Predicting Flight Delays and Cancellations

Kyle Durfee, Mehrad Moradshahi, Ajit Punj
(kpdurfee, mehrad, apunj)

December 15, 2017

1 Task Definition

1.1 Goal

Nothing ruins best laid plans more than a canceled or delayed flight. Our project uses data provided by the United States Department of Transportation [1] over the scope of many years to predict whether a flight will be canceled, and if not to predict the minute value of any delays. Our project focuses on underlying flight characteristics known at the time of ticket purchase to identify possible underlying causes as opposed to simply training a predictive model on identifying features such as flight number or using features that are clearly indicative but not known until flight time such as weather or security data.

1.2 Tools

Our prediction and classification models were implemented in python using the scikit library in conjunction with pandas, as well as in Matlab to provide another point of comparison and sanity checking. The associated code for python and Matlab are included along with our report. The final tuned python models have been uploaded onto the CodaLab worksheet 0xdcc653253e0143c4aecc85ea963c9db1 along with a variety of runs showcasing the performances of the different tuned models. Finally, to further explore the usefulness of our models and features we implemented a neural network using Matlab and the 'Nerual Designer' application. The source files for our model (which are in the form of .m and .ndp) are also included in our hand in and can be used to reproduce our results.

1.3 Scope

Our project attempts to classify flights as cancelled or not cancelled, and to predict the minute value of a flights delay. Due to the scale of the data set and the limits of the hardware at our disposal we chose to develop a proof of concept by training and testing our models on data for domestic flights from 1987-2008 between the five airports with the most traffic: ATL, LAX, ORD, DFW, JFK.

1.4 Evaluating Cancellation Classifications

When evaluating the success of our cancellation predictions, we must keep in mind that a very small percentage of flights are cancelled every year ($<$2% in 2016 according to BTS) and many of them are due to reason not represented in our data set such as mechanical failure. However, for a user of our model what they really care about is how trustworthy the classifications are. If our model tells a user that a flight will be cancelled should they avoid it altogether? If our model tells a user the flight will not be cancelled can they plan to fly the day before an important function? For this reason when evaluating our cancellation classifications we will use precision and recall metrics, separating the precision and recall for each type of prediction to give further insight into the results. With regard to cancellations:

- $+precision = (\#correctlypredictedascanceled)/(\#predictedcanceled)$
- $+recall = (\#correctlypredictedcanceled)/(\#actuallycanceled)$
- $-precision = (\#correctlypredictedasnotcanceled)/(\#predictednotcanceled)$
- $-recall = (\#correctlypredictednotcanceled)/(\#actuallynotcanceled)$

1.5 Evaluating Delay Predictions

When evaluating delay predictions, what a user of our model is truly concerned with is not only the accuracy of our predictions in terms of whether a flight is delayed or not, but how close to the actual delay our models prediction is. Many models in related works simply classify delays based on a cutoff, but if our model correctly predicts a delay of 20 minutes and the flight is delayed 4 hours we do not consider that useful / a success. For this reason we focus on the MSE of our prediction calculations. For our python models, we also report the % of predictions that are within 20 minutes of the actual. Obviously the usefulness of these predictions is still related to how trustworthy they are, and for this reason we look at the precision and recall of delay/ not delayed predictions similar to classifications. These statistics were reported to give further insights into the models behavior during experimentation.With regard to delays:

- $+precision = (\#correctlypredicted > 0)/(\#predicted > 0)$
- $+recall = (\#correctlypredicted > 0)/(\#actually > 0)$
- $-precision = (\#correctlypredicted <= 0)/(\#predicted <= 0)$
- $-recall = (\#correctlypredicted <= 0)/(\#actually <= 0)$

2 Literature Review

There are many previous studies trying to predict flight delays and cancellation using different sets of data such as origin, destination, time, distance, weather, etc. A notable work which provides a survey of different previous approaches is [9]. We studied this paper thoroughly to get a good idea of various interpretations one could have about successfully modeling a flight delay. The main four feature types used are: Planning, Temporal, Spatial, Weather. In this project we only used the first three features as they are the ones a customer would have access to at the time making decisions. The methods they reviewed includes Network Representation, Probabilistic Models, and Statistical Analysis. Due to complexity, compute, and time constraints we were only able to try out statistical methods but using our AI knowledge (graphical methods, logic, etc.) can also improve the final results. In many of the papers it was noted that weather was used as a feature. [2] specifically states that by not using weather data they experienced a decrease in accuracy from 86.9% to 69.1% with delay threshold of 60 minutes. They implemented their method with MapReduce for scalability purposes and their data source was huge compared to ours. What we attempted in this project was to improve upon their results using only flight data (excluding weather). [4] is another paper which gives a simple description of the problem outline and uses multiple linear regressions to generate predictions. While they achieve good results, the scope of used model is very small they cannot go beyond a certain threshold which limits the usefulness. While many related projects are similar, they differ not only in the models used but in the scope of data used. In [3] they used a very large portion of data (about 130 million of flights) to calculate the results, while in another paper [6] they focused on one airport and used the data from the past 2 years to do the prediction. In a similar work [5] , seven major carriers, and fifty highest-traffic airports were under focus. They used Binary Classification, and Probability Estimation to estimate the delay and concluded the latter is a better approach. In our project we focused on the main 5 airports which has a trade-off between generality and precision. In a more recent work [8] , some new features were being incorporated in the model as well as new evaluation techniques. Security and delay of previous flights are among those features and as a result less error and higher recall was been achieved. It has also been shown in works such as [7] that neural networks tend to work better than other methods in this problem since they are able to detect and capture the complex features affecting the end results. In general, related works tended to limit the scope of the problem by using aggressive cutoffs (such as delayed > 30 minutes or not) or used retrospective data such as weather. While incorporating many of the techniques of these other works our project differs in that we look at both cancellations and delay predictions. We compare the results of multiple models and attempt to provide results that are truly useful to a consumer with only data that would be available at time of purchase.

3 Infrastructure

3.1 Raw Data

We obtained a data set provided by the United States Department of Transportation Bureau of Transportation Statistics, which contains data from domestic flights between the years 1987 to 2008. This data set contained many columns of data including flight number, origin and departure airport, scheduled and actual arrival and departure time to name a few. This data set had relevant and irrelevant information for our purposes, so it was necessary to refine the data set before using it to train our models.

3.2 Selecting Data

Since the raw data set had a lot of information that we did not deem necessary for our model and was very large due to the inclusion of every flight from every day of the year, we had to select a portion of the data to use. First, we opted to use just flight data from the month of December since we estimated the busy holiday travel season would have more flight delays and cancellations and would serve as a relevant proof of concept. Next, we cut the data set to only have flight data between the top 5 busiest airports, for example JFK, LAX, and SFO to make the set more manageable and allow us to experiment given limited compute resources. Finally, there were many columns of data we did not deem relevant for our prediction model. For example airplane tail number was not relevant and often not populated, and other information such as time spent taxing would not be known until the flight was over. Once we had a manageable data set with relevant features, we had to pre-process our data further to allow our model to use it appropriately. In the end the features extracted from our data set included:

- Day of Month
- Day of Week
- Scheduled Departure Time
- Scheduled Arrival Time
- Airline Code
- Scheduled Flight Time
- Origin Airport Code
- Departure Airport Code
- Flight Distance

3.3 Pre-processing Data

Once we had a subset of data ready, we had to refine it before it could be used. First, any data in the set that was incomplete in terms of features of interest were discarded. Second, delays were represented by the total time (minutes) that each flight was delayed in one data set, but cancellations were represented in a binary way (cancelled or not cancelled) in another. When considering how to treat cancellations in the delay data set, we had to replace the "N/A" delay (of essentially infinity) with some value. We experimented with different values and found that 120 minutes was suitable for our models.Because features such as "Airline Code" do not have a linear relationship we split most features into multiple indicator features. For example "Airline Code = AA" became one indicator feature and "Airline Code = UA" became another. Some features such as flight distance were left as is, but departure and arrival times, airlines, and departure and arrival airports were all split out into indicator features (resulting in over 150 total features).In order to try to improve our prediction results, we first tried to bucket certain fields into smaller subsets. For example, since there was not much overlap with scheduled departure times, we grouped all departure and arrival times within 30 minute increments into one bucket by rounding down to the nearest half hour so we only had 48 possible departure/arrival time features. This gave better performance than creating an indicator feature for each minute, but gave diminishing returns beyond 30 minute increments.In order for our models to function as expected each feature had to be scaled. Finally, as airline traffic has changed significantly over the years we had to randomly shuffle our data set to ensure a relatively even spread of traffic on different airlines between training, evaluation, and test sets.

4 Approach

4.1 Challenges

This problem (of cancellation classification and delay prediction) is unsolved because of the complexity, and we are faced with a number of challenges. First, the data set is incredibly large and in the case of delays contains a wide range of actual values allowing lots of room for error. The actual delay values also contain a number of outliers, for example while most delays are clustered around 27 minutes a small number are over 500 minutes.The data set is also incomplete in that many rows feature blank or N/A inputs, airlines go out of and come into business over the years etc. The problem also encapsulates cancellations and delays that cannot be known ahead of time without additional information that is unavailable to us at time of purchase (or not published). This includes information such as weather that a user could not know in advance at purchase time as well as information about mechanical maintenance and/or flight crew scheduling that is not published by airlines. The data that we are using is published every year and is

undoubtedly analyzed by airlines resulting in a self correcting data set. This is especially relevant because in order to allow for our models to deduce traffic patterns based on dates we must use data over multiple years (during which airlines presumably adapt). In the case of delay predictions we are faced with the task of how to use cancellation data, as they are likely caused by similar underlying factors but have an effective delay of infinite value. In general, the main challenge is working with an incomplete set of sparse cancellation data and prediction data that is likely self correcting and so lacks more obvious feature relationships.

4.2 Baseline

Our baseline differs when classifying cancellations and predicting delay values, as the problems and data sets are fundamentally different. When creating a baseline for classifying predictions we acknowledge the fact that the percentage of cancelled flights is a published statistic every year, so we randomly assign a given flight to be cancelled based on the percentage of flights in our data set that are actually cancelled (which is arguable more accurate than basing your guess of recently published data in the real life case). The baseline for delay predictions is based largely on the actions of a common consumer. Generally, someone flying does not know any underlying information to glean flight delay information so they plan for the flight being up to 30 minutes early, and as much as an hour late. Our baseline therefore assigns a random delay value between -30 and 60 for each flight.

4.3 Oracle

The oracle for our cancellation classification uses a rounded number from our data set and information from publications made by the Bureau of Transportation that about 15% of delays and cancellations are caused by extreme weather related issues. With this in mind, our cancellation oracle 'knows' 15% of cancellations and predicts them correctly. The other flights (100% of not canceled and remaining 85% of canceled) are randomly predicted based on the actual percentage of cancellations that are recorded in the data set. The oracle for delay predictions similarly predicts 15% of delays with 100% accuracy, assuming that the oracle 'knows' about extreme weather related delays. The oracle then guesses randomly within 45 minutes of the actual delay value for the rest, greatly reducing the possible error in its predictions.

4.4 Logistic Regression

In order to predict delay values we used logistic regression both from the python scikit-learn library and from Matlab. These implement regularized logistic regression to attempt to reduce overfit on the training data. These models allowed for little tuning but provided reasonable results and allowed us to quickly analyze the impact of the decisions we made in terms of pre processing the data and creating feature vectors.

4.5 SVM

In an attempt to produce more desirable results we implemented a support vector machine using SGD to minimize loss for both predictions and cancellations. For delay predictions we minimized the squared loss, and for cancellations we minimized hinge loss. In both python and Matlab the SVM results in terms of delay predictions were comparable to those produced using logistic regression, however we were able to reduce over fit slightly by reducing the step size, adjusting the regularization parameter, and reducing the number of training iterations in our models. As a result we were able to produce results that were slightly better than those produced from our logistic regression model. For cancellations a separate classification model was created and trained using the cancellation data, and the same parameters were tuned. For both regression and classification we had to drastically reduce the step size to get reasonable results, due to the sparsity of our data set and the existence of outliers as mentioned previously.

4.6 Random Forest

As logistic regression and minimizing loss using SGD produced comparable results we wanted to experiment with other, less related classification and regression methods. Random forests are an ensemble learning method that trains a model by creating decision trees based on features and then reporting either the mode (for classification) or the average (for regression) of the results of all the decision trees in the model. Random forests were seen to outperform logistic regression and loss minimization using SGD, however we saw a significant drop off between the performance on the training data and the performance using evaluation / test data. To minimize this we tuned the models by reducing the number of and depth of decision trees. We also found that we were able to eliminate some features all together when using random forests (arrival time, distance, day of week) and achieve better results which was not the case with our other models.

4.7 Neural Network

Neural networks achieve great results despite their simplicity. By extracting the relationship between different features and producing new features that capture the correlation between data features they are able to outperform most other current machine learning techniques. For this part, we used MATLAB and Neural Designer (ND) to perform our simulations. Through experimentation we found that unlike our other models, we produced better results with our original feature vector as opposed to converting to indicator features. We first normalized each feature vector by dividing the values by the maximum value for that specific feature for MATLAB models. ND itself normalizes data by subtracting the mean and dividing by the standard deviation. We then shuffled the values and and chose "70%, 15%, 15%" metric for training, developing, and testing. For the neural network, we used different architectures and chose the one which gave the lowest validation MSE for fewer iterations. The result was a four-layered neural with sigmoid activations for all layers except the first, which is linear. In ND we used a five-layered network with the same properties as before and used Quasi-Newton method for training. It should be noted that due to computing limitations some experiments run with our neural network models had to use random subsets of our full data set. Throughout our experiments we produced more desirable results using ND and so have reported those results as our "NN" results in this paper. The ND architecture is shown below:

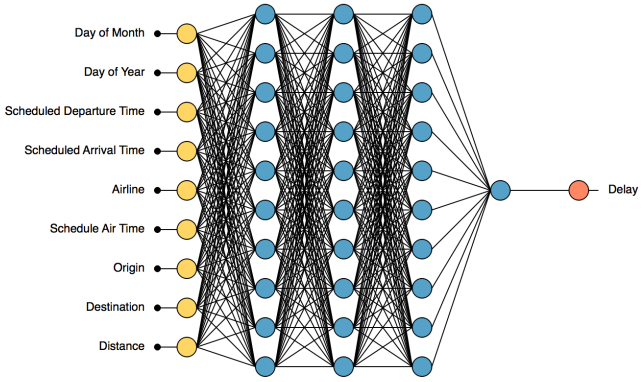

Figure 1: Neural Network Architecture

4.8 Clustering

One way we thought we could improve our prediction values would be to first group all similar data points into clusters using k-means clustering, and then running regression on each cluster. First, to form clusters in our python models there were two ways to initialize each cluster: random, and "k-means++". The random method simply randomizes the initial centroid point, while "k-means++" speeds up convergence by intelligently picking the initial centroid values. Sticking with "k-means++" to improve run-time, we had to then tune the number of clusters. We observed a peak somewhere around n=5 to n=10 in our python models, so we stuck with n=5 since this had the best trade off between running time and results. Since we had to perform regression per cluster, using a large n value just increases running time without much improvement in results. To simplify clustering for our neural network models we used four clusters and mapped the features using SOM (self-organized map) network to a 2-D space. Figure 2 below shows the relative density of the clusters used in our neural network delay predictions.

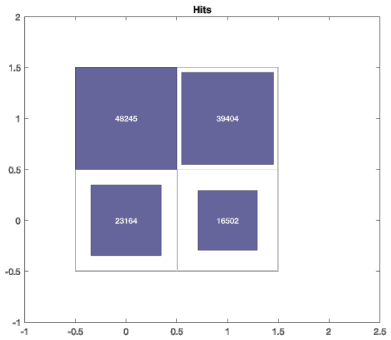

Figure 2: Neural Clustering

5 Results

*LRK = Linear Regression with K-means clustering using n=5
*RFK = Random Forrest with K-means clustering using n=5

Metric	Baseline	Oracle	LR	LRK	SVM	RF	RFK
MSE (min)	3392.9	586.5	2670.4	2668.98	2670.2	2151.4	2151.5
+Precision	57.2%	76.6%	57.2%	57.2%	57.2%	57.6%	57.6%
+Recall	65.9%	78.1%	99.9%	99.9%	99.9%	99.9%	99.9%
-Precision	42.7%	70%	58.1 %	51.7%	59.1 %	96.6%	96.9%
-Recall	34%	68.2%	.082%	.051%	.084%	1.5%	1.49%
% within 20	38.7%	51.4%	47.6%	47.5%	48.4%	54%	54%

Table 1: Delay Predictions Training Data

Metric	Baseline	Oracle	LR	LRK	SVM	RF	RFK
MSE (min)	3271.5	589	2552.9	2553.1	2552.9	2480.4	2563.4
+Precision	57.1%	77%	57.4%	57.4%	57.5%	57.7%	57.5%
+Recall	65.9%	77.9%	99.9%	99.9%	99.9%	99.3%	99.8%
-Precision	42%	69.8%	66.6%	62.5%	64.3%	68.6%	66.7%
-Recall	33.3%	68.7%	.114%	.115%	.103%	2.16%	.44%
% within 20	38.6%	51.1%	48%	48.1%	48.9%	50.3%	49.7%

Table 2: Delay Predictions Test Data

Metric	Baseline	Oracle	SVM	RF
+Precision	2.4%	100%	12.8%	100%
+Recall	2.37%	17.2%	.3%	4.98%
-Precision	97.5%%	98%	97.5%	97.6%
-Recall	97.6%	100%	99.9%	100%

Table 3: Cancellation Predictions Training Data

Metric	Baseline	Oracle	SVM	RF	NN
+Precision	4.2%	100%	84.7%	100%	4.2%
+Recall	4.23%	16.6%	2.9%	.192%	21.7%
-Precision	97.5%%	97.9%	97.5%	97.5%	98.0%
-Recall	100%	100%	99.2%	100%	88.2%

Table 4: Cancellation Predictions Test Data

Metric	delay-regression	delay-binned	cancel
MSE	1161.15	242.59	—
Accuracy	—	57.82%	90.01%

Table 5: Neural Networks Train Data

Metric	delay-regression	delay-binned	cancel
MSE	1483.97	263.86	—
Accuracy	—	51.55%	86.68%

Table 6: Neural Networks Test Data

—	Predicted negative delay	Predicted near zero delay	Predicted positive delay
Actual negative delay	424	0	130
Actual near zero delay	106	0	34
Actual positive delay	214	0	91

Table 7: Neural Network Binned Delays Test Data

—	Predicted no cancellation	Predicted cancellation
Actual no cancellation	5	18
Actual cancellation	115	861

Table 8: Neural Network Test Data Cancellations

6 Analysis

6.1 Model Analysis

One of the first things we noticed was that the results for most of our models are much closer to those of our baseline then they are to our Oracle. While not ideal, this is indicative of the difficulty of the problem and the lack of easily exploited feature relationships in our data set. As seen above in section 5, the results using logistic regression and minimizing loss via SGD were similar, but we were able to tune our SVM models more aggressively to reduce over fit and so in the end the SVM models outperformed logistic regression. However both models were significantly outperformed by random forests both when predicting delays and classifying cancellations. As shown below, there is a much more linear relationship between predicted and actual delays using random forests on test data than SVM which clustered predictions heavily around the mean.

(a) SVM Regression Delay Prediction on Test Data (b) Random Forest Delay Prediction on Test Data

Figure 3: SVM vs Random Forest Delay Predictions

This is true in terms of all of our main metrics with the exception of cancellation recall, as the random forest model classified fewer flights as cancelled (though with greater precision). In all models, when running with test data as opposed to training data we saw less of a linear relationship between predicted and actual and experienced clustering around the mean which is a sign of over fit and a lack of relevant feature relationships. Finally, after hand tuning these models we utilized a neural network to identify complex feature relationships in our data set. The delay prediction results of the neural network were significantly better (in terms of MSE) than any of our other models. The results for the neural network on each subset of our data is shown graphically below, as well as outlined in section 5.

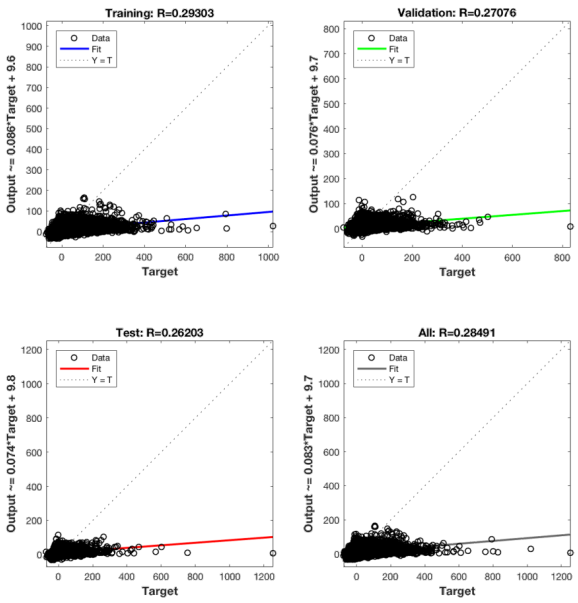

Figure 4: Neural Network Delay Regression Plots

In an attempt to produce more user friendly output we used our neural network to classify delays into three "bins". The hope was that by restricting our domain to either no/negative delay, small delay, large delay, we would be able to give much more reliable results. While understandably the MSE of this approach is much lower (as our outputs can essentially have 3 values) results in section 5 show that the overall accuracy of this binning was not high enough to be considered reliable. As a result we deemed with approach to be less useful than our minute based predictions (the results of which are reported in section 5).

In terms of cancellation classification the neural network tended to predict a higher number of cancellations which resulted in a low prediction precision. In this case, the lack of precision made the neural network less desirable than other methods such as RF classification.

6.2 Clustering Analysis

When observing the results of our experiments (some of which can be seen in section 5) it becomes apparent that clustering the data had little positive effect (if any) on the performance of our models. For this reason we abandoned clustering in our final models. When attempting to use clustering with our neural network especially we saw a large increase in the complexity and run time for a negligible increase in performance.

6.3 Over Fitting Analysis

One very apparent issue with all of our models is that when predicting delay values they all produced better results on our training data. The random forest model in particular produced vastly superior results on the training data compared to the evaluation and test data. Though we attempted many techniques to reduce this, we had limited success when comparing results to our oracle. Techniques applied with limited results include:

- Introducing regularization factors
- Adjusting the step size in SGD applications
- Reducing iterations in SGD applications
- Reducing features in training set
- Normalizing features
- Restricting bounds of training set
- Adding a bias feature
- Clustering our data before training
- Grouping results into meaningful 'buckets' (i.e. 30 minute increments etc)
- Reducing the number and depth of decision trees in our random forest models

The over fitting issue with random forests in particular can be seen in the following figure as well as in the result tables of section 5 for all models.

(a) Random Forest Delay Prediction on Training Data (b) Random Forest Delay Prediction on Test Data

Figure 5: Performance Dropoff from Training to Test Data

This led us to believe that performance was hindered by a lack of relationships between the actual results and our features. In order to further explore this we analyzed our SVM models and utilized Neural Networks to extract complicated feature relationships. The following figure shows the weights from our trained SVM regression model for delay predictions. By analyzing the weights we found that while there are a small number of features that are more significant delay indicators, many of the weights are clustered around 0.

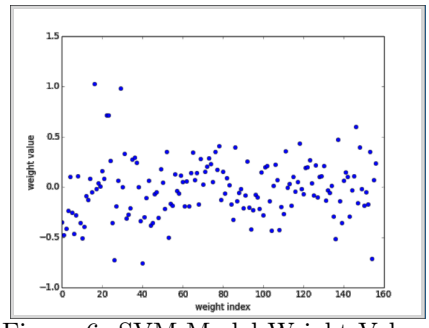

Figure 6: SVM Model Weight Values

Essentially this means that our models are able to identify a small number of relevant indicator features, but beyond those our predictions collapse around the mean which is evident in our results. The same issue, though even more apparent due to sparsity, was seen when analyzing the cancellation classifier weights. This is reflective of our sparse data set, and the fact that over the years airlines have self corrected for any "low hanging fruit". Through the use of neural networks we were able to extract complex feature relationships resulting in more significant features for prediction. As a result, the use of neural networks greatly decreases our MSE and brought our delay predictions much closer to the oracle. However, even our neural network suffered a noticeable drop off in MSE when predicting the training and test sets indicating that the issue of relevant features is still negatively impacting our results. As stated previously, even the neural networks were unable to provide meaningfully accurate cancellation predictions due to the sparsity issue.

6.4 Overall Prediction Analysis

Overall, our hand tuned models were able to outperform our baseline when predicting airline delays. This is significant as this problem is incredibly difficult even when using more obvious features such as weather or when using vast amount of data (as Google does). By analyzing model weights and our results it is obvious that our models suffer from over fit due to a lack of highly significant features. In general our models are able to identify a small number of flights accurately and regresses towards the mean when predicting the remainder. This analysis was bolstered by our implementation of a neural network, which showed improved results by identifying complex relationships between our features. Techniques such as clustering, or reducing the output domain of our models were shown to have little impact on these results across all models. These results show that while there are relevant relationships in our data set in terms of delay prediction, to achieve better results we require more data and more relevant features for training.

6.5 Overall Classification Analysis

While the binary nature of modeling cancellation classification is arguably simpler than that of predicting delays, many of the shortcomings we saw in the delay prediction data set were magnified in the cancellation set. While we were able to outperform our baseline, we saw a lack of highly significant features. As a result our models were either able to accurately predict a very small number of cancellations (RF for example) or predict a larger number of cancellations

with lower precision (NN for example). This is unsurprising due to the sparse nature of cancellations in our data set and the tendency for cancellations (more so than delays) to be due to extreme and unforeseen events such as extreme weather or mechanical failure. As a result, while we were able to outperform our baseline we would require more relevant features in our data set to come closer to the performance of our oracle and achieve cancellation prediction precision and recall that would be useful to users.

6.6 Final Thoughts

Though able to outperform our baseline, we found the results of hand tuned models in terms of delay predictions and cancellation classification to fall short of what would be considered useful to an airline customer at the time of purchasing. By using neural networks we were able to produce far superior delay predictions, though still not ideal from the point of view of a user attempting to purchase tickets. Cancellation classifications using all models resulted in a lack of reliability that limits usefulness. Our goal was to deviate from the large number of related works that use the simple/obvious correlations with weather data and instead provide insight into the frustrations of delays and cancellations of flights using only the data that someone purchasing airline tickets would have at the time of purchase. While our models are able to exploit relationships with some of these features when predicting delays, we simply did not have enough data to give cancellation classification results that are truly useful in a real world application. There is hope that with additional computing power the relationships we identified for both delays and cancellations could be further exploited and expanded upon by using data from all airports during all months of the year. However through tuning and analyzing various models and their results we have determined and shown through our experiments that in order to give truly useful and reliable delay and/or cancellation predictions one would need more features to be provided from airlines such as information pertaining to flight crews or mechanical maintenance schedules.

References

[1] http://stat-computing.org/dataexpo/2009/the-data.html.

[2] Loris Belcastro, Fabrizio Marozzo, Domenico Talia, and Paolo Trunfio. Using scalable data mining for predicting flight delays. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 8(1):5, 2016.

[3] William Castillo Dieterich Lawson. Predicting flight delays. *CS229 Final Report:*, 2012.

[4] Yi Ding. Predicting flight delay based on multiple linear regression. In *IOP Conference Series: Earth and Environmental Science*, volume 81, page 012198. IOP Publishing, 2017.

[5] Brett Naul. Airline departure delay prediction. *CS229 Final Report*, 2008.

[6] Rafael Guerrero Raj Bandyopadhyay. Predicting airline delays. *CS229 Final Report:*, 2012.

[7] Banavar Sridhar Yao Wang Richard Jehlen, Alexander Klein. Modeling flight delays and cancellations in the us. *Eighth USA/Europe Air Traffic Management Research and Development Seminar (ATM2009)*, 2009.

[8] Jonathan Leaf Romain Sauvestre, Louis Duperier. Modeling flight delays. *CS229 Final Report:*, 2016.

[9] Alice Sternberg, Jorge Soares, Diego Carvalho, and Eduardo Ogasawara. A review on flight delay prediction. *arXiv preprint arXiv:1703.06118*, 2017.