Evaluating Models For Predicting Flight Delays Using ROC-AUC Metrics

Jayesh Pamnani
University of Maryland, College Park
College Park, USA
jpamnani@umd.edu

***Abstract*—This paper focuses on solving the problem of predicting flight delays in the aviation industry by providing a comparative analysis of two machine learning models: logistic regression and decision tree, applied to a dataset of historical flight data. The results reveal that logistic regression outperforms the decision tree model in terms of accuracy (76.3% compared to 62.6%) and the area under the receiver operating characteristic curve (0.700 compared to 0.620). This finding underscores the practical significance of this study for aviation professionals seeking accurate flight delay prediction models.**

***Index Terms*—Decision Tree, Logistic Regression, Receiver Operating Characteristic Curve, Area Under Curve**

I. Introduction

Forecasting whether a flight will experience delays is a decision that carries significant consequences for both airlines and passengers. Inaccurate forecasts can lead to financial losses, operational inefficiencies, and passenger inconvenience, underscoring the importance of precise delay predictions in the aviation industry. In this paper, the author addresses the critical problem of predicting flight delays by selecting the most suitable model from Logistic Regression and Decision Tree through the evaluation of the Area under the Receiver Operating Characteristic Curve (ROC-Curve) metric. The ROC curve illustrates the trade-off between the true positive rate (TPR) and the false positive rate (FPR) across various classification thresholds [1].

Our analysis relies on a dataset comprising historical flight data from the year 2002, encompassing 1500 samples. This dataset includes various attributes such as scheduled departure and arrival times, departure and arrival airports, flight duration, distance covered, airline information, and other pertinent features. Known as the ”flights and airports dataset” [2], it is provided by the U.S. Department of Transportation's (DOT) Bureau of Transportation Statistics, specifically tracking the on-time performance of domestic flights operated by large air carriers. This dataset serves as the cornerstone for constructing and evaluating our machine learning models, which are designed to accurately predict flight delays. Throughout the paper, this dataset is leveraged to forecast whether a flight will experience delays or not, focusing on specific parameters within the dataset. Our research contributes to the ongoing discourse on model selection and validation in the context of real-world datasets.

The paper is structured into four key sections to provide a well-organized and comprehensive guide for selecting the most suitable algorithm to predict flight delays. The 'Introduction' sets the stage by introducing the problem definition, the algorithms used, and the paper's objectives. 'Methodology' delves into technical details, covering model initialization methods, ROC-Curve implementation, and subsequently, the area under the curve and metrics calculation. In the 'Results' section, the author presents the metrics obtained for each of the models. The 'Discussion' critically examines these results and highlights the best model for the flight dataset, suggesting potential directions for future research.

II. Methodology

In this section, the methodological approach undertaken for the evaluation of machine learning models in the context of the flight and airport dataset is outlined. The Decision Tree Model [3] and the Logistic Regression Model [4] are trained on this dataset, and subsequently, the ROC Curve is plotted. Following this, the area under the curve (AUC) is calculated for the two models. The numerical range for the area under this curve is between 0 and 1, providing a measure of a model's overall performance. A greater AUC indicates a better balance, with a higher true positive rate and a low false positive rate. Maximizing the area under the curve is crucial, as it measures the likelihood that the model will assign a higher predicted probability to a randomly selected positive instance compared to a randomly chosen negative instance.

The following subsections will elaborate on the steps involved in data cleaning, versions of software used, data preprocessing, data processing, and result validation.

A. Data Cleaning

The flight data for the year 2002 included various attributes such as Year, Month, DayofMonth, DayOfWeek, DepTime, CRSDepTime, ArrTime, CRSArrTime, UniqueCarrier, Name, FlightNum, TailNum, ActualElapsedTime, CRSElapsedTime, AirTime, ArrDelay, DepDelay, Origin, Dest, Distance, TaxiIn, TaxiOut, Cancelled, and diverted fields, CarrierDelay, WeatherDelay, NASDelay, SecurityDelay, and LateAircraftDelay, some of which had missing values.

To ensure the effectiveness of our model in predicting outcomes for unseen data, the focus was exclusively on the scheduled data, excluding real-time information. Consequently, DepTime, ArrTime, AirTime, TaxiIn, and TaxiOut were removed from the dataset. Additionally, TailNum was eliminated, recognizing that changes in this attribute could occur without affecting other critical flight information such as scheduled departure, origin, or destination. Furthermore, Cancelled, Diverted, as well as CarrierDelay, WeatherDelay, NASDelay, SecurityDelay, and LateAircraftDelay fields were omitted, as these data points were not available for prediction purposes. For this paper, the author has taken 500 records each for the months 1, 2, and 3 of the dataset, totaling 1500 samples.

B. Versions Used

The methodology employed in this study is grounded in Python, utilizing version 3.11.2, and relies on crucial libraries such as sklearn (version 1.3.0), NumPy (version 1.25.2) and Pandas (version 2.1.0) for data manipulation and analysis.

C. Data Preprocessing

The CSV file was preprocessed using the 'genfromtxt()' function in NumPy, as detailed in [3]. The arrDelay field from the flight dataset was chosen as the target variable. Notably, the arrDelay field in the flight dataset takes a negative value when a flight reaches its destination ahead of schedule and a positive value when it arrives late. To create a binary classification for delays, instances were examined to see if the 'arrDelay' value exceeded the value of 5 minutes, introducing a buffer of 5 minutes. However, some samples had 'NA' as the value for this column. To handle missing values, a preprocessing step was introduced where the 'NA' entries were replaced with 0 for the affected samples, using the 'np.where()' function from the NumPy array library.

Listing 1. Converting NA Entries in the Target Column to 0

```
y = np.where(y == 'NA', 0, y)
```

Following this, the main classification process was executed using the same 'np.where()' function, differentiating delays by evaluating if the 'arrDelay' value exceeded 5 minutes. In this binary classification, delays were assigned a value of 1, while non-delays were assigned a value of 0.

Listing 2. Converting Target Variable to Binary

```
y = np.where(y>5,1,0)
```

The 'where()' function in NumPy replaces each element in the original array with the second parameter if the boolean condition (specified as the first parameter) is true. Conversely, if the condition is false, it assigns the value specified as the third parameter.

The dataset consists of string values in the 'origin' and 'destination' columns. Many algorithms, such as logistic regression or decision trees, require numerical input. By encoding categorical variables, the model can effectively learn patterns and relationships present in the data. To achieve this, encoding was performed using the code below:

Listing 3. Encoding

```
encoder = OrdinalEncoder()
X_8 = x[:,8].reshape(-1, 1)
X_8_encoded = encoder.fit_transform(X_8)
X_9 = x[:,9].reshape(-1, 1)
X_9_encoded = encoder.fit_transform(X_9)
x = np.hstack((x, X_8_encoded, X_9_encoded))
x = np.delete(x, [8, 9], axis=1)
```

'Origin' is the 8th column of the dataset, and the 'destination' is the 9th column (0-indexed). The encoder utilized for converting categorical data to numeric values is the OrdinalEncoder [5], a component of scikit-learn's preprocessing library. It operates by taking a 2D NumPy array as input and subsequently converts the categorical values into a range of integers, typically from 0 to n-1, where "n" represents the number of unique categories within the column.

Following the encoding process, the newly generated numeric values are incorporated into the dataset. This integration is achieved through the use of the np.hstack() function, a part of the NumPy library that horizontally stacks arrays, effectively concatenating the encoded values with the original feature array.

Subsequently, the code utilizes the np.delete() function from the NumPy library to remove the original "Origin" and "Destination" string columns from the feature array. This step is essential as the encoded values have replaced the categorical data, optimizing the dataset for compatibility with a logistic regression model.

Thereafter, the dataset was divided into training and testing sets, adhering to the 85%/15% split and employing the methodology outlined in [3].

D. Data Processing

The decision tree and logistic regression models were trained on a training dataset, consistent with the approach established in [3] and [4]. Their accuracy scores and predictions are then calculated along with the ROC Curve [6].

Listing 4. Calculating characteristics of ROC curve

```
metrics.roc_curve(y_test, y_pred)
```

The above function is a part of the scikit-learn library and accepts two parameters:

1) y_test: The true labels (ground truth) of the test set.
2) y_pred: The predicted probability scores or binary predictions from the model.

The function returns three arrays:

1) fpr (False Positive Rate): The x-axis values of the ROC curve.
2) tpr (True Positive Rate or Sensitivity): The y-axis values of the ROC curve.
3) thresholds: The decision thresholds used to calculate the TPR and FPR.

To calculate the area under the curve generated from the above steps, the FPR and TPR values obtained as a result of the roc_curve function are passed as parameters to the

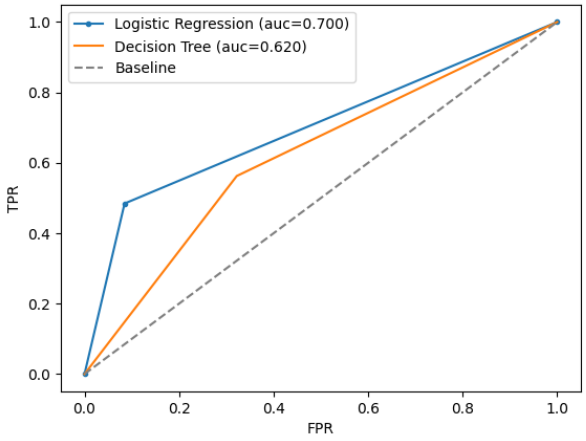

Fig. 1. Receiver Operating Characteristic Curve

metrics.auc() function, which is also a part of the scikit-learn library. This function calculates the area under the ROC curve, representing the probability that a classifier will correctly identify a positive instance.

Listing 5. Calculating Area under Curve

```
metrics.auc(fpr, tpr)
```

The function returns a single value, which represents the area under the ROC curve. A higher AUC generally indicates better model performance.

This approach was implemented on the logistic regression and decision tree models, generating their corresponding ROC curves. Following that, the characteristics for both models were extracted and plotted alongside a baseline model curve, which represents a 50% chance of correct prediction, facilitating visual comparison. Matplotlib was utilized for plotting, adhering to the methodology detailed in [7].

E. Result Validation

The confusion matrix was calculated by the author to check the number of true positives and true negatives.

III. Results

The predictions from the model were tested against the testing dataset, which comprised 15% of the data.

A. Accuracy

The Decision Tree model exhibited an accuracy of 62.6%, whereas the Logistic Regression model achieved a higher accuracy at 76.3%.

B. Area Under the Receiver Operating Characteristic Curve

The area under the Receiver Operating Characteristic curve for the Decision Tree Model was 0.620, while for the Logistic Regression Model, it was 0.700.

The visual analysis of the ROC curves suggests that the Logistic Regression model has a larger area under the curve and a steeper slope compared to the Decision Tree model. This indicates that the Logistic Regression model performs better in distinguishing between positive and negative instances and has a higher discriminatory power than the Decision Tree model.

C. Results Discussion

The higher area under the ROC curve in the logistic regression model, when compared with the decision tree model, implies a superior suitability for the dataset. Additionally, the steep slope of the logistic regression curve signifies heightened responsiveness to variations in the input features, allowing for more accurate classifications.

IV. Discussion

The analysis of the area under the ROC curve indicates that both the logistic regression and decision tree models surpass random guessing, as their curves consistently lie above the baseline. Although both models exhibit superior performance, with their ROC curves distinctly above the diagonal line, the logistic regression model demonstrates a notably higher area under the ROC curve (AUC) compared to the decision tree model. This suggests a heightened ability to distinguish between positive and negative instances. The steeper slope of the logistic regression curve further signifies increased sensitivity to changes in input features, resulting in more precise classifications. Overall, the logistic regression model emerges as the preferred choice for prediction, aligning with expectations based on its inherent characteristics in handling binary classification tasks and capturing complex relationships in the data.

Accurate delay predictions are critical for airlines and passengers to mitigate financial losses, operational inefficiencies, and inconvenience. The superior performance of the logistic regression model emphasizes its potential as a reliable tool in predicting flight delays, contributing valuable insights to aviation professionals.

To further enhance the robustness of the predictive models and extend their applicability, future research could explore several avenues. Firstly, the study focused on historical flight data for the year 2002, taking only 1500 samples from the dataset; expanding the dataset by considering all samples and incorporating more recent information could enhance the models' relevance in a dynamic aviation landscape. Secondly, incorporating delays due to weather conditions could elevate the model's utility in real-life scenarios. Addressing these aspects would contribute to the continuous refinement of predictive models for more accurate and timely flight delay predictions.

REFERENCES

[1] https://www.geeksforgeeks.org/auc-roc-curve/
[2] https://www.kaggle.com/datasets/usdot/flight-delays
[3] J. Pamnani, A. Ketkar, J. Hasija, and D. Lnu, Decision Tree Analysis on Iris Dataset
[4] J. Pamnani, A. Ketkar, J. Hasija, and D. Lnu, Classification using Logistic Regression
[5] https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OrdinalEncoder.html
[6] https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html
[7] J. Pamnani, A. Ketkar, J. Hasija, and D. Lnu, Python Essential Training: A Comprehensive Guide