DS 4420: Machine Learning and Data Mining II
Spring 2021
Project Checkpoint #1
Project Title: Analyzing Mass Shooting Data
Team Members: Ryan Liang

Introduction

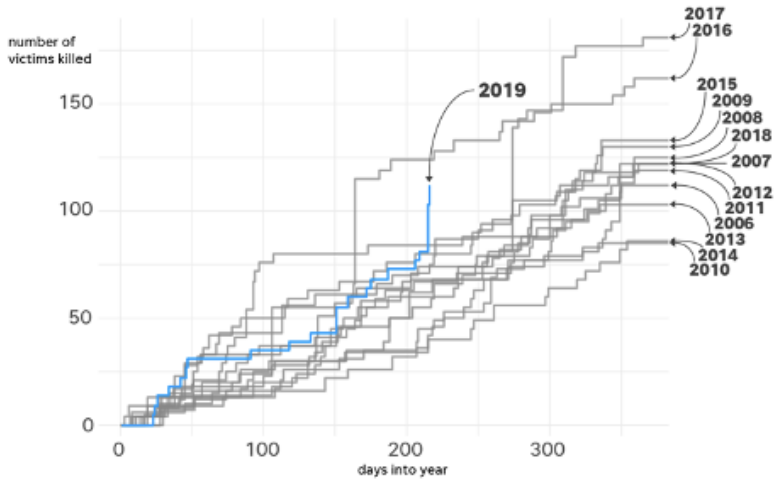
- **Problem Description**

Is it a classification or regression problem?

There are two approaches to my problem regarding how machine learning will be conducted. I plan to process the problem as mainly a classification problem with a later attempt at regression. **Classification**: My machine learning program will take in the variables that indicate the shooter's race, gender, and mental health status, and classify the data based on these variables.

Why is it important? Describe the motivation

Mass shootings in the U.S. are a serious problem, with 2019 showing record high numbers of both fatalities and shootings in just the first half of the year (as pictured in the graph). As we enter further into 2021, the effort to reduce the number of mass shootings has been aided negligibly, and almost nothing is getting better. The goal of the project will be developing key insights about the victims and also the perpetrators of mass shootings, and analyzing the mental health, race, location, and motivation (and any other important information) about the killers and where the mass shootings take place. When tackling a project of this magnitude, it's important to keep in mind the sensitivity of it, and how important it is to not to draw generalizations and conclusions about race and/or gender, as that is what fuels stereotypes and dangerously leads to racial profiling. However, it is also important to study and analyze key patterns in order to get a full comprehensive picture and to understand the problem better, especially when the issue is one of this caliber, and especially when enough resources haven't been allocated to studying the problem in the first place.



Dataset

There is one major dataset that I'm using. This dataset is the MassShootingsDatasetVer5.csv, and in it, we have information about the shooter, their race, their age, and other general demographic information. We also have several columns including location, description, summary, fatalities, injured, and # of victims total. The columns that we do not have to encode as numerical variables are the columns that already contain numerical values, which in this case are # of victims total, and the age of the victim. However, we encounter a problem when we want to run several machine learning models on the dataset, which happens to contain a lot of string variables such as race. The way I dealt with this problem was by encoding those several variables as numerical values.

For example, the variable "Mental Health Issues" was a column in our dataset that indicated whether the shooter suffered from Mental Health Issues during or before the time of the mass shooting. This was previously encoded as a variable with values "Yes", "No", "Unknown", etc., which is not translated well when running through a machine learning model. So in the form of preprocessing, I created a new list that would contain all the entries, but this time decoded as numerical values, and then iterated through the Mental Health Issues column and if the variable at the current entry was "Yes", I would encode that entry as "1", and for "No" it would be decoded as "0". I did this for the entire column, and I decoded/preprocessed other variables similar to this such as "Race" and "Gender". Mental Health and Gender ended up being binary variables (save for the "Unknown / 2" variable)

Similarly, in order to measure the performance of the machine learning models, I used the built-in StandardScaler() model to normalize and further preprocess the data. However, I ran into an error when it came to the structure of the data, and so this is something that I will be working on in the future. A good machine learning project isn't complete unless the data has been cleaned AND preprocessed/normalized properly, and so before I move too far into the next steps I will have to handle normalizing the data with StandardScaler().

Finally, I am splitting up the project by creating different tasks/problems for me to solve. For example, the first main task/problem I have set out to solve is the task of what is considered a mass shooting. In this problem, I explore what we consider a mass shooting by preserving the "Total victims" column, and from this finding out where the majority of mass shootings lie. In this case, the determinant of what is considered a mass shooting is measured by the total amount of victims. This can also be turned into a prediction task. I split the data up in two ways:

1. Training/Testing data (right now, encoded as a 50/50 split, but easily changeable)
2. Features/Labels (Features = Race, Gender, Mental Health Issues; Labels = Total Victims)

The best part about the way this is split and the inherent nature of the dataset is because we can change the label easily to “Fatalities” or “Injured”, and have the models be predicting almost the same real-world values.

In the future, I will find at least two more tasks/problems for me to solve and this will allow me to explore more of the dataset and get more experience with analyzing the dataset.

Methods

Describe your implemented methods and preliminary prospects. Given your current progress, what next steps are you planning?

The main methods that I have chosen for this project are running several classification models from scikit-learn and measuring their performance. The preliminary models that I have chosen are:

- Naive Bayes
- Logistic regression
- kNN
- Decision Tree
- Cross validation

After running these models through my training data features and labels and then testing them on my testing data, I do not see the performance I was hoping for. Before jumping into analysis, the metrics I had decided to use are:

- Accuracy
- Error
- Precision
- Recall
- ...which culminates into a Confusion Matrix

Unfortunately, the error on the first few models like Naive Bayes and Logistic Regression are way too high to even be considered. The error on kNN is similar. The model that performed somewhat decently was the Decision Tree, and so as a result I decided to run multiple iterations through this model to figure out the best possible depth of the decision tree to maximize performance. I plotted the performance of the Decision Tree on each iteration using matplotlib below.

In the future, since the models are not performing well and the Decision Tree performance is acting kind of out of caliber, I would like to revisit the way I had split up my data previously to make sure everything is accurate and split up the right way. I also think that issue may be from

the absence of StandardScaler(). Later on, in the next step, after I figure out what's wrong with StandardScaler(), I will rerun the models for this specific task and ensure that model performance improves.