# EMS Data - Predicting Ambulance Response Times

Data Analytics - Assignment 6 (Term Project)

Anya Tralshawala

12/09/2022

Dr. Thilanka Munasinghe

Level 4000

## Abstract and Introduction

As we know, quicker ambulance response times lead to greater survival rates. If we decrease response time by one minute, survival rates of patients increase by 20-30%[1]. We can use predictive analytics to estimate ambulance response times as a function of time, location, operational and environmental factors. By predicting response times, we can determine where to assign resources. Current models don't account for socio-demographic or political factors and by analyzing these we can improve models and increase the chance of survival.

It would also be interesting to see how political alignment and congressional district may impact response times. We hypothesize that response times would be slower in districts that lean red or more republican since democrats believe more in government intervention and invest in services that are controlled by the government such as EMS. During pre-processing some data were merged and joined on the congressional district column to allow us to add congressional district to the EMS data.

## Data Description and Exploratory Data Analytics

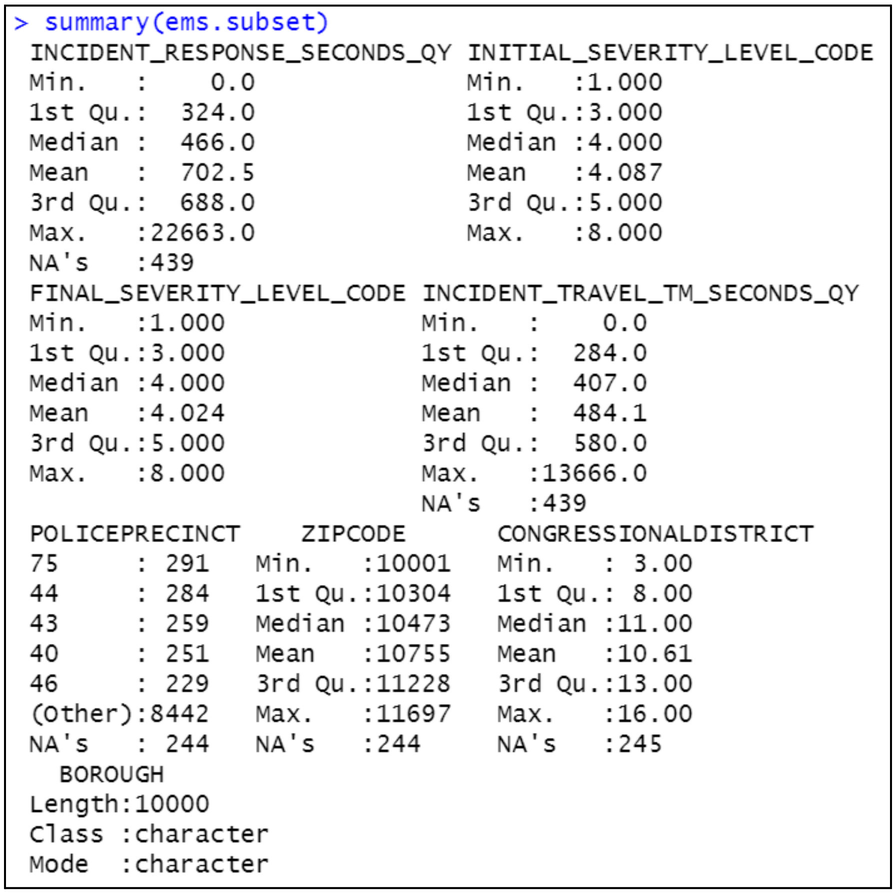
The data was provided to us by NYC Open data. This data was collected from EMS Computer Aided Dispatch System in NYC from 2008-2016[2]. "The data spans from the time the incident is created in the system to the time the incident is closed. It covers information about the incident as it relates to the assignment of resources and the Fire Department's response to the emergency." Due to data privacy policies, personally identifying information (PII) was omitted from this data. As per Health Insurance Portability and Accountability Act (HIPAA), patients have rights over their health data and healthcare workers are only at liberty to share if and only if permitted by the patient. This is clearly not feasible when working with such massive datasets so PII is redacted. This data has 23.3 million rows with 31 columns. This is a massive dataset and not feasible to work with all at once without R crashing on us. When reading this data I read in a randomized 10000-row subset of data.

Additionally, I found a data set from New York State that determined the political party affiliation of each of the congressional districts in NYS based on New York State Elected Officials. This data was collected in mid-November 2022 following the midterm elections in early November. This data was found, scraped, and collected from election results provided publicly by the NYS government, on elections.ny.com[3]. According to the US embassy[4], "Republicans broadly advocate for traditional values, a low degree of government interference, and large support of the private sector." Based on this viewpoint I thought it would be interesting to throw political party affiliation into the picture to see if response times varied. I would predict that response times would be slower in districts that lean red or more republican. On the other hand, NYC as a whole is fairly blue politically and progressive therefore response times shouldn't vary too much based on congressional district and political affiliation. This is something that I was interested in researching and paired with the EMS data described above the models below and analysis was conducted.

## Analysis

Prior to model development and applying these models to the EMS dataset. Many factors went into making this prediction model.

After looking through the data and performing preliminary analysis, like head() and summary() and creating boxplots other factors were determined to potentially be interesting to look at. I created a subset of the entire EMS dataset including features that seemed to logically impact response time. Summaries on all columns in ems_subset are seen below:



```
> summary(ems.subset)
 INCIDENT_RESPONSE_SECONDS_QY INITIAL_SEVERITY_LEVEL_CODE
 Min.   :    0.0              Min.   :1.000
 1st Qu.:  324.0              1st Qu.:3.000
 Median :  466.0              Median :4.000
 Mean   :  702.5              Mean   :4.087
 3rd Qu.:  688.0              3rd Qu.:5.000
 Max.   :22663.0              Max.   :8.000
 NA's   :439
 FINAL_SEVERITY_LEVEL_CODE INCIDENT_TRAVEL_TM_SECONDS_QY
 Min.   :1.000             Min.   :    0.0
 1st Qu.:3.000             1st Qu.:  284.0
 Median :4.000             Median :  407.0
 Mean   :4.024             Mean   :  484.1
 3rd Qu.:5.000             3rd Qu.:  580.0
 Max.   :8.000             Max.   :13666.0
                           NA's   :439
 POLICEPRECINCT    ZIPCODE      CONGRESSIONALDISTRICT
 75     : 291   Min.   :10001   Min.   : 3.00
 44     : 284   1st Qu.:10304   1st Qu.: 8.00
 43     : 259   Median :10473   Median :11.00
 40     : 251   Mean   :10755   Mean   :10.61
 46     : 229   3rd Qu.:11228   3rd Qu.:13.00
 (Other):8442   Max.   :11697   Max.   :16.00
 NA's   : 244   NA's   :244     NA's   :245
   BOROUGH
 Length:10000
 Class :character
 Mode  :character
```

*Figure 3.1 - Summary of ems_subset*

Some factors that I found to be impactful were; INITIAL_SEVERITY_LEVEL_CODE, FINAL_SEVERITY_LEVEL_CODE, BOROUGH, INCIDENT_DISPATCH_AREA, ZIPCODE, POLICEPRECINCT, INCIDENT_DT, INITIAL_CALL_TYPE, and FINAL_CALL_TYPE. Logically these columns should be impacting INCIDENT_RESPONSE _SECONDS_QY and that is why they were chosen to evaluate and pass into the models that will be explained further in the next section. Prior to any model generation, exploratory data analysis (EDA) was performed. Afterward, boxplots and histograms were generated to visualize the data. One boxplot that I generated, seen below, that was interesting was boxplots on POLICEPRECINCT vs INCIDENT_RESPONSE_SECONDS_QY. This shows us on average how quickly each of the precincts in NYC takes to respond to calls.

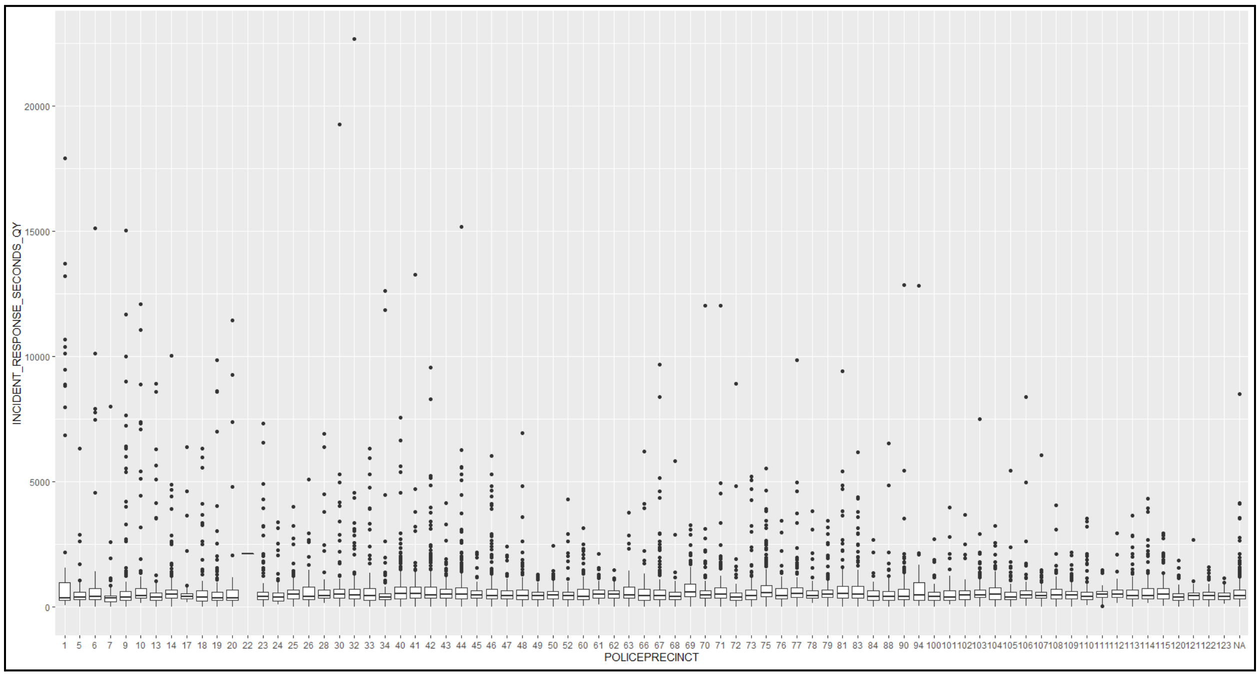
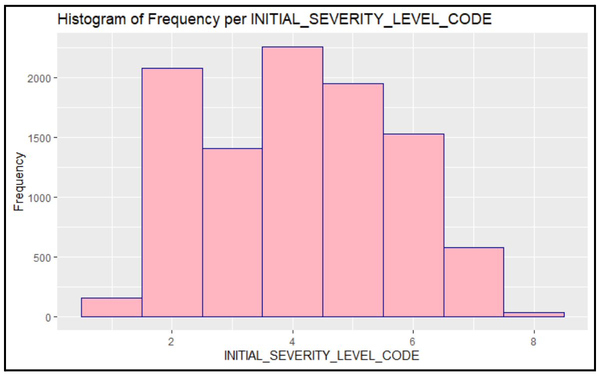


*Figure 3.2 - POLICEPRECINCT vs INCIDENT_RESPONSE_SECONDS_QY boxplot*

As you can see above, there are many, many outliers in the data that can skew values like mean and median. As a part of EDA, outlier detection tests were completed like cook's distance and interquartile range. Next, the outlier points were removed by creating a subset where only the points in the lower and upper ranges were kept. Even though there was a subset with no outliers that was generated. I chose to create models on the datasets with the outliers since this would be more realistic in real life. Including the outliers mimics the uncertainties that are in the real world.

Additionally, I created histograms to see how data was distributed. The features seen below are INITIAL_SEVERITY_LEVEL_CODE and FINAL_SEVERITY_LEVEL_CODE. You can see that the general severity distribution is similar, meaning that the EMS operator taking the calls labeled the call severity pretty accurately. This is good because this helps dispatch prioritize calls. The severity is pretty evenly distributed from 1 to 8. For this factor 1 is the highest and 8 is the lowest.



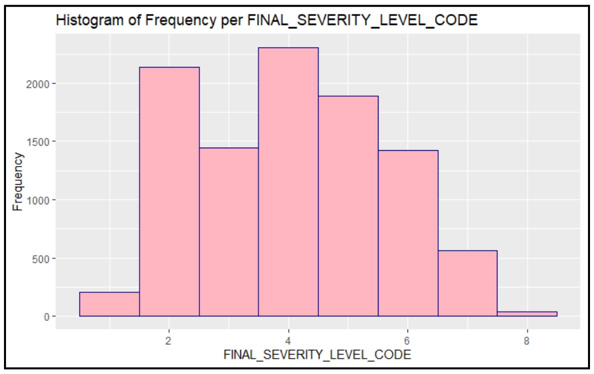


*Figure 3.3 (left) & Figure 3.4 (right) - INITIAL_SEVERITY_LEVEL_CODE and FINAL_SEVERITY_LEVEL_CODE histograms*

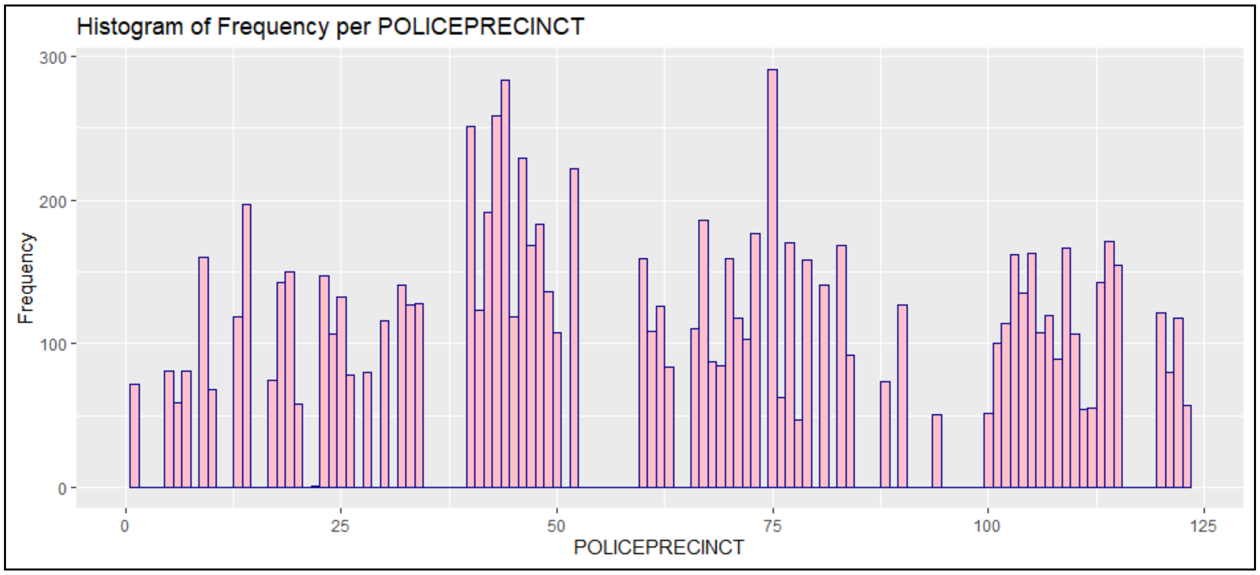


*Figure 3.5 - Histogram of frequency per Police Precinct*

The histogram above shows that some precincts get more calls than others. This might be because of the size or what part of NYC the precinct is located in. Additionally, after these plots were created, prior to feeding them into any of the models preprocessing was done I performed as.numeric() to cast all of the values to quantities and made sure to remove all the null values by using the is.na() command.

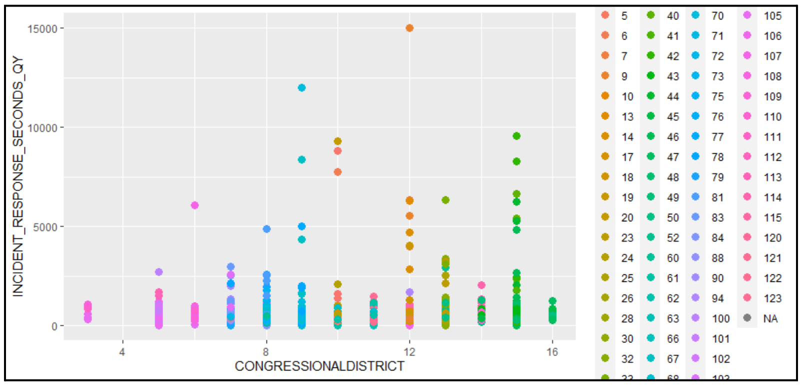


*Figure 3.6 - Congressional District vs Response time based on a police precinct*

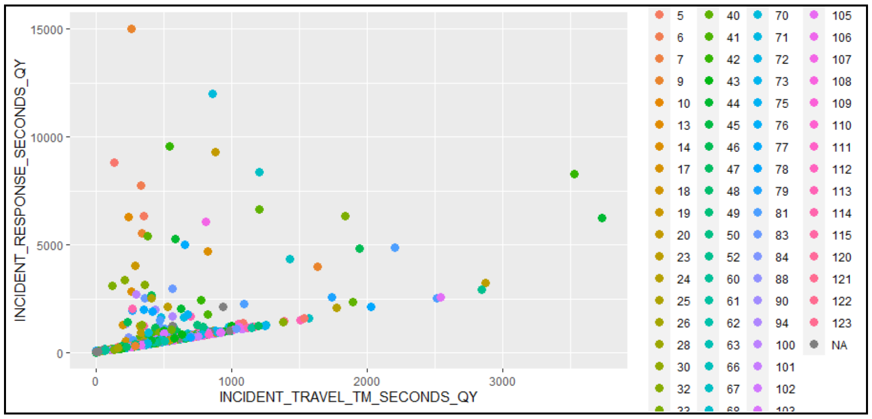


*Figure 3.7 - Travel time vs Response time based on a police precinct*

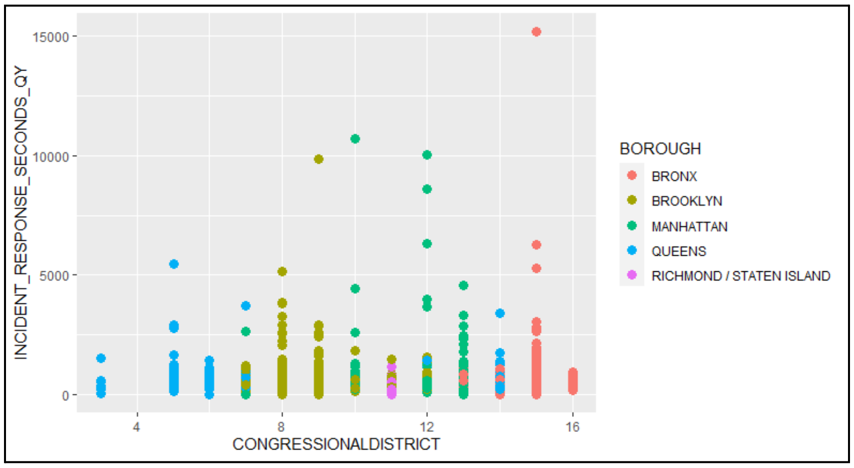


*Figure 3.8 - Congressional District Vs Response Time classified by Borough*

The plots above were created using ggplot2[6] and show various different features plotted against each other and grouped upon other variables. These plots allow us to see if there are any relationships present.

In terms of uncertainty, data accuracy is unknown since it is user generated and dependent on 911 phone calls. You don't know how accurate the data is since it is user-generated and priority is determined based on the phone calls. So if the patient leaves out information the data is classified incorrectly which makes training a model more difficult.

The ultimate goal of this project is to be able to predict response time, determine which factors cause slowdowns, and try to improve those features to increase survival rates.

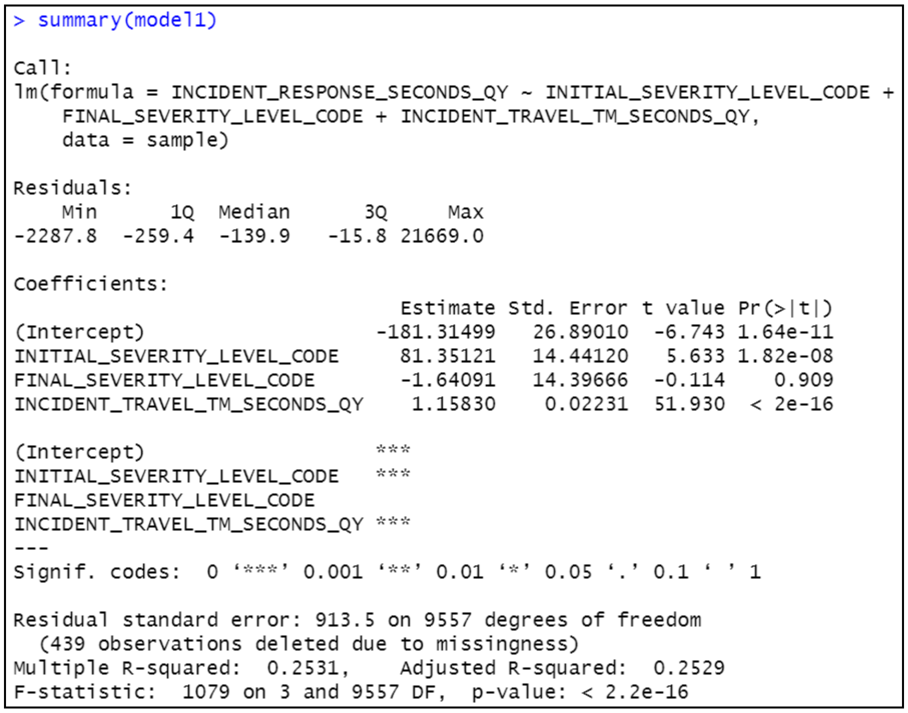
## Model Development and Application of model(s)

In total, I generated 3 models for this project. I first generated a Mulitvatiate Regression, then K-Means cluster plot, and lastly a KNN classifier.

### Multivariate Regression:

Multivariate regression helps determine the degree to which the independent variable and different dependent variables are linearly related to each other. This regression was used to determine which factors influence response time the most.
INCIDENT_RESPONSE_SECONDS_QY was the dependent/ response variable for this regression model. The other factors that we are testing are the independent variables/ predictors. In this case, I chose to select INITIAL_SEVERITY_LEVEL_CODE, FINAL_SEVERITY_LEVEL_CODE, INCIDENT_TRAVEL_TM_SECONDS_QY.

The R-Squared value helps us determine this. In the regression shown below the R-Squared value is less than 0.9 which means that there is a low level of correlation between factors. In the case of this model, the Adjusted R-Squared value is 0.2529. This relatively low R-Squared value also shows that the model can not be considered to be reliable. We can also see below in figure 4.1 that “the residual standard error which is the standard deviation of the residuals – Smaller residual standard error means predictions are better.” The residual standard error should ideally be as low as possible and for this model, it is close to 913 on 9557 degrees of freedom which is very high.



```
> summary(model1)

Call:
lm(formula = INCIDENT_RESPONSE_SECONDS_QY ~ INITIAL_SEVERITY_LEVEL_CODE +
    FINAL_SEVERITY_LEVEL_CODE + INCIDENT_TRAVEL_TM_SECONDS_QY,
    data = sample)

Residuals:
    Min      1Q  Median      3Q     Max
-2287.8  -259.4  -139.9   -15.8 21669.0

Coefficients:
                               Estimate Std. Error t value Pr(>|t|)
(Intercept)                  -181.31499   26.89010  -6.743 1.64e-11
INITIAL_SEVERITY_LEVEL_CODE    81.35121   14.44120   5.633 1.82e-08
FINAL_SEVERITY_LEVEL_CODE      -1.64091   14.39666  -0.114    0.909
INCIDENT_TRAVEL_TM_SECONDS_QY   1.15830    0.02231  51.930  < 2e-16

(Intercept)                   ***
INITIAL_SEVERITY_LEVEL_CODE   ***
FINAL_SEVERITY_LEVEL_CODE
INCIDENT_TRAVEL_TM_SECONDS_QY ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 913.5 on 9557 degrees of freedom
  (439 observations deleted due to missingness)
Multiple R-squared:  0.2531,	Adjusted R-squared:  0.2529
F-statistic:  1079 on 3 and 9557 DF,  p-value: < 2.2e-16
```

*Figure 4.1 - Multivariate Regression for Model 1 (INCIDENT_RESPONSE_SECONDS_QY ~ INITIAL_SEVERITY_LEVEL_CODE + FINAL_SEVERITY_LEVEL_CODE + INCIDENT_TRAVEL_TM_SECONDS_QY )*

In terms of tuning, I played around with other models and built other regressions. They all resulted in similar R-Squared and residual standard error values. Some slightly higher and some slightly lower. I chose to develop this model to see if there exists a correlation between these variables. And clearly, there is some sort of correlation, just not as strong linearly.

## K-Means Clustering Algorithm:

The next model I developed was the K-means clustering algorithm. K-means finds groups that have not been explicitly labeled in the data. The plot you see below was generated by using the ggplot2, dplyr[7], and ggfortify[8] libraries. These plots help make better visualizations with parameters that can be customized. First, I assigned a variable called data to a subset of the factors from the total I wanted to focus on during model generation. Figure 4.2 below plots all the points fed into the clustering model after they were normalized and clusters them based on these factors.

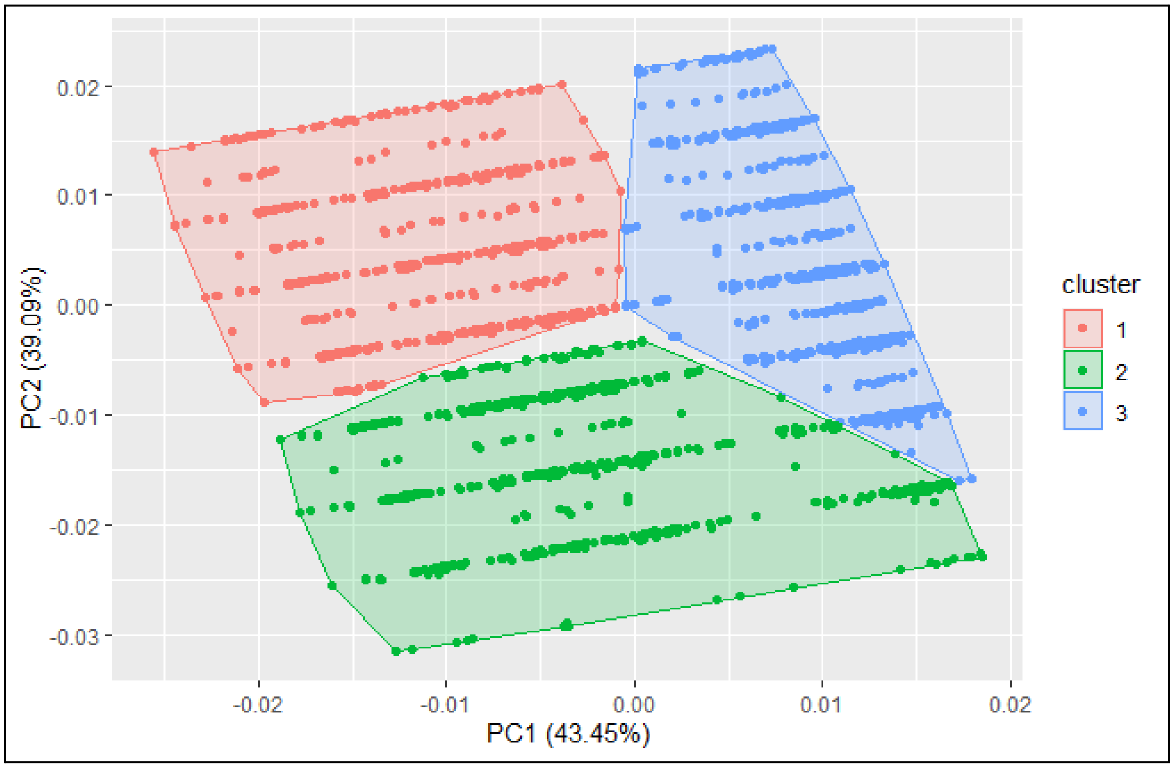


*Figure 4.2 - K-Means Cluster Plot*[9]

As you can see above the two axes are PC1 and PC2. This refers to the principal component of the data. Principal Component Analysis is a technique for reducing the dimensionality of a data set by transforming it into a new coordinate system such that the first principal component (PC1) has the largest possible variance, the second principal component (PC2) has the second largest variance, and so on. As seen above PC1 has a variability of 43.45 and PC2 has a variability of 39.09. Increasing the variance increases the accuracy of the clustering. In simple terms, variability refers to how separate the clusters are.

After I created the model above, I tested variability based on various clusters using the within-group sum of squares (WSS). The WSS is a measure of the variability of the data points within each cluster. It is calculated as the sum of the squared distances of each data point in the cluster to the centroid (mean) of the cluster. A lower WSS value indicates that the data points in the cluster are more similar to each other, while a higher WSS value indicates that the data points in the cluster are more diverse.[9] Figure 4.3 below shows this plot. I chose to still model with 3 clusters because the variability of PC1 and PC2 were still pretty similar and visually it was easier to cluster with a lower k value even though the elbow seems to be around 7 clusters.

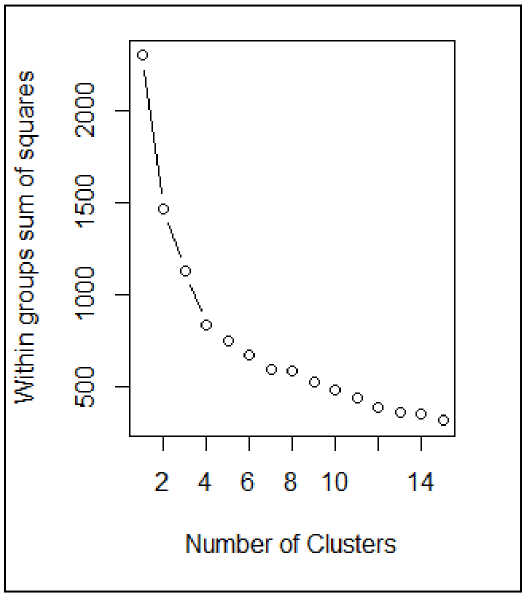


*Figure 4.3 - Number of Clusters vs WSS Plot*

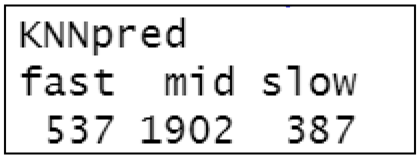
## KNN Classification:

The last model I built was a KNN classifier. I used the custom column I generated based on INCIDENT_RESPONSE_SECONDS_QY. This column was generated during the EDA process and used to see how and if response time was impacted when variables were plotted. I initially used this column to find relationships and try to see if correlations existed. The line of code below was used to cut INCIDENT_RESPONSE_SECONDS_QY into 3 groups; fast, mid, and slow.

```
total$responseSpeed <- cut(total$INCIDENT_RESPONSE_SECONDS_QY,
br=c(3,300,900,22663), labels = c("fast", 'mid', 'slow'))
```

Fast was classified as less than 300 seconds/ 5 minutes, mid was less than 900 seconds/ 15 minutes, and lastly slow was anything greater than 15 mins. After this, I used as.numeric() to cast the values as a number and is.na() to remove null values. Once this was completed this newly generated factor was ready to be used in modeling.

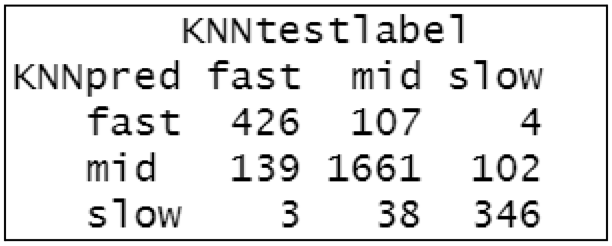
To generate the KNN model I initially started out by finding the min and max INCIDENT_RESPONSE_SECONDS_QY values. These values were used for normalization. The data in the total data frame needed to be normalized before feeding it to the model. We need to normalize data if the scale of features is very different. “This is because the distance calculation done in KNN uses feature values. When the one feature values are large than the other, that feature will dominate the distance hence the outcome of the KNN[5].” After the normalization step, the data was split into a training and testing set. The KNNpred is seen below Figure 4.4 shows how many values are categorized in each response speed.



```
KNNpred
fast  mid slow
 537 1902  387
```

*Figure 4.4 KNNpred response time*

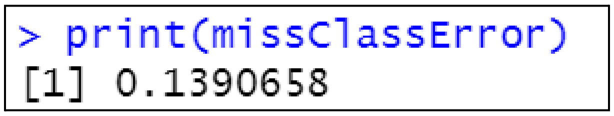
After this was predicted A confusion matrix was generated to compare KNNpred value to the actual values in response speed. This is used for validation purposes when evaluating the KNN models. The matrix is shown below in Figure 4.5.



```
       KNNtestlabel
KNNpred fast  mid slow
   fast  426  107    4
   mid   139 1661  102
   slow    3   38  346
```

*Figure 4.5 - KNNpred Confusion Matrix*

In terms of significance tests, I calculated the error rate and misclassification rate on the KNN model I generated above. The error rate was calculated by comparing the KNNtestlabel value which represents the correctly classified data and the KNNpred value with represents the value my model predicted.



```
> print(missClassError)
[1] 0.1390658
```

*Figure 4.6 - missClassError*

The misclassification rate was approximately 0.14, as seen above in Figure 4.6 this is calculated by taking the mean value of all the properties whose KNNtestlabel is not equal to the KNNpred value. According to the misclassification rate, the accuracy of this model is approximately 86%.

The K-value plot, figure 4.7 below shows the most optimal k-value is 1 when KNN was performed on the model. This the k values I used to evaluate the model. The model used CONGRESSIONALDISTRICT, INCIDENT_RESPONSE_SECONDS_QY, INITIAL_SEVERITY_LEVEL_CODE, FINAL_SEVERITY_LEVEL_CODE, INCIDENT_TRAVEL_TM_SECONDS_QY, and POLICEPRECINCT.

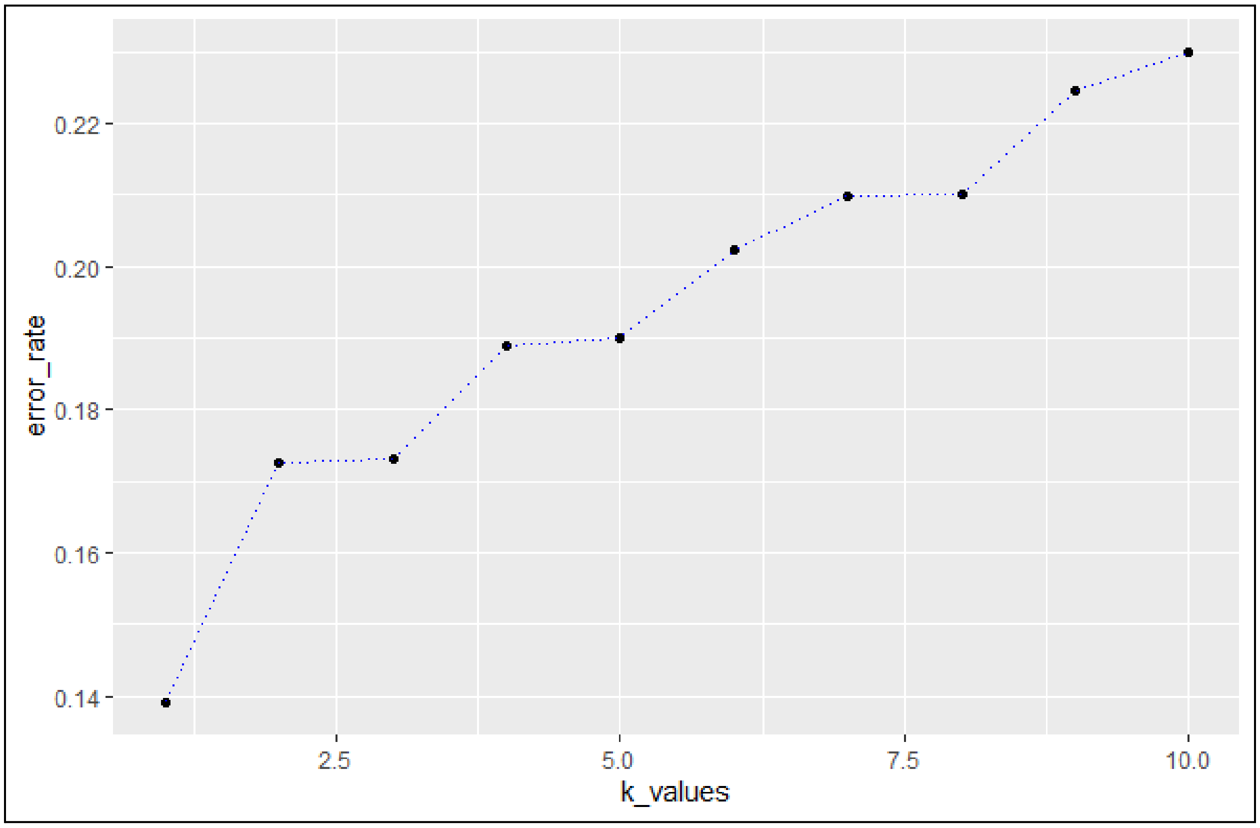


*Figure 4.7 - K-Values vs Error_rate*

## Conclusions and Discussion

Prior to analysis, it was hypothesized that political factors and which way the district leaned; red or blue would make an impact on EMS response times along with data like severity level, travel time to the incident, which borough the incident was in, and police precinct to name a few.

After conducting multivariate regression and feeding in a model using initial, final severity level, and incident travel time it was found that there is not a very strong linear correlation. The adjusted R-Squared value as seen in the modeling above was 0.2529. This means that there is some correlation, just not a strong one.

The next model generated was K-means clustering. PC1 had a variability of 43.45 and PC2 had a variability of 39.09. Currently, the K-means clustering model was generated with K = 3. The clustering could be modeled using various K values as per the elbow plot to see if variability increased.

Lastly, according to the models you can see that KNN classification is pretty accurate when you select the appropriate k-value and parameters to train the model. In the case of the model generated above the accuracy was 86%. More factors could have been added to this model to see if perhaps more parameters increase accuracy. Multiple KNN classification models could be run simultaneously with different factors to see which factors are influential.

In the future, if we were to merge congressional district data with EMS data and re-run these models we would make sure the data are accurate to the 2008-2016 time frame and not 2022. Some districts may have swung to the opposite political party and this might have impacted the model and resulted in no clear correlation being present when the incidents actually occurred. This was clearly an oversight when generating the datasets and models. In conclusion, by generating models we can determine where to assign resources to decrease response times and increase rates of survival. We can refine these models and increase accuracy to clearly see how factors like political affiliation, sociodemographics and incident severity impact response times.

## GitHub URL:

https://github.com/anya-tralshawala/DataAnalytics_TermProject_Anya_Tralshawala

## Citations

1. https://mediasite.mms.rpi.edu/mediasite/Play/77c72d7a29d44838a872d3a19e3086f31d
2. https://data.cityofnewyork.us/Public-Safety/EMS-Incident-Dispatch-Data/76xm-jjuj
3. https://www.elections.ny.gov/district-map.html
4. https://dk.usembassy.gov/da/youth-education-da/the-american-political-system/the-democrats-and-the-republicans/
5. https://stats.stackexchange.com/questions/287425/why-do-you-need-to-scale-data-in-knn#:~:text=If%20the%20scale%20of%20features,the%20outcome%20of%20the%20KNN
6. https://ggplot2.tidyverse.org/
7. https://dplyr.tidyverse.org/
8. https://cran.r-project.org/package=ggfortify
9. https://rstudio-pubs-static.s3.amazonaws.com/169319_b157b968f82e45df9e6d1ce8ee679fd3.html