Final Project

Flight Price Prediction

Authors of the Project:
Nivedita Gadade
Rishank Karkera
Preet Chaudhari
Atharv Mhatre

Introduction & Proposal:

The motivation for this project arises from the dynamic nature of flight pricing, which can significantly impact consumers' travel costs. Our goal is to develop a predictive model that accurately forecasts the price of flight tickets based on various factors such as days before the flight, class of the ticket, departure time, airline choice, and more. By analyzing historical data, we aim to uncover patterns and insights that will drive the prediction model's accuracy.

Methodology

Data Processing

Data processing encompassed several critical steps:

- Data Collection: The data was sourced from a CSV file containing historical flight price information.
- Data Exploration: Initial exploration involved understanding the structure, columns, and type of data using pandas' functions like .head(), .columns, .shape, .dtypes, and .describe().
- Data Cleaning: This step involved removing unnecessary columns, handling missing values, and ensuring data quality.
- Data Transformation: Label encoding was applied to convert categorical variables into a numeric format suitable for machine learning algorithms.
- Feature Selection: Utilized SelectKBest to identify the most significant features for predicting flight prices.
- Data Scaling: MinMaxScaler was used to normalize the feature values to a range that enhances the performance of gradient-based optimization algorithms.
- Data Preprocessing: A final step of data preparation before model training, ensuring that the data fed into the models is clean, transformed, and appropriately scaled.

Network Structure

We explored several machine learning models to predict flight prices. Given the structured nature of our dataset, we employed traditional regression models like Linear Regression, Decision Tree Regressor, and ensemble methods like Random Forest and XGBRegressor.

- Linear Regression: Serves as a baseline model, offering a fundamental approach to understanding the linear relationship between the independent variables and the flight price.
- Decision Tree Regressor: Provides a more flexible approach by capturing non-linear patterns.
- Random Forest Regressor: An ensemble of decision trees to improve the prediction and prevent overfitting.
- XGBRegressor: An advanced gradient boosting framework that uses boosting techniques for more accurate predictions.

Training & Validation Process

The dataset was split into training and test sets, with 70% of the data used for training and 30% for validation. Each model was trained using the training set, and hyperparameter tuning was conducted where applicable to optimize performance. The validation process involved evaluating the models against the test set to assess their predictive capabilities.

Evaluation & Results

Training & Validation Results

Each model's performance was evaluated using metrics such as Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and the R-squared value.
Performance Comparison to Baselines
The Linear Regression model acted as a baseline, and its performance was compared with more complex models. Decision Tree and Random Forest models provided deeper insight due to their ability to model non-linear relationships.

Performance Comparison to Baselines

The performance of each model was compared using a set of metrics:

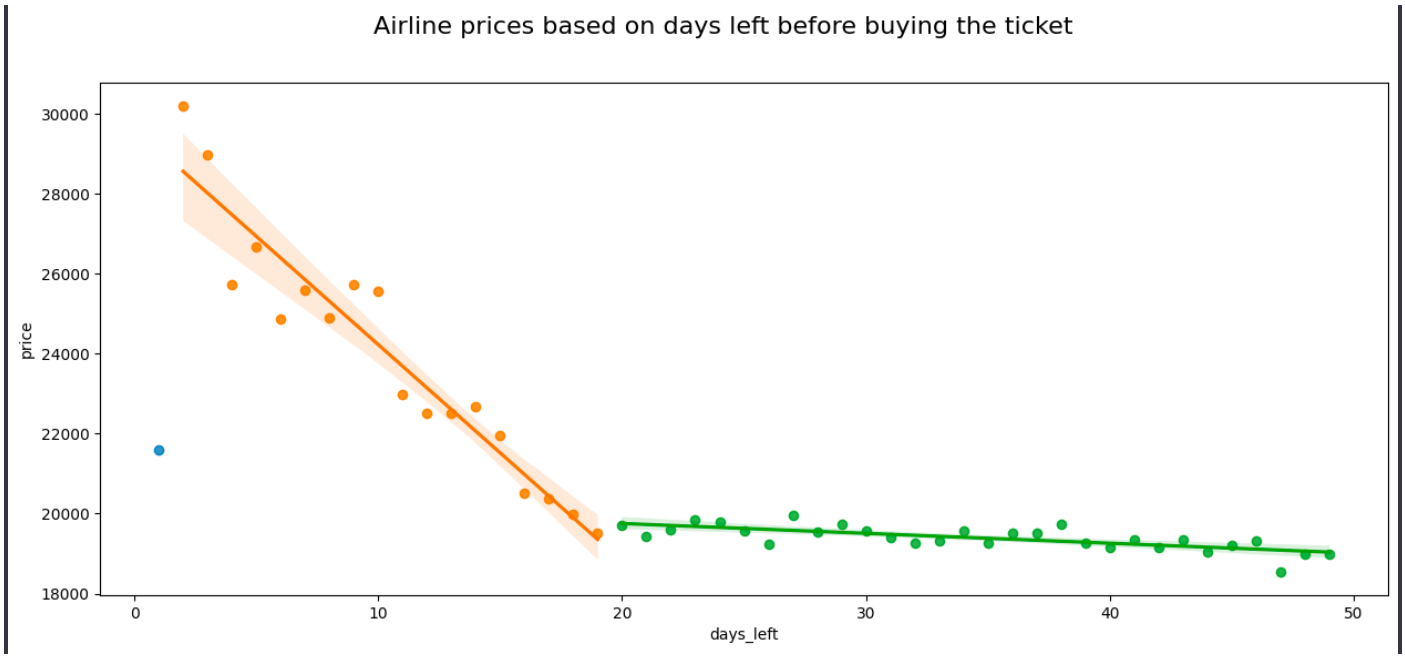
- Linear Regression: As the baseline model, its performance was measured to set a standard for comparison.
- Decision Tree Regressor: The performance was compared against the baseline to evaluate its capability to model non-linear relationships.

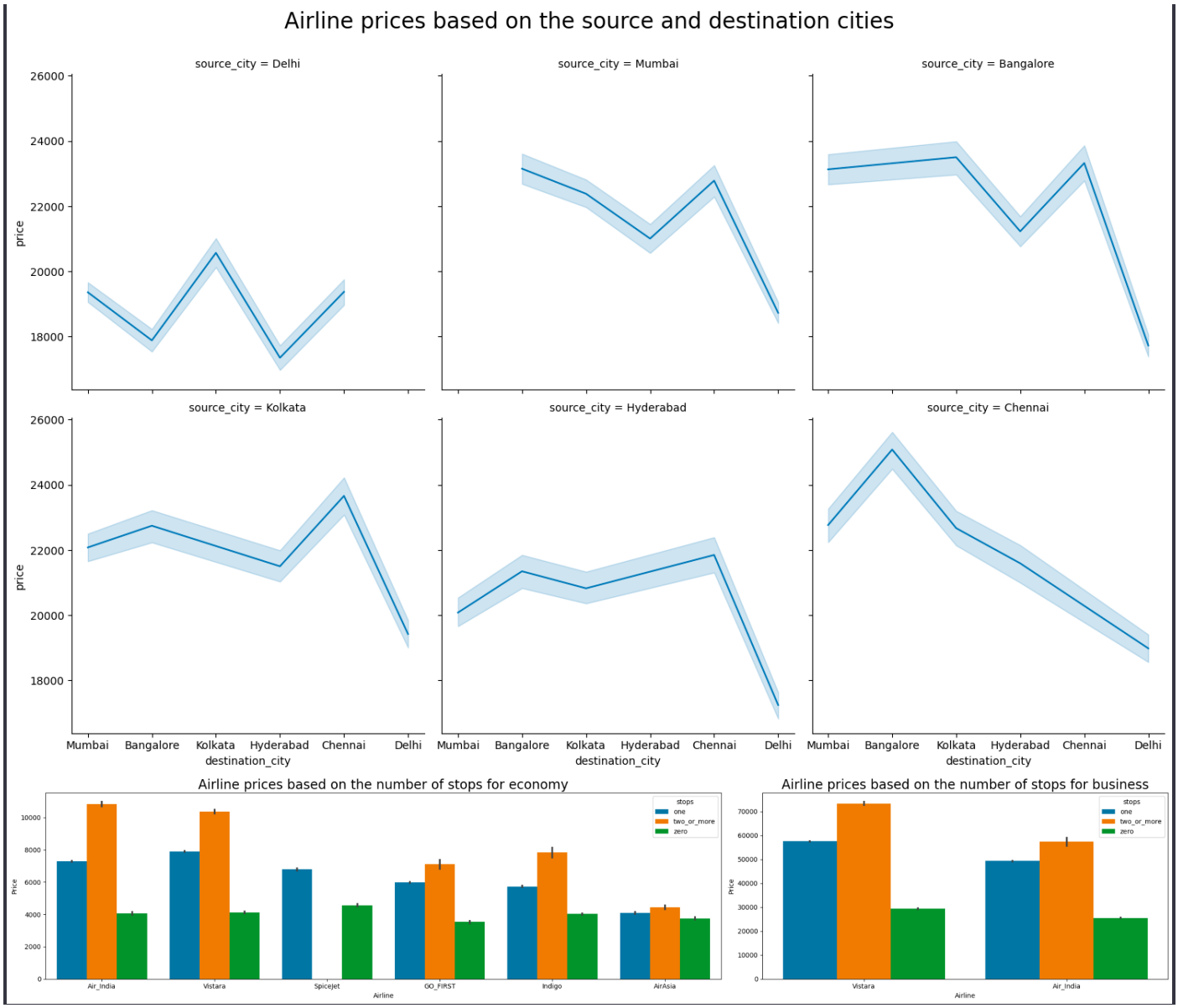
- Random Forest Regressor: Being an ensemble model, its performance typically surpasses that of a single decision tree, and its results were contrasted against the baseline.
- XGBRegressor: As a more sophisticated model, it often outperforms simpler models. Its performance was analyzed to see if the complexity of the model translated into better prediction accuracy.
- Ensemble Technique: The average predictions from all models were also evaluated to determine if combining predictions leads to better performance.

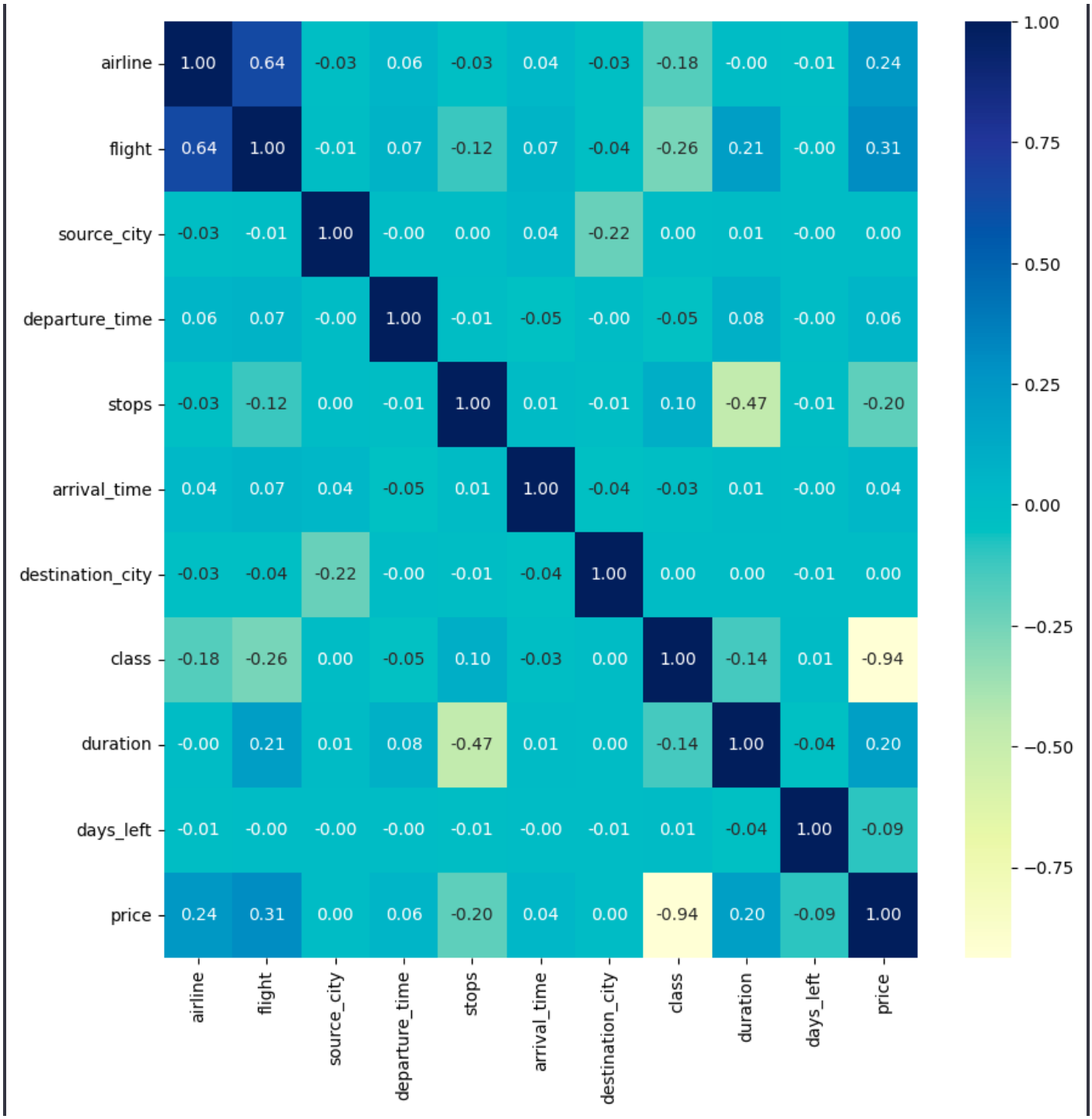
Hyperparameter Tuning

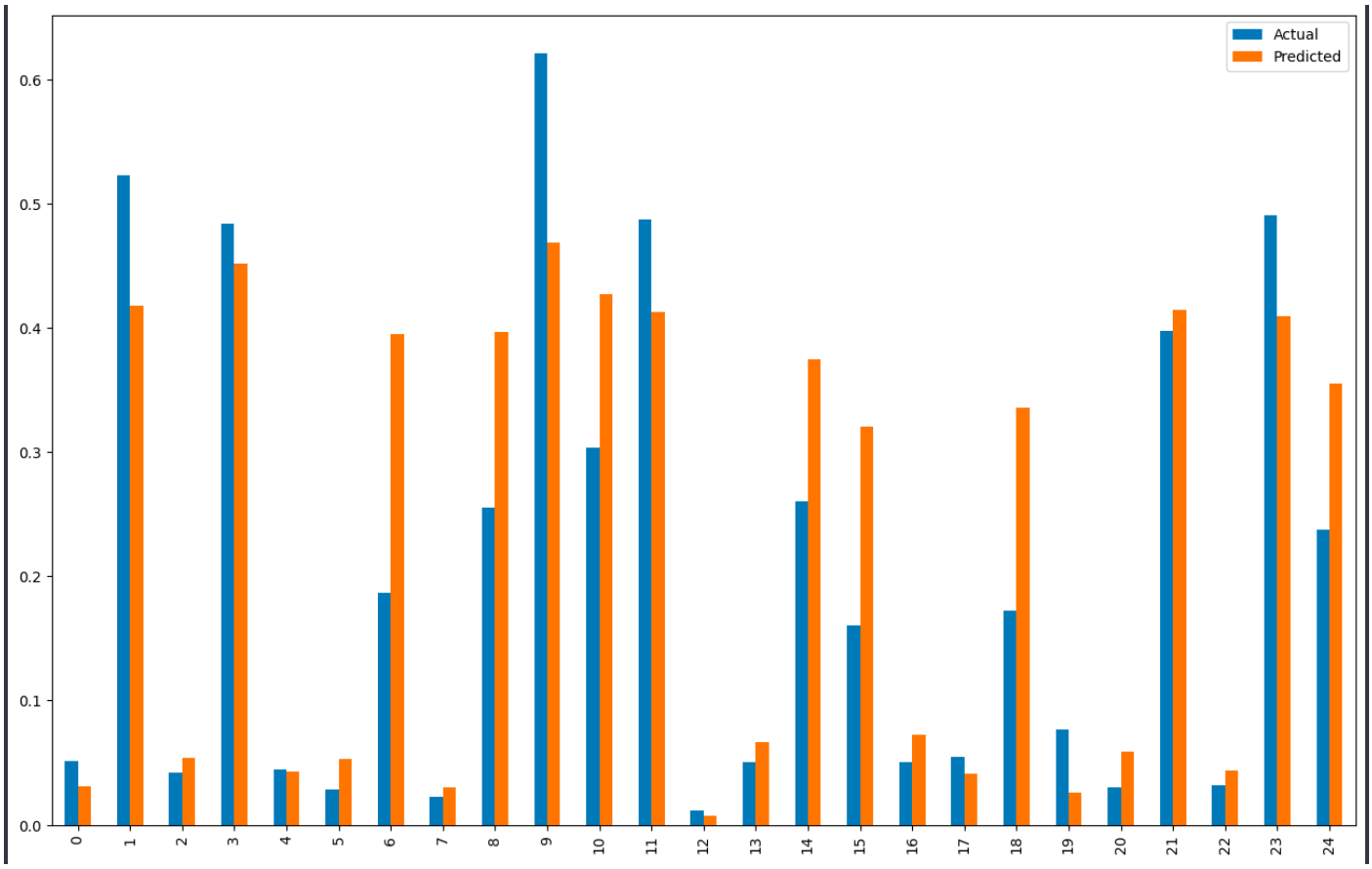
For models like Random Forest and XGBRegressor, hyperparameters such as the number of trees (n_estimators), max_depth, and learning rate were fine-tuned to achieve better performance.

Some Plots from the Code:




Airline prices based on the source and destination cities
source_city = Delhi
source_city = Mumbai
source_city = Bangalore
source_city = Kolkata
source_city = Hyderabad
source_city = Chennai
price
destination_city
Mumbai
Bangalore
Kolkata
Hyderabad
Chennai
Delhi
Airline prices based on the number of stops for economy
Airline prices based on the number of stops for business
stops
one
two_or_more
zero
Price
Airline
Air_India
Vistara
SpiceJet
GO_FIRST
Indigo
AirAsia


airline 1.00 0.64 -0.03 0.06 -0.03 0.04 -0.03 -0.18 -0.00 -0.01 0.24
flight 0.64 1.00 -0.01 0.07 -0.12 0.07 -0.04 -0.26 0.21 -0.00 0.31
source_city -0.03 -0.01 1.00 -0.00 0.00 0.04 -0.22 0.00 0.01 -0.00 0.00
departure_time 0.06 0.07 -0.00 1.00 -0.01 -0.05 -0.00 -0.05 0.08 -0.00 0.06
stops -0.03 -0.12 0.00 -0.01 1.00 0.01 -0.01 0.10 -0.47 -0.01 -0.20
arrival_time 0.04 0.07 0.04 -0.05 0.01 1.00 -0.04 -0.03 0.01 -0.00 0.04
destination_city -0.03 -0.04 -0.22 -0.00 -0.01 -0.04 1.00 0.00 0.00 -0.01 0.00
class -0.18 -0.26 0.00 -0.05 0.10 -0.03 0.00 1.00 -0.14 0.01 -0.94
duration -0.00 0.21 0.01 0.08 -0.47 0.01 0.00 -0.14 1.00 -0.04 0.20
days_left -0.01 -0.00 -0.00 -0.00 -0.01 -0.00 -0.01 0.01 -0.04 1.00 -0.09
price 0.24 0.31 0.00 0.06 -0.20 0.04 0.00 -0.94 0.20 -0.09 1.00
airline
flight
source_city
departure_time
stops
arrival_time
destination_city
class
duration
days_left
price
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75



Conclusion

The project demonstrated the feasibility of using machine learning models to predict flight prices with reasonable accuracy. Ensemble methods, especially XGBRegressor, showed promising results by effectively capturing the complex relationships within the data. Further improvements could include more sophisticated feature engineering, additional data, and exploring deep learning methods for potentially better results.