1 INTRODUCTION

In the present world, the major components of any transportation system include passenger airline, cargo airline, and air traffic control system. With the passage of time, nations around the world have tried to evolve numerous techniques of improving the airline transportation system. This has brought drastic change in the airline operations. Flight delays occasionally cause inconvenience to the modern passengers [1]. Every year approximately 20% of airline flights are canceled or delayed, costing passengers more than 20 billion dollars in money and their time.

1.1 Research Motivation

Average aircraft delay is regularly referred to as an indication of airport capacity. Flight delay is a prevailing problem in this world. It's very tough to explain the reason for a delay. A few factors responsible for the flight delays like runway construction to excessive traffic are rare, but bad weather seems to be a common cause. Some flights are delayed because of the reactionary delays, due to the late arrival of the previous flight. It hurts airports, airlines, and affects a company's marketing strategies as companies rely on customer loyalty to support their frequent flying programs.

1.2 Problem Statement

My case study was about LaGuardia Airport in New York, Logan International Airport in Boston, San Francisco International Airport in San Francisco, and O'Hare International Airport in Chicago, which are four major airports in the United States of America. But we focused the idea and research on LaGuardia International Airport. Compared with the data produced by all airports in USA, the data which we gathered was very limited, but it gave us a great direction on how weather plays a part in flight delays. In this project, the goal is to use exploratory analysis and to build machine learning models to predict airline departure and arrival delays.

1.3 Report Structure

This master project report is organized into nine chapters. The preface of the project, research motivation, and problem statement form chapter 1. Chapter 2 describes the basic concepts of flight and weather data. Chapter 3 focuses on structures of the project. Chapter 4 and 5 explain the data collection and data exploration part of the flight data, while the chapter 6 focuses on predictive modelling implemented on the flight data. Chapter 7 focuses on predictive modelling implemented on the weather data. Chapter 8 starts with the introduction of the Twitter data and some tweets exploration that helped me in the course of building the project. It focuses on predictive modeling of Twitter data using Random Forest and Support Vector

Machine. Chapter 9 concludes the paper and finally chapter 10 talks about the future scope of the project.

1.4 Related Work

The main concern of the researchers and analysts is to predict the reasons for flight delays and for that they have put in their efforts on collecting data about flight and the weather. Mohamed et al. [2] have studied the pattern of arrival delay for non-stop domestic flights at the Orlando International Airport. They focused primarily on the cyclic variations that happen in the air travel demand and the weather at that particular airport.

In Shervin et al.'s work [3], their motive of research is to propose an approach that improves the operational performance without hampering or effecting the planned cost.

Adrian et al. [4] have created a data mining model which enables the flight delays by observing the weather conditions. They have used WEKA and R to build their models by selecting different classifiers and choosing the one with the best results. They have used different machine learning techniques like Naïve Bayes and Linear Discriminant Analysis classifier.

Choi et al. [5] have focused on overcoming the effects of the data imbalancing caused during data training. They have used techniques like Decision Trees, AdaBoost, and K-Nearest Neighbors for predicting individual flight delays. A binary classification was performed by the model to predict the scheduled flight delay.

Schaefer et al. [6] have made Detailed Policy Assessment Tool (DPAT) that is used to stimulate the minor changes in the flight delay caused by the weather changes.

Bing Liu [7] has done a sentiment analysis and opinion mining that analyzes people's opinions, sentiments, and studies their behavior. The output of the research is a feature-based opinion summary which is also known as sentiment classification.

Using techniques such as Natural Language Processing, Naïve Bayes, and Support Vector Machine, researchers built algorithms for analysis that helped them in extracting features in the model. Most of them focused on predicting overall flight delays. Our research concentrated mainly on predicting flight delays for a particular airport over a specific period of time. First, we used a regression model to examine the significance of each feature and then, a feature selection approach to examine the impact of feature combination. These two techniques determined the features to retain in the model. Instead of using the whole set, we sampled 5,000 records at a time to run through different machine learning models. The machine learning models implemented here were Random Forest classifier and Support Vector Machine (SVM) classifier. Further, we applied an approach

called One-Hot-Encoder to create a variant of the model for evaluating potential prediction performance.

2 BASIC CONCEPTS

We will be looking into some of the variables that play a vital role in determining the flight delay. Each variable has a different weight that is used while training the model. Prior knowledge of these variables should be present to understand the dataset.

2.1 Flight Data

The flight data is from the year 2007 and 2008, which is taken from Bureau of Transportation Statics. Every flight has many variables, which give detailed information about the specific flight [8].

1. Flight Number
2. Carrier {American, United}
3. Destination {Airport Code: SFO}
4. Origin {Airport Code: LAX}
5. Date {MM/DD/YY}
6. Day of Week {Mon, Tue, Wed, Thu, Fri, Sat, Sun}
7. Scheduled Departure Time {HH:MM AM/PM}
8. Actual Arrival Time {HH:MM AM/PM}
9. Actual Departure Time {HH:MM AM/PM}
10. Minutes Late {+Late/-Early}
11. Scheduled Arrival Time {HH:MM AM/PM}

The 2008 on-time performance data contains 7 million records. The average number of daily flights is 19,178, with 24 data elements included in the flight database.

2.2 Weather Data

The Aviation System Performance Metrics (ASPM) [9] data from the FAA operations performance data website contains airport specific weather, flight demand, and airport capacity information. The data used in this project is from 2008. It provides hourly weather and operational data for 77 US airports. A list of data fields is presented below:

1. AirportID
2. Year
3. AdjustedMonth
4. AdjustedDay
5. AdjustedHour
6. TimeZone
7. Visibility
8. DryBulbFarenheit
9. DryBulbCelsius
10. DewPointFarenheit
11. DewPointCelsius

12. RelativeHumidity

13. WindSpeed

14. Altimeter

3 PROJECT DEVELOPMENT

In this section, there is an overview of the process of data mining and data modeling, from collecting the data, through the data preparation and finally the data modeling. Data cleaning and formatting can be considered as the most critical part of the whole project according to several data scientists [10]. Figure 1 shows how the process of data mining works to extract knowledge using algorithms [11].

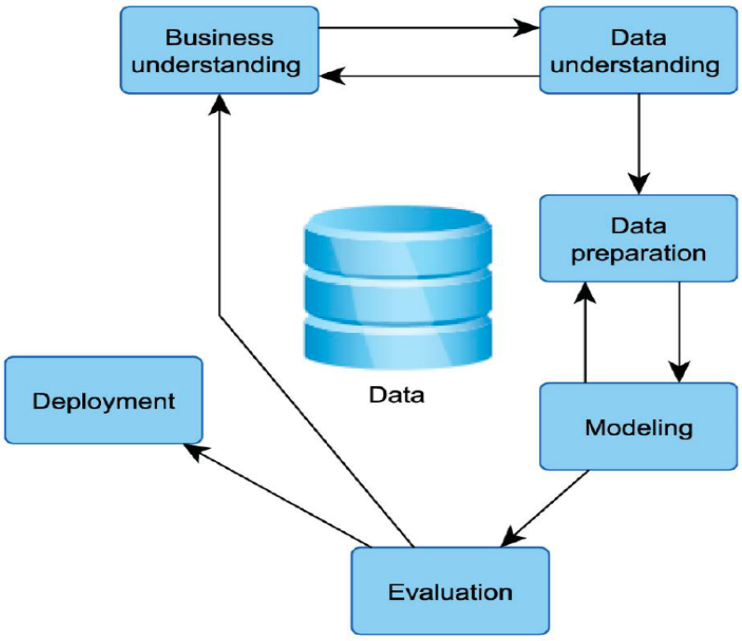

Figure 1: Project Development

4 DATA COLLECTION

Once the project undertaking is completely comprehended, our subsequent stage is to gather the information that is required for future model building. The information accumulation was an issue as data was not situated at a single source. The data was kept in unique information design. To achieve the end goal, it requires a clear understanding of the correct location of the data.

As we can see in Figure 2 [12], the US Bureau of Transportation Statistics gives detailed information on every single household flight, which incorporates their booking and take off circumstances and real takeoff, origin, destination, date, and carrier. We consolidated a portion of the information properties with Local Climatological Data from National Oceanic and Atmospheric Administration (NOAA) to shape a join data set. Since the datasets for every year are very massive, we decrease our concentration to one-year, i.e., 2008, which as of now contains 1 million records for the most significant airplane terminals. In this venture, we have taken 2007 as our preparation set and 2008 as our test set.

Handling speed is a noteworthy thought since the machine learning methodology that functions admirably on smaller datasets cause issues with the Jupyter Notebook establishments on our PCs.

	Number of Operations	% of Total Operations	Delayed Minutes	% of Total Delayed Minutes
On Time	5,473,439	73.42%	N/A	N/A
Air Carrier Delay	520,597	6.98%	28,827,070	28.55%
Weather Delay	72,307	0.97%	5,745,298	5.69%
National Aviation System Delay	598,258	8.02%	28,209,475	27.94%
Security Delay	4,939	0.07%	176,946	0.18%
Aircraft Arriving Late	607,928	8.15%	38,006,943	37.64%
Cancelled	160,809	2.16%	N/A	N/A
Diverted	17,182	0.23%	N/A	N/A
Total Operations	7,455,458	100.00%	100,965,732	100.00%

Figure 2: Data Collection

5 DATA EXPLORATION FOR FLIGHT DATA

As a first exploratory examination, we consider the watched likelihood of deferral in minutes on the whole dataset. The best path is through a histogram, taking a departure at flight and arrival delays independently.

5.1 Departure & Arrival Delay Distribution

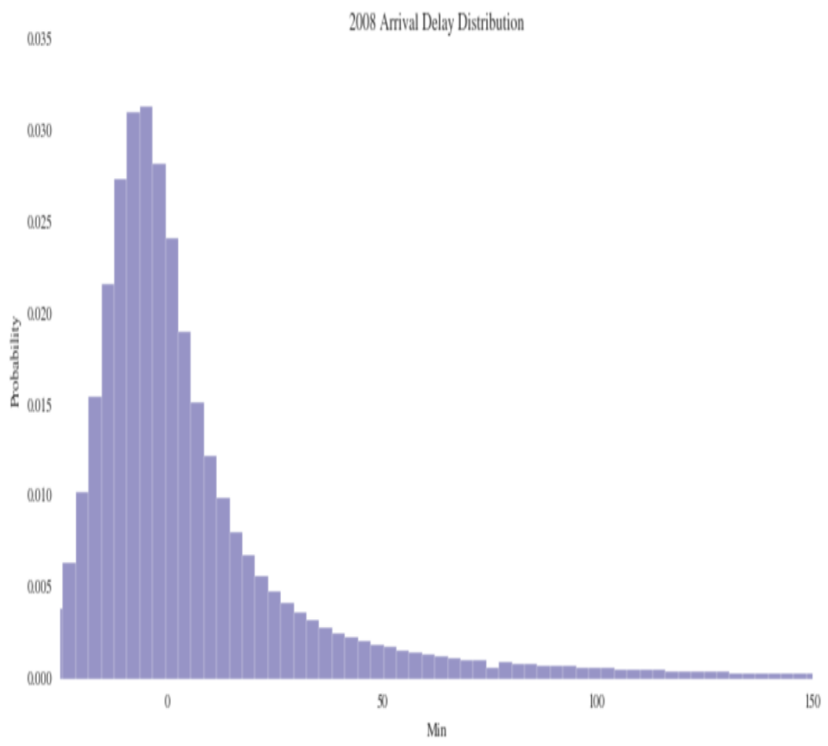

Figure 3: Arrival Distribution

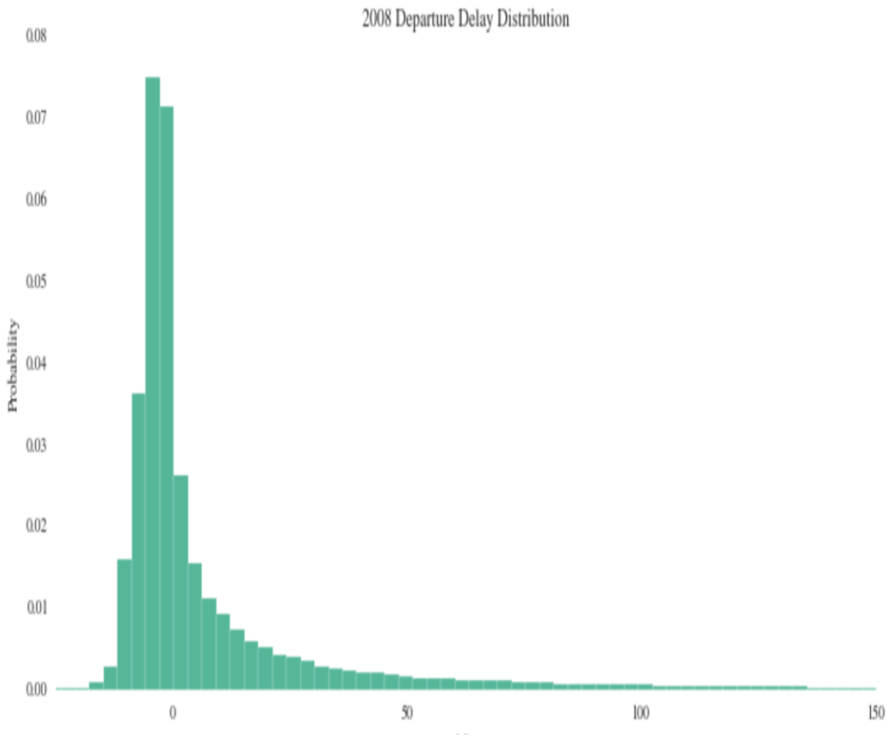

Figure 4: Delay Distribution

In Figure 3 and Figure 4, we notice a much higher probability of short delays. Notice the long right-hand tails from Figure 4. Some flights are delayed for very long time, e.g., over two hours. On the other hand, the delays were centered around just below zero. In both cases, the mode of the distribution is less than zero, meaning most of the flights leave from the gate and arrive at the gate even before the published schedule time of departure and arrival. As we will show below, the more extended delays cancel out, the shorter negative delays (advances), leading to average delays that are above zero.

The x-axis for the two plots is to scale. As a result, we can see that the arrival delay distribution, compared with the departure delay distribution, leans toward left. The scheduled time of an event defines a flight delay compared to the actual time of the event. Airlines usually put extra buffer time on a flight to ensure on-time arrival. Therefore, the departure delay and arrival delay distributions difference indicate that some departure delays were recovered during the flights due to the extra amount of time embedded in the flight time between two airports.

5.2 Average Departure & Arrival by Month

Next, we consider the impact of the months on the delays. We would expect that winter months have the most extended delay. A column chart with departure and arrival delay in minutes plotted by month is the most effective way to see the potential effects of the months.

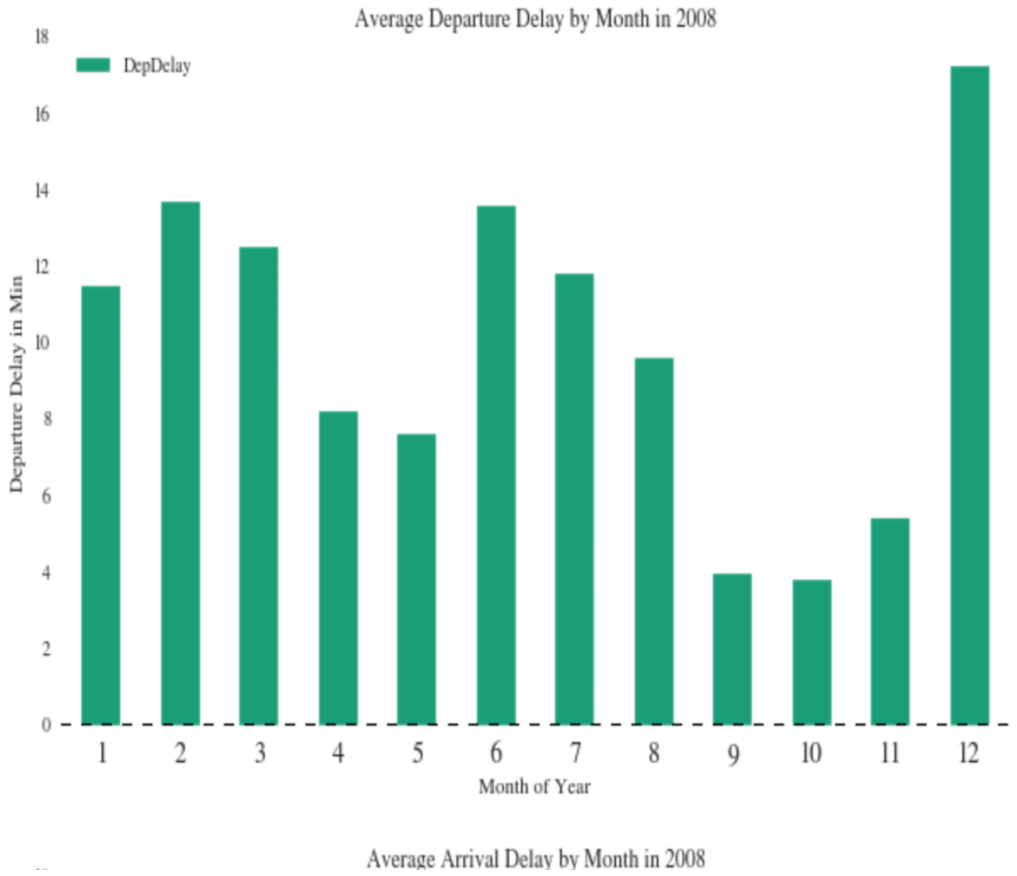

Figure 5: Average Departure Distribution

Figure 5 and 6 clearly show that for both departures and arrivals, the impact of December is clear - the highest delays are in that month. On the other hand, September, October, and November are the months with the least amount of delay. For the summer, June and July are marked by higher delays. Also, February posts high delay values as well. The reason for winter's high delay values is probably because of snowstorms in the northeast of the US.

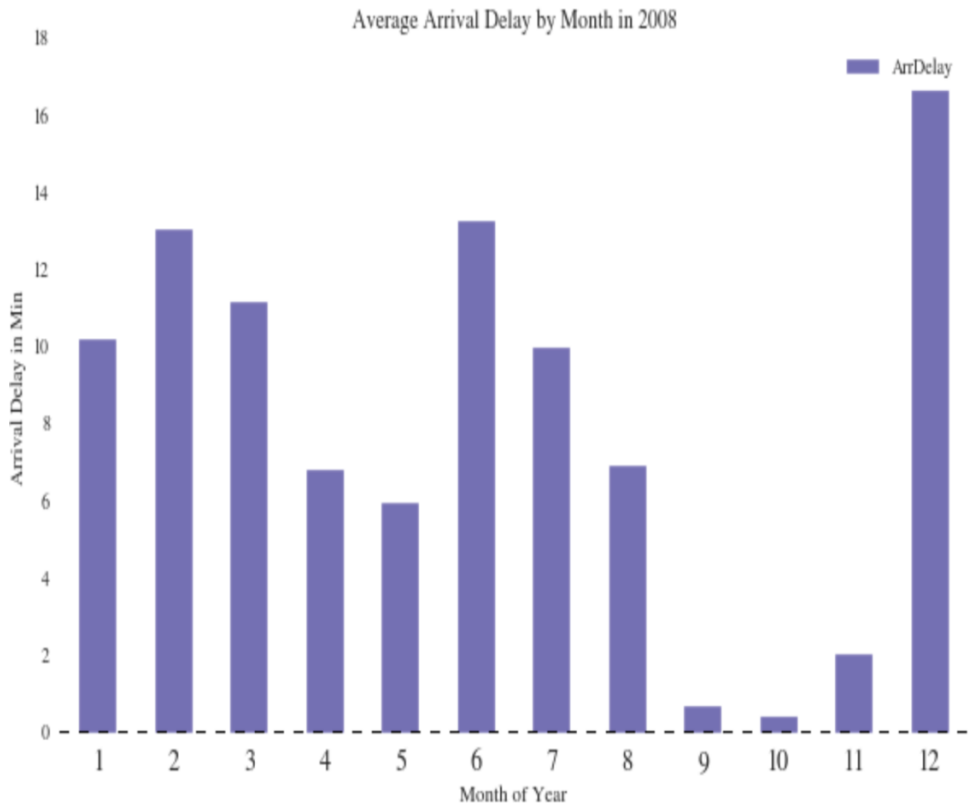

Figure 6: Average Arrival Distribution

Also, in Summer, thunderstorms in Chicago areas can cause high delay impact to the rest of the country. A snowstorm/storm may only affect operations at an airport or two. However, delay propagation, which marks as the significant contributor to the flight delay, can cause ripple effects on delay to downstream flight operations. The time of day should have an impact. Normally, flight delays cumulate throughout the day through a knock-on effect, where delayed flights provoke other delays because of tight schedules and runway congestion. We plot the mean delay by an hour of the day in a column chart.

5.3 Mean Delays across 4 Different Airports

For the next step, we have picked four significant airports and compare these distributions across these airports to see whether the impacts of month and hour of the day are similar at different airports. This analysis could be easily extended to all airports, but for the purposes here, it is sufficient to consider just four.

We subset the dataset into flights departing or arriving at Chicago O'Hare (ORD), Boston Logan (BOS), San Francisco (SFO) and New York LaGuardia (LGA).

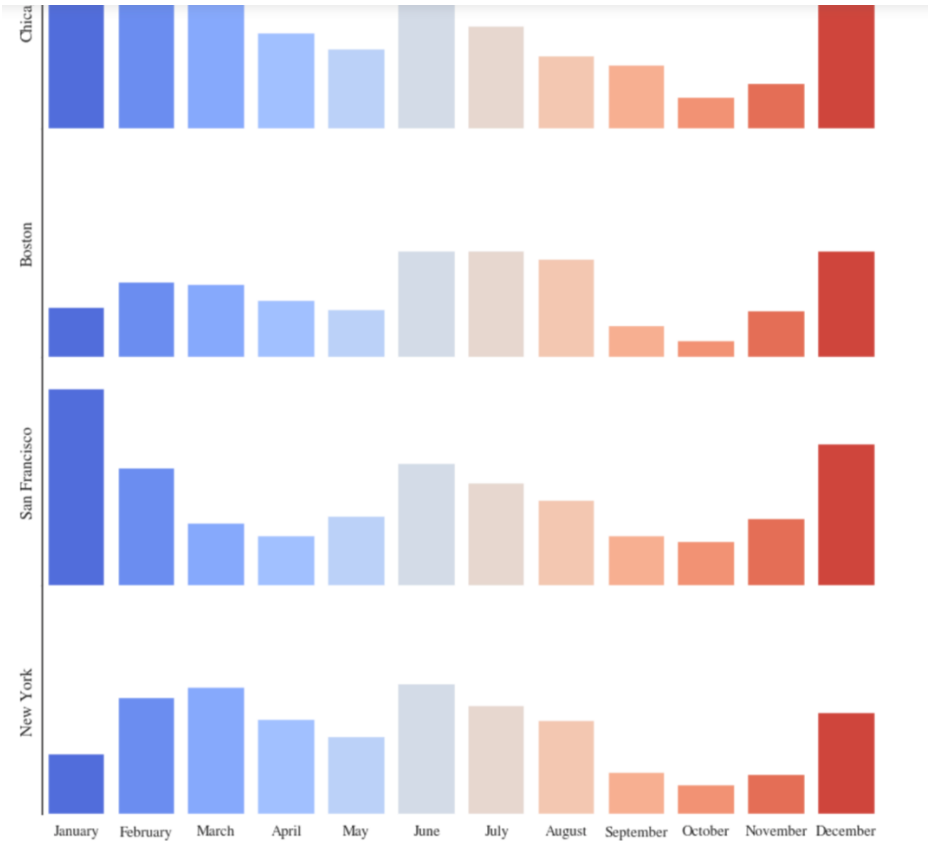

Figure 7: Departure Delays at Airport

First, we look into the departure delays from Chicago O'Hare (ORD), Boston Logan (BOS), San Francisco (SFO) and New York LaGuardia (LGA). As we can see in the Figure 7, the differences between Chicago O'Hare and San Francisco are similar to the overall profile for the mean delay at all airports, with higher delays in December and January and a midsummer bump. On the other hand, Boston Logan shows lower mean delays at the beginning of the year and more delays in all three summer holiday months. New York LaGuardia has many delays in the springtime in February, March, and June having a higher mean delay compared to December. In all locations, December is a month with higher than average delays.

Given that Northeast Corridor does receive a significant amount of rain/snow, further analysis is required for examining the cause and distributions (temporal and spatial relationship) of the flight departure delay.

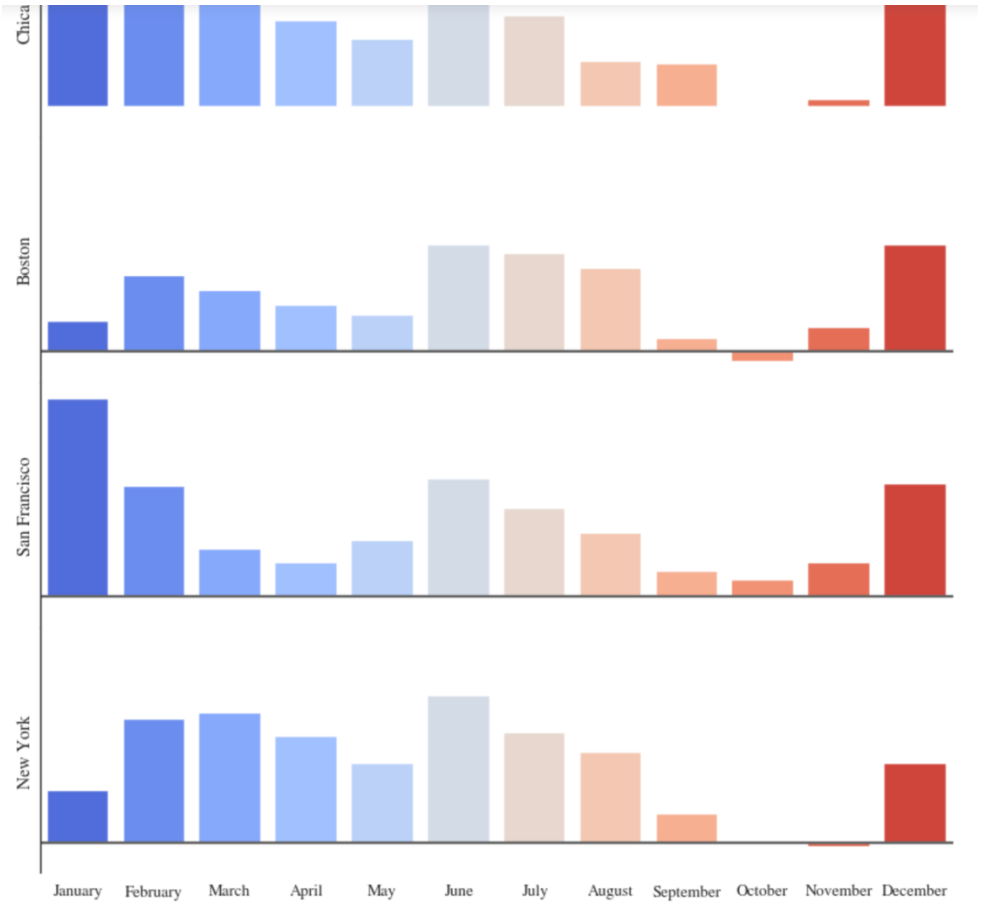

Figure 8: Arrival Delays at Airport

For arrival delays, we see four distinct peak months for Chicago O'Hare: December, January, February, and June in Figure 8. The latter may be because of holiday travel. The former three may be entirely weather related since these three months have the worst climatic conditions. San Francisco is similar to Chicago again. The peak delay is in January, possibly weather-related (bad winter conditions in January 2008). Still, the springtime peak in New York shows the most extended delays from February to

June. In all locations, the end of the year only has minimal delays in October and November, with delays rising back up in December.

6 PREDICTIVE MODELS FOR FLIGHT DELAY

Now, we have created several machine learning models to predict flight departure delays using 2008 data for flight departures from Chicago O'Hare International Airport (ORD) as an example. In this project, instead of predicting the quantity of flight departure delay, we opt to predict if a flight is delayed or not. A flight is on-time if the departure delay is within 15-min of the scheduled departure time (*CRSDepTime*). A flight is delayed if the departure delay is more than 15-min late from the scheduled departure time (*CRSDepTime*). We would like to build an analysis dataset by choosing the threshold of 15 minutes, beyond which we consider the class change to "delayed" flight. This is a standard threshold in the aviation industry, with indicators on delayed flights commonly based on 15 minutes of delay.

For flight departure delay prediction, the following features are potential candidates for the model:

1. Month
2. Day of Week (weekday vs. weekend)
3. Departure Hours (convert from (*CRSDepTime*))
4. Arrival Hours (convert from (*CRSArrTime*))
5. Departure Airport
6. Arrival Airport
7. CRSElapsedTime (total time for a flight)

8. Flight Distance

9. Carrier Name

In the following section, we choose the direction to consider - that is, whether Chicago ORD is the origin or the destination. Additionally, we decide what type of delay to consider (YColumns), either Departure delay measured by *DepDelay*, or the Arrival delay measured by *ArrDelay.* We can come back to this section to change the choices. We select a subset of columns of most interest (XColumns). *CRSArrTime* will be dropped since the hour is sufficient. Also, the *TaxiIn*, *TaxiOut* and *Diverted* variables will be dropped. *TailNum* is the tail number of the plane and could be interesting to match with plane information like the number of seats.

Since the dataset is so large, we have to constrain it so that Python can perform analysis on our system in a reasonable amount of time. To do so, the dataset is sampled randomly for 20k rows. Further, these observations have be randomly split into training and test sets, so that 10k observation is used for analysis.

6.1 Feature Metric

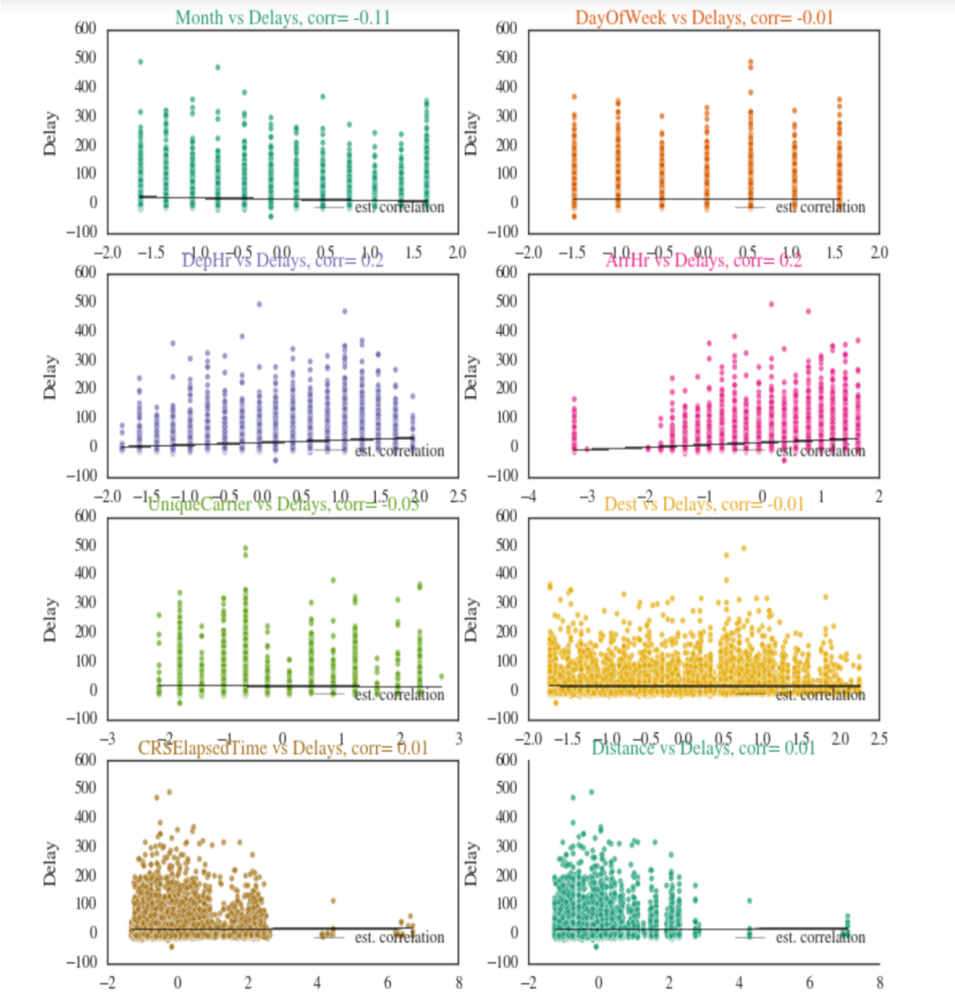

Figure 9: Feature Metric (Flight Data)

In Figure 8 none of them shows a clear correlation. In fact, the highest correlation coefficient is for *ArrivalHour*, because of the time of day effect that was noted earlier. This confounds by the early morning delays, which cause a spike at the lower end of the scatterplot. Also, *Carrier* and *Month* show some correlation. We have used the

Principal Component - the method that converts a number of correlated variables into a smaller number of uncorrelated variables. It is clear with these scatterplots that the Principal Component is mainly capturing the duration and distance characteristics.

6.2 Logistic Regression

Next, we move on to classification. The first model is a logistic regression model with L2 Regularization (Ridge Regression) penalty function [13]. We fit the training model that we randomly split 50-50 above. It is a relatively 'vanilla' classification that we shall use as a benchmark. Considering Ordinary Least Squares (OLS), that is the analysis used to estimate a relationship between a dependent variable and one or more independent variable. Given the OLS that we calculated, we are not sure that the linear regression model is the best choice for our problem, but it is good to evaluate against concerning of accuracy.

```
Confusion matrix
      0     1
0  4303  2906
1  1060  1731

precision = 0.37, recall = 0.62, F1 = 0.47, accuracy = 0.60

          0         1
0  0.596893  1.041204
1  0.147038  0.620208
```

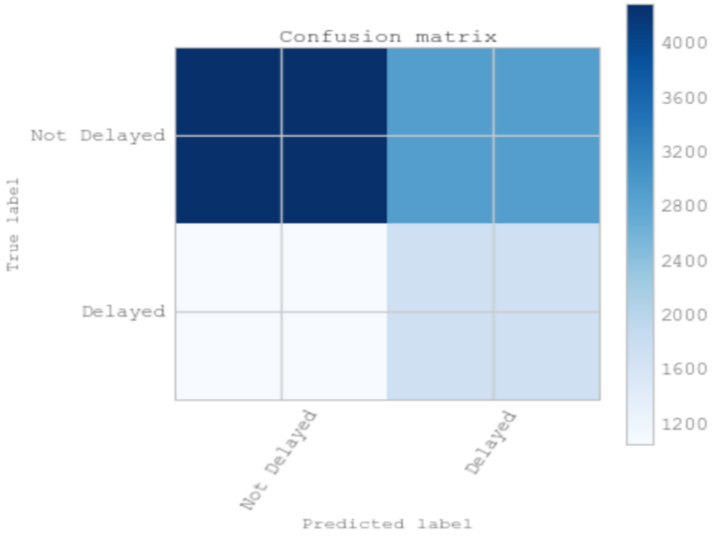

Figure 10: Confusion Matrix (Logistic Regression)

As we can see in Figure 10, the accuracy score of 60% is already encouraging, meaning that even this first classification model guesses right 60% of the time, better than a coin flip. We know that the positive case of a delay of more than 15 minutes is less frequent than no delay (around 2800 delayed instances to more than 7000 for no delay), so we may prefer to look at the F1 measure (47% here), which is calculated by using the harmonic mean of precision and recall.

6.3 Random Forest

We turn to a Random Forest classification [14]. To choose the optimal number of trees, we classify using 1 to 50 trees, and look at the mean, median and dispersion of the resulting accuracy measures.

```
Confusion matrix
      0     1
0  6270   939
1  1985   806

precision = 0.46, recall = 0.29, F1 = 0.36, accuracy = 0.71

           0         1
0   0.869746  0.336439
1   0.275350  0.288785
```

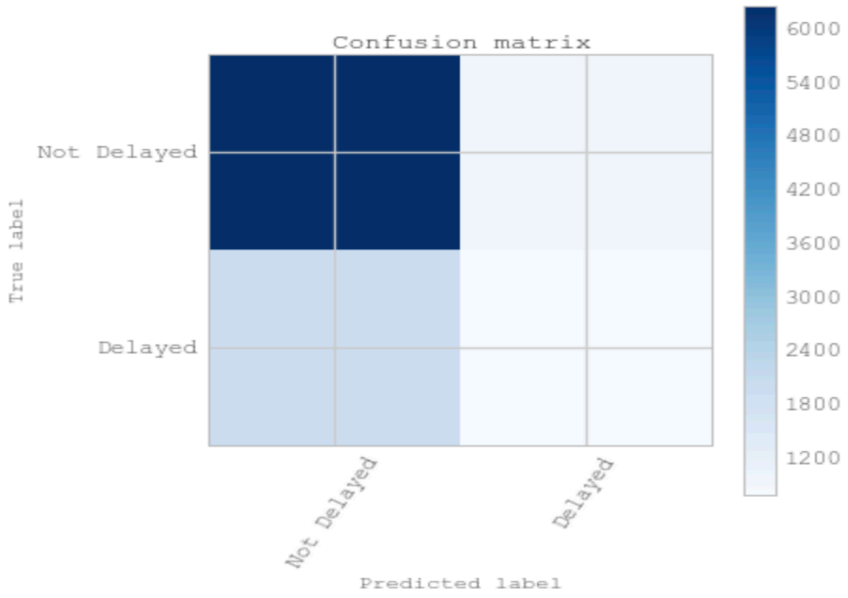

Figure 11: Confusion Matrix (Random Forest)

The left-hand side of the matrix is darker than the right, showing that the RF classifier guesses 'not delayed' more often than ‘delayed’ as shown in Figure 11. We are happy

with the improvement in accuracy but wondering how to improve the precision and the F1 score.

6.4 Support Vector Machine (SVM)

Next, we would like to consider a SVM classifier [15]. We will be first considering what might be considered a 'vanilla' choice of tuning parameters and RBF (Radial Basis Function) kernel, where C = 100 and gamma = 0.0001.

```
array([ 0.72127872,  0.72127872,  0.72127872,  0.72127872,  0.72127872,
        0.72172172,  0.72172172,  0.72172172,  0.72172172,  0.72172172])


(0.72127872127872128, 0.72150022150022153, 0.72172172172172178)
```

The SVM classifier does not seem to improve the performance by too much, with scores close to the RF classifier. The SVM requires a much longer time to compute. Given the small performance gain, we try to tune the SVM parameters, concentrating on the gamma parameter. We use a validation curve plotter on the results from the SciKit Shuffle Split routine using 10 values of gamma ranged from 10^{-5} to 10^{5}.

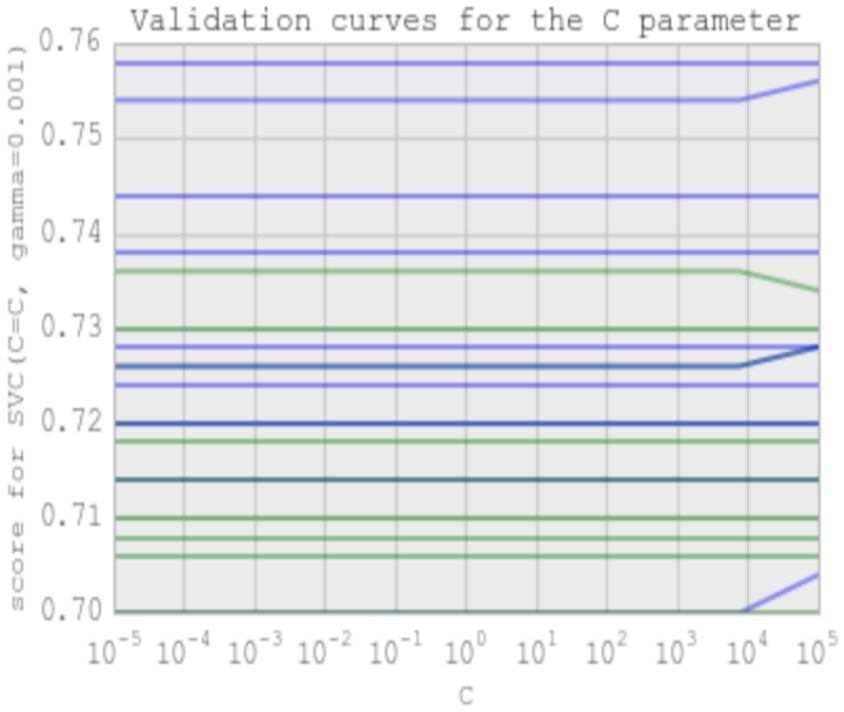

Figure 12: Validation Curve (Support Vector Machine)

Based on the results and the validation curves seen in Figure 12, we do not find that varying the gamma parameter have improved the model performance much. Given the results, we saw that looking at improving the RF classifier model is the best path to follow, next by encoding the qualitative variables into dummy variables that we had factorized previously into category numbers. SciKit Learn [16] has a OneHotEncoder routine that we can use for this. The advantage is that we can quickly transform the test variables based on the same encoding. By encoding the variables as dummies, the number of variables increases substantially.

```
Confusion matrix
      0    1
0  6522  687
1  2019  772

precision = 0.53, recall = 0.28, F1 = 0.36, accuracy = 0.73

          0         1
0  0.904702  0.246148
1  0.280067  0.276603
```

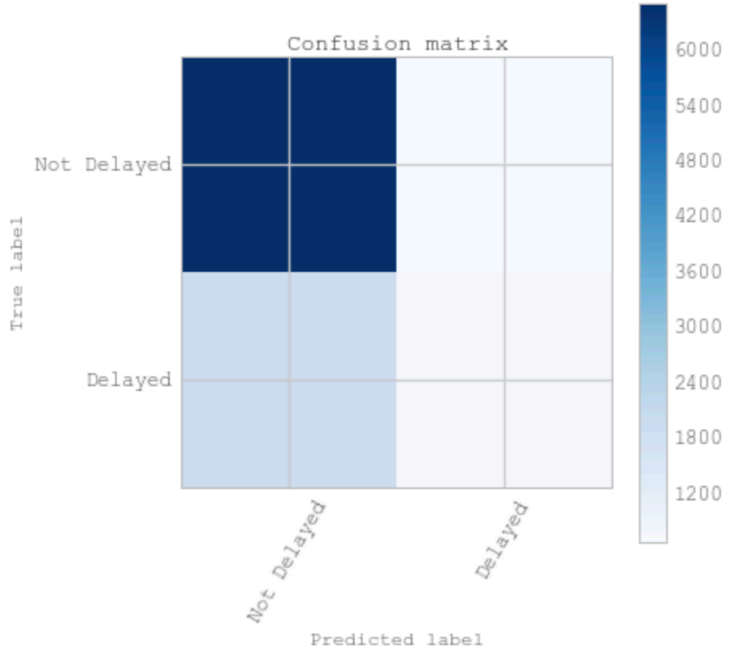

Figure 13: One Hot Encoder Confusion Matrix

The dummy encoding improved the accuracy score to 73% as seen in Figure 13. Precision slightly enhanced, and the F1 did not change. It seems that dummy encoding provides marginal performance gains. As for next steps, we need to look at adding explanatory variables. We have taken weather data and merge it to the 2008 database and then repeat the same steps

7 PREDICTIVE MODELLING ON WEATHER DATA

The dataset contains numerous factors that we might want to bar. The *Taxi in/out* variables and all the delay variables are to be dropped. We are most interested in the weather variables, so we decide to select only them from the dataset.

The new dataset contains variables for the scheduled operation at the airport - more or less an indicator of the demand per hour in the airport. This indicator is broken down into departures operations (flights per hour) and arrivals operations - both would count in determining the demand in any given hour at the airport, so we combine them into a sum of airport ‘demand’.

7.1 Feature Metric

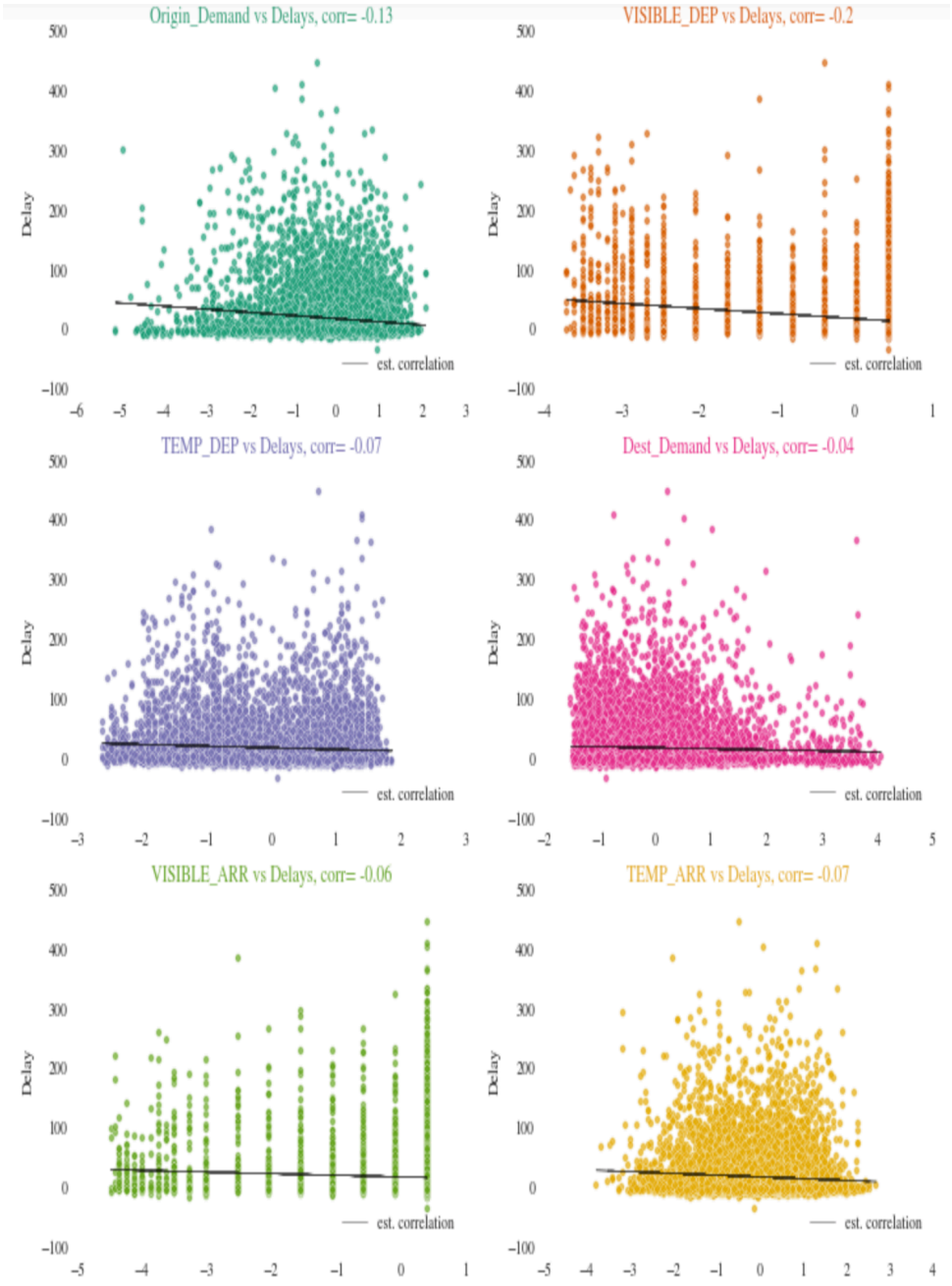

Figure 14: Feature Metric (Weather Data)

With the scatterplots of the new variables, we can see in Figure 14, that delay is correlated negatively. Also, *Visibility* is negatively correlated, which is intuitive, since as visibility worsens, more flights are delayed. As expected, the temperature at both the departure and arrival airports is negatively correlated with departure delays.

7.2 Logistic Regression Model

```
Confusion matrix
       0      1
0   4815   2337
1   1045   1803

precision = 0.44, recall = 0.63, F1 = 0.52, accuracy = 0.66

           0          1
0   0.673238   0.820576
1   0.146113   0.633076
```

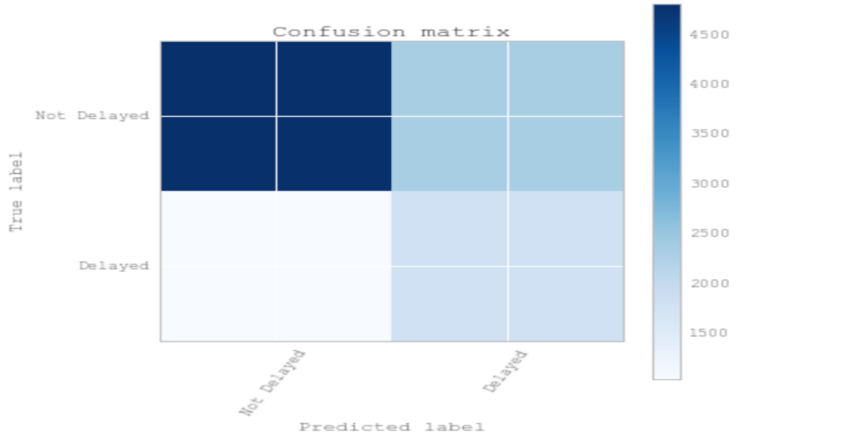

Figure 15: Confusion Matrix (Logistic Regression)

The accuracy has improved for the LR classification as seen in Figure 15, going from 60% to 66%. The F1 score is 52%. It appears that the additional features help to obtain better prediction performance.

7.3 Random Forest Classifier

```
Confusion matrix
      0     1
0  6615   537
1  1805  1043

precision = 0.66, recall = 0.37, F1 = 0.47, accuracy = 0.77

          0         1
0  0.924916  0.188553
1  0.252377  0.366222
```

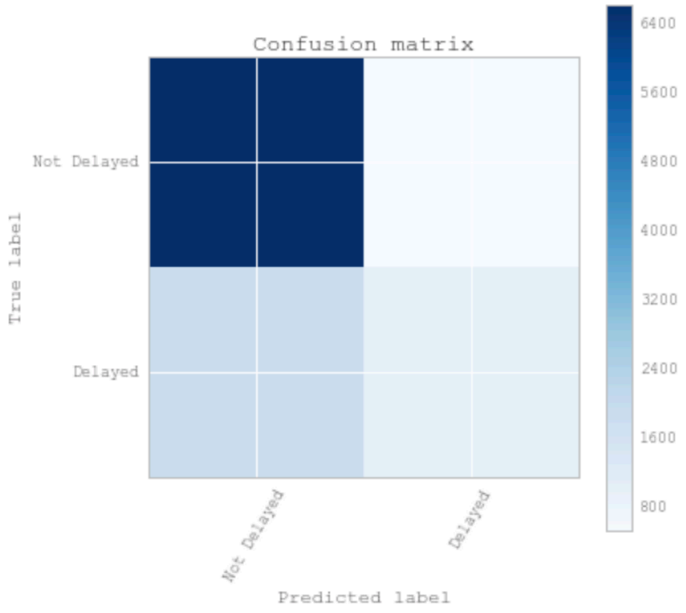

Figure 16: Confusion Matrix (Random Forest)

In Figure 16, the accuracy now has improved to 77%, which is a good gain. Precision has also improved to 66% from 53% before, and the F1 has also jumped to 47%. The new model seems to perform better.

7.4 Feature Importance

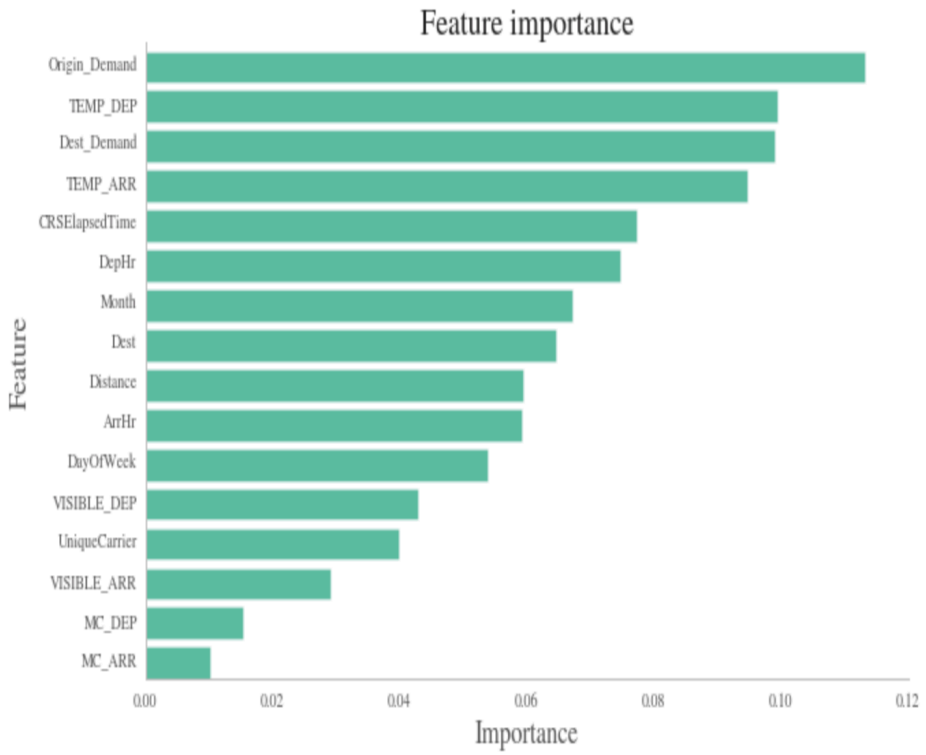

Figure 17: Feature Importance

The most important features are the *Demand* and *Temperature* attributes at the Departure and Arrival airports, followed by the duration and distance as seen in Figure 17. Departure airport attributes are generally more important than the arrival airport ones - something that should reverse in the analysis of arrival delays. Visibility and MC are low in importance, as is Carrier.

Let's look at cross-validation scores, using 10-fold CV and 50 trees.

```
clf_rf = RandomForestClassifier(n_estimators=50, n_jobs=-1)\nRF_scores =
cross_validation.cross_val_score(clf_rf, TrainX, np.where(TrainY >=
delay_threshold,1,0), cv=10, scoring='accuracy')\nprint RF_scores.min(),
RF_scores.mean(),
RF_scores.max()","execution_count":62,"outputs":[{"name":"stdout","output_typ
e":"stream","text":"
0.740740740741 0.756696705497 0.769230769231\n"}]}]
```

The previous score of 77% is among the highest. As we can see that in the minimum score of the model is 74% and the mean score comes out to be 76% which shows the model is robust.

from sklearn.svm import SVC

svc = SVC(kernel='rbf', C=100, gamma=0.001).fit(TrainX_scl,

np.where(TrainY >= delay_threshold,1,0))\nSVC_scores = cross_val_score(svc,

TrainX_scl, np.where(TrainY >= delay_threshold,1,0), cv=10,

scoring='accuracy', n_jobs=-1) \nprint SVC_scores.min(), SVC_scores.mean(),

SVC_scores.max()","execution_count":63,"outputs":[{"name":"stdout","output_

type":"stream","text":"

0.732267732268 0.7369003896 0.743256743257\n"}]}]

The scores from SVM are lower. This could be from the choice of tuning parameters, but it does not seem likely that more tuning would bring us above 77%, because if we increase the tuning parameters there is a risk of losing the generalization properties of the classifier. So, we concentrated on improving accuracy with dummy encoding for the categorical variables.

7.5 One Hot Encoding

One hot encoding is a process in which categorical variable is converted into a form type, and that form type can be used as an input to machine learning algorithms to perform better while making a prediction [17]. The input to this transformer should be a matrix of integers, denoting the values taken on by categorical (discrete) features.

7.6 Random Classifier with 35 Trees

```
Confusion matrix
      0    1
0  6693  459
1  1868  980

precision = 0.68, recall = 0.34, F1 = 0.46, accuracy = 0.77

           0         1
0  0.935822  0.161166
1  0.261186  0.344101
```

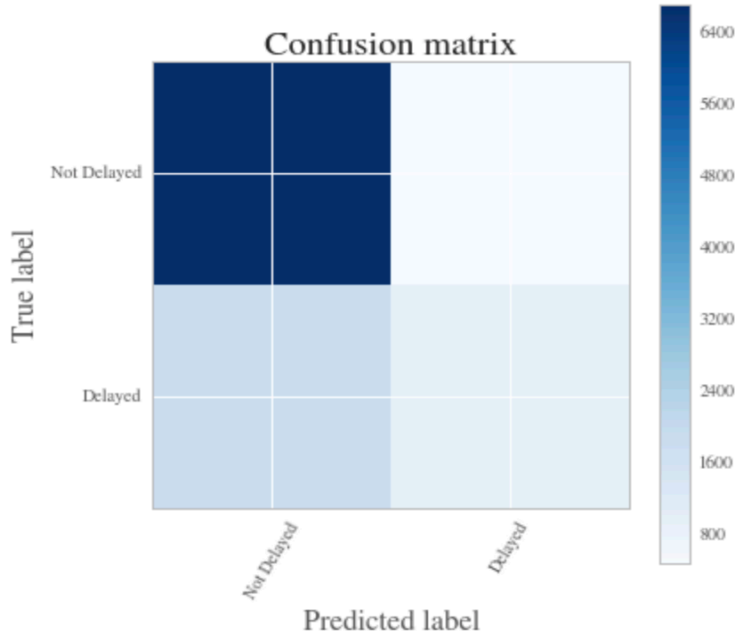

Figure 18: Random Classifier (35) Trees

In Figure 18, we increase precision from 67% to 68%, but accuracy stays the same, i.e., 77%. Finally, as the last step, we go back to the full dataset and perform classification using the RF with 50 trees and the new weather variables. We need to

split into train and test datasets randomly and then scale them (actually we do not need to mount for RF, but we do it anyway).

7.7 Random Classifier with 50 Trees

```
clf_rf = RandomForestClassifier(n_estimators=50, n_jobs=-1)
clf_rf.fit(TrainX, np.where(TrainY >= delay_threshold,1,0))
# Evaluate on test set
pred = clf_rf.predict(TestX)
# print results
cm_rf = confusion_matrix(np.where(TestY >= delay_threshold,1,0), pred)
print("Confusion matrix")
print(pd.DataFrame(cm_rf))
report_rf = precision_recall_fscore_support(list(np.where(TestY >=
delay_threshold,1,0)), list(pred), average='micro')
print "\nprecision = %0.2f, recall = %0.2f, F1 = %0.2f, accuracy =
%0.2f\n" % \
(report_rf[0], report_rf[1], report_rf[2],
accuracy_score(list(np.where(TestY >= delay_threshold,1,0)), list(pred)))
print(pd.DataFrame(cm_rf.astype(np.float64) / cm_rf.sum(axis=1)))
```

```
Confusion matrix
       0      1
0  72592   6988
1  18659  13373

precision = 0.66, recall = 0.42, F1 = 0.51, accuracy = 0.77

          0         1
0  0.912189  0.218157
1  0.234468  0.417489
```

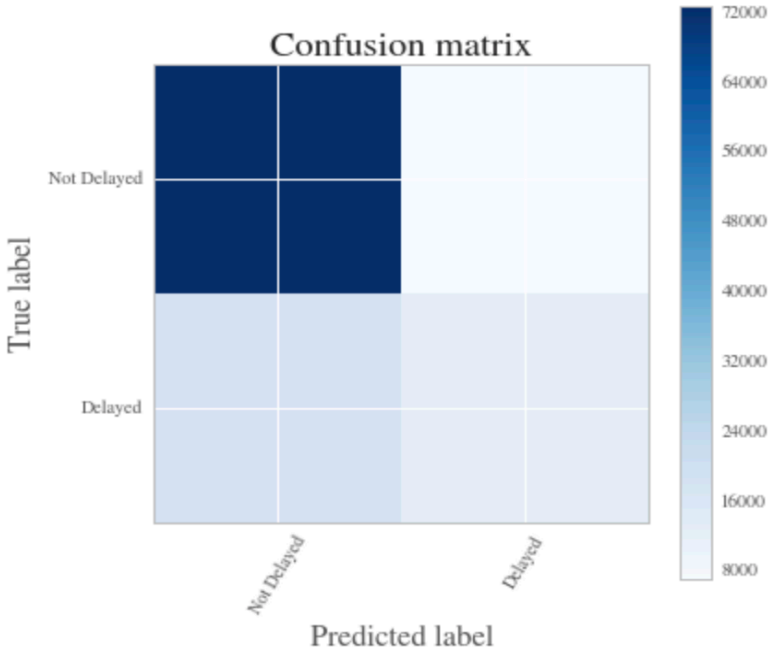

Figure 19: Random Forest (50) Trees

We get accuracy at 77%, and the F1 score is above 50%, as shown in Figure 19. Precision is at 66%. The model predicts 91% of the non-delayed flights and 41% of the delayed flights correctly. The selected model is good at predicting departure delays of more than 15 minutes.

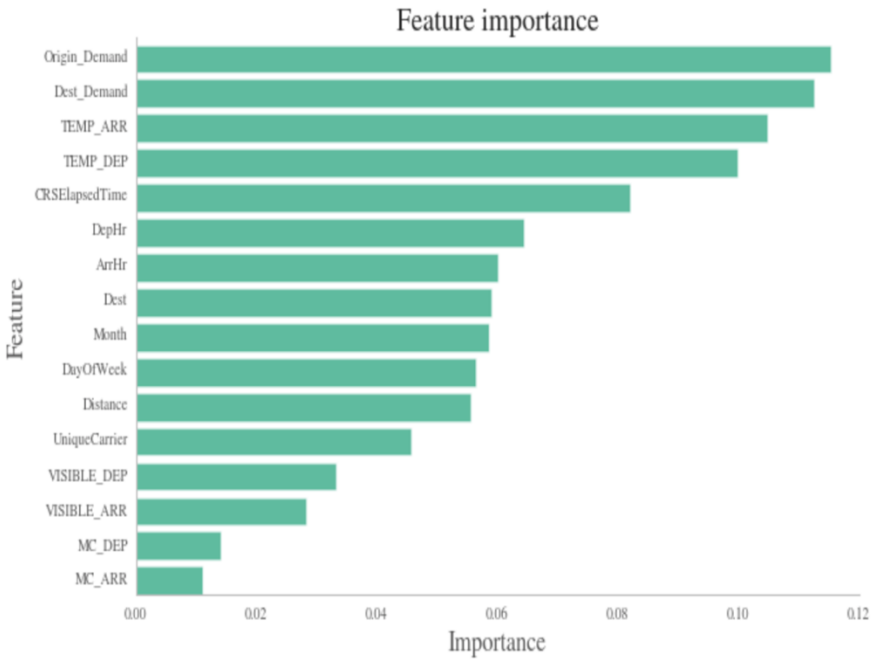

Figure 20: Feature Importance (Improved)

On the full dataset, origin and destination demand are the essential features, followed by the temperature at the destination and origin. Then duration and departure hour are the most important. Again, the *MC* and *Visibility* variables are the least important.

In Figure 20, the analysis has shown that we can arrive at reasonable prediction with the combination of flight origin and destination information such as time of day and the month. Adding weather and airport demand information improves the prediction. The Random Forest classifier seems to be the best for this flight delay problem.

8 PREDICTING DELAYS USING TWITTER

8.1 Introduction

This project aims to find methods to predict the probability of flight delays. Here, we are attempting to use tweets (posts on Twitter) to predict delays [18].

The premise is simple. People who are experiencing flight delays are bored and unhappy, and bored and miserable people tend to complain on Twitter. By analyzing the volume of tweets that mention a given airport, we believe that it may be possible to use these tweets as an indicator for delays that may otherwise not yet be published.

One factor that we analyze here is tweet volume. However, we go beyond this. After all, what if a person is tweeting because they're excited to go on a trip, and their tweets don't indicate unhappiness from a delay? To account for this, we employ sentiment analysis, using the TextBlob package [19]. This analyzes strings of text for emotion, either positive or negative. It also assesses whether a string is objective or subjective. We employ for this by looking for negative tweets, as well as unusually subjective tweets.

8.2 Data Exploration

At the start, we should note several issues we encountered along the way. First, Twitter does not currently allow users to archive tweets from an indefinite period. Instead, they only allow their API to access tweets from around the past week. Because of this, we were not able to work with the same data sets from the other part of the project, which focuses on 2008. Instead, we are working with a separate dataset for delays which we obtained, covering late November and early December of this year. This only allowed for an overlap of several days, in which we had both tweets and weather data. Second, Twitter does not provide all tweets to their API, just those that its algorithm determines are "interesting" or "popular." Therefore, we are working with a subset of all tweets, which may contain bias. For these reasons, this should be considered a proof of concept, as there was insufficient Twitter data available for a wide-ranging analysis, beyond what is presented here.

For this analysis, we focus specifically on predicting departure delays from LaGuardia Airport, in New York City.

8.3 Tweets Volume by Hour

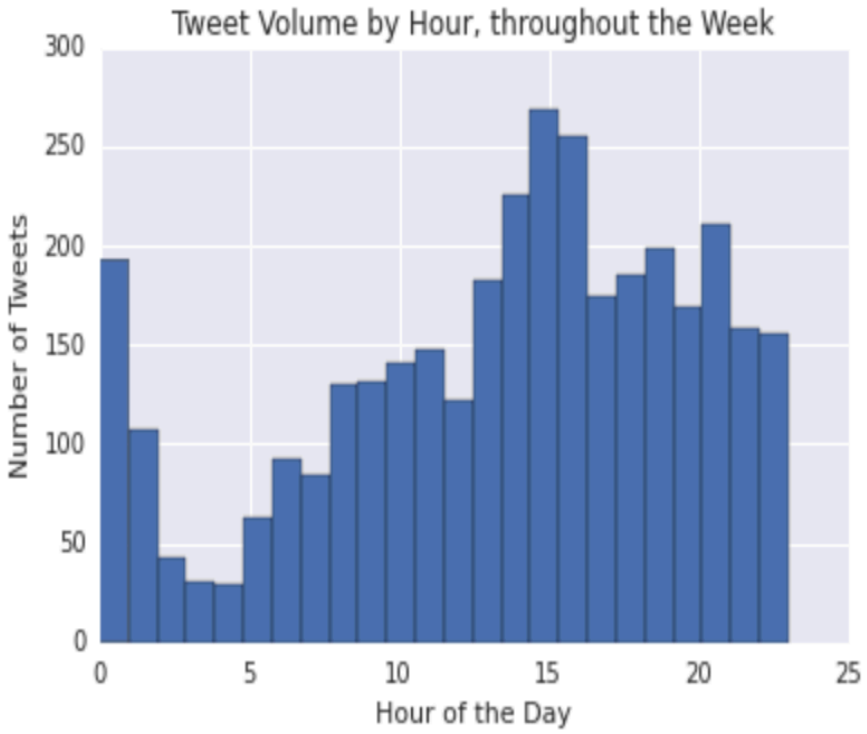

Figure 21: Tweets Volume by Hour

In Figure 21, a histogram is arranged by hour for all of the tweets over the past week that mention LaGuardia.

After this, we start importing a dataset with delay information for all of the major US airports since late November. It is very large (about 750 MB), so we have used another computer to isolate just the LaGuardia departures, and load them in. But when we plot the delays on December 4 by hour, unfortunately, the data had a "hole", where for a

few hours we don't have delay data for flights. It's not clear why the original data doesn't provide this, but we have worked with the data from the rest of the day.

8.4 Tweets Volume on a Day

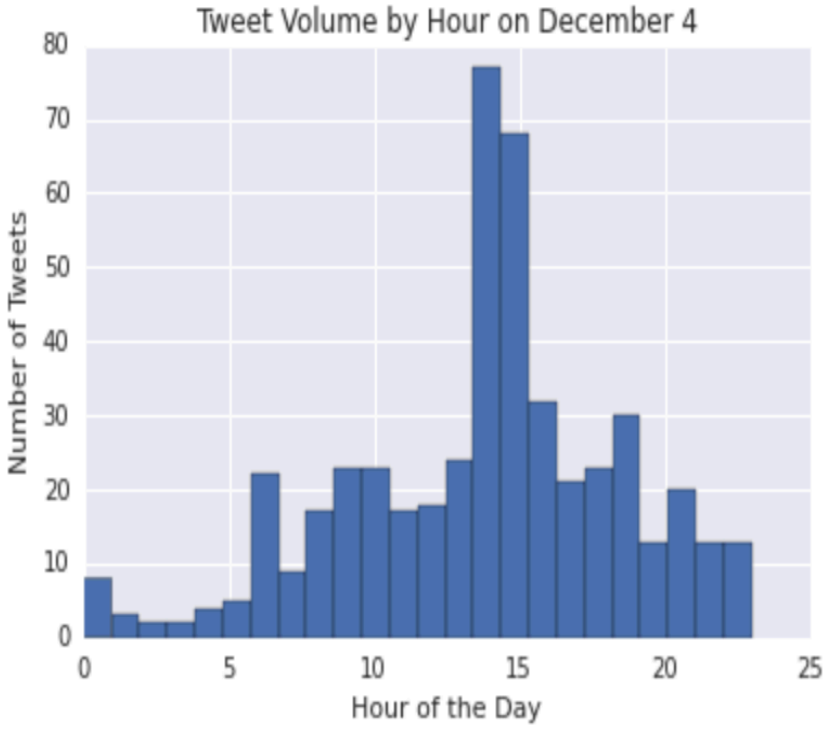

Figure 22: Tweets Volume on A Day

In Figure 22, it seems as though it may be skewed to the middle of the day; one concern for possible bias is spam, in which vast quantities of tweets may be posted at the same time. For a simple heuristic, let's assume that most posts containing a link are spams, and even the ones that don't spam are probably not complaining about delays.

Therefore, we should be able to remove all tweets containing a link safely. We can also remove tweets that are duplicates of others because those are likely spams too.

8.5 Removing Duplicates Tweets

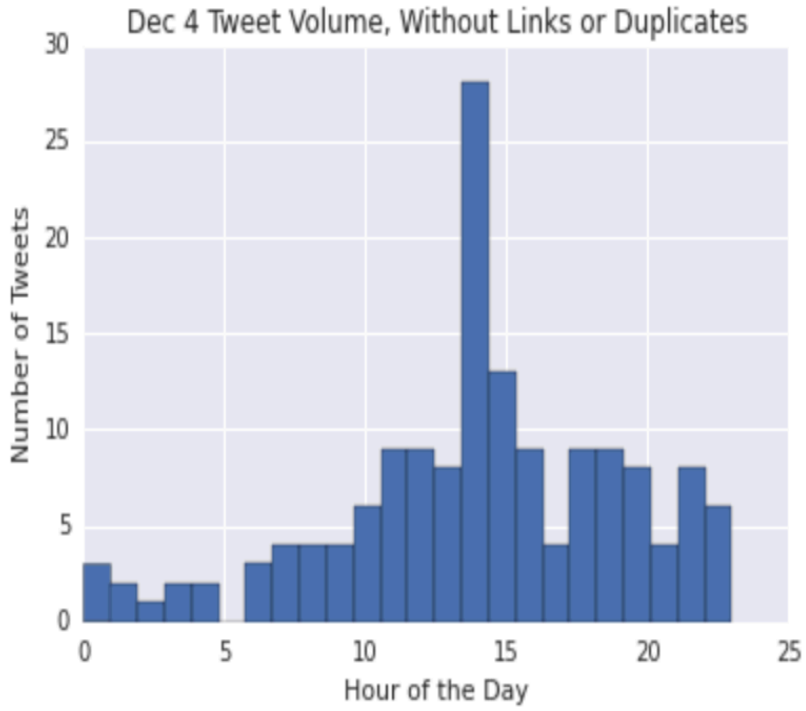

Figure 23: Tweets Volume Without Duplicates

In Figure 23, even though we removed a lot of potentially irrelevant tweets, we still have more to do. We need to address the concern that many of the tweets in the above group are not complaining about delays. Anecdotally, many of the tweets that complain about delays do so without mentioning the word "delay", so filtering for that

keyword would not be very helpful. Instead, we're going to find these tweets through sentiment analysis.

The package we use, TextBlob, provides two separate assessments about a string. The first is polarity: this measures happiness versus unhappiness, with a scale from -1 (most unhappy) to 1 (most happy). The second is subjectivity, on a scale from 0 (most objective) to 1 (most subjective).

8.6 Tweets for Negative Sentiment

First, we have analyzed the tweets for negative sentiment.

To begin, we need to decide how extreme the negative tweets should be in order to be included in our set [20]. Therefore, we need to choose a threshold for polarity, between -1 and 0, such that only tweets below that threshold get included in the set. Let's plot the volume of tweets at each threshold.

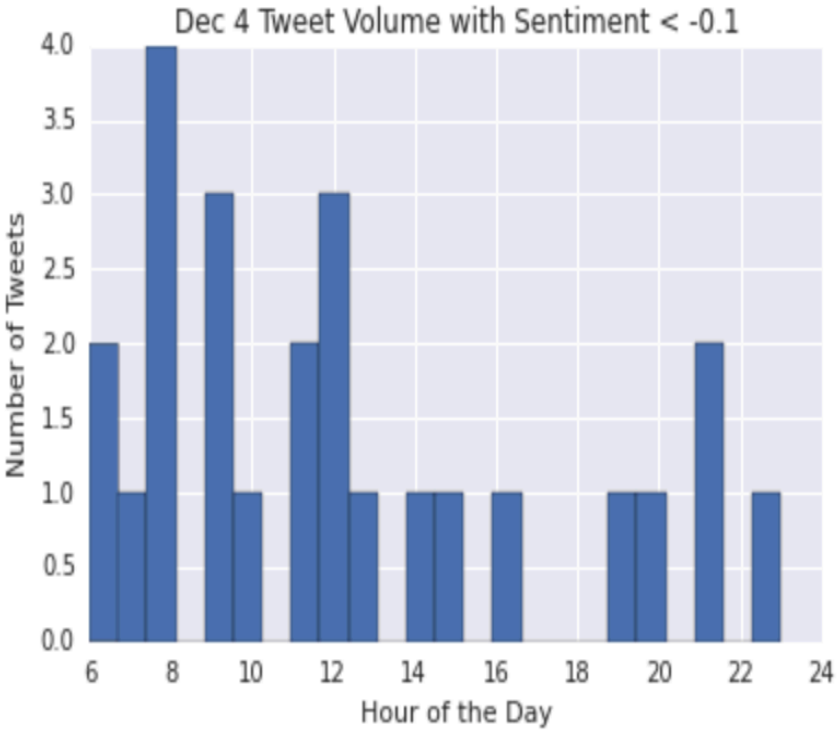

Figure 24: Tweets Volume with Sentiment <-0.1

In Figure 24, we still don't have a vast number of tweets per hour but was enough for a machine learning algorithm. Given that we are not working with a massive volume of tweets, rather than making our set more selective, we have left it at -0.1.

8.7 Tweets for High Subjectivity

Next, we have analyzed the tweets for high subjectivity.

As we did above, first comparing the volume of tweets at each subjectivity threshold. In Figure 25, possible subjectivity ranges from 0 (entirely factual) to 1 (thoroughly opinionated) [21]. At each limit, we have considered the size of the set of tweets which are at or above that threshold.

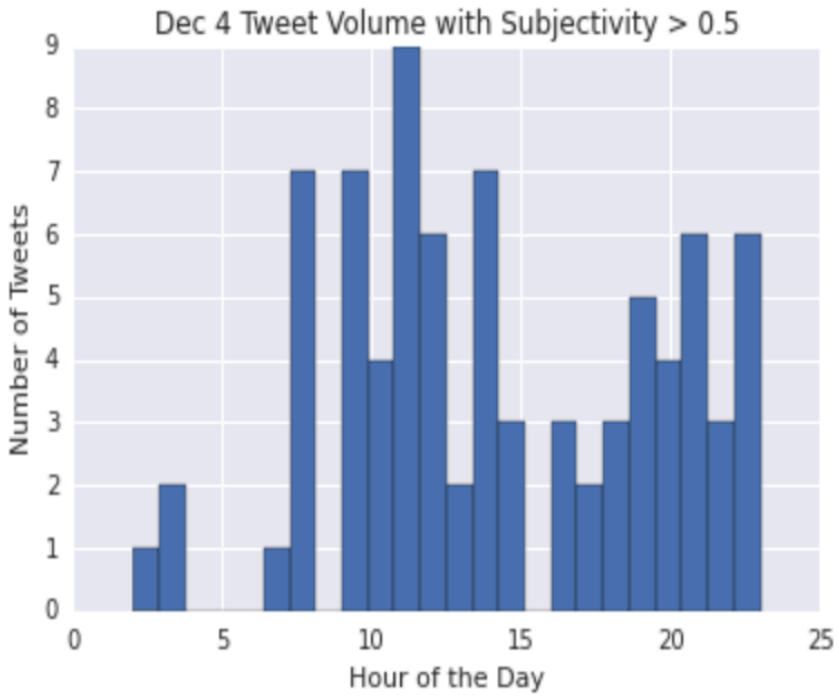

Figure 25: Tweets Volume with Subjectivity >0.5

At this threshold level, we have enough tweets to have much variation hour-to-hour. We have kept this threshold for the rest of the analysis. By observation, the graph for subjectivity seems to track much more closely to the delay patterns than the graph for negative sentiment. Both analyses, plus the total tweet volume, may be useful for the machine learning algorithms.

Now, to try the proof of concept, we are going to try to predict delays using machine learning. We have assembled a matrix with the three relevant factors for each hour -- tweet volume, number of tweets with negative sentiment, and number of subjective tweets.

We have a concise range of time for which we have both tweets and data. Therefore, we have set a small goal: Using the tweets and delay data from December 4 as a training set, how well can we predict delays for December 5?

First, we have created a Data Frame with the same results as before. After creating a Data Frame, we have added the average delay time per hour, and finally we removed the rows for which we do not have delay data. Now we created a training set from the actual delay data. Rather than trying to predict delays as a continuous variable, we predicted delays at the point that they become problematic -- perhaps whether they exceed 20 minutes. Since we have worked with each hour of the day separately, we have tried to predict whether the average delay time in a given hour will be more considerable or less than 20 minutes.

8.8 Random Forest

```
Y=pd.DataFrame.transpose(pd.DataFrame([pd.concat([pd.DataFrame([Y_test])[i] for i
in pd.DataFrame([Y_test])], ignore_index = True), forest_predictions]))
true1 = 0
false1 = 0
true0 = 0
false0 = 0
for x in range(0, len(Y)):
if Y.ix[x,0] == 0:
if Y.ix[x,1] == 0:
true0 = true0 + 1
else:
false1 = false1 + 1
else:
if Y.ix[x,1] == 1:
true1 = true1 + 1
else:
false0 = false0 + 1
print '                Confusion Matrix'
print ' '
print '         ', 'No Delay (Predicted)  ', 'Delay (Predicted)'
print 'No Delay   ', true0, '                    ', false1
print 'Delay      ', false0, '                    ', true1
```

Confusion Matrix

	No Delay (Predicted)	Delay (Predicted)
No Delay	1	1
Delay	4	11

Figure 26: Random Forest Confusion Matrix

In Figure 26, Random Forest algorithm had some mistakes. It predicted the majority of the delayed hours correctly, but only one of the two instances of the on-time hours.

8.9 Support Vector Machine

Confusion Matrix

	No Delay (Predicted)	Delay (Predicted)
No Delay	2	0
Delay	1	14

Figure 27: Support Vector Machine Confusion Matrix

In Figure 27, the results are slightly better using the SVM. From the perspective of delays: there were no false positives, one false negative, fourteen true positives and two true negatives.

9 CONCLUSION

In this project, we use flight data, weather, and demand data to predict flight departure delay. Our result shows that the Random Forest method yields the best performance compared to the SVM model. Somehow the SVM model is very time consuming and does not necessarily produce better results. In the end, our model correctly predicts 91% of the non-delayed flights. However, the delayed flights are only correctly predicted 41% of time. As a result, there can be additional features related to the causes of flight delay that are not yet discovered using our existing data sources.

In the second part of the project, we can see that it is possible to predict flight delay patterns from just the volume of concurrently published tweets, and their sentiment and objectivity. This is not unreasonable; people tend to post about airport delays on Twitter; it stands to reason that these posts would become more frequent, and more profoundly emotional, as the delays get worse. Without more data, we cannot make a robust model and find out the role of related factors and chance on these results. However, as a proof of concept, there is potential for these results. It may be possible to routinely use tweets to ascertain an understanding of concurrent airline delays and traffic patterns, which could be useful in a variety of circumstances.

10 FUTURE WORK

This project is based on data analysis from year 2008. A large dataset is available from 1987-2008 but handling a bigger dataset requires a great amount of preprocessing and cleaning of the data. Therefore, the future work of this project includes incorporating a larger dataset. There are many different ways to preprocess a larger dataset like running a Spark cluster over a server or using a cloud-based services like AWS and Azure to process the data. With the new advancement in the field of deep learning, we can use Neural Networks algorithm on the flight and weather data. Neural Network works on the pattern matching methodology. It is divided into three basic parts for data modelling that includes feed forward networks, feedback networks, and self-organization network. Feed-forward and feedback networks are generally used in the areas of prediction, pattern recognition, associative memory, and optimization calculation, whereas self-organization networks are generally used in cluster analysis. Neural Network offers distributed computer architecture with important learning abilities to represent nonlinear relationships.

Also, the scope of this project is very much confined to flight and weather data of United States, but we can include more countries like China, India, and Russia. Expanding the scope of this project, we can also add the flight data from international flights and not just restrict our self to the domestic flights.

BIBLIOGRAPHY

[1] A. B. Guy, "Flight delays cost $32.9 billion, passengers foot half the bill". [Online] Available : https://news.berkeley.edu/2010/10/18/flight_delays/3/. [Accessed on June 2017].

[2] M. Abdel-Aty, C. Lee, Y. Bai, X. Li and M. Michalak, "Detecting periodic patterns of arrival delay", *Journal of Air Transport Management,,* Volume 13(6), pp. 355–361, November, 2007.

[3] S. AhmadBeygi, A. Cohn and M. Lapp, "Decreasing Airline Delay Propagation By Re-Allocating Scheduled Slack", *Annual Conference*, Boston, 2008.

[4] A. A. Simmons, "Flight Delay Forecast due to Weather Using Data Mining", M.S. Disseration, University of the Basque Country, Department of Computer Science, 2015.

[5] S. Choi, Y. J. Kim, S. Briceno and D. Mavris, "Prediction of weather-induced airline delays based on machine learning algorithms", *Digital Avionics Systems Conference (DASC), 2016 IEEE/AIAA 35th*, Sacramento, CA, USA, 2016.

[6] L. Schaefer and D. Millner, "Flight Delay Propagation Analysis With The Detailed Policy Assessment Tool", *Man and Cybernetics Conference*, Tucson, AZ, 2001.

[7] B. Liu "Sentiment Analysis and Opinion Mining Synthesis", *Morgan & Claypool Publishers,* p. 167, 2012.

[8] Statistical Computing Statistical Graphics. [Online]. Available: http://stat-computing.org/dataexpo/2009/the-data.html. [Accessed on April 2017].

[9] FAA Operations & Performance Data. [Online]. Available: https://aspm.faa.gov/.[Accessed on April 2017].

[10] B. Bailey, "Data Cleaning 101". [Online]. Available: https://towardsdatascience.com/data-cleaning-101-948d22a92e4. [Accessed on March 2018].

[11] P. Panov, L. Soldatova and S. Džeroski, " OntoDM-KDD: Ontology for Representing the Knowledge Discovery Process", *Discovery Science 2013,* Volume 8140, pp. 126-140, 2013.

[12] Bureau of Transportation Statistics. [Online]. Available: https://www.transtats.bts.gov/carriers.asp. [Accessed on 2 April 2017].

[13] How to Predict Yes/No Outcomes Using Logistic Regression. [Online]. Available: https://blog.cleaarbrain.com/posts/how-to-predict-yesno-outcomes-using-logistic-regression [Accessed on 3 Feubrary 2018].

[14] S. Polamuri, "How The Random Forest Algorithm Works In Machine Learning". [Online]. Available: https://medium.com/@Synced/how-random-forest-algorithm-works-in-machine-learning-3c0fe15b6674. [Accessed January 2018].

[15] S. Ray, "Understanding Support Vector Machine algorithm". [Online]. Available: https://www.analyticsvidhya.com/blog/2017/09/understaing-support-vector-machine-example-code/.[Accessed November 2017].

[16] OneHotEncoder. [Online]. Available: http://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.OneHotEncoder.html. [Accessed on March 2018].

[17] R. Vasudev, "Why and When do you have to use OneHotEncoder?".[Online]. Available: https://hackernoon.com/what-is-one-hot-encoding-why-and-when-do-you-have-to-use-it-e3c6186d008f. [Accessed on March 2018].

[18] Twitter API Twitter. [Online]. Available: https://developer.twitter.com/en/docs.

[19] S. Loria , "TextBlob: Simplified Text Processing", 2016. [Online]. Available: http://textblob.readthedocs.io/en/dev/ [Accessed on December 12, 2017].

[20] A. Agarwal, B. Xie, I. Vovsha, O. Rambow and R. Passonneau, "Sentiment Analysis of Twitter Data," Columbia University, New York, December, 2011.

[21] V. A. Kharde and S. Sonawane, "Sentiment Analysis of Twitter Data: A Survey of Techniques", *International Journal of Computer Applications (0975 – 8887),* Volume 139, no.11, p.11, April 2016.