Predicting Flight Delays Using Machine Learning

Karan Deo Burnwal, Turanyaa Garg

Department of Electrical Engineering, IIT Delhi

Abstract

This project aims to predict flight delays using historical data and machine learning models. Data preprocessing, feature engineering, and model selection are employed to enhance prediction accuracy. Various models are compared, with a focus on balancing performance and interpretability.

Keywords
flight delay prediction, machine learning, feature engineering, classification

Github: Link

1 Introduction

Flight delays are costly and disruptive for airlines and passengers alike. The objective of this study is to develop a predictive model based on historical flight data to anticipate delays. Using machine learning techniques, we aim to identify key factors influencing delays and assess their predictive power.

2 Dataset

2.1 Preprocessing

The raw flight data underwent several preprocessing steps to prepare it for modeling:

2.1.1 Handling Categorical Variables

- Identified key categorical columns: `UniqueCarrier`, `Origin`, and `Dest`
- Handled category mismatches between train and test sets by:
  - Replacing categories present in training but absent in test set with 'Unknown'
  - Replacing categories present in test but absent in training set with 'Unknown'
- Applied one-hot encoding using `OneHotEncoder` with `drop='first'` to avoid multicollinearity

2.1.2 Data Cleaning

- Removed `'c-'` prefix from temporal columns:
  - `Month`
  - `DayofMonth`
  - `DayOfWeek`
- Converted these columns to integer type
- Transformed target variable `dep_delayed_15min` to binary format:
  - 1 for 'Y' (delayed)
  - 0 for 'N' (not delayed)

2.1.3 Feature Engineering

- Extracted temporal features from `DepTime`:

$$\texttt{DepHour} = \lfloor \frac{\texttt{DepTime}}{100} \rfloor$$
$$\texttt{DepMinute} = \texttt{DepTime} \bmod 100$$

- Created distance categories using the following bins:
  - 0: [0, 500] miles
  - 1: (500, 1000] miles
  - 2: (1000, 1500] miles
  - 3: $>$ 1500 miles

2.1.4 Class Imbalance Treatment

- Applied Synthetic Minority Over-sampling Technique (SMOTE)
- Used `random_state=42` for reproducibility

2.1.5 Data Splitting

- Implemented train-validation split:
  - 80% training set
  - 20% validation set
- Used stratified splitting to maintain class distribution

Feature Name	Description
Month	Month of departure
DayofMonth	Day of the month
DayOfWeek	Day of the week
DepTime	Departure time (HH)
UniqueCarrier	Airline code
Origin	Origin airport code
Dest	Destination airport code
Distance	Distance in miles
dep_delayed_15min	15+ minutes delay

Table 1: Feature Description

Feature Name	Description
DepHour	Hour of the departure time, extracted from 'DepTime'.
DepMinute	Minute of the departure time, extracted from 'DepTime'.
DistanceCategory	Binned distance between origin and destination: 'Short', 'Medium', 'Long', 'Very Long'.

Table 2: New Features in the Dataset

- Applied splitting to both:
  - Original dataset
  - SMOTE-resampled dataset

3 Model Performance Insights

Our analysis evaluated ten different machine learning models, each tested on both original and resampled datasets.

The performance hierarchy reveals interesting insights:

- **Top Tier** (AUC-ROC > 0.90):
  - Random Forest (Resampled): 0.9361.
  - AdaBoost (Resampled): 0.9334.
  - Gradient Boosting (Resampled): 0.9330.
  - Decision Tree (Resampled): 0.9085.
- **Mid Tier** (AUC-ROC 0.70–0.90):
  - KNN (Resampled): 0.8731.
  - Logistic Regression (Resampled): 0.7258.
  - LDA (Resampled): 0.7308.
- **Lower Tier** (AUC-ROC < 0.70):
  - GaussianNB (Original): 0.5707.
  - QDA (Original): 0.5119.

3.1 Impact of Model Complexity

- **Simple Models**: Linear and probabilistic models struggled on both datasets:
  - Logistic Regression (Original): AUC-ROC of 0.7078.
  - LDA (Original): AUC-ROC of 0.7062.
  - QDA (Original): AUC-ROC of 0.5119.
- **Intermediate Complexity**:
  - Decision Tree (Resampled): AUC-ROC of 0.9085, demonstrating significant improvement with resampling.
  - KNN (Resampled): AUC-ROC of 0.8731, highlighting its effectiveness post-resampling.
- **Complex Ensemble Methods**:
  - Random Forest (Resampled): AUC-ROC of 0.9361, the top performer overall.
  - AdaBoost (Resampled): AUC-ROC of 0.9334, showing robust performance.
  - Gradient Boosting (Resampled): AUC-ROC of 0.9330, maintaining high consistency.

3.2 AUC-ROC Plots for Original and Resampled Datasets

The AUC-ROC curves illustrate the performance of AdaBoost, Gradient Boosting, and Random Forest models on both the original and resampled datasets. These plots provide insights into the models' trade-offs between sensitivity and specificity.

3.3 Class Imbalance and Resampling Effects

3.3.1 Original Dataset Performance

The original dataset's class imbalance led to significant challenges:

- **Biased Predictions**: High accuracy but poor F1 scores for certain models:
  - Random Forest: F1 score of 0.0010 despite 0.8096 accuracy.
  - Extra Trees: F1 score of 0.0000 despite 0.8095 accuracy.
- **Underperforming Models**: Probabilistic models like GaussianNB and QDA showed limited utility.

Model	Accuracy	F1 Score	AUC-ROC Score
Random Forest	0.8096	0.0010	0.7306
Gradient Boosting	0.8175	0.1028	0.7266
AdaBoost	0.8146	0.1377	0.7215
Decision Tree	0.8132	0.2625	0.6836
Logistic Regression	0.8145	0.1000	0.7078
Linear Discriminant Analysis (LDA)	0.8123	0.1225	0.7062
k-Nearest Neighbors (KNN)	0.7870	0.1689	0.5996
Extra Trees	0.8095	0.0000	0.7062
Gaussian Naive Bayes	0.3856	0.3299	0.5707
Quadratic Discriminant Analysis (QDA)	0.2316	0.3235	0.5119

Table 3: Model Performance on Original Dataset

Model	Accuracy	F1 Score	AUC-ROC Score
Random Forest	0.8802	0.8662	0.9361
AdaBoost	0.8825	0.8695	0.9334
Gradient Boosting	0.8809	0.8668	0.9330
Extra Trees	0.8392	0.8265	0.9088
Decision Tree	0.8491	0.8279	0.9085
k-Nearest Neighbors (KNN)	0.7779	0.8078	0.8731
Linear Discriminant Analysis (LDA)	0.6723	0.6785	0.7308
Logistic Regression	0.6678	0.6733	0.7258
Gaussian Naive Bayes	0.5628	0.6690	0.6058
Quadratic Discriminant Analysis (QDA)	0.5199	0.6710	0.5202

Table 4: Model Performance on Resampled Dataset

3.3.2 Resampling Impact

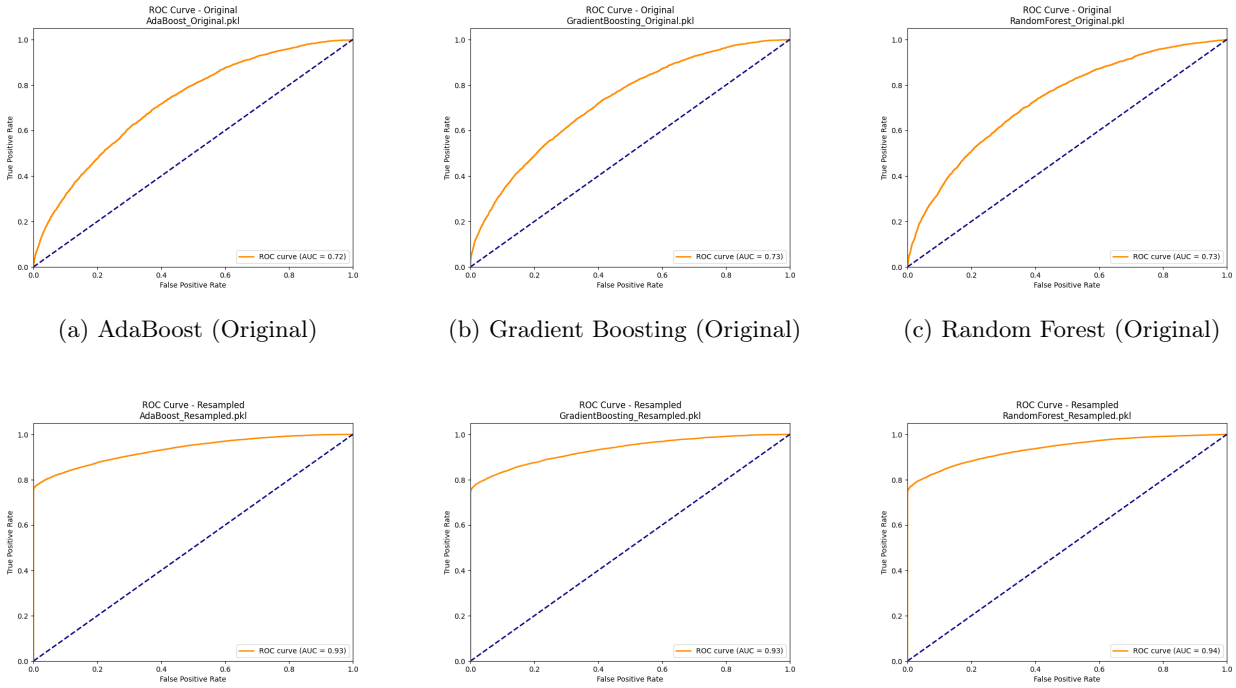
Resampling substantially improved model performance:

- **Positive Impacts**:
  - Boosted ensemble methods (Random Forest, AdaBoost, Gradient Boosting).
  - Enhanced performance for Decision Trees and KNN.
- **Mixed Results**:
  - Minor improvements for linear models like Logistic Regression and LDA.
  - GaussianNB and QDA remained relatively weak performers.

4 Practical Implementation Considerations

4.1 Model Selection Trade-offs

- **Random Forest (Resampled)**: Top performer with balanced accuracy, F1 score, and AUC-ROC, suitable for robust applications.
- **AdaBoost (Resampled)**: Offers similar performance to Random Forest but may be more sensitive to noise.
- **Gradient Boosting (Resampled)**: Consistent high performance with moderate computational demands.
- **KNN (Resampled)**: Strong F1 score and AUC-ROC but may face scalability issues with larger datasets.



(a) AdaBoost (Original) (b) Gradient Boosting (Original) (c) Random Forest (Original)

(d) AdaBoost (Resampled) (e) Gradient Boosting (Resampled) (f) Random Forest (Resampled)

Figure 1: AUC-ROC Curves for AdaBoost, Gradient Boosting, and Random Forest on Original and Resampled Datasets

4.2 Resource Requirements

- **Memory Usage**: Ensemble methods like Random Forest and AdaBoost require significant memory for model storage.
- **Computation Time**: Gradient Boosting and KNN require careful optimization for training and prediction phases.
- **Real-Time Suitability**: Logistic Regression and Decision Trees offer faster inference, making them viable for real-time applications.

5 Limitations and Future Opportunities

5.1 Current Limitations

The analysis highlights several challenges:

- **Data Quality Issues**: Imbalanced dataset heavily impacts results. Model performance is dependent on data quality.
- **Feature Engineering**: Weather data and traffic patterns not integrated

5.2 Future Directions

Enhancements can be pursued in the following areas:

- **Data Integration**: Incorporate real-time updates and advanced features (e.g., weather, holiday schedules, passenger load factors, crew availability, Aircraft maintenance).
- **Model Optimization**: Explore hybrid models and deep learning techniques for further gains.
- **Scalability**: Develop optimized pipelines for deployment across large-scale datasets and systems incorporating real-time prediction capability.

References

- Yuemin Tang. Airline Flight Delay Prediction Using Machine Learning Models. University of Southern California, USA.
- Juan Pineda-Jaramillo, Claudia Munoz, Rodrigo Mesa-Arango, Carlos Gonzalez-Calderon, Anne Lange. Integrating Multiple Data Sources for Improved Flight Delay Prediction Using Explainable Machine Learning.
- Kaggle Dataset: Flight Delay Prediction. Available at: https://www.kaggle.com/.
- Airline Flight Delay Prediction Using Machine Learning Models. Scientific Reports. **2024**. Available at: https://doi.org/10.1038/s41598-024-55217-z.