Predicting Flight Delays at SFO
Applications of Data Mining

Andy Chen, Zhengyu Qin, Ted Samore

Rose-Hulman Institute of Technology

Abstract. In this report, various methods to generate models for predicting flight delays for outbound San Francisco International (SFO) Airport flights and their results are discussed.

1 Preface

The overall objective of this project was to produce accurate delay time predictions for flights out of San Francisco International Airport.

One of the primary challenges in this project was the magnitude of size with regards to the data sets. This made processing data and returning results non-trivial and time-consuming.

1.1 Software Utilized and Testbench

To generate delay time predictions, the Weka 3.7.10 library was used to build models from the training set, which would then be used to predict the correct delay times for the test set. A combination of Panda and NumPy alongside Python 2.7 was used to parse, post-process, and analyze the data.

Models were run on the following computing hardware:

- CPU : Intel Core I5-3470 4 Cores (4 maximum concurrent threads), running at 4.45GHZ
- Memory: 16GB DDR3 1600

2 Data Set

The dataset contains 168080 rows (or different flights) between December 2012 to November 2013 and features 16 columns that describe the following: Carrier, Destination, Scheduled Departure Time, Actual Departure Time, Delay Time, Delay Group, Distance, Distance Group, Cancellation Status, Diversion Status, Latitude/Longitude of Destination. The dataset was limited to domestic flights only and to carriers who were authorized to fly point-to-point between domestic destinations in the United States.

2.1 Preprocessing

We found this raw dataset to be sparse and not normalized. Additionally, building models against high dimensional data like this is both time consuming and expensive as it incurs the curse of dimensionality found in large multi-dimensional datsets.

We preprocessed the training set by pruning columns that were deemed to have information that we thought were not relevant or redundant. Some of these pruned attributes involved delay groups, distance groups, and cancellation codes. Due to the nature of Weka's classifier only takes nominal class, we did a numerical to nominal mapping on the DEP_DELAY class, so instead of discrete numerical delay time,we now have a look up table of possible delay classes, and each test instance's test class also falls into this table.

After preproccessing, the final data-set has 7 attributes that we believed closely relates to flight delays.

Name	Description	Type
DEST	Destination ID	Nominal
CRS_DEP_TIME	Scheduled Departure	Numeric
DEP_TIME	Actual Departure Time	Numeric
DEP_DELAY	Delay Time	Numeric
AIRLINE_ID	Airline ID	Nominal
TAXI_OUT	Time spent Taxing to Runway	Numeric
DISTANCE	Distance to Destination	Numeric

Because there were difficulties in processing the large amounts of data on our machine, we performed reservoir sampling to generate our refined training set of 16000 instances and test set of 2000 instances. Reservoir sampling is a family of randomized algorithms for choosing k samples from a list of n items with high probability of dataset coverage.

With our test set, we found that DEP_TIME and TAXI_OUT were either redundant or irrelevant as we generated models and tried to predict the delay time for flights in December 2013.

All preprocessing was done with a combination of Python scripts and the use of Pandas/NumPy.

3 Generating Models

We used several techniques on our preprocessed data to generate models and fit our test set to these models. Models were generated from Logistic Regression, Random Forest Decision Trees, C4.5 Decision Trees, Support Vector Machines, and K-Nearest Neighbors through Weka. These models were chosen because they were fast enough to handle the data set on our computers. Additionally, these models were believed to be robust to over-fitting which was a primary concern. Over-fitting was a major concern due to the inherent randomness in airplane delays, and the limitations of our dataset.The following list briefly describes algorithms applied and the reasoning for why the team would chose these models.

Logistic Regression - Logistic Regression is a probabilistic classification model. It works by using the logistic transform to directly predict the probabilities that data belong to certain classes. We chose to use this model because we assumed that the attributes are independent from each other, and that the logstic model can provide us the hidden relations between attributes that will be helpful when we use training other models.

Random Forest Decision Trees - The random forest algorithm constructs trees that depend on the values of random vectors sampled independently. The Random Forest algorithm provides us a randomized view of the whole dataset, and builds multiple decision trees simultaneously. We specified to build 20 trees with random seed of 5.

C4.5 Decision Trees - C4.5 decision tree is a algorithm that builds a decision tree using the concept of information entropy. As a classic and powerful algorithm, it handles both continuous and discrete attributes, which is very suitable in our case. In order to prevent from over-fitting the data, we chose to grow the tree using binary split. Additionally, we limited the height of the tree to 25 and pruned the tree when the subtree's confidence was lower than we specified.

Support Vector Machines - Support Vectors Machines algorithm defines the boundaries between classes with vectors so that hyperplanes exist between the classes. SVM is one of the most sophisticated algorithms, and, because of nominalized classes, SVMs can find the hidden categories of the classes using a non-homogeneous kernel.

K-Nearest Neighbors - K-Nearest Neighbors is one of the simplest classifications algorithm It works by finding the classifications of nearby data points.

Simple Classification And Regression Tree (CART)- CART is a recursive partitioning method, and it builds classification and regression trees for predicting continuous dependent variables (regression) and categorical predictor variables (classification). A major issue that arises when applying regression or classification trees to "real" data with random error noise is the decision when

to stop splitting. We used cross-validation with 10 folds to prevent this model from over-fitting.

4 Evaluating Models

To evaluate our models, we found the number of instances in the predicted set that were within a specified tolerance of the original test set. Nomenclature and pseudo-code of this can be seen below.

X = Each instance in the original test set
Y = Each instance in the predicted test set
α = Original delay time
γ = Predicted delay time
ω_1 = Original test set
ω_2 = Predicted test set
τ_1 = 3 minutes, first class tolerance
τ_2 = 5 minutes, second class tolerance
C1 = Correctly classified instances count under first class tolerance C2 = Correctly classified instances count under second class tolerance M_1 = Accuracy with first class tolerance
M_2 = Accuracy with second class tolerance
$|\omega|$ = The cardinality of the test set

for *X, Y in* ω_1, ω_2 **do**
  α = X.DEP_DELAY;
  γ = Y.DEP_DELAY;
  if $|\alpha - \gamma| \leq \tau_1$ **then**
    C1++;
  end
  if $|\alpha - \gamma| \leq \tau_2$ **then**
    C2++;
  else
end
$M_1 = \frac{C_1}{|\omega|}$
$M_2 = \frac{C_2}{|\omega|}$

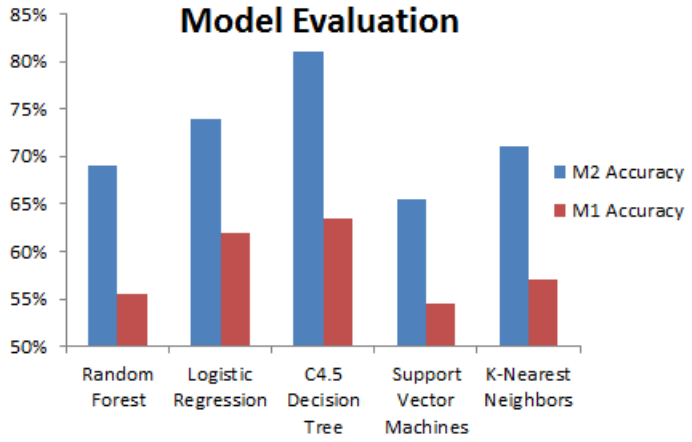
Algorithm 1: Algorithm for determine delay model accuracy

5 Results

The results from our experiments are shown in the following table:
All methods were run with their default configuration from Weka.

Name of Classifier	Tolerance Used	Accuracy
Logistic Regression	M_1	62%
Logistic Regression	M_2	74%
Random Forest	M_1	55.5%
Random Forest	M_2	69%
C4.5 Decision Tree	M_1	63.5%
C4.5 Decision Tree	M_2	81.0%
Support Vector Machines	M_1	54.5%
Support Vector Machines	M_2	65.5%
K-Nearest Neighbors	M_1	57%
K-Nearest Neighbors	M_2	71%

As expected, we can see that each method saw an improvement in accuracy when we loosened the tolerance from M_1 to M_2. The C4.5 decision tree yielded the most accurate results on the training data for both tolerances. A bar graph of this data can be seen below.



6 Test Set

Fortunately, TranStats had released the December 2013 flight data in time for us to use it as a test set. This test set contained approximately 16,000 rows for outbound SFO flights. This data was pruned of all flights that were cancelled.

6.1 Generating Models on the Test Set

The initial results with classification had displayed a great number of errors, though we also acknowledge that there may have been certain external factors

into play with respect to flights that departed in December 2013 compared to flights from the months before. It was at this point that we realized that pre-processing our data-sets of potentially irrelevant and redundant data would help increase prediction accuracy. Weka was still reporting a 93.7% error rate after focusing on the main features of the data set with respect to training the models. Our team had thought that this error rate was too high, and as a result, we did not run the same model evaluation procedures that we had done with respect to figuring out classification error where Weka had reported a 60-70% error.

It became very apparent that Occam's Razor and its principles (e.g. complex models and over-fitting) were in effect with respect to the models that were being generated from the training data.

Our next approach consisted of the following:

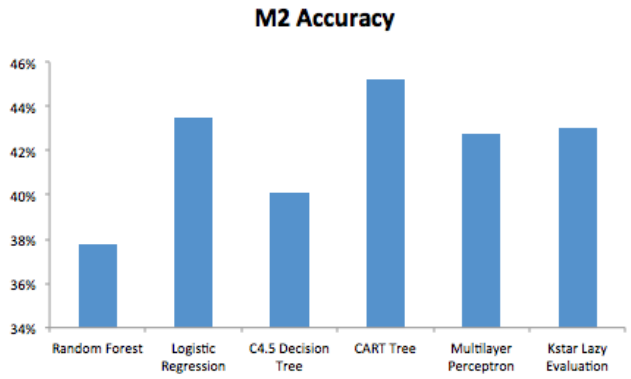
1. Perform Reservoir Sampling on Training Data and Take 16000 samples
2. Perform Reservoir Sampling on Test Data and Take 2000 samples
3. Train Models
4. Re-evaluate Results

6.2 Results

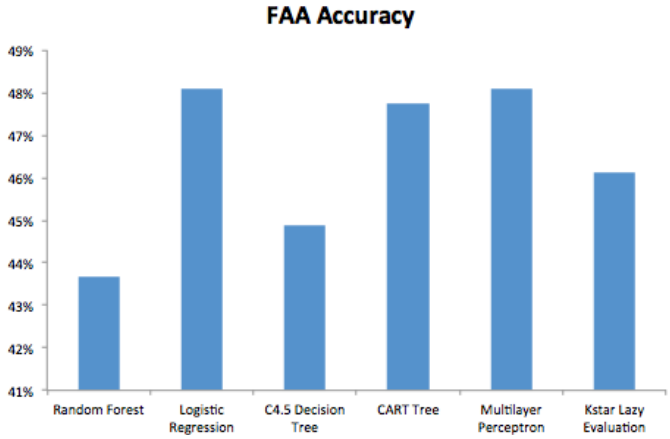
We evaluated our test data results using M_2 (from section 4) and an industry-set tolerance of 15 minutes (from the Federal Aviation Association) for the determination of flight delays.

Name of Classifier	M_2 Accuracy	FAA Accuracy
Logistic Regression	43.48%	48.09%
Random Forest	37.75%	43.675%
C4.5 Decision Tree	40.13%	44.89%
Classification And Regression Tree (CART) Decision Tree	45.21%	47.75%
Multilayer Perceptron	42.75%	48.1%
KStar Lazy Evaluation	43.01%	46.13%

Below are the graph interpretations of the performance difference of each model
M_2 Accuracy vs Model



FAA Accuracy vs Model



The simpler Classification And Regression Decision Tree leads in accuracy among the predicted times when using the M_2 accuracy metric. When the tolerance for flight delay predictions is smaller, the simpler model performs better. However, we observe that our Multilayer Perceptron and Logistic Regression models fare a little better when the tolerance is adjusted to be looser.

7 Model Calculation Performance

This section provides a brief overview of the relative performance between the methods used in this project to generate models for flight predictions:

Training Instances: 158451
Attributes: 5

- Logistic Regression:
  Time taken to build model: 568.19 seconds

- Random Forest: Max Number of trees: 10 Time taken to build model : 3.11 seconds

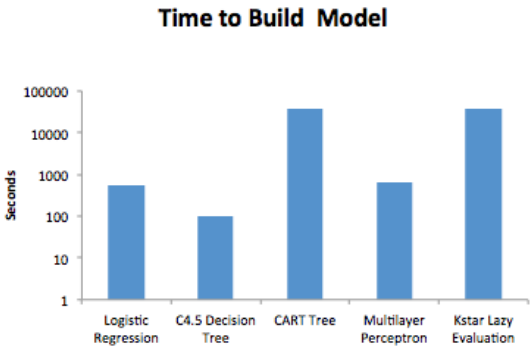
- C4.5 Pruned Tree:
  Number of Leaves : 6413
  Size of the tree : 11723
  Time taken to build model: 100.19 seconds

- Classification And Regression Tree (CART) Decision Tree:
  Number of Leaf Nodes: 15
  Size of the Tree: 29
  Time taken to build model: 36277.48 seconds

- Multilayer Perceptron:
  Time taken to build model: 643.28 seconds

- KStar Lazy Evaluation:
  Time taken to test model on supplied test set: 37767.9 seconds

Model vs Build Time(s)



The decision trees (with exception of the CART tree) were the fastest models to build compared to the rest of the methods used.

8 Conclusion

The models performed significantly worse on the test data then on the training data. The average accuracy for the models on the training data with the M_2 metric was 72.1% with a standard deviation of 5.86%. The average accuracy for the models on the test data was 42.05% with a standard deviation of 2.67%. In short, all of the models performed poorly, with less difference between them.

However, we saw different improvements with how each model was evaluated

with the M_2 and the FAA metrics on the test data. The average accuracy with the FAA metrics jumped to 46.44% (from 42.05%). The model that saw the most improvement was Multilayer Perceptron.

The C4.5 decision tree model produced the most counter-intuitive results. It performed the best with the training data with an accuracy of 81.0% for the M_2 metric. However, with the exception of the Random Forest Model, all of the other models out-performed the C4.5 decision tree model on the test data with the M_2 metric. This means that C4.5 decision tree model was the most over-fit model to the training data. However, it should be taken into account that all of the models performed poorly with the test data.

Models differed substantially more in their speed than in their accuracy. The quickest model, the Random Forest, took only 3.11 seconds to build. On the other hand, the CART Decision tree and the KStar Lazy Evaluation model each took more than 10 hours to build. This is greater than a 12,000 factor difference. Despite this large difference in the time, the accuracy of the Random Forest model was only marginally lower than the accuracy for the KStar Lazy Evaluation and CART Decision trees model (43.675 % compared to 46.13 % and 47.75%).

9 Reflection

Predicting flight delays accurately remains a difficult problem. We observed that past performance was not always indicative of future performance. We believe that the predictions would be improved if we had incorporated our dataset with features like the following:

- Multi-dimensional Weather data of departing/arriving airport
- Air Traffic Control Data (most likely not publicly accessible)
- Aircraft Registration / Tail Number of the Aircraft used during the flight
- Aircraft Type

While the team had hoped for higher accuracy rates in its predictions of SFO-outbound flights, the accuracy rate given the provided data and the attributes used was considerably impressive (having started from the point where it was unknown whether flight times and destinations by themselves had a strong correlative effect on delay times).

10 Future Plans and Possible Applications

Through our experiments, we attempted to find a way to classify the possible delay for particular flight. As a result, one can plan their trip accordingly and

minimize the possibility of delays.

Future extensions of this project can include:

- Expansion of scope of airports covered.
- Provide a front-end for user to simply type in destination, airline, time of flight and the system will give predicted delay time accordingly.
- Provide analytics data for airline companies.
- Cooperate with travel planning agencies and provide a filter for people who are flexible with schedule and willing to travel on low delay flights (Kayak, Expedia, etc.).

11 Interesting Findings and Trends

From our data - we observed:

- Regional airlines tend to have a much greater number of delays. Potential reasons for this is the contracting that major airlines do for regional carriers and how their operational procedures differ from those of larger aircraft.
- United Airlines, compared to the other airlines observed in this study, is the legacy carrier with the greatest number of delays.
- US Airways is the legacy carrier with the least amount of delay time.
- In the value carrier segment, JetBlue beats Southwest with regard to delay time - however it should be noted that Southwest serves more destinations in general.
- Flights around 11:00-15:00 have the highest potential for longer delays.
- Major hub airports are subject to a greater frequency of delays (e.g. Atlanta (ATL) , Dallas Fort-Worth (DFW), Chicago O'Hare (ORD))

12 Appendix

12.1 Look-up Tables

AIRLINE ID	CARRIER
19393	Southwest
19690	Hawaiian Airlines
19790	Delta
19805	American Airlines
19930	Alaska Airlines
19977	United Airlines
20304	Regional Airlines (Skywest)
20355	US Airways
20409	JetBlue
20436	Frontier
20437	AirTran
21171	Virgin America

12.2 Diagram of Trees

The trees (produced by Weka) are too complex to print on paper, therefore the following hyperlinks contain a textual representation of the trees in our dataset.

C4.5 Tree

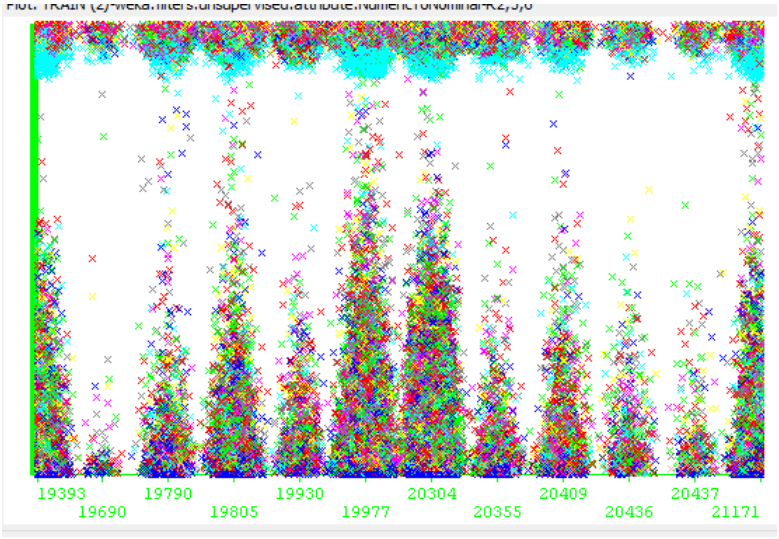
Classification and Regression Tree

Random Forest

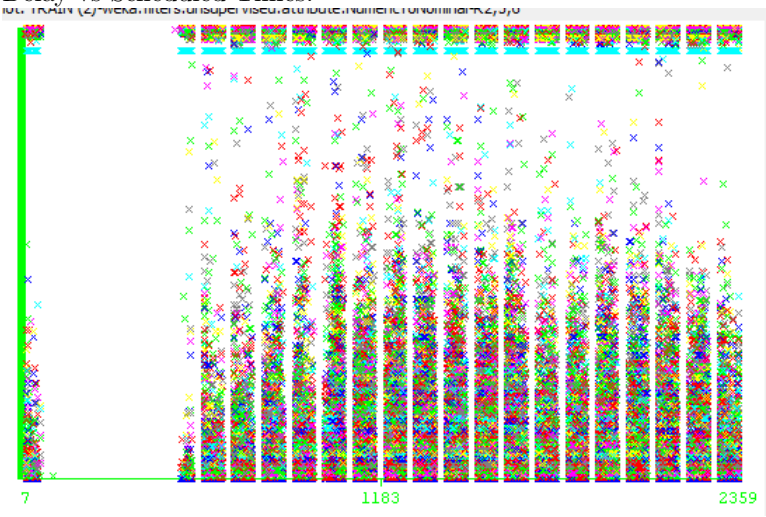
12.3 Heatmaps

Heatmaps were explored to present some of the interesting findings and trends among flight times/airline and its potential correlation with delay time. The respective heatmaps are illustrated below:

Delay vs Airlines:



Delay vs Scheduled Times:



12.4 WebGL Visualization

A web visualization was made to illustrate average flight delays from SFO based on destination.

Note: We acknowledge there are some discrepancies with some of the longitude/latitude data we had received from an external source

WebGL Visualization

12.5 Links to Software/Data

Dataset Source: Transtats On-Time Performance

Software:

Python

NumPy/Pandas

Weka Machine Learning Tool