AeroForecast: Leveraging Machine Learning to Visualize and Predict Flight Delay

Kartik Narang, Amish Saini, Ninaad Lakshman, Nathan Wang, Aryan Musharaf, Jaegook Kim
Team 018

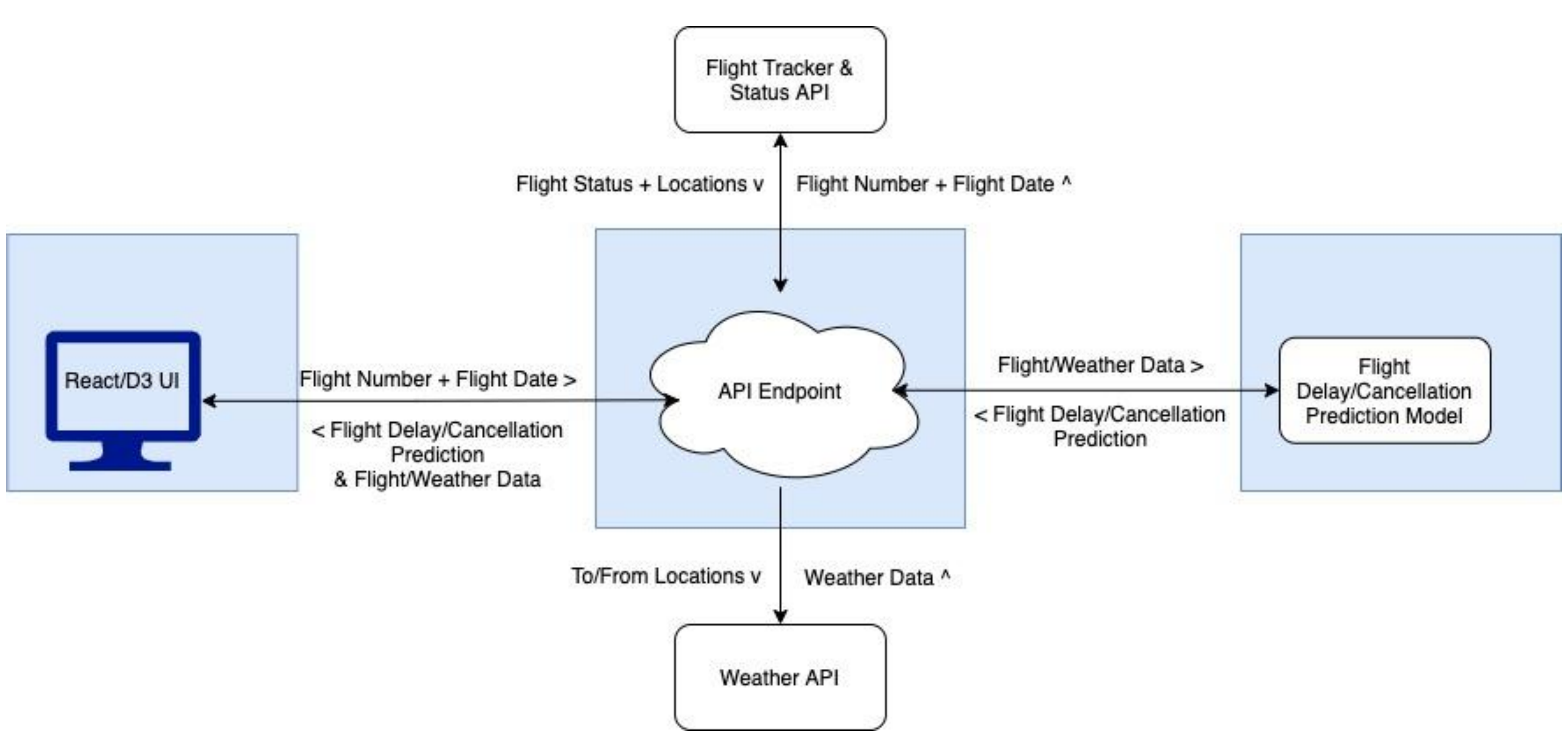
Motivation / Introduction

Flight Delays and cancellations are far from rare. An average of 30,000 delayed flights occur per day globally, and within the United States alone, that includes 7,000 to 9,000 flights per day. This impacts a significant amount of people's travel plans every day, and often these delays are last minute or unexpected so it's difficult for people to plan their personal or business-related circumstances around them. Therefore, there is value in predicting flight delay. A visualization tool to track and predict these delays clearly is a need in the status quo.

Approaches

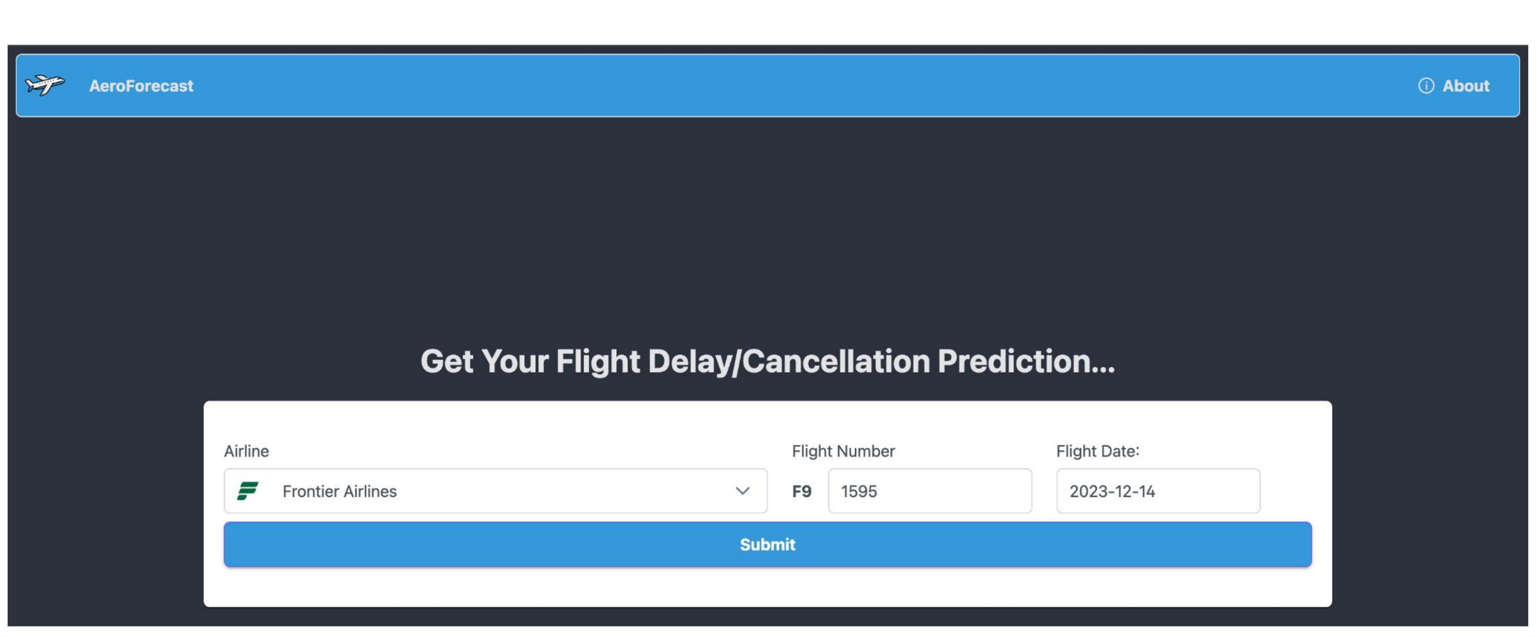
We have the following three approaches: Interactive Visuals using various APIs, Data Preprocessing and Feature Analysis, and Predictive Classification models. We will cover how they work, the intuition behind them, and their novelty in this section.

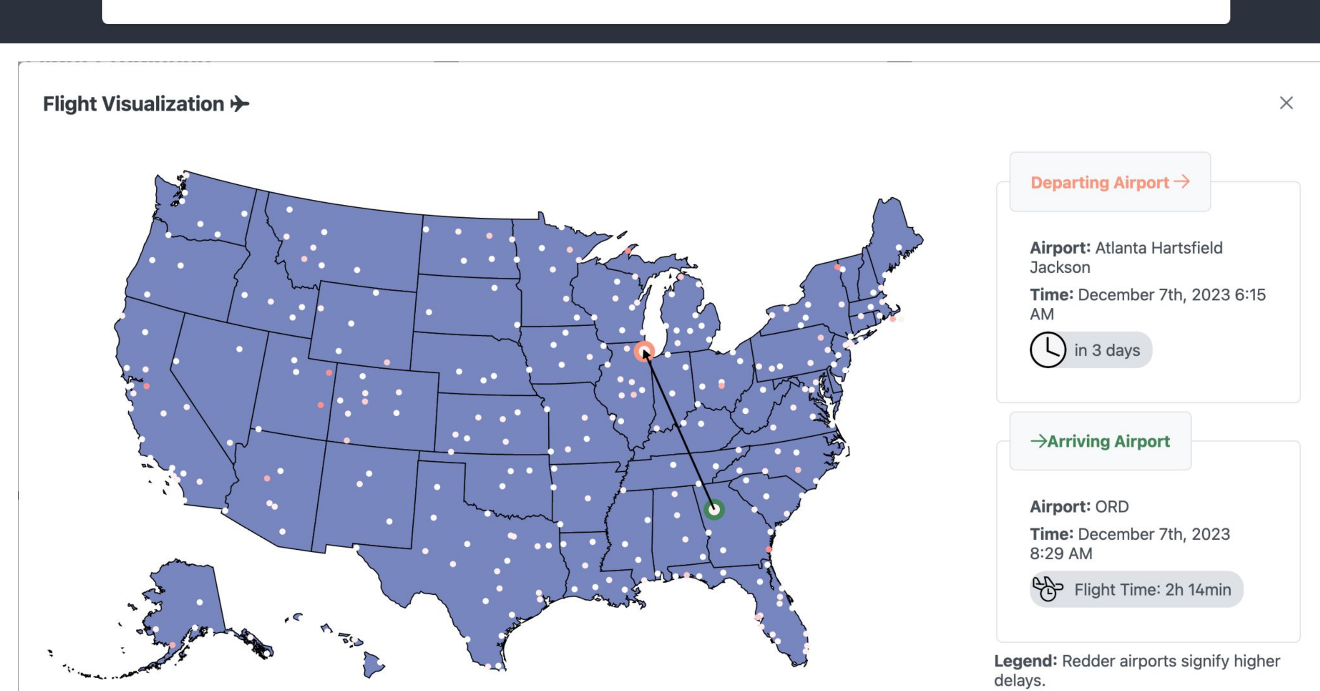
Let's first discuss how it works. The user can enter the airline, flight number, and the date of departure. We take this information and hit the Aerodata Box API to get information about the flight and append it to the weather data from CheckWX. Our model is then trained on this aggregate data and historical joined data from Department of Transportation and NOAA and returns information about confidence of flight delay and by how many hours. The details of this are in the below diagram:



Moving onto the intuition behind the approaches, the combination of weather data with real-time flight data is a combination that we're hopeful will increase the accuracy and effectiveness of flight predictions. As discussed further in Experiments and Results, the UI has been tested through user interviews to ensure that it's friendly and easy to use. The Exploratory Data Analysis also ensures that we drop features that are redundant so our training is more effective. Lastly, we try a bunch of different classification models to see which one fits our data the best, and we use a two model approach as described further in Experiments and Results.

Lastly, with the novelty of this approach, we determined that previous approaches suffered from shortage of features as well as relying purely on historical data without any spatio-temporal data. We append multiple data sources together and use real-time data in efforts to alleviate these limitations.





Data

<u>41 features</u>
9.7 MB on Disk
20,326 rows
January 2022 - August 2023
Sampled 1000 random flights from each month

<u>5 features</u>
Average Wind Speed
Snowfall
Precipitation
Average Temperature
Direction of fastest 2-minute wind

<u>9 features</u>
Airline
Aircraft
Flight Number
Departure Airport
Arrival Airport
Status
Duration
Flight Percent Complete
Last Updated

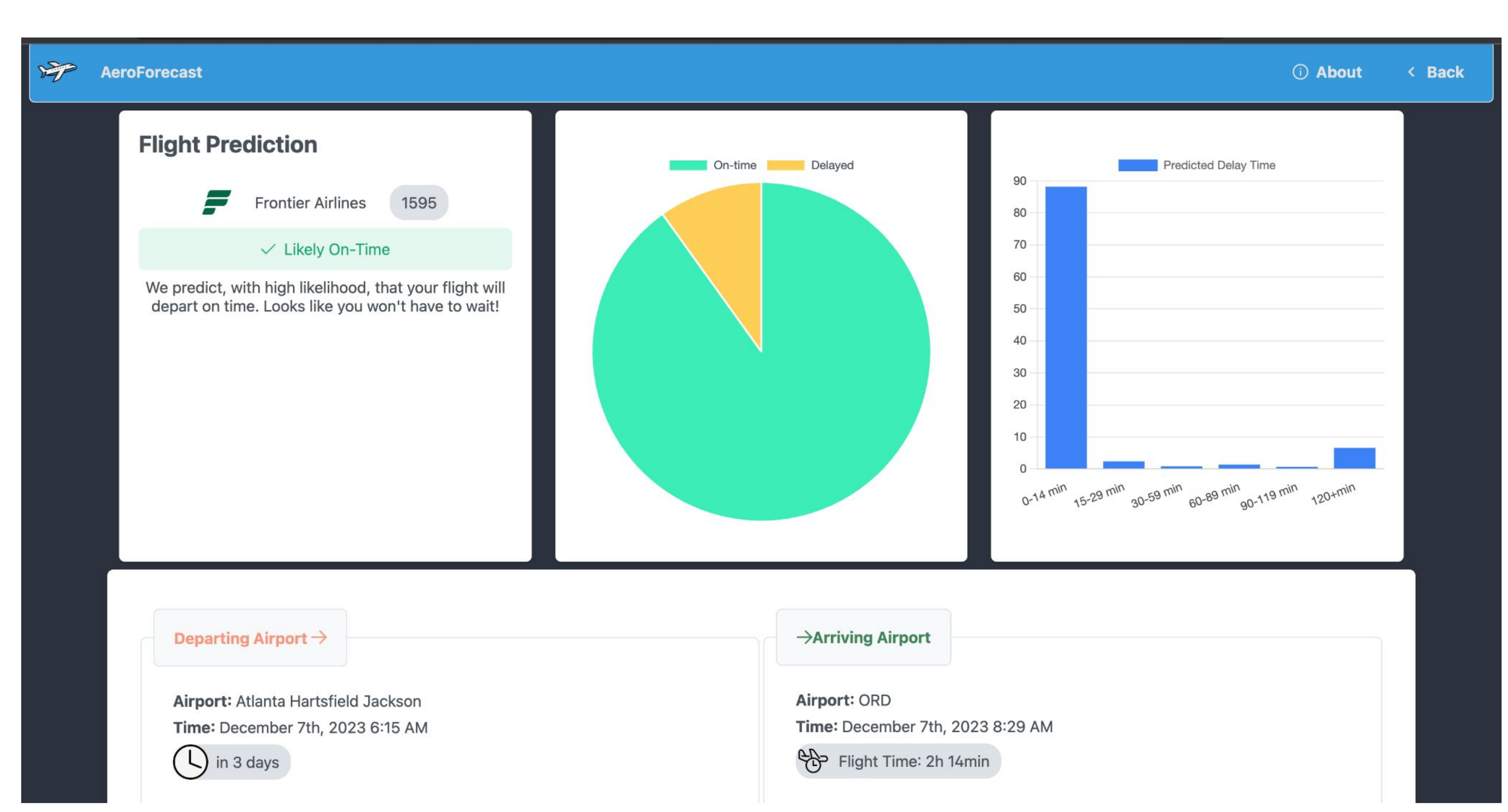
Experiments and Results

In order to judge the effectiveness of our system, we trained and evaluated on data from the past two years. The following are the results of our predictive model on the collected data. We have two predictive models: the first one gives a binary value (0 or 1) if there's a delay or not given the current conditions. If there's a delay, the second model will tell us how much the delay in buckets. We evaluated the effectiveness of the model with the following metrics:

First Model
Accuracy: 0.7852
Precision: 0.7655
Recall: 0.1118
AUC of ROC: 0.73

Second Model
Accuracy: 0.7872
Precision: 0.8236
Recall: 0.7872
AUC of ROC (per bucket):
(0.71, 0.67, 0.67, 0.69, 0.64, 0.67)

For the first two approaches, we did a sample of User Interviews (15 interviews) with the current UI with A/B testing to determine how the layout should be. We finalized on the following UI flow with the backend APIs powering it.



In terms of comparing to other existing methods, we were able to get close to state-of-the-art average flight prediction accuracy, which hovers around 82% across the 18 papers we read. Most methods tried novel architectures, ranging from Graph-CNNs to VAEs, whereas others tried novel feature combinations. We gathered inspiration from the model architectures and the feature sets to drive our own novel set of features and predictive learning methods, and our visualization sets us apart by focusing on User Experience.