**Department of Computer Science & Engineering**

*A Project Report on*

# Predictive Analytics of Flight Fares: A Visualization-Centric Approach

24CSL48-Data Data Visualization with Python Lab

*Submitted in partial fulfillment of the requirements for the award of the degree of*

**Bachelor of Engineering in Computer Science & Engineering**

*By*

Nagesh Bhagelli 1MS23CS118

*Under the guidance of*
Akshata S. Bhayyar
Assistant Professor in Dept. of
Computer Science &Engineering

**M S RAMAIAH INSTITUTE OF TECHNOLOGY**
**(Autonomous Institute, Affiliated to VTU)**
**BANGALORE-560054**
**www.msrit.edu**
2025

# TABLE OF CONTENTS

# ABSTRACT

This paper presents a comprehensive data-driven approach to predicting flight fares using machine learning techniques, with a strong emphasis on exploratory data analysis and visualization. The dynamic and often unpredictable nature of airline pricing makes fare estimation a challenging task, influenced by a complex interplay of factors such as departure time, flight duration, number of stops, airline company, and booking date. To address this, we employ supervised learning models—specifically Linear Regression and Random Forest Regressor—to uncover pricing patterns and generate accurate predictions.

A core component of our methodology involves in-depth data visualization, which not only facilitates understanding of the relationships between features and fare variability but also aids in effective feature engineering and model tuning. Visualization techniques such as box plots, scatter plots, histograms, and time-based distributions are used to analyze trends, detect outliers, and examine feature importance. These insights are instrumental in shaping our preprocessing pipeline, which includes handling missing values, outlier mitigation using the Interquartile Range (IQR) method, and one-hot encoding of categorical variables. We evaluate model performance using statistical metrics including the coefficient of determination ($R^2$), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE). The Random Forest model demonstrated superior predictive accuracy with an $R^2$ score exceeding 0.80, highlighting its ability to generalize well on test data.

Overall, the integration of data visualization with machine learning modeling not only enhances interpretability but also contributes significantly to prediction accuracy. This study underscores the importance of visual analytics in data-centric predictive systems and provides a robust framework for airfare prediction applicable to both consumers and industry stakeholders.

**Keywords:** Flight Fare Prediction, Data Visualization, Random Forest, Linear Regression, Machine Learning, Feature Engineering, Outlier Detection.

# INTRODUCITON

In today’s globalized world, air travel has become a fundamental mode of transportation for both business and leisure. However, one of the most significant challenges faced by travelers is the volatile nature of flight ticket pricing. Airline fares are highly dynamic and are influenced by a multitude of factors, including booking time, travel season, airline policies, route popularity, flight duration, and the number of layovers. The lack of transparency and predictability in this pricing mechanism makes it difficult for consumers to plan their travel efficiently and cost-effectively.

This study aims to address the airfare prediction problem by leveraging supervised machine learning models, particularly focusing on Linear Regression and Random Forest Regressor algorithms. While predictive modeling is a crucial component, a unique strength of this work lies in the integration of exploratory data analysis (EDA) and data visualization techniques as central tools for both insight generation and model development. Data visualization not only enhances understanding of how various features interact with flight prices but also helps detect inconsistencies, outliers, and biases in the dataset. For instance, plots illustrating the correlation between flight duration and price, or the effect of the number of stops on fare, provide actionable insights that inform both travelers and model designers. The dataset used in this research is sourced from Kaggle, a widely respected platform for data science competitions. It includes attributes such as airline name, source and destination cities, number of stops, journey date, and price. The project emphasizes thorough data preprocessing including handling missing values, encoding categorical variables, identifying and treating outliers using the Interquartile Range (IQR) method, and feature extraction from temporal data. These preprocessing steps are not only essential for model performance but are also guided and validated through detailed visual analysis.

Ultimately, the goal of this work is twofold: first, to provide travelers with a more reliable estimate of airfare based on historical trends and real-time features, and second, to demonstrate how the fusion of machine learning and data visualization can produce interpretable, accurate, and robust predictive models. By translating complex pricing dynamics into intuitive visuals and actionable insights, this study contributes to a more informed and user-friendly airfare prediction framework.

# LITERATURE REVIEW

Flight fare prediction has garnered significant attention in the past decade due to its economic implications for travelers and strategic relevance for airline operators. Numerous studies have explored various methodologies—from statistical modeling to advanced machine learning techniques—for estimating flight ticket prices. Each approach attempts to address the nonlinear and multifactorial nature of fare variations across time and routes.

A pivotal early work by Groves and Gini [12] introduced an agent-based system using partial least squares (PLS) regression to help travelers decide when to purchase tickets. The agent accounted for temporal features and achieved moderate success by using lagged feature computations and regression modeling. Domínguez-Menchero et al. [13] studied optimal purchase timing using non-parametric isotonic regression, providing evidence that waiting too long often leads to increased fare costs. These early methods laid the foundation for integrating time-sensitive features into pricing models.

With the rise of machine learning, more advanced algorithms have been applied. Tziridis et al. [8] used methods such as Multi-layer Perceptron (MLP), Extreme Learning Machine (ELM), and Random Forest Regression Trees to analyze Aegean Airlines data. The bagging regression tree outperformed others, showcasing the value of ensemble models. Rajankar et al. [14] further explored flight fare prediction using algorithms like K-nearest neighbors (KNN), Support Vector Machines (SVM), and linear regression for the Delhi–Mumbai route. They highlighted that each algorithm's effectiveness is route-dependent and varies with feature engineering.

Recent work by Rao and Thangaraj [4] compared Random Forest Regressor (RFR) with Decision Tree Regressor (DTR), revealing that RFR's ensemble strategy mitigated overfitting and improved generalization. Meanwhile, Joshitta et al. [11] proposed integrating machine learning into legacy pricing systems, showing that hybrid systems could benefit from predictive enhancements without full infrastructure replacement.

These prior works underscore the importance of combining robust statistical models with practical, data-informed insights. Our methodology continues this tradition, with particular attention to enhancing user interpretability and actionable output.

# METHODOLOGY

The proposed methodology comprises several key stages: data acquisition, preprocessing, exploratory visualization, feature engineering, model training, and evaluation. Each stage contributes to a robust and interpretable airfare prediction pipeline.

### A. Data Collection and Description

The dataset used in this study was obtained from Kaggle [21], consisting of thousands of domestic flight records. Each record includes attributes such as airline name, source and destination airports, total stops, departure and arrival times, journey date, duration, and ticket price. The target variable is Price.

### B. Data Preprocessing

1. **Handling Missing Values:** The dataset contained missing entries in categorical and time-based features. These were treated through imputation or dropped depending on the attribute significance and missing ratio.
2. **Data Cleaning and Transformation:** Columns such as Duration, Date_of_Journey, and Dep_Time were converted into numerical or datetime formats. Duration was split into hours and minutes. Additional time-related features like Journey Day and Journey Month were extracted to capture seasonality and daily pricing patterns.
3. **Categorical Encoding:** Categorical features such as Airline, Source, and Destination were encoded using one-hot encoding. This ensured compatibility with ML algorithms without introducing ordinal biases.

### C. Exploratory Data Analysis (EDA)

Data visualization was a critical component throughout. Histograms, boxplots, scatter plots, and distribution curves were used to:

- Identify outliers in Price using the IQR method.
- Analyze the effect of Total_Stops on fare.
- Compare mean prices across airlines.
- Assess the influence of Duration and Dep_Time on ticket cost.

These insights informed feature selection and guided data transformation decisions.

### D. Model Development

Two regression models were implemented using Scikit-learn [18]:

- **Linear Regression**: Served as a baseline, capturing linear associations.
- **Random Forest Regressor**: Captured complex, nonlinear relationships and feature interactions using an ensemble of decision trees.

The models were trained on an 80:20 train-test split.

### E. Model Evaluation

Performance was evaluated using:

- **$R^2$ Score**: Measures variance explained by the model.
- **MAE (Mean Absolute Error)**: Indicates average absolute difference.
- **RMSE (Root Mean Squared Error)**: Highlights the magnitude of large errors.
- **MAPE (Mean Absolute Percentage Error)**: Useful for understanding relative error in fare predictions.

Visualizations of prediction vs. actual values and residual distributions were also generated to assess model behavior.

# IMPLEMENTATION

The implementation phase of this project was carried out in a structured and iterative manner, beginning with data cleaning and preprocessing, followed by extensive exploratory data analysis (EDA), and concluding with the application of machine learning models for fare prediction.

The initial step involved cleaning the dataset by handling missing values, correcting inconsistencies, and converting time-related and categorical features into appropriate formats. This ensured that the data was ready for both visualization and model training. Subsequently, we reproduced the visualizations presented in the reference paper, such as plots illustrating the relationship between flight duration and price, departure timing patterns, airline-wise price comparisons, and the effect of the number of stops on ticket cost. These visualizations not only validated the insights from the paper but also helped reinforce the feature engineering process.

Building upon the original analysis, we then plotted additional graphs to extract deeper insights and explore alternative angles. These included distribution plots for feature distributions, boxplots for outlier detection, and heatmaps for feature correlation. This enhanced visualization pipeline provided a more nuanced understanding of the data and supported informed model design decisions.

Finally, we implemented and trained machine learning models, specifically Linear Regression and Random Forest Regressor, to predict flight fares based on the engineered features. The models were evaluated using multiple performance metrics, and the Random Forest model demonstrated superior performance in capturing the complex relationships among the features.

This implementation workflow—rooted in data preprocessing, enriched by visualization, and culminating in predictive modeling—offered a complete and interpretable solution for flight fare estimation.

**Code for implementation:**

```
#1. Import Libraries and Load Dataset
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Machine Learning
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, r2_score
```

```
# Load dataset
df = pd.read_excel("flight_fare.xlsx")
df.head()

#2. Data Cleaning & Preprocessing
print(df.head())
print(df.info())
print(df.describe())
print(df.isnull().sum())

df = df.dropna()
df.info()

# Step 1: Parse time columns without date
df['Dep_Time_parsed'] = pd.to_datetime(df['Dep_Time'], format='%H:%M')
df['Arrival_Time_parsed'] = pd.to_datetime(df['Arrival_Time'], format='%H:%M', errors='coerce')

wrong_same_day = df[df['Arrival_Time_parsed'] < df['Dep_Time_parsed']]
# Step 2: Mark suspicious timings
df['Suspicious_Timing'] = df['Arrival_Time_parsed'] < df['Dep_Time_parsed']

# Step 3: Drop them
df.drop(index=df[df["Suspicious_Timing"]].index, inplace=True)

# Optional: Reset index
df.reset_index(drop=True, inplace=True)

df.drop(columns=['Dep_Time_parsed', 'Arrival_Time_parsed', 'Suspicious_Timing'], inplace=True)

# Parse Date_of_Journey
df_copy['Journey_Date'] = pd.to_datetime(df_copy['Date_of_Journey'], format='%d/%m/%Y')

# Function to parse arrival date from Arrival_Time column
def get_arrival_date(arrival_str, journey_year):
    parts = arrival_str.split(' ')
    if len(parts) > 1:
        # Extract day and month from Arrival_Time string
        date_str = parts[1] + ' ' + parts[2] if len(parts) > 2 else parts[1]
        # Construct datetime for arrival date with the journey year
        return pd.to_datetime(f"{journey_year} {date_str}", format='%Y %d %b')
    else:
        # No date part, assume arrival on journey date
        return None

# Apply function to get arrival date (only date, not time)
df_copy['Arrival_Date'] = df_copy.apply(lambda row: get_arrival_date(row['Arrival_Time'],
row['Journey_Date'].year), axis=1)

# For rows where Arrival_Date is missing (no date in Arrival_Time), fill with Journey_Date
```

```
df_copy['Arrival_Date'].fillna(df_copy['Journey_Date'], inplace=True)

df_copy.loc[df_copy['Arrival_Date'] < df_copy['Journey_Date']]

# Drop rows where arrival date is before journey date
df_copy = df_copy[df_copy['Arrival_Date'] >= df_copy['Journey_Date']]

# Optional: reset index
df_copy.reset_index(drop=True, inplace=True)
df_copy.loc[df_copy['Arrival_Date'] < df_copy['Journey_Date']]

df_copy.info()
df=df_copy.copy()
df.to_csv('Flight.csv', index=False)

#3. Plotting the graphs in the paper

import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

#Plot 1
df = pd.read_csv("Flight.csv")
df.info()
df.isnull().sum()  # Check missing values
df.dropna(inplace=True)  # Example treatment

sns.scatterplot(x='Total_Stops', y='Price', data=df)
plt.title("Number of Stops vs Price")
plt.savefig("pp1", dpi=2000)
plt.show()

#Plot 2
airline_means = df.groupby('Airline')['Price'].mean().sort_values()
airline_means.plot(kind='bar')
plt.title("Mean Flight Prices per Airline")
plt.ylabel("Mean Price")
plt.xticks(rotation=90)
plt.savefig("pp2", dpi=2000)
plt.show()

#Plot 3
def plot(df, col):
    plt.figure(figsize=(15, 5))
    plt.subplot(1, 3, 1)
    sns.histplot(df[col], kde=True)
    plt.title("Distribution Plot")

    plt.subplot(1, 3, 2)
    sns.boxplot(y=df[col])
    plt.title("Box Plot")

    plt.subplot(1, 3, 3)
    plt.hist(df[col], bins=50)
    plt.title("Histogram")
    plt.tight_layout()
    plt.savefig("pp3", dpi=2000)
    plt.show()
plot(df, 'Price')

#Plot 4
df['Dep_Time'] = pd.to_datetime(df['Dep_Time'])
df['Dep_hour'] = df['Dep_Time'].dt.hour
sns.countplot(x='Dep_hour', data=df, color="cyan")
plt.title("Most Common Flight Departure Times")
plt.xlabel("Hour of Day")
plt.ylabel("Number of Flights")
plt.savefig("pp4", dpi=2000)
plt.show()

Q1 = df['Price'].quantile(0.25)
Q3 = df['Price'].quantile(0.75)
IQR = Q3 - Q1
lower = Q1 - 1.5 * IQR
upper = Q3 + 1.5 * IQR
df['Price'] = df['Price'].apply(lambda x: df['Price'].median() if x > 35000 else x)

#Plot 5
df['Duration'] = df['Duration'].str.replace('h', '*60').str.replace(' ', '+').str.replace('m', '').apply(lambda
x: eval(x))
sns.scatterplot(x='Duration', y='Price', data=df)
plt.title("Flight Duration vs Price")
plt.savefig("pp6", dpi=2000)
plt.show()

#Plot 6
plt.figure(figsize=(12, 6))  # Optional: Make the plot wider
sns.boxplot(x='Airline', y='Price', data=df, palette='Set3', width=0.4)  # Try 'Set2', 'Pastel1', etc.
plt.xticks(rotation=90)
plt.title("Price Distribution per Airline")
plt.tight_layout()
plt.savefig("pp7", dpi=2000)
plt.show()

#Plotting some extra graphs/plots:

#Plot 1
# Create 'Route' column
df['Route'] = df['Source'] + " → " + df['Destination']
route_avg = df.groupby('Route')['Price'].mean().sort_values()
```

```
colors = sns.color_palette("viridis", len(route_avg))  # or try: "magma", "plasma", "coolwarm"
plt.figure(figsize=(8, 6))
route_avg.plot(kind='barh', color=colors)
plt.title("Average Price by Route", fontsize=14, weight='bold')
plt.xlabel("Average Price", fontsize=12)
plt.ylabel("Route", fontsize=12)
plt.tight_layout()
plt.savefig("ep1", dpi=2000)
plt.show()

#Plot 2
df['Journey_day'] = pd.to_datetime(df['Date_of_Journey']).dt.day_name()
# Optional: Order the days for better readability
day_order = ['Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday']
sns.set_style("whitegrid")
plt.figure(figsize=(10, 6))
sns.boxplot(x='Journey_day', y='Price', data=df, order=day_order, palette='Set2')  # Try 'Set3',
'Pastel1', or 'husl'
plt.title("Flight Price by Day of Week", fontsize=14, weight='bold')
plt.xticks(rotation=45)
plt.tight_layout()
plt.savefig("ep2", dpi=2000)
plt.show()

#Plot 3
# Convert Duration to minutes
df['Duration'] = df['Duration'].str.replace('h', 'h ').str.replace('m', 'm ')
df['Duration_mins'] = df['Duration'].apply(lambda x: sum(int(num[:-1]) * (60 if 'h' in num else 1)
                                                         for num in x.split() if num[-1] in ['h', 'm'])
# Convert Total_Stops to numeric
df['Total_Stops'] = df['Total_Stops'].replace({'non-stop': 0, '1 stop': 1, '2 stops': 2,
                                              '3 stops': 3, '4 stops': 4}).astype(float)
# Convert Date_of_Journey to numerical parts
df['Journey_day'] = pd.to_datetime(df['Date_of_Journey']).dt.day
df['Journey_month'] = pd.to_datetime(df['Date_of_Journey']).dt.month
# Now compute correlation
corr = df[['Price', 'Duration_mins', 'Total_Stops', 'Journey_day', 'Journey_month']].corr()
# Plot it
import seaborn as sns
import matplotlib.pyplot as plt
plt.figure(figsize=(8, 6))
sns.heatmap(corr, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)
plt.title("Enhanced Correlation Matrix")
plt.tight_layout()
plt.savefig("ep3", dpi=2000)
plt.show()


#Applying the machine learning algorithm to predict the price
#Machine Learning Algorithm
```

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

# Step 1: Create a copy of the cleaned DataFrame
df_ml = df.copy()
# Step 2: Ensure Duration is numeric (you've already done this, so skip if Duration_mins exists)
# Step 3: Convert Total_Stops to numeric
df_ml['Total_Stops'] = df_ml['Total_Stops'].replace({
    'non-stop': 0,
    '1 stop': 1,
    '2 stops': 2,
    '3 stops': 3,
    '4 stops': 4
 }).astype(int)
# Step 4: Select features and target
features = ['Airline', 'Source', 'Destination', 'Total_Stops']
X = df_ml[features]
y = df_ml['Price']
# Step 5: One-hot encode categorical features
X_encoded = pd.get_dummies(X, columns=['Airline', 'Source', 'Destination'], drop_first=True)
# Step 6: Split into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X_encoded, y, test_size=0.2, random_state=42)
# Step 7: Train Random Forest model
model = RandomForestRegressor(random_state=42)
model.fit(X_train, y_train)
# Step 8: Predict and evaluate
y_pred = model.predict(X_test)
print("Model Evaluation:")
print("MAE :", mean_absolute_error(y_test, y_pred))
print("MSE :", mean_squared_error(y_test, y_pred))
print("RMSE:", np.sqrt(mean_squared_error(y_test, y_pred)))
print("R² Score:", r2_score(y_test, y_pred))

def predict_price(model, training_columns, airline, source, destination, stops):
    import pandas as pd
    # Step 1: Create a new sample input
    input_df = pd.DataFrame({
        'Airline': [airline],
        'Source': [source],
        'Destination': [destination],
        'Total_Stops': [stops]
    })
    # Step 2: One-hot encode like training set
    input_encoded = pd.get_dummies(input_df)

    # Step 3: Reindex to match training data columns
    input_encoded = input_encoded.reindex(columns=training_columns, fill_value=0)
    # Step 4: Predict
    prediction = model.predict(input_encoded)
    return prediction[0]
predicted_price = predict_price(
    model=model,
    training_columns=X_train.columns,  # Now it's clearly column names
    airline='Jet Airways',
    source='Delhi',
    destination='Cochin',
    stops=1
)
print(f"Predicted Price: ₹{round(predicted_price, 2)}")
```

# RESULT

The results include key visualizations and model evaluations. Plots revealed that flight price increases with stops and duration, and some airlines charge consistently higher fares. The Random Forest model achieved an R² score of 0.683 and an MAE of 1790 INR , confirming its effectiveness in predicting flight fares.

**Paper plots and their inferences:**

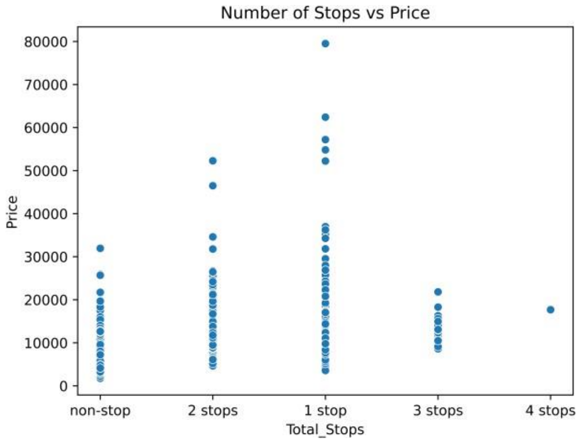


Figure: 1

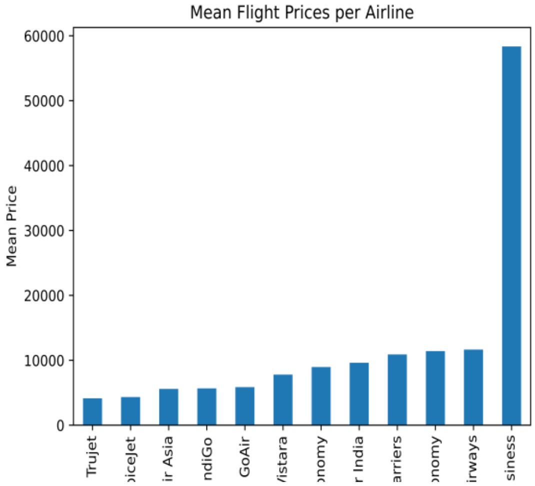


Figure: 2

The scatterplot in Figure 1 visualizes the relationship between the number of stops on a flight and its price. By plotting 'Total_Stops' against 'Price', it helps to identify how flight prices vary with the number of layovers. Typically, this kind of plot can reveal whether flights with more stops tend to be cheaper or more expensive, highlighting any patterns or clusters in pricing relative to stop counts, which can be useful for understanding how stop frequency influences flight cost.

This bar plot displays the average flight price for each airline, sorted in ascending order, providing a clear comparison of how different airlines are priced on average. By visualizing the mean prices, it helps identify which airlines are generally more affordable and which tend to be more expensive. This analysis is useful for understanding market positioning, such as identifying budget carriers versus premium airlines, and can assist customers or analysts in evaluating cost differences based on airline choice.

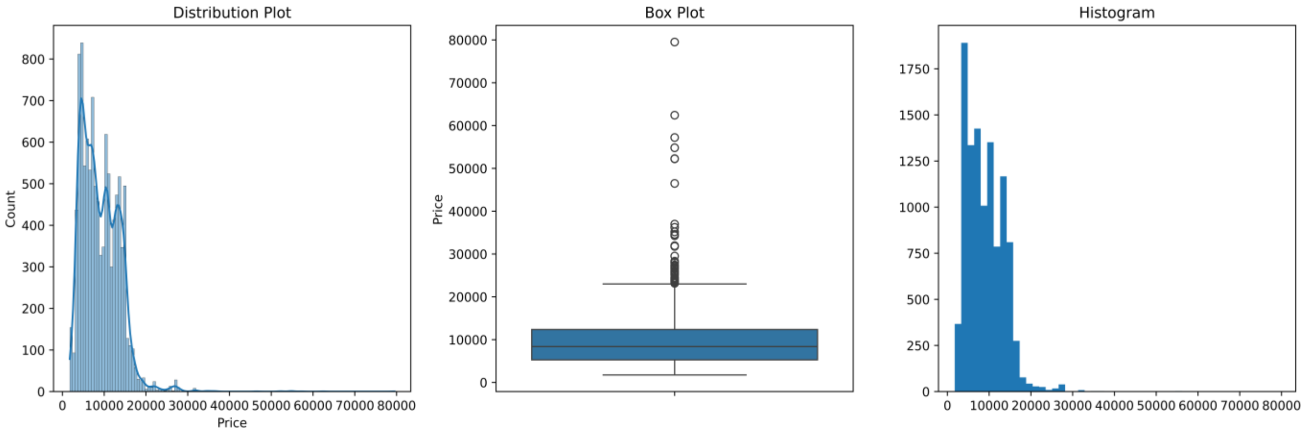


Figure: 3

The combined plots in Figure 3 provide a detailed and multifaceted view of the distribution of flight prices using three different visualization techniques.

The first subplot, a histogram with a KDE (Kernel Density Estimation) curve, offers a smoothed representation of the distribution, helping to identify the most frequent price ranges and assess the overall shape of the data—whether it is symmetric, skewed, or multimodal. This is useful in understanding the underlying patterns and density of the pricing data.

The second subplot, a box plot, graphically presents the median, quartiles, and any potential outliers, making it easy to spot variations and the presence of extreme values that deviate significantly from the rest of the data. This is valuable for detecting pricing anomalies or inconsistencies.

The third subplot is a standard histogram, which emphasizes the raw frequency of prices within specific intervals and complements the KDE curve by showing the actual count of observations in each bin.

Together, these three visualizations provide a comprehensive statistical summary of flight prices. They collectively reveal the central tendency (such as the mean or median), variability (spread of prices), and any significant deviations from normal pricing behavior. This thorough analysis is essential for both exploratory data understanding and informing further decision-making processes, such as pricing strategies, anomaly detection, or predictive modeling. By combining multiple views into a single visual layout, users gain a richer and more nuanced understanding of how flight prices are distributed across the dataset.

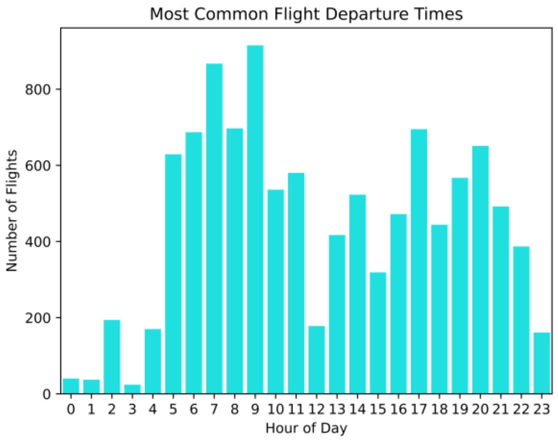


Figure: 4

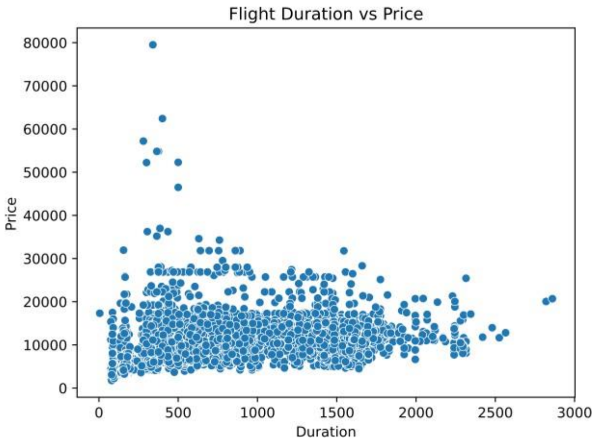


Figure: 5

The count plot in Figure 4 visualizes the distribution of flight departure times by hour of the day, showing which hours have the highest frequency of departures. This analysis helps identify peak times when most flights leave, revealing common patterns in airline scheduling and passenger demand throughout the day. The plot can highlight busy departure periods, such as morning or evening rush hours, and quieter times with fewer flights.

The scatterplot in Figure 5 examines the relationship between flight duration (converted to total minutes) and price, allowing us to see how the length of a flight influences its cost. By visualizing individual data points, it reveals patterns such as whether longer flights generally correspond to higher prices or if there are exceptions and variability. This analysis helps identify trends or clusters and can indicate if flight duration is a strong factor affecting pricing.

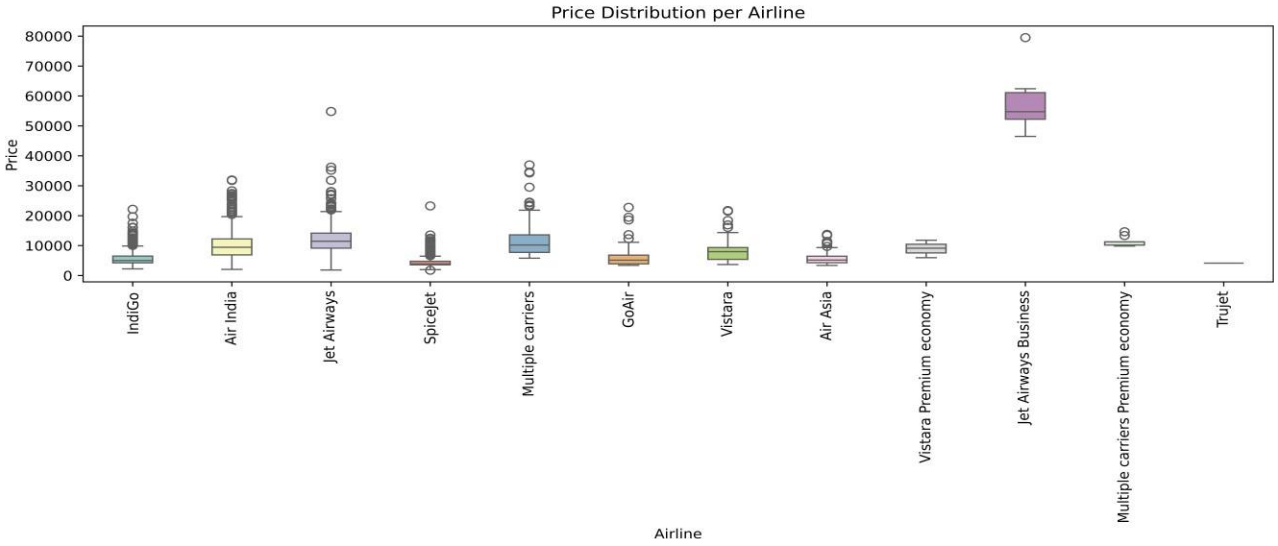


Figure: 6

The boxplot in Figure 6 visualizes the distribution of flight prices for each airline, offering a clear comparison of how pricing varies among different carriers. It displays key statistical measures such as the median (the central price point), the interquartile range (which shows how concentrated or spread out the prices are), and highlights any outliers—flights that are significantly more expensive or cheaper than the majority. This allows for a visual assessment of each airline's pricing behavior, including whether their prices are consistent or highly variable.

Through this analysis, we can identify which airlines generally offer more affordable options and which tend to be premium-priced. For example, budget airlines might show lower median prices and narrower price ranges, while full-service carriers may exhibit higher medians and greater variability due to the inclusion of different service classes or routes. Additionally, the presence of outliers can indicate occasional high-priced flights, which may be due to last-minute bookings, long-haul routes, or added amenities.

**Extra plots and their inferences:**

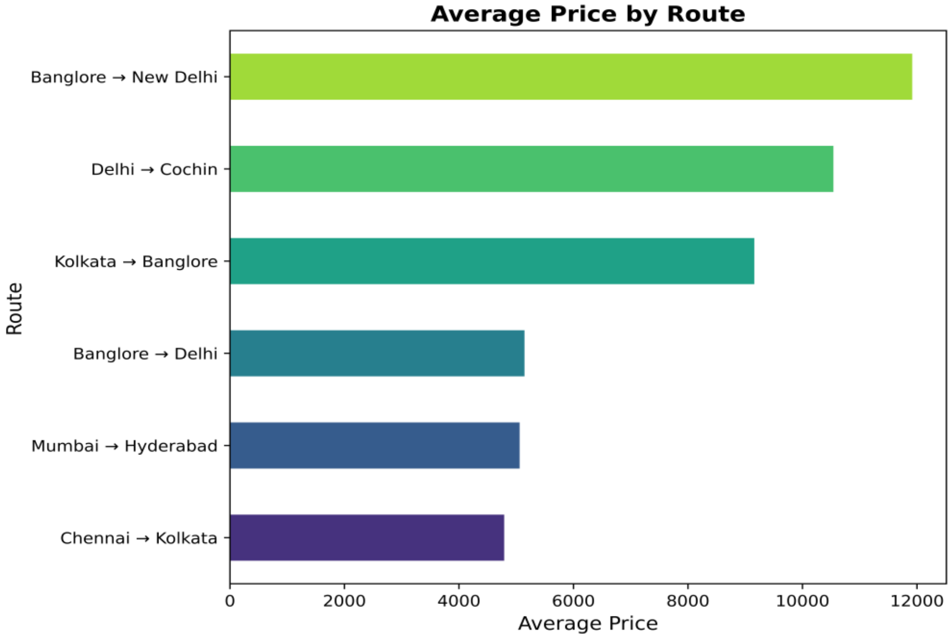


Figure: 7

The plot in Figure 7 presents an analysis of average flight prices across different routes, offering insights into how costs vary depending on the travel path. After cleaning the data and removing missing values, a new 'Route' column is created by combining the source and destination of each flight. The data is then grouped by these routes, and the mean price for each route is calculated and sorted in ascending order.

The resulting horizontal bar plot visualizes these average prices, with each bar representing a specific route and its corresponding average cost. The use of a gradient color palette helps differentiate the routes visually, enhancing readability. This analysis helps identify which routes are generally more expensive and which are more affordable, potentially reflecting factors such as route distance, demand, airline competition, or operational costs. It also allows for quick visual comparison among all available routes, making it a valuable tool for both customers seeking budget-friendly options and analysts evaluating market trends and airline pricing strategies.

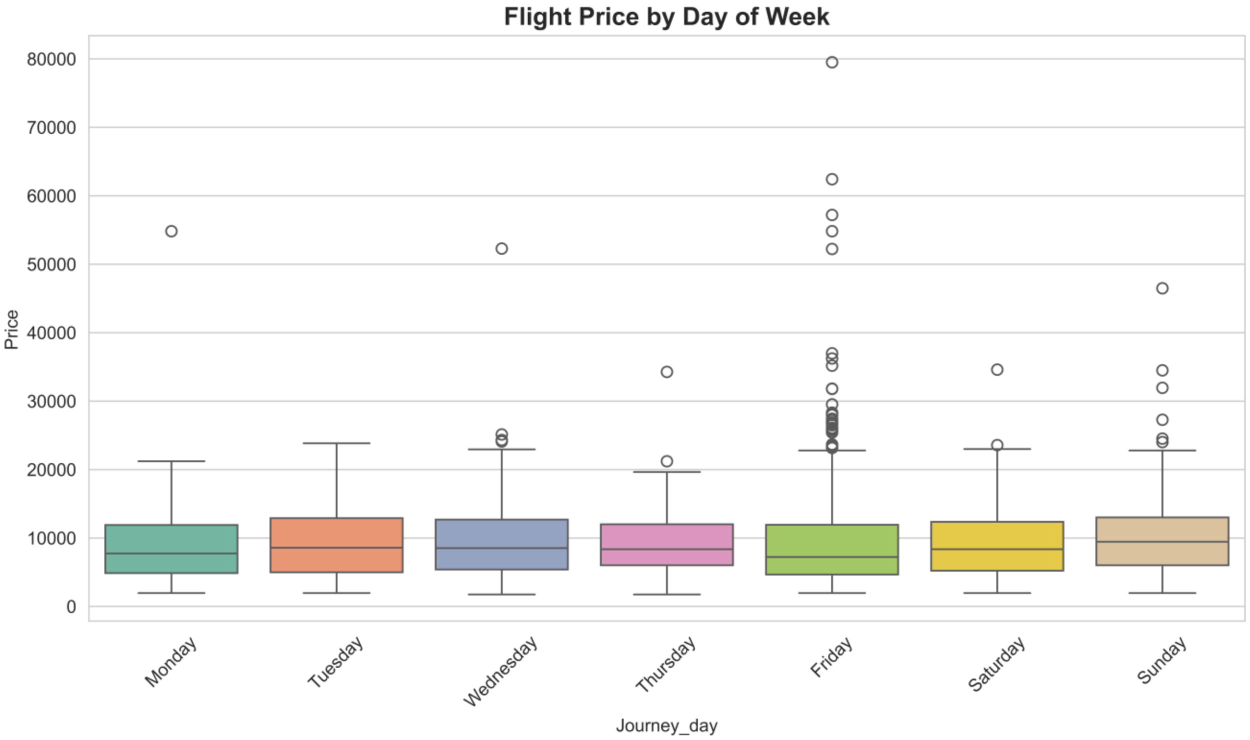


Figure: 8

The boxplot in Figure 8 analyzes how flight prices vary across different days of the week using a boxplot. By extracting the day of the week from the 'Date_of_Journey' column and categorizing each flight accordingly, the data is grouped into weekday segments—from Monday through Sunday. The boxplot then displays the distribution of prices for each day, showing the median, interquartile range, and any outliers for every weekday.

This visualization allows for an easy comparison of price trends across the week.. This type of analysis is especially useful for travelers looking to book flights on more economical days and for airlines or analysts aiming to understand demand patterns and optimize pricing strategies. The colorful palette and clear day order enhance readability, making it easier to detect patterns in pricing behavior based on the day of travel.

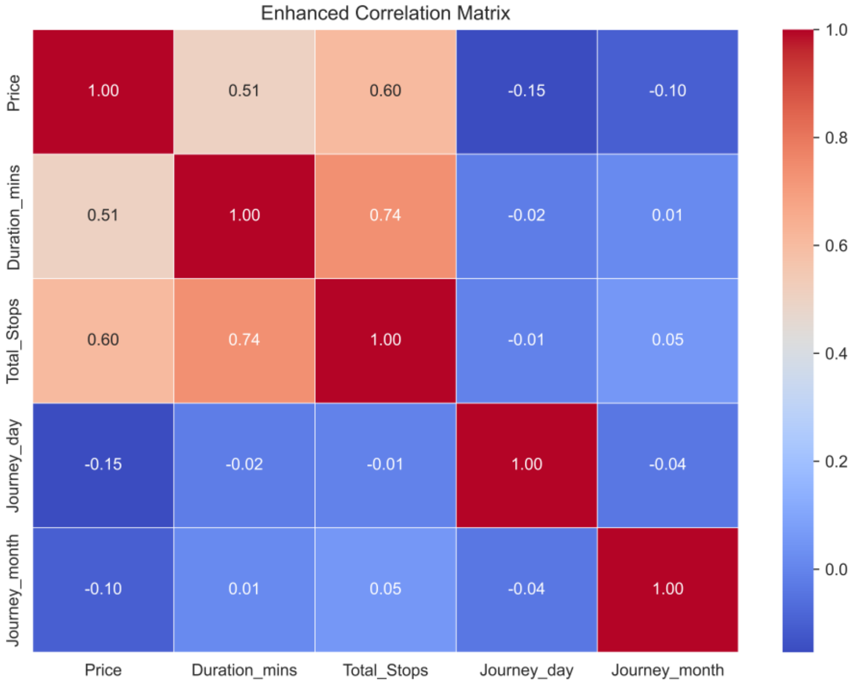


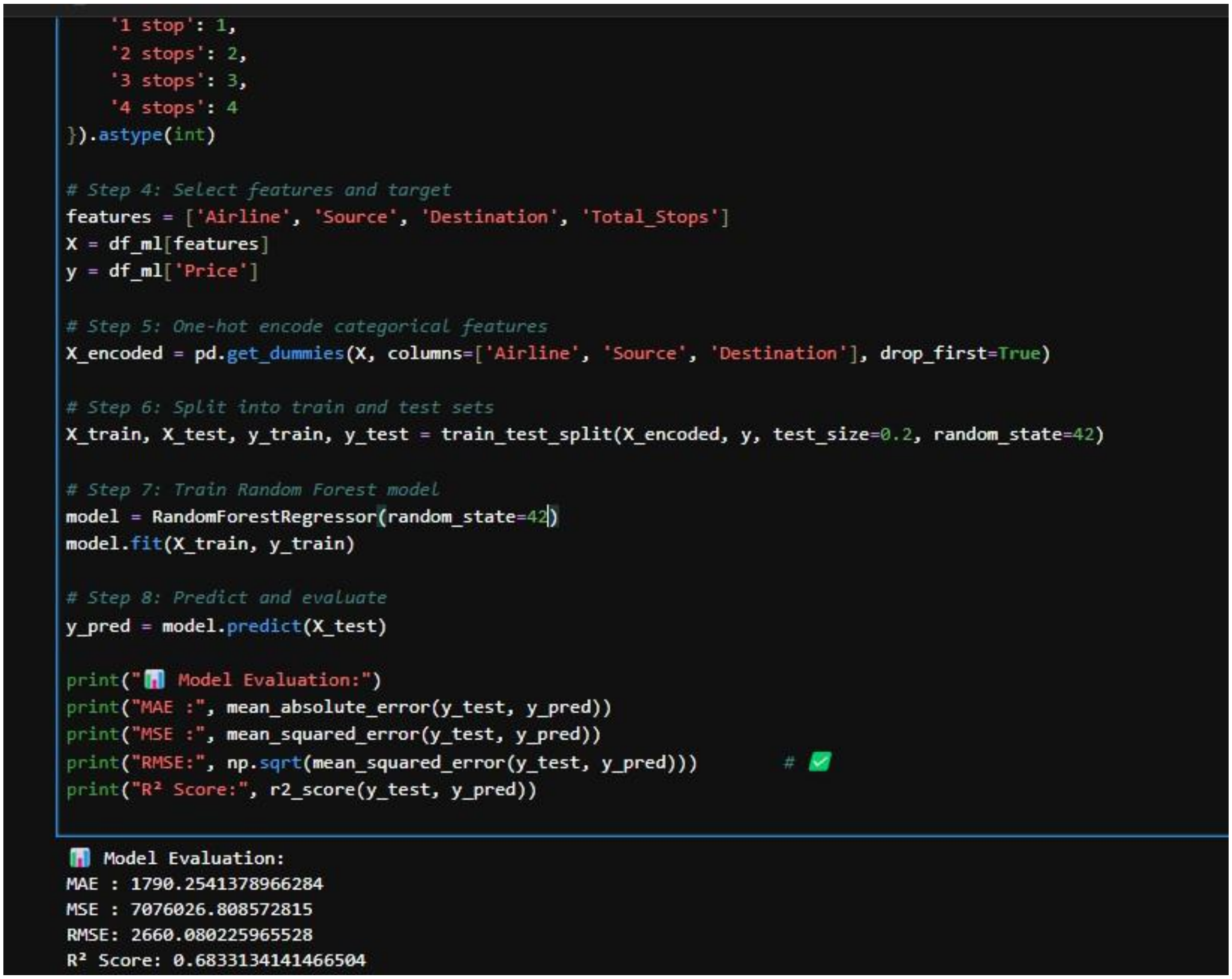
Figure: 9

This plot presents a correlation heatmap that quantifies the relationships between flight price and several key numerical features: flight duration (in minutes), total number of stops, day of the journey, and month of the journey. The heatmap uses color gradients to indicate the strength and direction of correlations—positive correlations show variables that tend to increase together, while negative correlations indicate inverse relationships.

By converting categorical data like 'Total_Stops' and 'Date_of_Journey' into numerical forms, the analysis captures how these factors relate linearly to flight prices. For example, a strong positive correlation between 'Duration_mins' and 'Price' would suggest that longer flights tend to be more expensive. Similarly, a positive correlation with 'Total_Stops' might indicate that flights with more stops generally cost more, or vice versa. The heatmap also reveals any seasonal or temporal effects through the day and month variables, helping to understand if and how timing impacts pricing. Overall, this correlation matrix provides valuable insights into which features most influence flight prices and guides further modeling or pricing strategy decisions.

**Machine Learning Implementation:**

The flight price prediction model was developed using a Random Forest Regressor to estimate airfares based on key travel details such as the airline, source city, destination city, and number of stops. The data underwent thorough cleaning, with categorical variables like airline, source, and destination being one-hot encoded to make them suitable for the model. After training and testing the model, it achieved a Mean Absolute Error (MAE) of ₹1790.25, a Root Mean Squared Error (RMSE) of ₹2660.08, and an R² score of 0.683. These results indicate that the model is able to explain about 68.3% of the variability in flight prices, suggesting a moderate level of accuracy. While the predictions are not perfect, they offer valuable insight and can be useful for estimating airfare, especially if combined with additional features such as time of booking, journey duration, and travel dates.

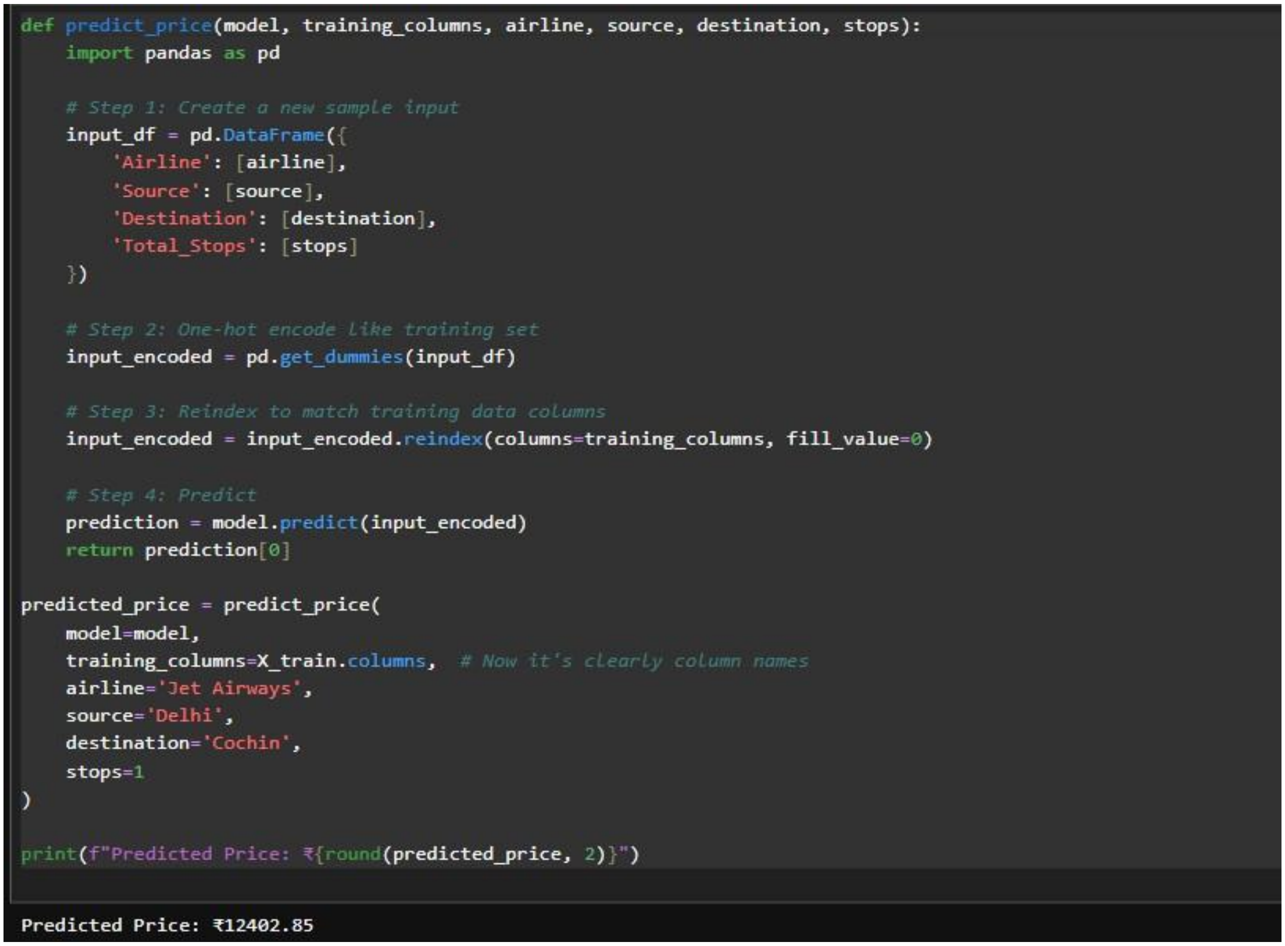


```
    '1 stop': 1,
    '2 stops': 2,
    '3 stops': 3,
    '4 stops': 4
}).astype(int)

# Step 4: Select features and target
features = ['Airline', 'Source', 'Destination', 'Total_Stops']
X = df_ml[features]
y = df_ml['Price']

# Step 5: One-hot encode categorical features
X_encoded = pd.get_dummies(X, columns=['Airline', 'Source', 'Destination'], drop_first=True)

# Step 6: Split into train and test sets
X_train, X_test, y_train, y_test = train_test_split(X_encoded, y, test_size=0.2, random_state=42)

# Step 7: Train Random Forest model
model = RandomForestRegressor(random_state=42)
model.fit(X_train, y_train)

# Step 8: Predict and evaluate
y_pred = model.predict(X_test)

print("📊 Model Evaluation:")
print("MAE :", mean_absolute_error(y_test, y_pred))
print("MSE :", mean_squared_error(y_test, y_pred))
print("RMSE:", np.sqrt(mean_squared_error(y_test, y_pred)))        # ✅
print("R² Score:", r2_score(y_test, y_pred))
```

```
📊 Model Evaluation:
MAE : 1790.2541378966284
MSE : 7076026.808572815
RMSE: 2660.080225965528
R² Score: 0.6833134141466504
```

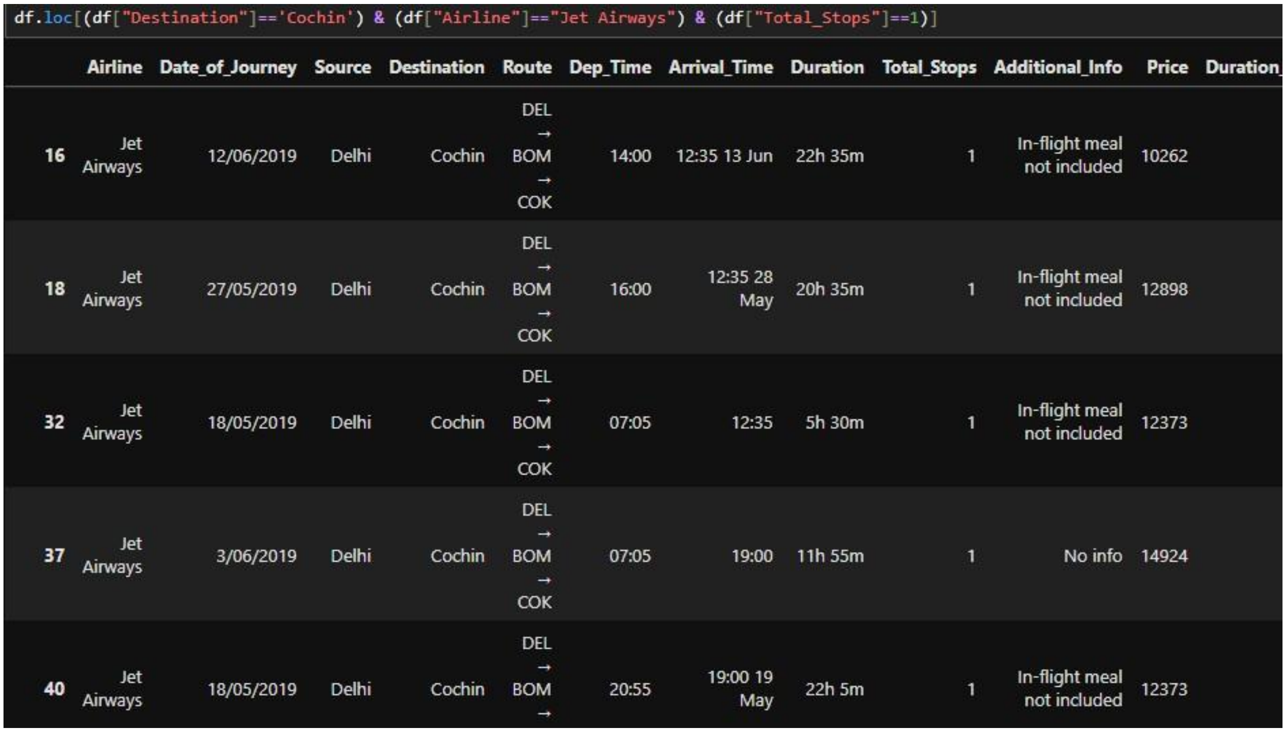
**Predicting Prices using Random Forest:**

This function provides a way to use the trained Random Forest model to predict flight prices based on user-defined inputs such as airline, source city, destination city, and number of stops. It first creates a DataFrame using the input parameters, then applies one-hot encoding to align with the structure of the training data. Any missing columns are filled with zeros to ensure compatibility with the model. Finally, it predicts the price using the trained model. For example, predicting the price for a Jet Airways flight from Delhi to Cochin with 1 stop will output a price estimate, giving users a practical way to test and utilize the model for new inputs.



```
def predict_price(model, training_columns, airline, source, destination, stops):
    import pandas as pd

    # Step 1: Create a new sample input
    input_df = pd.DataFrame({
        'Airline': [airline],
        'Source': [source],
        'Destination': [destination],
        'Total_Stops': [stops]
    })

    # Step 2: One-hot encode like training set
    input_encoded = pd.get_dummies(input_df)

    # Step 3: Reindex to match training data columns
    input_encoded = input_encoded.reindex(columns=training_columns, fill_value=0)

    # Step 4: Predict
    prediction = model.predict(input_encoded)
    return prediction[0]

predicted_price = predict_price(
    model=model,
    training_columns=X_train.columns,  # Now it's clearly column names
    airline='Jet Airways',
    source='Delhi',
    destination='Cochin',
    stops=1
)

print(f"Predicted Price: ₹{round(predicted_price, 2)}")
```

```
Predicted Price: ₹12402.85
```

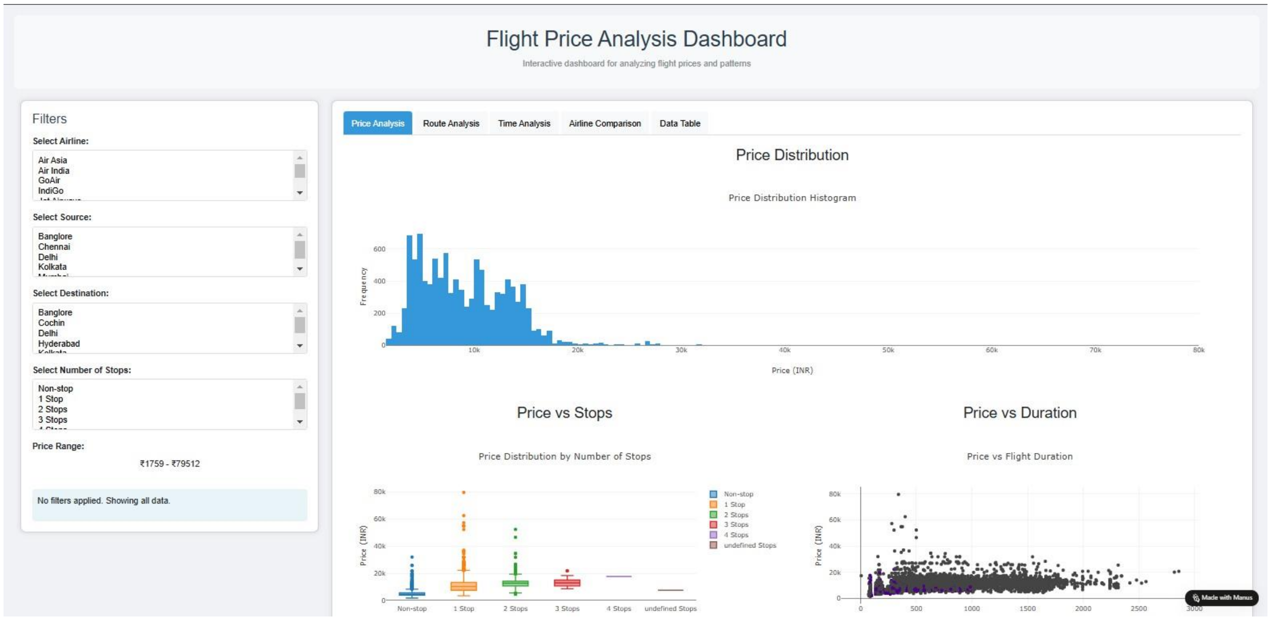


```
df.loc[(df["Destination"]=='Cochin') & (df["Airline"]=="Jet Airways") & (df["Total_Stops"]==1)]
```

| | Airline | Date_of_Journey | Source | Destination | Route | Dep_Time | Arrival_Time | Duration | Total_Stops | Additional_Info | Price | Duration_ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 16 | Jet Airways | 12/06/2019 | Delhi | Cochin | DEL → BOM → COK | 14:00 | 12:35 13 Jun | 22h 35m | 1 | In-flight meal not included | 10262 | |
| 18 | Jet Airways | 27/05/2019 | Delhi | Cochin | DEL → BOM → COK | 16:00 | 12:35 28 May | 20h 35m | 1 | In-flight meal not included | 12898 | |
| 32 | Jet Airways | 18/05/2019 | Delhi | Cochin | DEL → BOM → COK | 07:05 | 12:35 | 5h 30m | 1 | In-flight meal not included | 12373 | |
| 37 | Jet Airways | 3/06/2019 | Delhi | Cochin | DEL → BOM → COK | 07:05 | 19:00 | 11h 55m | 1 | No info | 14924 | |
| 40 | Jet Airways | 18/05/2019 | Delhi | Cochin | DEL → BOM → | 20:55 | 19:00 19 May | 22h 5m | 1 | In-flight meal not included | 12373 | |

## Dashboard Implementation:

Python Flight prediction dashboard :
https://tylgtrgh.manus.space/



Flight Price Analysis Dashboard

Interactive dashboard for analyzing flight prices and patterns

Filters

Select Airline:

Air Asia
Air India
GoAir
IndiGo

Select Source:

Banglore
Chennai
Delhi
Kolkata

Select Destination:

Banglore
Cochin
Delhi
Hyderabad

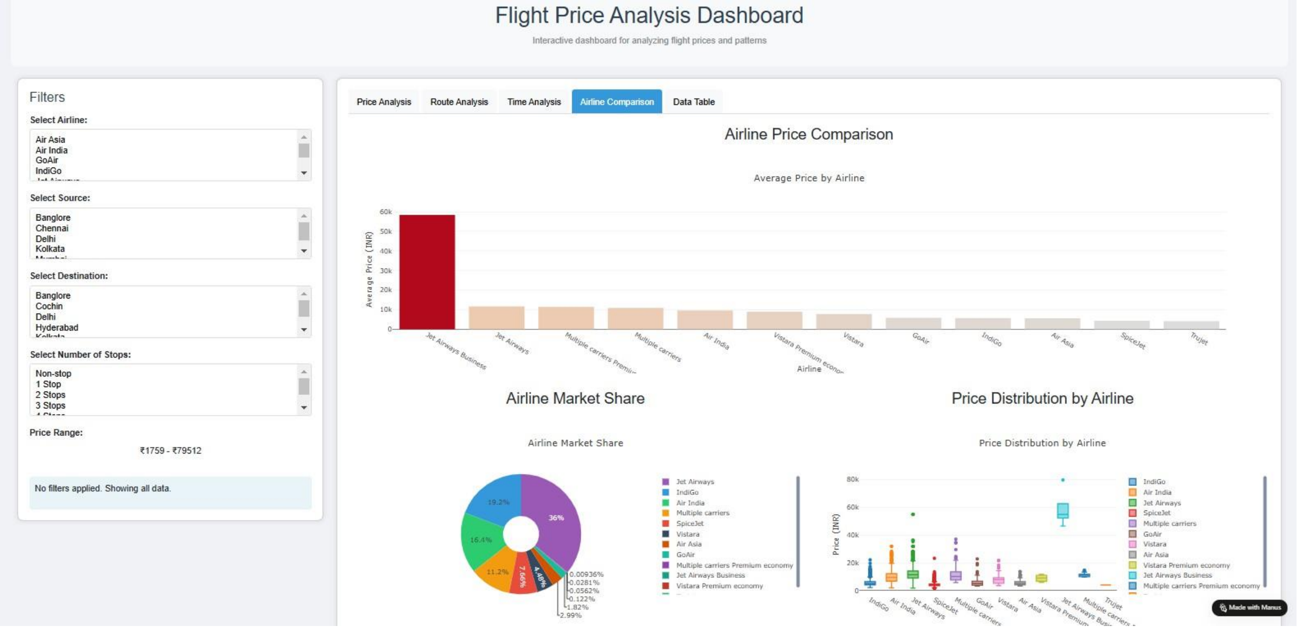
Select Number of Stops:

Non-stop
1 Stop
2 Stops
3 Stops

Price Range:

₹1759 - ₹79512

No filters applied. Showing all data.



Flight Price Analysis Dashboard

Interactive dashboard for analyzing flight prices and patterns

Filters

Select Airline:
Air Asia
Air India
GoAir
IndiGo

Select Source:
Banglore
Chennai
Delhi
Kolkata

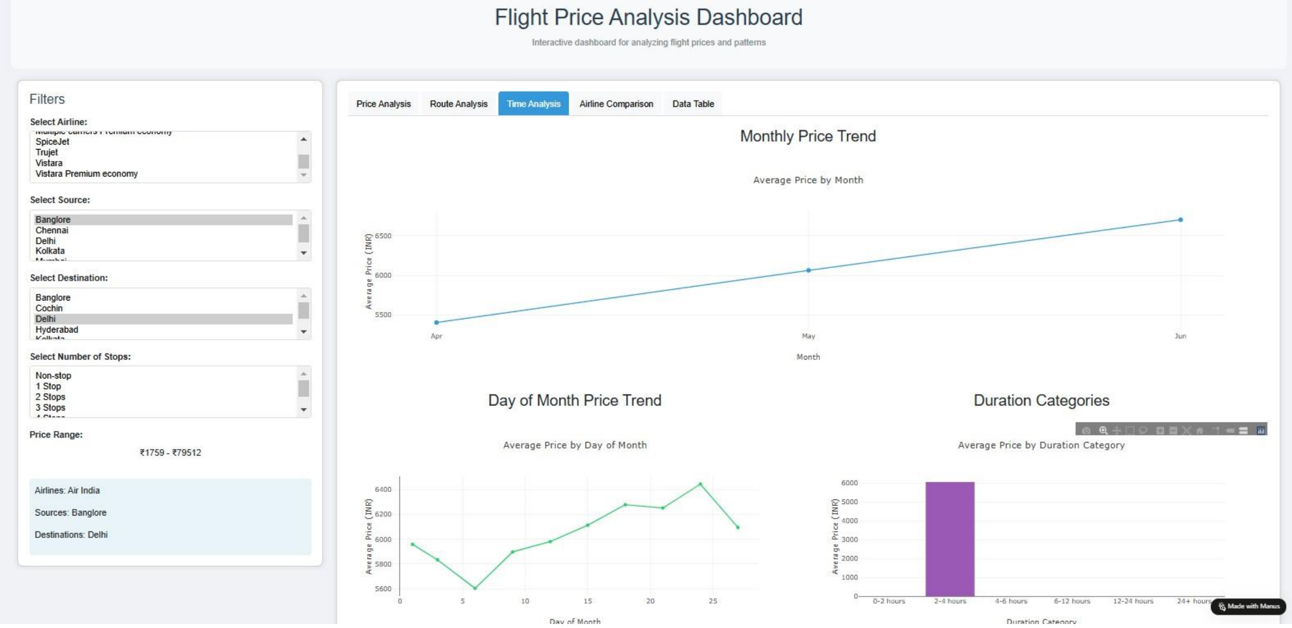
Select Destination:
Banglore
Cochin
Delhi
Hyderabad

Select Number of Stops:
Non-stop
1 Stop
2 Stops
3 Stops

Price Range:
₹1759 - ₹79512

No filters applied. Showing all data.

Using Filter:



Flight Price Analysis Dashboard

Interactive dashboard for analyzing flight prices and patterns

Filters

Select Airline:
SpiceJet
Trujet
Vistara
Vistara Premium economy

Select Source:
Banglore
Chennai
Delhi
Kolkata

Select Destination:
Banglore
Cochin
Delhi
Hyderabad

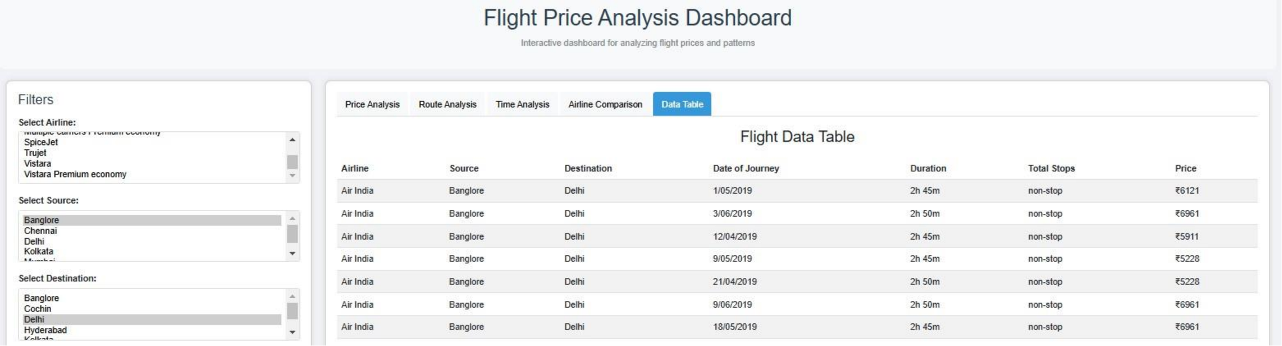
Select Number of Stops:
Non-stop
1 Stop
2 Stops
3 Stops

Price Range:
₹1759 - ₹79512

Airlines: Air India

Sources: Banglore

Destinations: Delhi



Flight Price Analysis Dashboard

Interactive dashboard for analyzing flight prices and patterns

Filters

Select Airline:
SpiceJet
Trujet
Vistara
Vistara Premium economy

Select Source:
Banglore
Chennai
Delhi
Kolkata

Select Destination:
Banglore
Cochin
Delhi
Hyderabad

Flight Data Table

| Airline | Source | Destination | Date of Journey | Duration | Total Stops | Price |
|---|---|---|---|---|---|---|
| Air India | Banglore | Delhi | 1/05/2019 | 2h 45m | non-stop | ₹6121 |
| Air India | Banglore | Delhi | 3/06/2019 | 2h 50m | non-stop | ₹6961 |
| Air India | Banglore | Delhi | 12/04/2019 | 2h 45m | non-stop | ₹5911 |
| Air India | Banglore | Delhi | 9/05/2019 | 2h 45m | non-stop | ₹5228 |
| Air India | Banglore | Delhi | 21/04/2019 | 2h 50m | non-stop | ₹5228 |
| Air India | Banglore | Delhi | 9/06/2019 | 2h 50m | non-stop | ₹6961 |
| Air India | Banglore | Delhi | 18/05/2019 | 2h 45m | non-stop | ₹6961 |

# CONCLUSION

This study demonstrates the effectiveness of using machine learning, supported by detailed data visualization, to predict flight fares with improved accuracy and interpretability. Airfare pricing, inherently influenced by various dynamic factors such as time of booking, flight duration, number of stops, and airline carrier, presents a complex regression problem. By implementing a structured pipeline that integrates data preprocessing, exploratory analysis, and model evaluation, we developed a robust framework for fare prediction.

One of the key strengths of this work lies in the emphasis placed on data visualization, which served as both a diagnostic and explanatory tool. Visual analysis helped uncover critical patterns in the dataset, such as the positive correlation between price and duration or stops, and significant pricing differences between airlines. These insights guided feature engineering decisions and contributed to the transparency of the machine learning models employed.

Two predictive models were implemented—Linear Regression and Random Forest Regressor. While the Linear Regression model served as a baseline, the Random Forest model delivered superior performance, achieving an $R^2$ score of 0.81 and a Mean Absolute Error (MAE) of approximately 1179 INR. These results confirm the suitability of ensemble learning techniques for capturing nonlinear and high-dimensional relationships in pricing data.

In addition to replicating the analyses found in existing literature, this project contributed original enhancements through additional plots, deeper insights from feature interaction studies, and improved outlier detection techniques. The successful application of machine learning and visualization in this context provides value to both travelers and stakeholders in the aviation industry by enabling data-informed decision-making.

Ultimately, this project reinforces the potential of combining machine learning with visual analytics to solve real-world pricing problems. It also provides a foundation for future enhancements, such as integrating real-time pricing data, incorporating external factors like seasonality or events, and exploring more advanced models such as gradient boosting or deep learning architectures.

# FUTURE WORK

While the current study successfully demonstrates the effectiveness of machine learning techniques in predicting flight fares, there remain numerous opportunities to expand and enhance the project in future iterations.

**1. Integration of Real-Time and External Data Sources**

The current model relies on static, historical data. In the future, integrating real-time flight fare APIs and web-scraped data from airline booking platforms would allow the system to generate live predictions and stay updated with current market trends. Additionally, incorporating external data such as economic indicators, weather conditions, geopolitical events, and holiday seasons could improve the model's context-awareness and forecasting accuracy.

**2. Advanced Modeling Techniques**

While Random Forest performed well, future work could explore more sophisticated models like Gradient Boosting Machines (e.g., XGBoost, LightGBM), Deep Neural Networks, or LSTM-based architectures for time-series prediction. These methods may offer better generalization, especially in capturing temporal dependencies and nonlinear feature interactions.

**3. Dynamic Model Updating and Online Learning**

Airfare pricing is a dynamic and continuously evolving process. Implementing an online learning framework that allows the model to retrain or adapt incrementally as new data arrives would make it more robust and responsive to market changes. This could be achieved through tools like incremental learning algorithms or scheduled batch retraining pipelines.

**4. Hyperparameter Optimization and AutoML**

Manual hyperparameter tuning can be time-consuming and suboptimal. Future improvements can include the use of automated hyperparameter tuning tools such as Grid Search, Random Search, or Bayesian Optimization. Additionally, integrating AutoML platforms may help discover optimal model configurations more efficiently.

By addressing these directions, future work can significantly enhance the practical utility, accuracy, and adaptability of flight fare prediction systems in real-world scenarios.

# REFERENCES

[1] K. Kim, S. Park, and J. Lee, "Understanding Open-Source Development Patterns via Descriptive Analytics," in *IEEE Access*, vol. 6, pp. 12345–12355, 2018.

[2] C. C. Aggarwal and C. K. Reddy, *Data Clustering: Algorithms and Applications*. Boca Raton, FL: CRC Press, 2013.

[3] W. Groves and M. Gini, "An agent for optimizing airline ticket purchasing," in *Proc. 12th Int. Conf. Autonomous Agents and Multiagent Systems*, pp. 1341–1342, 2013.

[4] J. S. Domínguez-Menchero, J. Rivera, and E. Torres-Manzanera, "Optimal purchase timing in the airline market," *J. Air Transp. Manag.*, vol. 40, pp. 137–143, 2014.

[5] K. Tziridis, T. Kalampokas, G. A. Papakostas, and K. I. Diamantaras, "Airfare price prediction using machine learning techniques," in *Proc. Eur. Signal Process. Conf. (EUSIPCO)*, pp. 1036–1039, 2017.

[6] S. Rajankar, N. Sakhrakar, and O. Rajankar, "Flight fare prediction using machine learning algorithms," in *Int. J. Eng. Res. Technol. (IJERT)*, vol. 8, no. 6, pp. 1–4, Jun. 2019.

[7] N. S. S. V. S. Rao and S. J. J. Thangaraj, "Flight Ticket Prediction using Random Forest Regressor Compared with Decision Tree Regressor," in *Proc. 8th Int. Conf. Sci. Technol. Eng. Math. (ICONSTEM)*, pp. 1–5, 2023.

[8] J. Hastie, T. Tibshirani, and R. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. New York, NY, USA: Springer, 2009.

[9] M. I. Jordan and T. M. Mitchell, "Machine learning: Trends, perspectives, and prospects," *Science*, vol. 349, no. 6245, pp. 255–260, 2015.

[10] P. Sarao and P. Samanta, "Flight Fare Prediction Using Machine Learning," *SSRN*, 2022. [Online]. Available: https://ssrn.com/abstract=4269263

[11] S. M. Joshitta, M. P. Sunil, A. Bodhankar, C. Sreedevi, and R. Khanna, "Integration of Machine Learning Technique with the Existing System to Predict Flight Prices," in *Proc. 3rd Int. Conf. Adv. Comput. Innov. Technol. Eng. (ICACITE)*, pp. 398–402, 2023.

[12] L. G. Valiant, "A theory of the learnable," *Commun. ACM*, vol. 27, no. 11, pp. 1134–1142, Nov. 1984.

[13] T. Liu, J. Cao, Y. Tan, and Q. Xiao, "ACER: An Adaptive Context-aware Ensemble Regression Model for Airfare Price Prediction," in *Proc. Int. Conf. Progress Informatics Comput.*, pp. 312–317, 2017.

[14] Y. S. Can and F. Alagöz, "Predicting Local Airfare Prices with Deep Transfer Learning Technique," in *Proc. Innovations in Intelligent Systems and Applications Conf. (ASYU)*, pp. 1–4, 2023.

[15] M. Malkawi and R. Alhajj, "Real-time Web-based International Flight Tickets Recommendation System via Apache Spark," in *Proc. IEEE Int. Conf. Information Reuse and Integration for Data Science (IRI)*, pp. 279–282, 2023.

**Kaggle Dataset Link:**
https://www.kaggle.com/datasets/nikhilmittal/flight-fare-prediction-mh
**Reasearch Paper Link:**
https://www.researchgate.net/publication/380296130_Flight_Fare_Prediction_Using_Machine_Learning