Predicting PIT Flight Delays

36-662: Final Project Write-up

Perry Lin and Andrew Shih

2025-04-25

1. Introduction

Flight delays remain a persistent challenge in the air transportation industry, affecting passengers, airline operations, and downstream connections. In this project, we aim to build a predictive model to estimate whether a flight departing from Pittsburgh International Airport (PIT) will be delayed by 15 minutes or more. Using historical flight data from 2022 and 2023 as our training base, we use domain knowledge to engineer features and build a random forest model to predict 2024 departure delays.

The task is framed as a binary classification problem: predicting whether a delay will occur. The aim of the model is to produce predicted delay probabilities for a held-out set of 2024 flights with missing delay labels. These probabilities are evaluated using two metrics: 1) the error rate on the held-out set, and 2) the area under the ROC curve (AUC), which rewards probabilistic predictions over binary guesses.

Accurate predictions can help travelers make informed decisions and allow airports or airlines to manage disruptions more proactively.

2. Data Exploration

We began with an exploration of the 2022 and 2023 flight records to establish a baseline understanding and assess the nature of delays. Combining the flight data from the two years, there are 137,425 flights recorded. The dataset describes several features about each flight including ones related to the time of flight, the duration, the origin and destination, the carrier, and many more. The following subsections outline notable characteristics of the data that later informed our preprocessing and feature engineering efforts.

Initial Findings and Data Filtering

- The target variable is binary: `DEP_DEL15` indicates whether a flight was delayed by 15 or more minutes.
- We restricted our attention to flights *departing* from Pittsburgh, as our intent is to predict departure delay. Flights that were cancelled were excluded, as their value for `DEP_DEL15` would be 'NA'. Following these filters, there were 67,099 flights to be used in the training set for the eventual model.

Class Imbalance

- In Figure 1, non-delayed flights outnumber delayed flights significantly. Over 80% of flights are not delayed (which we surmise to be a percentage that hovers around this area on unseen data), so a base classifier that predicts every flight to have `DEP_DEL15` $= 0$ already performs relatively well.

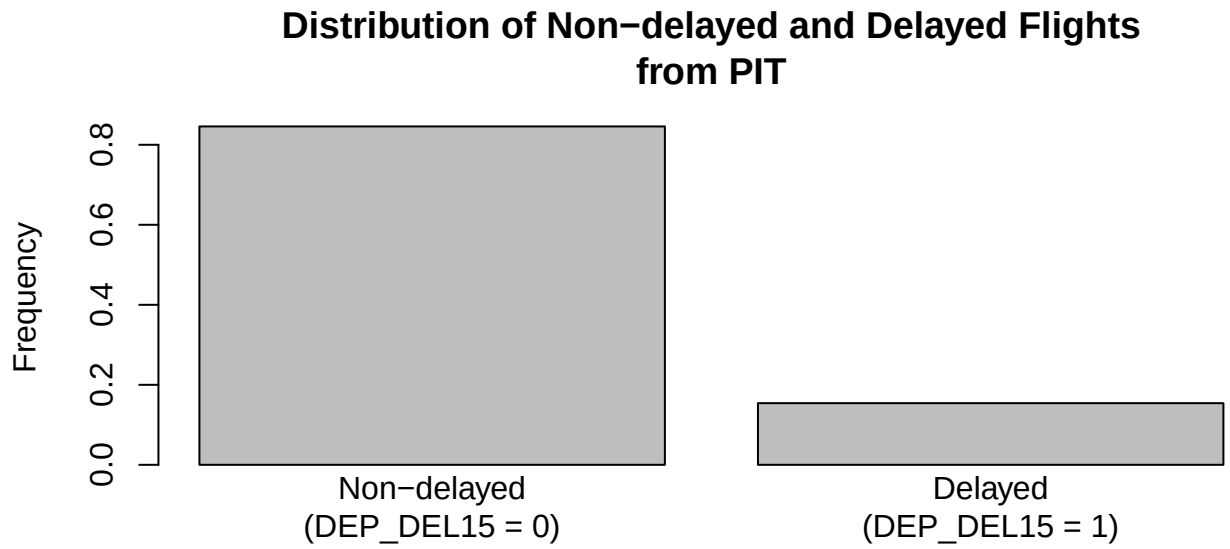

Figure 1: From PIT, delayed flights outnumber non-delayed flights.

Temporal Insights

- We found little correlation between 1) the scheduled arrival time `CRS_ARR_TIME` and whether there was a delay, and 2) the scheduled departure time `CRS_DEP_TIME` and whether there was a delay.
- In Figure 2, delay probabilities appeared to vary across the day of the month, by the day of the week, and by month. Based on an analysis of the distributions of delayed flights across different temporal stratifications, there appears to be some temporal dependence.

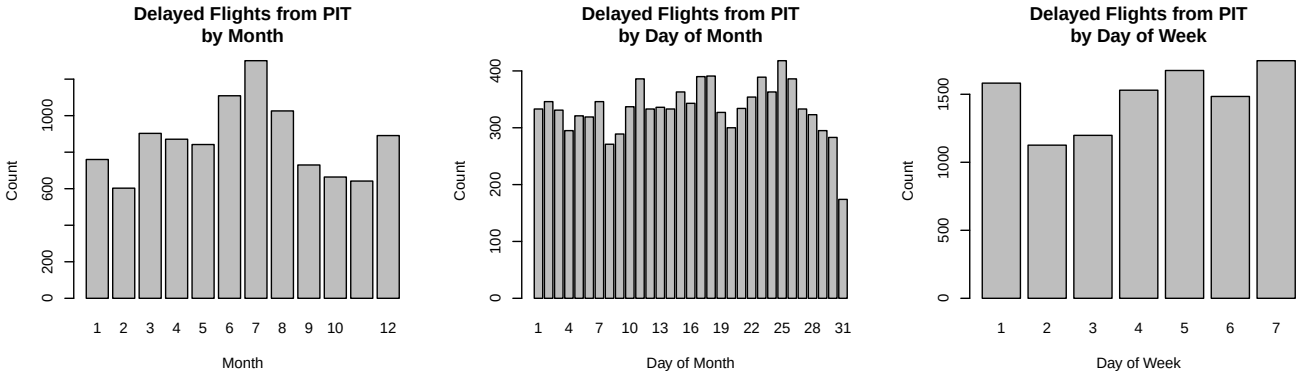

Figure 2: There is an observable temporal dependence in flight delays from PIT.

Carrier Insights

- We believed that the airline with which a particular flights is associated with delay probability. Figure 3 shows that delay probabilities vary across carriers. WN (Southwest Airlines) made up over 25% of all delayed flights from PIT, while other airlines made up considerably smaller proportions.

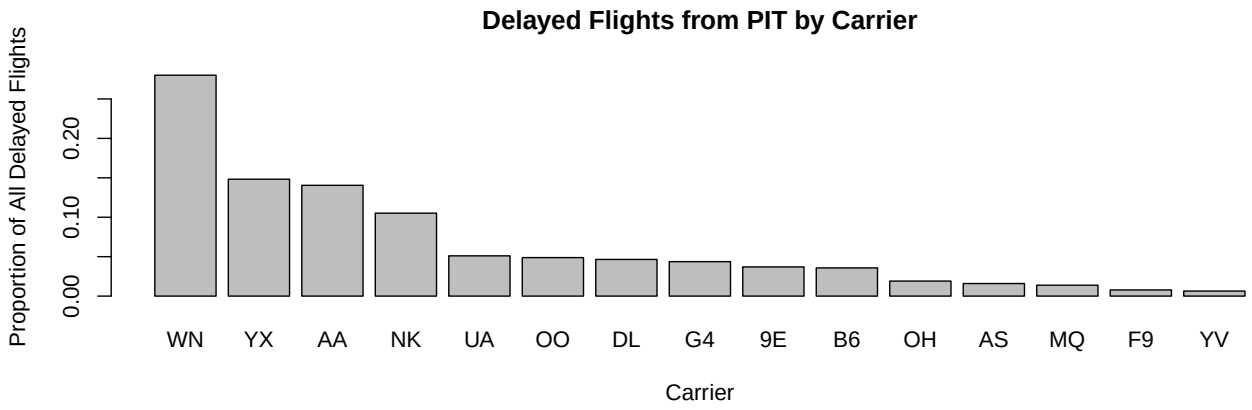

Figure 3: Delayed flights from PIT are uenqually distributed across airlines.

3. Supervised Analysis

For prediction, we used random forests for their ability to reduce variance and improve generalization to unseen data, as well as generate probability estimates than just binary class estimates. Additionally, there are two primary utilities that random forests have which make them ideal models to derive plane delay prediction probabilities. First, they are able to intake high-dimensional data, which meant that we did not have to perform dimensionality reduction (PCA) or select features without having any prior knowledge about covariates' true relationship to the target variable. Second, they provide insight into feature importance, which can be leveraged to pay greater attention to a pointed set of variables to feature engineer.

Feature Engineering

Our feature engineering pipeline evolved through iterative exploration. Each added feature, especially cyclical time encodings, lag delay indicators, and weather covariates, was justified by either 1) observed patterns or intuition, or 2) validated through feature importance scores. This process yielded a refined dataset suitable for effective supervised modeling.

Temporal Features

During EDA, we observed that flight delays exhibited some temporal reliance. In the dataset, the month, the day of the month, and the day of the week are encoded as numbers, but the meaningful relationship of a value of 12 (December) in the `MONTH` column preceding 1 (January) is not preserved in this encoding. To capture these patterns, we added sine and cosine transformations of `MONTH`, `DAY_OF_MONTH`, and `DAY_OF_WEEK` as features into our dataset to reflect their periodic nature.

We also experimented with adding features related to public holidays and days surrounding them, based on the assumption that travel demand and airport traffic could affect delays. In Table 1, out of the 10 dates in 2022 or 2023 with the most delayed flights, the only two dates that are on or near a major holiday are 12/23/2022 and 12/24/2022. We surmised that a flight occurring on a holiday was low signal for a delay and thus ultimately excluded adding a holiday feature to our final model.

Table 1: Dates from 2022/2023 with the Most Delayed Flights Departing from PIT

Date	No. of Delayed Flights	Date	No. of Delayed Flights
1/11/2023	61	1/3/2022	42
8/21/2022	51	12/23/2022	40
7/25/2022	47	7/28/2023	39
12/24/2022	45	6/26/2023	37
7/29/2023	43	7/17/2022	35

Approximating Delays

We had reason to believe that, if a particular plane experienced a delay early on in the day, future legs of flight involving the same plane would also be delayed. This belief is observed in Figure 4, where delayed flights are shown to cluster together and tend to not happen in isolation.

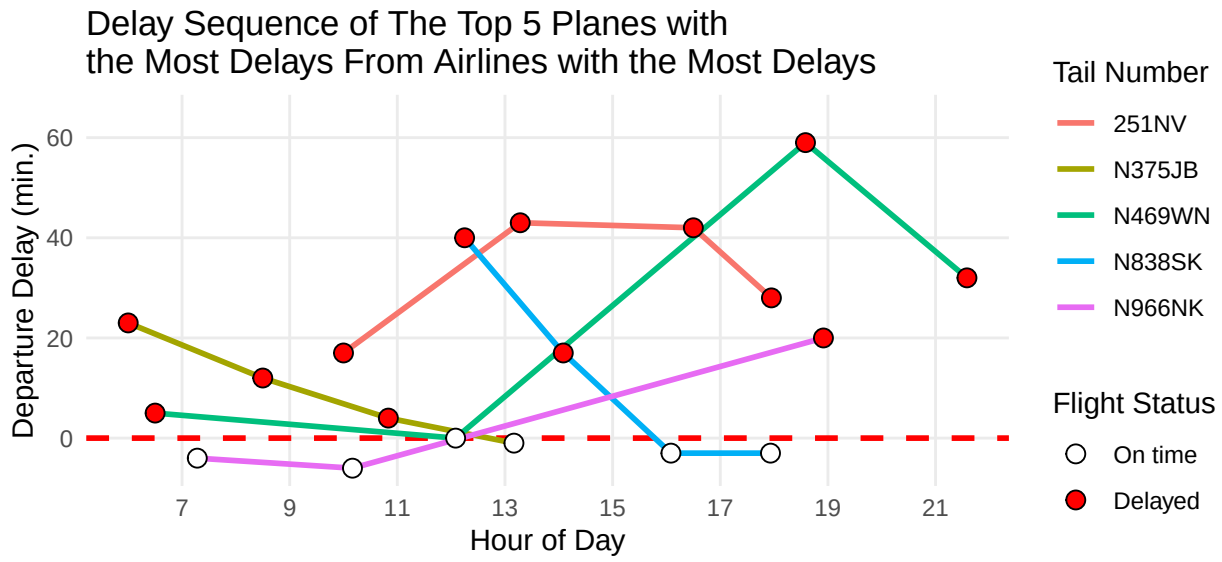

Figure 4: Plane delays tend to cluster together and persist. They don't tend to happen in isolation.

To best approximate delays, we engineered a `PREV_LEG_DELAYED` feature using tail numbers from the same airline carriers to indicate whether the previous leg flown by the same aircraft was delayed (specifically, if `DEP_DEP15`=1). In order to do this, flights were grouped by date and tail number and sorted chronologically using a converted local departure time variable `CRS_DEP_TIME_PIT` that we created. Then, a lag window of 1 was created to set `PREV_LEG_DELAYED` to either 0 or 1 accordingly.

There are two scenarios where `DEP_DEP15` would be unavailable in the test set, which we accounted for. The first scenario is if a particular leg was the first the plane took that day; `DEP_DEP15` is set to 0 here. The second scenario arises with planes that have two legs of flight *after* the random point in time each day when delay information becomes unavailable (in the training set, the maximum number of post-random point flight legs a plane takes is two). The second flight inherits the `DEP_DEP15` value from the first flight, but this itself is an inference. We were not extremely concerned with the second scenario because it is relatively rare in the dataset after further analysis.

From this point on, variables that directly reflect delay outcomes (e.g., `DEP_TIME`, `DEP_DELAY`, `ARR_DELAY`, and all cancellation-related fields) were removed to avoid target leakage when creating features.

Weather Features

Anecdotally, we have reason to believe that weather conditions could affect whether a flight is delayed or not. Using Pittsburgh daily weather data downloaded from Visual Crossing Weather, we merged six features onto each flight in the training data via the flight date: `winddir`, `humidity`, `sealevelpressure`, `windspeed`, `cloudcover`, and `humidity`. A Random Forest model trained on weather-only features confirmed that this set of features were the most important in distinguishing between delays and non-delays.

Interaction Effects

We wanted to make `CRS_DEP_TIME`, which was in the form *hhmm*, less granular because it reduced the influence of small and meaningless temporal variations. Thus, `CRS_DEP_TIME` was converted into categorical time bins named the following: `early_morning`, `morning_rush`, `midday`, `afternoon`, `evening_rush`, `evening`, and `night`. In part due to noticing the higher variable importance scores of the `MONTH`, `DAY_OF_MONTH`, `DAY_OF_WEEK`, and selected weather features, we created interaction terms between `PREV_LEG_DELAYED` and binned scheduled departure times (`CRS_DEP_TIME`) to account for a hypothesis that different weather conditions affect delays differently.

Hyperparameter Tuning

To compensate for the class imbalance identified earlier, we set the class weights in our random forest model to be 1 to 5, penalizing misclassifications in the `DEP_DEL15` class more heavily. The `randomForest()` function offers many hyperparameters that could be tuned, and we focused on two of them: `mtry` and `ntree`. Table 2 shows that the AUC for different values of `mtry` (the number of variables randomly sampled at each node) from 3 to 15 are similar. Thus, we chose to keep `mtry` = 7, which is the square root of the number of features (the default value of `mtry`).

Table 2: AUC Values for Different Values of mtry

mtry	AUC
3	0.7060
5	0.7107
7	0.7093
9	0.7083
11	0.7115
13	0.7096
15	0.7072

We had concerns about diminishing returns with large values of `ntree`, so we similarly ran the model with lower values of `ntree` (compared to the default value of 500), and found that the AUC on the validation set was the highest at the default value. Thus, we kept `ntree` = 500.

Running the Model

The dataset containing 2024 flight records can be considered as a union between the held-out (test) set of flights with missing delay labels and the (validation) set of flights for which the delay labels are not missing. The training data fed into our random forest model were solely the combined 2022 and 2023 flight data. We also had to take care of unseen levels in the validation set. Notably, San Diego was a flight destination in the validation set, but it was not one in the training set. We replaced such labels with the string `"Unknown"`.

To explain the relationship between the predictors in our model and the predictions themselves, we can use a variable importance plot (Figure 5) to determine what features are most informative in predicting flight delays.

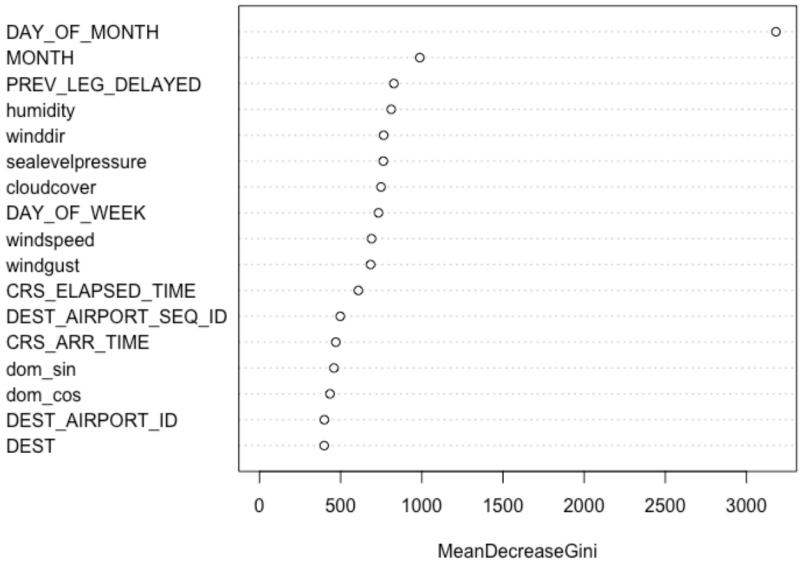

Figure 5: Variable importance plot of our random forest model

The most informative features turn out to be ones that describe the time of year and the weather-related variables. `PREV_LEG_DELAYED` is also a high-scoring feature. As expected, more label-like information is relatively less important.

Surprisingly, the cyclic temporal variables turn out to be uninformative.

4. Analysis of Results

Model Performance Overview

To evaluate the model's effectiveness in predicting flight delays (`DEP_DEL15`), we analyzed both the predicted probability distributions and performance metrics across different classification thresholds.

Predicted Probability Distribution

Figure 6 shows the distribution of predicted probabilities on the 2024 dataset containing "visible" flights (the validation set), colored by the true delay label. The red dotted line indicates the probability threshold at which the accuracy is optimized, which is 0.48.

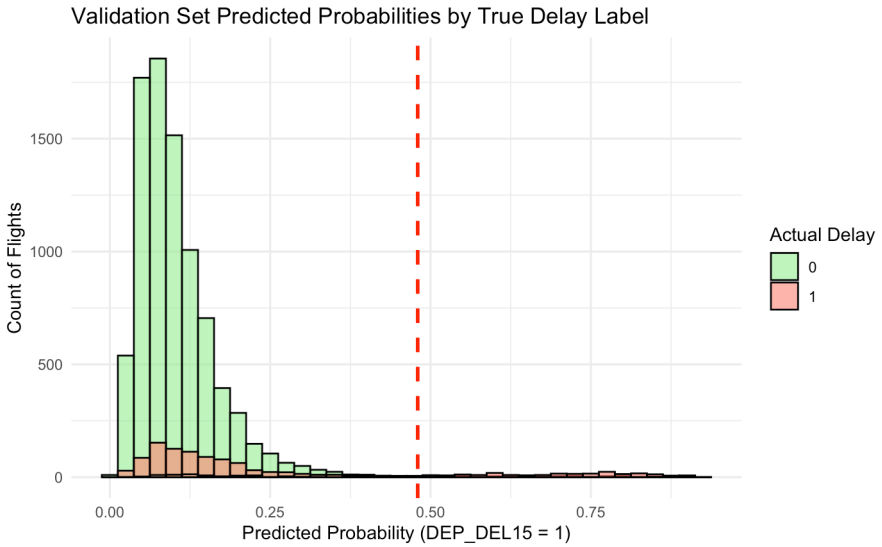

Figure 6: The distribution of predicted probabilities of the validation set highlights the class imbalance.

Several key observations emerge:

- **Class Imbalance:** The majority of flights are not delayed, resulting in a large cluster of predictions near 0. This reflects the imbalance in the data and is expected.
- **False Negatives and Right-Tailed Distributions:** Since there is significant class imbalance, we observe that both the predictions for non-delayed flights (0) and delayed flights (1) exhibit a right-tailed distribution, with most predicted probabilities concentrated near 0. This is expected because the model, trained on a majority of non-delayed flights, tends to predict lower probabilities overall. However, it is encouraging that at the right tail of the distribution (high predicted probabilities), we see an increase in the number of correctly predicted delayed flights (1). This suggests that, despite the imbalance, the model is able to capture important patterns and confidently identify flights likely to be delayed based on the features provided.
- **Confident Predictions:** The model is able to minimize uncertainty in prediction. Few predictions are made near the 0.5 threshold, implying that it tends to commit to either delay or non-delay predictions rather than remaining ambiguous.
- **High-Confidence Positives:** Beyond a probability of 0.7, there is an increasing presence of predicted delays. Despite the imbalance, the model successfully identifies a subset of delayed flights with high confidence.

Threshold-Based Performance Metrics

Figure 7 presents how accuracy, precision, recall, and the F1 score change as the classification threshold is varied:

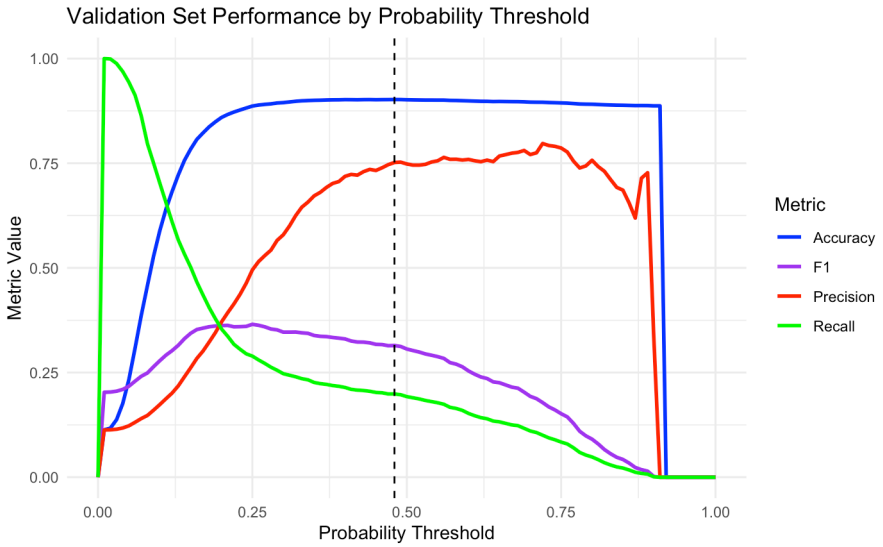

Figure 7: The distribution of predicted probabilities of the validation set reflects the class imbalance.

- **Accuracy:** Remains consistently high due to the dominance of the majority class. However, it is not an informative metric in imbalanced settings. The accuracy at the optimal threshold of 0.48 stated earlier is comparable to those of other thresholds.
- **Precision vs. Recall:** Precision increases with the threshold, while recall drops sharply. This reflects the classic trade-off: higher thresholds lead to fewer false positives but more false negatives.
- **F1 Score:** Peaks around thresholds of 0.35–0.4, suggesting that this range balances the trade-off between precision and recall most effectively.
- **Operational Implication:** If minimizing missed delays is more important than avoiding false alarms, using a lower threshold (e.g., 0.35) is preferable. This would increase recall at the cost of some precision.

Overall, the model performs reasonably well given the imbalance, with confident predictions and clear separation between likely and unlikely delays. However, addressing the false negatives concentrated around low predicted probabilities may further improve delay detection.

Prediction Performance on Different Stratifications

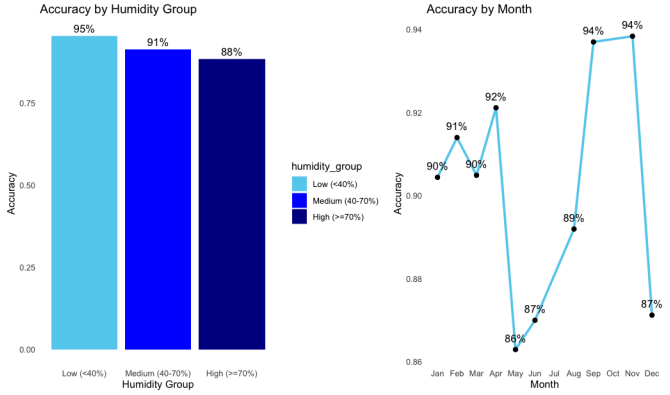

Figure 8: The random forest model has high predictive ability among humidity groups and month.

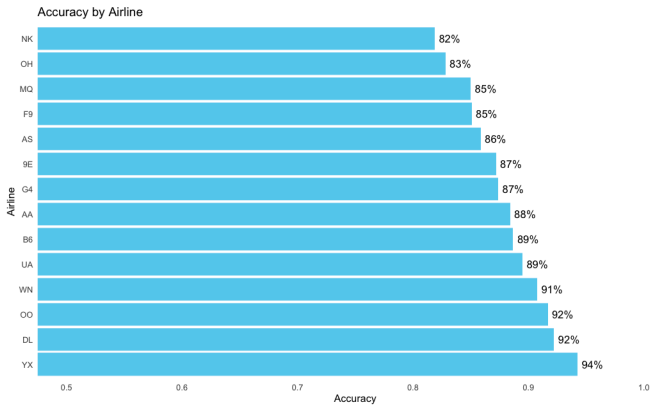

Figure 9: The random forest model has high predictive ability among different airlines.

Based on Figure 8 and 9, across different airlines, prediction accuracy was generally high. All airlines achieved over 80% accuracy, and four airlines had accuracy levels exceeding 90%. This indicates that the model performs well across a variety of carriers. July and October were missing from the 2024 dataset with visible flights, so no evaluation could be made for these months. For the available months, flights in January, February, March, April, September, and November achieved over 90% accuracy. In contrast, May, June, and December showed lower accuracy, likely reflecting greater variability during holiday and peak travel seasons. Humidity emerged as the most important weather-related variable in our random forest model, according to the variable importance plot. Flights operating under low to medium humidity conditions exhibited higher prediction accuracy. In contrast, flights during high humidity (70%) conditions showed slightly lower accuracy, possibly reflecting greater operational variability under such conditions. Overall, flights with more predictable characteristics, such as moderate weather and regular seasonal patterns, were classified more accurately by the model.

Regression Approach for Predicting Delay Minutes

If the goal shifts from predicting a binary delay indicator to predicting the continuous quantity of minutes delayed, then it would be appropriate to reframe the problem as a regression task rather than a classification task. Instead of using a random forest classifier, we would switch to a random forest regressor, which is capable of predicting continuous outcomes. A suitable loss function in this case would be the Mean Squared Error (MSE). MSE penalizes larger errors more heavily than smaller ones, which aligns well with the practical context of flight delays: underestimating a significant delay (e.g., missing a flight by several hours) is far more costly than small errors (e.g., a few minutes late). Therefore, minimizing MSE would encourage the model to prioritize accuracy on larger delays, which is often critical in operational decision-making.

The general steps to adapt our method would be:

- Replace the random forest classifier with a random forest regressor.
- Train the model using minutes delayed as the continuous target variable.
- Evaluate model performance using MSE (and optionally MAE for additional robustness checks).
- Potentially adjust feature engineering to better capture factors influencing delay duration, not just delay occurrence.

Strategy Adaptation for Cost Trade-off

If we cared about a trade-off of a fixed cost C for missing a flight and an hourly cost r of waiting for a delayed flight, instead of the impractical strategy of just showing up on time for all flights, we could model the situation as a classification problem again. We would move the probability threshold, which is determined by the ratio of C to r. When C is greater than r, it becomes more costly to miss a flight than to wait, so we want to be more conservative and predict more flights to be delays (lowering the threshold). When C is less than r, it becomes more costly to wait than to miss a flight, so we are less conservative and predict fewer flights to be delays (raising the threshold).