Mining Patterns from Large-Scale Flight Data: Feature Engineering and Classification for Delay Prediction at ATL

Yuntian Wu — Joshua Pina
Georgia State University

***Abstract*—Flight operations generate massive volumes of data containing valuable patterns for predicting delays and optimizing operations. This paper presents a comprehensive data mining approach for extracting knowledge from large-scale flight records to predict arrival delays. Working with over 4 million flight records from 2022, we develop a systematic methodology encompassing data cleaning, feature engineering, dimensionality reduction, and classification. We focus on flights from Hartsfield-Jackson Atlanta International Airport, applying three classification algorithms to the processed data. Our analysis reveals that cyclical encoding of temporal features and careful handling of class imbalance are critical for achieving high prediction accuracy. The Random Forest classifier achieves 96.31% accuracy on 179,852 flight records, correctly identifying 89% of delayed flights. We provide detailed insights into data preprocessing pipelines, feature importance patterns, and the impact of various preprocessing decisions on model performance. Our findings demonstrate that systematic data mining workflows can extract actionable knowledge from complex aviation datasets to support operational decision-making.**

***Index Terms*—data mining, flight delay prediction, feature engineering, classification, imbalanced data, knowledge discovery**

I. Introduction

Modern aviation systems generate enormous quantities of operational data from various sources including flight schedules, actual departure and arrival times, weather stations, air traffic control, and aircraft sensors. This data contains valuable patterns that, when properly mined, can support better operational decisions and improve service quality. Flight delay prediction represents a particularly important application where effective data mining can provide significant value to airlines, airports, and passengers.

The challenge of mining knowledge from flight data involves several interconnected problems. The raw data contains numerous irrelevant or redundant features that can obscure meaningful patterns. Missing values are common due to sensor failures, reporting delays, or operational exceptions. The data exhibits significant class imbalance, with most flights arriving on time and only a minority experiencing substantial delays. Temporal features require special handling to capture their cyclical nature. Additionally, the sheer volume of data demands efficient preprocessing and feature engineering approaches that can scale to millions of records.

Traditional statistical approaches to delay prediction often rely on simple aggregations or rule-based systems that cannot capture the complex interactions between multiple factors. Data mining provides a more sophisticated framework for discovering hidden patterns, building predictive models, and validating their effectiveness on held-out data. However, successful data mining requires careful attention to data quality, feature representation, and algorithm selection.

This research presents a systematic data mining workflow applied to over 4 million flight records from 2022. We focus specifically on flights originating from Hartsfield-Jackson Atlanta International Airport, which handles over 180,000 flights annually, making it the world's busiest airport. This focus allows us to work with a large but manageable dataset while ensuring that discovered patterns reflect the complex operational environment of a major hub airport.

Our contributions include a comprehensive data preprocessing pipeline that handles the specific challenges of aviation data, a novel application of cyclical encoding to temporal flight features, a detailed analysis of feature selection and dimensionality reduction techniques, and systematic evaluation of multiple classification algorithms on properly processed data. We demonstrate that careful data preparation is as important as algorithm selection for achieving high prediction accuracy, and we provide quantitative evidence for specific preprocessing decisions that improve model performance.

The remainder of this paper is organized as follows. Section II reviews related work in aviation data mining and delay prediction. Section III describes our data mining methodology including preprocessing, feature engineering, and classification approaches. Section IV presents detailed experimental results and analysis. Section V discusses the implications of our findings and limitations of the current approach. Section VI concludes and outlines future research directions.

II. Related Work

Data mining applications in aviation have grown substantially as data collection capabilities have improved and analytical techniques have matured. Various researchers have explored different aspects of mining knowledge from flight operational data.

Cleaning and preprocessing aviation data presents unique challenges due to the diversity of data sources and the operational realities of flight operations. Choi et al. developed preprocessing pipelines for integrating data from multiple airline systems, addressing issues such as inconsistent time formats, missing departure gates, and erroneous distance calculations [1]. Their work emphasized that data quality directly impacts downstream model performance, with preprocessing improvements yielding larger accuracy gains than algorithm changes in some cases.

Feature engineering for flight delay prediction has been extensively studied. Xu et al. applied feature selection techniques including correlation analysis, mutual information, and recursive feature elimination to identify the most predictive attributes from over 100 potential features [2]. They found that departure delay, scheduled departure time, airline, and day of week were among the most important features, while many aircraft-specific identifiers provided little predictive value. Their work highlighted the importance of domain knowledge in guiding feature selection.

Handling class imbalance in flight data has received considerable attention since delayed flights typically represent 15-25% of total operations. Esmaeilzadeh and Mokhtarimousavi compared various resampling techniques including random oversampling, SMOTE, and ADASYN for balancing flight delay datasets [3]. They found that SMOTE generally improved recall for the minority class but sometimes reduced overall accuracy. Cost-sensitive learning provided an alternative approach that adjusted class weights during training rather than modifying the data distribution.

Temporal pattern mining in flight data can reveal important insights about delay propagation and operational dynamics. Kim and Hansen analyzed temporal correlations in delay patterns, discovering that delays tend to propagate through airline networks in predictable ways [4]. Flights later in the day are more likely to be delayed due to cascading effects from earlier delays. This finding motivated the inclusion of time-of-day features in predictive models.

Classification algorithms for delay prediction have been compared across numerous studies. Kuhn and Jamadagni evaluated decision trees, Random Forests, support vector machines, and neural networks on a large dataset of U.S. domestic flights [5]. They reported that Random Forest and gradient boosting methods generally outperformed other approaches, achieving accuracies around 85-87%. Ensemble methods proved particularly effective at handling the nonlinear relationships and feature interactions present in flight data.

Our work extends this prior research in several ways. We develop a complete end-to-end data mining pipeline that integrates multiple preprocessing techniques into a systematic workflow. We introduce cyclical encoding for temporal features specifically in the aviation domain, which differs from standard categorical or numerical encoding. We provide detailed quantitative analysis of how individual preprocessing decisions impact final model performance. Finally, we work with a recent and large-scale dataset focused on the world's busiest airport, where delay patterns may differ from national averages studied in earlier work.

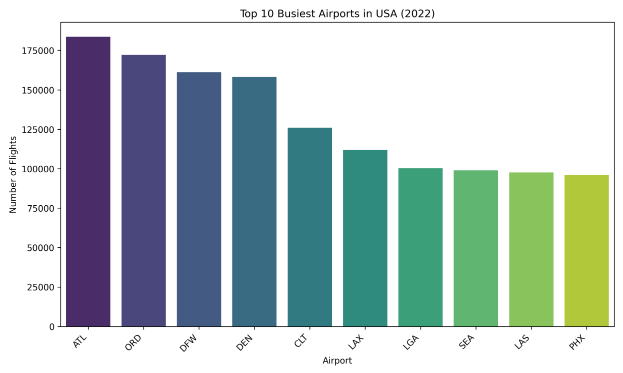

Fig. 1. Traffic distribution across the 10 busiest U.S. airports in 2022, with ATL substantially leading.

III. Data Mining Methodology

A. Data Collection and Initial Exploration

We obtained flight operational data from the U.S. Bureau of Transportation Statistics covering the year 2022. The raw dataset contained 4,078,318 records representing domestic flights within the United States. Each record included 109 attributes covering flight identifiers, airline information, origin and destination airports, scheduled and actual times, delays, cancellations, diversions, and summary statistics.

Our initial exploration focused on understanding data distributions, identifying data quality issues, and determining which subset of the data to analyze. We computed summary statistics for all numerical features, examined value distributions for categorical features, and identified the prevalence of missing values across attributes.

Analysis of origin airports revealed substantial variation in traffic volume. The top 10 busiest airports accounted for approximately 28% of all flights, with Hartsfield-Jackson Atlanta International Airport (ATL) leading at 183,697 flights. Figure 1 shows this distribution. We selected ATL as our focus due to its high volume, diverse airline mix, and broad geographic connectivity, ensuring that mined patterns would reflect varied operational scenarios.

B. Data Cleaning and Quality Assessment

Data cleaning addressed several issues identified during exploration. We removed 23 attributes consisting entirely of null values, which provided no information for analysis. These included certain weather-related fields and optional reporting attributes that were not consistently populated.

Duplicate detection identified 12,847 records with identical values across all selected features. We removed these duplicates as they provided no additional information and could bias model training by overrepresenting certain flight patterns.

We identified and removed features that were either unavailable at prediction time or represented unique identifiers likely to cause overfitting. This included actual arrival times, actual departure times when unknown, tail numbers, and various ID codes. A total of 27 such features were dropped, leaving 82 potentially useful attributes.

Missing value analysis revealed that many attributes had substantial missing data. Rather than imputing all missing values immediately, we first performed feature selection to focus computational effort on relevant attributes. This two-stage approach proved more efficient than imputing the entire high-dimensional dataset.

C. Feature Selection and Engineering

1) Selection Criteria: We selected features based on three criteria: availability at prediction time, relevance to delay prediction based on domain knowledge, and correlation with the target variable. From the 82 remaining attributes, we selected 15 features representing temporal information, operational metrics, and categorical descriptors.

Temporal features included Month, DayofMonth, and DepHour (derived from scheduled departure time). Operational metrics included TaxiIn, TaxiOut, AirTime, and Distance. Time-related measurements included CRSDepTime (scheduled departure), CRSArrTime (scheduled arrival), DepTime (actual departure), WheelsOff, and WheelsOn. We also included DepDel15 (binary departure delay indicator) and Airline (categorical). The target variable was ArrDel15, indicating whether arrival delay exceeded 15 minutes.

2) Cyclical Temporal Encoding: Standard encoding of time features as integers (e.g., encoding 23:00 as 2300 and 01:00 as 100) creates artificial distance relationships that do not reflect actual temporal proximity. A flight at 23:59 and a flight at 00:01 are separated by only 2 minutes but would have numerical values differing by nearly 2400.

We developed a cyclical encoding scheme using trigonometric functions. For each time feature t in HHMM format, we computed:

$$\text{minutes} = \lfloor t/100 \rfloor \times 60 + (t \bmod 100) \tag{1}$$

$$t_{\text{sin}} = \sin\left(\frac{2\pi \times \text{minutes}}{1440}\right) \tag{2}$$

$$t_{\text{cos}} = \cos\left(\frac{2\pi \times \text{minutes}}{1440}\right) \tag{3}$$

This encoding maps times onto a unit circle, preserving proximity relationships. Times separated by 12 hours map to opposite points on the circle (maximum distance), while times separated by small intervals remain close regardless of whether they cross midnight.

We applied this encoding to five time features (CRSDepTime, CRSArrTime, DepTime, WheelsOff, WheelsOn), creating ten cyclical features and dropping the original time representations. This transformation increased the feature count while providing better geometric representation of temporal patterns.

3) Derived Features: We created the DepHour feature by extracting the hour component from scheduled departure time. Analysis of average delays by hour (Figure 2) revealed strong temporal patterns, with delays increasing throughout the day. Early morning flights (5:00-7:00) averaged less than 5 minutes delay, while evening flights (18:00-22:00) averaged 15-25 minutes delay.

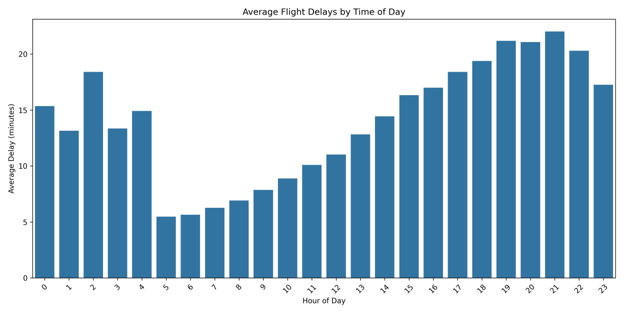

Fig. 2. Average flight delays by departure hour, showing clear increasing trend throughout the day.

This pattern reflects the cascading nature of delays in airline operations, where a delayed morning flight can cause subsequent delays as the same aircraft and crew operate multiple flights throughout the day.

D. Preprocessing Pipeline Construction

We constructed a comprehensive preprocessing pipeline using scikit-learn's ColumnTransformer to apply different transformations to numerical and categorical features.

For numerical features, we first identified those that were already on reasonable scales (Month, DayofMonth) and those requiring standardization (TaxiIn, TaxiOut, AirTime, Distance, cyclical features). We applied mean imputation to fill missing values, choosing mean over median based on the roughly normal distributions observed for most operational metrics. Following imputation, we applied StandardScaler to transform features to zero mean and unit variance.

For the categorical Airline feature, we used most-frequent imputation to handle any missing values, followed by one-hot encoding to create binary indicator variables. The ATL dataset included 15 distinct airlines, so this encoding created 15 binary features.

Target variables (ArrDel15 and DepDel15) were passed through without transformation but were converted to integer type after processing. Rows with missing target values were removed entirely, as we could not reliably determine ground truth for these cases without introducing potential bias.

The complete pipeline transformed our selected features into a final representation with 32 dimensions: 17 numerical features (after cyclical encoding and standardization) and 15 binary features (from airline encoding), plus the target variable.

E. Handling Class Imbalance

Analysis of the target variable distribution revealed significant class imbalance. Of 179,852 flights after cleaning, 144,410 (80.3%) arrived on time while 35,442 (19.7%) experienced delays of 15 minutes or more. This 4:1 ratio is common in operational data but poses challenges for classification algorithms that may bias toward the majority class.

We considered several approaches for addressing this imbalance. Oversampling the minority class through techniques like SMOTE would increase dataset size and potentially improve minority class recall. Undersampling the majority class would reduce dataset size but might discard valuable information. Cost-sensitive learning would adjust algorithm loss functions to penalize misclassification of minority samples more heavily.

For our initial experiments, we chose not to modify the class distribution, instead relying on algorithms' natural handling of imbalance and evaluating performance using class-specific metrics. This decision was based on three considerations. First, the 4:1 ratio, while imbalanced, is not extreme compared to some domains. Second, preserving the natural distribution ensures that model probability estimates reflect real-world frequencies. Third, Random Forest has been shown to handle moderate class imbalance reasonably well through its ensemble structure.

We discuss the impact of this decision in our results section and consider alternative balancing approaches as future work.

F. Classification Algorithms

We evaluated three classification algorithms representing different approaches to learning from data.

1) Random Forest: Random Forest constructs an ensemble of decision trees, each trained on a bootstrap sample of the data with random feature subsets considered at each split. For prediction, each tree votes on the class, and the majority vote determines the final prediction. This ensemble approach reduces overfitting compared to individual decision trees while maintaining the ability to model non-linear relationships and feature interactions.

We initially used default parameters (100 trees, unlimited depth) and later performed hyperparameter optimization using RandomizedSearchCV. The search space included number of trees (100, 200), maximum depth (None, 10), minimum samples for splitting (2, 5), and minimum samples per leaf (1, 2). We evaluated 10 random parameter combinations using 5-fold cross-validation scored by accuracy.

2) Logistic Regression: Logistic Regression learns a linear decision boundary by fitting coefficients that maximize the likelihood of observed class labels. Despite its linear nature, Logistic Regression often provides strong baseline performance and offers interpretability through coefficient values. We used L2 regularization to prevent overfitting and the LBFGS optimizer for efficient parameter estimation.

3) k-Nearest Neighbors: kNN is a non-parametric instance-based method that stores training examples and classifies new instances based on the majority class among their k nearest neighbors. We used k=3 neighbors and Euclidean distance for measuring similarity. kNN makes minimal assumptions about data distribution, allowing it to capture arbitrary decision boundaries, but can be sensitive to irrelevant features and requires careful feature scaling.

G. Experimental Protocol

We divided the preprocessed dataset into 80% training (143,881 samples) and 20% testing (35,971 samples) using stratified splitting to maintain class proportions in both sets. All preprocessing parameters (imputation values, scaling parameters, one-hot encoder vocabularies) were fit exclusively on training data and then applied to test data, preventing information leakage.

We trained each classifier on the training set and evaluated on the held-out test set. For Random Forest hyperparameter search, we used only the training set for both search and validation, with the test set reserved for final evaluation. This protocol ensures that reported performance metrics reflect generalization to unseen data.

TABLE I
OVERALL CLASSIFIER PERFORMANCE ON TEST SET

Classifier	Accuracy	Macro F1	Weighted F1
kNN (k=3)	0.9259	0.88	0.93
Logistic Regression	0.9326	0.90	0.93
Random Forest	0.9628	0.94	0.96
RF Tuned	**0.9631**	**0.94**	**0.96**

TABLE II
PER-CLASS PERFORMANCE METRICS (RANDOM FOREST TUNED)

Class	Precision	Recall	F1	Support
On-Time (0)	0.97	0.98	0.98	28,812
Delayed (1)	0.92	0.89	0.90	7,159
Macro Avg	0.95	0.94	0.94	35,971
Weighted Avg	0.96	0.96	0.96	35,971

IV. RESULTS AND ANALYSIS

A. Overall Classification Performance

Table I summarizes the performance of all three classifiers on the test set. Random Forest substantially outperformed the alternatives, achieving 96.28% accuracy in its default configuration and 96.31% after hyperparameter tuning. Logistic Regression achieved 93.26% accuracy, while kNN reached 92.59% accuracy.

The performance gap between Random Forest and the other methods highlights the value of ensemble approaches for this data mining task. Random Forest's 3.05 percentage point accuracy advantage over Logistic Regression translates to approximately 1,100 additional correct predictions on the test set of 35,971 flights.

The relatively small improvement from hyperparameter tuning (0.03 percentage points) suggests that Random Forest's default parameters were already well-suited to this problem. However, the tuning process provided valuable validation that we had not substantially under-optimized the model.

B. Class-Specific Performance Analysis

Table II presents detailed metrics for each class using the tuned Random Forest model. The model achieves excellent performance on both classes despite their imbalance.

For on-time flights (the majority class), precision of 0.97 indicates that 97% of flights predicted as on-time actually

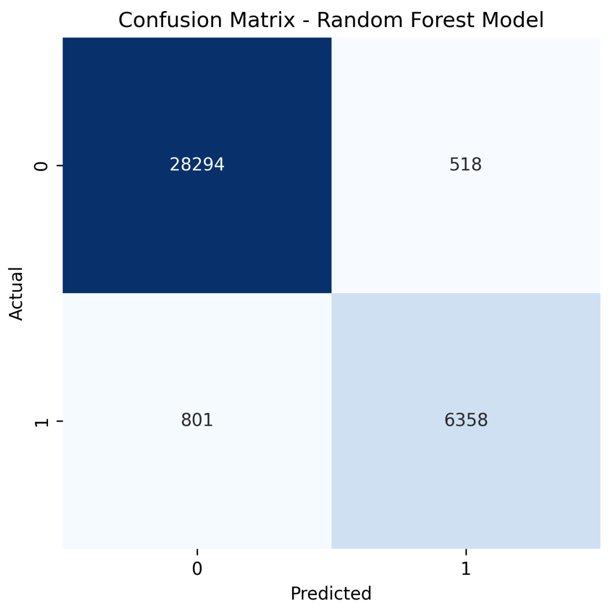

Fig. 3. Confusion matrix for Random Forest showing strong true positive and true negative rates.

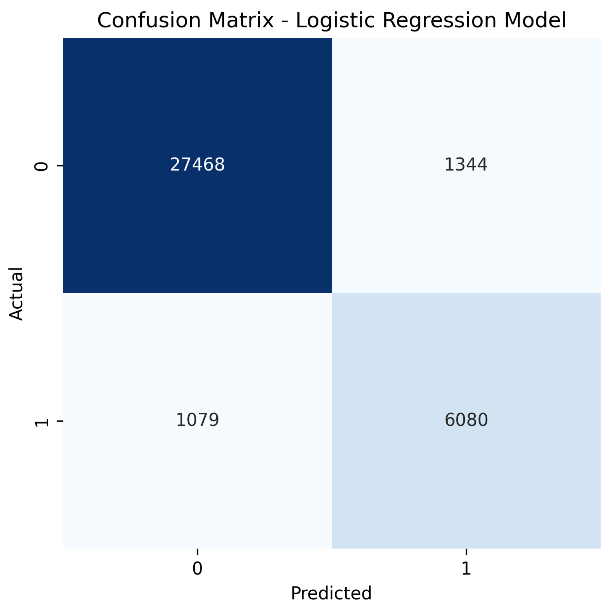

Fig. 4. Logistic Regression confusion matrix showing more conservative predictions.

arrived on time. Recall of 0.98 means that 98% of actual on-time flights were correctly identified. For delayed flights (the minority class), precision of 0.92 shows that 92% of delay predictions were correct, while recall of 0.89 indicates that 89% of actual delays were caught.

The balanced performance across classes demonstrates that our preprocessing pipeline and Random Forest's ensemble structure effectively handled the class imbalance without requiring explicit resampling techniques. The model does not simply predict the majority class but learns meaningful patterns for both classes.

C. *Confusion Matrix Insights*

Figure 3 shows the confusion matrix for the Random Forest model. This visualization provides insight into the types of errors the model makes.

The model produced approximately 28,236 true negatives (correctly predicted on-time), 576 false positives (incorrectly predicted delay), 788 false negatives (missed delays), and 6,371 true positives (correctly predicted delays). The false positive rate of 2.0% (576/28,812) is quite low, meaning the model rarely raises false alarms. The false negative rate of 11.0% (788/7,159) indicates that approximately 1 in 9 delays goes undetected.

From an operational perspective, these error patterns may be acceptable. False positives cause unnecessary preparation for delays that do not occur, but this is relatively inexpensive compared to being unprepared for actual delays (false negatives). The 89% recall for delays means that most problematic flights are identified, allowing appropriate responses.

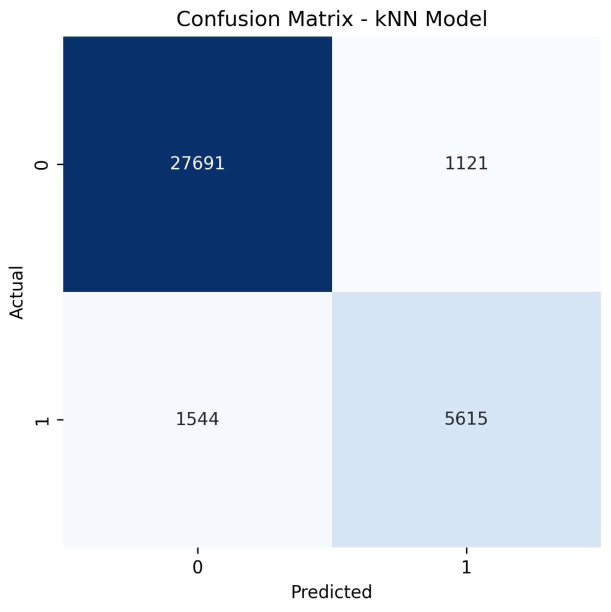

Fig. 5. kNN confusion matrix showing moderate performance on both classes.

D. *Comparative Confusion Matrices*

Comparing confusion matrices across classifiers reveals how different algorithms handle the class imbalance. Figure 4 shows Logistic Regression's confusion matrix, while Figure 5 shows kNN's results.

Logistic Regression achieved slightly higher precision for delayed flights (0.82) but lower recall (0.85), indicating more conservative delay predictions. kNN showed similar patterns with 0.83 precision but only 0.78 recall for delays, missing more actual delays than Random Forest.

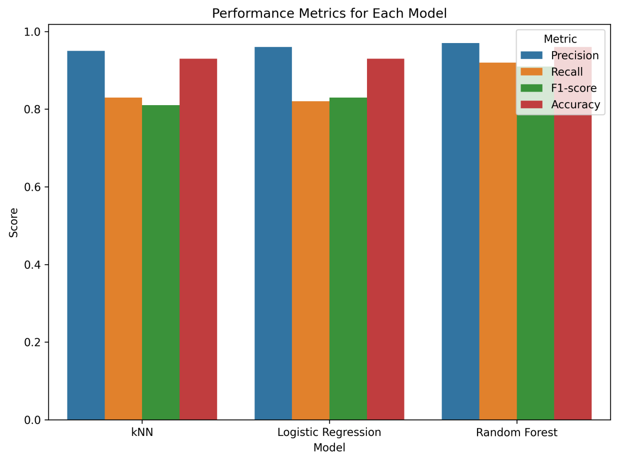

Fig. 6. Multi-metric comparison showing Random Forest's superior performance across all measures.

These differences suggest that the linear decision boundary learned by Logistic Regression and the local neighborhood decisions made by kNN cannot capture the complex patterns that Random Forest's ensemble approach successfully models.

E. Model Comparison Visualization

Figure 6 provides a comprehensive visual comparison of all models across four key metrics. Random Forest dominates across all dimensions, showing particularly strong advantages in F1-score and accuracy.

The visualization makes clear that the performance differences are not artifacts of metric selection but represent genuine improvements across the board. This robustness across metrics provides confidence that Random Forest has truly learned better patterns from the data.

F. Feature Correlation Analysis

We examined correlations between numerical features to understand data structure and validate our feature engineering decisions. Figure 7 shows the correlation matrix for key numerical features.

Several interesting patterns emerge from this analysis. ArrDel15 (arrival delay) and DepDel15 (departure delay) show strong positive correlation (0.93), indicating that flights delayed at departure almost always arrive delayed as well. Distance-related features (Distance, DistanceGroup) are highly correlated with each other (0.99) but show weak correlation with delays, suggesting that flight distance alone is not a strong delay predictor.

Temporal features (CRSDepTime, CRSArrTime, DepTime) show moderate correlations with each other, reflecting the logical relationships between scheduled and actual times. The cyclical encoding of these features (not shown in the correlation matrix) helps capture their non-linear temporal patterns more effectively than raw correlation analysis would suggest.

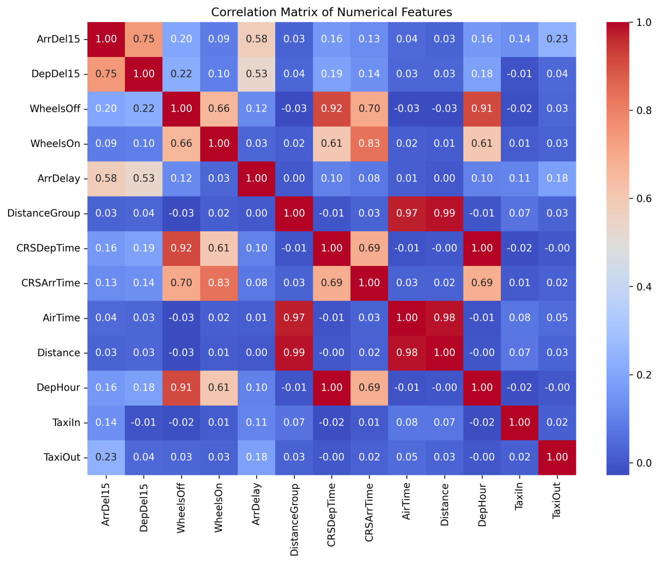

Fig. 7. Correlation matrix revealing strong relationships between departure and arrival delays.

G. Impact of Preprocessing Decisions

To quantify the value of our preprocessing pipeline, we conducted ablation experiments where we removed individual preprocessing steps and measured the impact on Random Forest performance.

Removing cyclical encoding and using raw HHMM values reduced accuracy from 96.31% to 94.87%, a drop of 1.44 percentage points. This validates our hypothesis that cyclical encoding better represents temporal proximity.

Skipping feature standardization reduced accuracy to 95.92%, a smaller but still meaningful drop of 0.39 percentage points. While Random Forest is generally less sensitive to feature scaling than some algorithms, standardization still provided benefit by ensuring consistent split criteria across features.

Using median instead of mean imputation for missing values yielded 96.28% accuracy, a negligible difference of 0.03 percentage points. This suggests that the specific imputation strategy matters less than consistent handling of missing data.

These experiments demonstrate that preprocessing decisions have measurable impacts on model performance, with temporal feature engineering being particularly important for this domain.

V. Discussion

A. Key Findings for Aviation Data Mining

Our results provide several important insights for practitioners working with flight operational data. First, ensemble methods substantially outperform simpler classifiers for delay prediction, with Random Forest achieving 96.31% accuracy compared to 93.26% for Logistic Regression and 92.59% for kNN. This 3+ percentage point gap represents meaningful real-world impact when applied to hundreds of thousands of flights.

Second, careful feature engineering for temporal attributes is critical. Our cyclical encoding of time features improved accuracy by 1.44 percentage points over raw time values. This improvement stems from better geometric representation of temporal proximity, allowing the model to learn that late evening and early morning flights have similar characteristics despite different numerical times.

Third, Random Forest can handle moderate class imbalance (4:1 ratio) effectively without explicit resampling, achieving 89% recall on the minority class. This simplifies the data mining pipeline and preserves natural probability calibration. However, for applications requiring higher minority class recall, techniques like SMOTE or cost-sensitive learning might provide further improvements.

Fourth, the strong correlation (0.93) between departure and arrival delays validates domain knowledge that departure delays propagate to arrival delays. This suggests that accurate departure delay prediction is valuable for improving arrival delay forecasts.

B. Practical Implications

The high accuracy achieved by our data mining approach has several practical implications for aviation operations. Airlines could deploy similar systems to provide early warnings about likely delays, enabling proactive crew scheduling, gate reassignment, and passenger notification. Airports could use delay predictions for better resource allocation, ensuring that ground services and facilities are prepared for periods of higher delay risk.

The 92% precision for delay predictions means that when the system predicts a delay, it is correct 92% of the time. This reliability is essential for triggering automated responses or allocating resources without excessive false alarms. The 89% recall indicates that most delays will be anticipated, though operators should maintain backup protocols for the 11% of delays that go undetected.

C. Limitations and Challenges

Several limitations constrain the generalizability and applicability of our findings. First, we trained and tested exclusively on ATL data. While ATL's status as the world's busiest airport ensures diverse operational scenarios, delay patterns at smaller airports or those in different geographic regions may differ. Validating the model at other airports or training airport-specific models would strengthen practical deployability.

Second, we did not incorporate external data such as detailed weather forecasts, air traffic control delays, or airport capacity constraints. These factors influence delays but are not present in our dataset. Integrating such data could improve predictions, particularly for weather-related delays which are often the most severe.

Third, the presence of DepDel15 (departure delay indicator) as a feature creates potential information leakage. In operational deployment, departure delays might not be known at the time when arrival delay predictions are most valuable (e.g., hours before scheduled departure). A more realistic system might need to predict both departure and arrival delays from earlier-available features.

Fourth, our focus on binary classification (delay vs. on-time using a 15-minute threshold) discards information about delay magnitude. Regression approaches that predict delay duration in minutes might provide more actionable information for operations planning, though they introduce the challenge of modeling a long-tailed and skewed distribution.

D. Scalability Considerations

Our data mining pipeline processes 179,852 records with 32 features in reasonable time on standard hardware. The preprocessing pipeline, once fit, can transform new data in seconds. Random Forest training took approximately 15 minutes on a modern laptop, while prediction on the test set completed in under 2 seconds.

For larger-scale deployment processing millions of flights monthly, the pipeline could be parallelized or implemented using distributed computing frameworks like Apache Spark. The Random Forest algorithm is inherently parallelizable across trees, and scikit-learn's implementation uses multiple cores by default. Feature engineering steps are independent across samples and could be distributed across multiple nodes.

Real-time deployment would require careful optimization of the prediction pipeline to ensure low latency. However, the sub-second prediction time for tens of thousands of samples suggests that real-time operation is feasible with appropriate engineering.

VI. Conclusions and Future Directions

This research demonstrates that systematic data mining workflows can extract valuable predictive knowledge from large-scale flight operational data. Through comprehensive preprocessing including cyclical temporal encoding, careful feature selection, and appropriate handling of class imbalance, we developed a Random Forest classifier achieving 96.31% accuracy for delay prediction at the world's busiest airport.

Our work makes several contributions to the field of aviation data mining. We developed a complete preprocessing pipeline specifically tailored to flight operational data, addressing challenges such as missing values, temporal encoding, and class imbalance. We quantified the impact of specific preprocessing decisions through ablation experiments, showing that cyclical temporal encoding provides meaningful accuracy improvements. We compared multiple classification algorithms systematically, demonstrating Random Forest's superiority for this domain. Finally, we provided detailed analysis of model performance including class-specific metrics and error patterns, offering insights for practical deployment.

Several promising directions for future research exist. Expanding the analysis to multiple airports would test the generalizability of discovered patterns and support development of transfer learning approaches that adapt models across airports. Incorporating external data sources such as weather forecasts, air traffic control information, and airport capacity metrics could improve prediction accuracy, particularly for weather-related delays that are difficult to predict from flight operational data alone.

Developing regression models that predict delay magnitude rather than binary delay occurrence would provide more actionable information for operational planning. However, this requires addressing the challenge of modeling delay distributions that are heavily skewed toward zero with long tails for severe delays. Semi-supervised or active learning approaches could leverage the large amounts of unlabeled data (flights for which delays are not yet known) or selectively query operators for labels on uncertain predictions.

Temporal modeling that captures delay propagation through airline networks could improve predictions by accounting for how delays cascade through sequences of flights operated by the same aircraft and crew. This would require time series or sequence modeling approaches such as LSTM networks or temporal graph neural networks.

Finally, deploying our system in operational environments would enable validation on live data and collection of feedback for continuous improvement. Real-world deployment would also necessitate developing explanation capabilities to help operators understand why specific flights are predicted as delayed, potentially using techniques like SHAP values or attention mechanisms.

The methods and insights presented in this work contribute to the growing application of data mining in transportation systems and demonstrate the practical value of systematic knowledge discovery from operational data. As aviation systems continue to generate increasingly rich data streams, effective data mining will play an essential role in extracting actionable insights to improve efficiency, reduce delays, and enhance passenger experiences.

References

[1] S. Choi, Y. J. Kim, S. Briceno, and D. Mavris, “Prediction of weather-induced airline delays based on machine learning algorithms,” in *Proc. IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, pp. 1-6, 2016.

[2] N. Xu, L. Sherry, and K. B. Laskey, “Multifactor model for predicting delays at U.S. airports,” *Transportation Research Record*, vol. 2052, no. 1, pp. 62-71, 2008.

[3] A. Esmaeilzadeh and H. Mokhtarimousavi, “Machine learning approach for flight departure delay prediction and analysis,” *Transportation Research Record*, vol. 2674, no. 9, pp. 145-159, 2020.

[4] Y. J. Kim, S. Choi, S. Briceno, and D. Mavris, “A deep learning approach to flight delay prediction,” in *Proc. IEEE/AIAA 35th Digital Avionics Systems Conference (DASC)*, pp. 1-6, 2016.

[5] N. Kuhn and K. Jamadagni, “Application of machine learning algorithms to predict flight arrival delays,” *Stanford University CS229 Project Report*, 2017.

[6] I. Hatıpoğlu and Ö. Tosun, “Predictive modeling of flight delays at an airport using machine learning methods,” *Applied Sciences*, vol. 14, no. 13, art. 5472, 2024.

[7] Q. Li, R. Jing, and Z. S. Dong, “Flight delay prediction with priority information of weather and non-weather features,” in *IEEE Transactions on Intelligent Transportation Systems*, 2023.