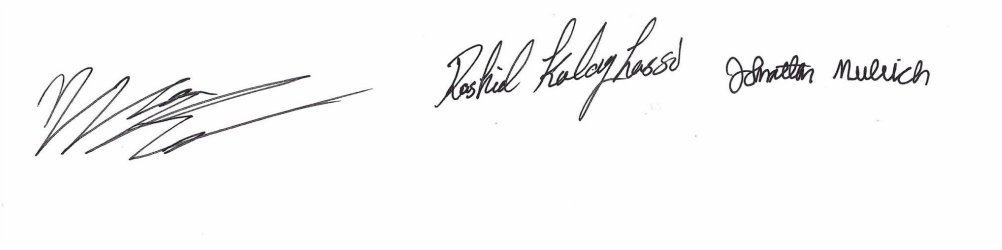
# **Air traffic congestion at 30 major U.S. Airports** during the winter holiday months

## A visual representation and prediction of data for potential redirection of Government funding

Martin Michael Dimo
Rashid Kolaghassi
Jonathan Muhvich

This work is authentic and has not been plagiarized.



## Introduction:

As air travel rose in prominence, early investors were very eager to gain capital from the new, soon to be a vital market. As early as 1924, wealthy entrepreneurs started building private airports across the nation. However, with the 1930s bringing the Great Depression to existence and subsequently sparking new deal policies, the private airport sector saw a devastating sweep of heavy government provided aid. This meant that by the end of the 1930s and well into the 1940s, cities across saw the monetary advantage of owning an airport and made almost all 1,100 private airports at that time to government-owned airports. Ever since then, congress has been taxing all forms of aviation. The advantage of making popular airports go public was the introduction of government laws such as the Federal Airport Act in 1946, which was the first fund available to airports that distributed 500 million dollars over seven years. In 1970, the Airport and Airway Trust Fund (AATF) was formed as an official path for the distribution of funds gathered by taxes and fees to reach air traffic control facilities and local and state airports. In recent history, the AATF raises 15 billion dollars annually from a 7.5 percent tax on domestic airline tickets, taxes on aviation fuels, international departure and arrival taxes, and a number of other charges.

While taxation may stay prominent, air travel is becoming cheaper and cheaper every year. With more travelers taking the skies in the holiday season, airports and airways experience increasing amounts of congestion and delays. Whether it be due to unpredictable weather patterns or technical difficulties, delays pose a problem to the general flow of travel. With many airports being out of date when it comes to terminal capacity and amenities, this data report aims to predict which airports struggle the most.

For the scope of this analysis, only three months will be taken into account: November, December, and January. From now on these months will be referred to as the holiday season, and it shall be assumed only these three months are used for data analysis. By analyzing arrival data from 30 of the U.S. largest airports, this paper aims to predict which airport will suffer from the busiest congestion and thus, required immediate attention. The data used was collected by the U.S. Department of Transportation, specifically the Bureau of Transportation Statistics, which provides total arrived flights as well as total delay flights and causes of said delays for all U.S. airports. The data set is linked below, from which a raw data set between September 2018 to September 2019 was downloaded for this project. This paper is aimed at government bureaus to help them predict which U.S. airports currently need the most amount of attention in terms of maintenance and funding. The end goal is to minimize delays due to unnatural causes to alleviate traveler stress and make the holiday season more enjoyable for everyone.

*Data set:*

https://www.transtats.bts.gov/OT_Delay/OT_DelayCause1.asp?pn=1

https://www.transtats.bts.gov/OT_Delay/print_ot_delaycause1.asp?pn=1

## Article on bias in Machine Learning:

Machine learning is not a new tool, but it is an extremely important one. If used properly it can be a very powerful tool for finding patterns, trends, and even for making predictions. Programs that utilize this tool typically work with very large data sets, and can make easy work out of what would otherwise be a simple task. But if used wrong, or carelessly, machine learning may give false results that give rise to wrong conclusions. We call this bias. Bias can happen for a few reasons.

The first reason is sample bias. This is what we call it when your data set does not represent what you are trying to model or predict. Take for example a program that is being used to model temperature patterns in New England. If it were given weather data from California, the model would be entirely off. This type of bias can happen for a number of reasons, but it is up to whoever may be testing the data to spot and address.

Another reason may be prejudice. This bias is entirely human and can be difficult to spot, especially if all the work is being done by a few people. Prejudicial bias has gotten a fair amount of attention lately, due to infamous examples like Amazon's facial recognition software that falsely matched congress members with mugshots. Again with this bias, the program can do nothing to correct for this, and the people who are entering the data must make sure they have fair sets.

There is also systematic distortion in data. This often occurs when the instruments used to gather the data have a flaw. As a result, all of the data will end up skewed. If the skew is subtle detecting this type of bias can be very difficult. If the skew is random, a large enough data set may ultimately workout the bias through the law of large numbers. But if the data is skewed in a single direction, the entire result will be thrown off regardless of data size.

Work Cited:

"Three Ways Biased Data Can Ruin Your ML Models." *Datanami*, 16 July 2018, https://www.datanami.com/2018/07/18/three-ways-biased-data-can-ruin-your-ml-models.

## Abstract:

The goal of this project is to interpret the flight delay data for 30 major airports in the United States during the time periods of 2018-2019. We then developed a model using machine learning to predict flight delay data for 2018-2019 based on flight delay data for the past 5 years. The aim was to interpret the current state of the worst US airports in terms of flight delays to recommend increased federal spending on these facilities. The model was used to predict whether this trend will hold for the next year to judge whether this spending is warranted in the eyes of the federal government.

## Data Scrubbing:

The first step in processing our data was scrubbing the data set. The website we imported data from eased the process as it allowed us to specify the time period and airports we wanted to analyze. Considering the goal of this project, we saw it fit only import the data for the largest 30 airports in the US, as flight delays in these airports affect the largest amount of passengers. We also only imported the data for the winter holiday months, November, December, and January, as it is the time period that witnesses significant uptick in air traffic, and any flight delays will cause a chain effect in causing more delays as aircraft miss their scheduled landing time at a runway and start interfering with the times of other scheduled aircraft arrivals. The raw data imported looked like this:

| year | month | carrier | carrier_nam | airport | airport_nam | arr_flights | arr_del15 | carrier_ct | weather_ct | nas_ct | security_ct | late_aircraft | arr_cancelle | arr_diverted | arr_delay | carrier_dela | weather_del | nas_delay | security_del | late_aircraft_de |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2017 | 11 | AA | American Air | ATL | Atlanta, GA: | 889 | 91 | 34.24 | 0 | 20.88 | 0.34 | 35.54 | 1 | 0 | 3948 | 1533 | 0 | 628 | 7 | 1780 |
| 2017 | 11 | AA | American Air | BOS | Boston, MA: | 2091 | 281 | 92.04 | 1.81 | 104.7 | 0.26 | 82.18 | 12 | 0 | 13677 | 4851 | 57 | 3220 | 5 | 5544 |
| 2017 | 11 | AA | American Air | BWI | Baltimore, M | 492 | 46 | 23.39 | 0 | 12.78 | 0 | 9.83 | 0 | 1 | 2338 | 1364 | 0 | 374 | 0 | 600 |
| 2017 | 11 | AA | American Air | CLT | Charlotte, N | 7730 | 657 | 253.98 | 7.18 | 164.77 | 2.81 | 228.25 | 15 | 2 | 36644 | 16011 | 581 | 4432 | 574 | 15046 |
| 2017 | 11 | AA | American Air | DCA | Washington, | 1919 | 222 | 66.48 | 1.44 | 79.17 | 0.9 | 74.01 | 5 | 4 | 11709 | 4040 | 65 | 2397 | 261 | 4946 |
| 2017 | 11 | AA | American Air | DEN | Denver, CO: | 814 | 84 | 39.14 | 0.67 | 20.24 | 0.67 | 23.29 | 1 | 1 | 4385 | 2377 | 32 | 563 | 14 | 1399 |
| 2017 | 11 | AA | American Air | DFW | Dallas/Fort V | 11106 | 851 | 302.7 | 10.05 | 222.16 | 8.65 | 307.45 | 5 | 7 | 58922 | 26069 | 2287 | 6101 | 1153 | 23312 |
| 2017 | 11 | AA | American Air | DTW | Detroit, MI: I | 458 | 62 | 23.99 | 1.45 | 29.96 | 0 | 6.59 | 1 | 0 | 2731 | 1143 | 40 | 1043 | 0 | 505 |
| 2017 | 11 | AA | American Air | EWR | Newark, NJ: | 591 | 140 | 31.59 | 0 | 93.46 | 0 | 14.95 | 0 | 0 | 7307 | 1969 | 0 | 4508 | 0 | 830 |
| 2017 | 11 | AA | American Air | FLL | Fort Lauderd | 469 | 63 | 25.01 | 0.75 | 16.56 | 0 | 20.68 | 0 | 1 | 2626 | 1104 | 12 | 444 | 0 | 1066 |
| 2017 | 11 | AA | American Air | HNL | Honolulu, HI | 208 | 37 | 19.36 | 0 | 13.25 | 0 | 4.4 | 0 | 2 | 2595 | 1303 | 0 | 478 | 0 | 814 |
| 2017 | 11 | AA | American Air | IAD | Washington, | 133 | 24 | 10.49 | 0 | 8.71 | 0 | 4.8 | 0 | 0 | 780 | 381 | 0 | 250 | 0 | 149 |
| 2017 | 11 | AA | American Air | IAH | Houston, TX: | 525 | 48 | 22.37 | 0 | 15.7 | 0 | 9.92 | 0 | 0 | 2085 | 1209 | 0 | 379 | 0 | 497 |
| 2017 | 11 | AA | American Air | JFK | New York, N | 1240 | 172 | 61.63 | 2.25 | 61.39 | 0.83 | 45.89 | 3 | 0 | 11718 | 4454 | 591 | 2224 | 25 | 4424 |
| 2017 | 11 | AA | American Air | LAS | Las Vegas, N | 1112 | 131 | 47.72 | 2.27 | 51.62 | 1.67 | 27.73 | 1 | 2 | 5632 | 2191 | 61 | 1466 | 108 | 1806 |
| 2017 | 11 | AA | American Air | LAX | Los Angeles, | 2937 | 316 | 116.63 | 2.12 | 111.51 | 3.12 | 82.62 | 2 | 10 | 18843 | 8536 | 109 | 3261 | 335 | 6602 |
| 2017 | 11 | AA | American Air | LGA | New York, N | 1769 | 250 | 68.24 | 2.76 | 114.41 | 0.95 | 63.64 | 10 | 0 | 10404 | 3028 | 161 | 3347 | 21 | 3847 |
| 2017 | 11 | AA | American Air | MCO | Orlando, FL: | 1329 | 186 | 80.38 | 1.04 | 45.18 | 0.19 | 59.21 | 3 | 4 | 9338 | 4567 | 61 | 1421 | 6 | 3283 |

The next step of data scrubbing was removing the unwanted columns from our data set. Because each row in the data set correspond to three variables, month, carrier and airport name we saw it fit to remove the carrier name variable. This is because for the point of this investigation, we are not interested in the name of the carrier operating the flight. We also removed the minute delays columns because we are only interested in the aggregate number of delays, not how long they took. After removing these columns, the data set looked like this:

| airport | month | arr_flights | arr_del15 | carrier_ct | weather_ct | nas_ct | security_ct | late_aircraft_ |
|---|---|---|---|---|---|---|---|---|
| ATL | 11 | 889 | 91 | 34.24 | 0 | 20.88 | 0.34 | 35.54 |
| BOS | 11 | 2091 | 281 | 92.04 | 1.81 | 104.7 | 0.26 | 82.18 |
| BWI | 11 | 492 | 46 | 23.39 | 0 | 12.78 | 0 | 9.83 |
| CLT | 11 | 7730 | 657 | 253.98 | 7.18 | 164.77 | 2.81 | 228.25 |
| DCA | 11 | 1919 | 222 | 66.48 | 1.44 | 79.17 | 0.9 | 74.01 |
| DEN | 11 | 814 | 84 | 39.14 | 0.67 | 20.24 | 0.67 | 23.29 |
| DFW | 11 | 11106 | 851 | 302.7 | 10.05 | 222.16 | 8.65 | 307.45 |
| DTW | 11 | 458 | 62 | 23.99 | 1.45 | 29.96 | 0 | 6.59 |
| EWR | 11 | 591 | 140 | 31.59 | 0 | 93.46 | 0 | 14.95 |
| FLL | 11 | 469 | 63 | 25.01 | 0.75 | 16.56 | 0 | 20.68 |
| HNL | 11 | 208 | 37 | 19.36 | 0 | 13.25 | 0 | 4.4 |
| IAD | 11 | 133 | 24 | 10.49 | 0 | 8.71 | 0 | 4.8 |
| IAH | 11 | 525 | 48 | 22.37 | 0 | 15.7 | 0 | 9.92 |

Data importing and processing:

I. The Goal:
Our first goal in importing the data was logging in the flight information based on airport code. Consequently, we preallocated matrices for each airport code in order to develop efficient code, considering that extending matrices for a data set this size as greatly inefficient for computer memory.

```
function structdata(airports,acdata)
[r , ~] = size(acdata);
ATLdat=zeros(3,7);
BWIdat=zeros(3,7);
BOSdat=zeros(3,7);
CLTdat=zeros(3,7);
MDWdat=zeros(3,7);
ORDdat=zeros(3,7);
DFWdat=zeros(3,7);
DENdat=zeros(3,7);
DTWdat=zeros(3,7);
FLLdat=zeros(3,7);
HNLdat=zeros(3,7);
IAHdat=zeros(3,7);
LASdat=zeros(3,7);
LAXdat=zeros(3,7);
MIAdat=zeros(3,7);
MSPdat=zeros(3,7);
JFKdat=zeros(3,7);
LGAdat=zeros(3,7);
EWRdat=zeros(3,7);
MCOdat=zeros(3,7);
PHLdat=zeros(3,7);
PHXdat=zeros(3,7);
PDXdat=zeros(3,7);
```

We decided to use matrices instead of data structures because although initially importing data would be easier using data structures, as we can save the airport codes in a cell array and for looping to allocate data structures for each airport code, using a numeric matrix makes it easier to access the data for plotting and performing machine learning. Each matrix had a dimension of 3 by 7, as 3 rows were needed for each month investigated and 7 rows were needed to record the number of arrival flights, number of flights delayed and the other 5 columns for the coefficients of the different causes of flight delays including carrier problems, weather, National Airspace System delay (NAS), security delays and finally late delays.

## II. Data Importing:

After scrubbing the raw data file, we needed to import the data. In order to ease our data importing, we imported the first column of the data as a string array, so that it can be used to return a logical output for our switch statements that log the data by airport code. The imported airport string column looks like this:


| | 1 |
|---|---|
| 1 | ATL |
| 2 | BOS |
| 3 | BWI |
| 4 | CLT |
| 5 | DCA |
| 6 | DEN |
| 7 | DFW |
| 8 | DTW |
| 9 | EWR |
| 10 | FLL |
| 11 | HNL |
| 12 | IAD |
| 13 | IAH |
| 14 | JFK |
| 15 | LAS |
| 16 | LAX |
| 17 | LGA |
| 18 | MCO |

The remainder of the data was imported as a numeric matrix in order to be able to log the numbers into the respective matrix for each airport code. The imported numeric matrix looked like this.

892x8 double

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| 1 | 11 | 889 | 91 | 34.2400 | 0 | 20.8800 | 0.3400 | 35.5400 |
| 2 | 11 | 2091 | 281 | 92.0400 | 1.8100 | 104.7000 | 0.2600 | 82.1800 |
| 3 | 11 | 492 | 46 | 23.3900 | 0 | 12.7800 | 0 | 9.8300 |
| 4 | 11 | 7730 | 657 | 253.9800 | 7.1800 | 164.7700 | 2.8100 | 228.2500 |
| 5 | 11 | 1919 | 222 | 66.4800 | 1.4400 | 79.1700 | 0.9000 | 74.0100 |
| 6 | 11 | 814 | 84 | 39.1400 | 0.6700 | 20.2400 | 0.6700 | 23.2900 |
| 7 | 11 | 11106 | 851 | 302.7000 | 10.0500 | 222.1600 | 8.6500 | 307.4500 |
| 8 | 11 | 458 | 62 | 23.9900 | 1.4500 | 29.9600 | 0 | 6.5900 |
| 9 | 11 | 591 | 140 | 31.5900 | 0 | 93.4600 | 0 | 14.9500 |
| 10 | 11 | 469 | 63 | 25.0100 | 0.7500 | 16.5600 | 0 | 20.6800 |
| 11 | 11 | 208 | 37 | 19.3600 | 0 | 13.2500 | 0 | 4.4000 |
| 12 | 11 | 133 | 24 | 10.4900 | 0 | 8.7100 | 0 | 4.8000 |
| 13 | 11 | 525 | 48 | 22.3700 | 0 | 15.7000 | 0 | 9.9200 |
| 14 | 11 | 1240 | 172 | 61.6300 | 2.2500 | 61.3900 | 0.8300 | 45.8900 |
| 15 | 11 | 1112 | 131 | 47.7200 | 2.2700 | 51.6200 | 1.6700 | 27.7300 |
| 16 | 11 | 2937 | 316 | 116.6300 | 2.1200 | 111.5100 | 3.1200 | 82.6200 |
| 17 | 11 | 1769 | 250 | 68.2400 | 2.7600 | 114.4100 | 0.9500 | 63.6400 |
| 18 | 11 | 1329 | 186 | 80.3800 | 1.0400 | 45.1800 | 0.1900 | 59.2100 |
| 19 | 11 | 4024 | 413 | 159.1200 | 3.1200 | 131.6600 | 4.8800 | 114.2100 |

III. Data processing

Now that the data is imported, and the length of the airport code column is the same as the length of the matrix columns, we used a for loop to shift through each row of the data. A total of 30 switch statements were used to validate which airport code this data corresponds to. This is shown below:

```
for i=1:r
    switch(airports(i))
        case 'ATL'
            switch(acdata(i,1))
                case 11
                    ATLdat(1,1:7)= [acdata(i,2),acdata(i,3),acdata(i,4),acdata(i,5),acdata(i,6),acdata(i,7),acdata(i,8)]+ATLdat(1,1:7);

                case 12
                     ATLdat(2,1:7)=[acdata(i,2),acdata(i,3),acdata(i,4),acdata(i,5),acdata(i,6),acdata(i,7),acdata(i,8)]+ATLdat(2,1:7);

                case 1
                     ATLdat(3,1:7)=[acdata(i,2),acdata(i,3),acdata(i,4),acdata(i,5),acdata(i,6),acdata(i,7),acdata(i,8)]+ATLdat(3,1:7);
             end

        case 'BWI'
            switch(acdata(i,1))
                case 11
                   BWIdat(1,1:7)=[acdata(i,2),acdata(i,3),acdata(i,4),acdata(i,5),acdata(i,6),acdata(i,7),acdata(i,8)]+BWIdat(1,1:7);
                case 12
                    BWIdat(2,1:7)=[acdata(i,2),acdata(i,3),acdata(i,4),acdata(i,5),acdata(i,6),acdata(i,7),acdata(i,8)]+BWIdat(2,1:7);
                case 1
                    BWIdat(3,1:7)=[acdata(i,2),acdata(i,3),acdata(i,4),acdata(i,5),acdata(i,6),acdata(i,7),acdata(i,8)]+BWIdat(3,1:7);
```

Once the airport code was identified, the data must be logged to the corresponding matrix. This was done using another switch statement going through the flight delay data (acdata) with 3 cases. The switch statement allowed us to log the data for each month in a separate row for the airport matrix by checking the month variable found in the original data set. Since the raw data is arranged in rows by delays and by carrier too, we needed to aggregate the delays for each month as there are multiple entries for each airport for each month depending on the carrier name. Thus, we kept on summing the rows for each respective month.
Finally, to check the validity of the data processing, we unsuppressed the output for one airports matrix, as shown below:


```
DCAdat =

   1.0e+04 *

    0.5934    0.0664    0.0223    0.0005    0.0199    0.0002    0.0235
    0.5870    0.1011    0.0296    0.0030    0.0271    0.0003    0.0411
    1.0752    0.2125    0.0550    0.0079    0.0679    0.0002    0.0816
```

We then manually summed up the values for each corresponding element by sifting through the raw data set to ensure the validity of the code.

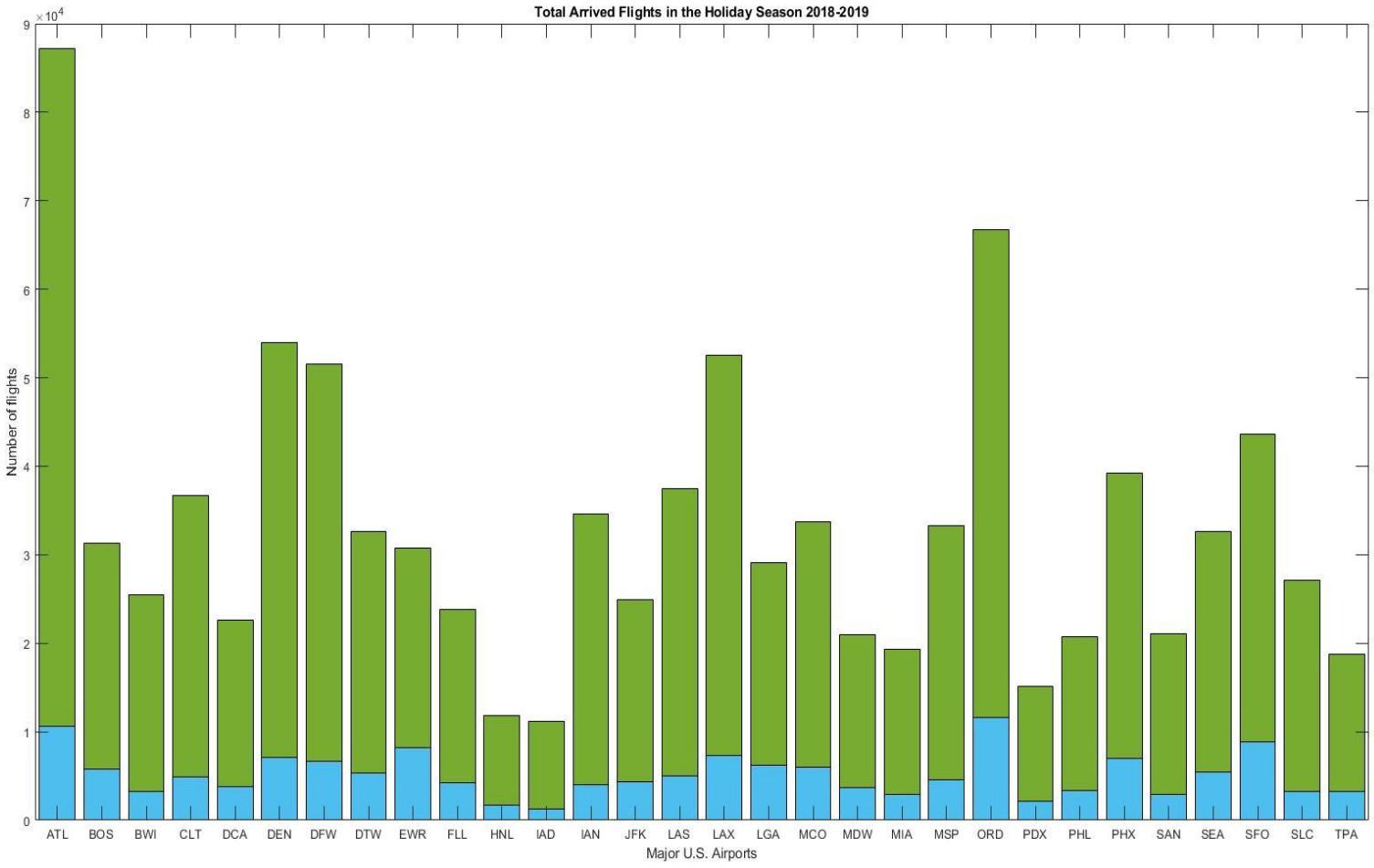
## Interpreting the Data:

Once the data is scrubbed and formatted, each of the airport matrices was compounded into a three-dimensional matrix for easier use with for loops. Next, two new vectors were created, “arr” and “arr_del”, to contain the sums of columns 1 and columns 2 respectively. The individual values of “arr” signify the total amount of arrived flights for each airport over the holiday season, and the “arr_del” values correspond with the total number of delays for each airport over the holiday season. For both vectors, the numeric value of the index corresponds with the airport name in the categorical array “X”.

```
383
384 -   X = categorical({'ATL', 'BWI', 'BOS', 'CLT', 'MDW', 'ORD', 'DFW', 'DEN','DTW', 'FLL', 'HNL'...
385         'IAN', 'LAS', 'LAX', 'MIA', 'MSP', 'JFK', 'LGA', 'EWR', 'MCO', 'PHL', 'PHX', 'PDX'...
386         'SLC', 'SAN', 'SFO', 'SEA' 'TPA', 'DCA', 'IAD'});
387
388 -   ThreeD_arr = ATLdat;
389 -   ThreeD_arr(:,:,2) = BWIdat;
390 -   ThreeD_arr(:,:,3) = BOSdat;
391 -   ThreeD_arr(:,:,4) = CLTdat;
392 -   ThreeD_arr(:,:,5) = MDWdat;
393 -   ThreeD_arr(:,:,6) = ORDdat;
394 -   ThreeD_arr(:,:,7) = DFWdat;
395 -   ThreeD_arr(:,:,8) = DENdat;
396 -   ThreeD_arr(:,:,9) = DTWdat;
397 -   ThreeD_arr(:,:,10) = FLLdat;
398 -   ThreeD_arr(:,:,11) = HNLdat;
399 -   ThreeD_arr(:,:,12) = IAHdat;
400 -   ThreeD_arr(:,:,13) = LASdat;
401 -   ThreeD_arr(:,:,14) = LAXdat;
402 -   ThreeD_arr(:,:,15) = MIAdat;
403 -   ThreeD_arr(:,:,16) = MSPdat;
404 -   ThreeD_arr(:,:,17) = JFKdat;
405 -   ThreeD_arr(:,:,18) = LGAdat;
406 -   ThreeD_arr(:,:,19) = EWRdat;
407 -   ThreeD_arr(:,:,20) = MCOdat;
408 -   ThreeD_arr(:,:,21) = PHLdat;
409 -   ThreeD_arr(:,:,22) = PHXdat;
410 -   ThreeD_arr(:,:,23) = PDXdat;
411 -   ThreeD_arr(:,:,24) = SLCdat;
412 -   ThreeD_arr(:,:,25) = SANdat;
413 -   ThreeD_arr(:,:,26) = SFOdat;
414 -   ThreeD_arr(:,:,27) = SEAdat;
415 -   ThreeD_arr(:,:,28) = TPAdat;
416 -   ThreeD_arr(:,:,29) = DCAdat;
417 -   ThreeD_arr(:,:,30) = IADdat;
418
419 -   arr = zeros(30);
420 -   arr_del = zeros(30);
421 -   for i = 1:length(X)
422 -       sumdat1 = sum(ThreeD_arr(:,:,i));
423 -       arr(i) = sumdat1(1);
424 -       sumdat2 = sum(ThreeD_arr(:,:,i));
425 -       arr_del(i) = sumdat2(2);
426 -   end
```

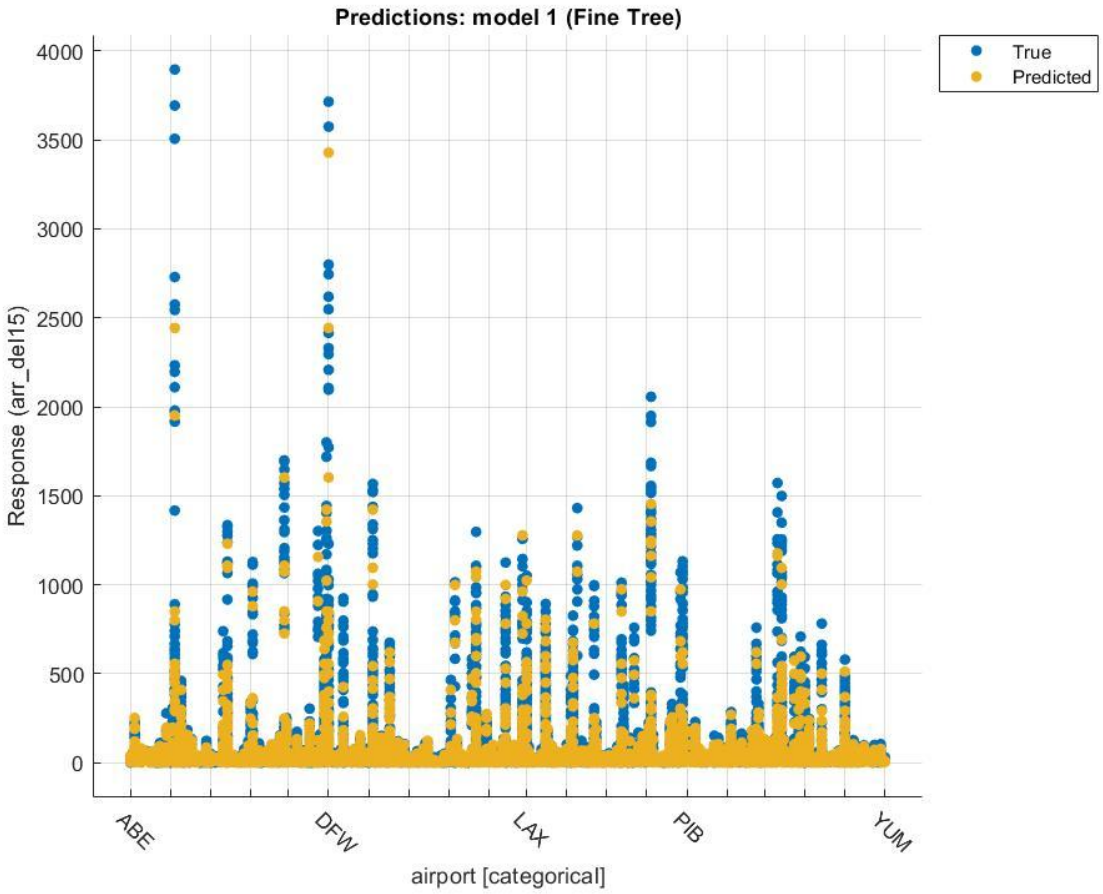
Then, the code was written to portray the values of “arr” and “arr_del” as a stacked bar graph to display the percentage of delayed aircraft out of the total that arrived. The X-axis consists of the 30 major airports and the Y-axis is the number of aircraft.

```
427
428 -    bar(X, arr)
429 -    hold on
430 -    bar(X, arr_del)
431 -    title("Total Arrived Flights in the Holiday Season 2018-2019");
432 -    ylabel("Number of flights");
433 -    xlabel("Major U.S. Airports");
434
435
```

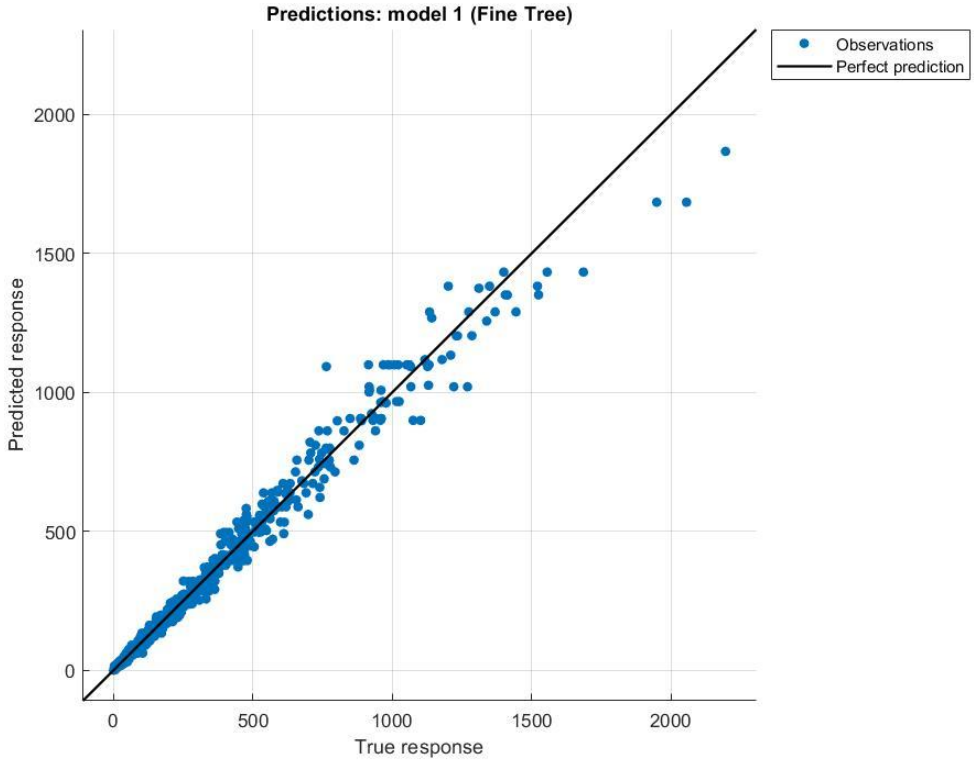


## Machine Learning:

We began by just playing around with the ML Toolbox in Matlab. We used the Fall Exam Data to just try out the different models and understand how to work with the program. Then we spent some time reading through Matlab's own guide to data scrubbing, holdout verification, the different types of plots, and any other relevant help pages we could find. After getting somewhat familiar with ML we began to play around in it with our own data. The first model we made was with the same data above, and here is the trained plot.

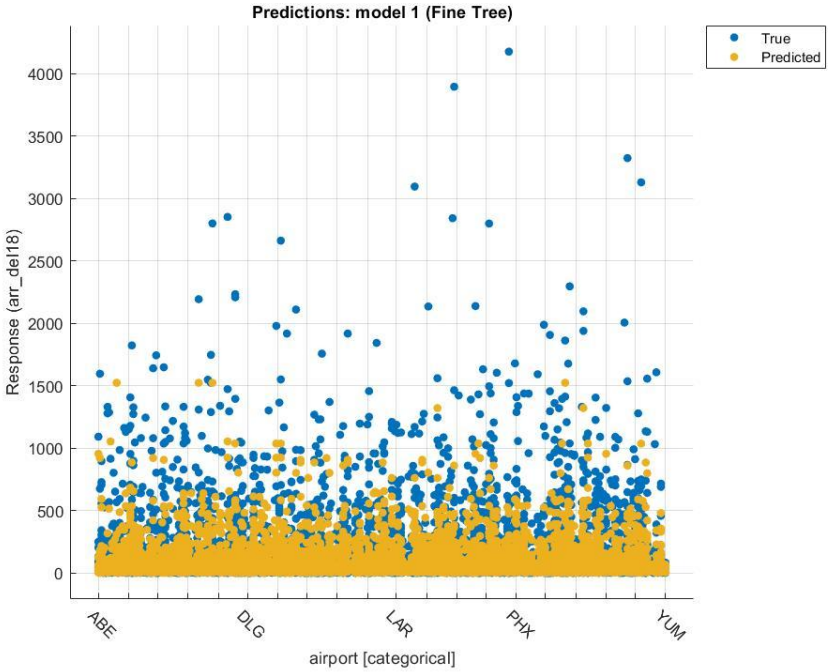



The blue dots represent the actual data and the yellow are the predictions. We used the number of delayed flights as the response variable. This is the variable that the machine uses to predict values. The rest of the data, i.e. airport, weather delays, security delays, was used as the predictors. This model was a fine tree regression with a 25% holdout verification. This model ended up being a very accurate fit for the data. With the predicted responses being very close to the real responses.

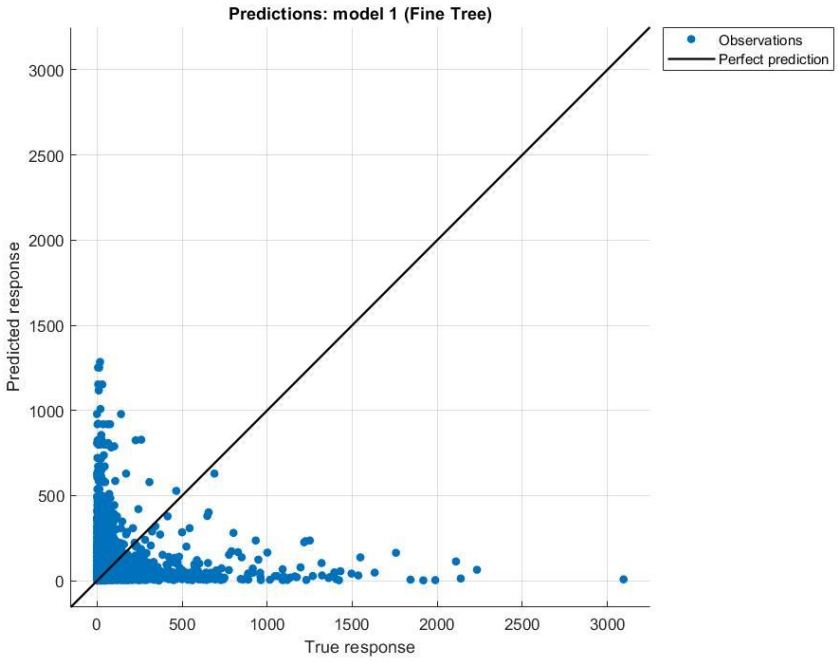



While this initially seemed really good, we had a suspicion something was off. We realized that about half of the data in the prediction variables were used to calculate the data in the response variable. We believe that this is why the model is so good. Because the data it is training on is literally what makes up the data it is trying to predict. So we decided to grab more data from our source to try and play around with a few more models.

We downloaded data from the past five years of airports across the U.S. and scrubbed as we did for the first step. This time, however, we took the data from the first four years and used that as the predictor variable data. For the response variable, we used the number of delayed flight data for 2018 ( arr_del18) and made this model with the same setup as the first.




This model is a lot less accurate, despite having nearly four times the data as the first. The most glaring difference is in the predicted response vs true response graph.




As you can see, the model has very little accuracy. This could have been due to a few different things. Perhaps when we scrubbed this data we should've taken a different approach considering how the set differs from our original, or perhaps we were misguided in what we were comparing.

## Conclusion:

From the data that we graphed, we can see that there is some correlation between total arrived flights and number of delays. This is a logical conclusion because as an airport is used by more aircraft the likelihood of problems arising and affecting more flights increases. As of the winter season 2018-2019, ORD, which is located in Orlando, Florida, experienced the most amount of delays out of any major airport while also not having the greatest number of total arrived aircraft. Our machine learning program ultimately gave us one model that worked well, but may only work for 2018, which would be useless for predicting delays for future years. Just because ORD had the most this year does not mean it will have the most next year. Therefore our prediction models are not suitable for real-world applications, but if you could create a working model using machine learning it could be extremely useful for helping both the government and travels plan around these delays.

# Works Cited

Edwards, Chris. "Privatizing U.S. Airports." Downsizing the Federal Government, 28 Nov. 2016, www.downsizinggovernment.org/privatizing-us-airports. Accessed 11 Dec. 2019.