PREDICTING FLIGHT DELAYS CAUSED BY WEATHER

Olivia Guo
MehtA+

Shreyas Singh
MehtA+

July 29, 2022

ABSTRACT

This project is an attempt to predict the amount by which a given flight is delayed due to weather-related causes. We used a dataset on flight delays and their causes, and another about weather conditions in the US, and determined, for each plane, if its takeoff and/or landing time is during a severe weather event. If so, we used the time between the end of the event and the departure/arrival, as well as the type of weather event at each airport as features to try to predict the actual delay time due to weather. We tested two different models, KNN and SVM, to see which gave the better accuracy. Neither model was very successful due to a large number of confounding variables.

Link to our code:
https://gist.github.com/shreyas-s125/f42d9c435786aaa472e73b81c7d12eb8

1 Introduction

Delays are a very common occurrence in flights all over the world. Due to the complexity and number of variables in flight, delays can happen for a myriad of reasons and last several hours or more, sometimes warranting an outright cancellation. However, weather events are the primary cause of flight delays according to the Federal Aviation Administration. Specifically, from 2008 to 2013, weather caused "69 percent of system impacting delays of greater than 15 minutes...The portion of delay due to weather represented nearly 10 million minutes in 2013" (FAA). Flight delays have measurable consequences: for example, "the cost to the air carrier operators for an hour of delay ranges from about $1,400 to $4,500, depending on the class of aircraft and if the delay is on the ground or in the air" (FAA). We attempt to apply machine learning methods to predict the amount by which a flight will be delayed beforehand, using information about the flight and data from weather forecasts.

2 Methodology

2.1 Dataset

We implemented two datasets: the first described weather events around US airports in 49 states, and the second described flight delays for US flights. The weather dataset included columns about the type of weather condition, its severity, the airport the weather event was observed from, the date it occurred, the time it started and ended (UTC), and more. The flight delays dataset included the airport, the flight date, the scheduled departure time (denoted CRS_DEP_TIME), the time the airplane actually took off, the departure and arrival delay time, how minutes the flight was delayed by each cause of delay (weather, carrier, security, National Air System (NAS), and late aircraft), and other columns.

Preprocessing The aforementioned datasets were first converted into Pandas dataframes for further analysis. Several columns in the original datasets were not relevant for our purposes, so only the following set of columns was retained:

Weather Data: [Severity, StartTime(UTC), EndTime(UTC), TimeZone, AirportCode, LocationLat, LocationLng, Type]

Delay Data: [FL_DATE, ORIGIN, DEST, CRS_DEP_TIME, DEP_TIME, DEP_DELAY, CRS_ARR_TIME, ARR_TIME, ARR_DELAY, DISTANCE, CARRIER_DELAY, WEATHER_DELAY, NAS_DELAY, SECURITY_DELAY, LATE_AIRCRAFT_DELAY]

Further, in the delay data, we added a column labeled "non_weather_ddelay" defined as the sum of the entries in the columns [CARRIER_DELAY, SECURITY_DELAY, LATE_AIRCRAFT_DELAY] for later convenience.

Each individual dataframe required preprocessing beyond simply removing columns. First, due to the large collection of data present, the rows in both dataframes were filtered as follows: any row between April 2017 and October 2017 (including endpoints) would be kept, unless it was on a day numbered greater than 27. The motivation behind this decision was the need for time zone adjustments later on in preprocessing. In preparation, it was optimal to ensure that all data points followed daylight savings time, and that no date was near the end of a month (so that time zone adjustments would not cause an extension into the next month). The data was restricted by applying Pandas filtering functions on the dataframes.

Moreover, the delays dataframe was cleared of diverted flights and any rows with NaN labels in the columns [DEP_TIME, ARR_TIME, DEP_DELAY] to filter out any cancelled flights. The remaining NaN values were replaced with 0.

The weather dataframe was limited to only those with a severity labeled "Severe" to ensure a higher chance of weather delay occurring.

Combining Both datasets were converted to NumPy arrays for further processing. We replaced each entry in the delay data column [CRS_DEP_TIME] to an "adjusted departure time" (henceforth called ADT) found by adding the "non_weather_ddelay" column values to the original [CRS_DEP_TIME]. This was done to take care of all non-weather departure-related delays first, before implementing any sort of method for predicting further weather delay. Similarly, the [CRS_ARR_TIME] column was replaced by its sum with [DEP_DELAY] to obtain an "adjusted arrival time" (henceforth called AAT). Note that in both cases, a simple summation did not suffice, as the first columns in each case denoted time in hhmm format, while the second columns used only minutes as a unit.

Due to the nature of our method for obtaining ADT and AAT values, some of them turned out to be negative. It is certainly possible to restructure our code in order to replace these negative values with the true hhmm time (as well as replacing the date with the one before), but due to time constraints and simplicity, we simply filtered out any negative AAT values. Addressing these negative cases was necessary as not to interfere with later processing algorithms.

Next, we defined the following three functions to correct formatting differences between the datasets and to obtain useful information for features:

- tzoneadder(a, b): Because the weather data was all presented in UTC, but each delay data point was presented in the local time zone of the airport in question, we needed to resolve this disparaty by defining a function to convert a string of the form "yyyy-mm-dd hh:mm:ss" into another string with the same formatting, but shifted to correct amount from local time to UTC. The parameter a is the original date string, and the parameter b is the correction amount in hours.
- format(a, b): Because the delay data was presented with "yyyy-mm-dd" and "hhmm" in different columns, and that the "hhmm" data did not have leading zeros, we defined this function that takes the date string and time values as inputs and outputs a single string of the form "yyyy-mm-dd hh:mm:ss"
- timer(a, b): This function takes in two strings of the form "yyyy-mm-dd hh:mm:ss" and outputs the total time, in minutes, between b and a, with b considered later.

One formatting was taken care of, we partitioned the weather data into a dictionary, with each key being the airport code at which the weather event was recorded. Then, we extracted our features in the following way:

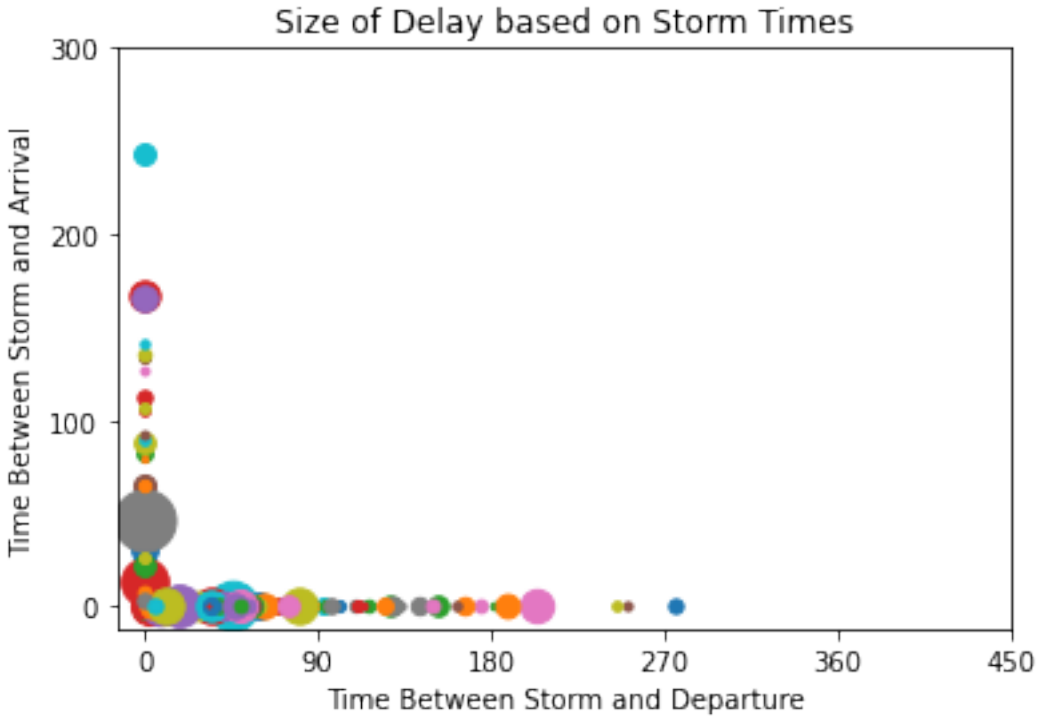
1. Extend the delay NumPy array to have eight more columns, all initially filled with 0.
2. For each delay data point, we define a "true" departure time, by taking the raw ADT and running it through tzoneadder and format, sourcing the second parameter in tzoneadder using the [TimeZone] column in the weather data.
3. We loop through each element in the weather dictionary value that corresponds to the airport code given in the delay data point and check if the time period of the weather event coincides with the ADT. If so, we assign

the first of the eight extra columns to the following value: timer("true" departure time, end of weather event). Additionally, we fill the third fourth and fifth columns with a one-hot vector of length 3 corresponding to the type of weather event. The reasoning for the implementation of the weather dictionary drastically reduced runtime by reducing the number of weather events necessary to loop through.

4. We perform a similar looping, this time for the arrival data using a "true" arrival time derived from the AAT.

5. We finally filter out any delay data points with all zeros for the last 8 columns, and then extract the last 9 columns for each remaining data point, resulting in a set of points with one label (weather delay time) and 8 feature elements:

- Feature 1: Time interval between ADT and the time at which the departure weather event stops (zero if there is no departure weather event).
- Feature 2: Time interval between AAT and the time at which the arrival weather event stops (zero if there is no arrival event).
- Features 3-5 One-hot vector for departure weather event type (3-fog, 4-storm, 5-cold).
- Features 6-8 One-hot vector for arrival weather event type (3-fog, 4-storm, 5-cold).



In the above plot, the x axis is feature 1, the y axis is feature 2, and the size of each dot represents the delay in minutes of that particular flight due to weather.

2.2 Models

We tested two different models on our data: K-Nearest Neighbors (KNN) and Support Vector Machines (SVM) with a linear kernel. Further, for each feature, we used two versions of the data: the first was simply the entire collection, and the second was only the data points that had a nonzero delay. Also, we tested these models under two different sets of features: one with only the first two features (the time intervals), and one with all eight features. Additionally, for KNN, we tested different values of K, ranging from 1 to 20. We split each data collection for each feature set into training and validation in a 80:20 ratio, and all models were imported from SciKit Learn.

3 Results

Features	Model	Accuracy (all delays)	Accuracy (only positive delays)
1-2	Linear SVM	84.34%	0.0%
1-2	KNN, K=13	84.34%	5.41%
1-8	Linear SVM	89.68%	0.0%
1-8	KNN, K>3	89.68%	2.70%

4 Conclusion and Future Work

Note that the accuracies we obtained varied between different random states used in the train-test split, for example, one random state had a lower accuracy of around 85 percent for the full-feature full-data trial. 184 out of the total 1402 data points actually had a nonzero delay, so the percent of points with no weather delay equals 86.88 percent. Due to the very large proportion of zero-delay flights, the SVM and KNN models are likely predicting values close to 0 most, if not all, of the time. When these zero points are removed, the accuracies expectedly drop dramatically, so our model is not very successful in a larger sense. As shown in the plots, the data in question is likely not linearly separable to any appreciable degree, and there is a large degree of randomness due to factors left unaccounted for.

We also found that there were thousands of data points that did not correspond to any departure or arrival weather event, but still had a nonzero weather delay. This could be due to factors such as weather delay en route between the two airports, which we did not consider. Although the storms we used were all classified as severe, storms don't maintain the same severity for their entire duration. It's possible that the storm was classified as severe, but during takeoff, the storm was not that severe, and departing was not dangerous. Also, the dataset was unclear about how it determined what each cause of delay contributed to the total delay time. If, for example, the plane was delayed due to security, but during that time, there was also extreme weather, we didn't know if that would add time to both security delay and weather delay, just security delay, or just weather delay. If the dataset used a different scheme compared to our interpretation while defining ADT, our results would be negatively impacted.

Simply put, there are an extremely large number of variables pertaining to any given flight. If we had more time, we could have tested many more factors that influence the decision of whether or not to delay a flight, such as the departure and arrival airports, pilot competency and confidence, runway ILS category, plane size and model, statistical trends for high-altitude weather in between airports, and many more. However, considering the large number of difficulties from linking together even two separate datasets due to problems such as formatting difference, and due to the fact that not all datasets include the same flights, it would certainly be much more difficult. We could also try more models, like testing out a neural network. Also, another factor that could possibly ad to airport traffic is if there are too many planes that need to use the same runway. If we had more time, we would expand our problem to also look at plane models, their takeoff and landing distances, and the runways in each airport to try to predict more aspects of delay time other than just weather. Finally, we could take into account instances where flights were cancelled or redirected due to weather, rather than excluding those cases.

5 Division of Labor

We divided the work for each component of this project individually, so each member worked at least partially on each aspect.

6 Acknowledgements

We would like to acknowledge the Mehta+ Instructions Haripriya Mehta, Bhagirath Mehta, Andrea Jaba and Anna Muyan Li, as well as Prof. Awad Khireldin from the Singapore Institute of Technology.

References

https://aspm.faa.gov/aspmhelp/index/Types_of_Delay.html
https://www.faa.gov/nextgen/programs/weather/faq#faq6
https://www.kaggle.com/datasets/yuanyuwendymu/airline-delay-and-cancellation-data-2009-2018 (Delay dataset)
https://www.kaggle.com/datasets/sobhanmoosavi/us-weather-events (Weather dataset)

Moosavi, Sobhan, Mohammad Hossein Samavatian, Arnab Nandi, Srinivasan Parthasarathy, and Rajiv Ramnath. "Short and Long-term Pattern Discovery Over Large-Scale Geo-Spatiotemporal Data." In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery Data Mining, ACM, 2019.