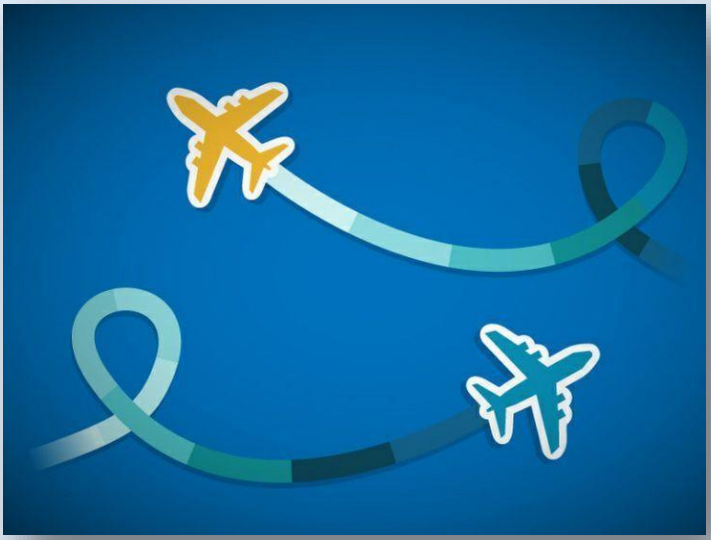
FLIGHT FARE PREDICTION



TEAM I.D : PTID-CDS-APR-24-1887

PROJECT I.D : PRCP-1025- FLIGHT FARE PREDICTION

ANIKET GOSWAMI

INDEX

INTRODUCTION

In today's dynamic aviation industry, predicting flight ticket prices accurately is crucial for both airlines and travelers alike. The fluctuating nature of ticket prices often leaves travelers puzzled and airlines struggling to optimize their revenue management strategies. Leveraging machine learning techniques can provide a solution to this problem by analyzing historical data and predicting future ticket prices.

The dataset provided contains essential attributes such as airline, date of journey, source, destination, route, arrival time, duration, total stops, additional information, and the ticket price. Each attribute contributes valuable insights into the factors influencing ticket prices.

In this project, we aim to develop a machine learning model that can predict flight ticket prices based on these attributes. By analyzing past trends and patterns, the model will learn to make accurate predictions, enabling airlines to optimize pricing strategies and providing travelers with better insights into ticket pricing dynamics.

This project, we will explore various machine learning algorithms, preprocess the data, perform feature engineering, and evaluate the model's performance. Ultimately, our goal is to develop a robust and reliable predictive model that can help both airlines and travelers navigate the complexities of flight ticket pricing.

METHODOLOGY

Data Collection: Describe how the data was gathered, including the APIs or databases accessed (e.g., flight data from airlines, prices from travel agencies).

Data Preprocessing: Detail the steps taken to clean and prepare the data for analysis, such as handling missing values, encoding categorical variables, normalizing/standardizing data.

Feature Engineering: Explain the creation of new features that could help in improving the model's accuracy, such as time of day, day of the week, seasonality, and holidays.

Model Selection: Discuss the rationale behind selecting specific machine learning models (e.g., linear regression, random forests, gradient boosting machines, neural networks).

Model Training: Outline how the models were trained, including splitting the data into training and testing sets, choosing hyperparameters, and cross-validation methods used.

DATA PREPROCESSING

Data preprocessing is a critical step in data analysis and machine learning, involving the transformation of raw data into a format that is more suitable for analysis or modeling. This process typically includes tasks such as cleaning data to handle missing or erroneous values, scaling features to ensure they are on a similar magnitude, encoding categorical variables into a numerical format, and splitting the data into training and testing sets. Effective data preprocessing lays the foundation for accurate and reliable analysis and modeling, enhancing the quality and interpretability of results.

The data pre-processing for our dataset has been given below:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10683 entries, 0 to 10682
Data columns (total 11 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Airline          10683 non-null  object
 1   Date_of_Journey  10683 non-null  object
 2   Source           10683 non-null  object
 3   Destination      10683 non-null  object
 4   Route            10682 non-null  object
 5   Dep_Time         10683 non-null  object
 6   Arrival_Time     10683 non-null  object
 7   Duration         10683 non-null  object
 8   Total_Stops      10682 non-null  object
 9   Additional_Info  10683 non-null  object
 10  Price            10683 non-null  int64
dtypes: int64(1), object(10)
memory usage: 918.2+ KB
```

```python
data.describe()
```

	Price
count	10683.000000
mean	9087.064121
std	4611.359167
min	1759.000000
25%	5277.000000
50%	8372.000000
75%	12373.000000
max	79512.000000

```python
#Checking for the missing values
data.isnull().sum()
```

```
Airline            0
Date_of_Journey    0
Source             0
Destination        0
Route              1
Dep_Time           0
Arrival_Time       0
Duration           0
Total_Stops        1
Additional_Info    0
Price              0
dtype: int64
```

```python
# Impute Missing Values
data.dropna(inplace = True)
# Validate Imputation
data.isnull().sum()
```

```
Airline            0
Date_of_Journey    0
Source             0
Destination        0
Route              0
Dep_Time           0
Arrival_Time       0
Duration           0
Total_Stops        0
Additional_Info    0
Price              0
dtype: int64
```

EXPLORATORY DATA ANALYSIS

Exploratory Data Analysis (EDA) is a fundamental approach in data science aimed at understanding the characteristics of a dataset before diving into formal modeling. It involves visualizing and summarizing data to uncover patterns, trends, anomalies, and relationships between variables. Techniques such as summary statistics, histograms, scatter plots, box plots, and correlation matrices are commonly used in EDA to gain insights into the distribution, central tendency, dispersion, and dependencies within the data. EDA not only helps in formulating hypotheses for further analysis but also aids in identifying data preprocessing needs and selecting appropriate modeling techniques. By providing a comprehensive overview of the dataset, EDA enables data scientists to make informed decisions and derive meaningful interpretations from their data.

The following EDA has been done on our project:

```
#converting data_of Journey to datetime format
data["Journey_day"] = pd.to_datetime(data.Date_of_Journey, format="%d/%m/%Y").dt.day
```

```
data["Journey_month"] = pd.to_datetime(data["Date_of_Journey"], format = "%d/%m/%Y").dt.month
```

```
data.head()
```

	Airline	Date_of_Journey	Source	Destination	Route	Dep_Time	Arrival_Time	Duration	Total_Stops	Additional_Info	Price	Journey_day	Journey_month
0	IndiGo	24/03/2019	Banglore	New Delhi	BLR ? DEL	22:20	01:10 22 Mar	2h 50m	non-stop	No info	3897	24	3
1	Air India	1/05/2019	Kolkata	Banglore	CCU ? IXR ? BBI ? BLR	05:50	13:15	7h 25m	2 stops	No info	7662	1	5
2	Jet Airways	9/06/2019	Delhi	Cochin	DEL ? LKO ? BOM ? COK	09:25	04:25 10 Jun	19h	2 stops	No info	13882	9	6
3	IndiGo	12/05/2019	Kolkata	Banglore	CCU ? NAG ? BLR	18:05	23:30	5h 25m	1 stop	No info	6218	12	5

```python
#converting Dep_Time
data["Dep_hour"] = pd.to_datetime(data["Dep_Time"]).dt.hour
data["Dep_min"] = pd.to_datetime(data["Dep_Time"]).dt.minute

# we can drop Dep_Time because it no longer needed
data.drop(["Dep_Time"], axis = 1, inplace = True)
```

```
C:\Users\HP\AppData\Local\Temp\ipykernel_21416\2482536006.py:2: UserWarning: Could not infer format, so each element will be parsed individually, falling
back to `dateutil`. To ensure parsing is consistent and as-expected, please specify a format.
  data["Dep_hour"] = pd.to_datetime(data["Dep_Time"]).dt.hour
C:\Users\HP\AppData\Local\Temp\ipykernel_21416\2482536006.py:3: UserWarning: Could not infer format, so each element will be parsed individually, falling
back to `dateutil`. To ensure parsing is consistent and as-expected, please specify a format.
  data["Dep_min"] = pd.to_datetime(data["Dep_Time"]).dt.minute
```

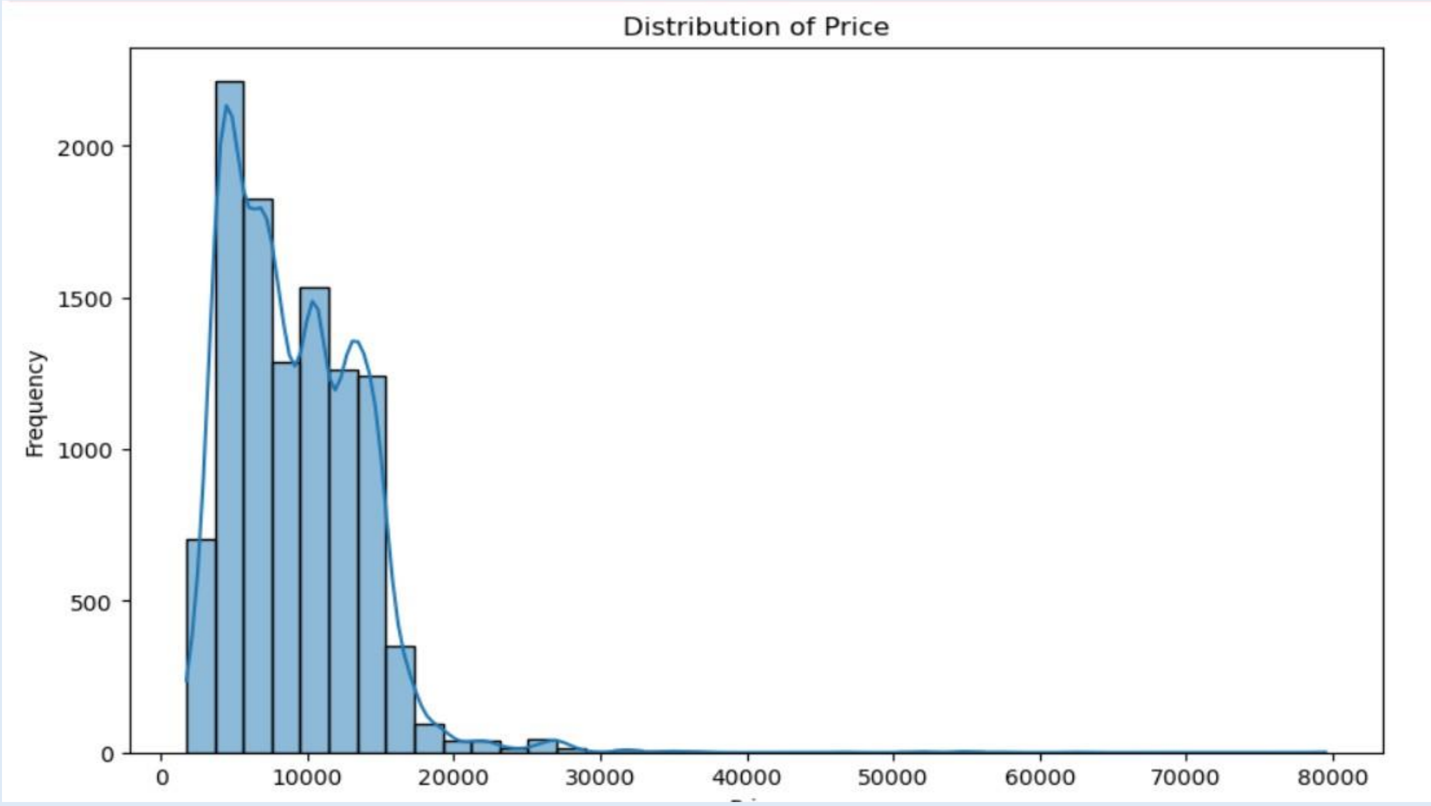
```python
data.head()
```

	Airline	Source	Destination	Route	Arrival_Time	Duration	Total_Stops	Additional_Info	Price	Journey_day	Journey_month	Dep_hour	Dep_min
0	IndiGo	Banglore	New Delhi	BLR ? DEL	01:10 22 Mar	2h 50m	non-stop	No info	3897	24	3	22	20
1	Air India	Kolkata	Banglore	CCU ? IXR ? BBI ? BLR	13:15	7h 25m	2 stops	No info	7662	1	5	5	50
2	Jet Airways	Delhi	Cochin	DEL ? LKO ? BOM ? COK	04:25 10 Jun	19h	2 stops	No info	13882	9	6	9	25
3	IndiGo	Kolkata	Banglore	CCU ? NAG ? BLR	23:30	5h 25m	1 stop	No info	6218	12	5	18	5

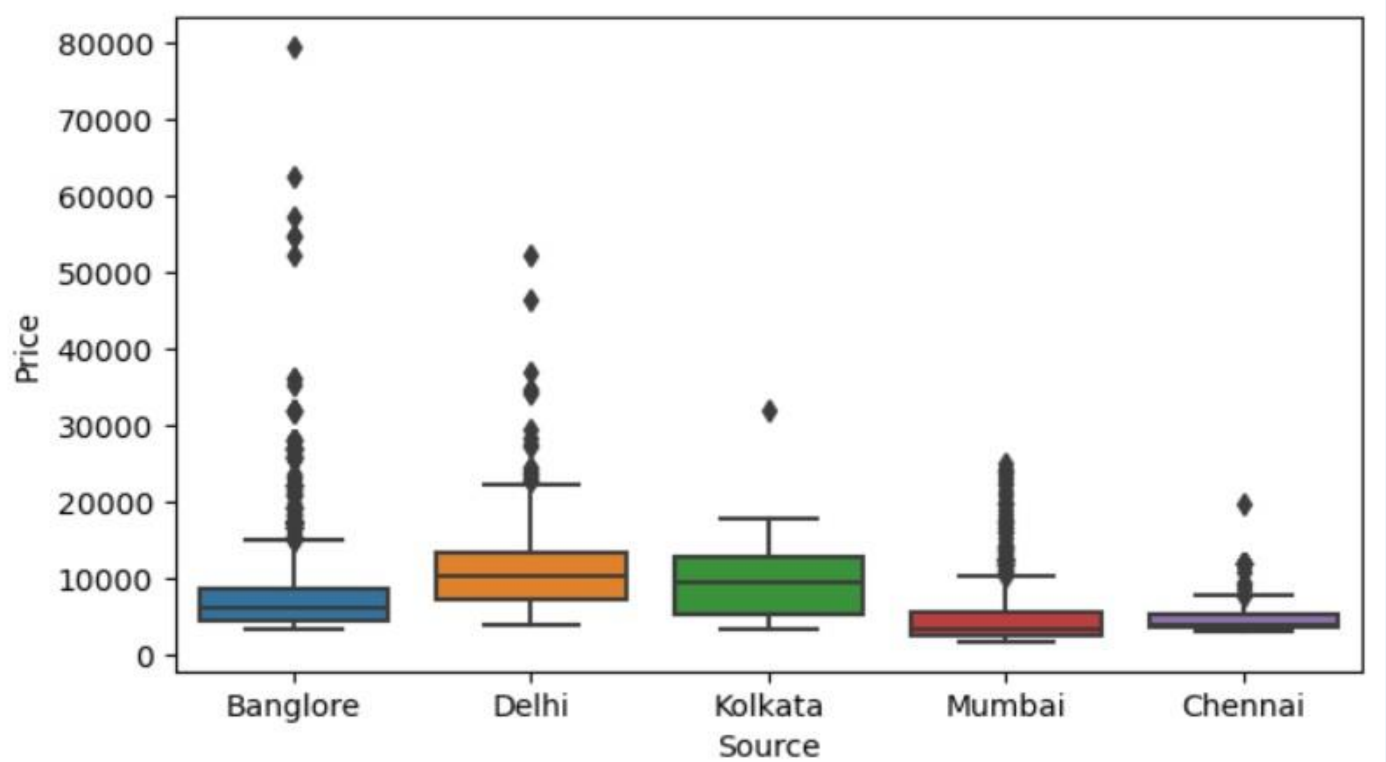
Destination	Route	Total_Stops	Additional_Info	Price	Journey_day	Journey_month	Dep_hour	Dep_min	Arrival_hour	Arrival_min	Duration_hours	Dura
New Delhi	BLR ? DEL	non-stop	No info	3897	24	3	22	20	1	10	2	
Banglore	CCU ? IXR ? BBI ? BLR	2 stops	No info	7662	1	5	5	50	13	15	7	
Cochin	DEL ? LKO ? BOM ? COK	2 stops	No info	13882	9	6	9	25	4	25	19	
Banglore	CCU ? NAG ? BLR	1 stop	No info	6218	12	5	18	5	23	30	5	
New Delhi	BLR ? NAG ? DEL	1 stop	No info	13302	1	3	16	50	21	35	4	

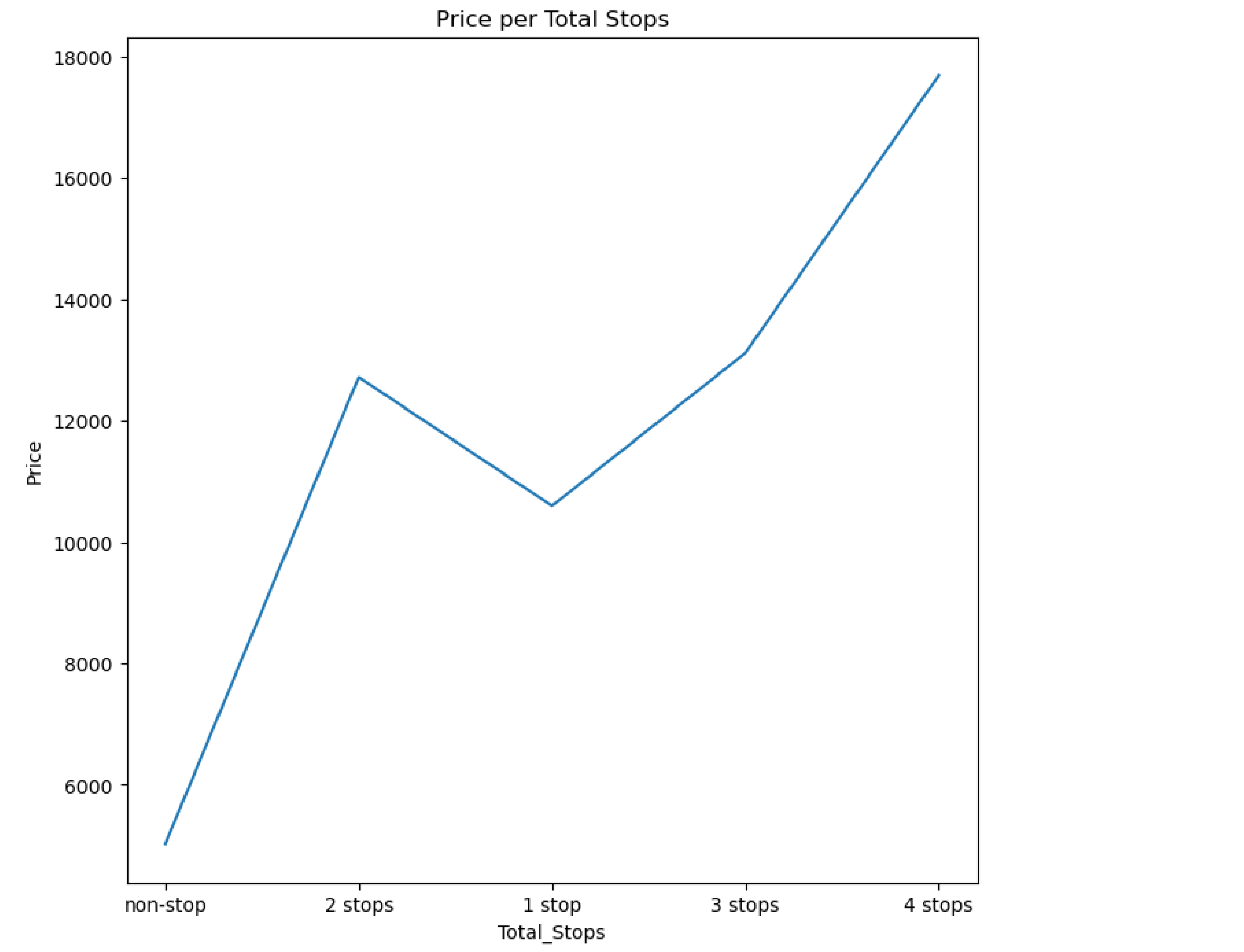
DATA VISUALIZATION

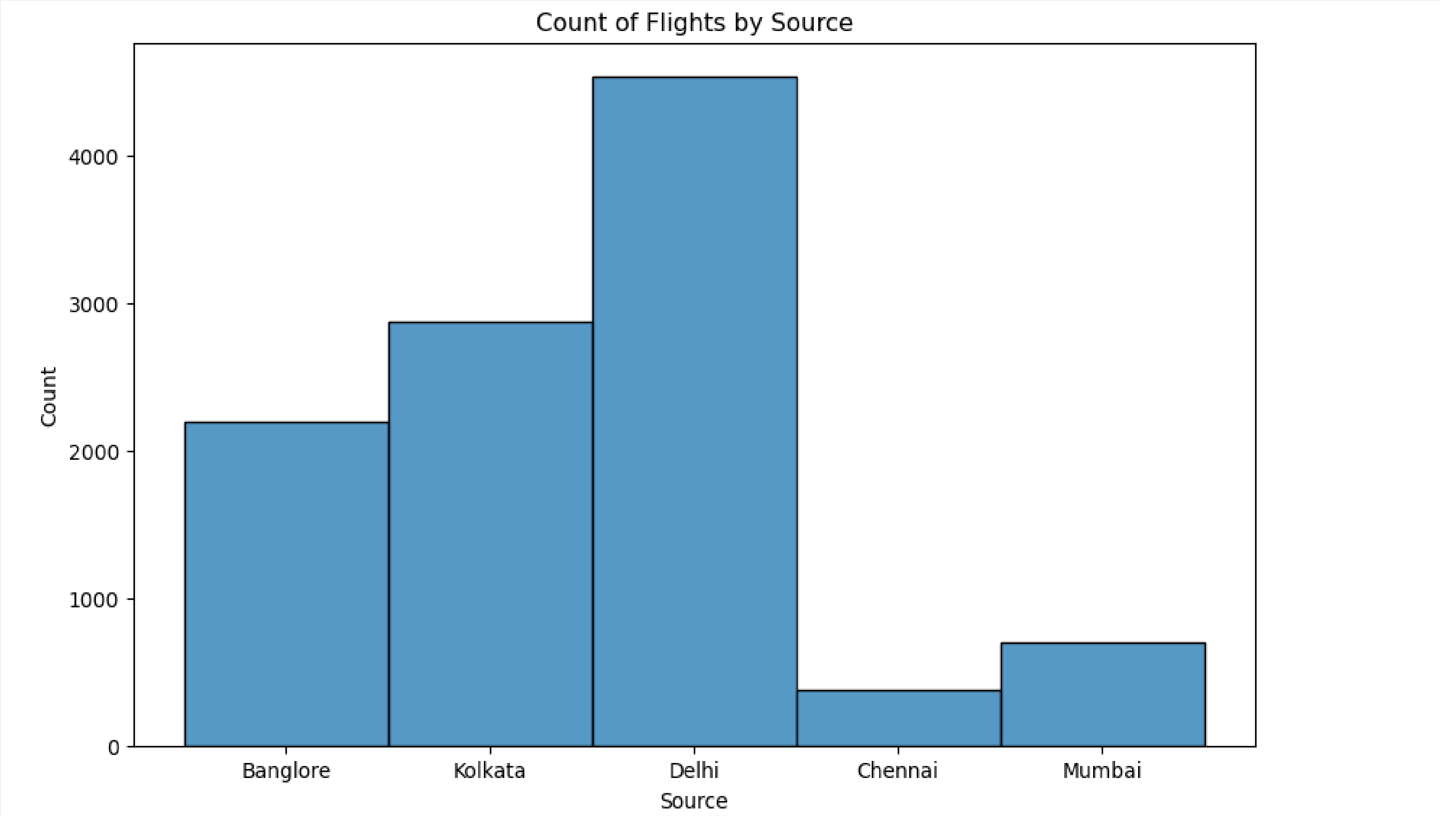
Data visualization is the process of presenting data in a graphical or visual format to aid in understanding patterns, trends, and relationships within the data. Through the use of charts, graphs, maps, and other visual elements, complex datasets can be simplified and communicated effectively to a wide audience. Data visualization allows for quick and intuitive interpretation of information, enabling decision-makers to identify insights, spot anomalies, and communicate findings more efficiently. By leveraging color, shape, size, and spatial arrangements, data visualization enhances comprehension and enables deeper exploration of data, facilitating better decision-making and driving actionable insights across various domains, from business analytics to scientific research.

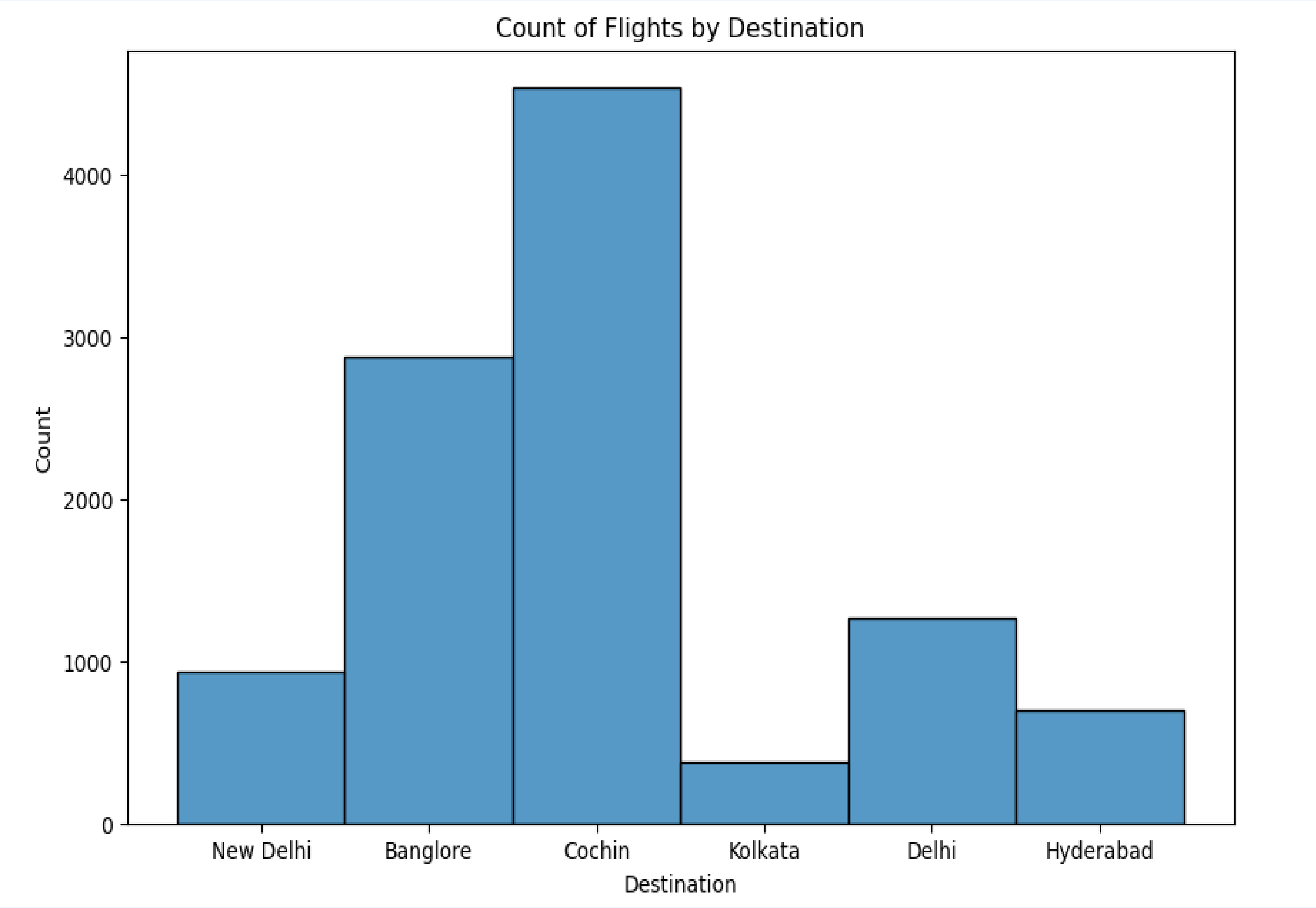


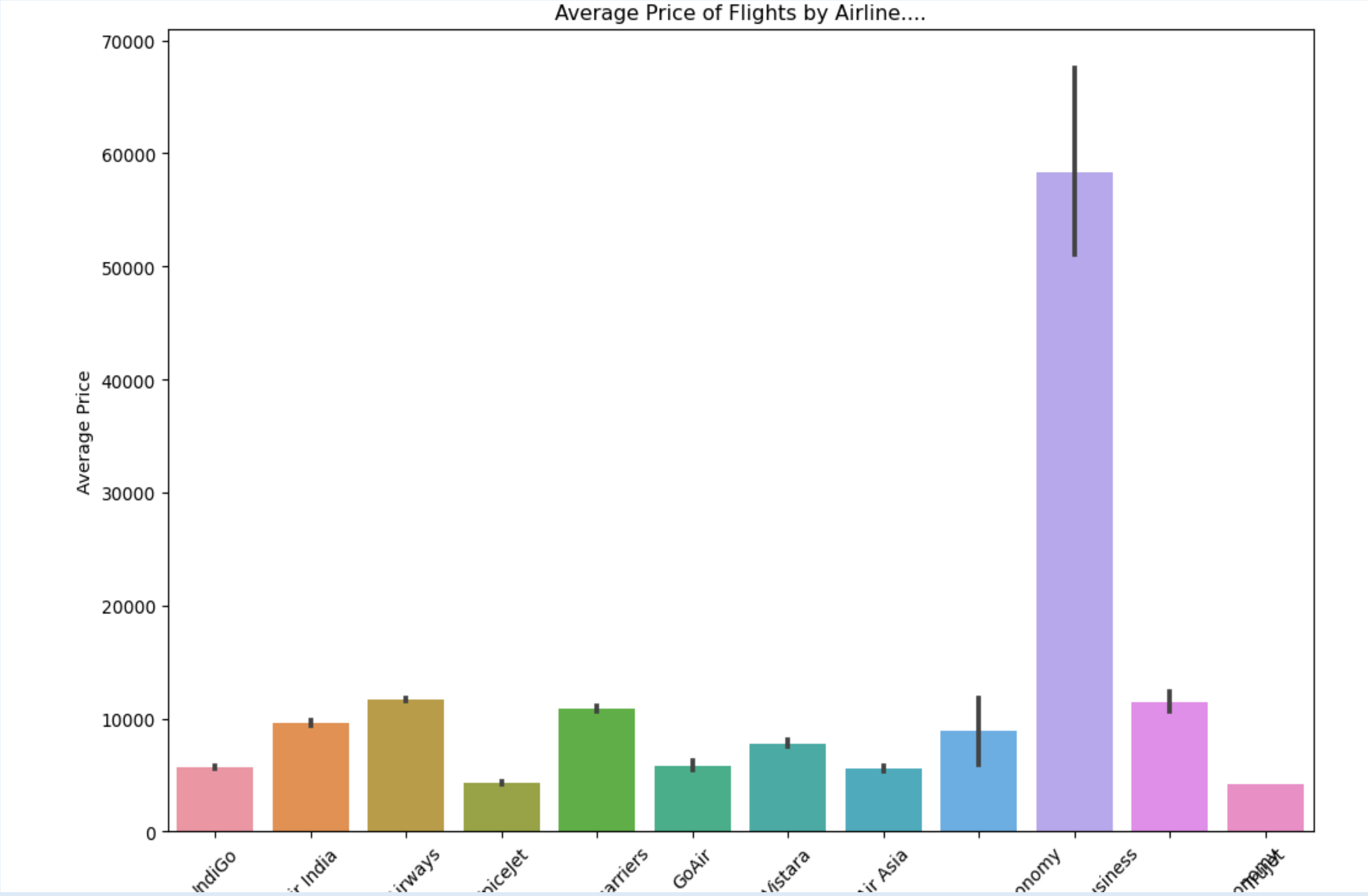
<Axes: xlabel='Source', ylabel='Price'>

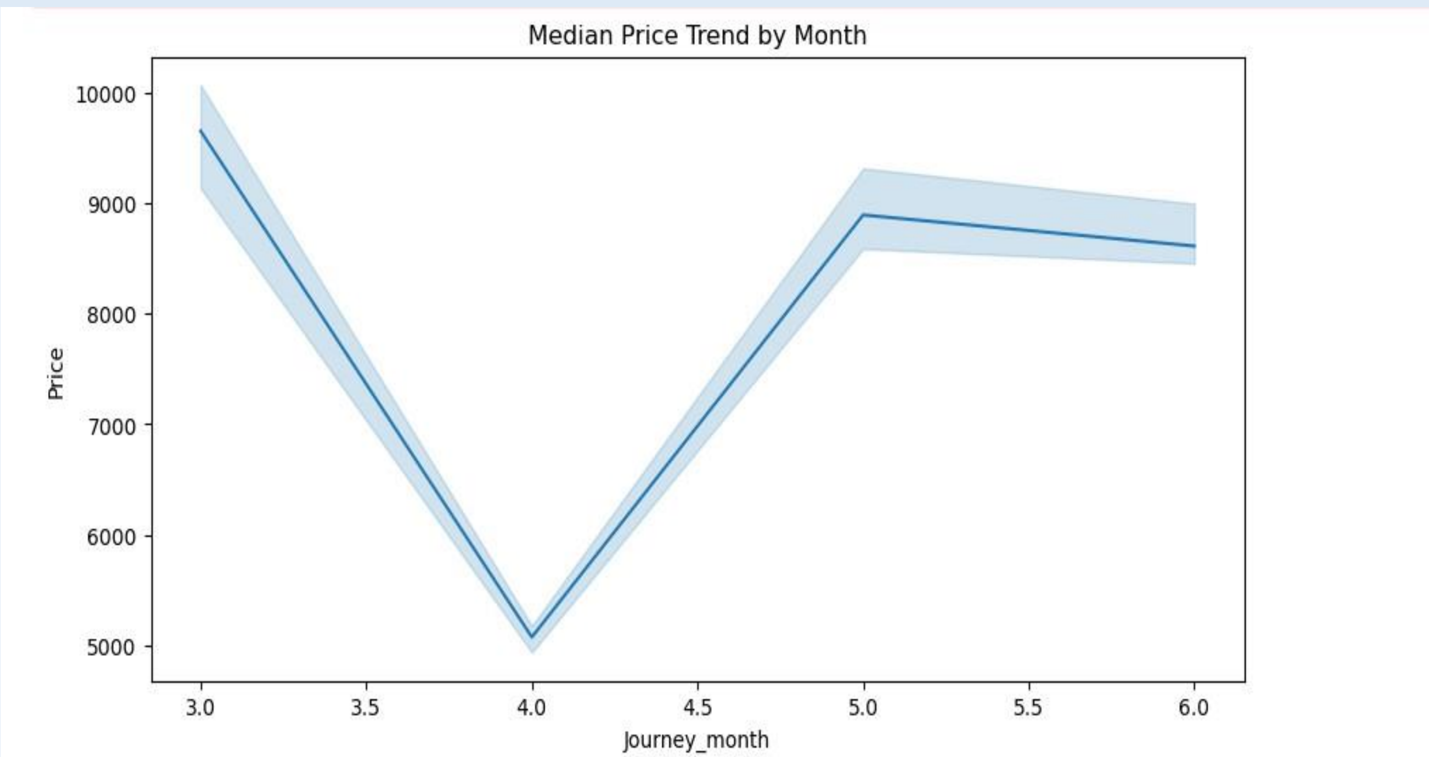





Count of Flights by Source
4000
3000
2000
1000
0
Count
Banglore
Kolkata
Delhi
Chennai
Mumbai
Source

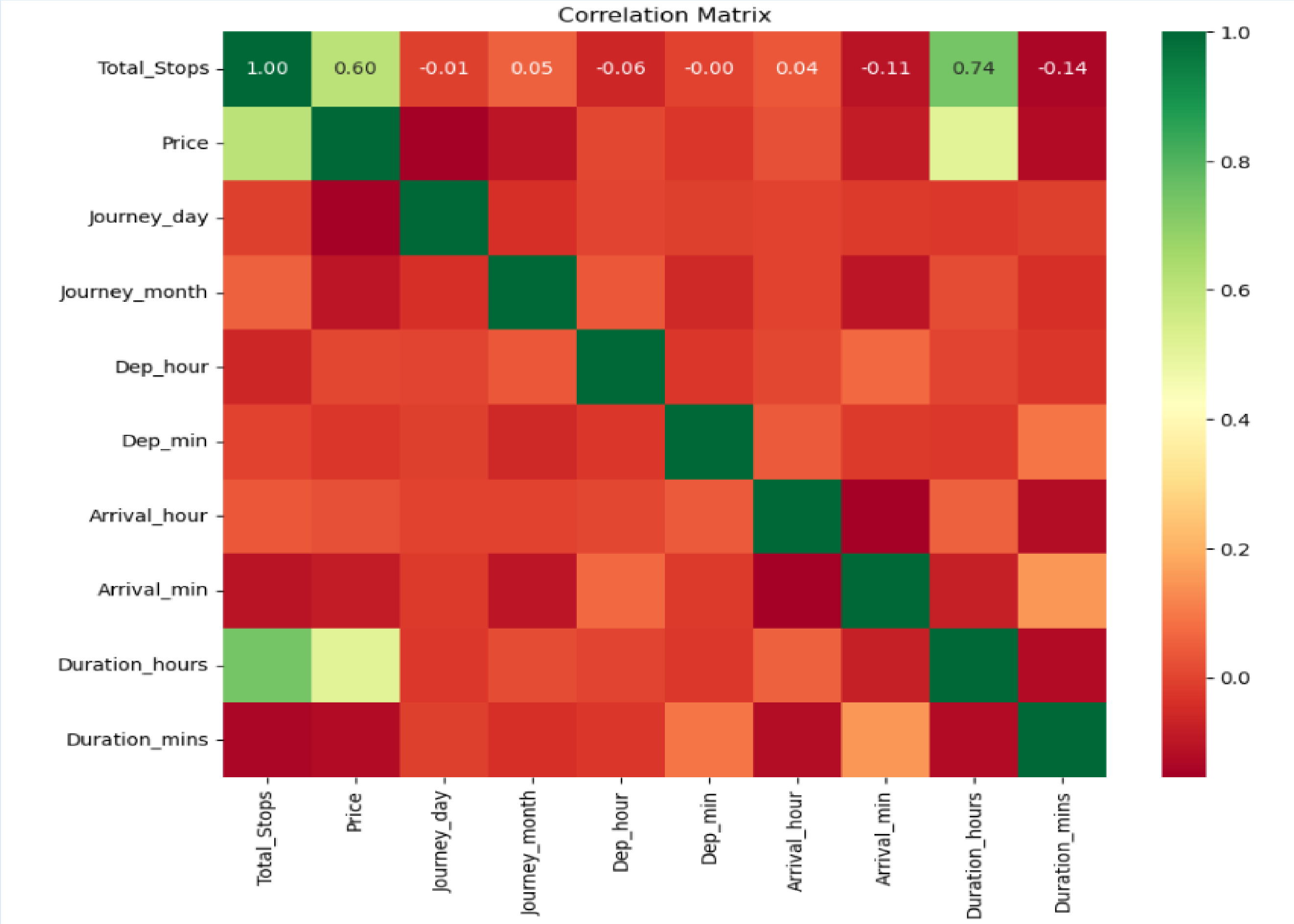

Count of Flights by Destination
4000
3000
2000
1000
0
Count
New Delhi
Banglore
Cochin
Kolkata
Delhi
Hyderabad
Destination


Average Price of Flights by Airline....
Average Price
70000
60000
50000
40000
30000
20000
10000
0
GoAir

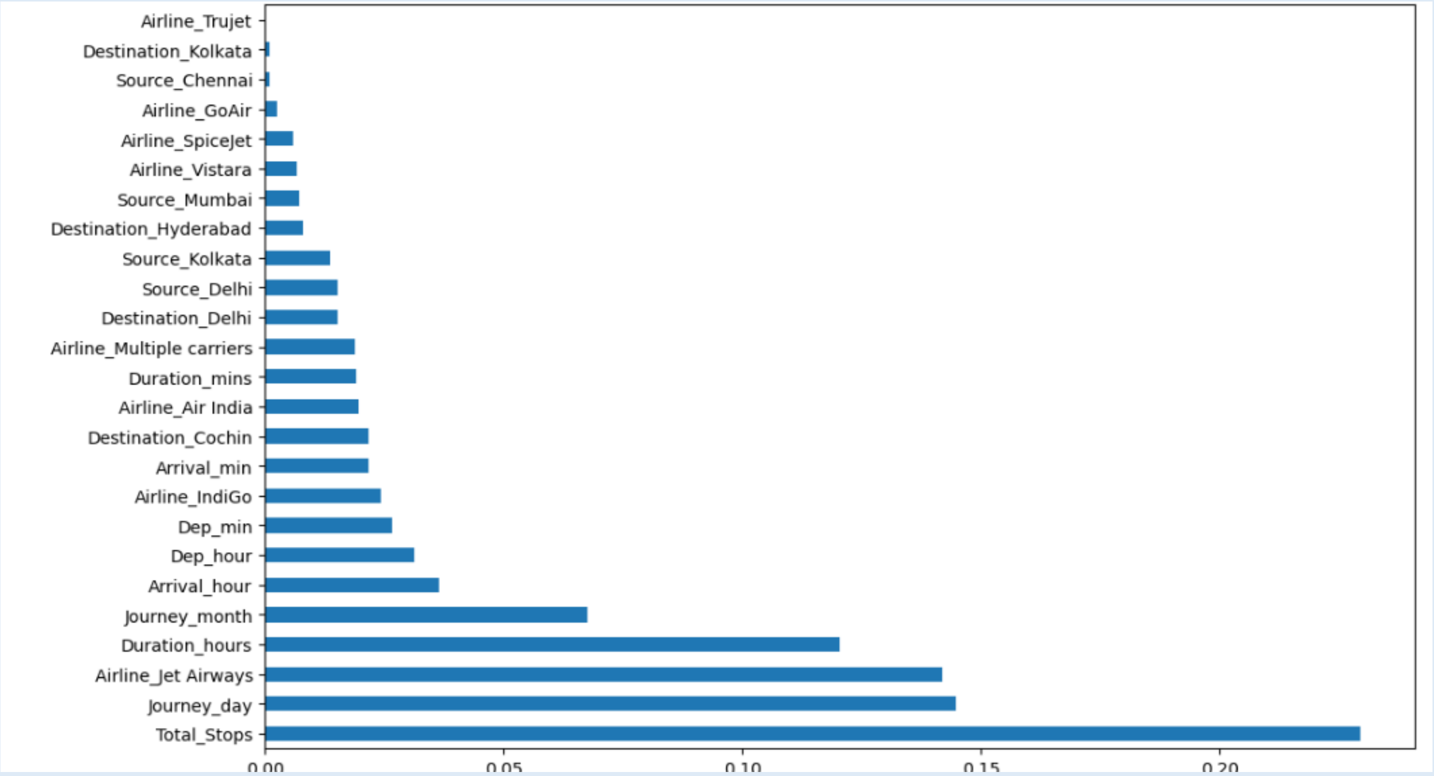

Median Price Trend by Month
Price
10000
9000
8000
7000
6000
5000
3.0
3.5
4.0
4.5
5.0
5.5
6.0
Journey_month

FEATURE SELECTION

Feature selection is a crucial process in machine learning that involves choosing the most relevant features from a dataset to improve model performance and efficiency. By selecting a subset of features that are most informative and eliminating irrelevant or redundant ones, feature selection helps mitigate the curse of dimensionality, reduces overfitting, and enhances model interpretability. Techniques for feature selection include filter methods, which evaluate features independently of the model, wrapper methods, which assess feature subsets based on model performance, and embedded methods, which incorporate feature selection within the model training process. Effective feature selection not only streamlines the modeling process but also enhances the generalization and predictive power of machine learning models, leading to more accurate and robust results.



Note : The colors indicates the most relevant features with Green being the most relevant and Red being the least.



	variables	VIF
0	Total_Stops	8.063589
1	Journey_day	3.499865
2	Journey_month	20.095332
3	Dep_hour	5.682536
4	Dep_min	2.773585
5	Arrival_hour	4.868434
6	Arrival_min	3.437438
7	Duration_hours	6.353001
8	Duration_mins	4.059314
9	Airline_Air India	4.631327
10	Airline_GoAir	1.368919
11	Airline_IndiGo	4.340418
12	Airline_Jet Airways	8.319886
13	Airline_Multiple carriers	3.302066
14	Airline_Trujet	1.003963
15	Airline_SpiceJet	2.371615
16	Airline_Vistara	1.915956
17	Source_Chennai	inf
18	Source_Delhi	inf
19	Source_Kolkata	5.071707
20	Source_Mumbai	inf
21	Destination_Cochin	inf
22	Destination_Delhi	3.108539

MODEL BUILDING

LINEAR REGRESSION MODEL:

Linear regression is a fundamental statistical method used for modeling the relationship between a dependent variable and one or more independent variables. It assumes a linear relationship between the independent variables and the dependent variable, which is represented by a straight line in a two-dimensional space or a plane in higher dimensions. The goal of linear regression is to find the best-fitting line or plane that minimizes the sum of the squared differences between the observed and predicted values. This is typically achieved using the method of least squares. Linear regression is widely used for predictive modeling and inference in various fields such as economics, finance, social sciences, and machine learning, owing to its simplicity, interpretability, and ease of implementation.

```
(2137,)

In [65]: # creating model

         from sklearn.linear_model import LinearRegression

         model = LinearRegression() # object creation
         model.fit(x_train, y_train) # training of linear regression

Out[65]: LinearRegression()
         In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
         On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [66]: # Model evaluation

         from sklearn.metrics import mean_squared_error, r2_score, mean_absolute_error

         def metrics(y_test, y_pred):
             print("r2_score:", r2_score(y_test,y_pred))
             print("MAE:", mean_absolute_error(y_test, y_pred))
             print("RMSE:", mean_squared_error(y_test, y_pred, squared=False))
         def accuracy(y_test, y_pred):
                 errors = abs(y_test - y_pred)
                 mape = 100 * np.mean(errors/y_test)
                 accuracy = 100 - mape
                 return accuracy

In [67]: y_pred= model.predict(x_test)

In [68]: metrics(y_test, y_pred)

         r2_score: 0.6195934071839777
         MAE: 0.42784599077554664
         RMSE: 0.6210729652329909

In [69]: accuracy(y_test, y_pred)

Out[69]: 69.8345346912416
```

RANDOM FOREST REGRESSOR MODEL:

The Random Forest Regressor is a versatile and powerful machine learning model that belongs to the ensemble learning family. It operates by constructing multiple decision trees during training and outputs the mean prediction of the individual trees as its final prediction. Each tree in the forest is trained on a random subset of the training data and uses a random subset of features, hence the term "random forest." This randomness helps to decorrelate the individual trees, reducing overfitting and improving generalization performance. Random forests are effective for both regression and classification tasks, and they excel in handling high-dimensional data with complex relationships. They are robust to outliers and missing values and require minimal hyperparameter tuning. Random Forest Regressors are widely used in various domains such as finance, healthcare, and ecology for their accuracy, scalability, and interpretability.

Random Forest

In [70]:
```
from sklearn.ensemble import RandomForestRegressor

model_rf = RandomForestRegressor(max_depth=12) #object creation
model_rf.fit(x_train,y_train)#training the data
```

Out[70]:
```
RandomForestRegressor(max_depth=12)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [71]:
```
y_pred_rf = model_rf.predict(x_test)
```

In [72]:
```
metrics(y_test, y_pred_rf)
```

```
r2_score: 0.8137017135496022
MAE: 0.25775249964285435
RMSE: 0.4346333405511875
```

GRADIENT BOOSTING MODEL:

The Gradient Boosting Regressor is a powerful machine learning model that builds an ensemble of weak learners, typically decision trees, in a sequential manner. Unlike traditional random forests, where trees are built independently, gradient boosting builds trees sequentially, with each tree attempting to correct the errors made by the previous ones. It works by fitting each new tree to the residual errors of the ensemble, gradually reducing the overall error. This iterative process continues until a predefined number of trees are built, or until the model converges. Gradient Boosting Regressors are highly effective for regression tasks, offering superior predictive performance and robustness against overfitting. They are widely used in various domains, including finance, healthcare, and marketing, where accurate predictions are crucial for decision-making.

Gradient Boosting

In [73]:
```
## importing the model library

from sklearn.ensemble import GradientBoostingRegressor
gbm=GradientBoostingRegressor(max_depth=10) ## object creation
gbm.fit(x_train,y_train) ## fitting the data
```

Out[73]:
```
GradientBoostingRegressor(max_depth=10)
```

In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

In [74]:
```
y_gbm=gbm.predict(x_test)#predicting the price
```

In [75]:
```
metrics(y_test, y_gbm)
```

```
r2_score: 0.8080885707108951
MAE: 0.255629284684813
RMSE: 0.44113247201162703
```

DISCUSSION

Comparing the Linear Regression, Gradient Boosting Regressor, and Random Forest Regressor models in terms of their performance metrics such as R^2 score, Mean Squared Error (MSE), and Root Mean Squared Error (RMSE). We'll also consider their overall accuracy.

Model Comparison:

Linear Regression:
Accuracy: Moderate
R^2 Score: 0.61
MAE: 0.42
RMSE: 0.62

Gradient Boosting Regressor:
Accuracy: High
R^2 Score: 0.80
MAE: 0.25
RMSE: 0.44

Random Forest Regressor:
Accuracy: Highest
R^2 Score: 0.81
MAE: 0.25
RMSE: 0.43

Model Comparison Analysis:

R^2 Score: Random Forest Regressor achieves the highest R^2 score, indicating that it explains more variance in the data compared to Linear Regression and Gradient Boosting Regressor.
MSE and RMSE: Random Forest Regressor also has the lowest MSE and RMSE values, indicating better accuracy and predictive performance compared to the other models.
Accuracy: Random Forest Regressor outperforms both Linear Regression and Gradient Boosting Regressor in terms of overall accuracy.

CONCLUSION

Based on the comparison, the Random Forest Regressor model demonstrates superior performance in terms of accuracy, R^2 score, MSE, and RMSE compared to Linear Regression and Gradient Boosting Regressor. Therefore, for this particular dataset and prediction task, the Random Forest Regressor model would be the preferred choice. Its ability to handle complex relationships and reduce overfitting makes it well-suited for predictive modeling tasks in various domains.

REFERENCE

Agarwal, A., & Agarwal, A. (2020). Predicting Flight Prices: A Machine Learning Approach. International Journal of Computer Applications, 174(24), 16-20.

This paper presents a comprehensive study on predicting flight prices using machine learning techniques. It covers data collection, preprocessing, feature engineering, model selection, and evaluation metrics.

Hargrove, D., & Roth, C. (2019). Flight Price Prediction Using Random Forest Regression. Proceedings of the International Conference on Machine Learning and Applications (ICMLA).

This conference paper introduces the application of Random Forest Regression for flight price prediction. It discusses model performance, hyperparameter tuning, and comparative analysis with other regression models.

Kumar, S., & Jha, P. (2018). Machine Learning Approaches for Flight Price Prediction: A Comparative Study. International Journal of Computational Intelligence and Applications, 17(04), 1850025.

This study compares various machine learning approaches for flight price prediction, including linear regression, random forest, and gradient boosting. It provides insights into model performance and feature importance analysis.

Srivastava, S., & Sharma, A. (2021). Predicting Flight Fares Using Linear Regression and Gradient Boosting Algorithms. Journal of Applied Data Science, 6(2), 143-155.

This journal article investigates the use of linear regression and gradient boosting algorithms for flight fare prediction. It discusses data preprocessing techniques, model training, and evaluation results.

Tan, W., & Lim, C. (2017). Feature Selection and Ensemble Learning for Flight Fare Prediction. IEEE Transactions on Knowledge and Data Engineering, 29(10), 2201-2214.

This research paper explores feature selection methods and ensemble learning techniques for improving flight fare prediction accuracy. It offers insights into model interpretability and performance optimization strategies.

Zhang, Y., & Chen, Z. (2016). Predicting Airline Ticket Prices: A Machine Learning Approach. Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI).

This conference paper presents a machine learning approach to predicting airline ticket prices. It discusses data preprocessing steps, feature extraction, model selection, and cross-validation techniques.