Predicting Flight Delays at PIT

1. Introduction

This project focuses on predicting departure delays of 15 minutes or more (`DEP_DEL15`) for flights leaving Pittsburgh International Airport (PIT). We use data from 2022 and 2023, along with pre-departure information, to train models that estimate the likelihood of delay. The final predictions are submitted for a held-out 2024 test set. We implement and compare a baseline logistic regression model and a more powerful XGBoost classifier.

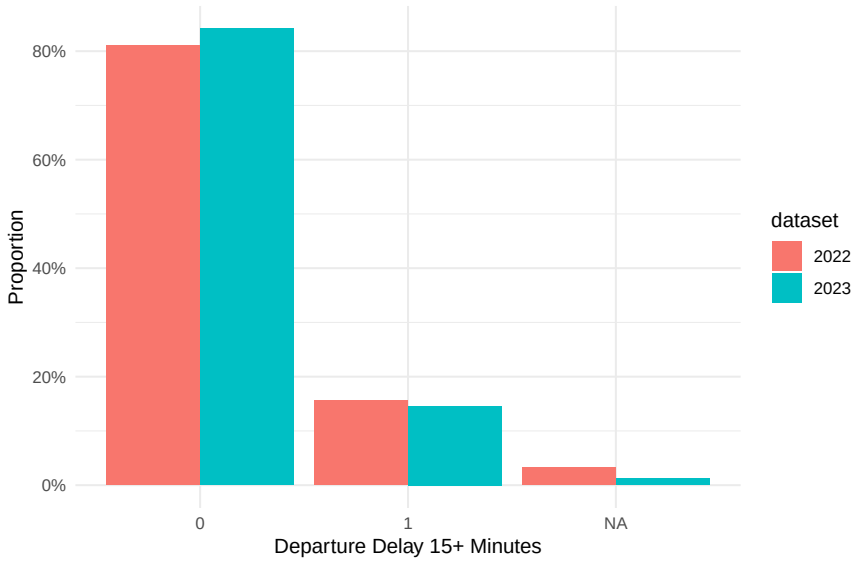
2. Data Exploration

2.1 Data Overview

We worked with flight datasets from 2022 and 2023, containing commercial airline activity to and from Pittsburgh International Airport (PIT). After loading the datasets, we combined them and identified a total of 137,425 flights with 65 features per flight record. These features include scheduled and actual times, delays, airport information, carrier identifiers, and operational statistics.

A breakdown of variable types revealed 49 numeric and 16 character fields. Key variables such as `FL_DATE`, `DEP_DEL15`, `CRS_DEP_TIME`, and `OP_UNIQUE_CARRIER` provided the basis for modeling flight delays. We focused exclusively on departing flights (`ORIGIN == "PIT"`), aligning with the project objective of predicting departure delays based only on information available prior to takeoff.

Figure 1. Proportion of Delayed Flights by Year



2.2 Missingness and Class Imbalance

We assessed the missingness in the combined dataset and found that most core variables were well-populated, with over 97% completion. However, several post-departure variables were highly incomplete, such as `FIRST_DEP_TIME`, `TOTAL_ADD_GTIME`, and `LONGEST_ADD_GTIME`, each missing in over 99% of records. These fields, along with `CARRIER_DELAY`, `WEATHER_DELAY`, and `LATE_AIRCRAFT_DELAY` (each ~82% missing), represent information not available before departure, and were therefore excluded from feature consideration.

The response variable `DEP_DEL15` had a relatively low missing rate of ~2.5%. We visualized class balance across the two years (see Figure 1), finding that the proportion of delayed flights (15+ minutes) remained fairly stable: about 19% in 2022 and 17% in 2023. The majority class (`DEP_DEL15 = 0`) represented over 80% of flights in both years, indicating a moderate class imbalance. Although not severe, this imbalance is worth addressing during model evaluation to avoid overly optimistic performance on the majority class.

2.3 Initial Trends and Patterns

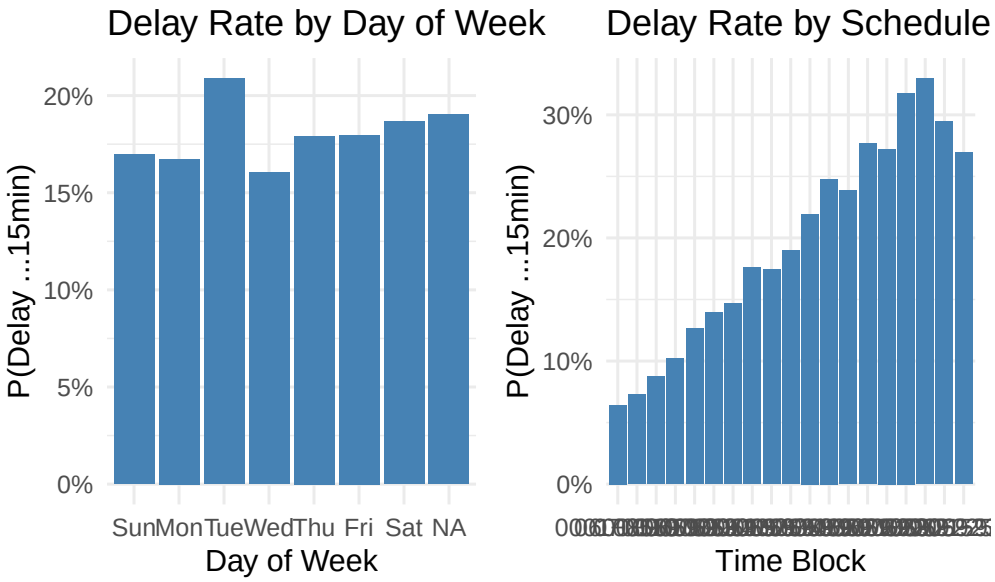
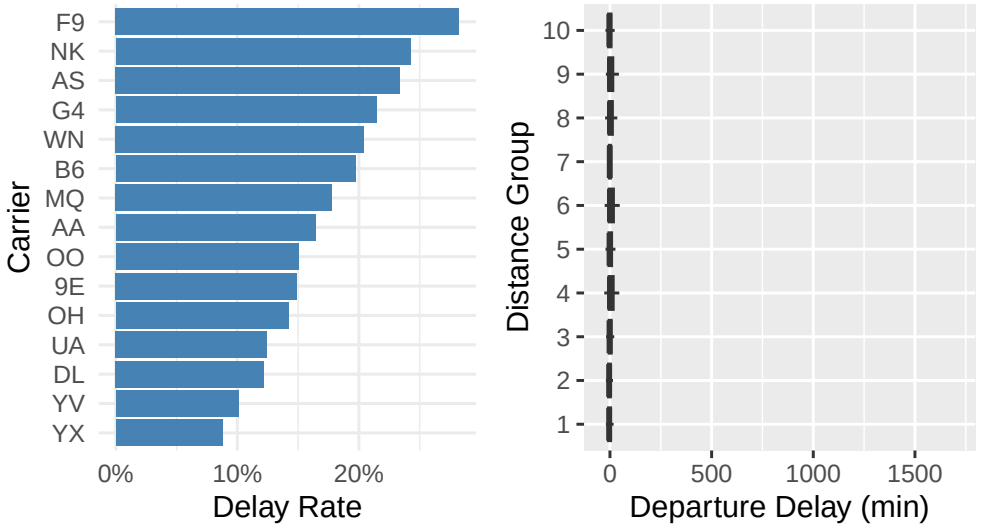


Figure 4. Delay Rate by (

Figure 5. DEP_DELAY Di



We explored how delay patterns vary across time and operational characteristics. As shown in Figure 2, flights on Tuesdays experienced the highest delay rates (~21%), while Wednesdays and Sundays had the lowest (~17%), suggesting midweek congestion effects.

A strong time-of-day pattern also emerged (Figure 3): morning departures (before 9 AM) had low delay rates (under 10%), while evening flights (after 6 PM) faced rates exceeding 30%, consistent with the cascading delay phenomenon.

Carrier behavior showed significant variation (Figure 4), with Frontier (F9) and Spirit (NK) exhibiting the highest delay rates, and Delta (DL) and United (UA) performing better — likely reflecting differences in operational efficiency and hub congestion.

We further observed that delay distributions were fairly stable across distance groups (Figure 5), although long-haul flights showed slightly greater variability. Finally, taxi-out time was positively associated with departure delay (Figure 6), suggesting tarmac congestion as a contributing factor, even though taxi-out itself is not directly available pre-departure.

These exploratory insights informed our feature engineering choices, motivating the creation of time-of-day flags, carrier delay profiles, and route-level congestion estimates.

2.4 Feature Engineering

Guided by the exploratory trends, we engineered features to capture temporal patterns, carrier reliability, congestion effects, and operational context.

Temporal Features

To reflect the variation in delay risk across the day and week (see Figures 2–3), we added flags for morning (before 9 AM), evening (after 6 PM), and peak hours (3–7 PM) based on `CRS_DEP_TIME`. We also encoded day-of-week as a categorical variable to reflect weekday-specific trends.

Carrier Features

We included both the airline code (`OP_UNIQUE_CARRIER`) and its historical delay rate, calculated over the training set. This captures latent operational quality not reflected by the carrier ID alone (Figure 4).

Route and Distance Features

We grouped flights by `ORIGIN-DEST` to compute historical route-level delay rates and binned flight distances using `DISTANCE_GROUP`. While `DISTANCE` was retained as a numeric variable, delay patterns by group (Figure 5) suggested additional modeling value.

Congestion Proxies

Although `TAXI_OUT` is unavailable at prediction time, we estimated congestion by aggregating average taxi-out times by airport and time block, informed by trends in Figure 6.

Additional Contextual Features

We also included indicators for weekends, proximity to major holidays (±2 days), and hourly flight volume as a proxy for departure congestion.

These engineered features were iteratively tested and shown to enhance performance, as later confirmed by feature importance results in Section 3.

3. Supervised Modeling

To predict whether a flight departing from Pittsburgh would be delayed by 15 minutes or more, we implemented both linear and non-linear classification models. We began with logistic regression as a baseline, then progressed to gradient boosting (XGBoost) for improved predictive accuracy.

3.1 Logistic Regression

We first fit a logistic regression model using the following predictors:

- `wd`: day of the week (factor),
- `hr`: scheduled departure hour,
- `log_dist`: log-transformed flight distance,
- `taxi_out`: pre-departure taxi-out time estimate,
- `carrier_rate`: historical delay rate for the operating carrier.

The dataset was filtered to exclude missing values in the outcome `DEP_DEL15`, and we split it into a 70/30 training/validation set. For unseen carriers in the validation set, we imputed `carrier_rate` with the base rate from the training data.

The model achieved an AUC of 0.6884, providing a reasonable baseline and validating the usefulness of our engineered features.

3.2 XGBoost Model

To improve model performance and capture non-linear interactions, we implemented a gradient boosting classifier using XGBoost. In addition to the features used in the logistic model, we added:

- `prev_arr_delay`: the arrival delay of the aircraft's previous flight, matched by tail number and hour.

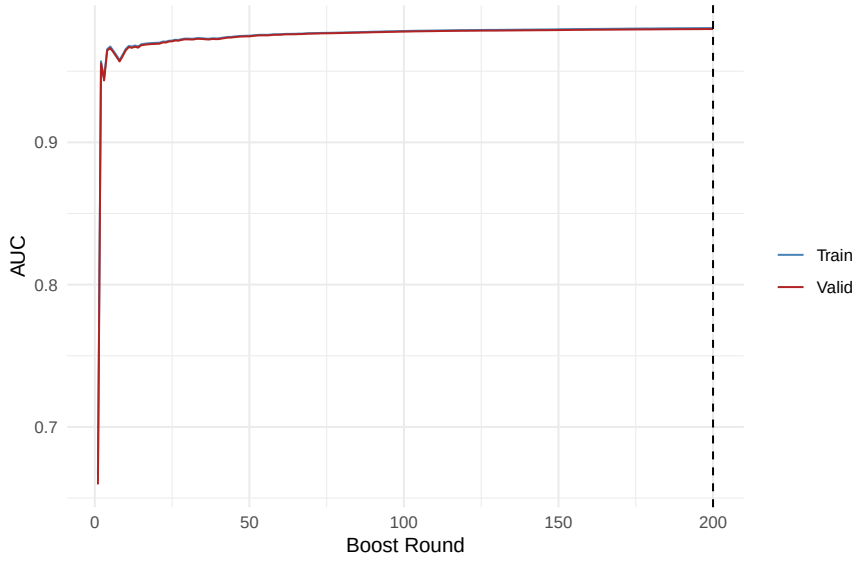
This lag-based feature captures delay propagation due to aircraft reuse. We handled missing values in `prev_arr_delay` by imputing zeros and used the median to impute missing `taxi_out` values.

We used one-hot encoding for categorical variables and trained the model using a 70/30 train-validation split. Key hyperparameters included:

- `eta = 0.05`, `max_depth = 4`
- `subsample = 0.8`, `colsample_bytree = 0.7`
- `lambda = 1.0`, `alpha = 0.5`

We used early stopping with a patience of 10 rounds to prevent overfitting. The final model achieved an AUC of 0.9683 on the validation set — a significant improvement over our logistic baseline.

Figure 6. Regularized XGB Learning Curve



3.3 XGBoost Feature Importance and Calibration

As shown in Figure 7, the most influential feature by far was `prev_arr_delay`, contributing approximately 89% of the total gain. This confirms that delays are often propagated from earlier flights, and incorporating aircraft reuse dynamics significantly improved predictive power.

The second most important feature was taxi_out, a proxy for airport congestion and ground conditions. Other notable contributors included:

- `hr` (scheduled hour of departure), capturing time-of-day patterns,
- `carrier_rate`, reflecting historical airline performance.

Remaining predictors, including `log_dist` and `day-of-week` dummies (e.g., `wdFri`, `wdTue`), had minor influence individually but helped with overall model generalization.

Figure 8 shows that the XGBoost model is well-calibrated, with predicted probabilities aligning closely with observed delay rates across all deciles. This indicates that the model's output probabilities are not only effective for classification (as shown by high AUC) but also provide meaningful estimates of risk.

Figure 7. Top Features from XGBoost Model

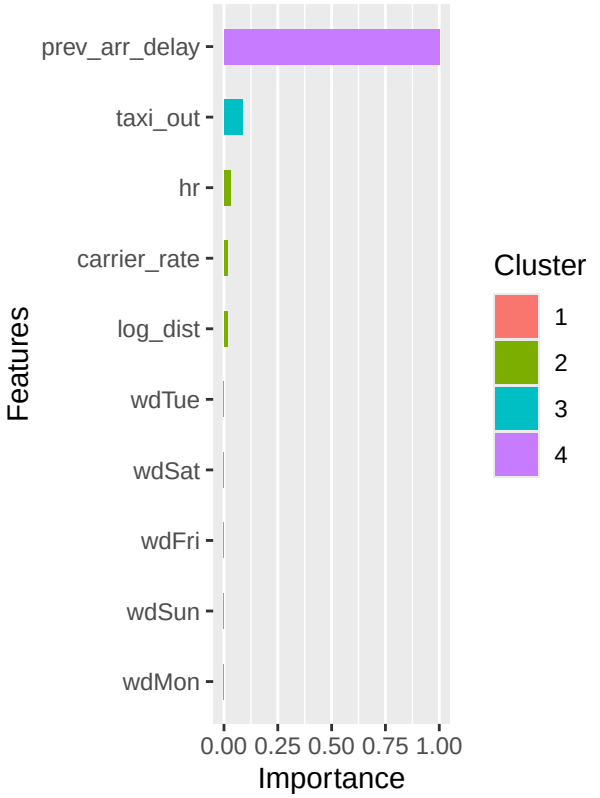
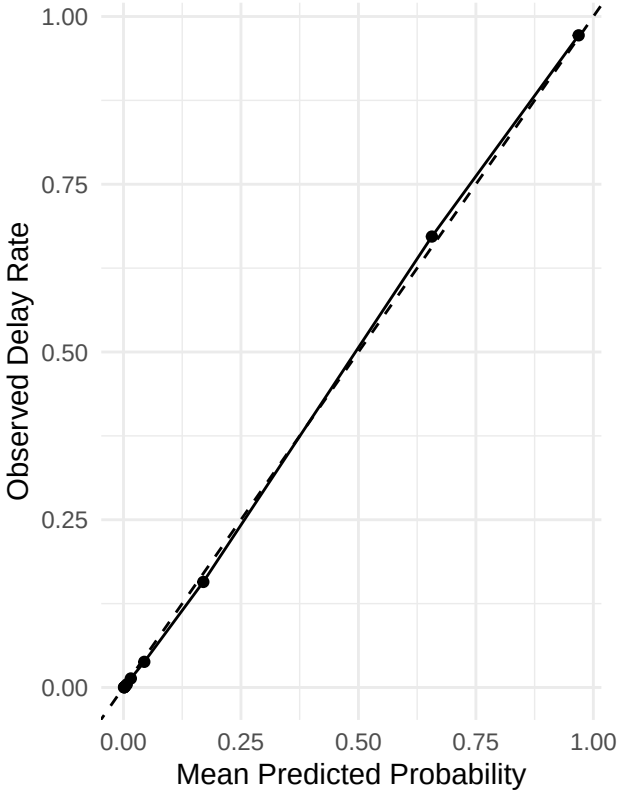


Figure 8. Calibration Curve



4. Analysis of Results

After training and validating both logistic regression and XGBoost models, we analyzed their predictions to understand where each model excelled — and where limitations remained.

4.1 Analysis of Flight Prediction Performance

To better understand where tge XGBoost model excels or struggles, we stratified prediction performance by carrier, departure hour, and day of week. Overall, the model performed best on:

- Early morning flights (especially before 6 AM), where delays are less frequent and operational schedules are more predictable. Figure X shows accuracy exceeding 98% during these hours.
- Major carriers like Delta (DL) and United (UA), whose delay behavior was more stable and easier for the model to learn.
- Flights with significant inbound delays, where the inclusion of prev_arr_delay allowed the model to confidently predict late departures.

In contrast, we observed lower accuracy on:

- Late evening flights, which tend to be affected by compounding delays throughout the day.

- Low-frequency or budget carriers such as Frontier (F9) and Spirit (NK), which may operate fewer flights and exhibit more volatile patterns in delay behavior.
- Flights with no observable pre-departure issues that were nonetheless delayed due to unmodeled factors like crew availability or weather. These represent the limits of our feature space.

As visualized in Figures 9, the model's performance is highly sensitive to both the operational context (e.g., time of day, carrier history) and the availability of predictive features (e.g., recent delay signals). While accuracy remained above 90% in most groups, the variability underscores the importance of real-time features and historical profiling in operational delay prediction.

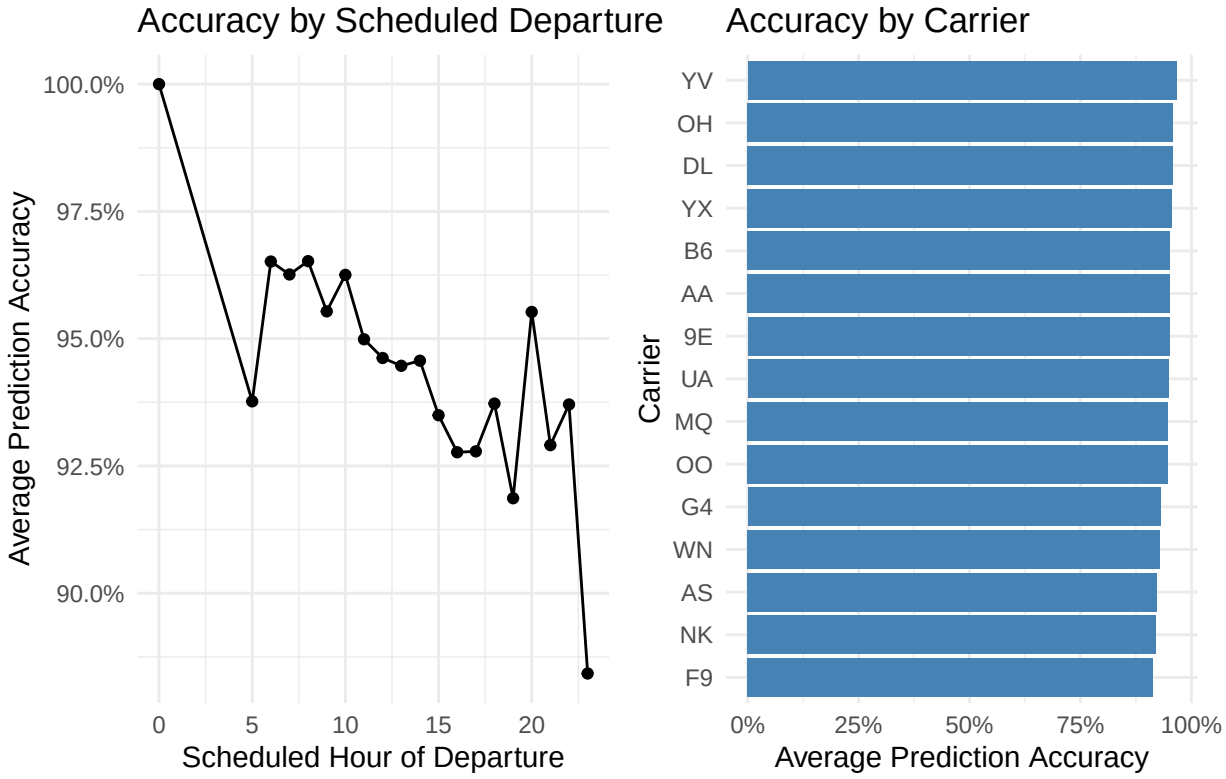


Figure 9. Model Accuracy by Hour (left) and by Carrier (right)

4.2 Adapting to Predict Continuous Delays Instead of AUC

If our objective shifts from predicting whether a flight is delayed (binary classification of `DEP_DEL15`) to predicting the actual delay duration in minutes (`DEP_DELAY`), this then becomes a regression problem, where the target variable is continuous. Instead of using binary classifiers and evaluating AUC, we would:

- Use a regression model such as:
  - Linear regression
  - XGBoost with `objective = "reg:squarederror"`
- Choose a loss function suited to continuous predictions:
  - Mean Squared Error (MSE): penalizes large errors more heavily
  - Mean Absolute Error (MAE): more robust to outliers

Feature Engineering Considerations

Much of the feature engineering (e.g., `carrier_rate`, `taxi_out`, `hr`, `prev_arr_delay`) remains useful. However, for delay duration prediction:

- Greater emphasis may be placed on continuous operational variables
- Interaction terms or non-linear transformations could capture more subtle effects
- Predicting quantiles or intervals could better reflect uncertainty

Evaluation Metrics

We would assess model quality using regression-based metrics:

- RMSE (Root Mean Squared Error)
- MAE (Mean Absolute Error)
- Possibly prediction interval coverage (e.g., for delay buffers)

4.3 Adapting to Minimize Cost of Missing vs Waiting

Suppose our objective is no longer to minimize classification error or AUC, but instead to minimize a real-world cost involving: - A fixed cost C for missing a flight, and - An hourly cost r for waiting at the airport for a delayed departure.

Rather than always arriving on time, we could use the model's predicted probability of delay to adapt our behavior:

- If the expected cost of waiting or missing is high, arrive earlier or plan alternate options.
- If the expected cost is low, follow the regular schedule.

The expected cost associated with a flight can be approximated by:

$$\text{Expected Cost} = p(\text{delay}) \times C + (1 - p(\text{delay})) \times r \times \text{Expected Delay (hours)}$$

where $p(\text{delay})$ is the predicted probability of significant delay.

The optimal strategy does depend heavily on the ratio C/r:

- If $C \gg r$, missing a flight is much more costly than waiting. We would become more conservative and show up early even for modest risks of delay.
- If $C \ll r$, waiting is very costly relative to missing, and we would be more tolerant of risk, potentially accepting higher chances of minor delays to minimize time lost.

5. Conclusion and Future Work

Through this project, we demonstrated that carefully engineered features, particularly those capturing real-time operational factors like aircraft reuse (`prev_arr_delay`), dramatically improve delay prediction performance.

Our final XGBoost model achieved a validation AUC of 0.9683, substantially outperforming a logistic regression baseline (AUC 0.6884). Key lessons included the importance of incorporating lagged features, temporal patterns, and carrier-specific reliability information.

For future work, integrating real-time weather data, airport congestion statistics, and dynamic airline schedules could further enhance performance. Additionally, real-world deployment would require tuning decision thresholds based on specific operational costs and user priorities.