Final Paper: Analysis of Emergency 911 Calls

Authors: Hiro Fujii, George Jiang, Sindhuja Satheesh Kumar

1. Introduction

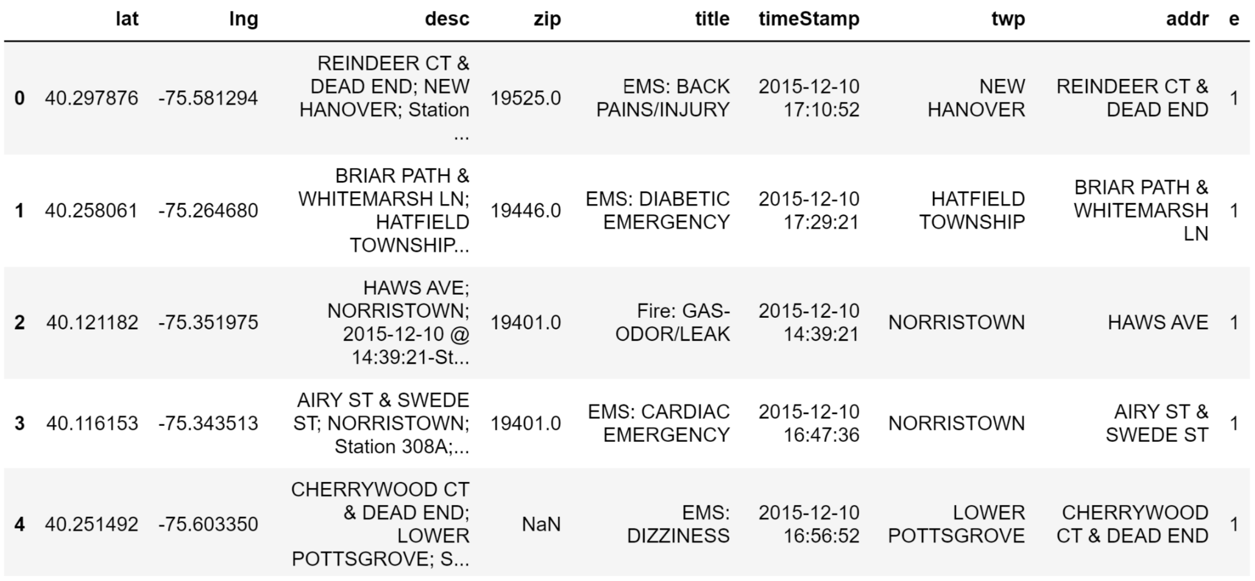
Within our 911 data set, we wanted to find patterns of 911 calls within the time of day. Since coming to Boston University and the Boston area, we have seen and heard sirens and ambulances multiple times throughout the day. It was very apparent when we would be walking around or even having the windows open at night. This led us to explore a 911 data set and the one we found happened to be the county from where one of our teammates was from. As such our two hypotheses were: 1. When are 911 calls more likely to occur? 2. What patterns can be drawn from the data overall? We were able to answer these questions through statistics and visualizations.

2. Methodology and Results

Kaggle is an online machine-learning repository that consists of over 50,000 public datasets. The *Emergency - 911 Calls* dataset by Mike Chirico was chosen from Kaggle as a primary source of data for this investigation. A .csv file includes the dataset which contains 663,522 rows of emergency calls made in Montgomery County, PA from the years 2015 to 2020. It includes 9 columns that consist of latitude (lat), longitude (lng), description of emergency (desc), ZIP code (zip), the title of emergency (title), date and time of call (timeStamp), township (twp), address (addr), and an index column (e).

2.1 Data Cleaning and Exploration

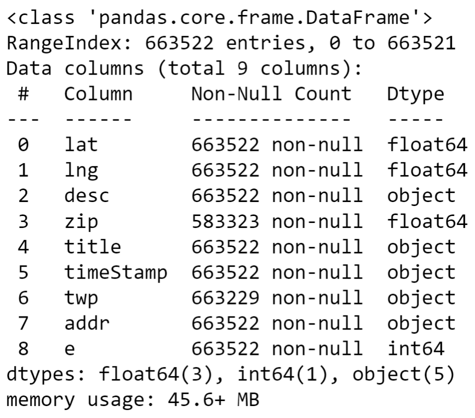
In order to get started with the investigation, the data needed to be read from the .csv file and converted into a suitable data structure so it can be explored, cleaned, and analyzed accordingly. The .csv file that contained the dataset was read into a Pandas DataFrame named *emerg* and the first 5 rows of data entry were displayed as shown in **Figure 1**.



	lat	lng	desc	zip	title	timeStamp	twp	addr	e
0	40.297876	-75.581294	REINDEER CT & DEAD END; NEW HANOVER; Station ...	19525.0	EMS: BACK PAINS/INJURY	2015-12-10 17:10:52	NEW HANOVER	REINDEER CT & DEAD END	1
1	40.258061	-75.264680	BRIAR PATH & WHITEMARSH LN; HATFIELD TOWNSHIP...	19446.0	EMS: DIABETIC EMERGENCY	2015-12-10 17:29:21	HATFIELD TOWNSHIP	BRIAR PATH & WHITEMARSH LN	1
2	40.121182	-75.351975	HAWS AVE; NORRISTOWN; 2015-12-10 @ 14:39:21-St...	19401.0	Fire: GAS-ODOR/LEAK	2015-12-10 14:39:21	NORRISTOWN	HAWS AVE	1
3	40.116153	-75.343513	AIRY ST & SWEDE ST; NORRISTOWN; Station 308A;...	19401.0	EMS: CARDIAC EMERGENCY	2015-12-10 16:47:36	NORRISTOWN	AIRY ST & SWEDE ST	1
4	40.251492	-75.603350	CHERRYWOOD CT & DEAD END; LOWER POTTSGROVE; S...	NaN	EMS: DIZZINESS	2015-12-10 16:56:52	LOWER POTTSGROVE	CHERRYWOOD CT & DEAD END	1

Figure 1. The first 5 rows of data from the emerg DataFrame

Next, to understand the data we will be working with, the size and basic information of the DataFrame were extracted through the .shape property and .info() method. It was discovered the DataFrame was of shape (663522,9). **Figure 2** shows the number of entries and columns, data types, and non-null count for each column.



```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 663522 entries, 0 to 663521
Data columns (total 9 columns):
 #   Column     Non-Null Count   Dtype
---  ------     --------------   -----
 0   lat        663522 non-null  float64
 1   lng        663522 non-null  float64
 2   desc       663522 non-null  object
 3   zip        583323 non-null  float64
 4   title      663522 non-null  object
 5   timeStamp  663522 non-null  object
 6   twp        663229 non-null  object
 7   addr       663522 non-null  object
 8   e          663522 non-null  int64
dtypes: float64(3), int64(1), object(5)
memory usage: 45.6+ MB
```

Figure 2. Information obtained from the .info( ) method

With this basic information to understand the data, the next step was to clean the data. Therefore, the DataFrame was checked for null values where it was discovered *zip* and *twp* contained 80,199 and 293 null values respectively. To further visualize the distribution of null values, a heatmap was generated as seen in **figure 3**. To fill these null values, KNN was used to find the missing zip codes and a 'NaN' was used to fill the null values for *twp*.

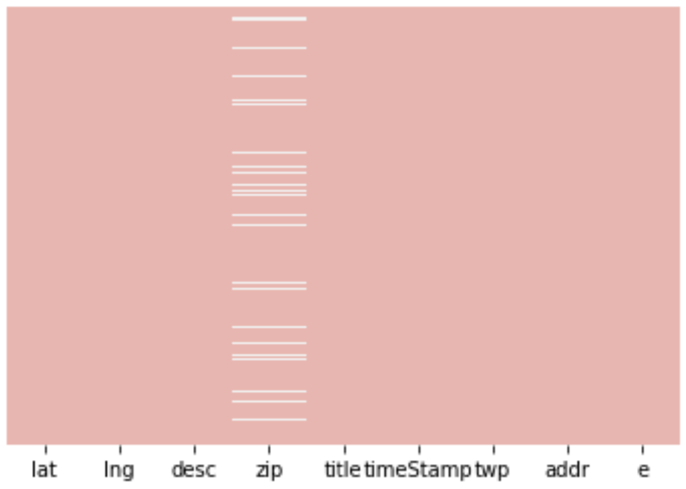


Figure 3. Heatmap to visualize null values

The KNN algorithm used was scikit learn's KNeighborsClassifier, and the features of the dataset used were a data point's latitude, longitude, and township name, which was converted to a unique integer using a hash function. The latitude and longitude coordinates were graphed, and outliers were determined, and cut from the dataset. These outliers were probably due to a misrecording of the call's latitude and longitude, but whatever the case, they were removed as they might have influenced other points in the KNN algorithm. Multiple iterations of the algorithm were trained and scored, with varying N_Neighbors against a 50/50 data train test split. As N_Neighbors increased, accuracy of classification decreased. However, all results were above 99% accuracy. However, the number chosen for N_Neighbors was 4 instead of 1, because we came to the conclusion that spread out points might be more influenced by large clusters of points if they were only given to one neighbor. Therefore, having multiple made the results more even, and 4 was a number within reasonable range of 1 and the max number of neighbors, 10. Thus, the algorithm was trained on the full dataset, with 4 N_Neighbors, and used to predict the zip codes for the data points missing them.

To perceive the frequency of reasons for the emergency calls found under the *title* column, a Word Cloud was chosen to visualize this due to its technique of representing text data in which the size of every word indicates its frequency or significance. As seen in **figure 4**, ‘EMS’, ‘Fire’, VEHICLE’, ‘PAINS’, and ‘DIABETIC’ are just a few crucial reasons for emergency calls.

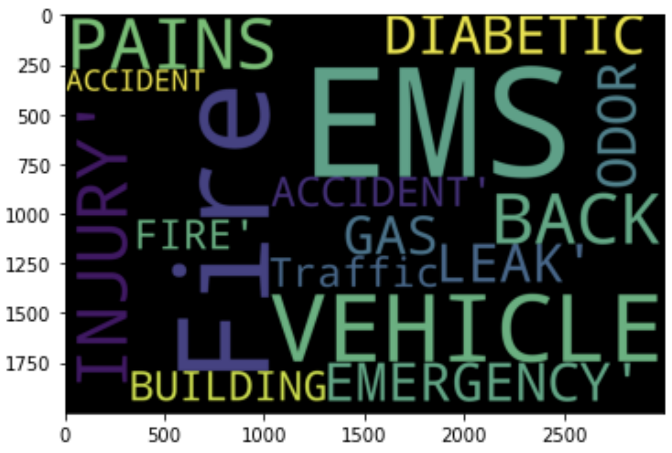
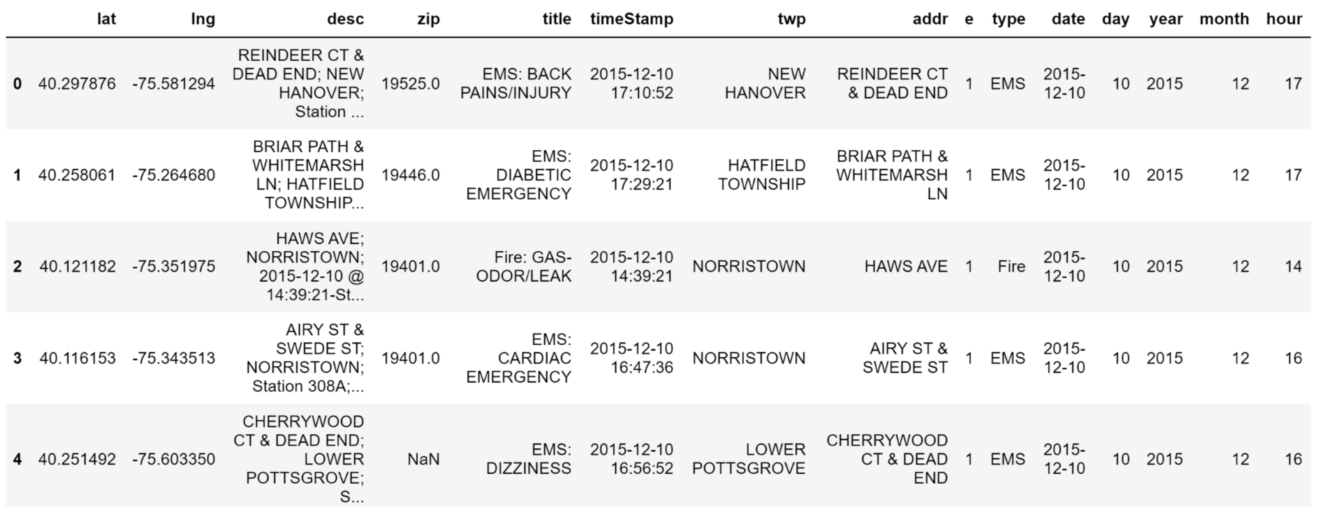


Figure 4. Word Cloud for title column

Following the visualization of the title column from *emerg* using a Word Cloud, the count of each type of emergency call seemed pertinent to explore. Hence, a value_counts( ) method was used to discover the count of all calls for each unique, specific title where it was found that Traffic: VEHICLE ACCIDENT had the most calls of 148,372, followed by Traffic: DISABLED VEHICLE with 47,909 calls and then Fire: FIRE ALARM with 38,336 calls. The broad category of the type of call being EMS was only the fourth most frequent where EMS: FALL VICTIM had 34,676 calls. In total, there were 148 specific types of reasons for an emergency call of which the broad categories were Traffic, Fire, and EMS. In order to simplify the cause for calls, a new column names *type* was created in the DataFrame that stored values of either Traffic, Fire, or EMS for each entry. When a value_counts( ) method was run on *type* an interesting fact was discovered which was EMS was the most frequent type of emergency call getting 332,692 calls, followed by Traffic with 230,208 calls, and Fire with 100,662 calls. This was the result although the most frequent type of specific emergency call was Traffic: VEHICLE ACCIDENT however, in the simple, generalized analysis, EMS is the most frequent type.

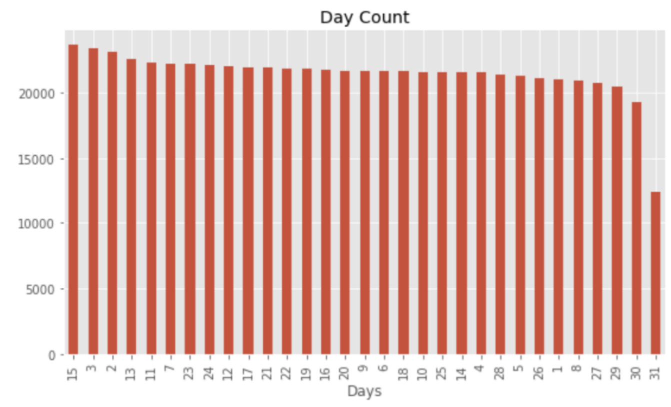
The last step in cleaning the data consisted of changing the *timeStamp* column values into the type DateTime using the to_datetime( ) method from the Pandas library. After this separate columns for date, day, year, month, and hour were created in the *emerg* DataFrame by splitting the date-time values as seen in **figure 5**. By manipulating the data this way, it will be easier to analyze the data later in the investigation.

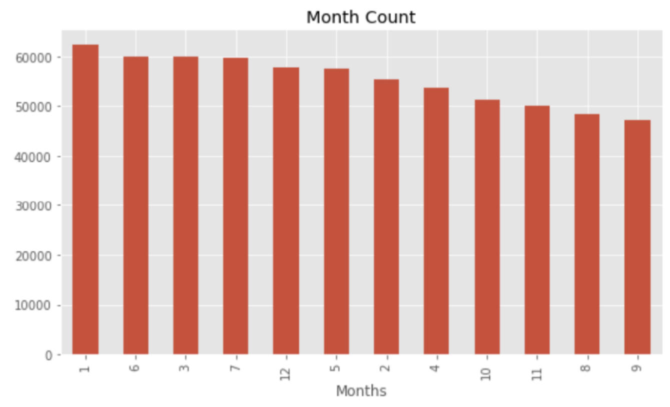


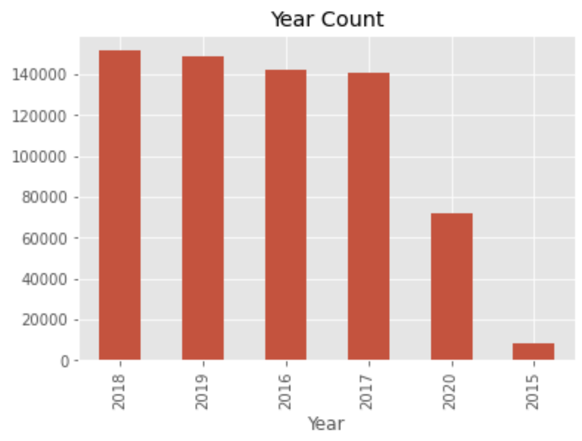
	lat	lng	desc	zip	title	timeStamp	twp	addr	e	type	date	day	year	month	hour
0	40.297876	-75.581294	REINDEER CT & DEAD END; NEW HANOVER; Station ...	19525.0	EMS: BACK PAINS/INJURY	2015-12-10 17:10:52	NEW HANOVER	REINDEER CT & DEAD END	1	EMS	2015-12-10	10	2015	12	17
1	40.258061	-75.264680	BRIAR PATH & WHITEMARSH LN; HATFIELD TOWNSHIP...	19446.0	EMS: DIABETIC EMERGENCY	2015-12-10 17:29:21	HATFIELD TOWNSHIP	BRIAR PATH & WHITEMARSH LN	1	EMS	2015-12-10	10	2015	12	17
2	40.121182	-75.351975	HAWS AVE; NORRISTOWN; 2015-12-10 @ 14:39:21-St...	19401.0	Fire: GAS-ODOR/LEAK	2015-12-10 14:39:21	NORRISTOWN	HAWS AVE	1	Fire	2015-12-10	10	2015	12	14
3	40.116153	-75.343513	AIRY ST & SWEDE ST; NORRISTOWN; Station 308A;...	19401.0	EMS: CARDIAC EMERGENCY	2015-12-10 16:47:36	NORRISTOWN	AIRY ST & SWEDE ST	1	EMS	2015-12-10	10	2015	12	16
4	40.251492	-75.603350	CHERRYWOOD CT & DEAD END; LOWER POTTSGROVE; S...	NaN	EMS: DIZZINESS	2015-12-10 16:56:52	LOWER POTTSGROVE	CHERRYWOOD CT & DEAD END	1	EMS	2015-12-10	10	2015	12	16

Figure 5. emerg DataFrame with the new date, day, year, month, and hour columns

In order to start data exploration, bar charts were used to highlight the data distribution. Three bar charts were plotted to observe the distribution of emergency calls for each day, month, and year using the matplotlib.pyplot library. Each bar graph shown below displays the count of calls for each day, month, and year. From this, it is possible to determine the most frequent number of calls occur on day 15, month 1, and year 2018 all independent of each other as it can not be deduced that the most number of calls fall in the combination of day 15 of month 1 in the year 2018.







To further understand the distribution of calls from another perspective, a faceted histogram was plotted from the *emerg* DataFrame. The faceted histogram exhibited the count of all calls for each township based on the year and type of emergency call (ex: EMS, Fire, or Traffic). Here the x values were *twp* to represent the townships, column values were *type* to represent the type of emergency call, and row values were *year* to represent the distribution of calls for all townships in each year. The resulting faceted histogram is seen in **figure 6** where Norristown had the most frequent EMS calls across all years and Lower Merion had the most frequent Fire and Traffic calls across all years. It can also be deduced that there is a denser concentration of calls from the years 2016 to 2019 where it gradually decreases in intensity in 2020.

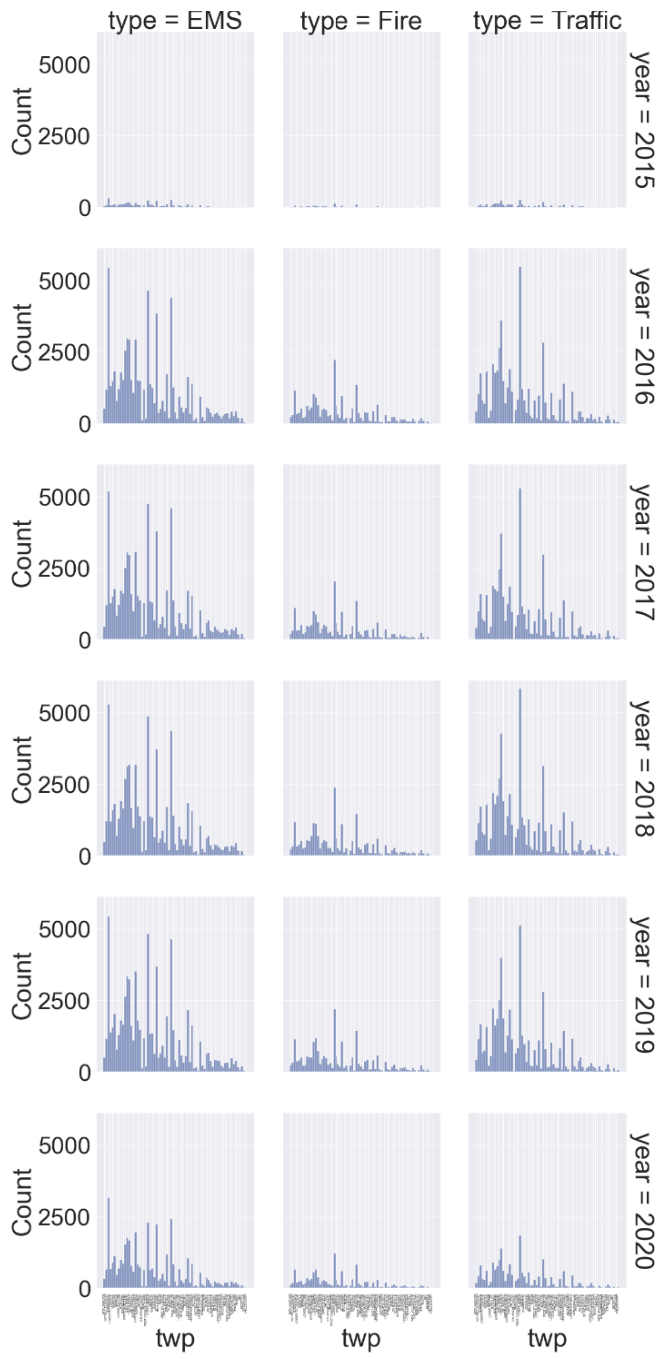


Figure 6. Faceted histogram.

2.2 Z-test and Chi-squared Test

To see whether the proportion of EMS calls from zip codes 19401 and 19403 were the same in the morning (12am-11am), afternoon (12pm-5pm), and evening (6pm-12am), a two proportion z test was done on the data. The test focused on two specific zip codes instead of the entire data set because the data set consisted of over 500,000 entries. By choosing to use the entire data set, the p-value would be significantly deflated due to the large sample size. 19401 and 19403 were chosen as the zip codes because they were closest to each other out of the entire data set.

Finding the overall proportion of EMS calls, while not taking into account the different times to create a baseline would allow an easier comparison to the different times of day. As such, the null hypothesis was "the proportion of EMS 911 calls from zip codes 19401 and 19403 are the same" and the alternative was "the proportion of EMS 911 calls from zip codes 19401 and 19403 are not the same." The z-test was conducted and got "z_stat: 8.044, p_value: 0". With a significance level of 0.05, the null hypothesis is rejected and the alternative hypothesis is accepted, meaning the proportion of EMS between zip codes 19401 and 19403 was very different. The actual difference ended up being 2.7%.

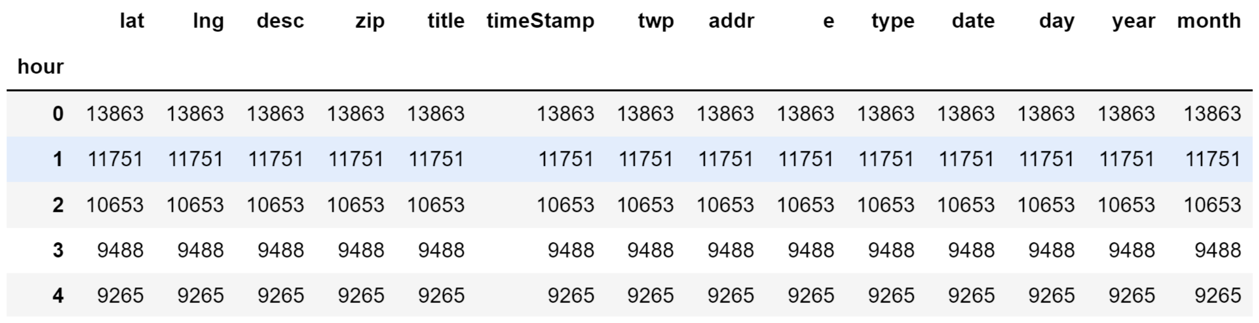
Now the time of day factor comes into play. Essentially, repeat the z-test, but take into consideration time. The null hypothesis was the same except it took into consideration morning, afternoon, and evening: "the proportion of EMS 911 calls from zip codes 19401 and 19403 in the [morning, afternoon, evening] are the same." After adding the parameter time, the result was "z_stat: 4.271, p_value: 0.000019482", indicating that the proportions were different. There was an actual difference of 2.2%. For the afternoon, the result was "z_stat: 4.347, p_value: 0.000013801", which meant that the proportions were still different. The actual difference was 2.4%. Finally, for the evening, the result was "z_stat: 4.738, p_value: 0.000002161". The actual difference was 3%. When you compare the actual difference, the proportion in the evening had the greatest difference. Even though the actual difference in the evening is larger than the actual difference overall, the p-value for the proportion overall was smaller due to the larger sample size.

Overall, the p-values were essentially 0. This led to the conclusion that there was a difference in the proportion of EMS calls between the two zip codes 19401 and 19403 despite the two zip codes being so close to one another. There are possibly other factors that were not considered such as geography or resource availability. However, the difference was only within a margin of ~3% and this still meant that the majority of EMS calls were relatively the same. The p-value couldn't help as much because the difference was larger than 1%.

For the chi-square test, I wanted to check whether the proportion of EMS calls were evenly distributed in the same 3 time periods (morning, afternoon, evening) across the entire data set. This would give me a better understanding of the data I was working with. The observed values I ended up getting were (79041, 152998, 100437, respectively with time of day). Clearly, the EMS values were not evenly distributed, with the afternoon receiving the most amount of calls. Since these numbers were so large and different, it made sense that the p-value I got was 0. It wasn't surprising to see that the afternoon had the most calls since that is the middle of the day when most people are outside, making it more likely for some sort of emergency to occur. It also made sense that evening had the second most EMS calls since people are likely to be at an outside event or tired in general. Morning saw the least amount of EMS calls most likely due to the fact that people have energy and their day is just getting started.

2.3 Regression Modeling

To answer the question of how the number of calls and the hour of the day is correlated and how values for the number of calls at a specific hour of the day can be interpolated over the span of 6 years, a regression model must be created with the data extracted from the *emerg* DataFrame. In order to execute this, first understanding the nature of the call counts for each day is necessary hence the value_counts( ) method is applied to observe the distribution of the top 5 days with the most call counts. Here it was found that day 17 had the highest with 44,119 calls. Next, a group DataFrame *emerg_hour* is created using the .groupby( ) method on the *emerg* DataFrame. The new DataFrame *emerg_hour* represents all the entities grouped together by the hour value that stores the count for each column entity. **Figure 7** showcases the outcome of the *emerg_hour* DataFrame.



	lat	lng	desc	zip	title	timeStamp	twp	addr	e	type	date	day	year	month
hour														
0	13863	13863	13863	13863	13863	13863	13863	13863	13863	13863	13863	13863	13863	13863
1	11751	11751	11751	11751	11751	11751	11751	11751	11751	11751	11751	11751	11751	11751
2	10653	10653	10653	10653	10653	10653	10653	10653	10653	10653	10653	10653	10653	10653
3	9488	9488	9488	9488	9488	9488	9488	9488	9488	9488	9488	9488	9488	9488
4	9265	9265	9265	9265	9265	9265	9265	9265	9265	9265	9265	9265	9265	9265

Figure 7. emerg_hour DataFrame

Next, a line graph was plotted between the *hour* and the *twp* which represents the number of townships that made an emergency call for each hour of the day over the span of 6 years. **Figure 8** presents the line graph showing that hour 17 has the highest number of calls.

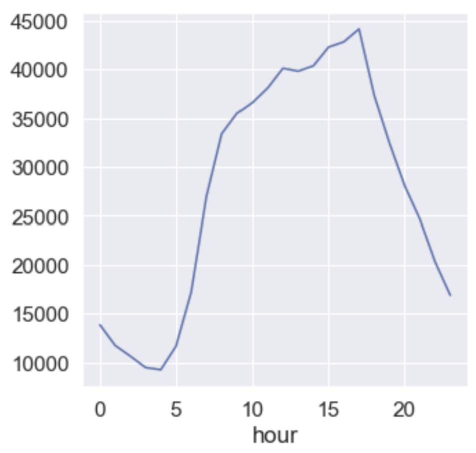


Figure 8. Call count vs Hour line graph

From the line graph, it can be deduced that in order to fit a regression model on this data, linear regression is not suitable due to the polynomial nature of the curve. Hence, a lowess regression model was first attempted to fit this data due to its generalization of moving mean and polynomial regression. Moreover, its nonparametric technique seemed beneficial to this case as we do not know how well these variables correlate with each other and if they depend on parameters as the shape of the curve is not a standard one. Therefore, the lowess regression was executed using the .regplot( ) method from Seaborn where its parameters were (x='hour', y='twp', data=emerg_hour.reset_index(), lowess=True, ci=None). The resulting lowess regression model can be seen in **figure 9**.

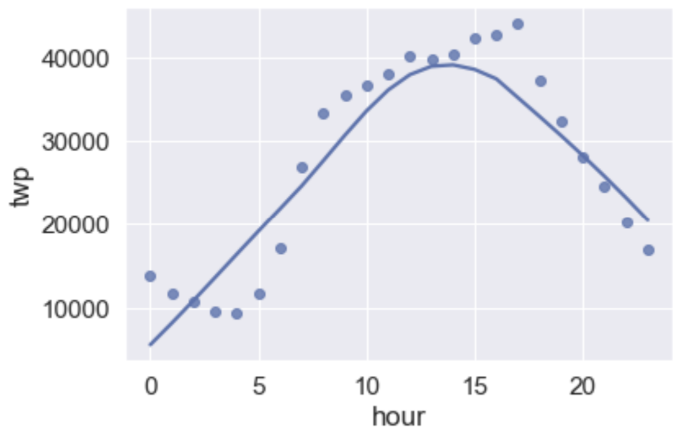


Figure 9. Lowess regression model

However, it can be observed that this was not a great data fit from the hours 0-7. Therefore, a polynomial regression, which is parametric, was then applied to test whether the need for a finite number of parameters was necessary to produce a great fit. The numpy.poly1d( ) class takes the argument .polyfit( ) which is a method that calculates the least squares polynomial fit. The .polyfit( ) method takes the x, y, and degree of the polynomial as arguments hence by varying the degree, we can get a better regression model. However, by varying the degree and getting too close to the original data points, we might be overfitting the model and begin to start to represent the noise in the data too. In order to evaluate the regression model genuinely, the R-Squared score was used to get a value between 0-1 where 0 meant no relationship and 1 meant the perfect relationship by being 100% related. This was done using the r2_score( ) function from the sklearn.metric library that takes the true y values and predicted y values as arguments.

To start building the model, a numpy.poly1d object is created and named *polyreg* which takes the .polyfit( ) method as an argument. The .polyfit( ) method takes the true x and y values along with the degree of the polynomial. **Figure 10** shows the first model which was set to have a degree of 3 for which the R^2 score was 0.916. The second model was set to have a degree of 5 which produced a 0.966 R^2 score. The third model was set to have a degree of 9 which produced a 0.992 R^2 score. From the results of the R^2 score, we can see that the third model is undesirable as it overfits the data points by having an extremely high score and almost connecting all the data points in the scatter plot that does not generalize the model. The second model is the most suitable as it has a moderately high R^2 score which is greater than the first model's score too.

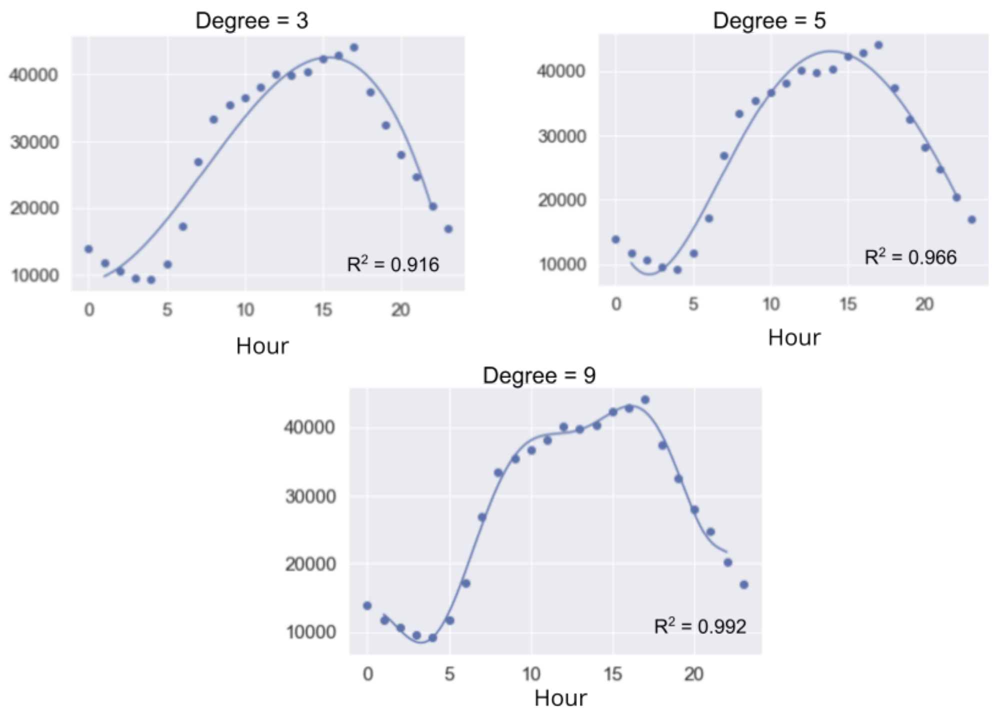
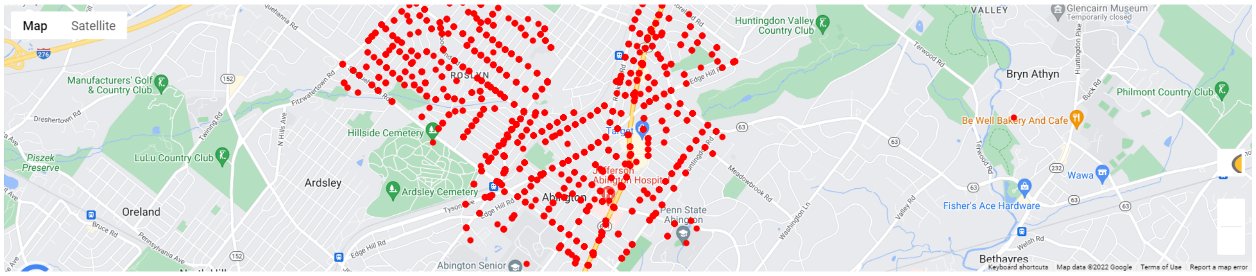


Figure 10. Three polynomial regression models with varying degrees

2.4 Geolocation

In order to display some of the points on a map, the Google Maps API was used. Using an API key and the gmaps library, a figure was generated in which the latitude and longitude points of the data set were fed to it, and a map with the points displayed was outputted.



4. Conclusion

Through our use of statistics, we were able to conclude that it was likely for 911 calls to occur in the evening. In our z-test and chi-square test, we found most calls to occur around the afternoon/evening time of the day. Even though the z-test resulted in the p-value essentially being zero from the two zip codes overall, the p-value from time of day confirmed that there was a difference in calls between the two. In our regression analysis, we found that hour 17 had the most calls, which was similar to what was found from the chi-square test, where afternoon and evening had the most calls. With our visualizations, we showed these results through word cloud, barcharts, faceted histograms, and regression models. Our Google Maps incorporated the latitude and longitude coordinates to visualize where the 911 calls were coming from. From here, we could identify places where 911 calls were a common place to occur.