Application of Machine Learning for predicting Flight Delays

Lalit Saini
Second Year Undergraduate, Electrical Engineering
Indian Institute of Technology, Bombay
180070030
180070030@iitb.ac.in

Abstract

Task is to predict the flight status given a set of information. Predicting flight delays can be useful to a variety of organizations: airport authorities, airlines, aviation authorities. At times, joint task forces have been formed to address the problem. Such an organization, if it were to provide ongoing real-time assistance with flight delays, would benefit from some advance notice about flights likely to be delayed. Using algorithms of the Machine Learning we can learn some differences between the case when the flight is delayed or is ontime. After learning the differences we can use the differences to predict the result given any unseen condition.

1 Problem Statement

In the problem statement we are given a dataset, task was to create a model to predict the unseen cases. The dataset is taken from website of Bureau of Transportation statics (www.transtats.bts.gov). Our data consist of all flights from the Washington, DC area into the New York City area during January 2004. The percent of delayed flights among these 2201 flights is 19.5%. Interest is to come up with a decision if the flight is delayed or it is ontime. I have used python as programming language and libraries like sklearn, pandas for modelling and processing the data. I divided the randomly shuffled data in two parts of size ratio **60:40** to act as training and test set for a model. The algorithms used are linear svm, logistic regression, and decision tree classifier. The given data contains following information about a flight flight carrier, origin, destination, departure time, reserved departure time, weekdays, day of month, flight date, distance of flight, flight number, weather related delay. Main aim was to compare model using same set of feature and then choosing suitable features to predict the results.

2 Data Description and Features

The data was given in the csv format. I used **pandas** for processing the data. The data head is given in the Figure 1. Next part was to find that wheter

Figure 1: Data Head

	CRS_DEP_TIME	CARRIER	DEP_TIME	DEST	DISTANCE	FL_DATE	FL_NUM	ORIGIN	Weather	DAY_WEEK	DAY_OF_MONTH	TAIL_NUM	Flight Status
0	1455	OH	1455	JFK	184	01/01/2004	5935	BWI	0	4	1	N940CA	ontime
1	1640	DH	1640	JFK	213	01/01/2004	6155	DCA	0	4	1	N405FJ	ontime
2	1245	DH	1245	LGA	229	01/01/2004	7208	IAD	0	4	1	N695BR	ontime
3	1715	DH	1709	LGA	229	01/01/2004	7215	IAD	0	4	1	N662BR	ontime
4	1039	DH	1035	LGA	229	01/01/2004	7792	IAD	0	4	1	N698BR	ontime

the given data contains some null values of not. On analzying that I found out that the dataset contains **no null** value. This saves our work of assigning dummy entries at the place of null values. Most of the data is categorical and that too giving in form of a string. Next task was to assign an integer value to each category. This assignment was done for features like 'CARRIER', 'ORIGIN', 'DEST' and ofcourse 'Flight Status'. For assignment of an integer I first checked out the unique entries in a feature column that act as my total number of categories. Then using this number each category in alphabetical order was replaced by an integer lower than the number of unique entries in a column. Assignment made were as follows:
CARRIER: {'RU': 6, 'DL': 2, 'CO': 7, 'DH': 1, 'OH': 0, 'US': 5, 'MQ': 3, 'UA': 4}
ORIGIN: {'BWI': 0, 'IAD': 2, 'DCA': 1}
DEST: {'JFK': 0, 'EWR': 2, 'LGA': 1}
Flight Status: {'ontime':0, 'delayed':1}
Features like flight date, tail number , flight number were rejected because they analytical makes no sense to affect the flight status.
I added a new feature capturing the difference between the CRS departure time and departure time. The feature is meant to see how delaying is related with the time difference of departure from the origin airport.

For visulaising the data I used matplotlib.pyplot and seaborn libraries.

Figure 2: Carrier based delay fraction

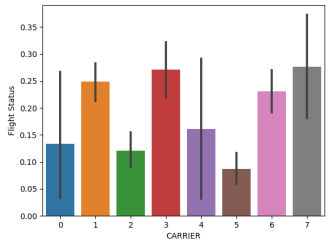


Figure 3: Orgin of flight based delay fraction

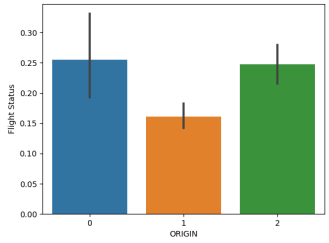


Figure 4: Destination of flight based delay fraction

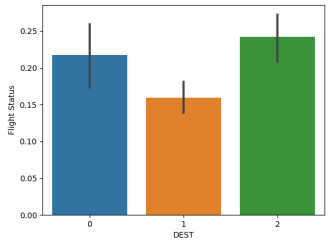


Figure 5: Weekday based delay fraction

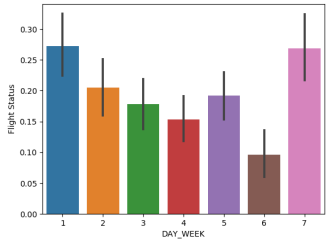


Figure 2-9 shows the data visualization with y-axis as the delay fraction (no of delays/examples of particular entity).
As we can see that weather based delay is a very string feature. If there is some weather related issue there is flight delay that can be seen from the bar plot having 1 as fraction of delays. The departure time difference takes shape similar to normal curve with highest delay fraction corresponding to the peak, this points out to the fact that this time difference can be modelled.

Figure 6: Month day based delay fraction

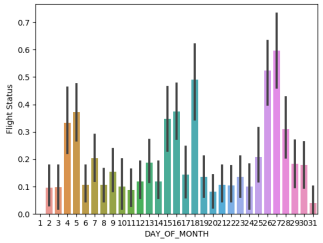


Figure 7: Weather delay based delay fraction

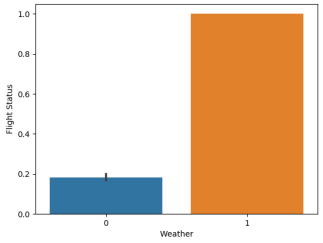


Figure 8: Time based delay fraction histograms

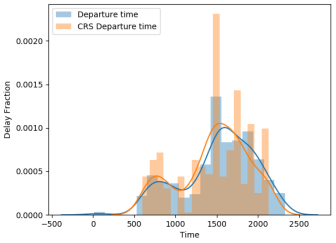
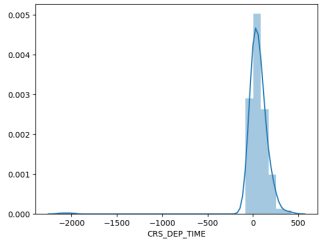


Figure 9: Time difference based delay fraction histogram



3 Methods

To start with I choosed origin, destination, carrier, timedifference, departure time, weekday, weather

as my features. After randomly shuffling the dataset I divided it into 60:40 ratio to use as my training and test set.
To start with I fit the data with all the above features and then choose top 4 most important features. Using these four I fit the algorithms like Logistic Regression and SVM on it.

3.1 Decision Tree

The main idea behind decision tree algorithm is to create a upside down tree like structure with a root node and terminated with leaf nodes making a yes or no decision. The tree is provided with a input, each node ask a yes/no question and pass the input accordingly to the leaf node. The main motivation behind using decision tree for this data was that the weather is acting as a strong feature to distribute the data. The challege is to build a tree and choose which node will ask which question. This is taken by analysing the information gain due to node. For the algorithm entropy is used as a measure of information gain.

$$Entropy = -p_+ \log_2 p_+ - p_- \log_2 p_-$$

$$Gain(S, A) =$$

$$Entropy(S) + \sum_{\forall\, V \in\, values(A)} p(V) Entropy(V)$$

3.2 Logistic Regression

Logistic regression is simple algorithm that uses the given hypothesis:

$$h_\theta(x) = \frac{1}{1 + e^{-\theta^T X}}$$

where $\theta^T X = \theta_0 + \sum_{j=1}^{n} \theta_j x_j$. The parameters are determined by minimising the given cost function and using gradient descent:

$$J(\theta) =$$

$$-\sum_{i=1}^{m} y^{(i)} \log h(x^{(i)}) + (1 - y^{(i)}) \log(1 - h(x^{(i)}))$$

3.3 Linear SVM

In the algorithm we try to find a hyperplane to separate two or multiple classes from each other. The hyperplane is chosen such that the nearest data points from the classes have maximum distance from it. It uses hinge loss function given as:

$$max(0, 1 - y_i(\vec{w}.\vec{x_i} - b))$$

I used soft margin SVM in which aim was to minimize,

$$\left[\frac{1}{n}\sum_{i=1}^{n} max(0, 1 - y_i(\vec{w}.\vec{x_i} - b))\right] + \lambda||\vec{w}||^2$$

4 Results and Discussion

Using my feature vector as :
X=[carrier, origin, dest, timediff, weekday, weather]
On fitting the data to **Decision Tree**, I obtained accuracy of **87.40%** with **0.802** f1 score. The top 4 most important feature obtained were :

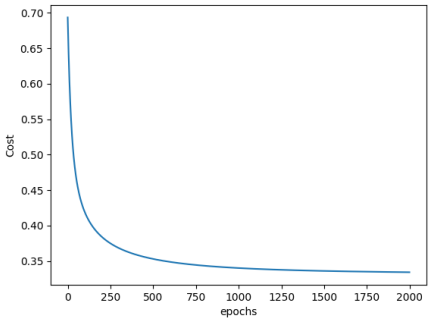
Feature	Importance Score
timediff	0.786
weekday	0.089
carrier	0.079
dest	0.024

I tried to fit the other algorithms on this feature vector but the model was having very high variance. This calls for reduction in features. Now keeping the feature vector as only the top 4 features we obtained,
X=[carrier, dest, timediff, weekday, weather]
Time difference comes out to be the most important feature becuase of the fact after a certain time delay a flight can be considered as delayed. As we can see from the Figure 5 that for a day the fraction of flight delays quite low as compared to others. This implies that for that day the flight delay probability is less, thus it also become as important feature. Similarly for some carrier chances of getting delayed is less, so that also become an important feature. I included weather in the new feature vector because it is very strong feature.
I again fit the data to the **Decision Tree** and obtained accuracy of **88.08%** with **0.806** f1 score. Now using the same feature vector I trained other algorithms.
Before procceding to other algorithm I concatenated **x0**, a vector having all entries as 1 and shape numTrain$\times$1.
Now the features like carrier, destination and weekday which are categorical need to be converted to **one-hot vector**. So basically the carrier feature becomes a vector of length 8 where each position indicates a carrier. Same treatment for destination and weekday vector. The timediff vector is converted in the one hot vector by

dividing the values in bins of size 25. So finally my feature vector is of length 38.
I trained Logistic Regression algorithm with learning rate=**0.06** and regularisation parameter=**0.065**, obtained accuracy of **73.29%** with **0.817** f1 score. Figure 10 shows the cost history trend.

I also used sklearn's Logistic Regression algorithm to train on this data and got an accuracy of **92.28%** with **0.855** f1 score.

Figure 10: Cost history for Logistic Regression

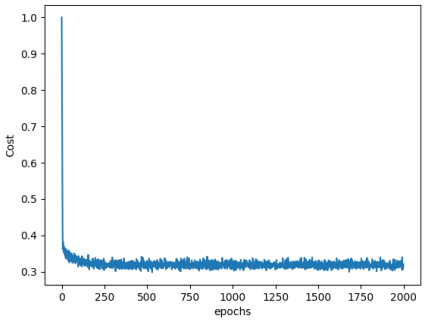


One seeing the weights associated with the features I found that, for carrier weight corresponding to **MQ (American Eagle)** is highest and that with **CO (Continental)** is lowest, for destination **LaGuardia (LGA)** has lowest weight rest two have similar weight, for time difference more than 125 minutes the weight is positive and high being highest for difference of 125 minutes and for weekday **Monday** has highest weight and **Thrusday** has the lowest weight. As expected I got height of all weight as the weight for **Weather**. Highest weight here implies that the factor is having highest affect in delaying the flight. So we can say that if you are going to LaGuardia with American Eagle on monday then your flight is more likely to be delayed.

On training Linear SVM with learning rate=**6e-2** and regularization parameter=**0.105**, obtained accuracy of **90.46%** with **0.824** f1 score. Figure 11 shows the cost history trend.

I also used sklearn's Linear SVM algorithm to train on this data and got an accuracy of **90.12%** with **0.806** f1 score. My model come out to be performing better than sklearn algorithm because my no. of iterations are greater than that of algortihm.

Figure 11: Cost history for Linear SVM



Predicting the best conditions to not getting delayed when travelling from **DC(DCA)** to **New York(JKF)**, I found out following conditions, carrier should be **CO (Continental)** on **Thrusday**, around **1600** having no weather related delays.

5 Conclusion

For dealing with large features we must make sure that the model is not having high variance. If so then reduce the number of features to make the model less complex and thus reduce the variance. Good way to choose features to use is to visualise them see if there is any feature in which target is nearly equal for all values if so then reject such features because they are not going to play any role in the decision making. Reducing features and measuring model accuracy is good way to select relevant features.

Answer to bonus questions

1. Veronica, Karen
2. Data Processing Inequality: For three random variables forming Markov chain say, $X \to Y \to Z$, and **I(a,b)** be mutual information between a and b then,

$$I(X;Y) \geq I(X;Z)$$

implying that conditional distribution of Z depends only on Y and is conditionally independent of X.
3. Rule of two
4. C3-PO and R2-D2
5. The team at Cards Against Humanity created an AI to write cards and there was competition between the algorithm and human writers to write cards.