Flight Delay Prediction

Machine Learning Project Report

- Akshay Ramakrishnan

1. Problem Statement Description

This project comprises of a two-stage model to predict the delay incurred at the destination airport by the flight **before the flight has taken off.** In the first stage, the model predicts whether a given flight would be delayed during arrival or not and in the second stage, the delay in minutes at the destination airport is predicted.

2. Introduction

It is important to have a general idea as a passenger and as an airline coordinator regarding whether or not a flight would be delayed. This helps airline companies to make necessary amends before-hand by letting the passengers know that a delay is predicted so that they would make the required arrangements as well. This helps to increase the overall customer satisfaction level and improve business for the airline as well.

3. Dataset

3.1. Dataset Description

The datasets used for this problem are:

- Weather data for 15 airports for the years 2013-2017. The data is given in JSON format. The weather features are given in Table: 1.
- Flight data of all the domestic flights travelling in USA for the years 2016 and 2017. The data is given in CSV format. The flight features to consider are given in Table: 2.

The fifteen airports to consider in predicting the delay are given in Table: 3.

Table: 1 – Recommended weather features to consider

WindSpeedKmph	WindDirDegree	WeatherCode	precipMM
Visibilty	Pressure	Cloudcover	DewPointF
WindGustKmph	tempF	WindChillF	Humidity
Date	time	airport	

Table: 2 – Recommended flight features to consider

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

Table: 3 – Airport codes to consider

ATL	CLT	DEN	DFW	EWR
IAH	JFK	LAS	LAX	MCO
MIA	ORD	PHX	SEA	SFO

3.2. Preprocessing

We have data of two types: One containing the weather features and another containing the flight features. In order to make the data we have meaningful, we have to **merge** these two datasets on:

- Origin / Destination
- Time
- Date

On merging, we obtain a new dataset containing both weather and flight data which we will use for prediction. This process would map the weather records and the flight records based on the above features and the new dataset would contain an extended number of features.

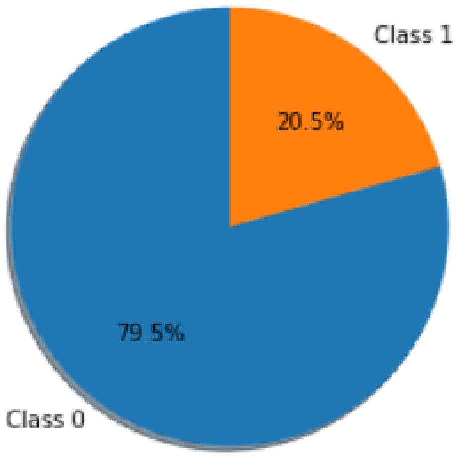
3.3. Class imbalance

Sometimes, the label to be predicted by the Machine Learning model might be highly skewed. The label 'ArrDel15' has two classes:

- Class 0: Indicating that the flight is not delayed on arrival
- Class 1: Indicating that the flight is delayed on arrival

Figure 1 shows the distribution of 'ArrDel15' values between the two classes.

Figure: 1 – Pie chart depicting the distribution of Arrival delay values



As we can see, the number of ArrDel15 values belonging to ‘Class 0’ is about 4 times higher than those belonging to ‘Class 1’. So as a result of this we can say that our model would tend to “favour” ‘Class 0’ and a majority of values would be predicted to fall under this class. This would in turn lead to biased predictions by our model.

4. Classification

4.1. Introduction

The desired output of this stage is to predict whether a given flight would be delayed or not at arrival. There are two classes as mentioned earlier.

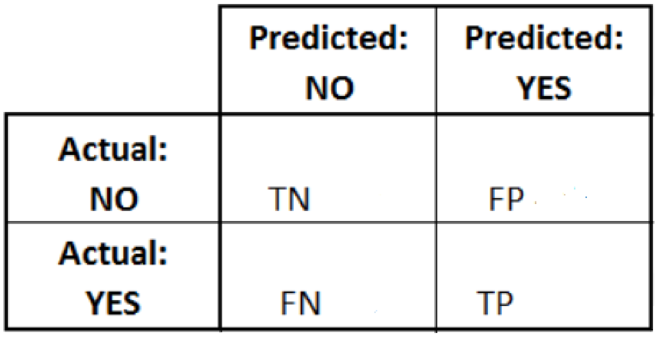
Class ‘0’ indicating that the flight would not be delayed and Class ‘1’ indicating that the flight would be delayed.

The data is divided based on the 80-20 rule where 80% of the data is used as the training set and 20% of the data is used as the test set.

4.2. Metrics used

A classification report for a given classifier would provide us values for Precision, Recall and F1-Score. Before going into deciding which classifier is best, it is important that we understand these aforementioned metrics. Figure: 2 shows us the example of a confusion matrix which would help us understand these metrics better.

Figure: 2 – Confusion Matrix



	Predicted: NO	Predicted: YES
Actual: NO	TN	FP
Actual: YES	FN	TP

True positive or TP: When the actual value and predicted value both belong to 'logic 1' or 'class 1'.

False positive or FP: When the actual value belongs to 'class 0' and the predicted value belongs to 'class 1'.

False negative or FN: When the actual value belongs to 'class 1' and the predicted value belongs to 'class 0'.

True negative or TN: When the actual value and predicted value both belong to 'logic 0' or 'class 0'.

1) Precision

Precision is given by the ratio of true positives to the sum of true positives and false positives.

Precision = TP / (TP + FP)

2) Recall

Recall is given by the ratio of true positive to the sum of true positives and false negatives.

Recall = TP / (TP + FN)

3) F1-score

F1- score is given by the **harmonic mean** of Precision and Recall i.e.

F1- score = 2 * Precision * Recall / (Precision + Recall)

4.3. Before sampling

Classifier Model	Class	Precision	Recall	F1-score
XGBoost				
	0.0	0.80	0.99	0.89
	1.0	0.68	0.04	0.08
Random Forest				
	0.0	0.82	0.94	0.88
	1.0	0.46	0.22	0.29
Decision Tree				
	0.0	0.83	0.84	0.84
	1.0	0.36	0.35	0.36
Extra Trees				
	0.0	0.82	0.83	0.83
	1.0	0.32	0.31	0.31

Now that we can understand the different metrics and that we have the values in front of us, we can choose the best classifier for our dataset. But before that, we have to evaluate the different metrics and choose as to which takes precedence over the other. This leads us to the big question: Precision vs Recall.

Precision gives us a measure of how often or how accurate our model has been when our predicted class is '1' or '0'. Recall gives us a measure as to how accurate our model has been when our actual class is '1' or '0'. Since for this problem, we are more concerned about the flight being delayed rather than the flight not being delayed, we are considering Precision and Recall for ‘class 1’ while ranking our classifier. Imagine a problem for cancer prediction, in such a case, it is extremely important that our model does not classify a patient actually having cancer to be free from cancer. In that case, the cancer in that patient goes unnoticed and the repercussions are catastrophic. Hence, we can say that our model should have a lesser number of **‘False negatives’**. Since we have to take this under consideration, we prefer **Recall** to be a better evaluation metric. However in our case, it is equally catastrophic for a flight to have been predicted delayed when no delay is incurred and for a flight to have been predicted that it would reach on time when a delay is incurred. So, we are considering **F1-score** to be the plausible metric to evaluate different classifiers.

Amongst the classifiers we have now, **Decision Tree Classifier** has the best F1-score, hence it is the best classifier for our model.

Note that we have not considered accuracy to be a plausible metric because of our dataset. Figure: 1 reveals that there are about four times the number of zeros as there are number of ones. This means that even if our model blindly predicts that our flight would not be delayed, it would still be 80% accurate as 4 out of 5 flights do reach on time. This doesn’t help us in any way. This is the effect of **“Class imbalance”** in our dataset.

5. Sampling

In order to eliminate this problem of Class imbalance, we adopt sampling methods. These methods are used to replicate or discard records of data so as to reduce the skewness of the dataset. **Oversampling** refers to the process by which records belonging to the minority class are replicated in order to establish approximately equal number of records in the two classes. **Undersampling** refers to the process by which some of the records belonging to the majority class are discarded so as to bring about approximately equal number of records in the two classes. Another type of sampling, and probably the most sophisticated one is **SMOTE** (Synthetic Minority Oversampling Technique). This method produces “Synthetic” samples and offers better datapoints as opposed to Random oversampling and undersampling techniques.

6. After Sampling

Classifier Model	Class	Precision	Recall	F1-score
XGBoost				
	0.0	0.78	0.89	0.83
	1.0	0.88	0.74	0.80
Random Forest				
	0.0	0.82	0.91	0.86
	1.0	0.90	0.80	0.84
Decision Tree				
	0.0	0.82	0.83	0.82
	1.0	0.82	0.82	0.82
Extra Trees				
	0.0	0.74	0.75	0.75
	1.0	0.75	0.73	0.74

Here, Random Forest Classifier offers the best F1-score for ‘class 1’ so we can choose this as the best classifier for our model. Refer section 4.3. for detailed explanation as to why we have chosen this metric to evaluate classifiers.

7. Regression

We move onto the next stage in our predictive analysis model which is to predict the delay (in minutes) on arrival. Here, we are only considering the records in the dataset where there exists an arrival delay (ArrDel15 = 1). The data is divided based on the 80-20 rule where 80% of the data is used as the training set and 20% of the data is used as the test set.

7.1. Metrics used

There are two main metrics used in order to evaluate regressor models namely:

- Mean Absolute Error (MAE)
- Root Mean Squared Error (RMSE)

Figure: 2 shows the formulae for the above.

Figure: 3 – Formulae for MAE and RMSE

$$MAE = \frac{1}{N}\sum_{i=1}^{N} |y_i - \hat{y}|$$

$$MSE = \frac{1}{N}\sum_{i=1}^{N} (y_i - \hat{y})^2$$

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N} (y_i - \hat{y})^2}$$

7.2. Scores observed

Regressor	MAE	RMSE
XGBoost Regressor	42.1378	71.9727
Extra Trees Regressor	45.6255	76.9113
Linear Regressor	42.5839	72.4776

7.3. Evaluation of metrics

Here the two main metrics we are considering are Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). Note that the MAE and RMSE are mentioned in minutes, so while choosing our Regressor we have to choose the one which involves the lower values of MAE and RMSE as we are looking to reduce the delay. Now amongst MAE and RMSE, we can choose to give one metric a higher precedence. RMSE is considered to be a more stringent metric than MAE as it penalizes the difference in delay between the actual and predicted values. This can be observed by looking at the formula for RMSE given in Figure: 3. Since XGBoost Regressor has the lowest RMSE score here, **XGBoost is the best regressor for our model.**

8. Analysis of Arrival Delay values

Earlier, we have tabulated our scores in the general case of predicting all the arrival delay values in our test set. However, if we look at our distribution of arrival delay values in Figure: 4a and 4b, we can observe that most of our values are concentrated within a specific range, also called as the **Interquartile range** of our dataset.

Figure: 4a – Distribution of Arrival delay values using density plot

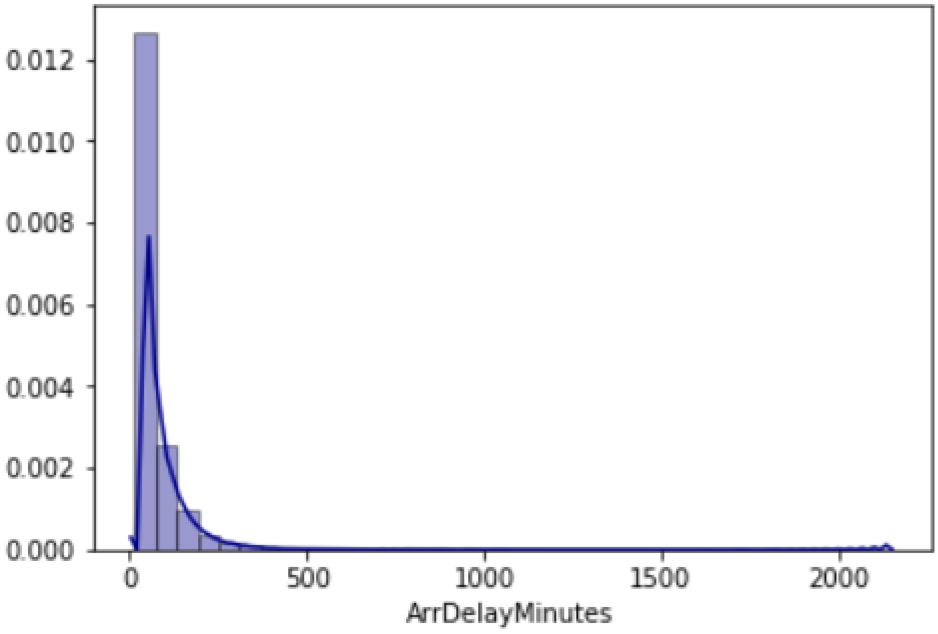
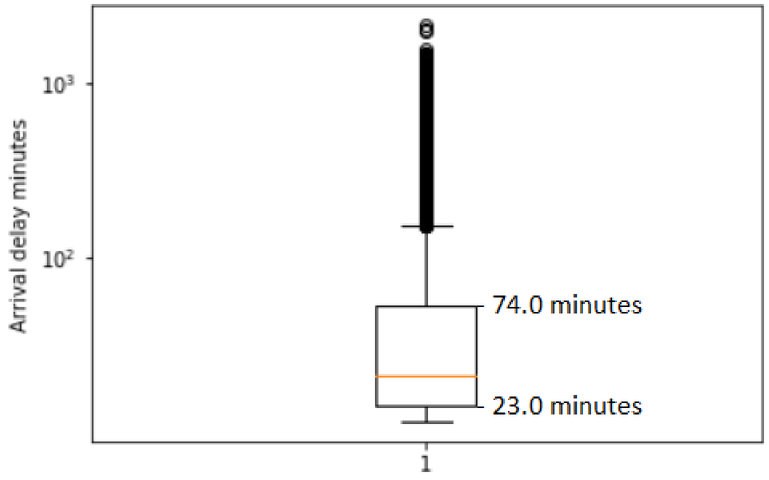


Figure: 4b – Distribution of Arrival delay values using box plot



The region contained within the box is the interquartile range of our dataset and we can conclude that our delays are concentrated in the region where there are between 23.0 and 74.0 minutes (25% and 75% percentiles respectively).

Therefore realistically, the main use case of our model would be to predict arrival delay values within the interquartile range. On scoring the values encompassed in this area, we observe a significant decrease in both our MAE and RMSE. We are choosing XGBoost because that was our best regressor. Hence, we can say that our model performs **significantly better** in the region where most of our data is concentrated. (Refer Table: 4)

Table: 4 – Score of XGBoost regressor predicting in IQR

Regressor	MAE	RMSE
XGBoost Regressor	22.8693	26.7071

We have obtained an MAE of **42.1378** for our entire dataset and a value of **22.8693** in the Interquartile range. Since most of our arrival delays would fall in the Interquartile range, we shall consider 22.8673 to be the value used for comparison. In our dataset, we have considered only delays spanning greater than 15 minutes and those below 15 minutes are neglected. If that can be neglected, a prediction error around that range could also be neglected to some degree. We have obtained an MAE of 22.8673 which is not exactly close to 15 minutes but it is good enough considering we have no other knowledge of the flight. However if we include the departure delays of the flight and use our model after the flight has taken off, we would probably observe lower error values and our model would be substantially better.

9. Results and Conclusion

Our machine learning model works in two stages. The first stage classifies as to whether or not our flight would be delayed on arrival. We have obtained a score of 0.84 using **Random Forest Classifier** after oversampling our data using SMOTE. The second stage of our model predicts the delay after which the flight arrives at the airport. Here, using **XGBoost Regressor** we were able to achieve an MAE of 42.1378 and a RMSE of 71.9727 and an MAE of 22.8693 and a RMSE of 26.7071 in the Interquartile range of our dataset. Therefore, our two stage model is successful in predicting the delay of the flight on arrival to a fair degree.

10. Annexure

Although the above information provides us with a plausible amount of knowledge about our model, it is not realistic. In reality, the two stages, classification and regression are pipelined. The model predicts as to whether the flight is going to be delayed on arrival and **if so** it will predict the arrival delay for the flights being delayed. Earlier, while evaluating our regressor, we have assumed that all our records are classified properly but that is simply the ideal scenario. To evaluate our realistic scenario, we have chosen our best classifier, Random Forest Classifier and our best regressor, XGBoost regressor. The results are tabulated for the regression stage as shown below.

Regressor	MAE	RMSE
XGBoost Regressor	53.9345	67.3844