Examining Factors in Flight Delay

By

Anoop Manjal

December 2023

Affiliation: Brown University

GitHub: https://github.com/amanjal/Data1030-Final

Introduction

When traveling, flight delays can be a large source of headaches for many people. Flight delays can be devastating for an airline, as it can ruin a carefully planned schedule, affect profitability, and harm an airline's reputation. For passengers, flight delays often cause passengers to miss carefully planned connections, and can even ruin travel plans altogether. In 2022, 24.3% of flights in the U.S. were either delayed or canceled [1]. This project aims to develop a model that can accurately predict flight delays well before the day of the flight. Accurate estimations can help airlines prepare more efficient travel schedules, and help passengers make more accurate travel plans and pick more timely flights.

For this project, 2015 flight data provided by the Department of Transportation was utilized [2]. To limit the scope of the project flights from Boston Logan International Airport were the only ones examined. In addition, the data was limited to what would be available at the time of prediction, further reducing the size of the data set. As a result, the dataset contained roughly 100,000 data points and 8 features.

It is important to note the goal of this project is to predict the arrival delay of a flight departing from Boston. While this model should be transferable to any airport, it was designed specifically for Boston Logan International Airport. In terms of prior work, extensive amounts of exploratory data analysis has been done, but no predictive model for arrival delay has been created.

EDA

In the exploratory data analysis, I wanted to focus on three main factors: month, destination airport, and departure time. I believe these three factors would be the most important when estimating flight delay.

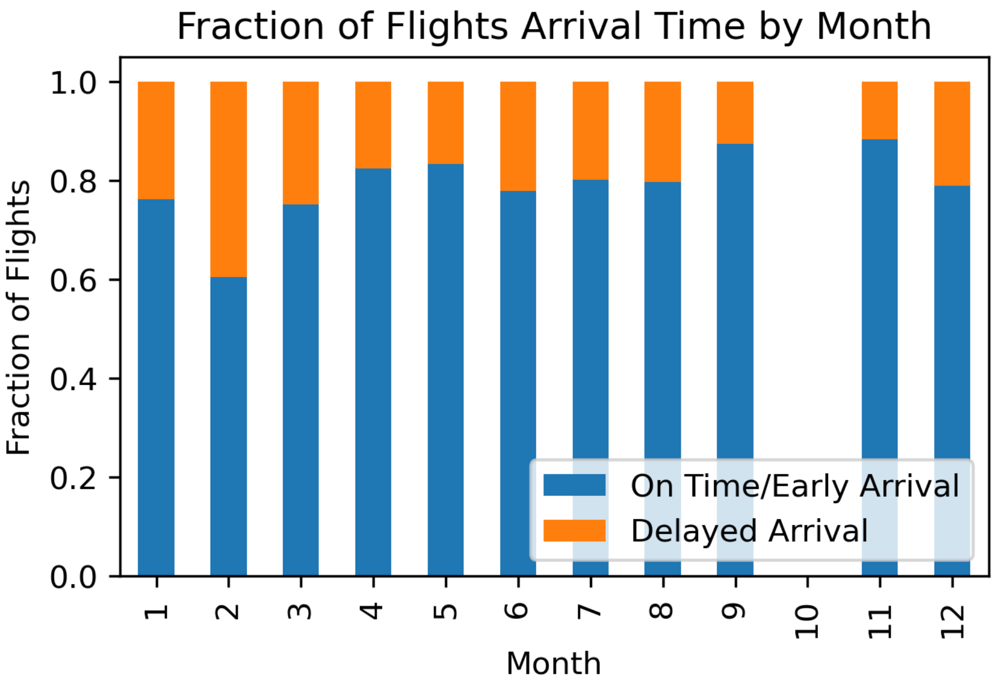

Figure 1: Fraction of flights departing from Boston either on time or early vs delayed at destination airport.

Figure 1 illustrates that flights tend to be more delayed in the winter months (December, January, February) which is seemingly correlated to the increased severity of weather often seen in those months. February exhibits a higher proportion of delays with over 40% of flights delayed. September exhibits the lowest fraction of delays with less than 20% of flights delayed. Interestingly, there also is a slight increase in flight delays in the summer months, which may be attributed to the large increase in overall travel seen during those months.

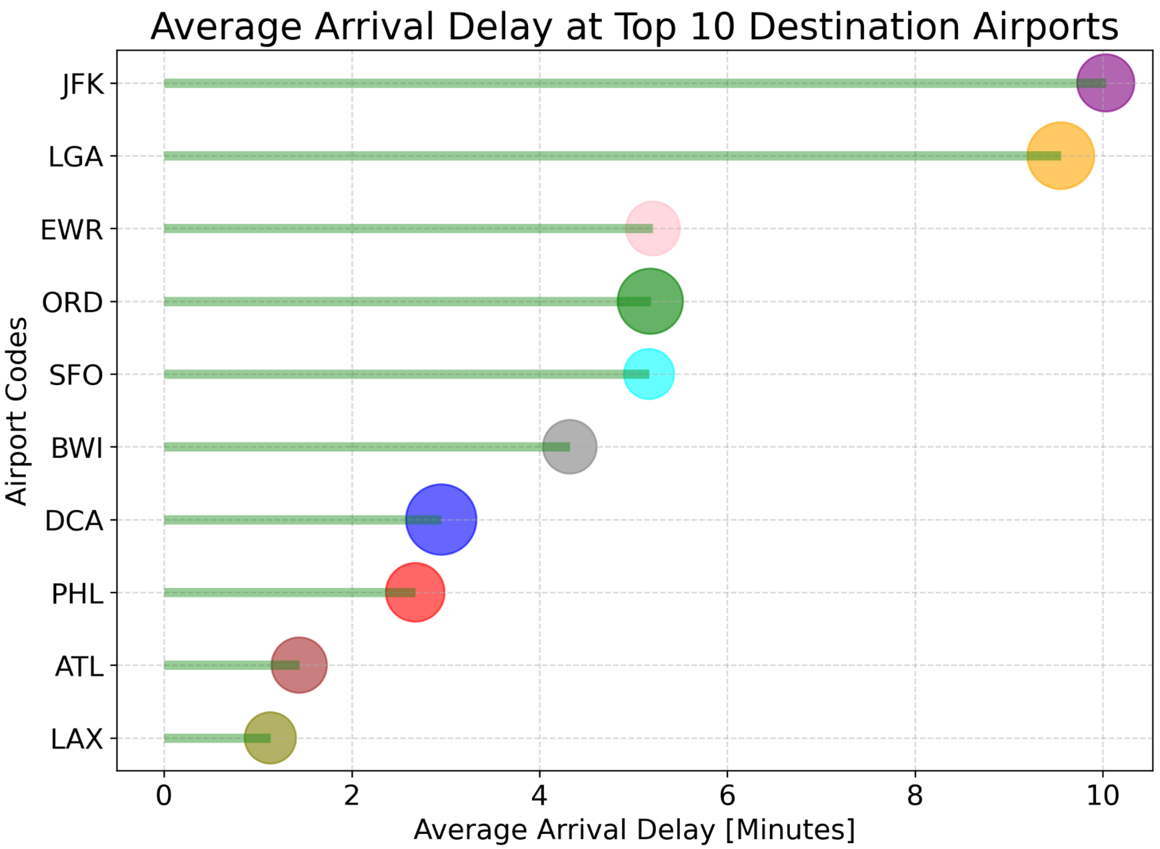

Figure 2: Average arrival delay in minutes for flights departing from Boston and arriving at one of the top 10 most frequent arrival destinations.

All of the top 10 destination airports in Figure 2 have an average delay of more than 0 minutes. This is particularly surprising since Figure 1 shows that a majority of flights are either early or on time. In addition, JFK and LGA airport, both located in New York City, have an average delay of over 9 minutes, which is much higher than the other airports on the list. Also, only two airports in the top 10 destinations (LAX, SFO) are Western airports, indicating that flights from Boston heavily fly to the east coast.

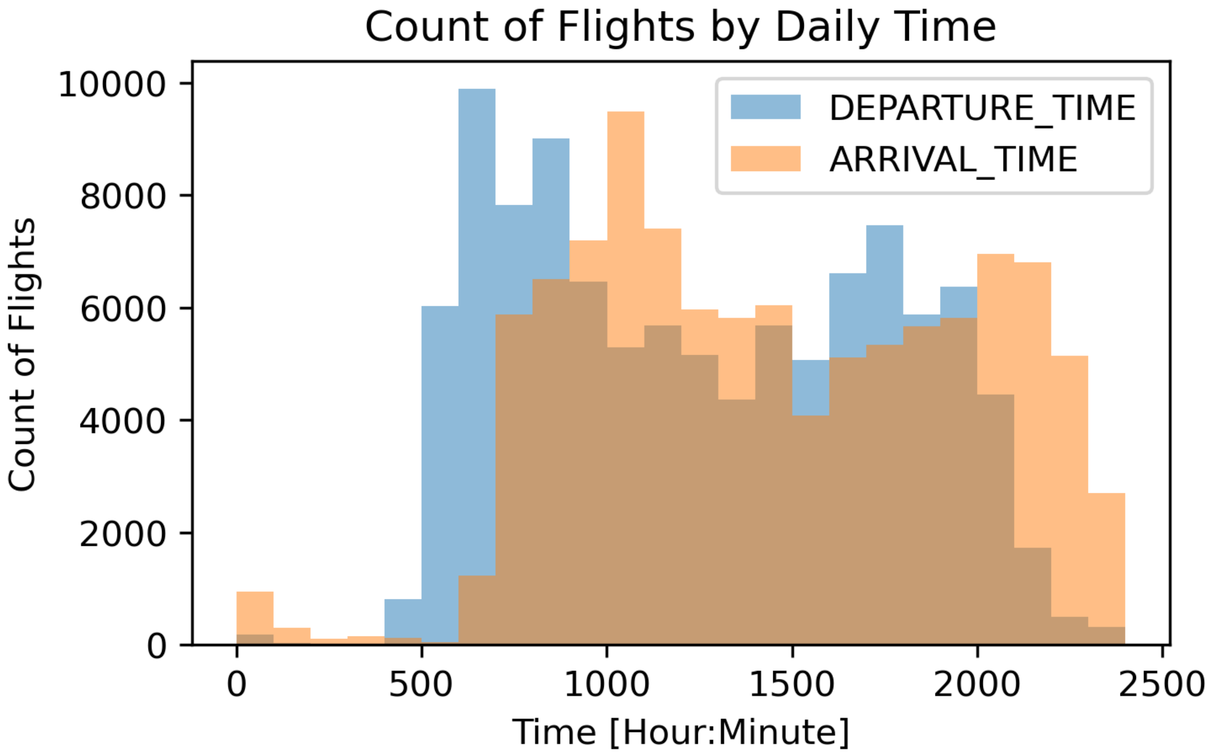

Figure 3: Departure time from Boston Airport & arrival time at destination airport

Figure 3 illustrates a wave pattern in flights departing from Boston. A large portion of flights leave in the 05:00 - 09:00 portion of the day. There is a second wave of departures in the 15:00 - 20:00 portion of the day. There is a minimal amount of departure out of Boston after 21:00, indicating aircrafts most likely arrive at Boston and wait for departure until the following morning. This would most likely minimize the amount of delays that propagate through multiple days. In addition, there is a steady amount of departures throughout the day, indicating Boston is a busier airport in terms of number of flights.

Methods

The airline dataset used in this problem is time series data, making a time series splitting strategy necessary. I used a nested cross-validation strategy, where the test set was 5% increments, and the training set was all the data preceding the test set. I used the back 50% of the dataset to form 10 test sets. As a result, for each test set, the training set is a different size, with a minimum training set size of 50%. For example, for test 1, the training set would be the 0%-50% range of the dataset, and the test set would be the 50%-55% range of the data set. For test 10, the training set would be the 0% - 95% range of the dataset, and the test set would be the 95% - 100% range of the dataset. I used this strategy to measure the uncertainty of the models when testing on unseen data points. In the cross-validation pipeline, time series splitting was utilized.

After the initial brief pre-processing mentioned in the introduction, the dataset contained 11 features. Since the arrival delay is the target variable, it was stored and removed from the dataset leaving 10 features that needed to be preprocessed. Only two categories were categorical data, Destination Airport and Airline. Since there is no implicit order to these categories I used one hot encoding. In addition, I used a standard scaler for features that had no implicit minimum or maximum. This included the distance and flight number features. Finally, since all the other features were time-based, I used a minmax scaler, since there is a known minimum and maximum.

One Hot Encoded	**MinMax Scaler**	**Standard Scaler**
Destination Airport, Airline	Scheduled Departure, Scheduled Arrival, Scheduled Time, Month, Day, Day of Week	Distance, Flight Number

Table 1: Table listing out the features and how they were preprocessed. Before preprocessing there were 10 features. After preprocessing there were about 81 features.

Model	Hyperparameters Tuned
Lasso Regression	Alpha: 21 evenly spaced values in (.001, 100)
Ridge Regression:	Alpha: 21 evenly spaced values in (.001, 100)
Elastic Net	Alpha: 21 evenly spaced values in (.001, 100) L1 Ratio: [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1]
Random Forest (RF)	Max Depth: [1, 2, 3, 4, 5, 6, 7, 8, 9] Max Features: [0.5, 0.75, 1.0]
XGB Regressor	N Estimators: [10, 100, 1000] Max Depth: [5, 6]

Table 2: Table of models and hyperparameters tuned in this project.

I decided to use mean squared error (MSE) as my evaluation metric. Typically, delays become more problematic the longer they stretch. To model this MSE works best, as it further punishes the model if it is very far off the correct answer. To achieve the best results I used 5 different machine learning algorithms with several different parameters tuned. For Lasso and Ridge Regression, I wanted to test a wide range of values for alpha. The limits of 0.001 and 100 were set to prevent underfitting and overfitting. For Elastic Net, I used the same methodology for the alpha values, but a smaller range was used for the l1 ratio. The RF model used max feature values linearly spaced from 0.5 to 1, and max depth values from 1 to 9. I used 9 as the max, since max depth should typically be less than the number of features in the data. I also implemented a XGB Regression model, though the parameters tuned were limited due to computational restraints.

Results

Based on the mean MSE score of the different models, Random Forest Regression had the smallest MSE of 1499.63. That being said, the results shown in Table 3 indicate that none of the models are significantly different from the others or the baseline. Figure 4 illustrates a box plot of the RMSE scores of the different models. Ridge Regression has the lowest median RMSE score out of the models and the baseline, though again not significantly different from the baseline.

Model	Mean MSE (Minutes Squared)	Standard Deviation MSE (Minutes Squared)
Random Forest Regression	1499.63	850.84
Lasso Regression	1523.70	836.17
Ridge Regression	1501.20	951.60
Elastic Net	1523.98	842.52
XGB Regression	1521.42	909.18
Baseline	1480.03	837.31

Table 3: Mean and standard deviation of MSE scores of the 10 test runs on each model.

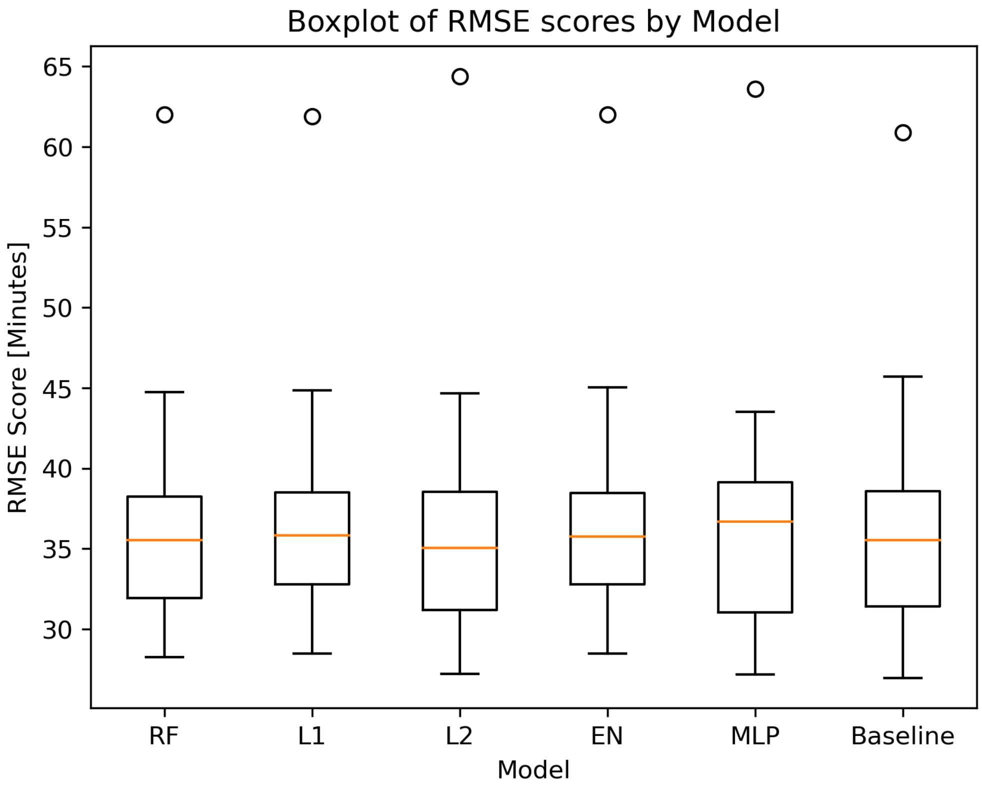

Figure 4: Boxplot of the RMSE scores from the 10 trial runs of each model.

Since the Ridge Regression (alpha = 100) model had the second lowest MSE and the lowest median RMSE, as well as far less computational strain than the RF model, I decided to use this model for further analysis. Figure 5 calculates the predicted results using the aforementioned model, and displays them against the true results. The model is more accurate with early arrivals, but performs poorly when estimating flights that are delayed. Overall, very few points are accurately predicted resulting in the high MSE score.

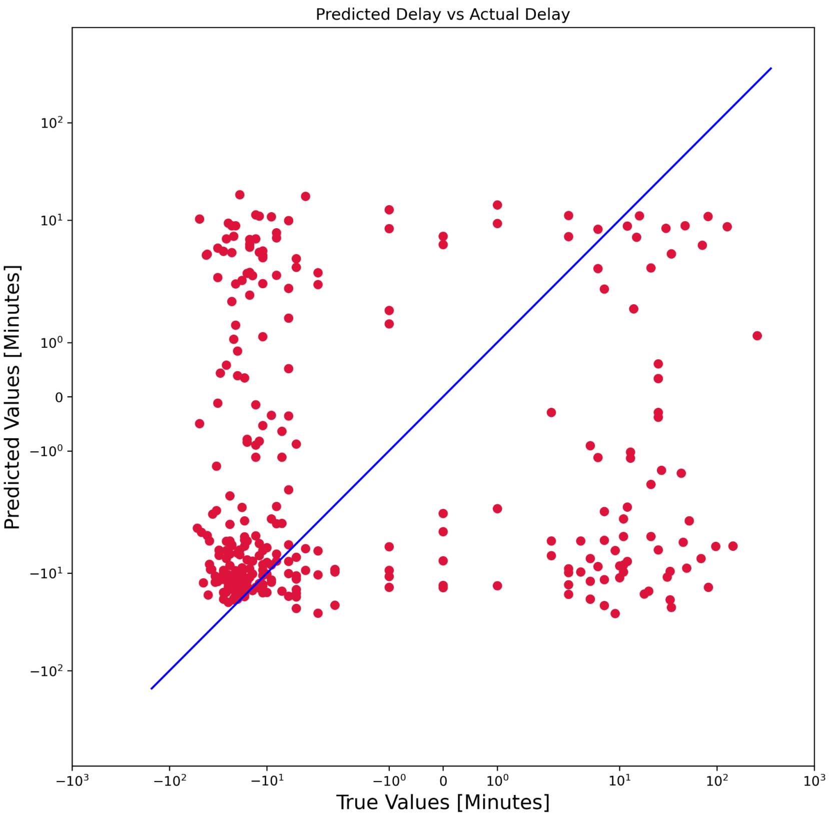

Figure 5: Predicted Values vs True Values in minutes. Accurate predictions would be closer to the blue line.

To study the global feature importance, SHAP values were calculated in Figure 6. The month feature is the most important feature, followed by scheduled departure and scheduled time'. Interestingly, no airport was included in the top 10 most important features, which somewhat contradicts the earlier findings in Figure 2. Surprisingly, scheduled time is an important feature, possibly suggesting that there is some relationship between length of the flight and length of the delay. In addition, all three time features (month, day of week, day) are graphed indicating that the day of the flight is an important factor when calculating delay.

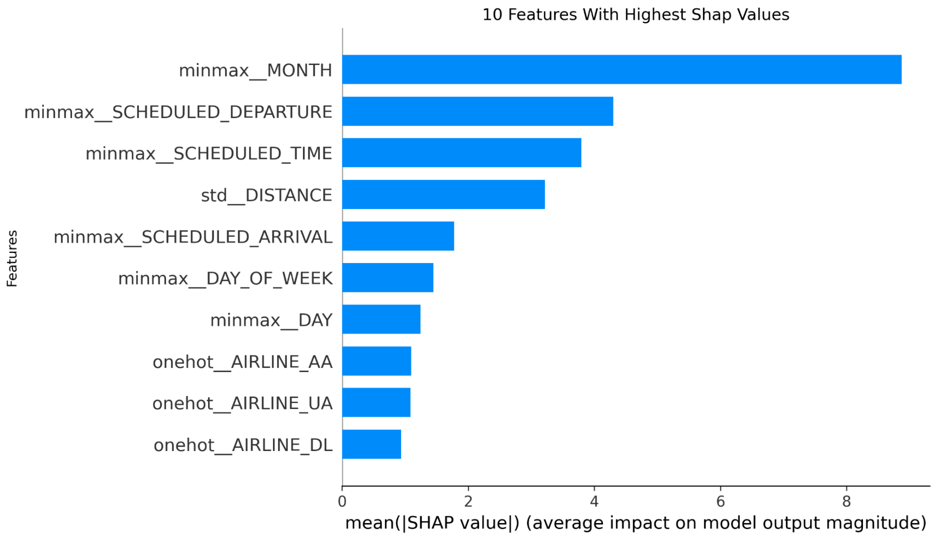

Figure 6: Global SHAP values of the best model (Ridge Regression, alpha=100)

To examine feature importance in more ways, the best XGB Regression model (max depth of 5, 10 estimators) was used to calculate feature importance and permutation importance. These are shown in Figure 7 and Figure 8. In both, time features such as scheduled departure, month, and day maintain their importance. Though, two airports also make the top 10 in feature importance, Detroit Metro Airport and JFK Airport.

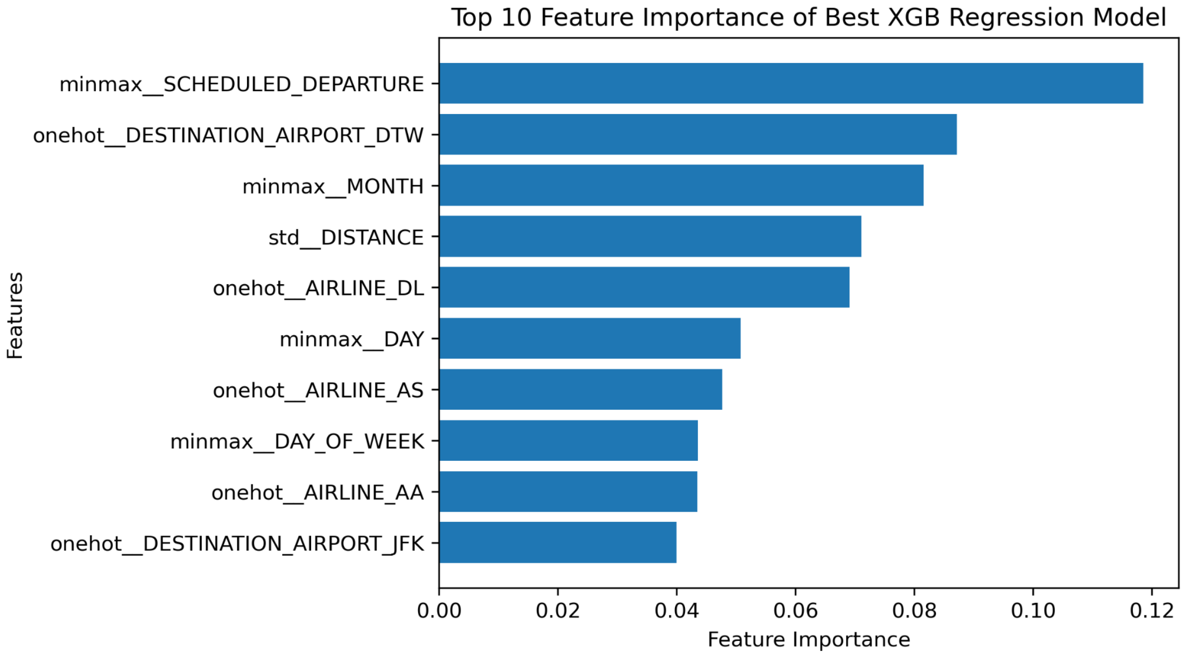

Figure 7: Feature Importances of XGB Regression Model

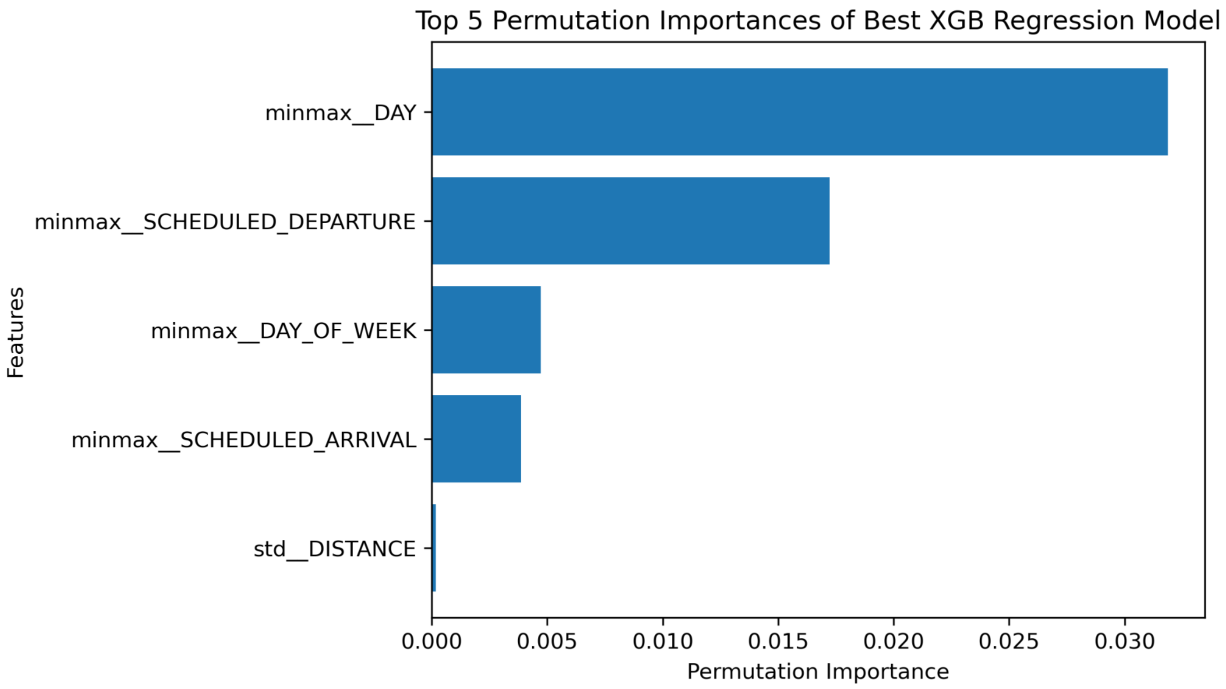

Figure 8: Permutation Importances of XGB Regression Model

In addition, local feature importance was also studied, with SHAP local feature values calculated in Figure 9 and Figure 10. In Figure 7, the month and destination airport play a big role in driving the arrival delay value lower. This would indicate that in general, the month of January and a destination airport of Atlanta tends to lead to smaller delays. In Figure 8, we again see the month drive the arrival delay down, but the scheduled departure time and the destination airport are more important features here, pushing the arrival delay up to 7.8 minutes. In this case, both flying to Fort Lauderdale Airport and flying towards the end played a big role in causing longer delays.

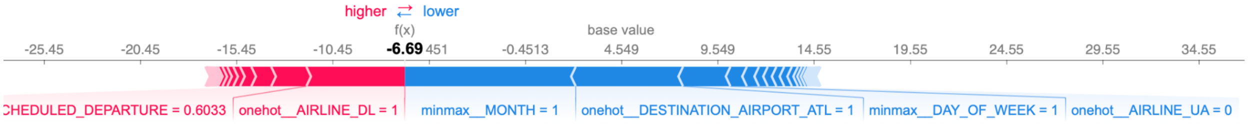

Figure 9: SHAP local feature importance for observation 100.

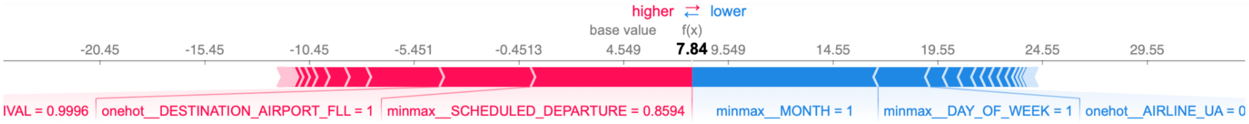

Figure 10: SHAP local feature importance for observation 205.

Future Outlook

The model would most likely be best improved with more data. Ideally, the model would learn over several years of data so it could determine the general dates delays tend to occur. Currently, when learning over one year, the model sees each data point exactly once, and is not able to generate any sort of useful pattern. In addition, appending weather data to the current data set could also prove fruitful. Weather data would provide more nuance for the conditions of the flight and perhaps allow the model to predict more accurately. In the future, an implementation of a multilayer perceptron model (MLP) could be useful in improving performance. In addition, the acquisition of more computing power could allow for further testing of the XGB Regression model to find better hyperparameters. Overall, while the current results are not very useful, they are a promising start as more data gets collected.

References

[1] Bureau of Transportation Statistics 2022 Flight Delay & Cancellation
[2] https://www.kaggle.com/datasets/usdot/flight-delays