INDIAN INSTITUTE OF
INFORMATION TECHNOLOGY
NAGPUR

Flight Price Prediction

Machine Learning

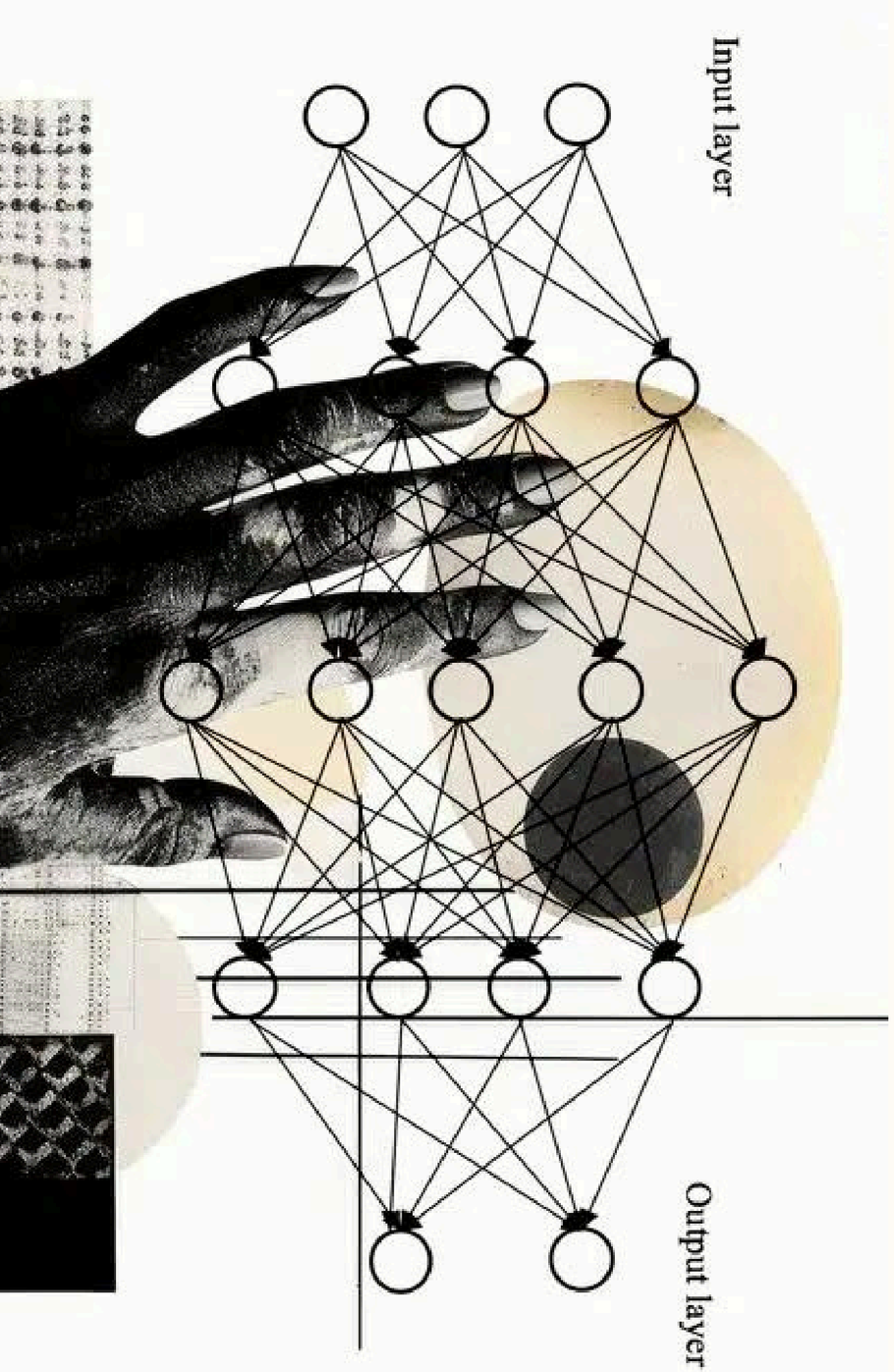
Team Members

BT21ECE012 Nehan Tanwar
BT21ECE054 Siddhant Panpatil

Under The Guidance of
Dr. Nikhil Dhengre

Overview

- Introduction
- Dataset Overview
- Data Preprocessing
- Feature Selection
- Model Training
- Model Evaluation
- Model Deployment
- Future Enhancements
- Summary and Conclusion



Introduction

- This project aims to develop a flight price prediction model using the Random Forest algorithm, leveraging historical flight data. Accurate price prediction benefits both travelers and airlines by enabling informed decision-making and optimized pricing strategies.

- By harnessing machine learning techniques, this project tackles the challenge of predicting flight prices accurately. The ability to forecast prices empowers travelers to make budget-friendly bookings and assists airlines in revenue optimization and efficient resource allocation.

Dataset Overview:

Data Characteristics:

- The dataset contains flight details such as airline names, source and destination cities, dates and times of flights, and ticket prices.
- After dropping missing values, the dataset was further processed to extract useful features such as the day and month of the journey, departure and arrival times, and the duration of the flights.

Feature Engineering:

- Date-related features were extracted by converting the 'Date_of_Journey', 'Dep_Time', and 'Arrival_Time' columns into their respective components (day, month, hour, minute).
- The 'Duration' column was split into 'Duration_hours' and 'Duration_mins', providing more granular insights into flight times.

Dataset Overview :

Categorical Data Transformation:

- Categorical data such as 'Airline', 'Source', and 'Destination' was converted into numerical data using one-hot encoding.
- The 'Total_Stops' column was also transformed into numerical values based on the number of stops, revealing a positive correlation with flight price.

```ipynb

train_data = pd.read_excel('Flight Dataset/Data_Train.xlsx')
train_data.dropna(inplace=True)

```

Data Preprocessing :

Handling Missing and Inconsistent Data:

- All rows containing missing data were dropped to ensure the data was clean and ready for further analysis.
- Inconsistent values in the 'Destination' and 'Source' columns were standardized (e.g., 'New Delhi' to 'Delhi').

- **Feature Engineering and Transformation:**
- Date and time-related features were extracted to get insights into travel patterns. Columns like 'Date_of_Journey', 'Dep_Time', and 'Arrival_Time' were transformed into separate features such as 'Journey_day', 'Journey_month', 'Dep_hour', 'Dep_min', 'Arrival_hour', and 'Arrival_min'.
- Flight duration was split into 'Duration_hours' and 'Duration_mins', providing numerical insights into the length of flights.

Data Preprocessing :

Categorical Data Encoding:

- The categorical features such as 'Airline', 'Source', and 'Destination' were converted to one-hot encoded numerical representations for easier modeling.
- The 'Total_Stops' column was transformed into a numerical representation indicating the number of stops, correlating directly with the flight price.

```ipynb
train_data.dropna(inplace=True)
train_data['Destination'] = train_data['Destination'].replace('New Delhi', 'Delhi')

train_data['Journey_day'] = pd.to_datetime(train_data['Date_of_Journey'], format='%d/%m/%Y').dt.day
train_data['Dep_hour'] = pd.to_datetime(train_data['Dep_Time']).dt.hour

airline = pd.get_dummies(train_data[['Airline']], drop_first=True)
```

Feature Selection:

Correlation Analysis:

- A heatmap was used to visualize the correlation between features to identify which features have strong relationships with the target variable, aiding in feature selection.

Feature Importance:

- The importance of each feature was calculated using the ExtraTreesRegressor model, which assigns importance scores to features based on their impact on predictions.

```
plt.figure(figsize=(10, 10))
sns.heatmap(train_data.corr(), cmap='viridis', annot=True)
```

Feature Selection:

Feature Selection Techniques:

- Techniques such as SelectKBest and feature importance rankings from tree-based algorithms were used to identify the most significant features for predicting flight prices.

```ipynb
reg = ExtraTreesRegressor()
reg.fit(X, y)
feat_importances = pd.Series(reg.feature_importances_, index=X.columns)
feat_importances.nlargest(20).plot(kind='barh')
plt.show()
```

Model Training:

Train-Test Split:

- The dataset was split into training and test sets to evaluate model performance. The train_test_split function from scikit-learn was used, with a typical 70-30 or 80-20 split between training and testing data.

Model Selection and Hyperparameter Tuning:

- A Random Forest Regressor was chosen for its robustness and efficiency. The RandomizedSearchCV method was employed to fine-tune hyperparameters such as the number of trees, maximum depth, and features considered at each split.

```

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=42)

```

Model Training:

Hyperparameter Tuning:

- RandomizedSearchCV was used to search for the optimal combination of hyperparameters, such as the number of trees, maximum depth, minimum samples split, and maximum features to consider at each split.
- By exploring various combinations, the model was fine-tuned to deliver the best performance on unseen data.

```
rf_random = RandomizedSearchCV(estimator=RandomForestRegressor(),
param_distributions=random_grid,
                               scoring='neg_mean_squared_error', n_iter=10, cv=5,
                               verbose=1, random_state=42, n_jobs=1)
rf_random.fit(X_train, y_train)
```

Model Evaluation:

Prediction and Residual Analysis:

- The trained model was used to predict prices on the test set, and residuals (the difference between predicted and actual prices) were analyzed to assess the model's accuracy.
- A distribution plot was used to visualize the spread of residuals, providing insight into the distribution and magnitude of prediction errors.

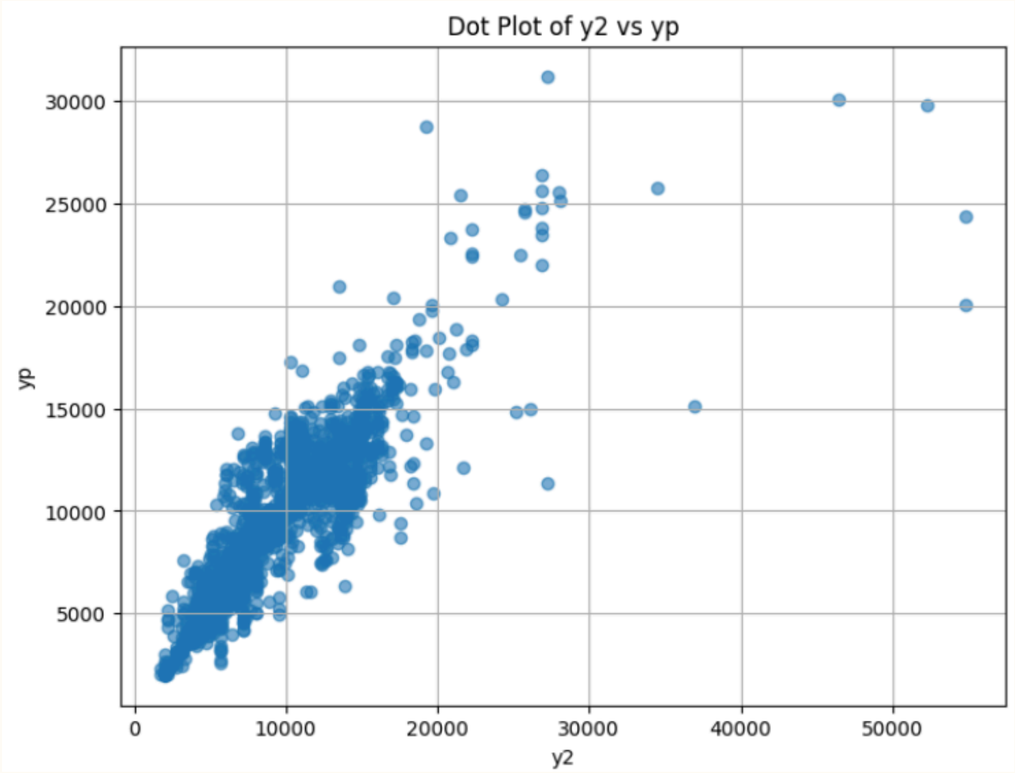
Performance Metrics:

- The R^2 score was calculated to quantify the proportion of variance in the dependent variable that the model could explain. A higher R^2 score indicates better model performance.
- Other evaluation metrics such as mean absolute error (MAE) and root mean squared error (RMSE) could also be considered to measure prediction accuracy.

Model Evaluation:

- **Visual Comparison:**
- A scatter plot was used to compare predicted vs. actual values. The closer the points lie to the diagonal line, the better the model performed.

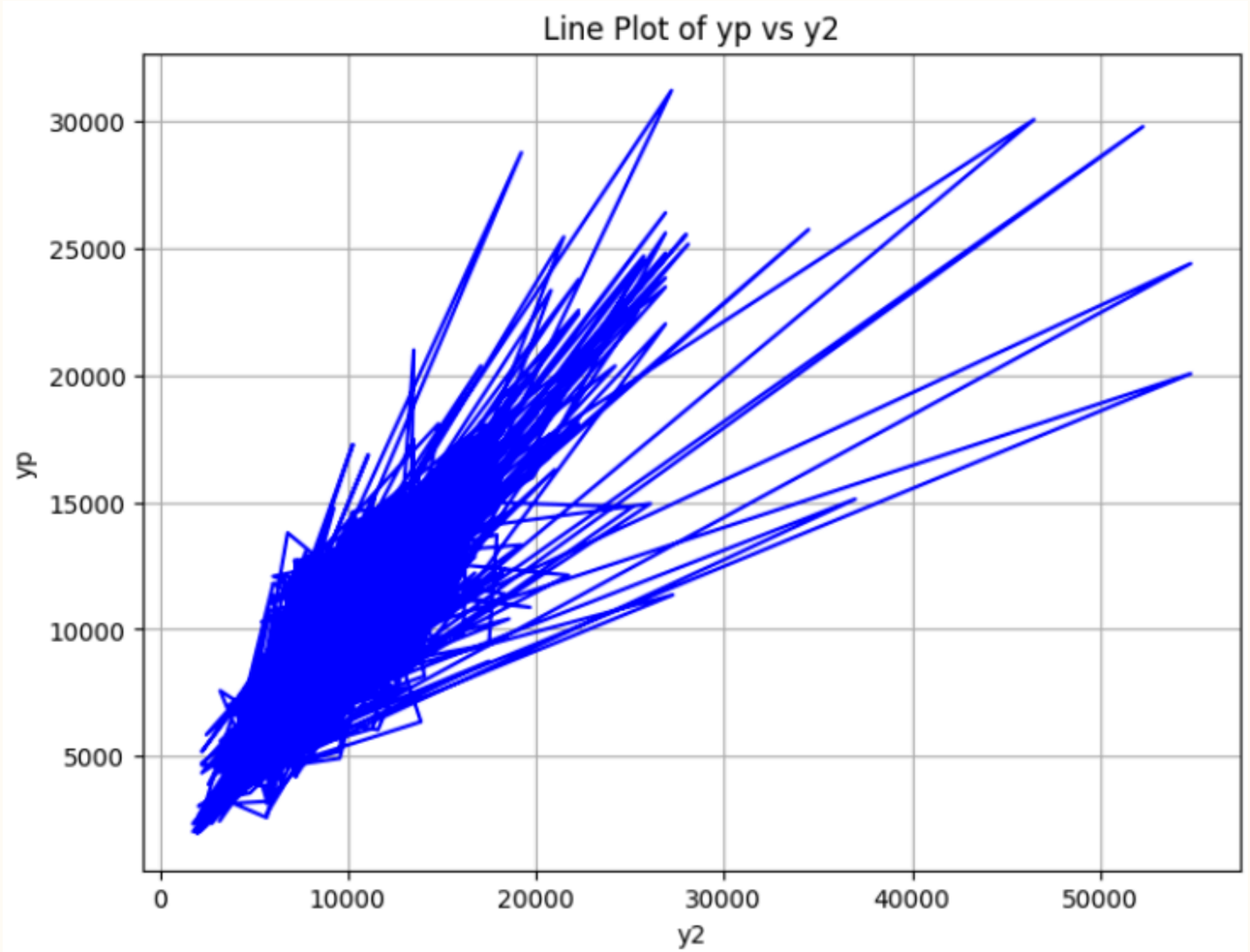
```
plt.figure(figsize=(8, 6))
plt.scatter(y2, yp,alpha=0.6)
plt.xlabel('y2')
plt.ylabel('yp')
plt.title('Dot Plot of y2 vs yp')
plt.grid(True)
plt.show()
```



Model Evaluation:

Line Plot of yp vs y2

```
plt.figure(figsize=(8, 6))
plt.plot(y2, yp, color='blue')
plt.xlabel('y2')
plt.ylabel('yp')
plt.title('Line Plot of yp vs y2')
plt.grid(True)
plt.show()
```



Model Deployment:

Interface Implementation with Streamlit:

- The trained model was saved and plans were made to develop a user-friendly interface using the Streamlit framework. This would allow users to input flight details and receive fare predictions through a web application.
- Streamlit simplifies the process of creating interactive data apps, making it suitable for quickly deploying machine learning models for practical use.

Model Export for Production:

- The model was serialized using the pickle library, enabling it to be easily loaded and used in different environments. This facilitates deployment in web servers or cloud platforms, allowing users to obtain predictions in real-time.

Code Snippet for Model Deployment:

- **Save the trained model to a file for future use**

```
with open('flight_rf.pkl', 'wb') as file:
    pickle.dump(rf_random, file)
```

Future Enhancements:

Additional Feature Incorporation:

- Integrating more features like weather data, holiday schedules, or airline-specific factors could enhance the model's predictive accuracy.
- Incorporating real-time data on flight bookings and availability could improve the model's responsiveness to current trends.

Improving Model Accuracy:

- Experimenting with advanced machine learning models like XGBoost or ensemble methods could further enhance prediction accuracy.
- Hyperparameter optimization through more extensive grid searches or automated tools like Optuna can refine model performance.

Future Enhancements:

Expanded Deployment and User Interface:

- Improving the Streamlit interface with more features, such as suggesting alternative flights and fares based on predictions, would enhance user interaction.

- Integrating with other platforms (like mobile apps or chatbots) could make the model's predictions more accessible and user-friendly.

Summary and Conclusion:

Project Summary:

- This project involved developing a machine learning model to predict flight prices using historical flight data. The process included data preprocessing, feature engineering, model selection, and training using a Random Forest Regressor. The model was evaluated based on its predictive accuracy and visualized through plots.

Conclusion:

- The project successfully demonstrated the effectiveness of machine learning in predicting flight prices. The model showed reasonable accuracy and provided insights into important features affecting flight prices. With further enhancements and deployment through a user-friendly interface, the model could be a valuable tool for users seeking to optimize their travel expenses.

Q&A Session

Thank you!