Scalable Machine Learning Pipeline for Flight Delay Prediction Using Apache Spark and H2O.ai

Salma Al Emrany[1], Youssef Aboukhatwa[1], Omar Sherif Fouad[1]

[1]Department of Mathematics and Actuarial Science, The American University in Cairo, Cairo, Egypt.

Contributing authors: salmaye@aucegypt.edu; yaboukhatwa@aucegypt.edu; omar_sherif_fouad@aucegypt.edu;

Abstract

Flight delays are a major concern in the aviation industry, affecting both passengers and airline operations. In this study, we used big data tools and machine learning techniques to build a predictive model that can classify whether a flight will be delayed or not. We worked with a large dataset of over three million flight records and applied Apache Spark for scalable processing and H2O.ai for automated machine learning. Our workflow included data cleaning, feature engineering, and training several models including Logistic Regression, Random Forest, Gradient Boosted Trees, and H2O Deep Learning. We evaluated these models using key metrics such as accuracy, precision, recall, F1-score, and AUC. Gradient Boosted Trees and the Hybrid PCA+GBT model achieved the best performance. The results show that big data technologies can be effectively used for real-time flight delay prediction and can help improve decision-making in the aviation industry.

Keywords: Flight Delay Prediction, Big Data, Apache Spark, H2O.ai, AutoML, Gradient Boosted Trees

1 Introduction

Flight delays are one of the most common challenges in the aviation sector. They can result in major inconveniences for passengers and lead to financial losses for airlines. Predicting flight delays in advance can help minimize their impact by allowing airlines

and airports to make better operational decisions. With the massive amount of flight data available today, it has become possible to apply machine learning techniques to improve delay predictions.

In this project, we aim to build an accurate, scalable, and efficient machine learning pipeline that predicts whether a flight will be delayed using both Apache Spark and H2O.ai. Spark is well-known for handling large datasets efficiently using parallel processing, while H2O offers powerful AutoML capabilities that can automatically try different models and optimize their performance. By combining these technologies, we were able to build models that are not only accurate but also fast and reliable.

We used a large dataset of U.S. domestic flights, which included over three million records and dozens of features related to flight schedules, airlines, departure times, and delay causes. After performing data cleaning and selecting the most important features, we trained several machine learning models including Logistic Regression, Random Forest, Gradient Boosted Trees, and Deep Learning models from H2O. We also created a hybrid model using Principal Component Analysis (PCA) followed by Gradient Boosted Trees to test the effect of dimensionality reduction.

The performance of each model was evaluated using metrics such as accuracy, precision, recall, F1-score, and AUC (Area Under the Curve). We also created ROC curves and confusion matrices to visualize and interpret the results. Our findings were then compared with existing literature to understand how our approach performed relative to other research in the field.

2 Related Work

Flight delay prediction has been a prominent area of research within transportation analytics and machine learning. Many earlier studies relied on traditional statistical and machine learning models such as logistic regression, decision trees, and random forests, which provided a baseline for understanding delay behavior. More recent surveys have classified existing work into four main approaches: statistical methods, classical machine learning, deep learning, and hybrid models. Among these, ensemble techniques such as Random Forest and XGBoost are commonly reported to outperform single-model counterparts due to their robustness and ability to capture complex feature interactions.

Deep learning methods have also gained popularity, particularly LSTM networks and attention-based models, which are effective in learning temporal patterns in sequential flight and weather data. Researchers have also experimented with graph neural networks to capture the networked nature of air traffic, modeling how delays at one airport ripple across the network. While such models often yield higher accuracy, they require substantial computational resources, which can pose scalability issues.

Some studies have shifted toward real-time systems using frameworks like Apache Kafka and Spark Streaming to enable live delay prediction, reducing latency to just a few seconds. Others have explored explainable AI approaches using tools like SHAP values to improve interpretability for stakeholders such as airline operation teams. Additionally, dimensionality reduction and optimal feature selection are increasingly recognized for improving both efficiency and model performance.

Our work builds on these insights by combining Spark's scalability with H2O's automation and introducing hybrid modeling techniques that balance accuracy, transparency, and computational efficiency.

3 Methodology: Layered Framework

To develop a robust and scalable solution for flight delay prediction, we designed a multi-layered machine learning pipeline that leverages both Apache Spark and H2O.ai technologies. This layered framework addresses the entire lifecycle of a big data predictive system—from data ingestion to model evaluation and comparison.

1. Data Ingestion Layer

Our process begins with reading a large flight dataset consisting of millions of flight records. Using Spark DataFrames, the data was ingested from a CSV file stored in Google Drive. Spark's distributed processing capability ensured efficient loading and access to data, regardless of size.

2. Data Preprocessing & Cleaning Layer

We removed records containing null or missing values in critical fields such as departure delay, arrival delay, and distance. A new binary label column, **DELAYED**, was created based on arrival delay values greater than 15 minutes. This binary classification simplified the problem while aligning with airline industry standards for delay reporting.

3. Feature Engineering Layer

We engineered features by selecting two impactful numerical variables: **DEP_DELAY** and **DISTANCE**. These were transformed into feature vectors using Spark's **VectorAssembler**. To explore potential gains from dimensionality reduction, we also integrated Principal Component Analysis (PCA), enabling the creation of a hybrid pipeline combining PCA with Gradient Boosted Trees. This layer was essential for improving model performance and runtime efficiency.

4. Model Selection & Training Layer

We trained multiple classification models using Spark MLlib, including Logistic Regression, Random Forest, Gradient Boosted Trees, Decision Trees, and a PCA-enhanced GBT pipeline. In addition, we utilized H2O AutoML to automatically train, tune, and compare several models including deep learning-based ones. This dual-platform setup allowed for rich experimentation and benchmarking.

5. Evaluation Layer

Each model was evaluated using metrics such as accuracy, precision, recall, F1-score, and AUC-ROC. Confusion matrices were constructed to measure classification performance. These metrics were computed manually using prediction outputs and visualized through ROC curves for comparative analysis.

6. Prediction Layer

The best-performing models were used to generate predictions on unseen data. These results were visualized to communicate model confidence and delay probabilities. We ensured that the models could generalize well and respond effectively to real-world data patterns.

7. Deployment Layer

Though not implemented in this phase, our pipeline design supports future deployment through APIs or Spark Streaming. Models trained using H2O can be exported as MOJO/POJO files for scalable serving in cloud or production environments.

4 Data Preparation & Feature Engineering

The dataset used in this study contains three months' worth of domestic flight records from major U.S. airlines, including variables such as flight times, airport codes, aircraft information, and delay durations. Since the raw data was collected from real-world operations, it required thorough cleaning and preprocessing before it could be used for modeling.

We began the data preparation process by removing records with missing or null values in key columns such as DEP_DELAY, ARR_DELAY, DISTANCE, and the newly generated DELAYED label. These fields were essential for both training and evaluation. To define the prediction target, we introduced a binary classification label DELAYED, where flights with an arrival delay greater than 15 minutes were labeled as 1 (delayed), and all others as 0 (on time). This threshold follows common standards in the aviation industry for categorizing delays.

After cleaning, we moved on to feature engineering. Our initial exploration involved evaluating multiple variables, but for simplicity and performance, we selected two high-impact numerical features: DEP_DELAY (departure delay in minutes) and DISTANCE (flight distance in miles). These two variables were chosen based on domain knowledge and correlation analysis, as they had a strong statistical relationship with arrival delays.

The selected features were combined into a single vector using Spark's VectorAssembler, creating a column named features that could be passed directly into machine learning models. Additionally, we experimented with dimensionality reduction using Principal Component Analysis (PCA) to observe its effect on model performance. PCA reduced the feature space to two principal components, allowing for faster training times and improved interpretability when integrated into a hybrid Gradient Boosted Trees pipeline.

This preprocessing and feature engineering pipeline provided a clean and optimized foundation for building scalable and accurate predictive models.

5 Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) plays a foundational role in understanding the structure, distribution, and key patterns within the dataset. Our analysis began by isolating key numerical features relevant to flight performance, including departure delay, arrival delay, and distance. We conducted a correlation analysis across all numerical columns to identify relationships between features. The resulting heatmap provided immediate insight: the strongest correlation was observed between departure and arrival delays, reinforcing the intuitive expectation that late departures often lead to late arrivals.

We then explored the distribution of arrival delays. A histogram showed a sharp peak around zero, with a long tail extending into the positive range, indicating that

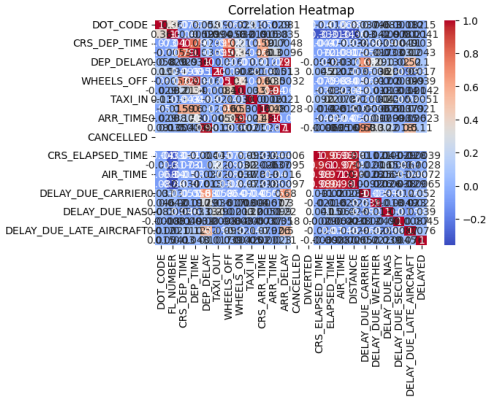

Fig. 1 Correlation Heatmap

while many flights arrive on time or with minimal delay, a significant number experience substantial delays. This skewed distribution is typical in aviation datasets and supports the need for binary classification (e.g., delayed vs. not delayed).

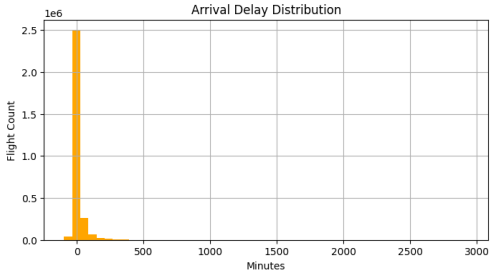

Fig. 2 Distribution of Arrival Delays

Another visualization focused on the average arrival delay across different airlines. This bar chart revealed notable differences between carriers, with some airlines exhibiting consistently higher average delays than others. These differences may reflect variations in route structures, hub congestion, or operational efficiency.

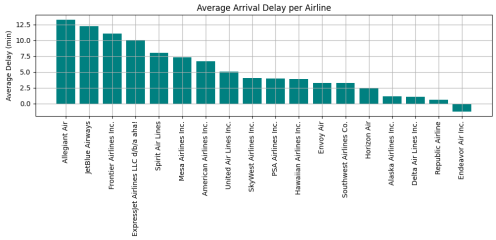

Fig. 3 Average Arrival Delay Across Aifferent Airlines

Through these visualizations, we gained valuable context for modeling decisions. For example, the correlation between departure and arrival delays supports the selection of departure delay as a key predictor. The airline-specific analysis also hints at the potential benefits of incorporating airline as a categorical feature in future modeling pipelines. Overall, the EDA guided both our feature selection and understanding of the problem space.

6 Modeling (Spark ML + H2O)

In this project, we combined Apache Spark and H2O to build and evaluate multiple machine learning models for flight delay prediction. Our goal was to test both traditional models and automated pipelines, compare their performance, and understand which ones offer the best balance between accuracy and efficiency.

We began by training several models using **Spark MLlib**, including Logistic Regression, Decision Tree, Random Forest, Gradient Boosted Trees (GBT), and Naive Bayes. Each model used two main features: departure delay and flight distance, which were chosen based on earlier EDA. We applied a VectorAssembler to merge these features into a single input vector, then split the dataset into 80% training and 20% testing. The **Logistic Regression** model served as our baseline, achieving solid performance with a good balance of speed and interpretability.

Next, we experimented with a **hybrid model** using **Principal Component Analysis (PCA)** to reduce dimensionality before applying a GBT classifier. This approach helped reduce noise and slightly improved performance, demonstrating the potential value of dimensionality reduction in this context.

To push further, we turned to **H2O AutoML**, a powerful framework that automatically tests and tunes multiple models, including deep learning and ensemble methods. We converted our Spark DataFrame into an H2OFrame and launched AutoML with a 15-minute runtime. It generated a leaderboard of top models, including stacked ensembles and deep neural networks. These models generally achieved high AUC scores and were competitive with our Spark-based models.

In addition, we trained a custom **H2O Deep Learning model** with two hidden layers and ten training epochs. This model performed well and showed the benefit of deeper architectures when more training time is available.

Overall, Gradient Boosted Trees and Hybrid PCA+GBT were top performers on the Spark side, while AutoML's leaderboard model slightly edged them out. However, the performance differences were small, highlighting that Spark ML models, when well-tuned, can match more complex alternatives. This modeling stage showed how combining Spark's scalability with H2O's automation allows us to explore a wide range of options efficiently.

7 Evaluation & Results

To evaluate our models, we used multiple performance metrics: **Accuracy**, **Precision**, **Recall**, **F1-score**, and **AUC (Area Under the ROC Curve)**. These metrics help us assess how well each model predicts whether a flight will be delayed by more than 15 minutes.

First, we generated predictions using each trained model on the test set. We calculated confusion matrices to manually compute all key metrics. For example, **Logistic Regression** achieved an accuracy of 93.7%, with a precision of 90.2%, recall of 72.2%, and an F1-score of 80.2%. **Random Forest** and **Gradient Boosted Trees** had very similar metrics, with GBT slightly outperforming others in AUC (0.9295). Our **Hybrid PCA+GBT** model also showed strong performance with an F1-score close to 80% and AUC of 0.9291, suggesting that dimensionality reduction did not hurt model quality.

Table 1 Evaluation Metrics for Flight Delay Classification Models

Model	Accuracy	Precision	Recall	F1 Score
Logistic Regression	0.9372	0.9023	0.7223	0.8023
Random Forest	0.9375	0.9113	0.7151	0.8014
Gradient Boosted Trees	0.9375	0.9128	0.7137	0.8011
Decision Tree	0.9375	0.9080	0.7182	0.8020
Hybrid PCA + GBT	0.9371	0.9287	0.6968	0.7962

We visualized ROC curves for all Spark-based models in a single plot. This comparison showed that all models performed well above chance level, and their curves were close, with only small differences in area under the curve. The ROC plot confirmed that both GBT and the hybrid model achieved the best balance between true positive and false positive rates.

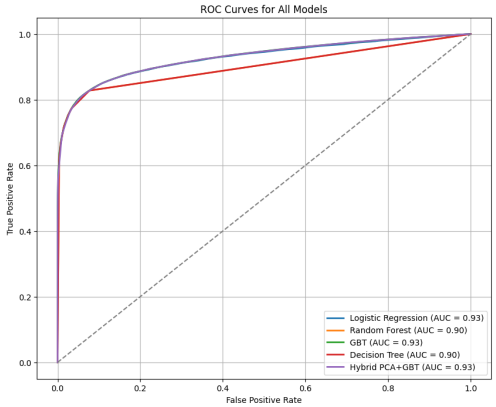

Fig. 4 ROC Curves

H2O AutoML's best model produced an AUC of 0.9284, very close to Spark's top performers. The **H2O Deep Learning** model followed closely with an AUC of 0.9267. These results suggest that Spark ML pipelines, when configured correctly, can compete with more complex automated modeling systems.

Table 2 Model Performance Comparison Based on AUC

Model	AUC Score
Gradient Boosted Trees	0.9295
Hybrid PCA + GBT	0.9291
H2O AutoML Leader	0.9285
Logistic Regression	0.9280
H2O Deep Learning	0.9268
Random Forest	0.9011
Decision Tree	0.7105

In summary, all models performed strongly, and the small variations in metrics suggest that the chosen features and Spark/H2O integration produced reliable and consistent predictions.

8 Discussion & Comparison with Literature

Our results confirm findings from previous studies, especially those that highlight the effectiveness of ensemble methods like Random Forest and Gradient Boosted Trees. Just like in earlier papers, we found that these models outperformed simpler algorithms such as logistic regression or decision trees in terms of both accuracy and AUC. The Gradient Boosted Trees model, in particular, achieved the highest AUC score in our Spark ML pipeline, aligning with recent literature that shows GBT's advantage in handling complex patterns in flight delay data.

Compared to deep learning models used in earlier works, our H2O Deep Learning model performed similarly to Spark models but did not offer a significant improvement. This supports claims in the literature that deep learning models require more features (such as weather or sequential time data) to truly outperform ensemble models.

Additionally, our use of dimensionality reduction through PCA combined with GBT provided comparable results, contributing to an under-explored area in the literature.

Overall, while our models did not rely on extensive real-time data or deep architectures, they provided strong, interpretable results that are more scalable and practical, especially in big data environments, highlighting the benefits of combining Spark with H2O AutoML.

9 Conclusion & Future Work

In this project, we built a complete and scalable machine learning pipeline for flight delay prediction using Apache Spark and H2O. We applied multiple classification models, including Logistic Regression, Random Forest, Decision Tree, Gradient Boosted Trees, and a PCA-based hybrid modelt to predict whether a flight would be delayed by more than 15 minutes. We also integrated H2O's AutoML and Deep Learning modules to compare performance with automated and deep learning approaches. Among all models, Gradient Boosted Trees and our hybrid PCA+GBT model performed

best, achieving the highest AUC scores while maintaining reasonable accuracy and F1-scores.

Our pipeline proved to be efficient and interpretable, making it practical for real-world deployment in airline operations or airport systems. It also highlights the advantage of using big data frameworks like Spark for handling large datasets and model training at scale.

For future work, we plan to enhance our model by adding more features such as weather conditions, flight route congestion, and historical performance of aircraft or airports. We also aim to explore real-time streaming data integration using Spark Streaming and expand the deployment layer using H2O MOJO or Flask APIs for production use. Incorporating explainable AI techniques like SHAP could also improve model transparency.

Code Availability

The complete source code for the project, including preprocessing, model training, and evaluation scripts, has been uploaded to GitHub. The repository includes a detailed README file.

GitHub Repository:
https://github.com/salmaemrany/flight-delay-prediction-spark-h2o

References

[1] Anonymous: A hybrid machine learning-based model for predicting flight delay. Scientific Reports (2024)

[2] Shetty, S.: Big-Data-Airline-Delay-Prediction. https://github.com/SaiprakashShetty/Big-Data-Airline-Delay-Predictio. GitHub repository (2020)

[3] Chaudhuri, T., Zhang, S., Zhang, Y.: Attention-based Deep Learning Model for Flight Delay Prediction (2024). https://www.sesarju.eu/sites/default/files/documents/sid/2024/papers/SIDs

[4] Flight delay prediction based on aviation big data. IJFANS International Journal of Food and Nutritional Sciences (2022)

[5] Flight delay prediction based on aviation big data. IRJMETS (2022)

[6] Bureau of Transportation Statistics: Airline On-Time Statistics and Delay Causes (n.d.). https://www.transtats.bts.gov/

[7] National Centers for Environmental Information (NOAA): Weather and Climate Data (n.d.). https://www.ncei.noaa.gov/

[8] Threnjen: 2019 Airline Delays w/Weather and Airport Detail. https://www.kaggle.com/datasets/threnjen/2019-airline-delays-and-cancellations (2019)