Building Models to Predict Delays in Flights Departing from New York City in December

Andrea Shealey

2022-12-09

Contents

Contents

List of Figures

List of Tables

Abstract

Every year, thousands of flights depart from New York City to destinations all over the United States during the month of December. Airports can be especially busy during this holiday month, and many of those flights end up delayed. Departure delays can cause major disruptions to airport schedules, and correctly estimating this delay can help transportation analysts to plan more efficiently. In the following report I discuss the process of building a model to estimate by how many minutes a flight's departure will be delayed. I will be using a subset of data from The United States Department of Transportation's Bureau of Transportation Statistics that reports 19 features on over 33,000 domestic flights that departed from New York City airports (EWR, JFK, and LGA) during 2013. I evaluated each feature and built several models to predict how many minutes a flight's departure will be delayed using k-nearest-neighbors, elastic net, and a regression forest. The model I consider optimal is able to produce an estimate that is, on average, 6.4 minutes above or below the actual departure delay. The following report explains the processes used and limitations of the study.

Section 1: Data and Motivation

I will be working with data collected and published by the U.S. DOT's Bureau of Transportation Statistics regarding the performance of flights operated by large air carriers. The dataset contains information on over 300,000 domestic flights that departed out of New York City that occurred between January 1st and December 31st of 2013. We have a total of 19 features that describe various characteristics of each flight. We know the date of each flight, which airline it was operated by, the flight number, code of the origin and destination airports, the distance between the airports in miles, the estimated departure and arrival time of the flight on a 24-hour scale, the actual departure and arrival time of the flight on a 24-hour scale, and many more. I have provided a link which describes each feature in the Works Cited section. We also have data on how many minutes the flight was delayed by, which will serve as the response variable. I want to build a model that can accurately predict by how many minutes a flight will be delayed, specifically domestic flights departing from NYC airports in the month of December. I will use k-nearest-neighbors, elastic net, and a regression forest to explore this regression prediction task.

Section 2: Data Cleaning and EDA

There are several changes that need to be made to the data set before I can begin creating prediction models. First I will remove all rows that provide data on flights between the months of January and November, as the scope of my investigation is flights in December. The nature of my task is that I want to predict how many minutes the flight will be delayed, before the flight actually happens. This means I will need to remove several columns that give information on the flight once it has occurred, for example every 'actual' variable, such as actual departure time, arrival time, and length of flight. After removing all data collected after the flight has begun, I am left with 15 columns including my response variable. Out of these 15 columns, I am going to remove a few more. I chose to remove the flight number column because it is unique to each variable and is not useful in creating patterns to be used for regression. Although sometimes flight numbers are repeated, they do not always give identifying information about the flight. I will also be removing the tail number column because it is a categorical variable with over 4,000 unique levels. Although it is possible that there is a pattern concerning the departure delay of a flight and the specific airplane, I do not think it is a feature worth exploring given its complexity. I have removed the year and month variables because all the data I'm working with is from December of 2013. Finally, I will be removing the hour, minute, and time hour columns because this data is found in the scheduled departure time column, and there is no value in repeating it.

The remaining data set has 8 features which consist of the number of the day that the flight was scheduled for, the operating airline, the scheduled departure and arrival times, the origin and destination airports, the distance between the two airports, and the total number of minutes the departure time was delayed. Each of

these features are available at an appropriate time to predict cancellation, and each make sense to use in our models. I will now begin to clean my data and check conditions.

I have conducted a simple cleaning of my data in which I removed any row that had information missing in any column. I chose to do such a full clean because my data set is already very large, and there are still plenty of data points after the 1,025 rows with missing information were removed. The data points that were removed were largely missing the total delay in the departure time, which is vital information as this serves as my response variable. Additionally, I only have 8 columns of information, and I feel strongly that my remaining data points should be complete. My cleaned data set has 8 columns, including the response variable, and 27,110 rows. Now that I am satisfied with my data, I will be exploring each of my features.

Figure 2.1

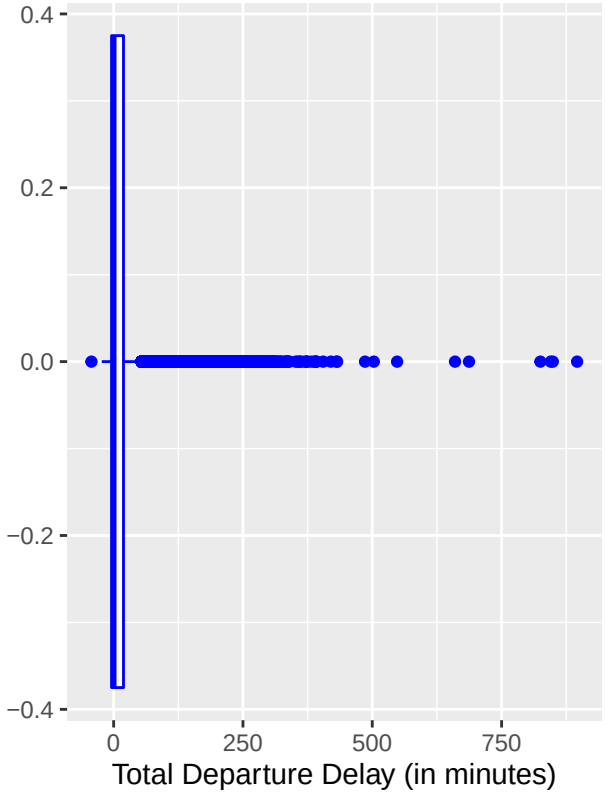
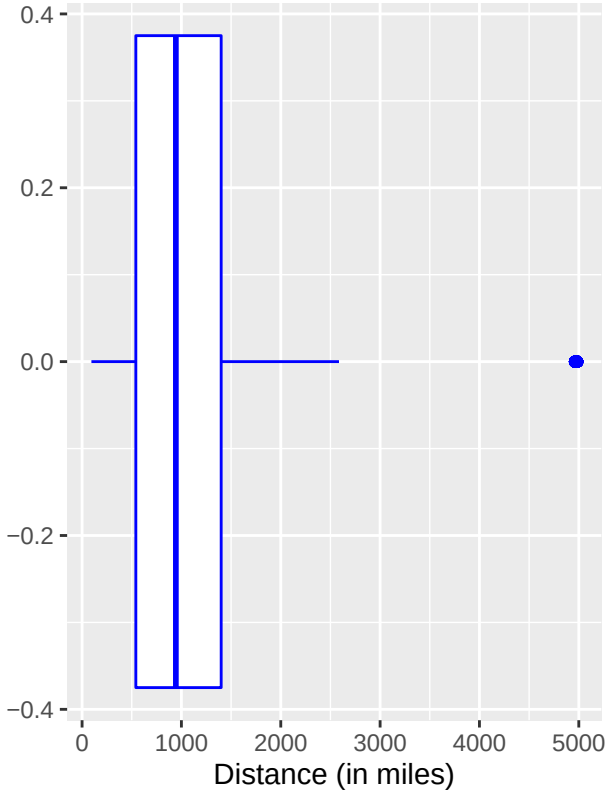


Figure 2.2



First, I am going to check my data for outliers. I am not worried about outliers in my day or scheduled arrival or departure time features. This is because these features can only hold values in a specified interval, thus there should not be significant outliers. Figures 2.1 and 2.2 above display a box plot of the values for total departure delay and flight distance. We can see in Figure 2.1 that there appear to be several outliers in which flights were extremely delayed. 75% of the data displays delay times less than 20 minutes, and the other 25% of the data ranges between 19 and 896 minutes. There also appear to be outliers in which a flight departs more than 20 minutes early. This amount of variation in response variable will make it very difficult for models to make accurate predictions. Because of this, I have chosen to only use data on flights that departed between -20 and 20 minutes late. There are also obvious outliers in the flight distance, as shown in Figure 2.2. The middle 50% of the data falls between a distance of 533 and 1400 miles. There are data points with distances as low as 94 miles and as much as almost 5000 miles, and these could greatly impact the regression techniques I will be using. I will only use data points with flight distances between 500 and 1500 miles to remove the outliers.

Figure 2.3

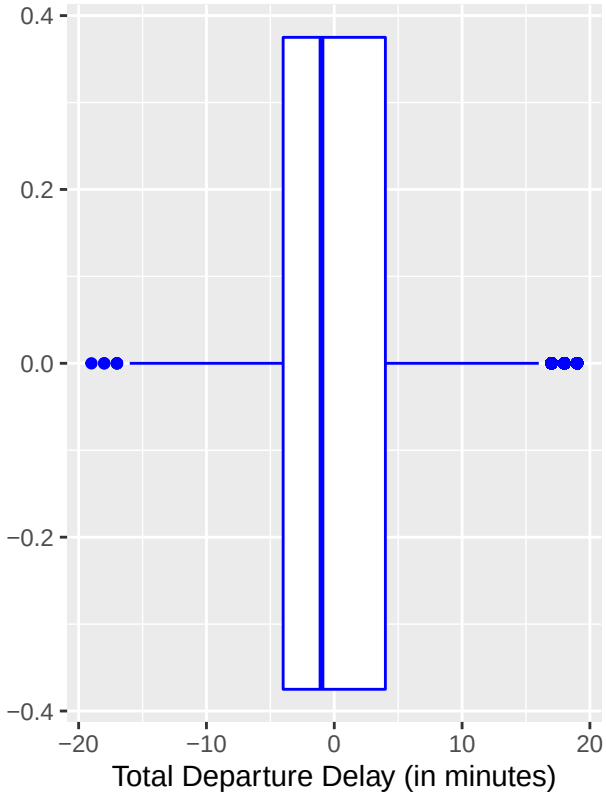
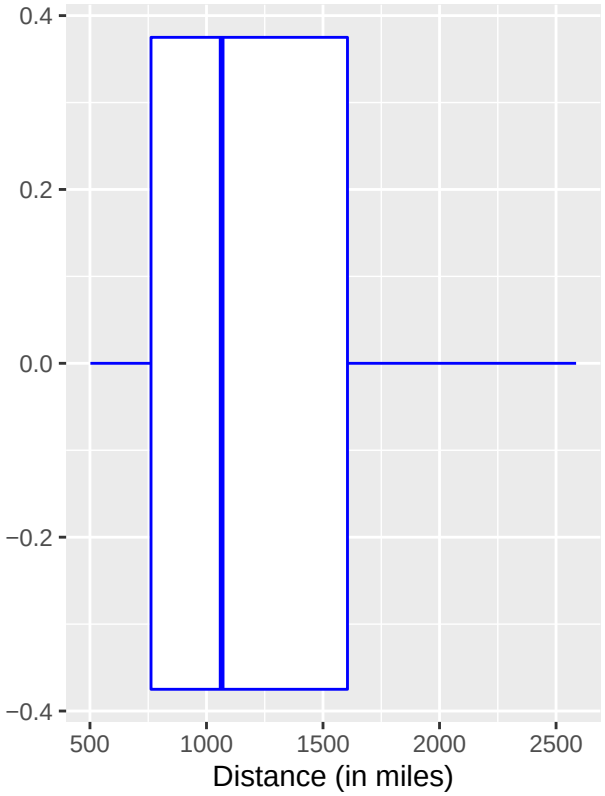


Figure 2.4



Figures 2.3 and 2.4 above show the distribution of total departure delays and flight distance after removing outliers. This data set has the same 8 columns with 16,268 rows. When I use techniques that can be negatively affected by outliers, I will use this smaller data set. When I use this data I will be greatly narrowing the scope of my investigation, as the model will only be able to predict a flight delay between -20 and 20 minutes. However, I can expect a model with high variance features might provide an estimate with high variance. A model with a smaller scope, but higher predictive accuracy could be more useful in practice.

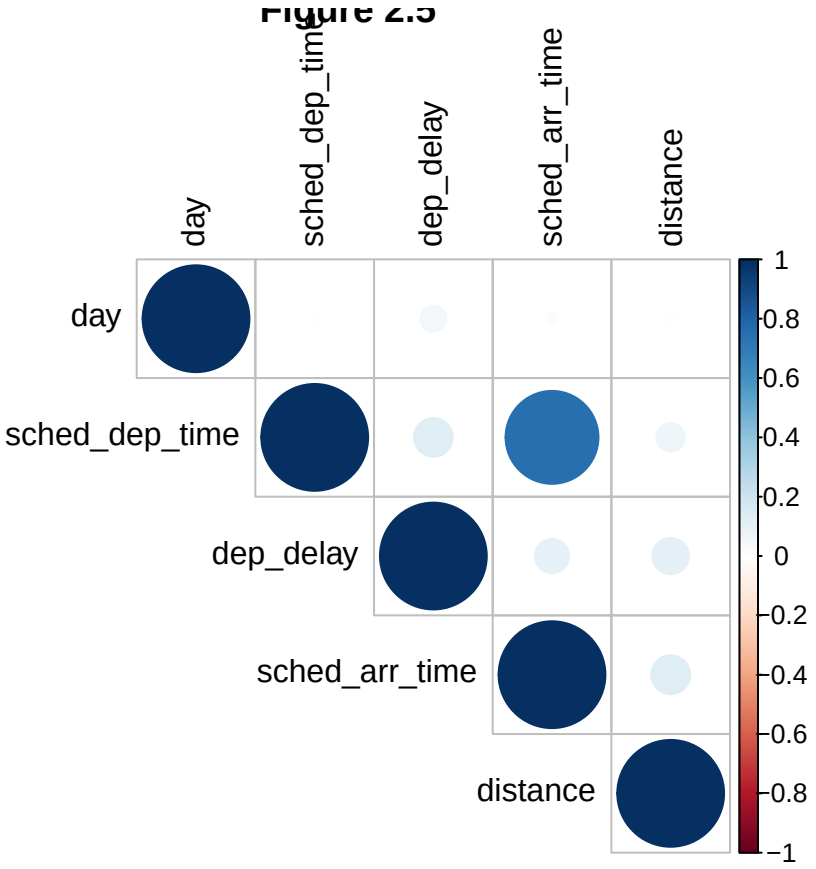


Figure 2.5 is a correlation plot of all the numeric features. We can see in the plot that the correlation between the scheduled departure and arrival times is rather high. This relationship is logical, however it is important to take this high correlation into considerations when building a linear regression model. There are linear regression models that are unable to effectively deal with high correlations.

Figure 2.6

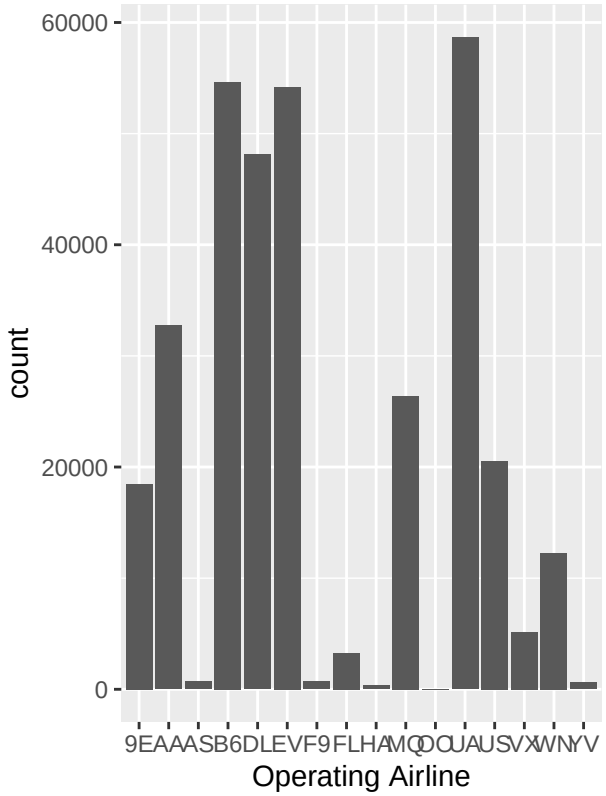
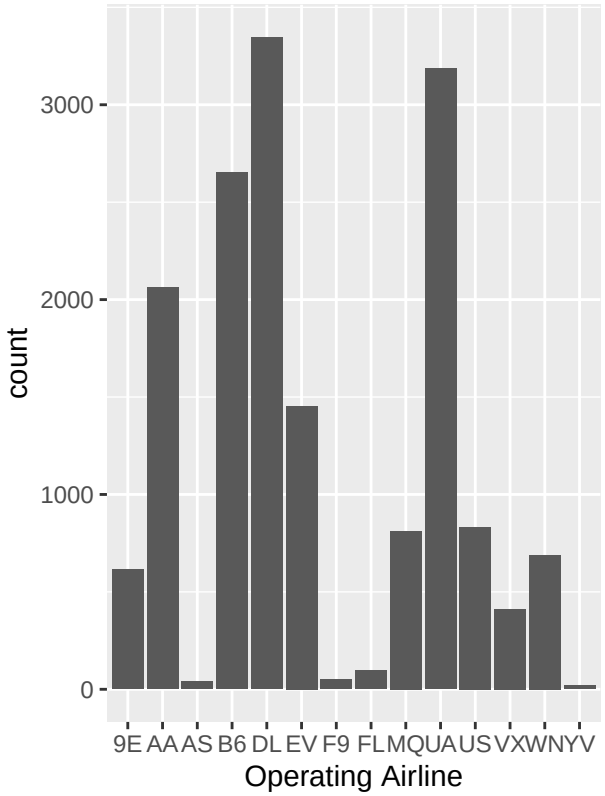


Figure 2.7



Figures 2.6 and 2.7 above demonstrate the distribution of flights between different operating airlines. I included both a bar graph for the original data set (2.6) and the data set with the outliers removed (2.7), to check if the data I removed has a significant impact on the distribution of the data. We can see that the distributions are very similar, however the dataset I will be using excludes flights done through SkyWest Airlines (OO) and Hawaiian Airlines (HA). I took a closer look at flights done through SkyWest Airlines and found that there was only data on flights between January and November of 2013. I am not concerned about losing data on an entire airline because I only want to analyze flights in December, which they do not have data on. I also took a closer look at flights done through Hawaiian Airlines. Every flight offered by this airline arrives in the Honolulu airport, totaling a distance of over 4,000 miles. Earlier I removed all these flights because I identified them as outliers. The distribution of data between the different airline carriers is otherwise very similar to the original data set.

Figure 2.8

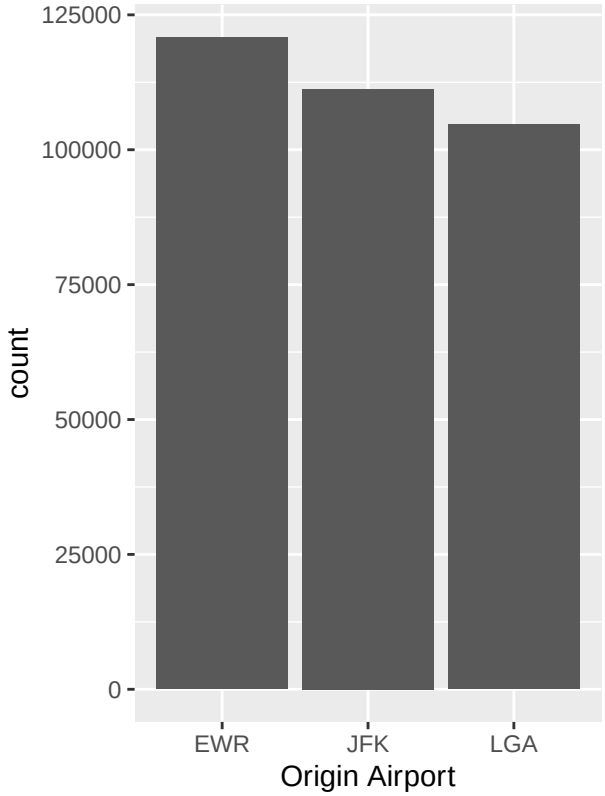
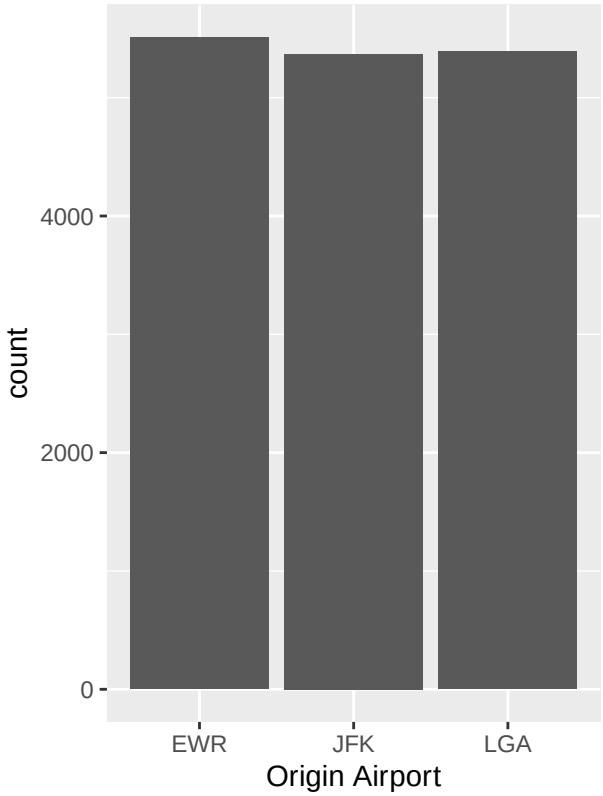
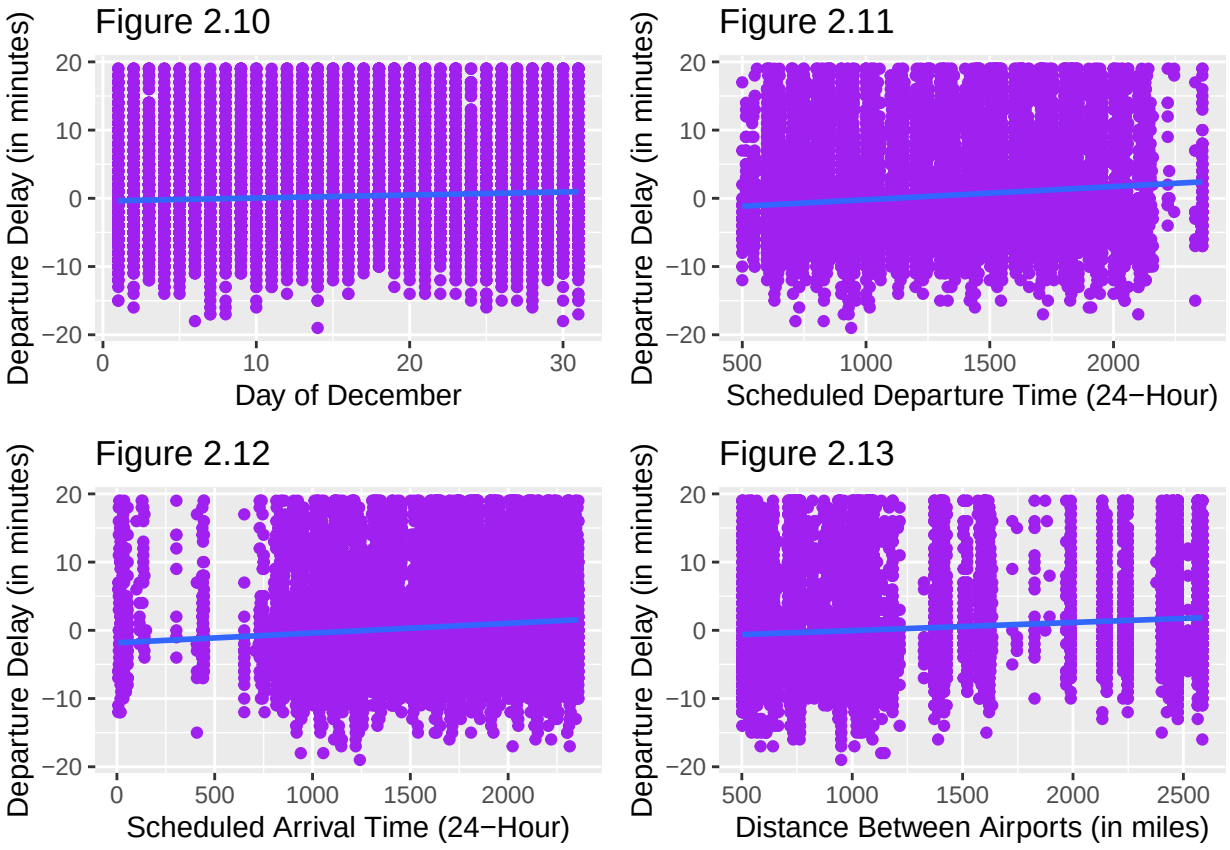


Figure 2.9



I also wanted to check the distribution of flights between different origin airports in the original data set verses the one cleaned of outliers. Figure 2.8 displays the distribution from the original data set, and Figure 2.9 uses the data from the data set without outliers set. It is clear that we lost more data from flights departing from EWR than those departing from JFK and LGA. However, there is still nearly equal data from flights departing from each airport so I am satisfied with this result. I will not be checking the change in distribution between the original and cleaned data set for the destination airport feature. I am choosing to forgo this step because the arrival airport feature has several levels and it is difficult to visualize this on a graph. Now I am going to check the relationships between the numeric variables and the response variable. I plan to use a linear regression and I want to ensure that none of my variables demonstrate relationships that are strongly non-linear.



Figures 2.9 through 2.12 above show the relationships between my numeric features and my response variables. While none of the relationships are strongly linear, they also do not show a relationship that is strongly non-linear. I am satisfied to use these variables in a linear regression model. Now that I have explored each of my features and thoroughly cleaned my data of missing values and outliers, I am ready to start building prediction models.

Section 3: KNN

3.1: Introduction

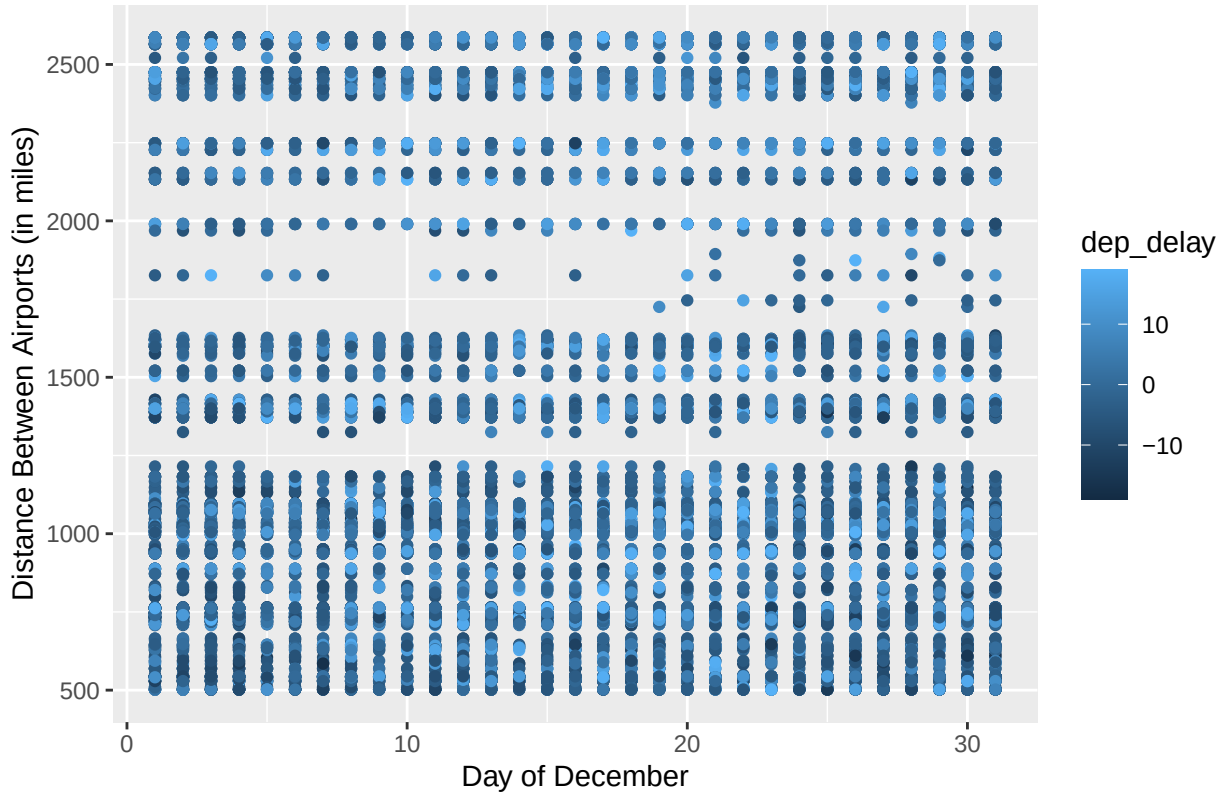
The first method I will be using to predict the delay in a flight's departure is a clustering technique called K-Nearest-Neighbors, or KNN. I have chosen to use KNN because it is relatively simple to understand and it can be used in regression prediction tasks. KNN is a clustering technique rather than a statistical model, which I think could be effective with my data by clustering different days, times, or distance that seem to inherit significant departure delays. I also want to use KNN because the clustering nature could be helpful in dealing with my large data set with many outliers. For this method, I will build one model with the cleaned data set before I removed distance and delay time outliers, and one with the dataset after I removed the outliers.

3.2: Method

The first step in conducting classification with KNN is to create a graph with features on each axis and plot the points in the data set. Next, choose a row you wish to predict and plot it on the same graph. You must find the 'k' number of data points with the shortest Euclidean distance from the test point. The predicted

measure for the row will be the average of these neighbor points. For additional clarity, I have provided an example below.

Figure 3.1



To demonstrate KNN I plotted two variables on the graph above, the day of December on the x-axis and the distance between airports on the y-axis. Notice in Figure 3.1 how each point on the graph is color coded according to the number of minutes the departure was delayed. If I wanted to predict the minutes a flight would be delayed, I would plot it on the graph according to the day it takes place and the distance between the two airports. Supposing a k=5, I would then find the 5 closest points according to the Euclidean distance and average their departure delay values. That number would be the prediction for my chosen flight.

To train my data, I will be using the day, scheduled departure and arrival times, and distance features. I will not be using the carrier, origin, or destination features because they are categorical variables. KNN is a classification system based on distance, and thus does not work well with the distance ultimatum present in categorical variables. I will be building two KNN models; one using my final cleaned data and the other using the data set before removing the outliers. I am able to do this because the clustering nature of KNN may allow it to deal with outliers more effectively than statistical techniques. I want to try and build a model with a larger scope, however I also want to build a model that I can compare to other models that are unable to account for major outliers. For both models I will test k values between 1 and 25 and check the predictive accuracy of each model using the root mean of squared errors (RMSE). The next section will discuss the results I obtained by building a KNN model.

3.3: Results

The table below displays the RMSE values obtained from models built with k equal to values 1 through 25. I chose to use the RMSE to compare my models because it uses the same scale as our response variable. In our case, a value of 50 indicates that, on average, our model provides a prediction that is 50 minutes away from

its actual value. Likewise, our optimal model will have the lowest RMSE value.

Table 1: Table 3.1

K	RMSE
1	57.54609
2	56.95883
3	56.82617
4	56.82195
5	55.38717
6	54.01007
7	53.51255
8	51.66205
9	50.49525
10	50.25436
11	48.89882
12	48.34145
13	47.62127
14	47.33581
15	46.71191
16	46.40755
17	46.11527
18	46.04819
19	46.03039
20	45.75699
21	46.04058
22	45.66971
23	45.73930
24	45.66242
25	45.79038

Table 2: Table 3.2

K	RMSE
1	8.571952
2	8.712960
3	8.680562
4	8.611835
5	8.438925
6	8.230815
7	8.105620
8	8.037367
9	7.882208
10	7.876720
11	7.795617
12	7.754889
13	7.721760
14	7.726754
15	7.701430
16	7.654342
17	7.640939
18	7.660583

K	RMSE
19	7.643899
20	7.618473
21	7.640524
22	7.616742
23	7.631351
24	7.614308
25	7.564915

For KNN with the outliers data set, we were able to obtain an optimal RMSE of 45.7 when k=24. At this model, our prediction of a flight's delay time is on average 45.7 minutes away from its actual delay time. 45 minutes is a large amount of error in practice, however it is important to note that this model is built with delays between -43 and 896 minutes. This means that 45 minutes is less than 5% of the range of the data. I anticipated a high RMSE value while using the outliers data, however I think that it is important to explore. This model can be used to predict a large variance in flight departure delays, which makes it applicable to a large amount of data.

For KNN with the dataset without outliers, we were able to obtain an optimal RMSE of 7.6 when k=10. At this model, our prediction of a flight's delay time is on average 7.6 minutes away from its actual delay time. In practice, having an error of 7.6 minutes in either direction is much easier to work with than an error of 45.7 minutes in either direction. An error of 7.6 minutes seems much greater than an error of 45.7 minutes, however this was collected using data from flights that experienced departure delays between -20 and 20 minutes. 7.6 minutes is almost 20% of the range of the data. Additionally, this model is applicable to a much smaller population of flights. In the next section, I will consider a linear regression model which allows me to use all of my features, rather than only the numeric ones.

Section 4: Elastic Net

4.1: Introduction

I am going to use elastic net because I want to use a shrinkage estimation technique for linear regression that balances both coefficient shrinkage and variable selection. In Figure 2.1 we saw that at least two variables in our data displayed high correlations. We must use a shrinkage estimation technique which allows us to build linear regression models with highly correlated variables. The data also has three categorical variables with several levels. While some of these levels might be useful, I want to use a selection technique to remove unhelpful variables. I will only build an elastic net model with the data that has been cleaned of outliers. The regression nature of this technique means that outliers can significantly impact the results of the model, so it is not worth exploring the larger set for a model with a wider scope. In the next section I will explain how Elastic Net is able to perform both shrinkage and selection.

4.2: Method

In elastic net, our goal is to minimize the residual sum of squares (RSS) plus a penalty term. Elastic net differentiates from other shrinkage estimation techniques because of its penalty term, which is,

$$\lambda\Sigma((1-\alpha)\hat{\beta}_j^2 + \alpha|\hat{\beta}_j|$$

We must choose α, λ, and β values to minimize this term.

The first step in running ridge regression is to build the design matrix. In this case, our design matrix will include all features we are using in our model preceded by a single column of ones. The column of ones is

important because it will ensure that our intercept is included after matrix multiplication. The design matrix for this ridge regression model has 87 total columns. There is one column for each numeric feature, and one column per level of each categorical feature. For example, the origin feature has 3 levels, so there will be 3 columns for origin airport. The design matrix for our ridge regression model has 16,268 rows and 87 columns

Once the design matrix is complete, the next step is to choose the tuning parameters. In elastic net, our tuning parameters are λ and α. We must choose a range of both these values to test out. I will test all λ values between 0 and 100 at a 0.5 interval, and α values between 0 and 1 at a 0.01 interval. For each value of α, we test each value of λ using 10-fold cross validation to find the β values which minimize the RSS plus the penalty term. We will store the optimal λ and β values at each α value, and choose the α value with the best test RMSE. Now that I have explained the process of elastic net I will test each λ value at each α value, and find the coefficients that minimize the RMSE.

Table 3: Table 4.1: RMSE of Each Elastic Net Model

Alpha	Lambda	RMSE
0.00	0.5	6.482505
0.01	0.5	6.480933
0.02	0.5	6.479984
0.03	0.5	6.476099
0.04	0.5	6.480775
0.05	0.5	6.478942
0.06	0.5	6.484633
0.07	0.0	6.483203
0.08	0.0	6.481664
0.09	0.5	6.485372
0.10	0.0	6.480891
0.11	0.0	6.483136
0.12	0.0	6.480004
0.13	0.0	6.481042
0.14	0.0	6.484701
0.15	0.0	6.481265
0.16	0.0	6.488312
0.17	0.0	6.480510
0.18	0.0	6.482616
0.19	0.0	6.484602
0.20	0.0	6.487133
0.21	0.0	6.482517
0.22	0.0	6.482220
0.23	0.0	6.483193
0.24	0.0	6.484454
0.25	0.0	6.479410
0.26	0.0	6.481588
0.27	0.0	6.481433
0.28	0.0	6.484616
0.29	0.0	6.480873
0.30	0.0	6.483008
0.31	0.0	6.481644
0.32	0.0	6.480003
0.33	0.0	6.480474
0.34	0.0	6.482110
0.35	0.0	6.480192
0.36	0.0	6.484791
0.37	0.0	6.482955
0.38	0.0	6.484031

Alpha	Lambda	RMSE
0.39	0.0	6.479947
0.40	0.0	6.482954
0.41	0.0	6.479145
0.42	0.0	6.480911
0.43	0.0	6.481824
0.44	0.0	6.482241
0.45	0.0	6.480637
0.46	0.0	6.484189
0.47	0.0	6.480052
0.48	0.0	6.478052
0.49	0.0	6.483467
0.50	0.0	6.478613
0.51	0.0	6.482220
0.52	0.0	6.484131
0.53	0.0	6.478909
0.54	0.0	6.481872
0.55	0.0	6.483348
0.56	0.0	6.483573
0.57	0.0	6.482490
0.58	0.0	6.482545
0.59	0.0	6.480305
0.60	0.0	6.482202
0.61	0.0	6.487876
0.62	0.0	6.486378
0.63	0.0	6.481999
0.64	0.0	6.480388
0.65	0.0	6.483869
0.66	0.0	6.481656
0.67	0.0	6.480372
0.68	0.0	6.483594
0.69	0.0	6.481613
0.70	0.0	6.484184
0.71	0.0	6.484251
0.72	0.0	6.481886
0.73	0.0	6.485081
0.74	0.0	6.481639
0.75	0.0	6.483035
0.76	0.0	6.484921
0.77	0.0	6.483743
0.78	0.0	6.479068
0.79	0.0	6.481485
0.80	0.0	6.485697
0.81	0.0	6.480349
0.82	0.0	6.482702
0.83	0.0	6.483442
0.84	0.0	6.481515
0.85	0.0	6.482382
0.86	0.0	6.481743
0.87	0.0	6.481338
0.88	0.0	6.481757
0.89	0.0	6.482022
0.90	0.0	6.485049

Alpha	Lambda	RMSE
0.91	0.0	6.478456
0.92	0.0	6.484969
0.93	0.0	6.480006
0.94	0.0	6.481876
0.95	0.0	6.481316
0.96	0.0	6.483397
0.97	0.0	6.480628
0.98	0.0	6.482829
0.99	0.0	6.480227
1.00	0.0	6.478199

In figure 4.1 we can see the test RMSE for the optimal λ at each α value. The highest predictive accuracy is found when α is equal to .03 and λ is equal to .5. I am going to train a model with these specifications and discuss the results in the next section.

4.3: Results

This model displays an RMSE of 6.5, meaning that the average predicted departure delay time is 6.5 minutes above or below the actual value. The table below lists the coefficients used in the model.

Table 4: Table 4.2

	Elastic
(Intercept)	-3.1548202
day	0.0400830
sched_dep_time	0.0016100
sched_arr_time	0.0003574
carrierAA	-0.3038974
carrierAS	0.0000000
carrierB6	-0.0258666
carrierDL	-0.5878287
carrierEV	0.0000000
carrierF9	-2.7241841
carrierFL	2.1329224
carrierMQ	-1.1068599
carrierUA	2.0121380
carrierUS	-1.0044055
carrierVX	0.3258297
carrierWN	3.4571575
carrierYV	-3.0837027
originJFK	-0.0588895
originLGA	-1.1258705
destATL	0.6513212
destAUS	-0.0550852
destAVL	-1.9129040
destBHM	1.5134414
destBNA	-0.7122335
destBQN	0.3426492
destBUR	-0.8548302
destBZN	11.1733875
destCAE	2.6643476

	Elastic
destCHS	-1.0197265
destCLT	-0.3757011
destCVG	-1.3608703
destDAY	-1.2572931
destDEN	1.5237860
destDFW	-0.7051624
destDSM	-4.0903987
destDTW	-0.5781220
destEGE	0.0000000
destEYW	0.3974454
destFLL	1.3415970
destGRR	0.0000000
destGSP	-2.1476917
destHOU	0.0593652
destIAH	1.5407387
destIND	-0.6364530
destJAC	3.7382852
destJAX	-0.6953004
destLAS	0.0340942
destLAX	0.0000000
destLGB	0.0000000
destMCI	-0.1750482
destMCO	-0.1697296
destMDW	0.0000000
destMEM	-0.2463892
destMIA	0.8907980
destMKE	0.4132115
destMSN	0.0000000
destMSP	0.3377086
destMSY	-1.5565008
destOAK	-0.5454240
destOKC	-2.3490259
destOMA	-2.6468001
destORD	-0.5386442
destPBI	0.7553997
destPDX	-0.2719727
destPHX	0.5903069
destPSE	1.1813934
destPSP	6.0151518
destRSW	-0.2285254
destSAN	0.6162008
destSAT	-1.3367159
destSAV	-0.6933869
destSDF	-0.9000163
destSEA	1.0535108
destSFO	0.0000000
destSJC	-1.2053127
destSJU	0.5623508
destSLC	1.1093043
destSMF	-1.3173355
destSNA	-0.2659681
destSRQ	-0.4594589

	Elastic
destSTL	0.3092736
destSTT	0.3994910
destTPA	0.0000000
destTUL	-0.7156785
destTYS	-1.7112189
destXNA	-1.2536887
distance	0.0001997

Our elastic net model uses 76 out of the 86 variables. There are 10 zero values, so we have a total of 77 β values in our model including the intercept. I was hoping that elastic net would eliminate more variables, however it decided that 76 variables displayed enough value to be used in regression. It is interesting to evaluate the relationship the different levels of the categorical variables have with th response variable. For example, the different values of carrier have very different effects on the predicted departure delay. If a flight is done through ExpressJet Airlines (EV), it has no effect on the predicted departure delay. Meanwhile, a flight done through Mesa Airlines (YV) decreases the predicted delay time by over 3 minutes, and a flight done through United Airlines (UA) increases the predicted delay time by over 2 minutes. This model is much more complex than the the one built with KNN, but provides a 14.47% increase in the RMSE. This concludes the analysis of an elastic net model, and I will discuss my third modeling technique in the next section.

Section 5: Bagged Regression Forests

5.1: Introduction

The final technique I will be using is a bagged regression forest. There are several reasons why I chose a bagged regression forest, most importantly because they are simple to understand and often provide a higher predictive accuracy than individual trees. This model will be especially useful in predicting a flight's departure delay because it is extremely user-friendly. With a bagged regression forest, I am able to easily use all features. I am also able to build a model with outliers using a regression forest, because the data can be split in so many different ways.

5.2: Method

In order to understand a bagged forest, it is important to first understand a regression tree. A regression tree begins with all the data in the data set in an initial node. At this phase, the predicted value is the mean of the entire data set. In our case, this would mean that the initial node would predict a flight to be 16.8 minutes late for the model including outliers, and 0.3 minutes late for the data excluding outliers.The data is then split into two different nodes according to one of the features. We choose which feature and value to split on based off the residual sum of squares (RSS), which provides a measure of how much error is present in our data. To build a tree we must find the RSS for each possible split at every value of each feature. We choose the split that gives us the lowest RSS. We can continue this process to split nodes and grow our tree until we have reached a result we are satisfied with

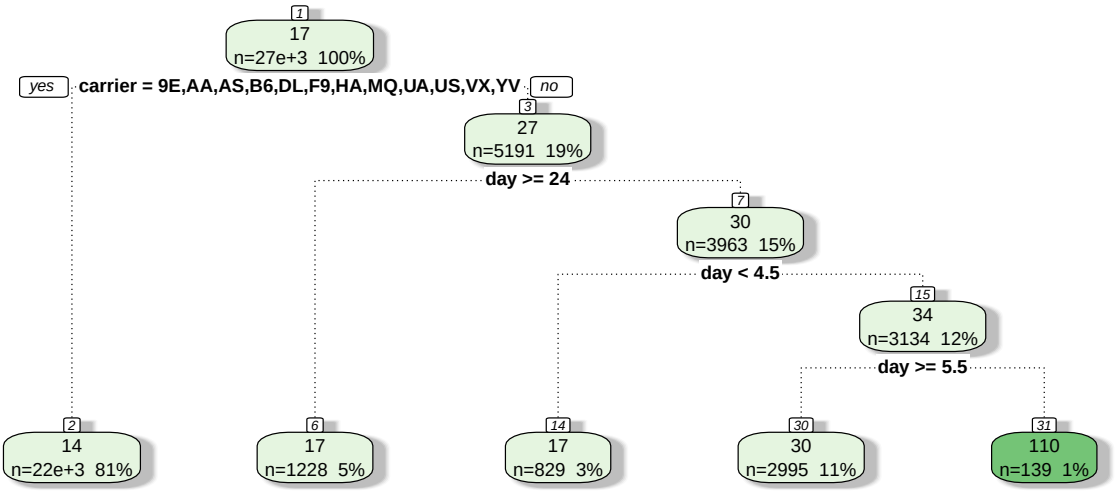


Figure 5.1

In Figure 5.1 above, I built a sample regression tree to demonstrate the process. I only used the day and airlines features to build this tree for simplicity. Notice how 100% of the data is placed in the initial node and given a predicted value of 17 which signals that the flight will be 17 minutes delayed. The first split was chosen to divide the airlines into two different groups in order to minimize the RSS. The lowest RSS was found when splitting the data between the 12 airlines listed and the airlines not listed. The tree continued to split until a stopping parameter was reached. To find how many minutes we would predict a flight's departure to be delayed, we would use the information we know and follow the tree until its final node. For example, a flight operated by ExpressJet Airlines (EV) on the 22nd of December would be predicted to have a delay of 17 minutes.

A bagged classification forest is a collection of trees. In order to build multiple trees, we first have to create a number of bootstrap samples. A bootstrap sample is as large as the original data set, and is created by sampling with replacement from the original data set. We then complete the process of building a tree for each bootstrap sample. In order to predict how late a flight's departure will be delayed, we would find the prediction from each tree and choose the most popular classification. It is important to build a large number of trees, most forests have several hundreds or even thousands. The first model will consider the outliers, as forests are able to split many ways and could potentially discount the effect of extremely high or low departure delay times. The second model will not consider the outliers, to build a more accurate model that can be used for a much smaller scope of data. In the next section I will discuss the results of my trained model. I have chosen to use 200 trees for my forest with outliers, and 1000 trees for my forest without outliers. I have chosen to do this because I want to use as many trees as possible, but the data set with outliers has over 27,000 rows and my computer does not have the capabilities of producing as many trees with such a large data set. Now that I have explained how a bagged classification forests works, I will train two models using 1000 trees and all 7 features.

5.3: Results

Table 5: Table 5.1: Forest with Outliers

Number.of.Trees	MSE	RMSE
200	1497	38.7

Table 6: Table 5.2: Forest without Outliers

Number.of.Trees	MSE	RMSE
1000	41.31	6.4

The bagged regression forest using flight delays between -43 and 896 minutes has an RMSE of 38.7, as is displayed in Figure 5.2. This means that the prediction given by our model is, on average, 38.7 minutes higher or lower than its actual value. Given the large spread of this data, our model does a satisfactory job in predicting how late a flight will be delayed. I am not sure how useful this information would be in practice for analyzing flight patterns, however it could be useful to plan for extreme situations.

We can observe in Figure 5.3 that the bagged regression forest using flight delays between -20 and 20 minutes has an RMSE of 6.4. This means that the prediction given by our model is, on average, 6.4 minutes higher or lower than its actual value. This gives a window that is 12.8 minutes long from an interval of only 40 minutes. I would interpret this to mean this model did not give a prediction as precise as I would have hoped, but it is still better than other models we built. I am going to analyze the importance of each feature to see if it gives any further value to my exploration.

Figure 5.2

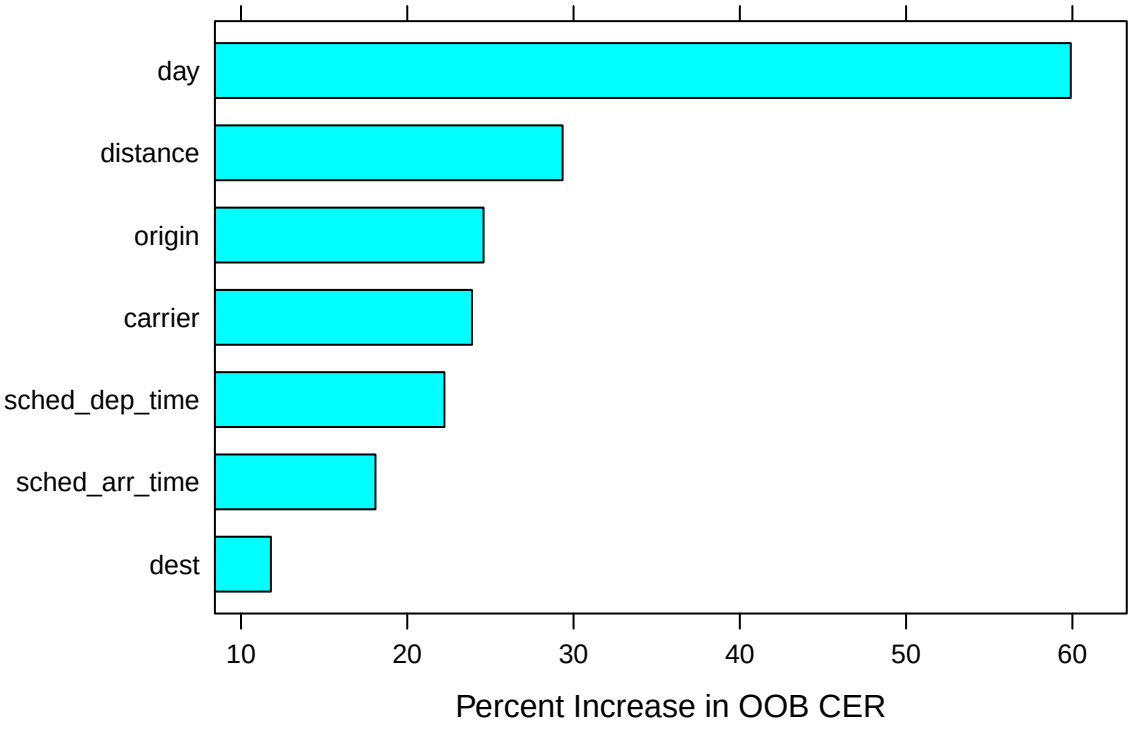
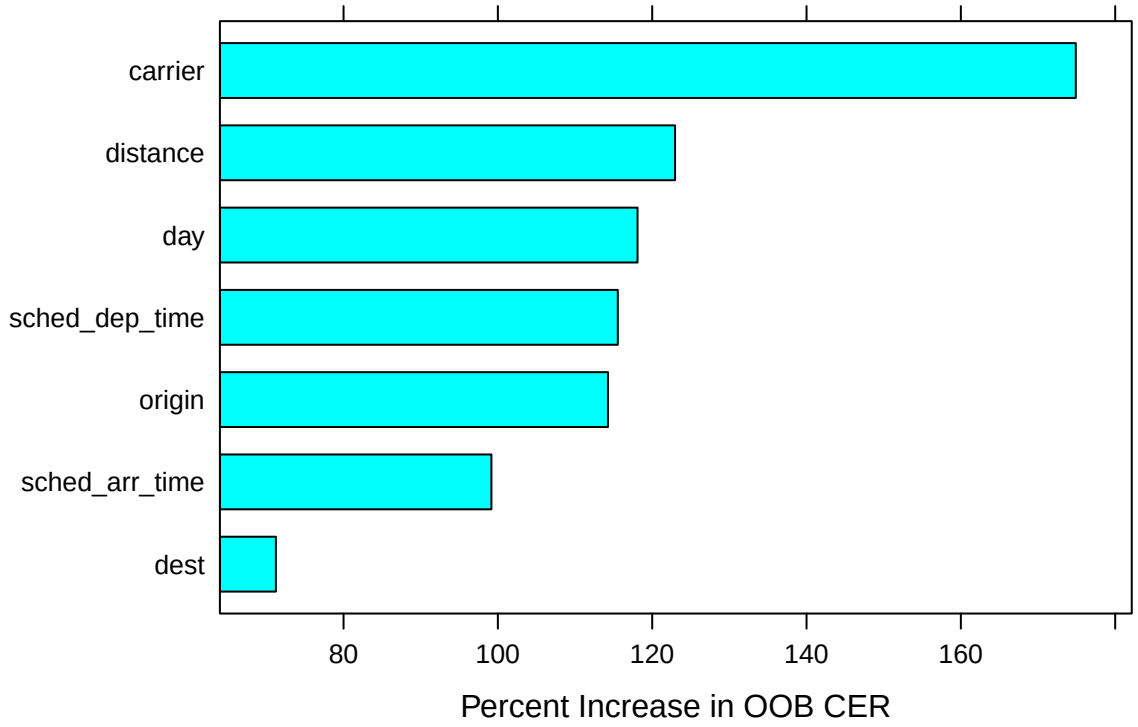


Figure 5.3



Importance is found by removing the relationship between the response variable and a certain feature, and observing the increase in the error rate. Figure 5.2 shows this data from the forest that includes the extreme outliers. In this data, it seems that the day of the flight is the most important feature, and the error rate increases by almost 60% with that information. This information is important because if I were to attempt to build better models in the future, I would know that using the day of the flight would be helpful to include. Figure 5.3 shows the importance of features when only considering flights that were delayed between -20 and 20 minutes. When a flight's departure is not significantly delayed, it seems that the operating airline feature has the highest importance. This model has 5 features that decrease the error rate of more than 100% when used. When predicting a normal flight delay, the operating airline, distance of the flight, day of the flight, scheduled departure time, and origin airport all have a great impact on how delayed the flight will be. I would interpret this information to mean that all 5 of these features are especially important when predicting an arrival delay, and the day of the flight should be taken into special consideration to determine if the flight will experience a major delay.

Conclusions

We have now built 5 different models that are able to predict the delay in a flight's departure for flights departing from New York City in December. Two of these models can be used for flights with extreme delays, and the other three can be used when a flight is delayed between an average interval. For predicting normally delayed or extremely delayed flights, I would suggest using the bagged regression forest model.

KNN	Elastic	Forest

Table 7: Table 6.1: RMSE of Models to Predict a Normal Delay

KNN	Elastic	Forest
7.6	6.5	6.4

I would suggest a regression forest to predict a normal delay because it gives us the highest RMSE at the most simple model. In Table 6.1 we can observe that the model built by KNN gives a RMSE of a whole minute greater than the other two models. While the RMSE measures of the elastic net and forest models are very similar, it is important to consider the complexity of each model. We discovered that the elastic net model built an equation with 77 beta variables. This is a large number of beta values for a 0.1 increase in the RMSE. Bagged forests are much easier to use, thus the simplicity makes it more usable than the elastic net model. From this model, we can develop a 12.8 minute window of error for a flight's departure. We can also conclude, from the importance graph in section 5.3, that the airline carrier has the greatest impact on if a flight will experience a departure delay between -20 and 20 minutes. This information can be used by transportation analysts to improve several operation processes.

Table 8: Table 6.2: RMSE of Models to Predict an Extreme Delay

KNN	Forest
45.7	38.7

Out of the two models I built, I would suggest a regression forest to predict an extreme delay because it provides a predictive accuracy 15.3% higher than that obtained by KNN. Predictive accuracy is important because we want to build a model that can be used by transportation analysts to better plan for delays in flight departures. While this is the best of the two models I built, I still do not think it is effective enough to be used in practice, as it gives us a 77.4 minute window of error in a flight's departure. I am not confident that a window this large would be significantly helpful, and I do not think these models were able to appropriately account for the extreme outliers. I anticipated that using the outliers in flight delay would provide a model with high variance, however I wanted to explore the data in case one of my models was able to still provide accurate predictions.

If transportation analysts were to use my models, I would suggest they initially use the bagged forest created with data on departure delays between -20 and 20 minutes to determine a predicted departure delay and a 12.8 minute window of error. Next, I would advise them to pay special attention to the day in December that the flight is to take place. I suggest they do this because Figure 5.3 in section 5.3 indicated that they day feature is important in predicting the delay time of extremely delayed flights. This concludes my analysis on building models to predict flight departure delays.

Works Cited

NYC_Flight_Delay, Version 1. Retrieved December 2, 2022 from https://www.kaggle.com/datasets/lampubhutia/nyc-flight-delay.

Feature explanation: https://docs.google.com/document/d/1OaYQvfeBlWoqjpncpbcbfE01EXAzyEChTDTpSAWJXWY/edit?usp=sharing