Terrorism Dataset Analysis

December 2, 2018

1 Visualization Techniques in Data Analysis

By Tessa Munoz ## Introduction

As technology advances and the need for understanding larger data increases, data visualization techniques are an extremely important tools to help aid our analysis. When looking at numeric data, one technique for analyzing it is to try to find patterns. For a very small amount of values and data that only has one variable, this may be easy. In contrast, as the size and complexity of the data increases, it becomes near impossible to see meaningful patterns by looking at numbers alone.

Using numerical analysis techniques, we can be given equations or quantities that will provide us with answers. It is typically easier for people to understand difficult concepts with the use of visual aids. Just as in the majority of educational textbooks, there are graphs or visuals to help readers quickly understand what concepts the numbers are conveying.

Graphs and plots can be made to adapt and augment descriptive statistics such as: averages, growth trends, comparisons, stability, regressions, and outliers. This is in contrast to the terse nature of exclusively analytical representations, such as equations.

Data visualization helps in many ways, such as: conveying concepts that can be incredibly difficult to compute and/or understand, providing captivating visual support for findings, accessing many "viewpoints" for analyzing data for different needs, and helping increase the interest in news, research, services and products. While it may be annoying when an ad company knows just how to pique your interest, it is great for getting individuals to pay attention to important concepts or quickly understand findings. An important aspect regarding visuals is deciding which visualization technique works best to suit specific needs or most illustratively represent the data that is being analyzed effectively [1].

My project explores different visualization techniques and why they are useful and their advantages/disadvantages versus other types of visuals. Some may have little to no difference other than being preferred due to a trend, while others may provide better detailed information or just not work in certain cases. New visualization techniques are being created each day, some for the purpose of being more appealing and some provide interactive and advanced ways to explore data and can be used for amazing education purposes. In my project I focused on many of the base visualization techniques to demonstrate their usefulness for the data I obtained and because these are the most commonly used. My visuals will include pie charts, bar graphs, histograms, scatter plots, box and whisker charts, line graphs, pair plots and map overlays. These base graphs, plots, and charts can be enhanced to provide quicker understanding or more engaging visuals and there is a spectrum of enhancement techniques, some which I discuss, like coloring and maps.

Majoring in Mathematics with a focus in in Computer Science, I wanted to incorporate both studies and learning outcomes to this project. I chose to use a modern programming language and software tools/applications that are open-source to complete my project in, including: - Python Programming Language [2]. - Jupyter Notebook - a web application that makes sharing and accessing live code easy [3]. - Pandas - Library made for Python meant for data manipulation/analysis [4]. - Other program libraries as listed in my code to help provide certain visual features like, maps and color schemes. - GitHub - a code repository. This is a place to store and share projects. It also allows for collaboration [5].

By taking this approach, I have not only learned about the process of data analysis, but I have gained experience in using other types of analysis software. I was also able to advance my programming, analysis and technological skills.

The data used for this project is called the, Global Terrorism Database (GTD) obtained from the web page of the National Consortium for the Study of Terrorism and Responses to Terrorism (START)[6]. Spanning from 1970-2017, it contains an amazing range of information for each event, such as: number of events, day, year, month, location in latitude/longitude, weapon type, target types, etc... This data was chosen because of the quality, quantity, variation of data, and interest it possessed. Not all the variables were included in my dataframe and some of the formats were modified, like, the event id which was changed to date time format and given a new column. These modifications were done to reduce the data size and/or provide the proper format based on the visualizations being demonstrated. Given my qualifications, the data presented is not intended to draw conclusions for analysis research. Rather, it is intended to show different types of visualizations, examples of possible analysis routes, and flexible programing techniques used to create the visuals.

Below is my project, presented in Jupyter Notebook. You'll find different "blocks" or separated sections; the sections are categorized as follows: - Markdown blocks: Contain my text that is not code that will provide information regarding the subject, code, visualization and findings. - Code blocks: Contain sections of my code that, along with the code, may contain comments that describe what it's doing. - Output blocks: These may appear below the corresponding code blocks and will display the intended output/effect of the code, for example, visualizations.

Links to each block is provided in the Contents and can be used to quickly navigate to that section. This project can be accessed on my GitHub, https://github.com/TesMunoz/SeniorSim/blob/master/Terrorism%20Dataset%20Analysis.ipynb. References can be found at the end of the notebook.

1.1 Contents

1. Project Intialization
2. Graphs

- Section **??**
- Section **??**
- Section **??**
- Section **??**
- Section **??**
- Section **??**
- Section **??**
- Section **??**

1.2 Project Intialization

1.2.1 Steps:

```
1) Import necessary libraries.
2) Write Any Needed Functions.
2) Import Original Data.
3) Clean Data based on bad entries, missing information, etc...
4) Save cleaned data to a new file to keep the original unedited.
5) Import cleaned data and begin to analyze.
```

1.2.2 Import Libraries:

In Python, libraries imported into the current process are held in memory. As memory references are disposed of by a garbage collection process, libraries can be globally imported (as seen below).

```
In [1]: import pandas as pd
        from pandas.plotting import scatter_matrix

        import numpy as np

        import matplotlib.pyplot as plt
        import matplotlib.collections as mcol
        from matplotlib import cm

        import seaborn as sns

        # use this style of plots
        plt.style.use('ggplot')
        # print plots in notebook
        %matplotlib inline

        import sklearn as sk
        from sklearn.cluster import *
        from sklearn.preprocessing import OneHotEncoder


        ## Hack for installing using Basemap in the root Anaconda virtual environment...
        import os
        import conda

        # conda_file_dir = conda.__file__
        # conda_dir = conda_file_dir.split('lib')[0]
        # proj_lib = os.path.join(os.path.join(conda_dir, 'share'), 'proj')
        # os.environ["PROJ_LIB"] = proj_lib
        conda_file_dir = conda.__file__
        conda_dir = conda_file_dir.split('lib')[0]
        proj_lib = os.path.join(os.path.join(conda_dir, 'share'), 'proj\\')
        os.environ["PROJ_LIB"] = proj_lib
        if not os.path.exists(proj_lib):
            print(proj_lib)
            os.makedirs(os.path.dirname(proj_lib))

        from mpl_toolkits.basemap import Basemap #map overlay
```

```
C:\ProgramData\Anaconda3\share\proj\
```

1.2.3 Generic Functions: [7]

- **Convert Event ID To Datetime Format**
- Many datasets use inconsistent date time labels. To circumvent domain-specific datetime encodings, they can be coerced to a native data structure that will allow for more flexible use.

```
In [4]: def convert_eventid_to_datetime(df, ignore_bad_data=True):
            """Converts dataframe with event_id column to date time formats. Returns new DF wi
            from datetime import datetime

            datetime_series = []
            ix_to_drop = []
            count = 0
            for i, row in df.iterrows():
                try:
                    year = int(row['iyear'])
                    month = int(row['imonth'])
                    day = int(row['iday'])
                    datetime_series.append( datetime(year, month, day) )
                except:
                    ix_to_drop.append(count)
                    if not ignore_bad_data:
                        print("Bad Event ID date format: {}, {}, {}. Event ID: {}"
                              .format(year, month, day, row['eventid']) )
                        raise ValueError('Event Id data contains bad dates.')
                count += 1

            print('Dropped {} malformed event id rows out of {}.'.format(len(ix_to_drop), df.sl
            df = df.drop(df.index[ix_to_drop])
            df['datetime'] = pd.to_datetime(datetime_series)

            return df
```

1.2.4 Steps 2-5: import, clean, create new data, and sub-sample data

```
1) Read in dataframe, specifying subset of columns.
2) Clean Data, dropping any NAN values from data set.
3) Use Convert to date time function.
```

```
4) Sub-sample data to manageable size using a random sample generator in order to provide a rep
5) Save as new data frame.
```

```
In [5]: #read in the data file and make into a pandas dataframe: location, only specific colum
        original_df = pd.read_csv('./inputs/globalterrorismdb.csv', encoding='ISO-8859-1', use
                                  'latitude', 'longitude','region', 'crit1', 'crit2', 'crit3',
                                   'weapsubtype1'],
                                  dtype={'iyear':'int', 'imonth':'int', 'iday':'int'});

        #use a manageable sample size
        num_rows = 5000 # Reduce to 5k for testing plots
        #shows the size of the original data
        # print('Original shape of CSV: {}'.format(original_df.shape))

        # DropNA() is dropping all NaN values from the dataset. Talk about potential bias. Inv
        #Free up memory by deleting the unused origial data
        terrorism_df = original_df.dropna().sample(n=num_rows)
        del original_df
        print(type(terrorism_df))

        #displays the columns being used
        print(terrorism_df.columns)

        # Convert known event times to timestamp column
        datetime_df = convert_eventid_to_datetime(terrorism_df, ignore_bad_data=True)
        print(datetime_df.shape)

        terrorism_df = datetime_df
        del datetime_df

        terrorism_df.iloc[0:10] #output the first 10 roos of the dataframe
```

```
<class 'pandas.core.frame.DataFrame'>
Index(['eventid', 'iyear', 'imonth', 'iday', 'country', 'region', 'latitude',
       'longitude', 'crit1', 'crit2', 'crit3', 'attacktype1', 'targtype1',
       'targsubtype1', 'weaptype1', 'weapsubtype1'],
      dtype='object')
Dropped 13 malformed event id rows out of 5000.
(4987, 17)
```

```
Out[5]:              eventid  iyear  imonth  iday  country  region   latitude  \
        120798  201309060021   2013       9     6        4       6  33.263079
        28066   198605230006   1986       5    23      159       3 -10.680652
        168343  201611010013   2016      11     1        4       6  34.453813
        62776   199607170010   1996       7    17       75       8  53.553813
        118003  201306170014   2013       6    17      137      11 -19.616667
        69106   199907110004   1999       7    11       93       5   5.589161
```

```
84970   200711250010  2007      11    25       153        6  32.969967
157828  201601270025  2016       1    27       147       11  10.869205
147324  201505030089  2015       5     3         4        6  36.781832
166802  201609210008  2016       9    21        37       11  11.099467

         longitude  crit1  crit2  crit3  attacktype1  targtype1  targsubtype1  \
120798   68.572411      1      1      1            2          3          24.0
28066   -76.259918      1      1      1            3          2          21.0
168343   70.484844      1      1      1            3          3          23.0
62776     9.991586      1      1      1            3         14          73.0
118003   34.750000      1      1      0            2          4          27.0
69106    95.360966      1      1      1            7          8          49.0
84970    70.276797      1      1      0            3          4          27.0
157828   12.847621      1      1      1            3          4          36.0
147324   71.076543      1      1      1            2          3          24.0
166802   14.195931      1      1      1            3         14          67.0

         weaptype1  weapsubtype1   datetime
120798           5           5.0 2013-09-06
28066            6          16.0 1986-05-23
168343           6          29.0 2016-11-01
62776            6          16.0 1996-07-17
118003           5           5.0 2013-06-17
69106            8          18.0 1999-07-11
84970            6          11.0 2007-11-25
157828           6          13.0 2016-01-27
147324           5           5.0 2015-05-03
166802           6          13.0 2016-09-21
```

1.2.5 Output:

The above output shows: - The rows we defined for our specific dataframe. - The amount of data that was excluded based on our criteria. - The first 10 rows of our new dataframe with the added datetime column.

2 Visualizations

2.1 Bar Chart or Bar Graph: [8]

Bar Charts/Graphs are among the most common types of viualizations used to display the relationship between numerical variables and catergorical variables. They are well suited to show variations of numerical values between different groups, for example; sales of different products within a certain time frame. The visualization below displays the amountt of terrorism attacks by year to get a quick visual of the changes in gross volume as the years progress.

These charts/graphs can be advanced in different ways to provide more information or aesthetics. Demonstrated and explained below are added colors.

```
In [6]: ## View Global Terrorism attack years volumes by type
```