Subway Delays and Crime in NYC

Sara Douglas, Fritz Grunert, Mason Lonoff, Susan Lu

Dev10 Data Science and Engineering Capstone

February 2, 2023

Problem Statement:

Since all group members live in New York City, the intersection of Subway delays and crime was interesting to us as it impacts our daily lives. After exploring the available data we decided to focus on answering the following questions:

- Are NYPD complaints and subway alerts/delays related?
- Does the distance to the nearest subway station affect crime?
- What crimes occur most when there are subway delays?
- Has the COVID-19 Pandemic had any impact on frequency of complaints or subway alerts/delays?

Data:

For our study we used three data sources.

1. NYPD Complaint Data Historic | NYC Open Data (cityofnewyork.us)

This dataset includes all valid felony, misdemeanor, and violation crimes reported to the New York City Police Department (NYPD) from 2006 to the end of 2021. It also contains information about the location of the crime and the demographic information about the victim and suspect. The dataset includes years 2006-2022 with 35 columns and 7.83 million rows.
Data Dictionary: https://data.cityofnewyork.us/api/views/qgea-i56i/files/ee823139-888e-4ad0-badf-e18e2674a9cb?download=true&filename=NYPD_Complaint_Historic_DataDictionary.xlsx

2. NYC Transit Subway Station Map | State of New York

This data file provides a variety of information on subway station entrances and exits which includes but is not limited to: Division, Line, Station Name, Longitude and Latitude coordinates of entrances/exits.

3. Alert Archive (mymtaalerts.com)

This webpage is an archive of alerts released by the MTA. It contains a date, agency, subject, and message.

Data Processing

NYC Transit Subway Station Map:

The list of Subway station entrances and exits had multiple locations for each station thus we reduced each station to one location and a single set of coordinates. Additionally we condensed the original 12 train line columns to one column for readability.

Retrieval of MTA Historic Alert Data:

We used a tool called Selenium to automate the web-scraping of the data. We input the url to start at, and after inspecting the webpage HTML, we can pull out the buttons and forms we need to manipulate. We filter out bus alerts, elevator alerts, and anything else that is not subway data. We select start and end dates, and then the website has been prepped for scraping with BeautifulSoup4. A maximum of fifty rows were displayed per page, which resulted in ~3000 pages of data that we wanted to scrape from 2017-2021, which is why automation was necessary. We parse through the HTML using BeautifulSoup4, and retrieve each row. After the table has been fully parsed, Selenium 'clicks' on the next page and repeats.

In our initial data analysis we incorporated 2017, however after further investigation we realized 2017 data was partially incomplete and it was therefore excluded. There were a few interesting finds within each dataset.

Retrieval of Real-Time/Recent Alerts Data Using Kafka:

A data factory was built to automatically webscrape MTA alerts. We noticed that the alerts are assigned a code chronologically using hexadecimal. Once we determine the starting code, the code is incremented by 1 within a for loop. The loop uses beautiful soup to parse through the data to pull out alert titles, date and time, alert message, and the alert agency. Since we were only interested in subway delays data, only the alerts with an agency of 'NYC' were kept. We were also not interested in elevator or escalator alerts, which were included in the mix, and filtered those out using regular expression. Each alert is then sent to Kafka as a message, serving as a producer.

It takes time for Kafka to receive and store the messages so a two and a half minute waiting time was added. We used a consumer to pull the messages out of Kafka and append the messages to a stored data frame of the previous messages. The data frame is then sorted so the alert codes are sorted in descending order. The entire data frame is then saved as a csv. This file is referenced in the data factory so that the next time the data factory runs, it will use the code on the first row to know which alert code it left off at.

Data Cleaning of Subway Alerts:

After the alerts data were collected, the alerts were filtered out by update status and delay status. We did not want to consider planned delays and we noticed that those that alert titles with the affected trains are unexpected delays, as opposed to planned service delays. We also did not want alerts categorized as updates because they are referring to a previous instance, not a new one.

Once these rows were removed, a list of the train lines was created and a new column was added to show which train line was affected. This could cause an alert to be represented multiple times since one alert can affect multiple train lines. In addition, another column was

added to show what borough is affected by the alert. Finally, a Latent Dirichlet Allocation model was used to categorize the types of alerts, completing our dataset. The LDA model is a form of topic modeling. We used this unsupervised learning model to sift through the message column and decipher the underlying categories. Through a mixture of the model and domain knowledge, we were able to create six message categories. Then, we had to assign each message a category. This was done by using two keyword dictionaries that assigned a message based on a keyword(s).

NYPD Historic Complaint Data:

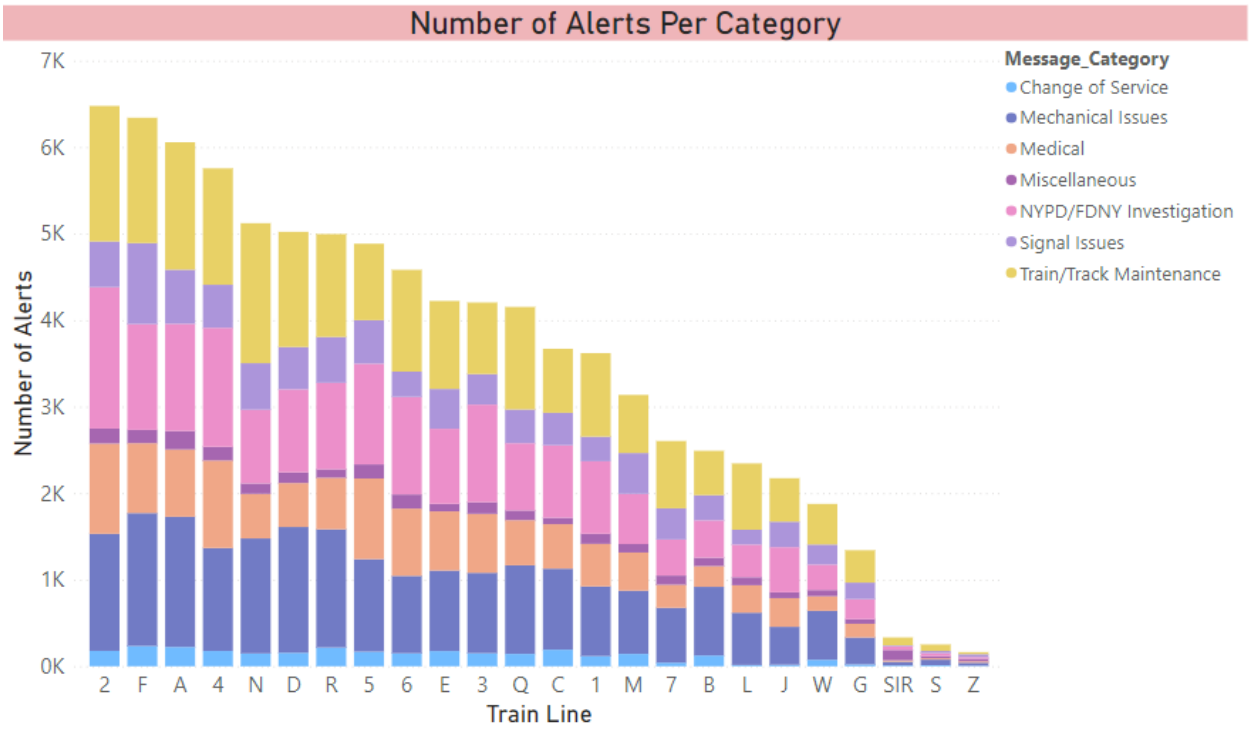
Our study focused on the years 2018-2021, reducing the row count to around 2 million. Some dates were entered incorrectly so we replaced the incorrect dates using pandas. Additionally we removed any rows lacking date of occurrence and location data. Due to the large size, the dataset had to be split up prior to processing and rejoined afterwards. In the process of uploading data, Brooklyn complaint data was deemed to be corrupted and was therefore removed from our analysis.

Joining Data Spatially and Temporally:

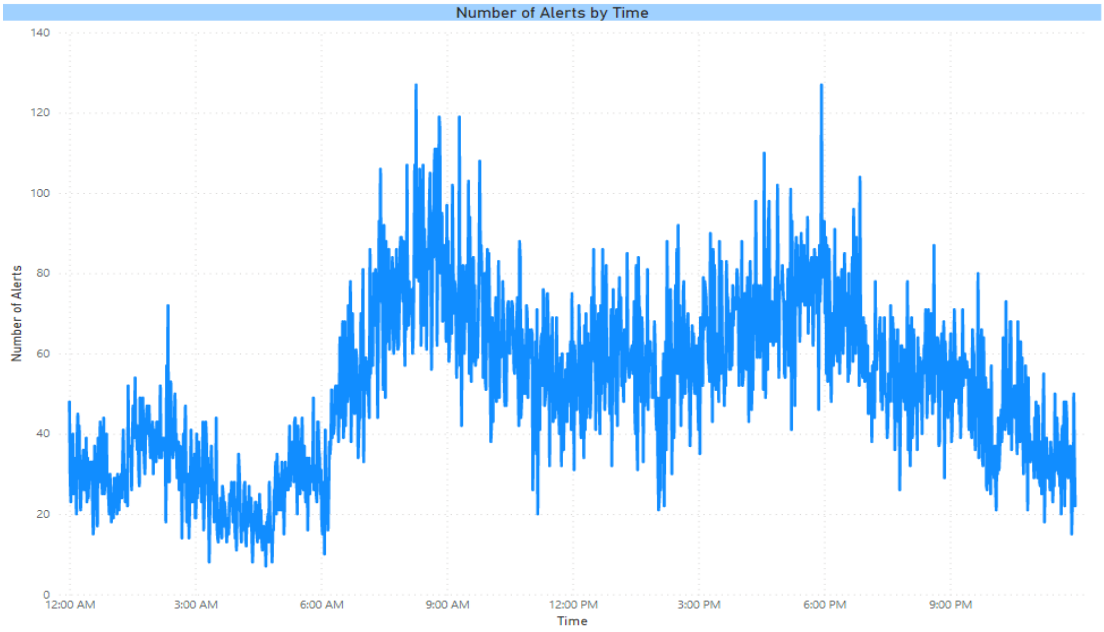
Firstly, we joined the complaints data with the subway station data by utilizing the coordinates provided in each dataset. For each complaint we added a column of the closest train station, the distance to that station and the train lines stopping at that station. To determine the distance between the complaint and the closest station we utilized the Haversine formula which accounts for the curvature of the earth. Each complaint is then linked temporally to delays. For each station that a delay affects, if a complaint is linked to that station within two hours after the delay, then that complaint is joined on the delay.

Data Analysis

Alerts/Delays Data:

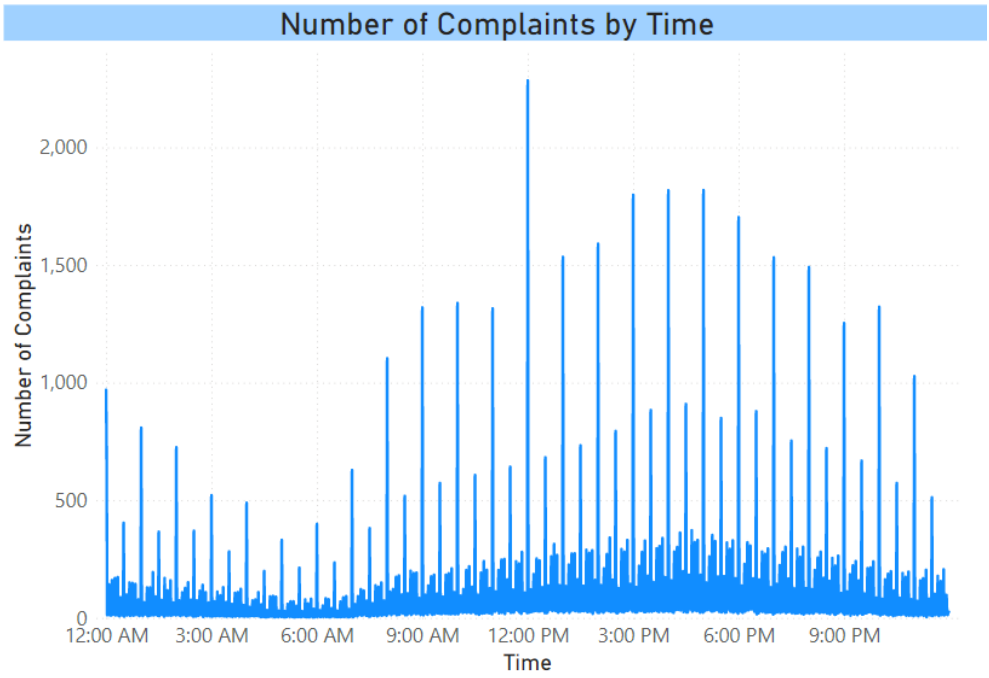


We've noticed that a significant number of alerts were due to MTA maintenance issues, which includes Mechanical Issues, Signal Issues and Train/Track Maintenance. There was a much smaller number of delays that were attributed to medical issues, such as a sick passenger. The number 2 Train appears to have the higher number of delays. This could be because the 2 Train starts deep in Brooklyn, through Manhattan and deep into the Bronx, covering 52 stations. On the contrary, the Z Train has the least amount of delays. This makes sense since the Z only goes from the bottom of Manhattan deep into Queens, covering only 21 stations.

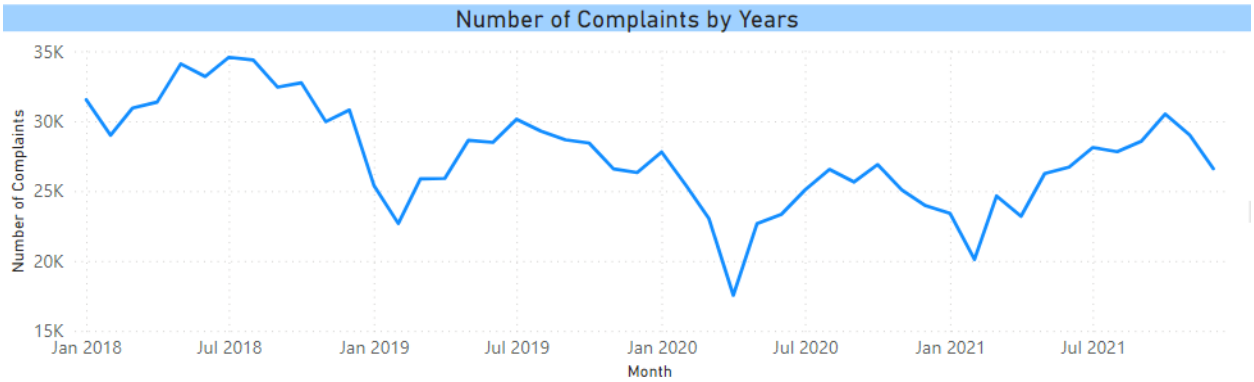


There are also two very noticeable peaks of the frequency of delays during rush hours, around 9AM and around 6PM, compared to the rest of the day. This might be due to more trains running during rush hours to accommodate the volume of riders.

Crime Data:



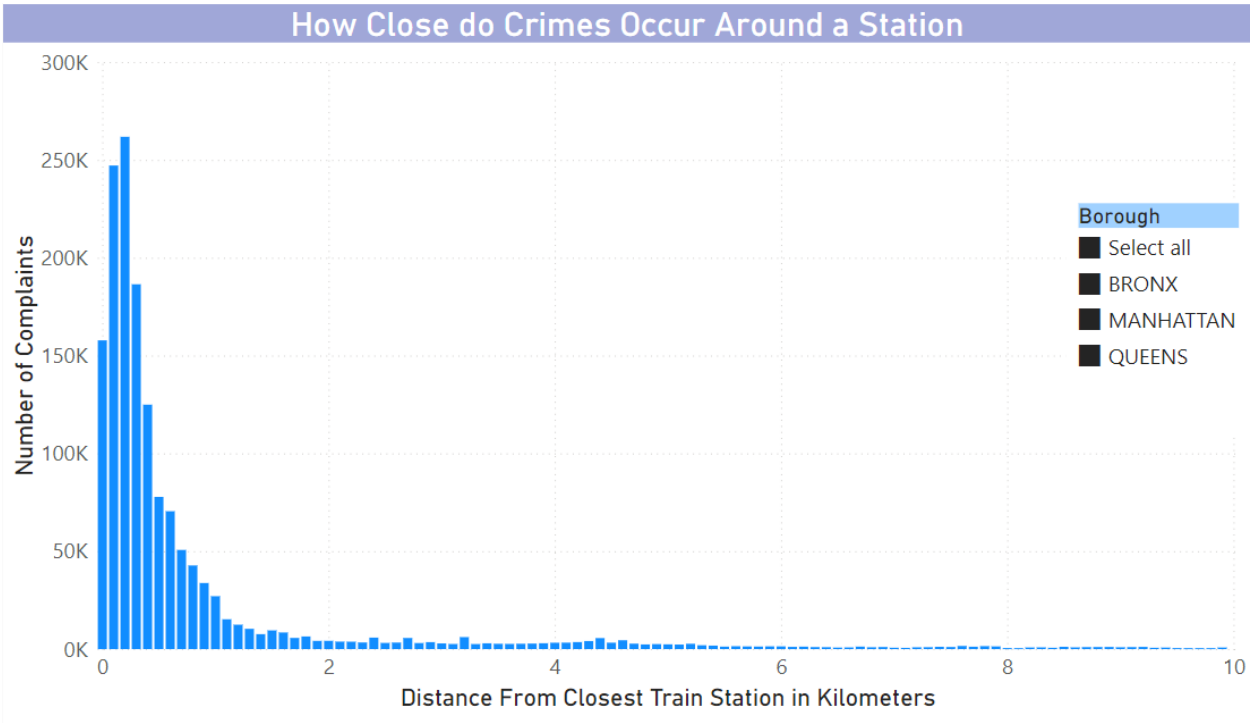
Looking at the frequency of reported crime throughout the times of a day, we noticed that most crime appears to occur around noon and on the hour. This is likely due to the estimation of time of occurrence either by the person reporting the crime, or the officer that took the report. We see a similar trend on a smaller scale on the half hours, likely for the same reasons.



The overall data is confirming and interesting. We see the dip during March and April of 2022 due to the pandemic. We can also see the seasonality of crime. The frequency of crime decreases as the temperature decreases and peaks around July. The most common crime is

petit larceny, followed by harassment. Of the four boroughs analyzed, Manhattan also has the highest number of crime, however, it also has the highest foot traffic and highest population density.

Spatial and Temporal Connections:



Within the three boroughs we have examined, the most frequent locations where crime occurred was 0.2 km - 0.3 km from a train station. The number of crimes could decrease as we get closer to a train station because the area covered is smaller. The number of crimes might increase as we move further than 0.3 km from a train station because the foot traffic and the density in those areas are lower.

Machine Learning

Our first model is based on a subset of the data, specifically the Parkchester subway station in the Bronx. This was chosen because it has a relatively high ridership, and we saw it as a typical example of ridership, delays, and crime. The model uses the combined delays and complaints data, where the complaints have Parkchester as the closest subway station, and the delays are on the subway lines that run into and out of Parkchester.

The delays and complaints are outer joined together, meaning that there are cases where no delay happens, but there is a complaint, and also cases where there is a delay, but no complaint. For the rest of the data, the complaints are joined to a delay if they occur in the two hours following the delay. In order to cut down on variance and see if our core idea of delays affecting crime has merit, we attempt to predict if a crime occurs based upon time of the delay, whether or not a delay occurs, and reason for the delay.

We used a logistic regression for classification. First, to see the effectiveness of our model, we looked at the prevalence of the majority class, which in this instance is a crime occurring. A crime occurs 61% of the time in our dataset. So, if the model were to guess that a crime occurs every single time, which is erroneous thinking, it would get an accuracy score of 61%. Our logistic regression model has an accuracy score of 73%, which is significant. To check the importance of time of day versus delays and delay categories, we removed all categories relating to delays, and retrained the classification model. This model has an accuracy score of 62%, which is a significant drop in accuracy. Thus, delays have some correlation with crime in the surrounding area, for this subset of data.

The second model we created is a random forest model that attempts to predict the type of crime being committed (i.e. harassment, robbery, etc). The model is only looking at 2018 data from the Bronx. The target variable is called 'OFNS_DESC'. Initially, the target variable had 53 values in it, however, a random forest model can't handle that many values to predict. We decided to cut down the amount of target values to five. Our five crime values are: Violence-Related Crimes, Property Crimes, Public Order Crimes, Sex Crimes, and Other Crimes.

Now, we will discuss some of the ETL for our model. Our model has 93 predictor variables. The only two variables that didn't need to be pre-processed are *Latitude* and *Longitude*. The column, *station_distance*, was calculated using the Haversine formula. The column, *alerts_in_range,* shows whether or not the closest subway station to the specific complaint currently had a delay as shown by an MTA alert. Each train line received its own column. The value "1" was given in the train line column if the closest station to the complaint had the particular line. Next, dummy variables were created for all the remaining categorical variables. Each station in the Bronx became its own column, and it got a value of "1" if the station was the closest to the complaint. We created dummy variables for *LAW_CAT_CD* which created three new predictors. Lastly, to pre-process the predictor variables, we cyclically encoded the dates. Cyclically encoding is the process of helping the model understand the cyclical nature of dates and time. For example, machine learning models don't understand that January and December are actually only one month apart. The model reads January as 1 and December as 12 and views that as 12 values apart. Cyclically encoding fixes that issue. To cyclically encode our dates, we created columns for each part of the datetime object. Then,

each of the new columns was cyclically encoded and then dropped. We were left with two columns for each part of the original datetime object. For example, we now had two cyclically encoded columns for months called, *month_sin* and *month_cos*.

Our model had 6 hyperparameters. These hyperparameters included: *n_estimators, max_depth, min_samples_split, n_jobs, max_samples, class_weights, and oob_score.* We used GridSearchCV (cross-validation) to find the optimal hyperparameters. However, we had issues with the cross validation taking too long to run, so we manually tested the hyperparameters to ultimately find the best combination. The last iteration of our model had these hyperparameters: *max_depth = 675, max_samples = 0.35, min_samples_split = 0.000000001, n_estimators=780, n_jobs=-1, oob_score=True, class_weight = {0:1, 1:28. 2:28. 3:28, 4:28}.*

After running the model, we could calculate the feature importances. Features importances score how important each predictor variable is to the model. We attempted to remove each variable that had an importance of zero. However, due to computational bottlenecks, we were not able to remove all the features with no importance. While those predictor variables don't impact the accuracy of the model, removing them would increase the computational speed of our model.

Our baseline score for accuracy was 36%. This was calculated by finding the proportion of the majority class. In other words, the most common target value made up 36% of the data. Any accuracy scores worse than 36% means that our model performed worse than just choosing the most common target value every time. Our accuracy score was 57.30%. The accuracy score was calculated by comparing the model's predictions, and the actual data. Our model was able to predict the correct crime category 57.30% of the time. Our accuracy score was 21 percentage points higher than the baseline, so we feel comfortable in saying that the model was performing well. Additionally, we calculated a validation accuracy score of 57.01% and an OOB (out-of-bag) score of 57.02%. Both validation accuracy and OOB test data that was not a part of the training data. Both scores are typically used together because they can reveal whether or not a model is overfitting. They reveal overfitting if their scores are significantly different from each other. Our two scores are only .01 away from each other, so that suggests that our model is not overfitting. In conclusion, our random forest model appears to be more accurate than the baseline, and it's not overfitting.

Conclusion

The goal of this investigation is to determine if there is a correlation between subway delays and crime. Because of the high variance of the data, it was a challenge to determine any kind of correlation. However, with the use of statistical modeling and machine-learning techniques we have found correlations between a delay occurring and a crime occurring in the area surrounding a subway station. We have also had relative success in predicting the type of crime that will occur based on the data, using a random forest model. Through data manipulation and graphing, we can see that the number of crimes compared to distance away from a subway station is heavily skewed to the right. Almost all crimes are committed within 0.5 miles of a station. Finally, our hypothesis was correct in assuming that the COVID-19 pandemic would affect both delays and complaints. There is a huge drop in both from February 2020 to April 2020.

After conducting our analysis we have determined potential steps to further our study. For Subway delay data we only utilized MTA alerts that resulted in delays. We think incorporating the time of the delays, using actual train times, could potentially impact our results. Additionally, finding a way to include population data could explain crime per capita instead of by overall borough. In the future, we would like to see how a neural network based deep learning model would perform in predicting crime, possibly with the addition of other data that may influence crime, such as pollution or weather. Additionally, we want to see how subway delays could be cut down or if maintenance can be scheduled more effectively, because they are not only inconvenient and appear to increase crime, but they cost businesses in NYC huge amounts of money.

Subway delays have far-reaching impacts. According to a report from the office of the NYC Comptroller, a ‘major delay’ of around five minutes costs businesses over $170 million dollars a year, and a delay going over twenty minutes costs businesses nearly $400 million dollars a year. According to the same report, the average wait time is 5 minutes. On the individual level, subway delays have harmful results. 74% of people have missed a business meeting due to subway delays. Additionally, 18% of people were reprimanded for being late and 13% of people had wages lost because of delays. In conclusion, delays cost businesses and individuals money that is unnecessarily lost. By decreasing the frequency of delays, we believe that crime will decrease, the economy will receive a boost, and people’s personal happiness will increase.

References

Stevens, T. (2017, October 1). *Comptroller Stringer: Subway Delays Hit City Economy, Cost Workers and Business Nearly $400 Million Each Year*. Comptroller.nyc.gov. Retrieved February 1, 2023, from https://comptroller.nyc.gov/newsroom/comptroller-stringer-subway-delays-hit-city-economy-cost-workers-and-business-nearly-400-million-each-year/

Stevens, T. (2017, October 1). *Comptroller Stringer: Subway Delays Hit City Economy, Cost Workers and Business Nearly $400 Million Each Year*. Comptroller.nyc.gov. Retrieved February 1, 2023, from https://comptroller.nyc.gov/newsroom/comptroller-stringer-subway-delays-hit-city-economy-cost-workers-and-business-nearly-400-million-each-year/