Forecasting Flight Delays: A Data Driven Approach

Chelsi Santiago
Brandon Habschied
Geordy Aponte

Problem Definition

Utilizing Flight Status data from 2018 to 2022, create a predictive model to determine if an upcoming flight will experience a delayed arrival, defined as an arrival occurring 15 minutes or more after the scheduled arrival time.

.Our problem definition for this project was to use historical flight data from the years 2018 to 2022 to create a predictive model to determine if a flight will experience a delayed arrival. This delayed arrival is defined as an arrival occurring 15 minutes or more after the scheduled arrival time. The aim is to create a reliable system that classifies upcoming flights as either delayed or on time

Motivation

- The objective is to build a system capable of accurately predicting flight delays, providing valuable insights for travelers, airlines, and airport management to anticipate and manage potential disruptions.
- Real world applications for airlines
  - Fewer cancellations or delays
  - Optimize staffing levels
  - Increased customer satisfaction
- Real world applications for airport management
  - Better management of airspace/runway capacity
  - Enhancing operational efficiency
  - Improvement to safety

By accurately predicting if a flight would be delayed or not, valuable insights could be provided for travelers, airlines and airport management.

For airlines, this would mean few cancellations or delays leading to increased customer satisfaction. They could also use this data to optimize staffing levels so they have the right number of people, whether it be flight crew or gate attendants to make sure their operations run smoothly.

For airports, there would be better airspace/runway capacity since flights would be arriving during their scheduled time. There would be more operational efficiency and improvements made to safety as well.

Key Issues

- Data Files were very large
  - Combined_Flights_2018.csv: 1.99 GB
  - Combined_Flights_2019.csv: 2.82 GB
  - Combined_Flights_2020.csv: 1.75 GB
  - Combined_Flights_2021.csv: 2.21 GB
  - Combined_Flights_2022.csv: 1.42GB

One of our biggest challenges was the massive size of our data. The dataset we chose spanned from the start of 2018 through July 31st of 2022. Trying to process all of this data proved to be a challenge. In order to reduce data size, we utilized the more efficient parquet files which combined the entire data set to approximately 1 GB in size so the data could be loaded into Jupyter notebooks.

However, even using the parquet file type, we were still only able to process the data in chunks due to to computational complexity of the models. Even with half of the features removed, it still took at least four hours to tune and create a random forest model. As a result of these extremely long run times, we decided to run the majority of our models with only the data from 2021 which was our most recent year with a full 12 months of data since the 2022 data was missing the last quarter.

Related Works

Airline delay prediction by machine learning algorithms

By H. Khaksar and A. Sheikholeslami

- Using a different data set, Khaksar and Sheikholeslami worked to predict commercial airline delays.
- Also used classifiers to determine if airlines would be delayed and by how long.
    - Accuracy: 61%-76%
    - Precision: 51%-79%
    - Recall: 50%-95%

Prior works used weather data to determine how often an airliner would be late and by how long. They focused on using modeling methods like decision trees, random forest, bayesian classification, k-means clustering, and a hybrid of clustering and decision trees. Similarly, we used sampling methods and normalizing the data in our preprocessing, along with removing features based on the correlation matrix to create a more compact data set. We see that of the methods they used, their metrics vary for accuracy, precision, and recall and we used these values to determine the overall performance of our models.

Approach

- Data preprocessing
- Test models on 2021 data before applying core algorithm to entire dataset
- Modeling
  - LightGBM (Core Algorithm)
  - Random Forest
  - Random Forest with ADABoost
  - QDA
  - K-NN

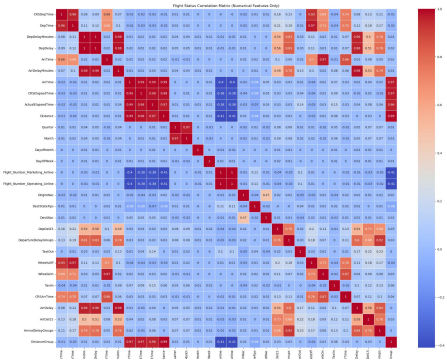
Data Preprocessing: Half of the features were removed from our data set because they would not be useful in our model or were highly correlated with other features. Due to an imbalance of the data favoring certain features when we initially ran our models, we implemented undersampling techniques to balance the data. We also normalized the data to ensure that the data was on an equal scale to further prevent features from dominating the learning process simply because they had larger magnitudes.

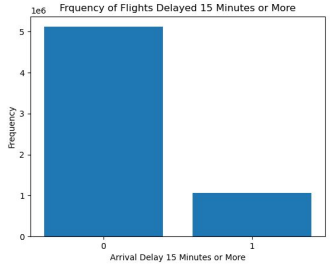
2021 Data set: As previously mentioned, we chose to focus on just one year of data in order to preserve computational power. We wanted the most recent full year data which was 2021.

Since a flight was either delayed or one time, our response variable was binary. Therefore we chose to focus on models that are effective in binary classification. We implemented Light Gradient Boosting Machine (LightGBM), Random Forest, random forest with ADABoost, QDA, and K-NN as our models.

Data Preprocessing

- Highly correlated variables removed
- Undersampled the predominate class due to class imbalance
- Normalized the data to reduce the impact of scale on the models
- Split data into test and training sets



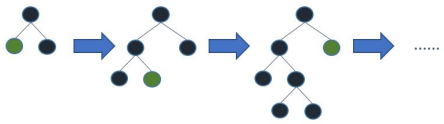


To begin our data preprocessing, we isolated the data from 2021 then dropped all useless or redundant features that mostly included ID values. Next, we created a correlation matrix of our remaining variables, from this matrix we found that there were multiple highly correlated features and decided to remove those as well. An example of this is the feature *distance group* which classifies a flight distance into a group such as a short flight, average flight, or a long flight. This feature was highly correlated with distance and total air time for obvious reasons when we look at the features knowing that a longer flight will have a greater distance and total air time. We proceeded to do this with any of the other similarly correlated features.

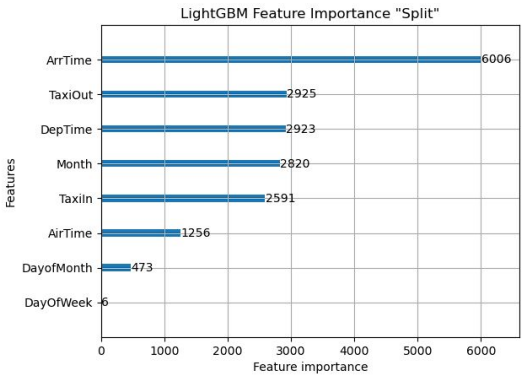
Next we undersampled the data due to an imbalance in our data. We found a significant imbalance in the distribution of our target variable where for every flight that was delayed, there were five flights that were not. We approached this issue by undersampling from our on-time arrivals group.
We also normalized the data because we found that we use a variety of different scales in this data set; an example is distance vs day. This data composes a large number of flights which ranges from 31 miles to over 6000 miles but the day of the month only ranges from 1-31.

Core Algorithm: LightGBM

- Without tuning:
  - Accuracy: 82.7%
  - Precision: 100%
  - Recall: 83%
  - F1: 91%
- Quickest run time
- Low memory usage
- Performs best with large scale data



Leaf-wise tree growth



The core algorithm that we selected is Light gradient Boosting Machine. LightGBM is a gradient boosting algorithm that runs very efficiently on large data sets and can be used for regression and classification. It works by building leaf-wise trees by choosing to grow trees on the leaf that will achieve the lower loss. It also utilizes bins when determining split points to speed up training and reduce cost and memory usage.

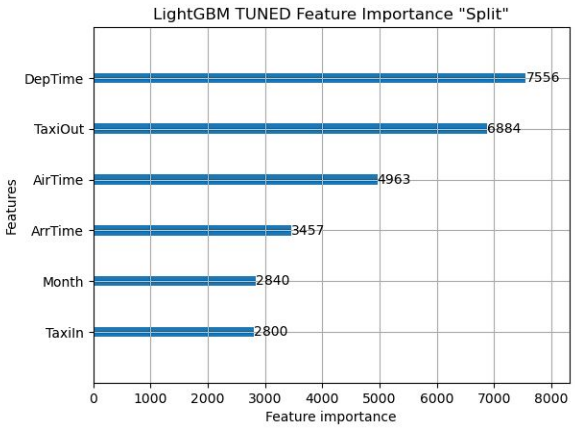
Without any tuning, the LightGBM algorithm ran very well, producing metrics that are seemingly desirable. The model is able to accurately label a flight 82.7% of the time and does this without any false positives with a 100% precision score. The model also catches 83% of all positive values.

The runtime of this model was also relatively quick compared to the other models that we ran in this project, highlighting its efficiency.

After creating our model and analyzing feature importance, we found that our most important features on are Arrival time, taxi out, and then departure time.

LightGBM Tuning

- Additional feature selection by removing **Day of Month** and **Day of Week**..
  - Accuracy: 90.64%
  - Precision: 100%
  - Recall: 89.8%
  - F1: 94.6%
  - NPV: 47%



	Predicted Ontime	Predicted Delayed
Actual Ontime	4666492	0
Actual Delayed	530712	470698

To tune our core algorithm, We performed some additional feature selection, removing poor performing feature like day of month and day of week. We additionally allowed the model to iterate for twice as long with a higher learning rate and lower feature fraction. After playing with these parameters, we were able to improve accuracy and recall scores while still maintaining our precision and f1 scores. We see that Departure time and taxi out time continue to be our most impactful features for prediction.

One statistic worth investigating is The NPV or Negative Predictive Value, which is the percentage of delayed flights that were correctly predicted is 47%. This number is slightly concerning when we look at our data as a whole. Although our Accuracy is 90%, we are only correctly predicting 47% of all delayed flights. This statistic is something that should be explored in future expansions on this dataset.

Random Forest

- Accuracy of 84%, precision of 53%, recall of 39% and F1 scores of 45%
- 5 fold cross validation with hyper parameterization
  - Number of trees in forrest tested from 1-200 in steps of 75
  - Max depth of tree tested from 5-15 in steps of 5
  - Min samples needed to split the data test from 10-41 in steps of 15
  - Min samples leaf was set to 20-50 in steps of 15
- Long run time

```
Feature ranking:
0. Feature TaxiOut - 0.284605465488644
1. Feature ArrTime - 0.2183034564118791
2. Feature DepTime - 0.20437684813296605
3. Feature Month - 0.12143744287611803
4. Feature TaxiIn - 0.10357994652162118
5. Feature AirTime - 0.03864157443695556
6. Feature DayofMonth - 0.02033832118304753
7. Feature DayOfWeek - 0.008716944948768575
8. Feature Year - 0.0
```

Another one of our models that we built was a Random Forest. It performed decently well with 84% accuracy while the the precision of this model met our expectation at 53%. However with this model, the recall and F1 scores are below our expectations with a value of 39% and 45% respectively.

One additional downside to this model was it being one of our more complex models due to cross validation, it took multiple hours to run before it was complete. This made it impractical for us to utilize in our dataset as we would have to re-run the model multiple times to tune it.

The result of this model did agree with our previous models and we can see that Taxi out, Arrival time, and departure time are similarly the most important features.

Hybrid: Random Forest with ADABoost

- Accuracy of 85%, precision of 78%, recall of 20% and F1 scores of 32%
- 5 fold cross validation with hyper parameterization
  - Number of trees in forrest tested from 1-200 in steps of 75
  - Max depth of tree tested from 5-15 in steps of 5
  - Min samples needed to split the data test from 10-41 in steps of 15
  - Min samples leaf was set to 20-50 in steps of 15
- Longer run time

```
Feature ranking:
1. Feature TaxiOut - 0.3242026868104626
2. Feature ArrTime - 0.2437466798799728
3. Feature DepTime - 0.15240007303287828
4. Feature TaxiIn - 0.12084804094315973
5. Feature Month - 0.09785509161384032
6. Feature AirTime - 0.036849029977936085
7. Feature DayofMonth - 0.017313358391660184
8. Feature DayOfWeek - 0.006785039350089938
9. Feature Year - 0.0
```

In another one of our models, we combined Random Forest with ADABoost in an attempt to improve the performance of the random forest model. From this we see that accuracy slightly improves while the precision values significantly increase. This shows that the model is better identifying true positives when it labels a flight as not delayed. With the addition of adaboost, our recall and f1 scores both decreased.

Similar to our previous models, Taxi out, arrival time, and departure time continue to be our most important features.

Because of the addition of ADA Boost to our already time consuming random forest method, this model became our most time consuming model, taking an excess of 12 hours to run making it impractical to make adjustments to this model for additional analysis.

Random Forests Feature Selection

- Due to over fitting, additional feature selection was used on both the Random Forest and Random Forest with ADABoost
  - Random Forest model results: accuracy 82%, precision 49%, recall 36%, F1 Score 41%
  - ADABoost Model results: accuracy 83%, precision 52%, recall 32% and F1 score 40%

```
Feature ranking:
0. Feature DepTime - 0.3645915690338681
1. Feature TaxiOut - 0.3351829688722545
2. Feature Month - 0.15097573646532986
3. Feature TaxiIn - 0.13029802870812549
4. Feature DayOfWeek - 0.018951696920421958
```

Random Forest

...

```
Feature ranking:
0. Feature DepTime - 0.41641662445870264
1. Feature TaxiOut - 0.3580695527188381
2. Feature TaxiIn - 0.14257274738815368
3. Feature AirTime - 0.0829410754343055
4. Feature Year - 0.0
```

Random Forest with ADABoost

To combat overfitting, additional feature selection was performed on the Random Forest model and Random Forest with ADAboost. This feature selection helped reinforce the feature importance that time of departure and Taxi time were the most important features in predicting our target variable.

It is important to note that for both of these models, their precision and recall were relatively low with around 50% precision and only around 34% recall.

Our additional use of feature selection had a negative impact on our results which lead us to choose to remain with our previously made models, even though these previous models took significantly longer to run.

QDA

- Accuracy of 83%, precision of 0%, recall 0% and F1 score of 0%
- 5 fold cross validation with hyper parameterization
  - Regularization parameters with values 0-1 in steps of 8
- Adequate model development time

Quadratic Discriminant Analysis

An accuracy of 83% with a recall and precision of 0% suggests that our model is making correct predictions for a majority of the instances but is failing to identify any positive instances (recall) and is not making correct positive predictions (precision). This can be caused by a few things like class imbalance and poor tuning of the model. As mentioned earlier, we performed data preprocessing that we used for all of the different models; in order to test which model performed the best, we decided to use the same preprocessing steps for all the models. When we performed our class balancing, it is possible that the class balancing favored other model but did not favor our QDA model.

Also, after tuning the model, we still had difficulty trying to improve the metrics of this model and with specific data preprocessing and feature selection, it is possible to achieve better results with this model.

KNN

- Accuracy of 82%, precision of 48%, recall 40% and F1 score of 44%
- 5 fold cross validation with hyper parameterization
  - K values: 10, 15, 25, 30
  - Distance metrics: Euclidean, Manhattan and Minkowski
- Best model used k=25 and Euclidean distance calculation
- Adequate model development time

This model performed relatively well, with good accuracy and acceptable precision and recall values. These results were achieved by using k = 25 and Euclidean distance. With more tuning and adjusting feature selection, it is possible to further improve these values.

Compared to the target values from our reference article, KNN is also performing well we an accuracy score of 82% but is performing poorly consistently identifying true positives with poor precision and recall values.

Validation

The research questions we wanted to answer were:

RQ1: Which feature(s) contributed the most to delayed flights?

RQ2: Which model would perform the best for our problem?

Cont. next 2 slides

RQ1: Which feature(s) contribute the most to flight delays?

Taxi Out: Taxi Out Time, in Minutes

> The duration an aircraft takes to move from the departure gate to the runway before takeoff.

Departure Time: Actual Departure Time (local time: hhmm)

> Actual time that the aircraft departs from the airport.

Throughout multiple different models, we see that both Taxi out and departure time are features that continuously rank as the most important features across multiple different models.

From this, we see that the main factor causing arrival delays is most directly related to before the plane leaves. The time that the plane departs from the airport has an effect on the plane actuals arrival time to its destination. This can further be impacted by the taxi out time, where a plane that has left the terminal but is waiting to take off can also affect the arrival time.

Based off of these features, it would be best for airports to have a focus on ensuring that there are no delays to aircraft departures as this can then lead to delayed arrivals as well.

RQ2: Which model would perform the best for our problem?

	LightGBM	Random Forest	Random Forest w/ ADABoost & Feature Selection	QDA	KNN
Accuracy	90.64%	84%	83%	83%	82%
Precision	100%	53%	52%	0%	48%
Recall	89.8%	39%	32%	0%	40%
F1	91%	45%	40%	0%	44%

LightGBM and KNN were our preferred models that we used for this project.

Because QDA was only to produce 0% recall and precision score, it was not possible to rank this model higher because of its poor performance.

Because of how much time that both random forest and random forest with adaboost took to run, we did not prefer this models either. Even though random forest with adaboost performed slightly better than random forest, because it took significantly longer to develop, we rank it below a normal random forest.

Both LightGBM and KNN performed better than the rest of the models but LightGBM had significantly better precision, recall, and thus f1 scores than KNN. LightGBM was also one of the methods that developed models the quickest. As such, we prefer this model over the rest.

1. LightGBM
2. KNN
3. Random Forest
4. Random Forest with ADABoost
5. QDA

Conclusions

Emphasis on pre-departure condition features such as departure time and time to taxi.

High Accuracy, Low NPV - Real world implications.

Recommendation to focus on improving scheduling and logistics

After creating multiple models and then tuning our LightGBM model, we can conclude that there is a significant importance on pre-departure conditions when it comes to predicting arrival delays in airlines. It should come as no surprise that if a plane has a significantly delayed departure, it would be very unlikely that the plane arrives on time at its destination.

From our models, we achieved satisfying accuracy scores, but upon further review, these accuracy scores may come with misleading statistics that don't fully consider the significant impact that delayed travel plans can have on travelers and the airline industry. Our lower Negative Predictive Values can have significant real world implications if not they are not explored further. A customer who is dissatisfied is more likely to voice their dissatisfaction than a satisfied customer would be to voice their satisfaction.

Our business recommendation based on the results of our models is to solidify the importance of improving scheduling and logistics when it comes to departures of airplanes from airports. If there are significant delays in the taxi process due to errors in scheduling, loading, fueling or any other factor that leads to delays in taxi time or departure time, the plane is going to be more likely to have a delayed arrival at its destination. If the airlines focus on improving these processes, they can make strides towards reducing the total amount of delayed delayed flights. This recommendation is very generalized as we did not segment our data per airline and each airline or airport may have unique issues that contribute to delayed arrivals.

Future Work

- Expand to the rest of the data set.
- Fine tune LightGBM.
- Potentially predict how long a flight will be delayed.
- Explore additional features that impact flight delay.
- Integrate weather forecasts into delay predictions.

With more time, we would be able to work with the rest of the data set instead of primarily focusing on just the 2021 data set. As stated, these data sets took a long time to run so we would need time to run each data set separately before we were able to combined them.

For us, we preferred the LightGBM because it yielded satisfactory results and did not take an excessive amount of time and memory to run like our other models. Creating a tuned LightGBM model for each year and then ensembling the models together would allow us to hopefully create a well performing model that can predict if a flight will be delayed or not.

We would also like to explore additional features such as weather impact on flight arrival times. The weather conditions at the time of departure and arrival could have a significant impact on why flights are being delayed and through combining weather forecasts and flight schedules, we could predict flight delays multiple days in advance.

Works Cited

Khaksar, H., and A. Sheikholeslami. "Airline Delay Prediction by Machine Learning Algorithms." *Scientia Iranica.Transaction A, Civil Engineering*, vol. 26, no. 5, 2019, pp. 2689-2702. *ProQuest*, https://ezproxy.rit.edu/login?url=https://www.proquest.com/scholarly-journals/airline-delay-prediction-machine-learning/docview/2307793294/se-2, doi:https://doi.org/10.24200/sci.2017.20020.

https://lightgbm.readthedocs.io/en/stable/#

Thank you for engaging with our presentation on creating models to predict airplane arrival delays; your attention and interest are truly appreciated