Submission: **Create a short document (1-2 pages) in your GitHub describing the data wrangling steps that you undertook to clean your capstone project data set. What kind of cleaning steps did you perform? How did you deal with missing values, if any? Were there outliers, and how did you decide to handle them? This document will eventually become part of your milestone report.**

Against what most people think, data wrangling is a big part of data preparation and the full life-cycle process. From experience, more than 50% of a data scientist's time and work is doing just this. Whether you call it data preparation, cleaning, pre-processing, cleansing or wrangling, it's all the same. And, it can also be very demanding.

My approach to achieving this is in three parts: data selection, data preprocessing and data transformation. Data selection points to getting the data that can answer the question. That means when one takes measurements on an individual or case study; each cross-section of such measures should say something about what you want to achieve. The cross-section of variables in statistical terms is called an example or observation. Many a time, we work with samples because we don't have access to the whole population in question. These examples are unit components of such samples. We then generalize findings on the sample to the entire population after meeting some statistical conditions. How one extrapolate these conclusions depends on the intended models and measurement of the data.

In my case, the data was already available and met the requirement of the questions that needed answering. Since I did not have control over the data collection process, I had to make some assumptions. One of which is that the data collection is random to allow an equal chance of representation of every instance of the population in the sample, which helps to extrapolate findings. Second, I assume that each observation and measurements on them are independent of the other to prevent unnecessary bias. I restricted my selection to cases in the US. Each row in the data set are individual cases of terrorism in the US, and the columns are measurements made on such.

In the data preprocessing phase, I checked for missing values. As it stands, the data set contains a lot of those, which is typical for data collected in real life. My approach for dealing with this was to drop any column that had eighty percent missing values based on the thought that such measurements may not have any significance in the overall tasks, but still, keep them should in case we need to revisit findings for improvements. There is no hard rule to that in the industry. It is just a possible approach. Second, I also dropped variables that contained texts data, which are just comments explaining what is already in some other columns. And then, removed unknown values in some of the columns and imputed missing values left with the median of the values in each column.

Checking for outliers was also part of this phase. There were some outliers, most notable "propvalue", "natlty1", "natlty1", "nperps", "nkill", "nkillus"and "nwound." Please refer to the Codebook for the meaning of the variables. And this causes skewness in the distribution of variables as seen in some of the plots above. These could have been an issue if we are considering linear models like Linear or Logistic Regression models. One would need to examine these issues closely and correct for such before building the model. Outliers are not necessarily bad. It depends on business needs, and how much accuracy one is trying to achieve. Since I am considering non-linear models like the Random Forest, these issues are usually implicitly addressed. That is a big advantage of using such models. I decided to leave them in there to see how the model performs with an option of revisiting them if the outcome is not favorable.

Last, it is advisable to hot code categorical variables with high cardinality. The recommendation is anything more than two categories in the feature to provide an optimum result. The data transformation phase was used to implement this by converting these categorical variables to numerical because the intended machine learning algorithm I propose using cannot handle string or text values. One way to do that is to use Label Encoding module in scikit learn library. However, with that, you now run a risk of the curse of dimensionality. Since I am not doing unsupervised clustering, I got around this by using Random Forest selection of essential variables feature. This feature prunes the dimension down to one that can provide useful accuracy.