Flight Delay Prediction using Machine Learning

Dhiganth Rao

SSN College of Engineering, Chennai, India
dhiganth18037@ece.ssn.edu.in

Abstract : Flight delay has long since been a problem that commercial airline companies have struggled with. The costs incurred due to the delay of a flight is large, and this delay also results in problems for the consumers themselves. This paper covers a simple Machine Learning approach to predicting a flight's arrival delay time based on weather conditions. This paper implements a two stage process - Classification and Regression.

Keywords: Machine Learning · Two-Stage Model · Flight Arrival Delay

1 Introduction

The air traffic sector is continually affected by delay of aircraft. Between 2004 and 2017, around 22% of airline flights within the United States were delayed or cancelled. (Bureau of Transport Statistics, 2018). Generally, delays emerge when interactions between air transport players (i.e., carriers, airports and air traffic control entities) and external factors (i.e., adverse weather conditions, strikes and other incidents) lead to airport congestion. Delays directly lead to extra travel time. In reaction to uncertain travel times, travelers may adjust their travelling schedule to account for delays. This, in turn, causes problems for the travellers. Hence, there is a need for a model that accurately predicts if a flight may be delayed or not. It is found that weather conditions play a role in a flight's arrival and departure timings. Existing literature suggests that the general impact adverse weather conditions have on airport and airline operations is considerable. Robinson(1989) analyzed the impact of different weather shocks on airline operations at the Atlanta Hartsfield International Airport for one airline. He found that annually over 165,000 min of delay are attributable to adverse weather conditions. Changnon (1996) also found that at the end of the 1970s, rainfall substantially increased the number of departures with a delay above 30 min at Chicago O'Hare airport.

With proof, we can now examine the impact of weather conditions on the delay of a flight on its arrival. Datasets recording weather conditions at 15 different airports during the interval of 2016 - 2017 were used. Datasets with information on the On-Time Statistics of various flights during this time interval were also used.

Section 2 deals with how the flight and weather datasets were processed and finally merged to form a dataset to be fed into the model. The next section, 3 deals with how different classifies were trained and tested with the dataset.The next section, 4 deals with how different regressors were also trained and tested with the dataset. Finally, section 5 deals with how the flow of data was described through a pipeline.

2 Data Pre-Processing

2.1 Raw Dataset Overview

Initially, each weather dataset provided contained information about the weather conditions hourly, for a particular month, at a particular airport. All the datasets were processed and merged into a bigger dataset containing records of each hour, for every day of the years 2016-2017.

The same approach was followed for the flight datasets as well. After the weather and flight datasets were merged, a dataset was formed which contained data such as weather conditions at the origin airport, and the delay(if any) at the Destination airport.

2.2 Selecting Relevant Features

Features selected from the weather and flight dataset are shown below.

Table 1. Selected Features from the Flight Dataset.

FlightDate	Quarter	Year	Month
DayofMonth	DepTime	DepDel15	CRSDepTime
DepDelayMinutes	OriginAirportID	DestAirportID	ArrTime
CRSArrTime	ArrDel15	ArrDelayMinutes	

Table 2. Selected Features from the Weather Dataset.

WindSpeedKmph	WindDirDegree	WeatherCode	precipMM
Visibility	Pressure	CloudCover	DewPointF
WindGustKmph	tempF	WindChillF	Humidity
date	time	airport	

2.3 Merging the two Datasets

In order to merge the two datasets properly, features such as:

- the airport where the weather conditions were recorded,
- the airport from which the flight departed from,
- the date on which the weather conditions were recorded,
- the date on which the flight departed,
- the time at which a particular weather condition was recorded, and

– the time at which the flight departed to the nearest hour formed a key to merge the two datasets.
  Thus, a new dataset was derived where the weather conditions during the time of departure of flights across various airports were mapped together. This dataset was then split into a Train dataset and a Test dataset with the train-test split being 75-25.

3 Classification

Four different classifier algorithms have been tested to see how well they classify the data into ’Delayed’ or ’Not Delayed’. In order to determine the performance of the classifiers, certain metrics have to be defined.

3.1 Classifier Metrics

The metrics used in Table 3 are defined below.

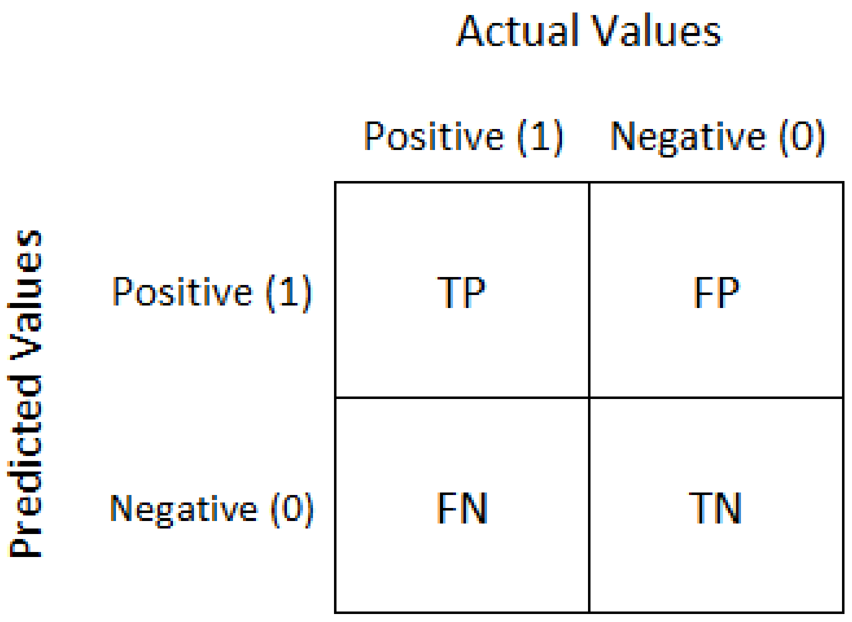

Figure 1. Confusion Matrix

From the Confusion Matrix depicted in Figure 1,
True Positives: True positives are the cases the model labels as ’Not Delayed’, that are actually ’Not Delayed’.

False Positives: False positives are the cases which the model labels as 'Not Delayed', while they are actually 'Delayed'.
True Negatives: True Negatives are the cases the model labels as 'Delayed', that are actually 'Delayed'.
False Negatives: False Negatives are the cases which the model labels as 'Delayed', while they are actually 'Not Delayed'.
Precision: Precision is defined as the number of True Positives divided by the number of True Positives plus the number of False Positives.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{1}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2}$$

$$\text{F1-Score} = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{3}$$

The results obtained by the different classifiers are tabulated in Table 3.

Table 3. Results obtained using different Classifier models.

Algorithm	Precision		Recall		F1-Score		Accuracy
	0	1	0	1	0	1	
Logistic Regression	0.92	0.89	0.98	0.68	0.95	0.77	0.92
Random Forest	0.92	0.87	0.97	0.70	0.95	0.78	0.92
Extra Trees	0.92	0.76	0.94	0.71	0.93	0.74	0.89
Gaussian Naive Bayes	0.94	0.75	0.93	0.76	0.93	0.75	0.90

To classify the data, the Random Forest Classifier model was chosen as it had the best results overall for both classes. The F1-Score metric was chosen to decide between the Logistic Regression model and the Random Forest model, as the F1-Score provides a balance between both Precision and Recall, both of which are important to consider.

3.2 Data Imbalance

From Table 3, we can observe that the results of precision, recall and F1-Score of any of the above classifiers for the label '1' are low when compared to the results obtained for the label '0'. This can be attributed as a result of class imbalance between the labels present in the merged dataset.The class imbalance is shown in Figure 2.

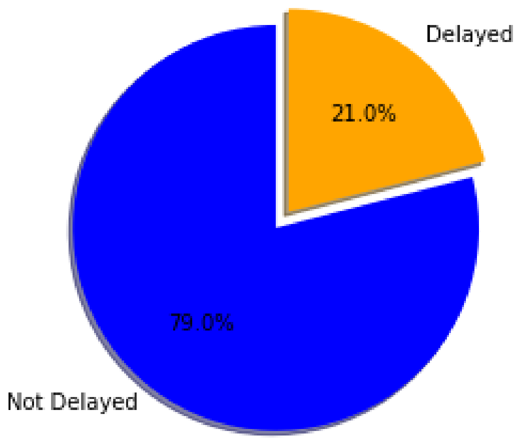

Figure 2. Plot of Class Imbalance.

3.3 Solving the Data Imbalance Problem: Sampling

From Subsection 3.2, we have established that the dataset is imbalanced, with the amount of rows labeled 'Not Delayed' accounting for 79% of the rows in the dataset. Since the classifier models have lesser rows with labels '1' to train on, the results of the classification when classifying label '1' are poor. This situation can be solved by sampling the data. The data can be sampled in many ways. Two modes of sampling are shown here.

- **A Sampling Method: Random Under Sampling**
  Random Under Sampling is a sampling technique which involves randomly selecting examples from the majority class and deleting them from the training dataset. It is important to note that while this improves the ratio between the two labels, it is essentially deleting data from the dataset - which might result in poorer performance of the model when compared to not sampling the data.

- **A Sampling Method: SMOTE**
  The Synthetic Minority Oversampling Technique, or SMOTE in short, is an oversampling technique which works by selecting examples that are close in the feature space, deriving a line between the examples in the feature space and drawing a new sample at a point along that line.

A comparison of the results obtained after Sampling is shown in Table 4.

Table 4. Comparison between the sampling methods used.

Sampling Method Used	Algorithm	Precision		Recall		F1-Score		Accuracy
		0	1	0	1	0	1	
Random Under Sampling	Logistic Regression	0.94	0.74	0.93	0.78	0.93	0.76	0.90
	Random Forest	0.95	0.68	0.90	0.81	0.92	0.74	0.88
	Extra Trees	0.95	0.58	0.84	0.82	0.89	0.68	0.84
	Gaussian Naive Bayes	0.94	0.73	0.93	0.77	0.93	0.75	0.89
SMOTE	Logistic Regression	0.94	0.74	0.93	0.78	0.93	0.76	0.90
	Random Forest	0.93	0.83	0.96	0.72	0.95	0.78	0.91
	Extra Trees	0.93	0.74	0.93	0.73	0.93	0.73	0.89
	Gaussian Naive Bayes	0.94	0.74	0.93	0.77	0.93	0.75	0.89

While trying to solve the problem of Data Imbalance, SMOTE was chosen as the sampling method. This is because the creation of synthetic data samples which are similar to the original data samples serves to improve the performance of the model used, as the model now has more data samples to work with. As we observed in Random Under Sampling, deletion of random data samples in the majority class may balance the ratio between classes, but reduces the performance of the model.

4 Regression

Once the first model classifies the dataset into 'Delayed' data and 'Non-Delayed' Data, the classified data is now fed into the regressor model for it to predict how long the flight is going to be delayed at Arrival. Four different regressor models have been tested to see how well they predict the delay time of a particular flight. The performace of the regressors are displayed in Table 5.

- the Linear Regressor Model,
- the Random Forest Regressor Model,
- the Extra Trees Regressor Model,
- and the AdaBoost Regressor Model.

In order to assess the performance of each of these regressors, certain metrics need to be defined.

4.1 Regressor Metrics

y_i: Ground truth
$\hat{y_i}$: Predicted value

$$Mean\ Absolute\ Error = \frac{\sum_{i=1}^{n} |y_i - \hat{y_i}|}{n} \tag{4}$$

$$Root\ Mean\ Squared\ Error = \sqrt{\frac{\sum_{i=1}^{n} (\hat{y_i} - y_i)^2}{n}} \tag{5}$$

$$R^2\ Score = 1 - \frac{MSE(model)}{MSE(baseline)} \tag{6}$$

4.2 Performance Analysis

Table 5. Performance of various Regression Models.

Algorithm	Mean Absolute Error	Root Mean Square Error	Rˆ2 Score
Linear Regression	12.3272	17.6878	0.9384
Random Forest	12.0286	17.0110	0.9430
Extra Trees	12.1950	17.2835	0.9412
AdaBoost	19.3185	24.3344	0.8834

The Root Mean Square Error was chosen as the deciding metric as it provides a better description of the performance of the model. Based on the value of the RMSE metric, the Random Forest regressor was chosen.

4.3 Regression Testing

The plot in Figure 3 shows the distribution of delay values in the dataset.

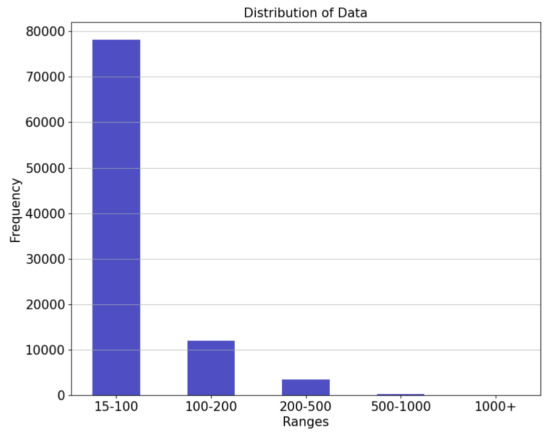

Figure 3. Distribution of delay in minutes in the test dataset.

Regression testing involves feeding the regressor certain blocks of data where the arrival delay of the flight in minutes is within a certain range, say 15-100 minutes or 100-200 minutes. The distribution of the original test dataset with respect to the arrival delay of the flight is shown below.

In order to test the performance of the regressor, these blocks are fed to the regressor and the MAE and RMSE obtained from each block is obtained. The results are tabulated as shown.

Table 6. Results obtained for each block fed to the regressor model.

Delay in Minutes (d)	Mean Absolute Error	Root Mean Squared Error
15 <d <100	10.67085994	14.12010925
100 <d <200	18.23946856	26.8284211
200 <d <500	19.34161915	29.44188209
500 <d <1000	15.94554347	22.04866921
d >1000	24.44189189	42.81110644

From Table 6, we can see that the value of MAE and RMSE for the block 500-100 is relatively low compared to the values of error in the other blocks. However, from Figure 3, we can see that the number of data samples in the range 500-100 is very low. This may be attributed to the imbalance in the dataset.

5 Pipelining

After determining the best models to use for Classification and Regression, a pipeline needs to be constructed to describe the flow of data. The figure shown below shows how the pipeline has been implemented in this paper.

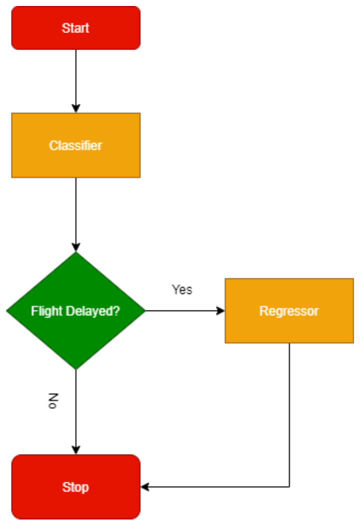

Figure 4. Pipeline showing the flow of data.

The regressor results obtained from the Pipeline are: **Mean Absolute Error** of **14.26**, and a **Root Mean Squared Error** of **19.05**. It is observed that the regressor performs better without being pipelined.This can be attributed to the classifier wrongly classifying data points. From Table 3, and using the formulae described in Section 3.1, we can see that the total number of wrong classifications (i.e, False Positives and False Negatives), turns out to be 81136, which accounts for 17% of the test dataset. Hence, due to these wrong classifications, the regressor performs worse when pipelined.

6 Conclusion

Hence, a simplified approach to predict the arrival delay of a flight based on weather conditions at its departure was discussed and implemented. Different classifiers were tested and the Random Forest Classifier was chosen as it had the best overall performance. Among the regressors, the Random Forest regressor was chosen as it had the lowest error values. After the classifier and regressor were chosen, a pipeline was formed to simulate the flow of data through the two-stage model.

References

1. Robinson, J Peter: The Influence of Weather on Flight Operations at the Atlanta Hartsfield International Airport (1989). https://journals.ametsoc.org/doi/abs/10.1175/1520-0434
2. Chagnon, SA : Effects of summer precipitation on urban transportation (1996). https://link.springer.com/article/10.1007/BF00140357