PREDICTING FLIGHT DELAYS

DAVID YU (YY743), ELAINE YU (SY372), ANNIE CHENG (ZC375)

ABSTRACT

As flying has become a more popular and commonplace means of transportation, the nature of the air travel experience began to change. People now face new problems of flight delays, leading to both mental and financial costs. This project combines two datasets on airline performance and weather events and builds several models to predict whether a flight will likely delay. The final model of random forest has reached a weighted F1-score of 0.67 with a discussion of model limitation and fairness in the end.

1. DATA ANALYSIS

1.1 BACKGROUND (THE PROBLEM)

We have all experienced pain and anxiety when our flights were delayed for an unknown duration, and we have to wait. In fact, flight delays have caused severe troubles for the economy. According to Rebollo's research, in 2007 alone, the U.S. economy had endured 31–40 billion dollars lost due to flight delays [1]. **If we can predict which flight will have delays, consumers, airlines, and air traffic controllers can all plan ahead and make adjustments accordingly. Although we cannot avoid flight delays entirely, having an accurate prediction ahead of time would mitigate the economic losses from flight delays.**

Thus, **we decided to develop a model to predict whether a flight will be delayed for U.S. air traffic.** As such predictions are more useful when made while ahead of the scheduled takeoff time, we decided to only use the information available before the takeoff, including its flight schedule and weather (forecast).

Certainly, predicting whether a flight will arrive late using its departure delay will yield good results, but such models would not be as informative. This prediction is only available after a plane has taken off. Our model, on the other hand, focuses more on the "Long Term" prediction process. The flight-related features are available once a flight is scheduled, and the weather features can be reasonably approximated using weather forecasts days beforehand. We envision our model to predict whether a flight would delay days before its departure for the customers and airlines to plan the trip accordingly.

1.2 OUR DATASET

Various factors can lead to flight delays, including the airport's traffic volume, different airline policies, and changing weather conditions. To make accurate predictions, we rely on the combination of two major datasets:

Airline reporting carrier on-time performance dataset, which contains flight information like start/destinations, airline code, etc [2].

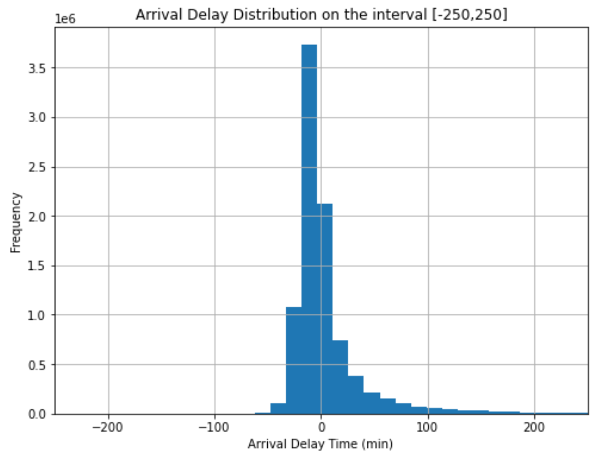
US Weather Events, which contains weather warnings issued by weather stations in different airports [3].

The first dataset reveals delay trends that differ by flight characteristics. From the second one, we expect to see how severe weather affects delay times.

1.3 PRELIMINARY DATA ANALYSIS

There are around 9 million data points in the Flight Performance dataset. We first look at the *ArrDelay* column since this is our response variable. The average delay time of all flights is about 4 minutes, of which 37% of the total flights were delayed. Negative delay time indicates that the flight arrived before the planned arrival time.

The following graph shows the distribution of arrival time delay in the [-250, 250] time interval.



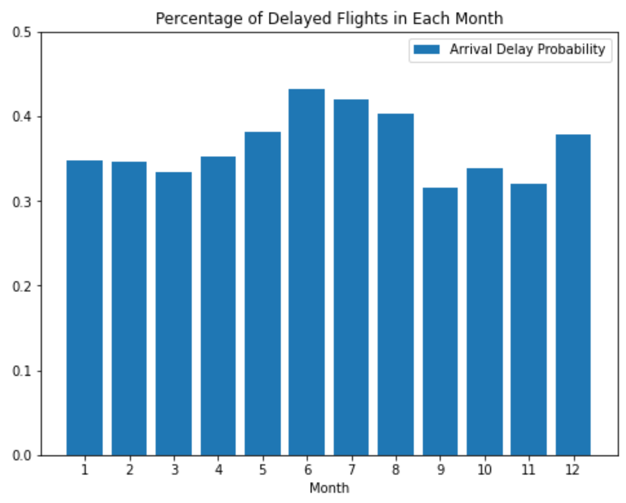
Note that we found the correlation between *ArrDelay* and *DepDelay* to be .91, which is very high and is expected. However, since we focus on the long-term prediction model, we cannot use the *DepDelay* feature to predict *ArrDelay*.

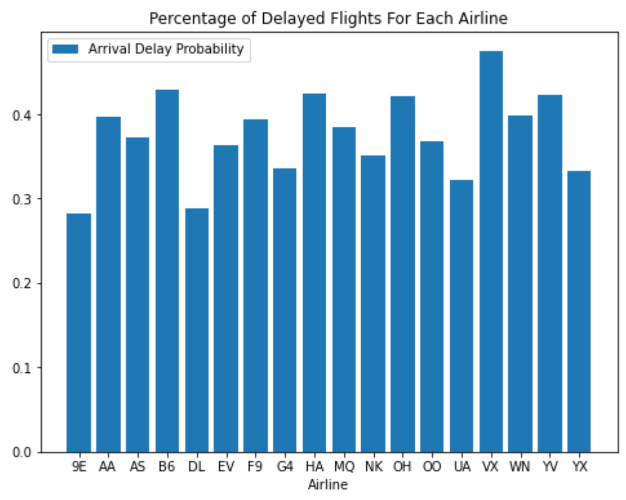
To help with our further analysis, we added an additional column *is_delay* that contains a boolean variable returning True if the flight was delayed. This helps us to define the problem as a classification problem. Now we want to look at the delay time distribution by: *Month*, *Reporting_Airline*, and *Origin_Airport*.

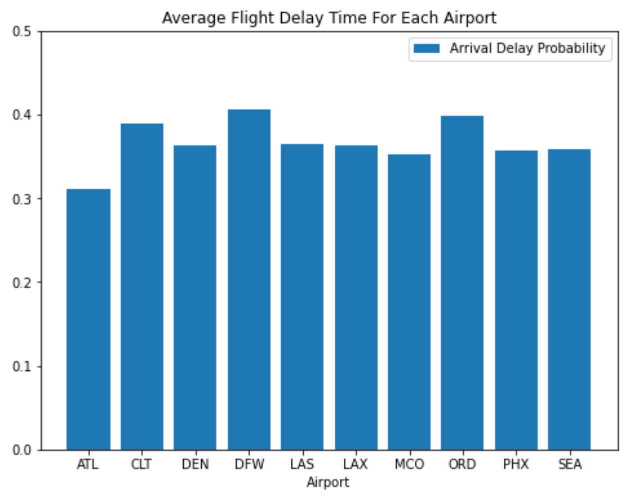
First, we look at months. We see that the total number of flights is considerably higher in December and January. We presume that this is due to traveling around Christmas times. However, when we look at the percentage of delayed flights, as well as the average flight delay time, we see the most delays during the summer: June, July, and August. This is a common time for family trips and vacation travels.

Second, we look at airlines. We see that Delta and Endeavor Air have the lowest flight delay rate. Both have rates less than 30%. Delta and Alaska Airlines have the lowest mean arrival delay time. Both are less than 1 min. To conclude, Delta Airlines stands out when we evaluate the delay time distribution with respect to airlines.

Finally, we look at airports. We see that among the top 10 largest airports for departure, the delay time distributes pretty evenly. *DFW* (Dallas) and *ORD* (Chicago) might have a slightly higher rate of delay and higher average delay time.







1.4 DATA SCOPE

After we conducted preliminary analysis and processing of the dataset, we realized the data size is too large, with over 9 million data points, and that each feature's data distribution is very diverse. This adds extra complexity for feature engineering, such as the departure and arrival airport. These factors may largely influence the performance of the model. Therefore, for more accurate predictions, we restrained the scope of the dataset to be flights between the 10 airports with most flights.

1.5 FEATURE ENGINEERING

To simplify data and enhance model accuracy, we first reviewed Javier's previous work on a similar dataset [4]. We decided to leverage the data to conduct feature engineering in the following aspects.

1. Weather-related: For condition severity, each weather condition (rain, fog, etc.) has 5 possible values: "NA" (no), "Light", "Moderate", "Heavy" and "Severe" to represent how serious the condition is. Thus, we used real encoding from 0 (NA) to 4 (Severe) for indication.
2. Time-related: For *DepTime*, if we turn each take off time into a category by one-hot, the number of features will be too large and leads to overfitting. Therefore, we matched each *DepTime* into midnight (0-6), morning (6-12), afternoon (12-18) and night (18-24) and then did one-hot encoding on these 4 categories instead. For flight date, we pick Quarter and *DayofWeek* as categorical data and encode with one-hot.
3. Airline-related: We consider airline companies to be more representative than flight number in prediction. Thus, we

encoded *Reporting_Airline* by one-hot and dropped *Flight_Number_Reporting_Airline*.

4. Location-related: We first encoded Origin (origin airport) by one-hot. Then for the arrival airport, there are too many values which may possibly lead to overfitting. Thus, we decided to use *Dest* (arrival airport) as destination category.

After feature engineering, we have 1969084 rows of data and 70 features. Our underlying assumption is to keep the features that can be acquired ahead of takeoff to raise the usability and generalization of our model. For example, we dropped the feature *depDelay* as through preliminary training, we observed that its importance on prediction is very high, far exceeding other features. Logically, it's also obvious that if the plane departs late, it might have an arrival delay.

2. MODEL

2.1 INTRODUCTION

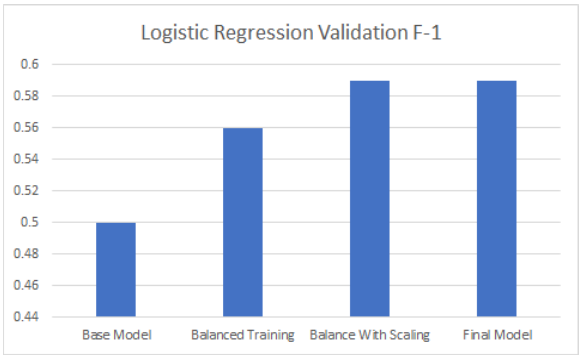
As we have phrased this project as a classification task, we picked some models we learned in class that fit this use case: **logistic regression, random forest, and boosting trees.** We use tree-based ensemble methods because we believe that the features affect flight delay in a non-linear way, and tree-based methods can better capture such trends.

In addition, we used XG boost, which is a dominating used library in machine learning which can develop fast and high-performance gradient boosting tree models. It improves upon Gradient Boosting by adding clever penalizations for trees and using Newton Boosting [5]. Though this method was not covered in class, we believe it is suitable in our case and is worth trying.

For all our models, we followed the same framework of train-test-validation split: We train our model on training data; validate and tune hyper-parameters on k-fold of training data or on validation data; then we finalize the hyperparameters and produces a final test accuracy on the test set. Since we are facing a classification problem with an unbalanced dataset, we decided to use the **weight F-1 score** as a measure of how good our model performs.

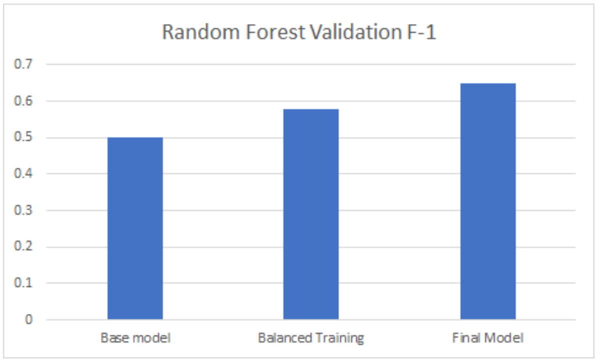
2.2 LOGISTIC REGRESSION

We decided to use logistic regression because it is a simple and straightforward way to approach classification problems. Due to the size of our dataset, we only used linear logistic regression without high order terms. The initial validation F-1 score was around 0.5. We were able to improve the model performance by normalizing features, using balanced training, and tune for penalty and solver. We were able to obtain a final F-1 score of **0.59**. Our result coincides with what we learned in class: logistic regression will benefit greatly from normalizing all features.



2.3 RANDOM FOREST

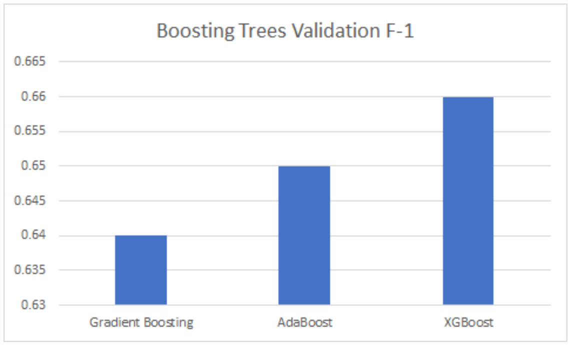
As our features are likely related in some non-linear ways, we decided to try out random forest because it can reduce variance of the vanilla decision tree. This is especially helpful if we want to use longer trees to model complex relationships like flight delay. Our baseline random forest yields an F-1 of 0.5, and we were able to reach a final F-1 score of **0.67** by applying class balance and tuning the hyperparameter via grid search and k-fold CV.



2.4 BOOSTING TREE

Despite our random forest's good performance, we believe boosting trees can further improve the accuracy because it can reduce the bias of each simple tree. As there are many boosting techniques, we decided to try and tune each of them and compare their results. Theoretically, AdaBoost and XGboost should yield better performance as their 'step size' is dynamically adjusted, compared to the gradient Boosting Tree.

Our best result in this section is achieved using XGBoost, with an F-1 score of **0.65**. Our result here is not significantly better than the random forest, we believe that could be due to the high intrinsic error.



2.5 RESULT SUMMARY

As a recap, the following table is the final F-1 score we achieved in the end. The final hyperparameters of each of the following models can be found in the corresponding Jupyter Notebook. We are confident that the results we obtained are unbiased estimations of the model's true performance, because we followed a strict train-test-validation data split, as learned in class. The grid-search we performed is conducted using training data with **k-fold cross-validation.**

By running our final model on unknown test data, the accuracy obtained should be an unbiased estimate of the true accuracy.

ALGORITHM	WEIGHT F-1 SCORE
Logistic Regression	0.5947
Random Forest	0.67
Gradient Boosting Tree	0.64
AdaBoost	0.65
XGBoost	0.66
Dummy Majority Classifier	0.49

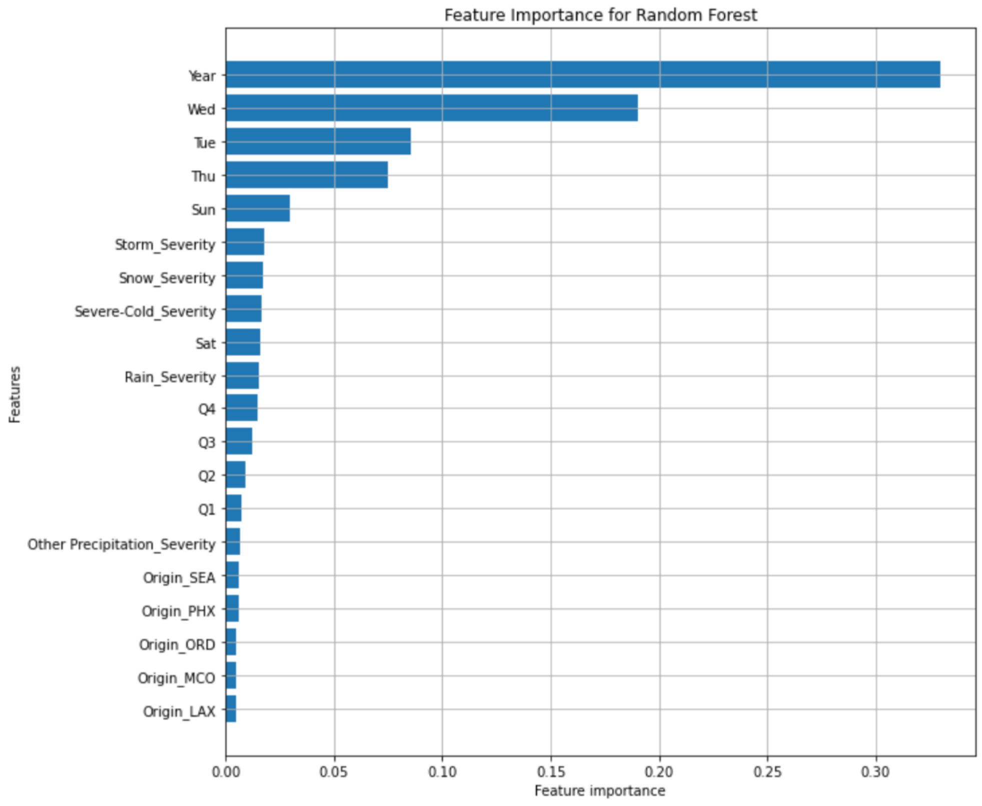
As weighted F-1 score sometimes can be misleading when the dataset is highly unbalanced, we also included a dummy majority model's weighted F-1 score as a reference. Clearly, all our models were able to extract useful information beyond simply guessing every plane arrived on time. Considering the fact that we are only using data available long before the plane's departure, we would say our model functions as a good source to look to for the customers and airlines when they plan their flights.

After balancing final performance and interpretability, we propose random forest as our final model. We will focus our interpretation and fairness assessment on the random forest model as well.

2.6 MODEL INTERPRETATION AND CONFIDENCE IN PRODUCTION

We decided to explore feature importance in the Random Forest model because it may reveal the difference in importance between each feature. We decided to use Sklearn's feature importance, calculated by averaging the accumulation of the impurity decrease within each tree [6].

As shown here, the most informative features are the flight year, days in week, quarter in year, and weather conditions. We were able to observe that the origin and destination of flights and their carriers **did not seem to have a large impact on the delay status,** which is very interesting because we observed meaningful differences in delay probabilities across different airports and airlines.



Relying on the year feature too much is also alarming because it presents a possibility that the delay characteristics change over time. As our problem is constructed as a stationary model (not changing over time), that could mean the model will soon become obsolete. To test it, we train a random forest model from 2016-2019 and use 2020 as a test sample. We achieved an F-1 score of 0.64, similar to what we had in the first place. **The fact that our model performed well on future data unavailable during training makes us more confident that it can work in reality.**

Overall, we are confident that our model can be used in production. Although we did not achieve exceptionally high accuracy, we believe that the model starts the process of providing more accurate, useful flight delay information to the public. By conducting various kinds of out-of-sample tests, our model achieved consistent performance. We believe that it can yield similar results in the production setting. Since our model is developed over a relatively short time horizon (2016-2020), there may exist some long-term changes in the dynamics of flight delays. To combat this, we propose the model to be periodically updated using the latest information, like a rolling 5-year window.

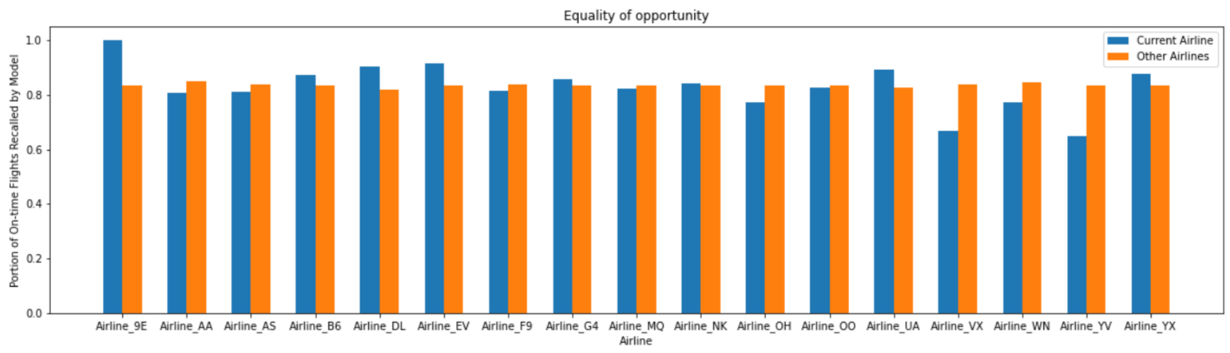
2.7 MODEL FAIRNESS

WMD (Weapons of Math Destruction) is a predictive model that has outcomes hard to measure, and predictions that cause negative consequences and create self-fulfilling feedback loops. We **do not** believe our model to be a WMD because:

Whether a flight is delayed or not is easily measurable. This means we can have a straightforward understanding of our model's performance and observe its test error.

Our prediction will not create self-enforcing feedback cycles. If a flight is predicted to be run late, air-traffic controllers can assign faster routes and mitigate the problem instead of intensifying it.

Although we don't believe our model to be WMD, we had to acknowledge that it could harm certain airlines if the model "discriminates" them and predicts their flights to be constantly late. This could lead to consumers not choosing this airline and damage their business and employees indirectly. To assess the fairness of our model on different airlines, we decided to use the equality of opportunity model. The reason is that different airlines exhibit different delay possibilities, and we do not want to leave that information out by using metrics like unawareness. Also, as the fairness of our model is not regulated by the law, we can use metrics that provide incentives to reduce error uniformly in all groups. Specifically, we want to ensure all the flights that are on time are given equal opportunity to be represented accurately by the model. The result for our Random Forest model on the test set is as follow:



As shown in the plot, the model yields no clear bias for most of the airlines, while predicting fewer on-time flights from airlines ‘VX, YV’ and more on-time flights from airline ‘9E’. After examining the dataset, we believe this is caused by the fact that all 3 airlines had too few entries in our dataset (summing to 0.8% of total flights). This is partially due to the limited scope of our project, and we believe that when the model is trained on the whole US domestic flights dataset, this apparent ‘unfairness’ towards airlines will be resolved.

3. CONCLUSION

Given the flight delay model, different agents, such as consumers, airlines, and air traffic controllers, can all benefit from its prediction. Although we cannot avoid flight delays entirely, having an accurate prediction ahead of time would mitigate the economic losses from flight delays. Based on the models we implement in this project, while all of them outperformed the dummy majority classifier, we proposed using a random forest with weighted-F1 score of 0.67 to get better predictions based on its precision and interpretability. Although we are satisfied with the model, we recognized the limitations embedded. Seen from the feature importance, it is highly dependent on the Year factor which may hinder the model to be applied on future values. Even with our improved model that takes out Year as a highly weighted feature, the weighted-F1 score, although outperformed the dummy classifier, is not considered to be high enough to provide very accurate predictions. We presume this can be improved when we use the dataset of the whole U.S. instead of only the top 10 airports to train our model, if we have had better processing power to handle such a large dataset. We would also recommend updating the model every year or every 5 years of the most recent data to keep up with the trend. Yet our analysis of model fairness has proved no clear bias towards any particular airline or airport.

REFERENCES

[1] J. Rebollo and H. Balakrishnan, "Characterization and prediction of air traffic delays," *Transportation Research Part C: Emerging Technologies*, vol 44, pp 231-241, Available: https://www.sciencedirect.com/science/article/pii/S0968090X14001041

[2] US Bureau of Transportation Statistics, "Airline Reporting Carrier On-Time Performance Dataset", U.S, Published on: June 25, 2020, Accessed on: Nov 2021. Available: https://developer.ibm.com/exchanges/data/all/airline/#get-this-dataset

[3] 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.

[4] J. Herbas, "Using Machine Learning to Predict Flight Delays," Oct. 17, 2020, Available: https://medium.com/analytics-vidhya/using-machine-learning-to-predict-flight-delays-e8a50b0bb64c\

[5] R. Gandhi, "Gradient Boosting and XGBoost," May. 6, 2018, Available: https://medium.com/hackernoon/gradient-boosting-and-xgboost-90862daa6c77

[6] "Feature importances with a forest of trees," Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011. Available: https://scikit-learn.org/stable/auto_examples/ensemble/plot_forest_importances.html