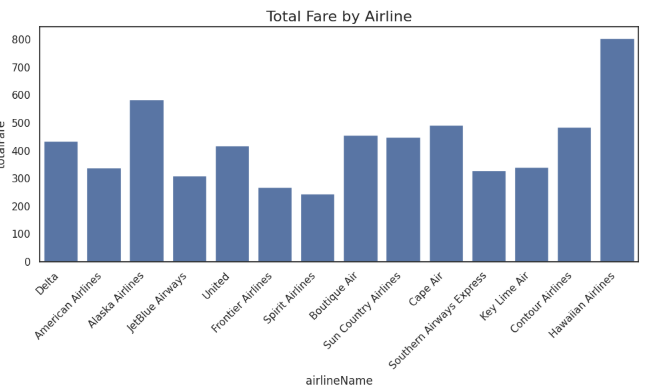
The various EDA graphs provide several insights:

* Frequency of fare usage in the dataset with a histogram.
* A bar plot visualizes total fare by airline,
* Scatterplot compares total fare to travel duration, indicating that nonstop flights have shorter durations.
* A pie chart displays the distribution of stops, revealing that most tickets are for one-stop journeys.
* Bar graphs highlight the frequency of source and destination cities, with New York and Los Angeles being the most preferred.

A graph of a number of bars

Description automatically generated with medium confidence

A graph of blue and orange dots

Description automatically generated

## Feature Engineering

Following steps are applied under Feature Engineering to create new features and modify existing data to understand and highlight important patterns in the data.

* Categorized arrival and departure hours into morning, evening, afternoon, and night.
* Converted months to seasons (fall, spring, summer, winter).
* Combined airport names and city names for both source and destination.
* Selected key columns for predictions, including fare, dates, times, duration, airline, airport cities, seats remaining, stops, distance, and fare-related features.
* Encoded categorical columns using the get\_dummies function and dropped the original categorical columns.
* Insights after feature engineering and EDA:
* Major arrival frequencies at night and departures fairly evenly distributed, with a slight peak in the afternoon.
* Travel duration, number of stops and total distance are the major influencers for total fare in Correlation matrix.

## Modeling and Evaluation

### Model 1: Random Forest Regressor

As the starting point, the cleaned and pre-processed data is split into training and test datasets and Random Forest Regressor algorithm is applied to train the model to predict flight fares. The model performance is evaluated using metrics- Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and R-squared (R2). The output shows the model’s performance on the test set:

* Mean Squared Error (MSE): 15020.32
* Root Mean Squared Error (RMSE): 122.55
* Mean Absolute Error (MAE): 79.10
* R-squared (R2): 0.66

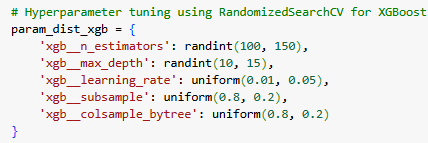
The model shows a moderate level of performance with an RMSE of 122.56 and an MAE of 79.10, indicating average prediction errors. The R² value of 0.66 suggests that the model explains about 66.24% of the variance in flight fares.

### Model 2: XG Boost Regressor

#### XGB Regressor with Hyperparameter Tuning

Since R2 score for RandomForest model was observed to be moderate, model training with XGBoost regressor is then implemented with hyperparameter tuning, using a parameter distribution including ranges for the number of estimators, maximum depth, learning rate, subsample, and column sample by tree.

A RandomizedSearchCV is then employed to perform a randomized search over these parameters, optimizing for the negative mean squared error. The best parameters found through this search indicate the optimal configuration for the XGBoost model.



Best parameters for XGBoost: {'xgb\_\_colsample\_bytree': 0.9202230023486418, 'xgb\_\_learning\_rate': 0.045403628889802275, 'xgb\_\_max\_depth': 14, 'xgb\_\_n\_estimators': 101, 'xgb\_\_subsample': 0.944399754453365}

#### XGB Regressor with best Parameters & Cross Validation

We retrained the XGBRegressor model using the best parameters obtained earlier. Cross-validation was performed with 5 folds, evaluating the model based on negative mean squared error (MSE). The model was then fitted to the training data, and predictions were made using the test data. Evaluation metrics were calculated and compared with the cross-validation mean.

* The model shows good performance with a training RMSE of 89.65 and test RMSE of 105.93, indicating it generalizes well.
* The R² scores (0.82 for training and 0.75 for test) suggest it explains a significant portion of the variance.
* Cross-validation confirms stability with a mean RMSE of 108.62 and low standard deviation.

## Model Improvement

### Handling Outliers / Skews

To validate outlier of the target variable total fare, skew function was implemented in the total fare which resulted as 2.25. This indicates a right-skewed distribution, meaning the data has a long tail on the right side with most values concentrated on the left.

Next, outliers were calculated using the interquartile range (IQR) by finding the difference between the 1st quartile (25%) and the 3rd quartile (75%). Using the lower and upper bounds of the IQR, filtered the DataFrame to find rows where total Fare was outside these bounds.

Finally, outliers were capped using the clip method, setting any value above the upper bound to the upper bound value. After capping the outliers, the skewness of total Fare was improved to 0.623, indicating a distribution closer to normal.

### XGB retraining and cross-validation

After handling outliers, data is retrained with XGBoost model for predicting flight fares. The model is configured with the best hyperparameters derived from the previous model training and performed the cross validation to evaluate the model performance.

* Mean Squared Error (MSE): 8307.88
* Mean Absolute Error (MAE): 67.37
* R-squared (R2): 0.7634
* Mean Cross-Validation MSE: 8386.32

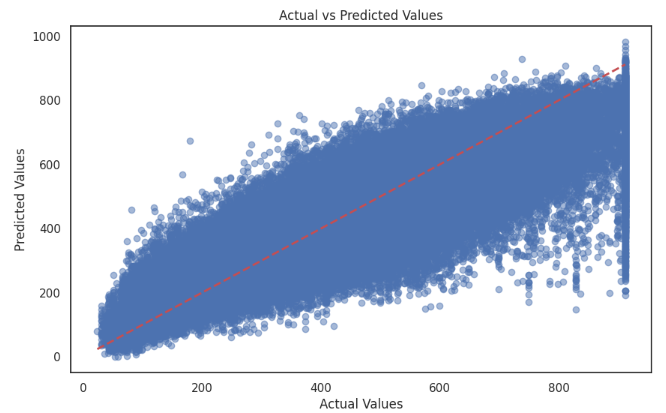
## Comparison of Model 1 and Model 2

* **Error Metrics:** Model 2 has a significantly lower MSE (8307.88 vs. 15020.32) and MAE (67.37 vs. 79.10) compared to Model 1. This indicates that Model 2’s predictions are generally closer to the actual values.
* **R-squared (R2):** The R² value for Model 2 (0.7635) is higher than that of Model 1 (0.6624). This suggests that Model 2 explains a greater proportion of the variance in the dependent variable, indicating a better fit.
* **Cross-Validation MSE Scores:** The cross-validation MSE scores for Model 2 are quite consistent, with a mean of 8386.32. This consistency suggests that Model 2’s model is robust and performs reliably across different subsets of the data.
* Overall, Model 2 outperforms Model 1 across all key metrics, making it the better model for predicting flight delays. The lower error metrics and higher R² value indicate more accurate and reliable predictions.

## Residuals Analysis

Residuals are the differences between the observed values and the predicted values which helps to understand how well the model is performing. Computed the difference between y test and predicted values to check residual analysis. Following is the residual analysis summary:

* **Mean:** 0.11 (close to 0, indicating unbiased predictions)
* **Standard Deviation:** 91.14 (indicates spread of residuals around the mean)
* **MSE:** 8307.88 (matching with the overall MSE reported earlier)
* **MAE:** 67.37 (average absolute difference between observed and predicted values)
* **R²:** 0.763 (76% of variance in observed data explained by the model)
* The residuals analysis shows that the model performs reasonably well, with a mean residual close to zero and a high R-squared value.

A graph of a normal distribution

Description automatically generated

## A screenshot of a computer Description automatically generated

## Summary

The analysis compares two models for predicting flight fares. The initial XGBoost model had better performance than the Random Forest regressor. After outlier handling, XGBoost's performance further improved. The final XGBoost model showed good performance with an RMSE of 91.58, an R-squared of 0.76, and consistent cross-validation scores. The model was considered reliable in predicting flight fares within a reasonable range of error.