CS229 Final Report: Modeling Flight Delays

Abstract

This work uses flight information and weather data to predict whether a flight will be delayed by more than 15 minutes. We collected publicly available flight and weather data to make these predictions on the 40 largest airports in the United States. Since a flight's delay can depend on a previous flight's delay, we added features to capture these second-order behaviors. Using Random Forest, Gaussian Naive Bayes, Logistic Regression, and Neural Networks, we classified the data and achieved a best overall F1-score of 82% using a Random Forest classifier.

1. Introduction

The idea of working on flight delays came from one member of our team who has his pilot license and often flies on his own. In his experience weather is an important factor. For larger, commercial airplanes, weather can still be a cause of disturbance, yielding increased delays and costs for airlines and customers: every year 23% of the flights are delayed or canceled, costing 8 billion dollars to airlines, and 20 billion dollars to travelers in lost time and money. Modeling delays is a first step in solving them.

In order to model flight delays, the input of our algorithms is a vector of features x regarding the flight (temperature, departure airport, plane model, etc.). We used Random Forests, Naive Bayes, Neural Networks and Logistic Regression. The output is a scalar (0 or 1) that predicts, at the time of departure, whether or not a flight will be delayed by more than 15 minutes at its arrival. Concretely, it's a binary classification problem, where 0 is ”non-delayed”, and 1 is ”delayed”. We only consider domestic flights in the US, during the year 2008, restricted to the 40 most important airports. Those restrictions allow us to avoid low-traffic airports that do not generalize well, and still capture a good share of the total number of flights (around 50% of total flights in the year 2008). Reducing the size of datasets also helped improve training times. We tried to find datasets to account for each of the 5 causes of delays listed by the Department of Transportation: Weather, National Airspace System (e.g. congestion due to holidays), Carrier (e.g. technical issues), Security (e.g. long security lines), Aircraft arriving late (snowball effect). For instance, we constructed new features to take into account the fact that some planes arriving late at an airport will entail other planes leaving late this same airport. We used F1 score as our main evaluation metric, keeping an eye on global and class-wise recall and precision metrics because of the imbalance of the dataset. We also conducted error analysis, monitoring the overfit of our model.

2. Related work

This problem has been tackled previously in this course by several teams. Most of them had a similar approach but we were able in the end to yield better results thanks to a more comprehensive set of features.

- Dieterich Lawson and William Castillo, in their project ”Predicting Flight Delays” in 2012 (Lawson & Castillo, 2012), used the same dataset of flights but included several years, which resulted in an impressive number of data points (135 million of flights). However they limited their features to weather data only (33 features only in the end) obtaining 40% recall only.
- Raj Bandyopadhyay and Rafael Guerrero, in their project ”Predicting airline delays” in 2012 (Bandyopadhyay & Guerrero, 2012), focused only on one airport and one airline company for 2 years, and only took into account weather data. Even though they were able to achieve a good score, their model is restricted to a very small subset of data and may not generalize well to other airlines and airports.

We also found several research papers trying to model flight delays:

- ”Modeling Flight Delays and Cancellations in the US” by researchers from the NASA and the FAA (Wang et al., 2009). They focused on weather related delays but used more sophisticated features such as key airspace metrics and also took into account the delay of previous flights to predict a new delay. They also showed that Neural Networks tend to perform better than ”regular” classifiers on this problem due to their ability to grasp the complex structure of the relations.
- ”Analysis of aircraft arrival and departure delay characteristics” by researchers from the NASA analyses

what are the main causes of delays (E.R.Mueller & G.B.Chatterji, 2002). They provide good insights on which feature to consider to better predict flight delays: for instance origin airport and hour of departure are very relevant features.

- ”Characterization and Prediction of Air Traffic Delays” by researchers at MIT used both classification and regression to attack this problem (Rebollo & Balakrishnan, 2014). Their classification was done using a Random Forest classifier, with an absolute test error of 19% on quite small sets (training + test = 4000 samples).

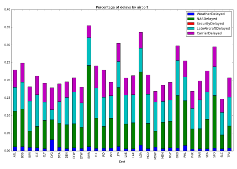
3. Datasets and features

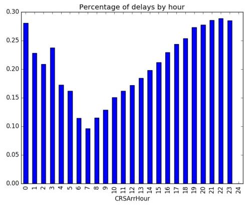
Our main dataset includes every domestic flight from in 2008, with scheduled times, actual times, airports (departure/arrival), delay (if any) and cause of delay, plane identifier. After cleaning (selecting only major airports), there are about 3.2M rows left. We used a 20/80 split.

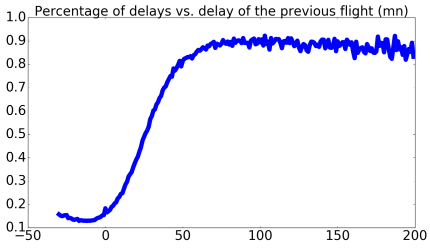
Data references are available here: (ASA, 2009), (Network, 2012), (Database, 2016), (Administration, 2008), (Administration, 2016).

3.1. Exploratory Data Analysis

In order to get a qualitative understanding of the different factors of delays, we did some exploratory data analysis. In our dataset, delays are given in minutes, but the regression problem proved to be extremely hard, so we only use two classes: delayed (more than 15mn delay) and non-delayed. About 23% of the flights in 2008 are delayed. The graphs below display interesting non-linear dependencies between delays and several features.

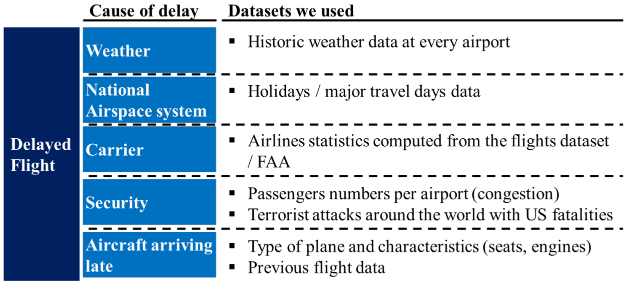






3.2. Additional Datasets

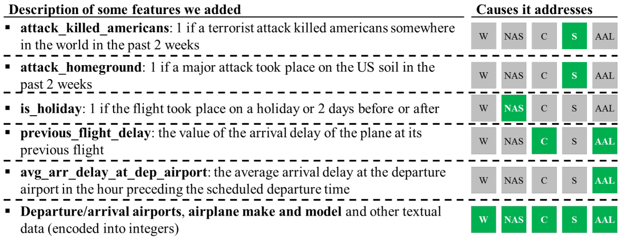
Our approach to this problem was to create new features for each flight. For each kind of delay, we selected a ”proxy” dataset: for instance, we think that a good proxy for security delay is to look at the terrorist attempts on American ground or with American casualties as they may entail reinforced control at the airport.



	Cause of delay	Datasets we used
Delayed Flight	Weather	▪ Historic weather data at every airport
	National Airspace system	▪ Holidays / major travel days data
	Carrier	▪ Airlines statistics computed from the flights dataset / FAA
	Security	▪ Passengers numbers per airport (congestion) ▪ Terrorist attacks around the world with US fatalities
	Aircraft arriving late	▪ Type of plane and characteristics (seats, engines) ▪ Previous flight data

3.3. Feature engineering

From each dataset, we were able to extract some relevant features for each flight. Below are some examples of features we selected, and the cause of delay they tackle. For instance, to model the congestion of an airport due to a high number of passengers, we used the dates of holidays and special national days (e.g. Valentine’s day). Some very important features we built were those designed to ”force the memory” of our classifier. Also, for a given feature X (e.g. number of passengers at the airport), sometimes we added not only X but also X^2. These 2 methods of feature engineering will be more detailed in the Method section.



Description of some features we added	Causes it addresses				
▪ **attack_killed_americans**: 1 if a terrorist attack killed americans somewhere in the world in the past 2 weeks	W	NAS	C	**S**	AAL
▪ **attack_homeground**: 1 if a major attack took place on the US soil in the past 2 weeks	W	NAS	C	**S**	AAL
▪ **is_holiday**: 1 if the flight took place on a holiday or 2 days before or after	W	**NAS**	C	S	AAL
▪ **previous_flight_delay**: the value of the arrival delay of the plane at its previous flight	W	NAS	**C**	S	**AAL**
▪ **avg_arr_delay_at_dep_airport**: the average arrival delay at the departure airport in the hour preceding the scheduled departure time	W	NAS	C	S	**AAL**
▪ **Departure/arrival airports**, **airplane make and model** and other textual data (encoded into integers)	**W**	**NAS**	**C**	**S**	**AAL**

3.4. Preprocessing

We leveraged the Python library Pandas to perform preprocessing on our dataset, such as parsing dates and text, then merging the datasets into one. For instance, we reinforced the weather dataset by replacing missing/erroneous data by data taken from closeby measurement stations. Using the FAA registry, we added several features related to plane (age, model, make, number of seats) using the identifier as a join key. We also handled missing features in various columns by replacing them with averages or medians, or dropping them when the replacing strategy did not make sense. We also used one hot encoding on text data (labels), such as carrier name or hour of departure, to convert them to boolean columns. We also standardized the data (mean-removal and variance scaling) for linear models. We considered using data augmentation via SMOTE (Synthetic Minority Over-Sampling Technique), building artificial samples of the underrepresented class using nearest neighbors, but results did not improved.

At the end of the preprocessing, we had our final dataframe, consisting of 3 million flights, with 152 features per flight.

4. Methods

Because of the imbalance of our dataset, reported scores are usually very high by default. Hence, we built a baseline: a dummy classifier that labels every data point as non-delayed (the main class). This allows us to get a better understanding of the relative performance of our algorithms.

The Exploratory Data Analysis showed that the problem is extremely non-linear. As such, the first algorithm we tried was Decision Trees, which recursively does binary splits of the feature space along one dimension at a time to predict classes. The leaves of the tree are an assignement to one of the classes. At each iteration, the goal is to pick the feature and the splitting point along this feature axis that will minimize the impurity (measured using the Gini impurity metric). For n classes, let f_i be the percentage of points labeled i:

$$I_G = \sum_{i=1}^{n} f_i(1 - f_i) = f_0(1 - f_0) + f_1(1 - f_1)$$

The main issue of decision trees is overfitting, which can be corrected using an ensemble/boosting method called **Random Forest**. It trains several Decision Trees on random subsets of the training data, and averages their predictions.

The **logistic regression** works as follows. We use a hypothesis

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

$$P(y = 1|x, \theta) = h_\theta(x)$$

Stochastic gradient descent is used to fit the model (it's faster than solving the equations on large datasets) by maximizing the log likelihood, which yields the update rule on sample i with learning rate α

$$\theta := \theta + \alpha\big(y^{(i)} - h_\theta(x^{(i)})\big)x^{(i)}$$

The monotonicity of the sigmoid function places an assumption of monotonicity on our features. By plotting features vs delays (our target variable), we can evaluate the monotonicity of each feature, and assess the need for feature engineering. Saturation and correlation between features should also be avoided for this type of linear models. Improving our features (using feature engineering and feature selection) allowed logistic regression to give good results. Hence, we added this algorithm to our selection, using weights for the cost function to penalize errors in the "delayed" class more, hence avoiding imbalance caveats.

Naive Bayes is a generative algorithm based on the strong assumption of independence between features. For discrete features, given feature vector x, we want to calculate (using Bayes formula) for $k \in 0, 1; p(y = k|x) = \frac{p(x|y=k)p(y=k)}{p(x)}$. Using the independence assumption, this boils down to estimating the $\phi_{i|y=k} = p(x_i|y = k)$ and $\phi_y = p(y = 1)$ by computing those probabilities as observed frequencies over our training data. Gaussian Naive Bayes is an extension of this model to the continuous space, assuming a normal distribution for $\phi_{i|y=k}$.

Leveraging the library Tensorflow, we experimented several shapes of fully connected **neural networks**. The idea is that each neuron combines the outputs from all neurons at the previous layer using a weight matrix W, and compares it to a bias b to trigger its output. Biases and weights are learned using the backpropagation algorithm which allows to compute gradients of the global cost function with respect to any bias/weight of the network, hence allows to use gradient descent. We settled on 2 hidden layers, respectively with 50 and 20 neurons, using rectified linear activation functions (basically thresholding the activation of the neuron to 0). We use a softmax output layer (with 2 neurons, 1 for each class), which transforms the final activations z into a distribution of probability

$$f(z)_j = \frac{e^{z_j}}{\sum_{i=1}^{n} e^{z_i}}$$

One crucial element of our dataset that those models don't take into account is the **time dependence**. Data points are not independent. For instance, the delay of a flight on plane P will likely impact the punctuality of the following flight of plane P. Similarly, the arrival delay can be caused by the congestion due to several airplanes arriving at the same time at the airport. Hence, our models need to be aware of the other datapoints (as long as they are not in the future) as features. For that, we use features such as the delay of the 1st, 2nd, etc. previous flights on the same day. Empirically, delays do not propagate through night: because there are very little domestic flights at this time, carriers can use night to "reset" their delays. We also include the number of flights scheduled for departure and arrival at the same airport. This is a way to force our model to develop "memory" without resorting to a more intricate model (LSTM was our initial goal). This is an implicit way of representing this time series, capturing only the features that actually persist from one point to one other.

Lastly, we tried **boosting** on the several weak classifiers we had, but this approach did not yield good results.

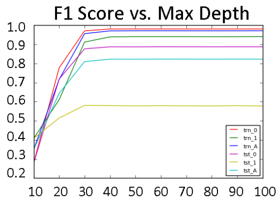
5. Experiments/Results/Discussion

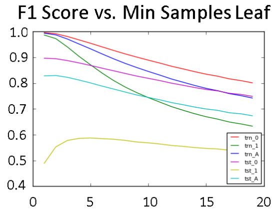
5.1. Experiments

To ensure that we are correctly using the set of algorithms for solving this problem, we chose to explore some of the parameters of each of the methods. For example, Random Forest has a number of interesting parameters: maximum depth of the tree, number of trees in the forest, maximum number of features to consider when looking for the best split, etc. We plotted F1-score against each of these parameters to ensure we are using an optimal configuration for Random Forest, some of which are shown below. In each of the following plots, ”trn_X” is the training data F1-score on X, where X=0 is the non-delayed class, X=1 is the delayed class, and X=A is the aggregate/total F1-score of the two classes. There is a similar nomenclature for ”tst_X”, where the data is the test data.

Here we see that the max depth of the tree must be larger than 40 to ensure that the F1-score is sufficiently high.

Here we see that the optimal minimum samples per leaf value is 3.



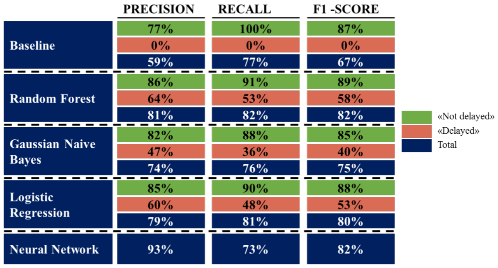


Here are the lessons we learned from exploring parameters of Random Forest, Gaussian Naive Bayes, and Logistic Regression:

- Random Forest's maximum tree depth should be larger than 40, which is close to the number of features used (37).
- Random Forest's min samples required to be a leaf node should be 3. Too small and the leaf node doesn't have enough children, to large and each node has too many children to compare.
- Random Forest's max features should remain unconstrained, the algorithm has a search feature that allows it to estimate how many features it should consider for any given leaf.
- Gaussian Naive Bayes's prior should be skewed 9 to 1, to help compensate for the dataset class imbalance.

5.2. Results

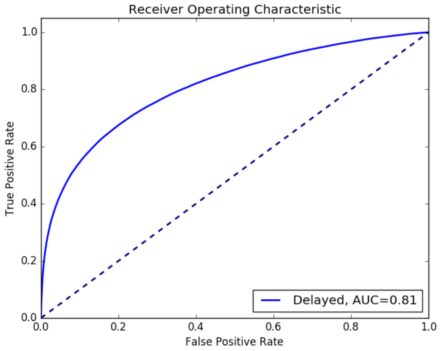
Because of the class imbalance we observe in our dataset, we cannot use accuracy as a metric. Instead, we focus on precision, recall, and most importantly the F1-score, which is the harmonic mean between precision and recall. In the following table we provide all of these metrics for delayed, not-delayed, and total, across each of the algorithms we used, as well as the baseline predictor (predicting all flights as not-delayed). Random Forest achieves the best total F1-score of 0.82. We used K-fold cross validation for final results reporting, using 5 folds (we have a dataset large enough to allow it).



		PRECISION	RECALL	F1 -SCORE
Baseline	🟩	77%	100%	87%
	🟥	0%	0%	0%
	■	59%	77%	67%
Random Forest	🟩	86%	91%	89%
	🟥	64%	53%	58%
	■	81%	82%	82%
Gaussian Naive Bayes	🟩	82%	88%	85%
	🟥	47%	36%	40%
	■	74%	76%	75%
Logistic Regression	🟩	85%	90%	88%
	🟥	60%	48%	53%
	■	79%	81%	80%
Neural Network	■	93%	73%	82%

🟩 «Not delayed»
🟥 «Delayed»
■ Total

Here is the ROC curve for the delayed class. Coincidentally, the AUC value for the delayed class is 0.81.



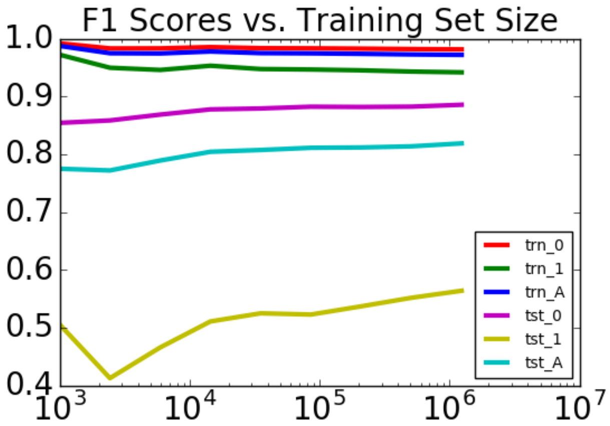
5.3. Discussion

There are two important comparisons to make. The first compares complex and simple non-linear models. The second compares non-linear and linear models.

Random forest and Neural networks are complex non-linear models that can capture dependencies between features. Gaussian Naive Bayes assumes independence between features, which is clearly not true for our problem. It is primarily this difference that allows Random Forest and Neural Network to outperform Gaussian Naive Bayes.

Random Forest can capture behavior of non-linear features, but Logistic Regression can only capture behavior of linear features. We were able to achieve good performance with Logistic Regression only because we converted our given non-linear features into a larger number of linear features (as mentioned in the dataset and features section).

In every algorithm we use, the performance on the training set significantly beats the performance on the test set. One way to diagnose the causes of this supposed over-fitting is to plot the F1-score versus the training set size (in this case, for Random Forest). The yellow curve is the F1-score on the delayed class, which noticeably increases as we run the algorithm with more training data.



This indicates we have not saturated our possible performance on the delayed class. Since the delayed class only represents 23% of our total features in our dataset, the low number of samples on this class necessitates getting more data overall so that the algorithm can better learn how to predict the delayed class. Although we only focused on one year's worth of data on these airports to keep the computation feasible, we in fact had access to 9 more years of flight and weather data. If we added samples from the other 9 years of data to our model, we would expect to improve our results.

Since 1 year contained 3 million samples, 10 years would have 30 million samples. Since our x-axis is logarithmic, and the delayed class continues to improve its F1-score at the same slope, we would expect incorporating all of the other data to increase our F1-score from 0.58 to 0.64.

Also, suppose instead of truncating to use only the 40 largest airports, we included all flights from all airports in the US. Taking these flights, over 10 years, would provide 70 million flights, which would improve our F1-score even more to 0.67.

6. Conclusion/Future Work

We were able to use publicly available data, intuition, and machine learning algorithms to predict whether a flight will be delayed by more than 15 minutes with an F1-score of 82%, using Random Forests or Neural Network. Those algorithms perform well because they are able to capture the underlying complexity of our problem, provided that we can control for overfit. We have not been able to find a better prediction result in the literature for similar projects. The three major sources of improvement we implemented were: aggregation of several weather sources to get complete and accurate data, careful tuning of models meta-parameters, feature engineering to capture time dependence. Our future work includes four different directions:

- Generate new features that capture the nuances of the co-dependence between a flight's delay and the next flights delay (as of now, we only account for first-order time dependencies, but the interactions in reality are more complex).
- Use additional features (twitter data, finer resolution terror data, etc.) to account for factors that contribute to delays but are invisible by our algorithms as of now due to the absence of proxy variables.
- Improve our neural network. Due to lack of time, we could not explore as much as we wanted to neural network approach, but results already were promising using only out-of-the-box algorithms.
- Use more of the already available data to improve our estimates on the delayed class.

References

Administration, Federal Aviation. Passenger boarding (enplanement) and all-cargo data for u.s. airports - previous years, 2008. URL `https://www.faa.gov/airports/planning_capacity/passenger_allcargo_stats/passenger/previous_years/`.

Administration, Federal Aviation. Faa registry n-number inquiry, 2016. URL `http://registry.faa.gov/aircraftinquiry/NNum_Inquiry.aspx`.

ASA. 2009 data expo. asa statistics computing and graphics, 2009. URL `http://stat-computing.org/dataexpo/2009/`.

Bandyopadhyay, R. and Guerrero, R. Predicting airline delays. Technical report, Computer Science Department, CS 229, Stanford University, Stanford, CA, 2012.

Database, Global Terrorism. Global terrorism database, 2016. URL `https://www.start.umd.edu/gtd/`.

E.R.Mueller and G.B.Chatterji. Analysis of aircraft arrival and departure delay characteristics. In *AIAA's Aircraft Technology, Integration, and Operations (ATIO) 2002 Technical*, NASA, Moffet Field, CA, USA, 2002.

Lawson, D. and Castillo, W. Predicting flight delays. Technical report, Computer Science Department, CS 229, Stanford University, Stanford, CA, 2012.

Network, ASOS. Iem :: Download asos/awos/metar data, 2012. URL `https://mesonet.agron.iastate.edu/request/download.phtml?network=CA_ASOS`.

Rebollo, J. J. and Balakrishnan, H. Characterization and prediction of air traffic delays. *Submitted to Transportation Research Part C*, 2014.

Wang, Y., Sridhar, B., Jehlen, R., and Klein, A. Modeling flight delays and cancellations at the national, regional and airport levels in the united states. In *Eighth USA/EUrope Traffic Management Research and Development Seminar (ATM 2009)*, NASA, Moffet Field, CA, USA and Washington DC, USA and Fairfax, VA, USA, 2009.