Author: **Arianna Lang Wang**

Global Terrorism Project Report

March 16th, 2018

DATA OVERVIEW

The Global Terrorism Database (GTD) is an open-source database including information on terrorist attacks around the world from 1970 through 2016. The GTD includes systematic data on domestic as well as international terrorist incidents that have occurred during this time period and includes more than 170,000 cases. For each GTD incident, information is available on the date and location of the incident, the weapons used and nature of the target, the number of casualties, etc. There is also a significant amount of missing information. For example, there are 4,606 missing longitude values and 4,606 missing latitude values. This raw data is available on the Kaggle website: https://www.kaggle.com/START-UMD/gtd/data.

You can access my python code via my github: https://github.com/ariannalangwang/Capstone-Project-Global-Terrorism/blob/master/Global%20Terrorism%20Project%20Python%20Code.ipynb

CLIENT AND PROPOSED APPROACH

My “imaginary” client is The Central Intelligence Agency (CIA), which is interested in understanding more about terrorist attacks that had happened around the world and in the United States.

There are many questions I can ask about this data set. For instance, is the number of terrorist attacks increasing or decreasing throughout the years? Which regions are the safest to live in, i.e. have the least number of terrorist attacks? What type of terrorism attack is most common in each region? Conversely, can I predict which region will have a particular type of terrorism next? Additionally, I can focus my analyses on terrorism in the United States. For instance, I can map out the number of terrorist attacks by state from 1970 to 2016. My overall approach is to use various Python libraries to clean this

‘messy’ dataset, make beautiful graphs and charts, and do useful statistical analyses so that the agency can visualize and make sense of the data.

DATA CLEANING

I downloaded my raw data as a CSV file from the Kaggle website https://www.kaggle.com/START-UMD/gtd/data. Initially, I tried to import the data with the Pandas package as df = pd.read_csv('global_terrorism_db_0617dist.csv') alone, but I received an error message “UnicodeDecodeError: 'utf-8' codec can't decode ...” After searching on Stack Overflow and trying a couple of different encoding methods, df = pd.read_csv('global_terrorism_db_0617dist.csv', encoding="ISO-8859-1") allowed me to import the data successfully.

The data frame has 170,351 rows and 135 columns. After examining all 135 columns, I decided to keep 22 columns for later analyses. The first three columns were ‘year’, ‘month’ and ‘day’ respectively. Each of the ‘month’ and ‘day’ columns has one missing value. After examining the missing value, I dropped both rows. There are 20 zeros in the ‘month’ column and 891 zeros in the ‘day’ column. Zeros are meaningless for these two columns since you cannot have month ‘zero’ or day ‘zero’; thus, they are considered ‘missing values’. After some careful consideration, I decided to impute a random number from 1 to 12 for the missing month and a random number from 1 to 28 for the missing day. After that, I converted the type of values in all three columns from ‘string’ to ‘int’. Finally, I used datetime.strptime to convert these three columns into a DateTime object and set it as a new column called ‘date’.

I further cleaned the data by converting some columns’ data types from float to int since these columns are ID’s and thus do not need the decimal points. I also converted the 'success' and 'suicide' columns to categorical data type since both have binary data. Furthermore, I renamed some column names and dropped duplicate data by using Pandas’ drop_duplicates() method. At this point, there are 158,285 rows left in the data frame.

Another crucial step during data cleaning is to check for outliers. Columns 'N_killed' and 'n_wound’ are the only two columns in the data frame that have meaningful numeric data. After using the describe() method, I realized that both have very large maximum values, which is an indication for potential outliers. I also drew a boxplot to visualize both columns. The boxplot looked strange -- individual points seem to stack on top of

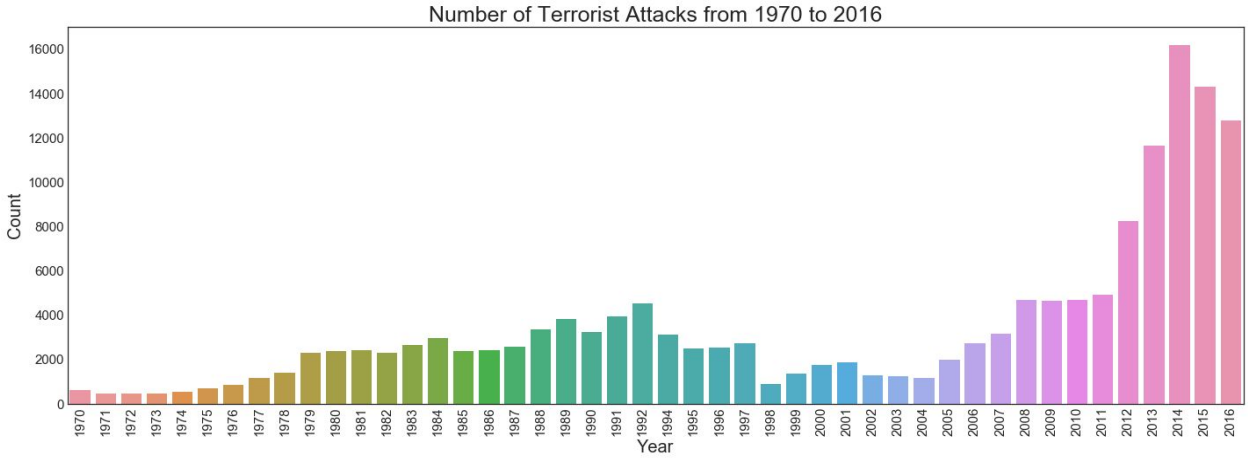
each other rather than forming a box. This alerted me that there might be a lot of potential outliers.

One way to access outliers is to use the IQR (interquartile range) method. We first compute IQR = Q3 - Q1. Then lower fence = Q1 - 1.5 * IQR and higher fence = Q3 + 1.5 * IQR. Any values outside the fences are considered outliers. For the 'n_killed' column, the fence is (-3.0, 5.0). The column does not have negative numbers, but any number greater than 5 would be deemed as an outlier according to the IQR method. There were a long list of values that were greater than 5. This result also explained why the boxplot looked so strange.

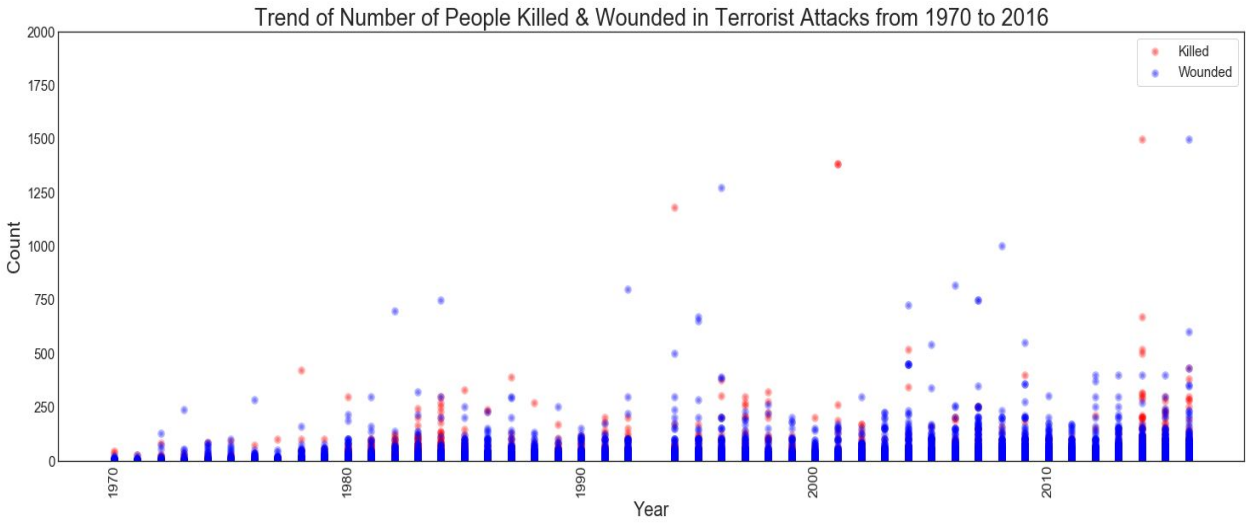
Clearly, using the IQR method to detect outliers would not be practical for this case, so I had to make a judgement call myself. After examining the boxplot again, I decided to set numbers greater than 1000 as outliers for both columns. By this new rule, I had 4 outliers for the 'n_killed' column and 7 outliers for the 'n_wound' column. I made a new dataframe called df_noout without the outlier rows.

INITIAL FINDINGS

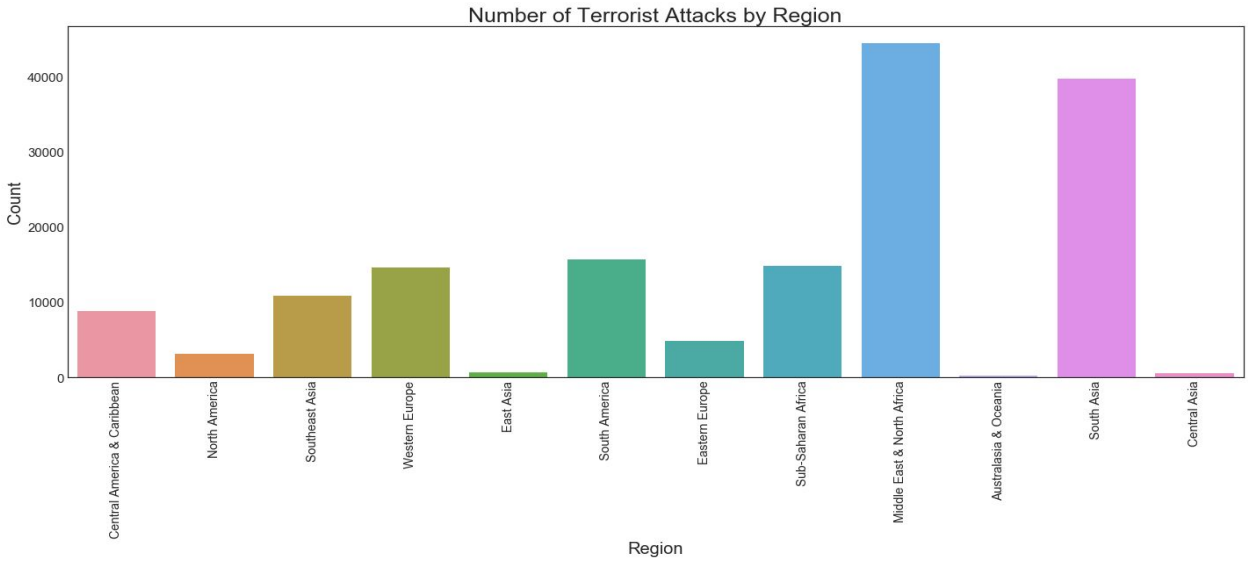
The first important question I asked was: "Has the world become a safer place today?" The answer seems to be "No". The number of global terrorist attacks has been increasing and it has increased drastically since 2012.



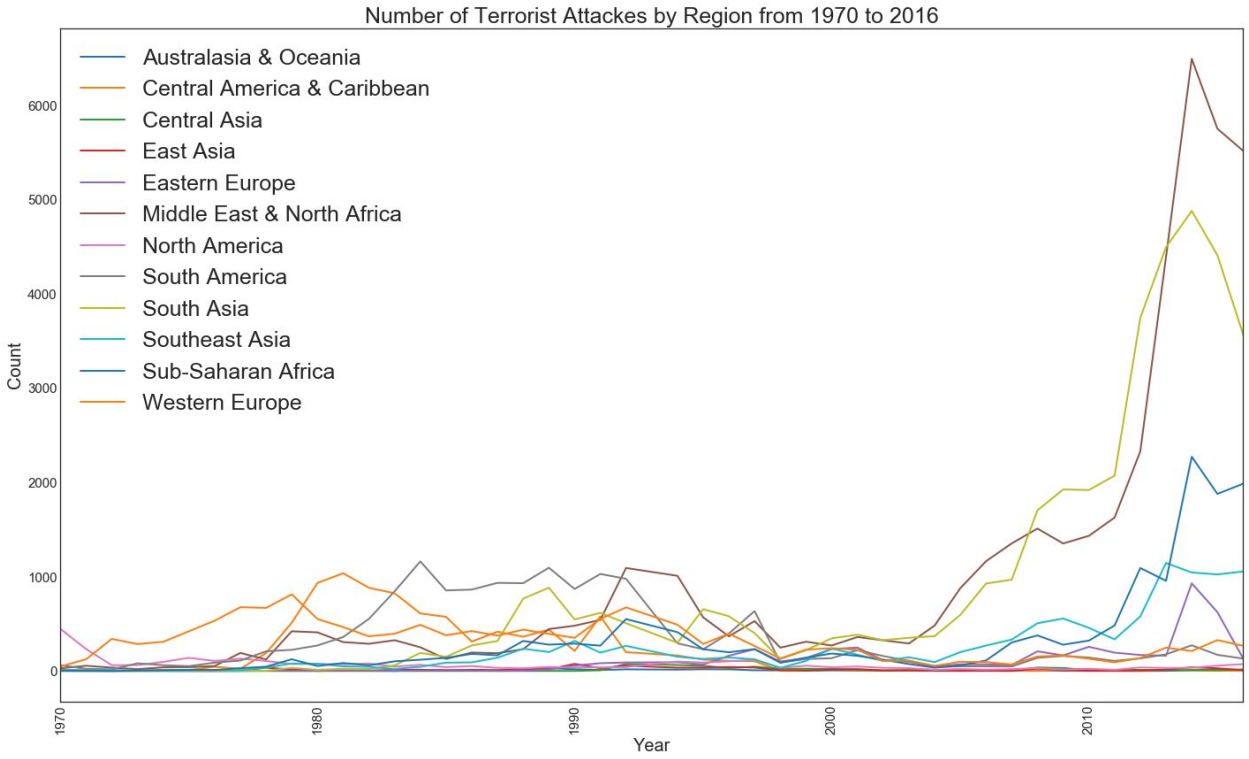
Not surprisingly, the numbers of people killed or wounded in these attacks are also increasing over the decades.



Now that we've seen a time trend, my second question was: "Is there a geographical trend?"

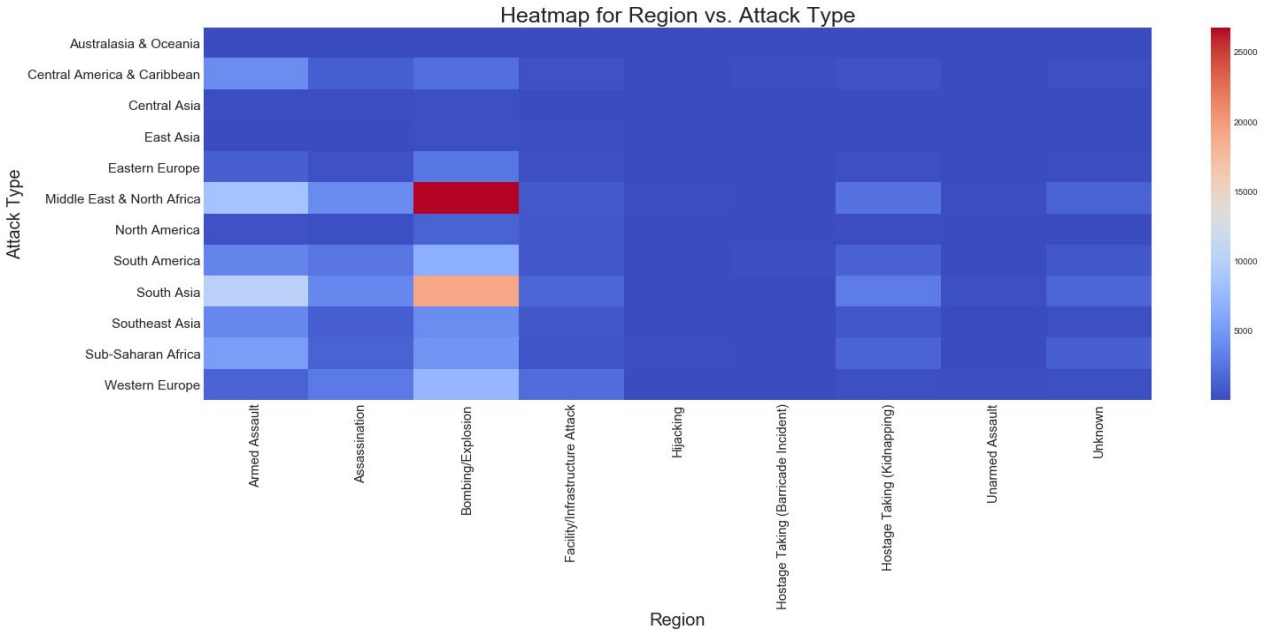


The answer is "Yes". We see that regions such as Middle East/North Africa and South Asia are most vulnerable to terrorist attacks. East Asia, Central Asia and North America are relatively safe regions. However, tying to the time-trend discussed before, a lot of regions suffer from increasing terrorist attack occurrences in the recent decade.



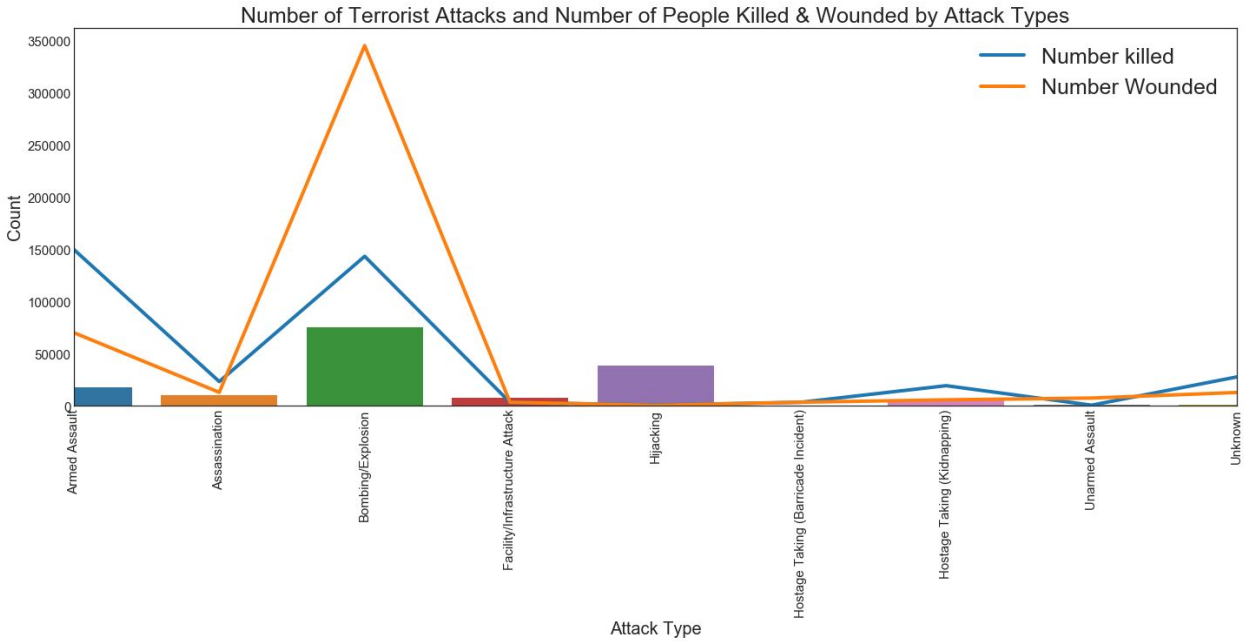
The third question I wanted to ask was around terrorist attack types.

We see previously that Middle East/North Africa and South Asia are the most vulnerable regions for terrorist attacks. “Are these regions prone to a particular type of terrorist attack?”



The answer is “yes”. We see from the heatmap above that Middle East/North Africa and South Asia have the highest number of bombing/explosion type of terrorist attacks.

Bombing/explosion is also the most used and the most lethal means by terrorists since it kills and wounds the most people.

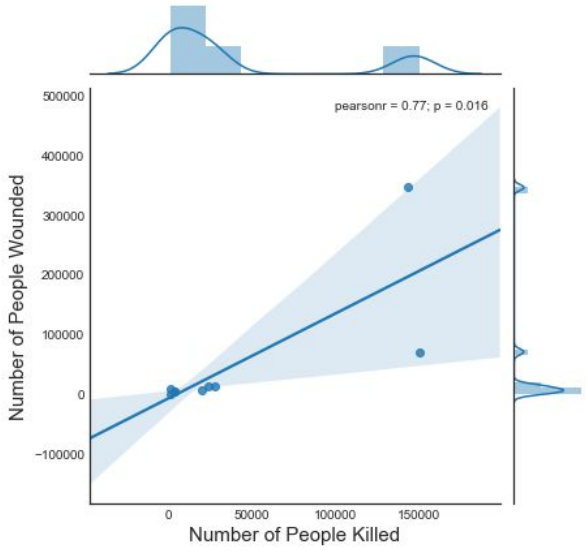


From the graph above, there seems to be a correlation between number of people killed and number of people wounded by attack types. To quantify my observation, I set up a hypothesis test to test this correlation.

Ho: under different attack types, number of people killed and number of people wounded are independent of each other.

Ha: under different attack types, number of people killed and number of people wounded are not independent of each other.

After running the test, the Pearson-correlation statistic, r, is 0.77 with the p-value of 0.016. Since the p-value is less than 0.05, we reject the null hypothesis and conclude that under different attack types, number of people killed and number of people wounded are not independent of each other. We can also say that there is a strong positive correlation between number of people killed and number of people wounded because the correlation coefficient r is 0.77.



From the “Number of Terrorist Attacks and Number of People Killed & Wounded by Attack Types” graph above, it is also easy to see that certain attack types kill more people than other types. For instance, Bombing/explosion incidences kill the most people compared to other attack types. However, Bombing/explosion is also the most frequent attack type used by terrorists. Doing a one-way ANOVA test allows us to compare the average number of people killed for each attack type on a per incident basis.

Ho: per incident, the average number of people killed for each attack type are all equal.

Ha: per incident, the average number of people killed for each attack type are not all equal.

After running the ANOVA test, the F-statistic is 416.76 and the p-value is zero. Hence, we reject the null hypothesis and conclude that per incident, the average number of people killed for each attack type are not all equal.

However, we still do not know which one(s) are different from the others. This is why ANOVA test is often followed by a post hoc analysis. Tukey's range test, named after the American mathematician John Tukey, is a common method used as post hoc analysis after one-way ANOVA. This test compares all possible pairs and we can use it to precisely identify pairs where the difference between two means is greater than the expected standard error.

For each pair of mean values:

Ho: the means are equal.

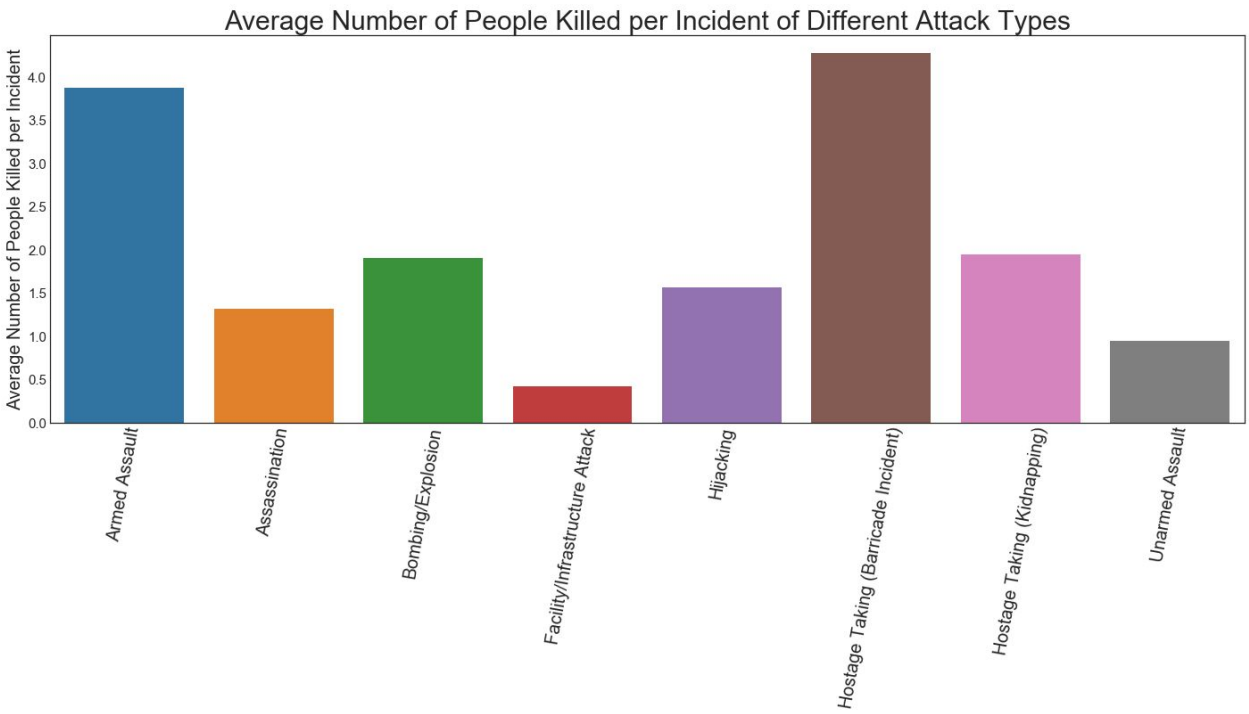
Ha: the means are not equal.

The result below shows each pairs' mean difference. If the pair's mean values are statistical-significantly different, then we reject the null hypothesis and conclude that that pairs' mean values are not equal. (In the table below, the 'reject' column will have a true value.)

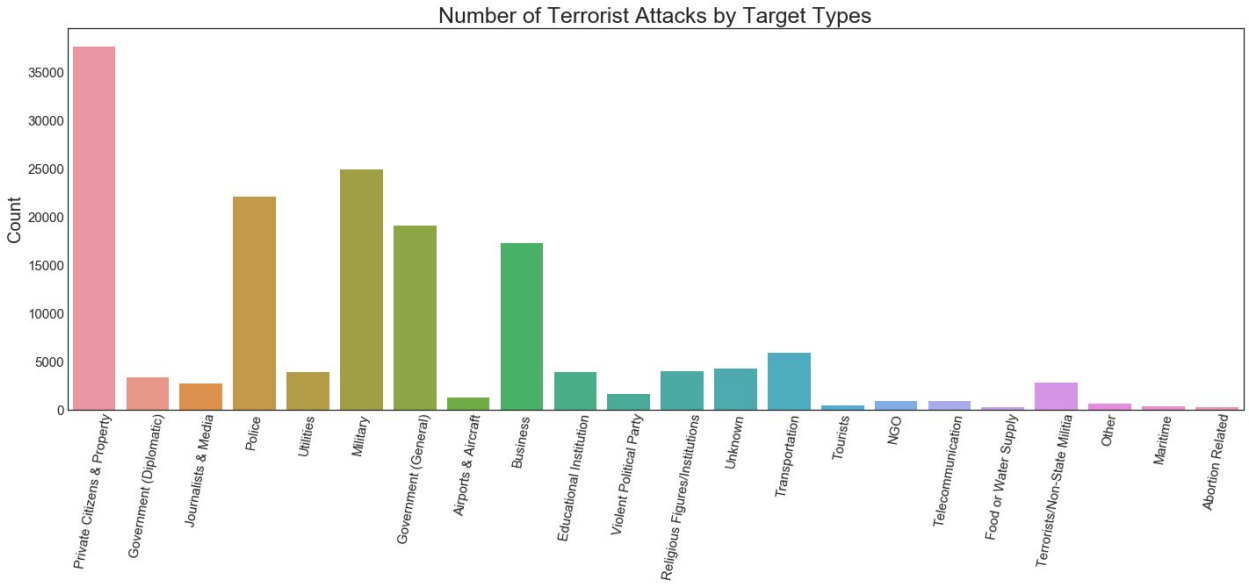
Multiple Comparison of Means - Tukey HSD,FWER=0.05

group1	group2	meandiff	lower	upper	reject
Armed Assault	Assassination	-2.563	-2.8046	-2.3214	True
Armed Assault	Bombing/Explosion	-1.9753	-2.1421	-1.8085	True
Armed Assault	Facility/Infrastructure Attack	-3.4607	-3.7851	-3.1363	True
Armed Assault	Hijacking	-2.3124	-3.4205	-1.2044	True
Armed Assault	Hostage Taking (Barricade Incident)	0.3962	-0.5409	1.3334	False
Armed Assault	Hostage Taking (Kidnapping)	-1.9285	-2.2279	-1.629	True
Armed Assault	Unarmed Assault	-2.9285	-3.8712	-1.9857	True
Assassination	Bombing/Explosion	0.5877	0.3654	0.81	True
Assassination	Facility/Infrastructure Attack	-0.8977	-1.2538	-0.5416	True
Assassination	Hijacking	0.2506	-0.8672	1.3683	False
Assassination	Hostage Taking (Barricade Incident)	2.9592	2.0106	3.9078	True
Assassination	Hostage Taking (Kidnapping)	0.6345	0.301	0.9681	True
Assassination	Unarmed Assault	-0.3655	-1.3196	0.5887	False
Bombing/Explosion	Facility/Infrastructure Attack	-1.4854	-1.7956	-1.1751	True
Bombing/Explosion	Hijacking	-0.3371	-1.4411	0.7668	False
Bombing/Explosion	Hostage Taking (Barricade Incident)	2.3715	1.4392	3.3039	True
Bombing/Explosion	Hostage Taking (Kidnapping)	0.0468	-0.2372	0.3309	False
Bombing/Explosion	Unarmed Assault	-0.9532	-1.8911	-0.0152	True
Facility/Infrastructure Attack	Hijacking	1.1483	0.0098	2.2868	True
Facility/Infrastructure Attack	Hostage Taking (Barricade Incident)	3.8569	2.8839	4.8299	True
Facility/Infrastructure Attack	Hostage Taking (Kidnapping)	1.5322	1.1346	1.9298	True
Facility/Infrastructure Attack	Unarmed Assault	0.5322	-0.4461	1.5106	False
Hijacking	Hostage Taking (Barricade Incident)	2.7087	1.2702	4.1471	True
Hijacking	Hostage Taking (Kidnapping)	0.384	-0.7477	1.5156	False
Hijacking	Unarmed Assault	-0.616	-2.0582	0.8261	False
Hostage Taking (Barricade Incident)	Hostage Taking (Kidnapping)	-2.3247	-3.2896	-1.3597	True
Hostage Taking (Barricade Incident)	Unarmed Assault	-3.3247	-4.6401	-2.0093	True
Hostage Taking (Kidnapping)	Unarmed Assault	-1.0	-1.9704	-0.0296	True

To visualize what we've been doing with the one-way ANOVA test and the Tukey's range test, I made a graph below showing the average number of people killed per incident of different attack types. This graph clearly indicates that not all of the average number of people killed for each attack type are equal. From the first row of the Tukey's test result table, we reject the null hypothesis that the average number of people killed from an armed assault terrorist attack equals the average number of people killed from an assassination terrorist attack. Again, this graph below clearly shows that these two means are different.

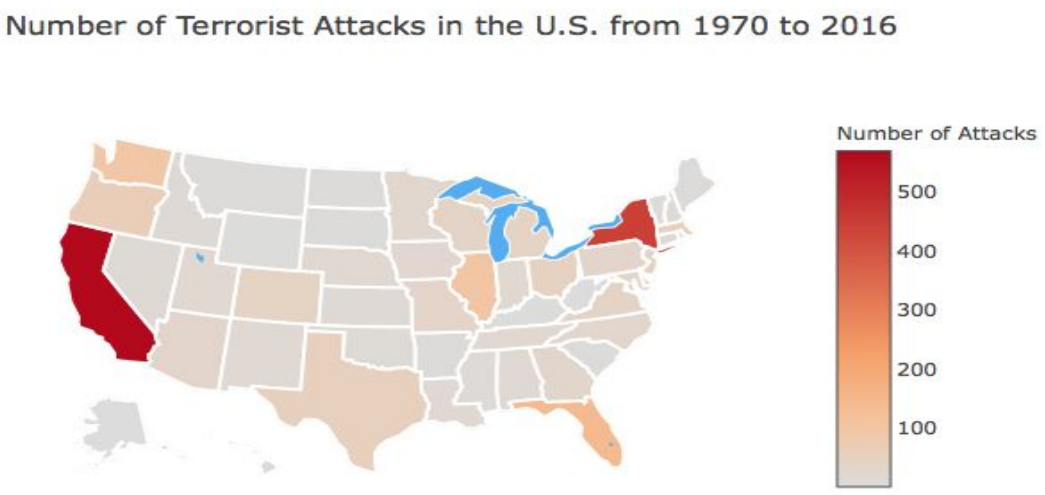


Now that we have some understanding of terrorist attack types, a natural question to ask next is on target types. Are there any particular target groups that are more prone to terrorist attacks? The bar chart below shows that private citizens/properties and military are the most frequent targets.

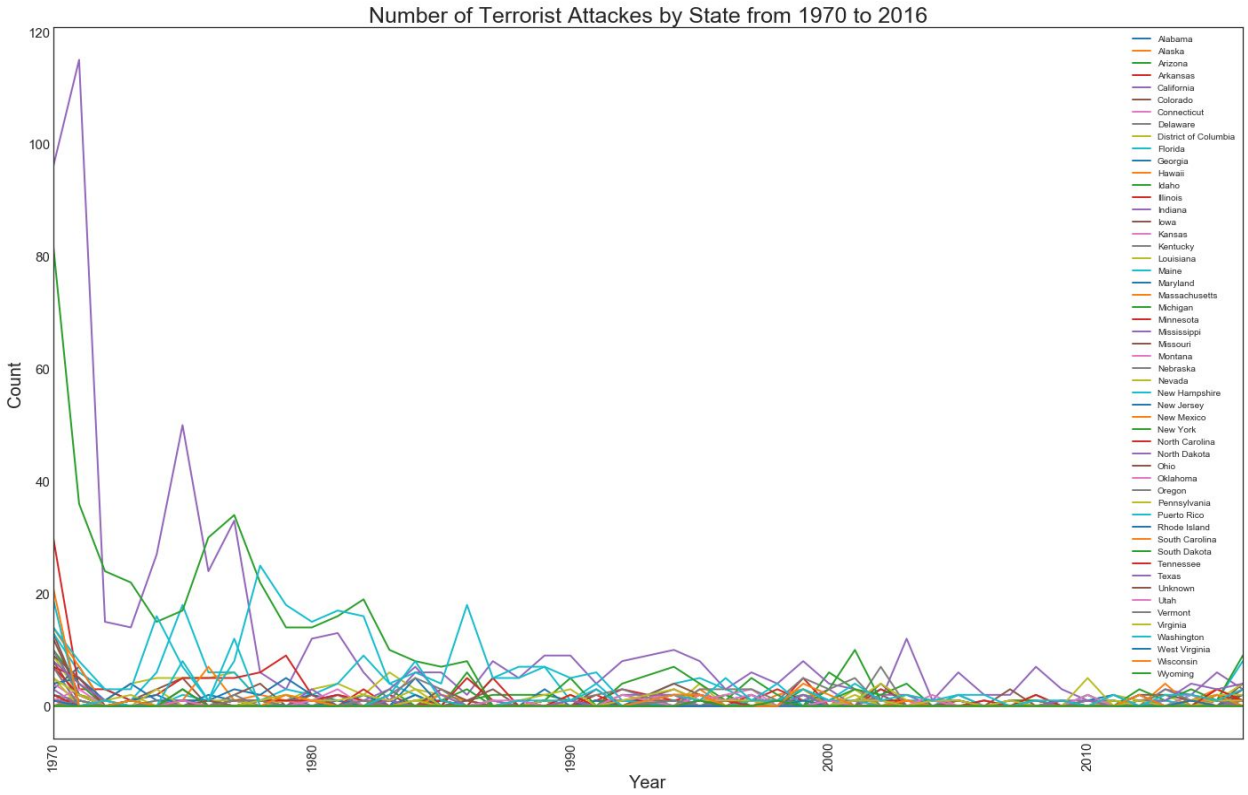


Along with exploring ‘global’ terrorism trends, I also want to explore a bit closer to home.

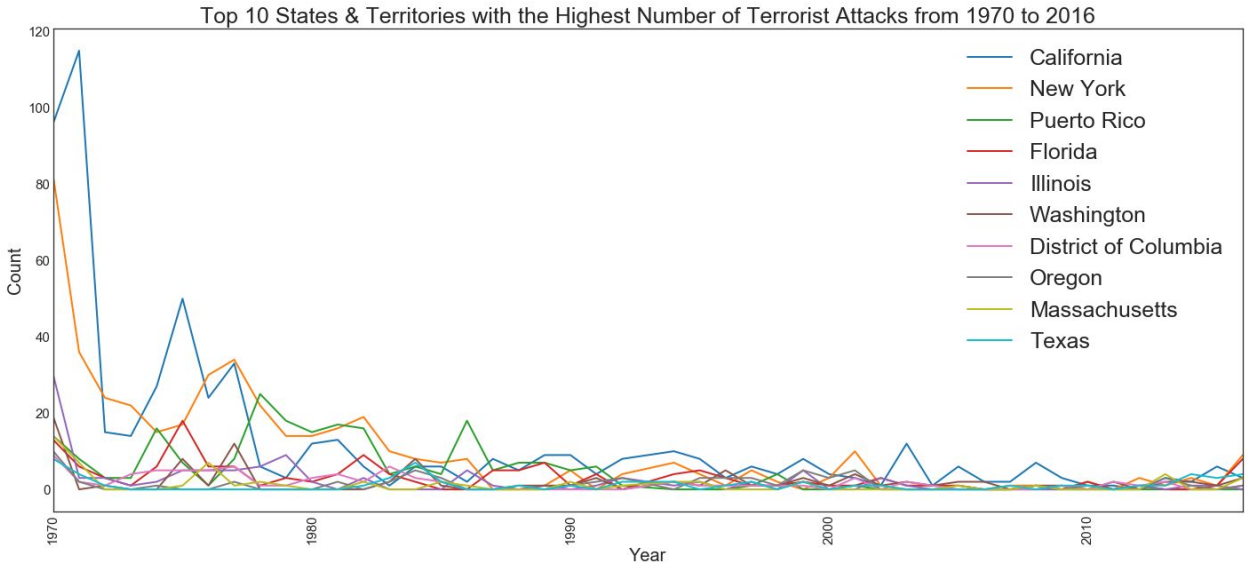
The map below shows that California and New York had the most number of terrorist attacks in the U.S. from 1970 to 2016.



To investigate further, I did a line chart with ‘year’ on the x-axis and ‘number of attacks’ on the y-axis. The chart shows that the United States has actually become a lot safer nowadays compared to the 1970s and 1980s.



In case this line chart with all the states and territories is too confusing, I made a new chart below zooming in on the top 10 states and territories that have the highest number of attacks from 1970 to 2016. Note that the legend is in a descending order of the number of attacks that occurred during this period of time.



PREDICTIONS USING MACHINE LEARNING

One of the columns in the original data set is called 'success'. This column has binary values of 1 and 0, with 1 being a 'successful' terrorist attack and 0 being a 'failed' terrorist attack. My goal is to use machine learning algorithms to make predictions on whether a terrorist attack will be a 'successful' attack. Since we already have labels of 'success' and 'failure', I propose to use supervised-learning algorithms to build three models, then compare which model does the best in terms of making the most accurate predictions.

To prepare the data set for the supervised-learning algorithms, I changed some column types to 'categorical', used pd.get_dummies( ) to expand each categorical column into many columns of 0s and 1s, then dropped any columns that can potentially cause collinearity.

The first model I used was a logistic regression model. I did a cross-validation grid search to decide on the best C parameter. I then fitted the model with C=0.1, used 3-fold cross-validation to make predictions, and computed the classification table.

	precision	recall	f1-score	support
0	0.61	0.21	0.31	16702
1	0.91	0.98	0.95	141573
avg / total	0.88	0.90	0.88	158275

We're interested in the precision rate and the recall rate for when the terrorist attack is a 'success', i.e., the row that is labeled as "1".

In statistical terms:

precision = true positive / (true positive + false positive) = true positive / positive predictions

recall = true positive / (true positive + false negative) = true positive / positive true state

In our case, precision tells us what proportion of terrorist attacks we predict would happen actually happened. In other words, this is the proportion of true positives in the

set of predicted terrorist attacks. On the other hand, recall tells us what proportion of terrorist attacks that actually happened were predicted by us correctly. In other words, the proportion of true positives in the set of actual terrorist attacks. To measure a model's performance, we want both rates to be as close to 1 as possible with 1 being the highest.

From the above classification table, we can see that for the logistic regression model, the precision rate we need is 0.91 and the recall rate we need is 0.98.

The second model I used was a random forest model. I did a cross-validation grid search to decide on the best n_estimators. I then fitted the model with n_estimators=30, used 3-fold cross-validation to make predictions, and computed the classification table. We see from the table below that the precision rate we need is 0.93 and recall rate we need is 0.96.

	precision	recall	f1-score	support
0	0.54	0.40	0.46	16702
1	0.93	0.96	0.95	141573
avg / total	0.89	0.90	0.89	158275

Lastly, I fitted a model using Principal Component Analysis (PCA) and Support Vector Machine (SVM). I first standardized the features with a scaling function, then reduced the dimension of the features with PCA to 10 orthogonal dimensions, then fitted the transformed features with an SVM algorithm. The classification table below shows that the precision rate we need is 0.90 and recall rate we need is 1.

	precision	recall	f1-score	support
0	0.53	0.04	0.07	16702
1	0.90	1.00	0.94	141573
avg / total	0.86	0.89	0.85	158275

Comparing all three models, model 2 - random forest gives the best precision rate and model 3 - PCA transformation and SVM gives the best recall rate. Depending on which rate we're interested in using, we can choose either of these two models.

Obviously, I can do a lot more tweaking on my three existing models. For instance, I can use RandomizedSearchCV to do a more extensive (but computationally efficient) grid search for the best parameters to fit the models. I can also try stacking more transformers into the pipeline to experiment with different combinations. Because of the time constraint on this project, I will leave my model fitting here.

FINAL WORDS

Before analyzing this dataset, I had no prior knowledge of global terrorism except for hearing about it occasionally in the news. It has been an interesting project to work on because it allowed me to gain some much-needed understanding on terrorism on both a global and a domestic scale.