Flight Delay & Event Impact Analysis
Fundamentals of Data Science 2

Professor:
Henrik Leopold

Group Members:
John Schlotfeldt, Felipe Arroyave, William Merino-Galindo

---

1. Introduction & Problem Conceptualization

Commercial flight delays impose substantial operational and economic costs on airlines, airports, and passengers. Although many delays arise from real-time factors such as weather or aircraft rotations, airlines must often make decisions before departure, when only static, pre-flight information is available.

This project analyzes U.S. commercial flight delays in 2024 using the full Bureau of Transportation Statistics (BTS) on-time performance dataset (≈7 million flights after cleaning), augmented with major sports event schedules mapped to nearby airports. The analysis follows the full data science lifecycle and is implemented in Python using pandas, scikit-learn, and LightGBM, with emphasis on scalability, leakage prevention, and decision-oriented evaluation via an Expected Value (EV) framework.

A material delay is operationally defined using the FAA standard of an arrival delay of at least 15 minutes. This definition aligns the modeling task with industry practice and enables predictions to be interpreted in terms of concrete operational actions such as proactive rebooking, buffer allocation, or customer notification.

The core business questions are:

- Which airports, routes, and times experience the highest delay risk?
- How large scheduled (NFL, NBA, MLB, NHL, MLS) events affect delays affect delay risk?
- Can we predict, before departure, whether a flight will arrive at least 15 minutes late?
- How predictive performance can be translated into operational value?

We frame the problem using pre-flight information only, combining classification, regression, and clustering to identify structural delay patterns and the limits of prediction without real-time data.

2. Business & Data Understanding

2.1 Exploratory Data Analysis

Initial exploration focused on identifying structural delay patterns across time, space, and external factors.

Delay rates are lowest in the early morning and rise steadily throughout the day, peaking at around 2000. This pattern is consistent with network-wide congestion propagation rather than isolated disruptions.

Figure 1. Arrival delay rate by scheduled departure hour

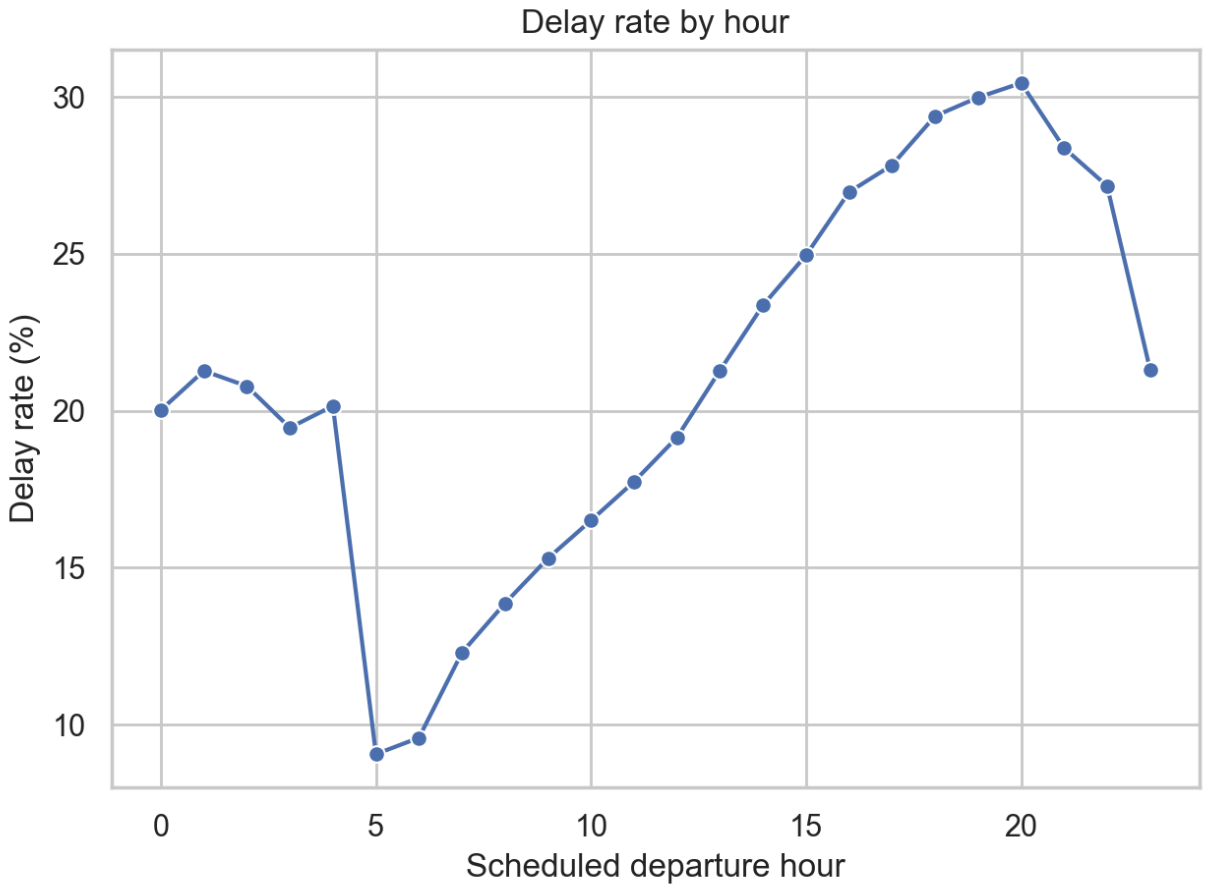


Figure 2 shows that while total flight volume remained relatively stable between July (613,938 flights) and October (608,514 flights), a difference of only 5,424 flights, the number of delayed flights declined sharply over the same period. Delayed flights decreased from 181,865 in July to 80,343 in October, representing a reduction of 101,522 delays. This divergence indicates that seasonal and operational factors, rather than flight volume alone, play a dominant role in monthly delay outcomes.

Figure 2. Delay rates by month

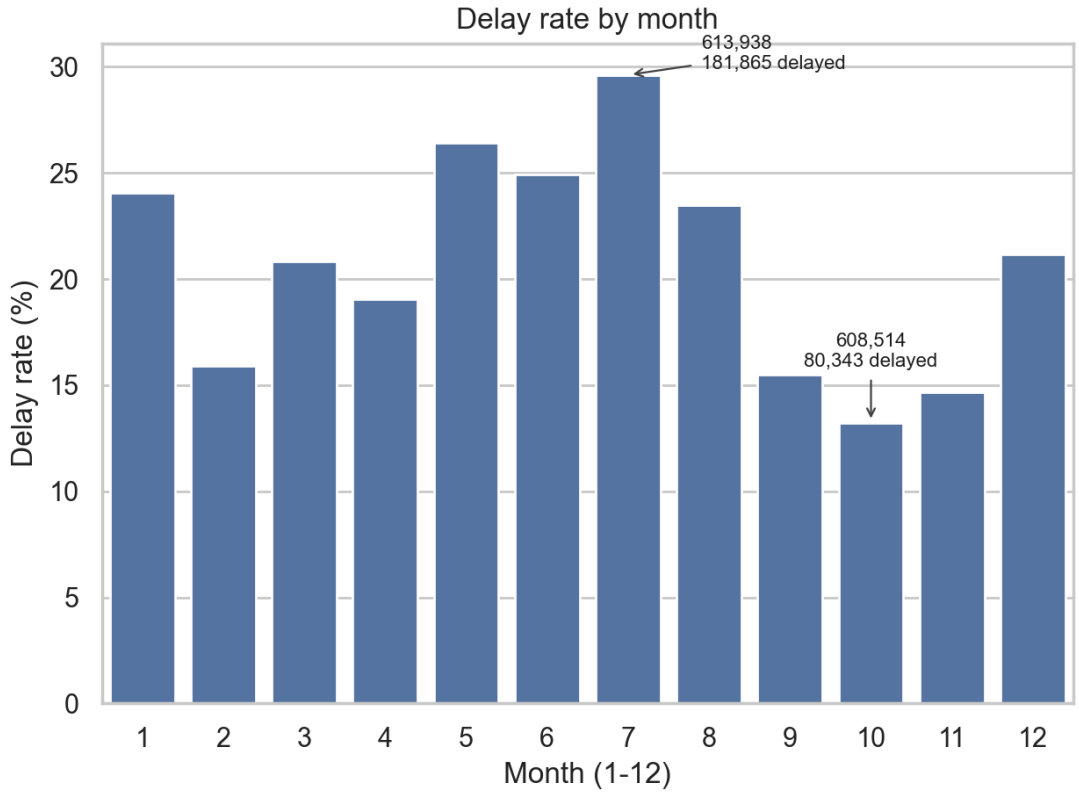
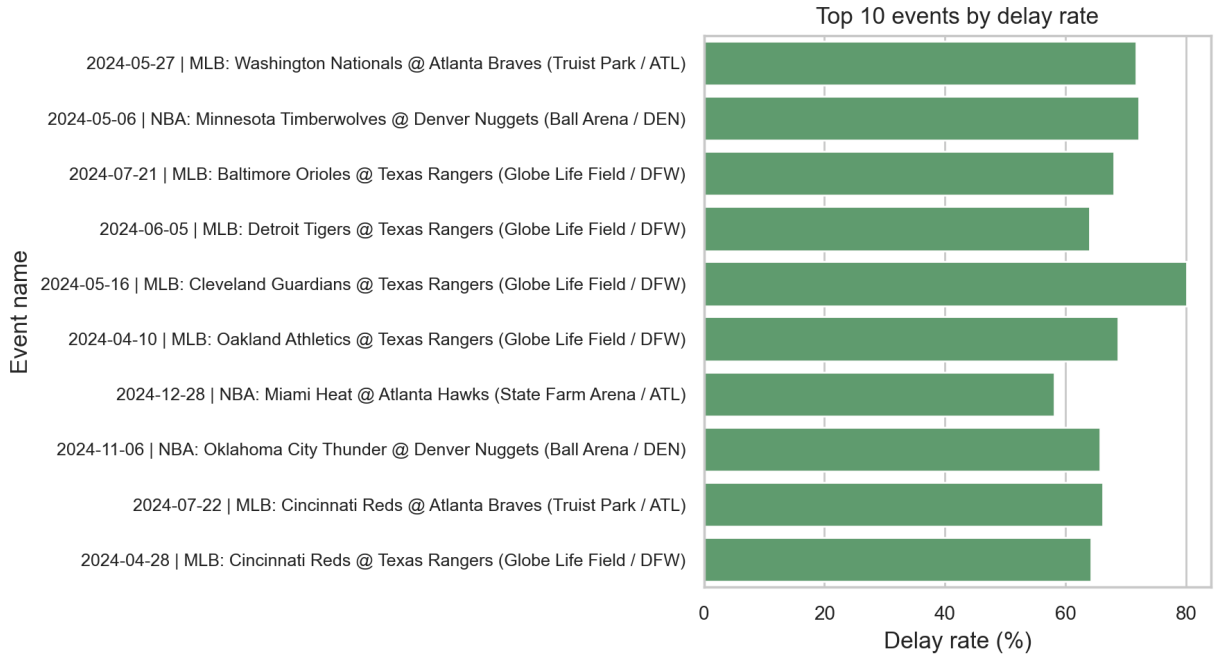


Figure 3 shows the ten individual sports events in 2024 associated with the highest arrival delay rates (over 15 minutes) at the nearest major airport on the event date. Each bar represents the percentage of flights delayed at the corresponding airport, highlighting that certain large, scheduled events act as extreme congestion shocks. Delay rates exceeding 60–80% illustrate that event-driven disruption is substantial, localized, and predictable using pre-flight information alone.

Figure 3. Top 10 sports event days by flight delay rate



3. Data Preparation

3.1 Handling big data

Given the dataset's scale, data loading is performed using chunked streaming (250,000 rows per chunk), loading only required columns and applying memory-optimized dtypes (int8/int16, float32, category). Dates are parsed per chunk to support time-based features. This approach enables full-dataset processing on modest hardware while maintaining fidelity.

3.2 Cleaning steps

Within each chunk, we apply the following cleaning logic before concatenation:

- Drop unusable flights (not meaningful for arrival delay prediction)
    - Remove cancelled flights (cancelled = 1).
    - Remove diverted flights (diverted = 1).
- Target availability
    - Drop rows with missing arr_delay, since we cannot train or evaluate without the target.
- Type coercion and validation
    - Convert numeric columns via pd.to_numeric(errors="coerce") where necessary.
    - Drop rows with invalid or missing scheduled times (crs_dep_time, crs_arr_time) to ensure meaningful time-derived features.

This cleaning isolates a consistent training population: completed flights with valid schedule information and observed arrival delays.

3.3 Feature engineering

Time-based features

- dep_hour: integer hour of scheduled departure (crs_dep_time // 100).
- day_of_year: day index from 1 to 365/366 derived from fl_date.
- is_weekend: indicator derived from day_of_week (BTS: 1=Mon … 6=Sat, 7=Sun).

Departure hour is treated as a cyclic variable and encoded using sine and cosine transformations to preserve temporal continuity (23:00 and 00:00).

Categorical handling

- All categorical fields (op_unique_carrier, origin, origin_state_nm, dest, dest_state_nm) are cast to string and then to category after imputation.
- Missing categories are filled with an explicit "Unknown" bucket rather than dropping rows, preserving as much training data as possible.

Event features

Using Event_Utils and the event schedules file:

- Each flight date and airport is checked against event schedules to produce:
    - An event_day flag (event vs. non-event).
    - Event attributes such as event type (NFL, NBA, MLB, NHL & MLS) and a coarse event category if relevant.
- These variables capture predicted congestion from major sporting and stadium events, integrating domain context into the predictive feature set.

Leakage prevention

With an emphasis on leakage for our project, we explicitly exclude any columns that encode information only known after departure or directly tied to the outcome, such as:

- Actual departure/arrival times: dep_time, arr_time, actual_elapsed_time, wheels_off, wheels_on.
- Delay cause fields (carrier_delay, weather_delay, nas_delay, late_aircraft_delay, security_delay) and any external weather APIs that reflect realized conditions at the time.

These variables are used only for descriptive analysis, not as predictors. Keeping the feature matrix strictly “pre-flight” ensures that performance metrics reflect realistically usable information.

3.4 Final feature matrix

After chunk-wise processing, all cleaned chunks are concatenated. Categorical columns are converted to category types once again at the global level to avoid inconsistencies. The final feature matrix comprises:

- Numeric features: all raw numeric variables plus engineered features (year, month, day_of_month, day_of_week, crs_dep_time, crs_arr_time, crs_elapsed_time, distance, dep_hour, day_of_year, is_weekend, dep_hour_sin, dep_hour_cos).
- Categorical features: op_unique_carrier, origin, origin_state_nm, dest, dest_state_nm, and selected event flags.

This matrix is used for both classification and regression, with LightGBM consuming numeric and categorical features natively, and linear/tree models using only numeric subsets as needed.

4. Modeling

We build three families of models: classification, regression, and clustering. Modeling decisions follow the baseline to stronger tree ensembles to optimized gradient boosting.

4.1 Classification: predicting delay ≥ 15 minutes

Target and class imbalance

We define Y_class = 1{arr_delay ≥ 15}, and observe:

- Positive rate: 20.8% of flights are delayed ≥ 15 minutes.
- Naive accuracy: Always predicting “on time” yields 79.2% accuracy.

This confirms a strong class imbalance and shows why accuracy alone is misleading: a “do nothing” model looks good numerically but never identifies any delayed flights.

Models

We compare three classifiers with increasing capacity: Logistic Regression as an interpretable baseline, Random Forest to capture non-linear interactions, and LightGBM as the optimized model due to strong performance on large tabular data and native categorical handling. We use a stratified train-test split (80/20) to preserve class imbalance across sets.

Hyperparameter tuning

LightGBM hyperparameters are tuned on a 200k subsample using cross-validation over tree depth, leaf count, and sampling parameters, optimizing F1/AUC. The selected configuration is then trained on the full training set.

4.2 Regression: predicting delay minutes

For the continuous target arr_delay, we compare:

- Baseline: predict the mean delay for all flights.
- Linear Regression: uses standardized numeric features.
- Gradient Boosting Regressor: tree-based ensemble on numeric features.
- LightGBM Regressor: gradient boosting with full categorical and numeric feature set.

The focus is on RMSE (to align with squared loss optimization) and MAE (as a more interpretable “typical error in minutes” for communication with non-technical stakeholders).

4.3 Clustering: unsupervised structure in flight operations

We apply MiniBatchKMeans on a subset of numeric features:

- distance, crs_elapsed_time, crs_dep_time, dep_hour.

After standardization, we fit a 5-cluster solution. The aim is not prediction but structural understanding: identifying typical flight “regimes” (short-haul morning, long-haul evening, etc.) and then overlaying delay rates on these clusters to see which regimes are structurally riskier.

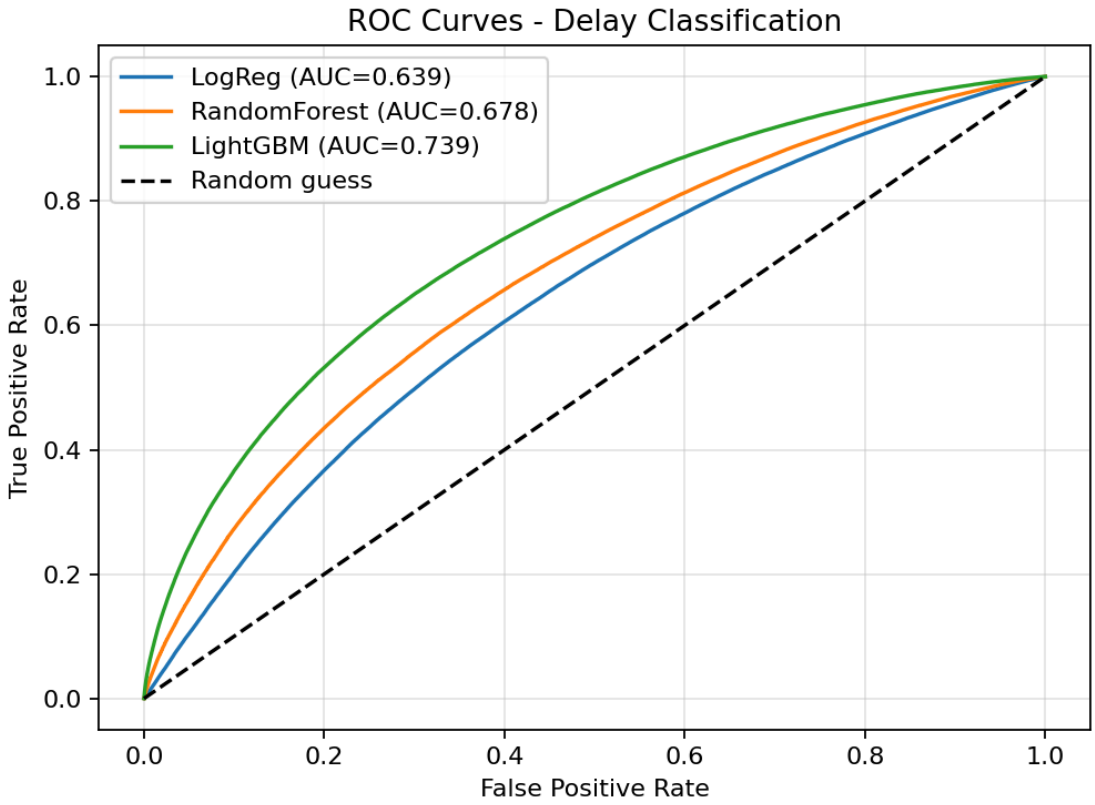
5. Evaluation and Advanced Diagnostics

5.1 Classification results and metric suitability

Compared to the naive “always on time” baseline (79.2% accuracy, 0% recall for delays), all models sacrifice accuracy to gain recall on the minority class. LightGBM beats the other models in F1, indicating a better balance between catching delayed flights and controlling false positives.

- Given class imbalance, evaluation focuses on recall, precision, F1, and ROC-AUC rather than accuracy.

ROC curves and AUC scores are computed for all three classifiers. The AUC for LightGBM is highest, and its ROC curve lies above those of logistic regression and random forest across almost the entire false-positive range, confirming that LightGBM has superior global discrimination independent of any specific threshold.



5.2 Threshold sweep and Expected Value framework

Using the LightGBM predicted probabilities, we evaluate thresholds at 0.3, 0.5, and 0.7:

- Threshold 0.3
  - Very high recall (≈ 0.94): almost all delayed flights are flagged.
  - Low precision (≈ 0.24) and poor accuracy (≈ 0.38): many false alarms.
- Threshold 0.5
  - Statistically balanced point.
  - Precision ≈ 0.35, recall ≈ 0.65, F1 ≈ 0.46.
  - Important note: This threshold results in a negative expected value and is financially infeasible.
- Threshold 0.7
  - Conservative setting.
  - High precision (≈ 0.53) and accuracy (≈ 0.80, slightly above the naive baseline) but recall drops to ≈ 0.25.

Using the cost/benefit matrix (TP = +100, FP = −10, FN = −200, TN = 0), only the 0.3 threshold yields a positive expected value (+10.92 per flight).

Accordingly, the optimal operating point is a low probability threshold (≈0.3). While this setting increases false alarms, it maximizes business value by aggressively capturing delayed flights and avoiding the substantially higher cost of missed disruptions. This demonstrates that economically optimal decisions may differ from statistically "balanced" thresholds.

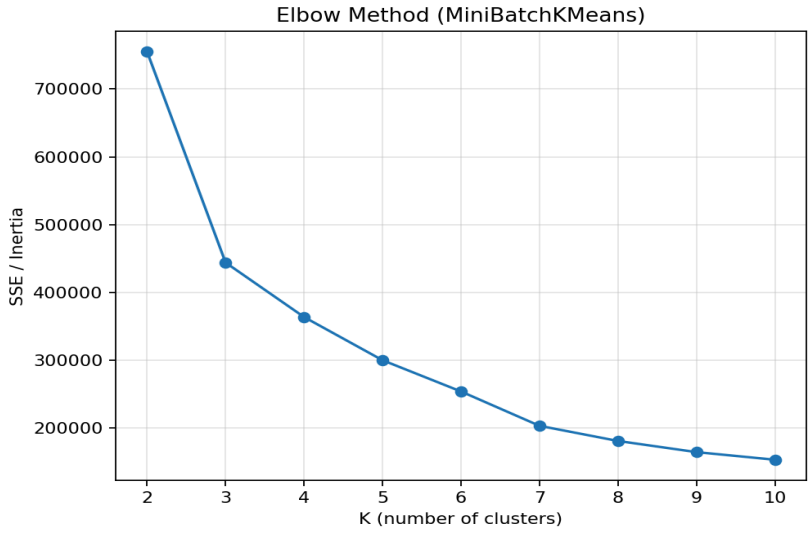
5.3 Regression performance and business interpretation

For regression, we compare models using RMSE and MAE:

- Baseline (mean delay): RMSE ≈ 58.1 minutes.
- Linear Regression: RMSE ≈ 57.8 (minimal improvement).
- Gradient Boosting: RMSE ≈ 56.8.
- LightGBM Regressor: RMSE ≈ 55.7 (best).

MAE is also computed for each model (values in the range of "typical error" on the order of tens of minutes). While LightGBM reduces RMSE by roughly 2.5 minutes over the baseline (about a 4% improvement), the absolute errors remain large relative to operational decision thresholds. This justifies using the classification framing with EV and thresholds as the primary decision tool, and treating regression outputs as supplementary.

5.4 Clustering results

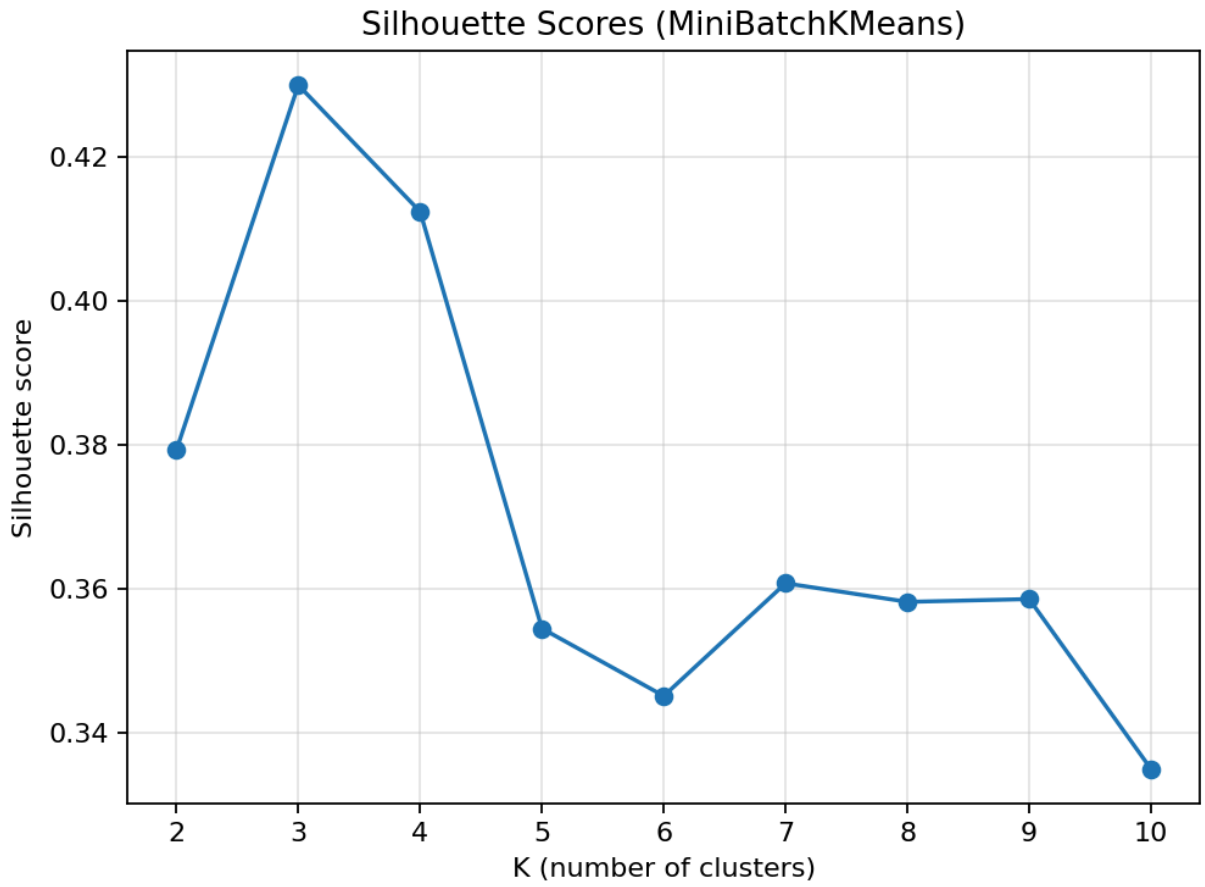


The MiniBatchKMeans clustering on standardized distance, crs_elapsed_time, crs_dep_time, and dep_hour yields five clusters with millions of flights each. When we compute delay rates and mean delays by cluster, we observe:

- Evening short- and medium-haul clusters have the highest delay rates (e.g., ≈ 26–29% of flights delayed ≥15 minutes, with average delays around 13–14 minutes).
- Morning clusters (both short- and medium-haul) have lower delay rates, around 15% with average delays below 2 minutes.
- Long-haul, midday flights sit in the middle.

This supports the EDA findings: time-of-day and route length significantly shape structural delay risk. Clustering is not evaluated by predictive metrics but by how well it explains these structural

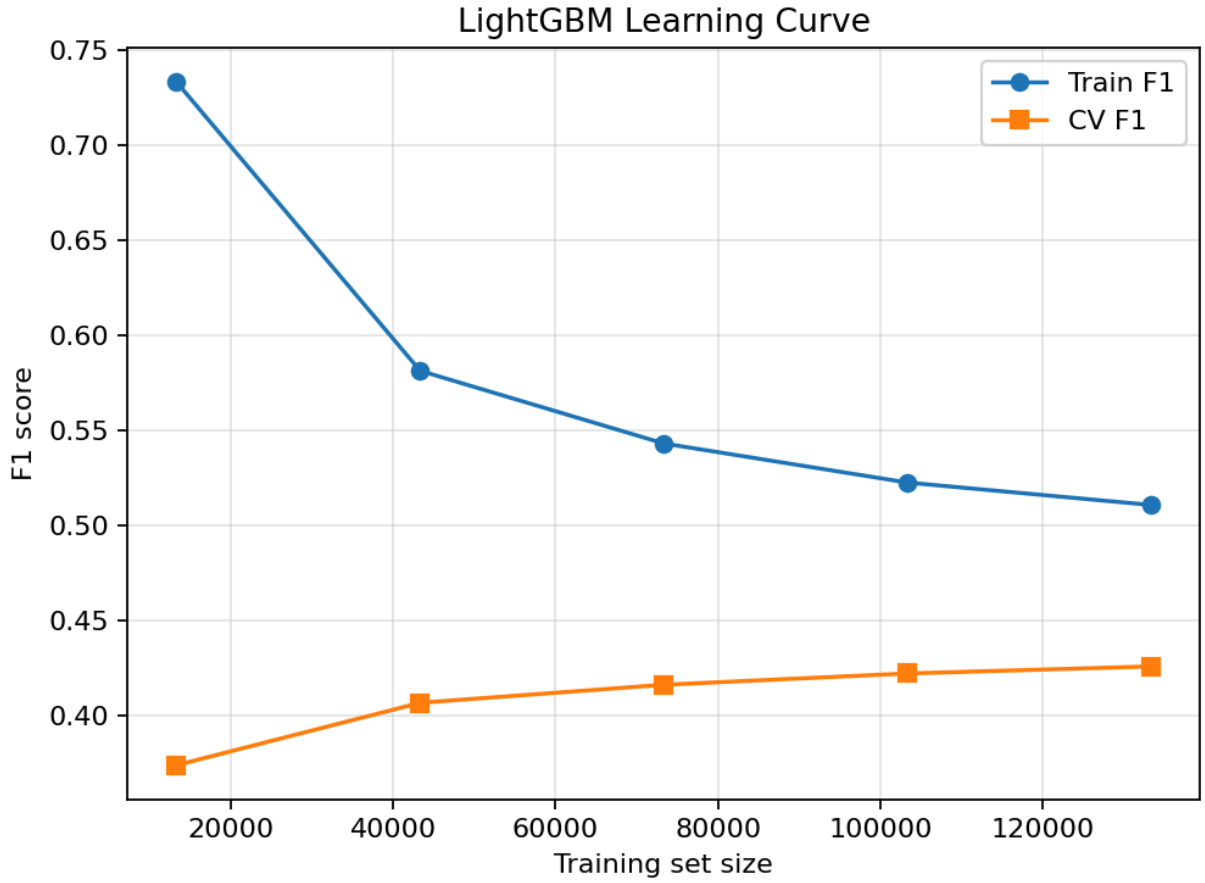
differences and suggests targeted interventions (e.g., more buffer on evening turns in particular clusters).



5.5 Complexity and learning curves

Using a diagnostic mode, we generate:

- Complexity curves (F1 vs. max_depth or num_leaves for LightGBM).
  These curves show that very shallow trees underfit (low F1), performance improves up to a moderate depth, and then plateaus, with little gain beyond the chosen depth—indicating that our final hyperparameters sit near a good bias–variance trade-off.
- Learning curves (F1 vs. training set size).
  These show that performance improves quickly when going from small samples to several hundred thousand rows, then exhibits diminishing returns. The curves flatten as we approach the full dataset, suggesting that we are close to the maximum value obtainable from these features and that simply adding more 2024 BTS records would not dramatically change performance.

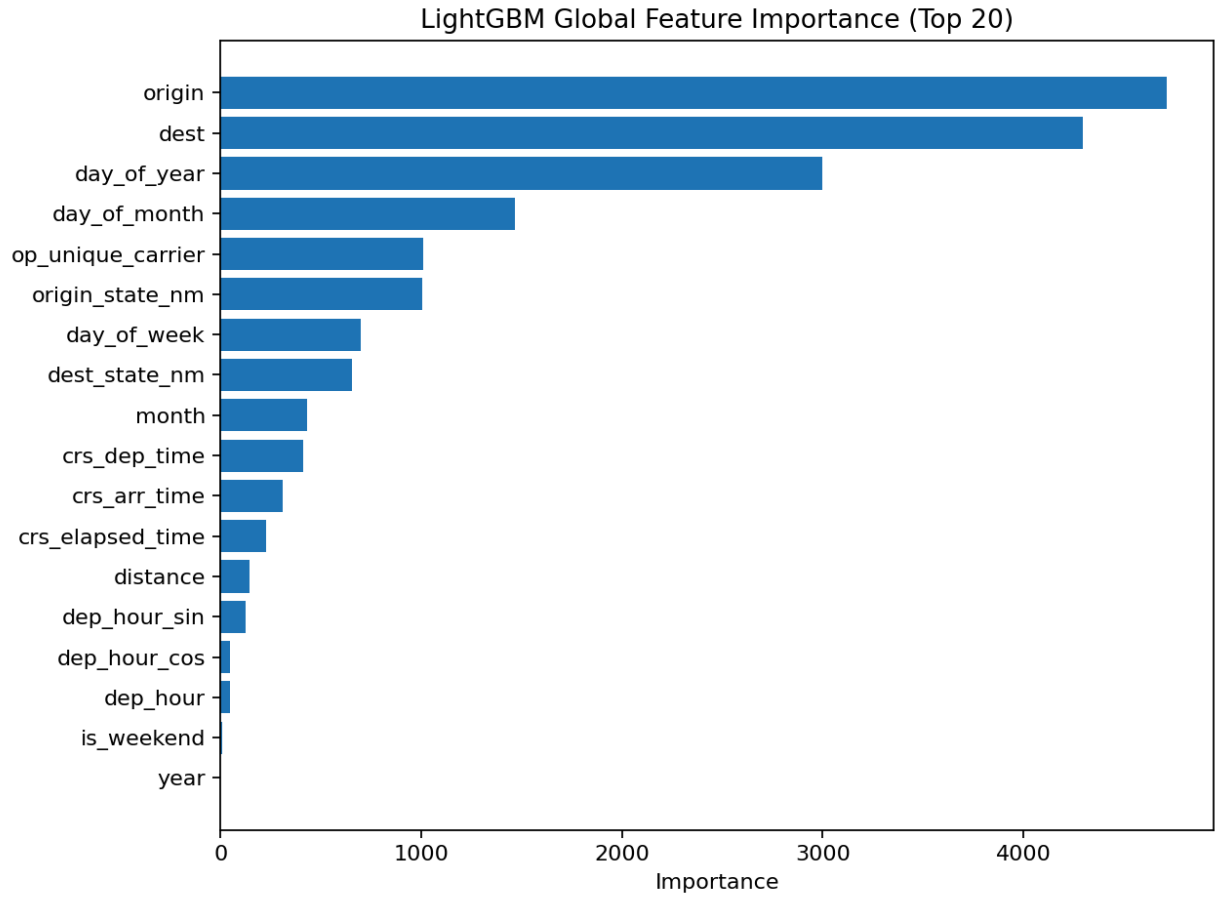


Both diagnostics support the model's stability and generalizability and show that we are not blindly over-fitting a complex model.

5.6 Feature importance and interpretability

To open up the "black box" of ensemble models, we extract feature importance from the final LightGBM classifier:

The below figure shows that the route is the primary risk behind delays followed by the day of the year, month and carrier. This shows that schedule timing and cyclic features add only marginal information, reinforcing that pre-flight delay risk is largely structural rather than flight-specific.



6. Reflection, Strengths, and Limitations

6.1 Model performance summary

The final model stack achieves:

- A strong LightGBM classifier that clearly outperforms logistic regression and random forest in F1 and ROC-AUC.
- A meaningful EV analysis that ties threshold choice to business costs and benefits.
- A LightGBM regressor that modestly improves over the naive baseline but confirms that exact delay minutes are noisy.

Clustering adds a complementary view, revealing that some flight regimes (especially evening operations) are structurally riskier than others.

6.2 Strengths

- Full data science lifecycle: The project covers business understanding, EDA, data preparation, supervised and unsupervised modeling, and reflection, aligning with the assignment structure.
- Leakage control: The model is strictly limited to pre-flight information; delay cause fields and realized weather or operational data are not used as predictors.
- Decision orientation: Threshold sweeps and EV analysis explicitly connect statistical performance to operational cost–benefit trade-offs.
- Interpretability: Feature importance and clustering outputs translate the model's mathematics into understandable narratives about when and where delays happen.

6.3 Limitations

- Weather detail: We rely on BTS weather_delay as a descriptive variable but do not integrate rich external meteorological data (e.g., NOAA hourly observations), which could improve predictive power if carefully lagged and leakage-controlled.
- Event coverage: The event dataset focuses on major sports and stadium events; other high-impact events (concerts, conferences, holidays beyond event dates) are only indirectly captured through seasonal patterns.
- Model diversity: We focus on LightGBM and standard ensemble baselines; other competitive approaches (XGBoost, CatBoost, or even deep learning architectures) are not explored in detail.
- Regression usefulness: Even with gradient boosting, regression errors remain large relative to operational thresholds, limiting the direct usefulness of point delay predictions.

6.4 Future improvements

Concrete extensions to strengthen the project:

- Integrate lagged, pre-flight weather forecasts for origin and destination airports, with careful temporal alignment.
- Broaden event coverage beyond sports (concerts, large conferences, holidays), potentially using web-scraped event calendars.
- Experiment with additional models such as XGBoost or CatBoost and compare them in terms of AUC, F1, and EV.
- Implement SHAP value analysis for the final LightGBM model to provide flight-level explanations.
- Deploy the model as a small dashboard (e.g., Streamlit) to expose risk scores and scenario analyses to non-technical stakeholders.

7. Conclusion

This project demonstrates a complete and conceptually grounded predictive analytics workflow for U.S. flight delays in 2024. We show that:

- Flight delays are not random; they are strongly structured by time-of-day, day-of-week, airport, carrier, route length, and event days.
- A carefully constructed LightGBM classifier, constrained to pre-flight information, can meaningfully predict delay risk despite class imbalance, outperforming simpler baselines.
- Threshold sweeps combined with an explicit EV framework turn model outputs into financially interpretable decisions rather than abstract scores where 0.3 probability threshold is the recommended operating point.
- Regression provides modest value for estimating delay magnitude, while clustering reveals underlying operational regimes with systematically different risk profiles.

By combining scalable data preparation, modern ensemble methods, advanced diagnostics (ROC/AUC, learning and complexity curves), EV analysis, and feature-level interpretability, the project not only satisfies the technical requirements of the assignment but also illustrates how predictive models can be responsibly integrated into real operational decision-making in aviation logistics.