ANALYSIS OF GLOBAL TERRORISM DATASET

AN ISYS812 PROJECT REPORT

December 13, 2019

PROFESSOR : Guillaume Faddoul
TEAM : Mukul Pathak, Shraddha Upadhyay, Mansa Veeramachineni and Disha Shah

Introduction

The dataset we are working is on Global Terrorism Database which is database containing information about terrorist attacks around the world from 1970 through 2017. The Global Terrorism Database (GTD) is the most comprehensive unclassified database of terrorist attacks in the world. It specifically has information about over 180,000 terrorist attacks. For each event, a wide range of information is available, including the date and location of the incident, the weapons used, nature of the target, the number of casualties, and – when identifiable – the group or individual responsible.
As we are aware that attacks caused by terrorists are increasing on a daily basis around the world. We wanted to use this project as an opportunity to analyse database like this where we can perform analytics on terrorism caused by terrorists globally and try to make sense of some factors related to it.

Characteristics of the GTD

- Contains information on over 181,000 terrorist attacks.
- Currently the most comprehensive unclassified database on terrorist attacks in the world.
- Includes information on more than 91,000 bombings, 20,000 assassinations, and 13,000.
- The database contains 135 columns and 181,692 rows of information about terrorist attacks
- We are working with the following data entries :
  - Event ID (primary key of the table)
  - Year of attack (Year in which the attack was done)
  - Country code (Numerical code for the country)
  - Country name (Name of the Country)
  - Type of attack (E.g. Mexico, Philippines)
  - Type of target (E.g. Journalists and media, Police)
  - Sub type of target (E.g. Embassy/Consulate, Police Security Forces/Officers)
  - Name of target (E.g. Julio Guzman, U.S. Embassy)
  - Gang name (E.g. MANO-D, Black Nationalists)
  - Sub type of weapon (E.g. Automatic or Semi-Automatic Rifle, Gasoline or Alcohol)
  - Number of kills (Numeric value of the number of people killed in the attack)
  - Value of the property (Numeric value of the loss of property due to the attack)
  - Criticality of attack (Criticality is “1” if the attack causes huge amount of loss in life and property and it is set to “0” if the attack is not very critical)
  - Success rate of the attack (Success rate of the terrorist attack)

DRIVING QUESTION

The driving question for us has been majorly to understand the dataset which leads us to analyse how the terrorist activity has affected the life of people, how much economical harm it has provided and majorly to figure out factors and it's affect on the success of attack.

In this project, we will be answering these 5 questions and involve in a prediction:

Q1. Most terrorism affected country each decade.

Q2. To find the most popular target type assassinated.

Q3. Most active terrorist group's activity across the years, weapon used and kills caused by it.

Q4. Economical loss of each decade and its comparison based on criticality of the attack.

Q5. Regression to see success of terrorist attack

Prediction : To predict success of terrorist activities based on multiple factors.

DATA CLEANING

Our dataset has 135 columns which we are going to modify according to the requirement of the question. We will clean our data based on each question's requirements, only keep the data which are required and drop the rest of data.

Data Cleaning for Question 1

Here we derive the most terrorism affected countries in each decade. To get the desired result we only need few variables. We have chosen to retain 'eventid','iyear','country', and 'country_txt'.

```
In [9]: #We filter out the columns which we need to analyse and answer this question.
        df1 = df.filter(['eventid','iyear','country', 'country_txt'], axis=1)
```

```
In [10]: df1
```

Out[10]:

	eventid	iyear	country	country_txt
0	197000000001	1970	58	Dominican Republic
1	197000000002	1970	130	Mexico
2	197001000001	1970	160	Philippines
3	197001000002	1970	78	Greece
4	197001000003	1970	101	Japan
...	...	...	...	...
181686	201712310022	2017	182	Somalia
181687	201712310029	2017	200	Syria
181688	201712310030	2017	160	Philippines
181689	201712310031	2017	92	India
181690	201712310032	2017	160	Philippines

181691 rows × 4 columns

Data Cleaning for Question 2

Our aim was to find the most popular target type assassinated from the data. so we start out by filtering the data and retaining few crucial variables such as ‘eventid’, 'attacktype1_txt', ‘targsubtype1_txt' and ‘target1’.

```
In [22]: df2 = df.filter(['eventid','attacktype1_txt','targtype1_txt','targsubtype1_txt','target1'], axis=1)
         df2
```

Out[22]:

	eventid	attacktype1_txt	targtype1_txt	targsubtype1_txt	target1
0	1.970000e+11	Assassination	Private Citizens & Property	Named Civilian	Julio Guzman
1	1.970000e+11	Hostage Taking (Kidnapping)	Government (Diplomatic)	Diplomatic Personnel (outside of embassy, cons...	Nadine Chaval, daughter
2	1.970010e+11	Assassination	Journalists & Media	Radio Journalist/Staff/Facility	Employee
3	1.970010e+11	Bombing/Explosion	Government (Diplomatic)	Embassy/Consulate	U.S. Embassy
4	1.970010e+11	Facility/Infrastructure Attack	Government (Diplomatic)	Embassy/Consulate	U.S. Consulate
5	1.970010e+11	Armed Assault	Police	Police Building (headquarters, station, school)	Cairo Police Headquarters
6	1.970010e+11	Assassination	Police	Police Security Forces/Officers	Juan Maria de Lucah/Chief of Directorate of in...
7	1.970010e+11	Bombing/Explosion	Utilities	Electricity	Edes Substation
8	1.970010e+11	Facility/Infrastructure Attack	Military	Military Recruiting Station/Academy	R.O.T.C. offices at University of Wisconsin, M...

Now, we drop and remove all unwanted data caused by NaN and null values.

```
df2a = df2.dropna(subset=['targsubtype1_txt'])
df2a.head(5)
```

	eventid	attacktype1_txt	targtype1_txt	targsubtype1_txt	target1
0	1.970000e+11	Assassination	Private Citizens & Property	Named Civilian	Julio Guzman
1	1.970000e+11	Hostage Taking (Kidnapping)	Government (Diplomatic)	Diplomatic Personnel (outside of embassy, cons...	Nadine Chaval, daughter
2	1.970010e+11	Assassination	Journalists & Media	Radio Journalist/Staff/Facility	Employee
3	1.970010e+11	Bombing/Explosion	Government (Diplomatic)	Embassy/Consulate	U.S. Embassy
4	1.970010e+11	Facility/Infrastructure Attack	Government (Diplomatic)	Embassy/Consulate	U.S. Consulate

Since the data is cleaned, now we need information about the target which has been attacked the most. This we achieve by applying value_counts on targtsubtype1_txt.

```
pd.set_option('display.max_rows',200)
```

```
df2a['targsubtype1_txt'].value_counts()
```

```
Unnamed Civilian/Unspecified                                  11596
Police Security Forces/Officers                               11178
Military Unit/Patrol/Convoy                                    8277
Military Personnel (soldiers, troops, officers, forces)        7963
Government Personnel (excluding police, military)              6610
```

Data Cleaning for Question 3

We here want to find the most active terrorist group's activity across the year, weapon used, and kills caused by it. We start out by cleaning and filtering the data and bringing it down to the few variables which will help in determining active terrorist groups, weapons used, and kills caused.

So, we filter the data and retain variables such as 'eventid','gname','weapsubtype1_txt' and 'nkill'.

```
df3 = df.filter(['eventid','gname','weapsubtype1_txt','nkill'], axis=1)
df3
```

	eventid	gname	weapsubtype1_txt	nkill
0	1.970000e+11	MANO-D	NaN	1.0
1	1.970000e+11	23rd of September Communist League	NaN	0.0
2	1.970010e+11	Unknown	NaN	1.0
3	1.970010e+11	Unknown	Unknown Explosive Type	NaN
4	1.970010e+11	Unknown	NaN	NaN
5	1.970010e+11	Black Nationalists	Unknown Gun Type	0.0

We work on the variables retained and drop the null values present in the data.

Data Cleaning for Question 4

This question is based on the most economically damaging attack of each year. As we clean the data, we would require 5 main variables which are 'eventid', 'iyear','country_txt', 'propvalue' and 'crit1.

```
df4 = df.filter(['eventid','iyear','country_txt','propvalue','crit1'], axis=1)
```

```
df4
```

	eventid	iyear	country_txt	propvalue	crit1
0	1.970000e+11	1970	Dominican Republic	NaN	1
1	1.970000e+11	1970	Mexico	NaN	1
2	1.970010e+11	1970	Philippines	NaN	1
3	1.970010e+11	1970	Greece	NaN	1
4	1.970010e+11	1970	Japan	NaN	1

Now we drop the null values in present data.

```
df4a = df4.dropna()
df4a
```

	eventid	iyear	country_txt	propvalue	crit1
7	1.970010e+11	1970	United States	22500.0	1
8	1.970010e+11	1970	United States	60000.0	1
10	1.970010e+11	1970	United States	0.0	1
11	1.970010e+11	1970	United States	305.0	1
14	1.970010e+11	1970	United States	2000000.0	1

Now, propvalue has lots of garbage data as “-99”. We have to remove them all.

```
df4a = df4a[df4a.propvalue != -99]
df4a
```

	eventid	iyear	country_txt	propvalue	crit1	decade
7	1.970010e+11	1970	United States	2.250000e+04	1	1970s
8	1.970010e+11	1970	United States	6.000000e+04	1	1970s
10	1.970010e+11	1970	United States	0.000000e+00	1	1970s
11	1.970010e+11	1970	United States	3.050000e+02	1	1970s
14	1.970010e+11	1970	United States	2.000000e+06	1	1970s

Data Cleaning for Question 5

Here we determine the success rate of terrorist attacks around the world. We filter out and retain variables such as eventid’, ‘success', ‘attacktype1_txt’ and ‘targtype1_txt’.

```
df5 = dfreg.filter(['eventid','success','attacktype1_txt','targtype1_txt','nkill'], axis=1)
df5
```

	eventid	success	attacktype1_txt	targtype1_txt	nkill
0	1.970000e+11	1	Assassination	Private Citizens & Property	1.0
1	1.970000e+11	1	Hostage Taking (Kidnapping)	Government (Diplomatic)	0.0
2	1.970010e+11	1	Assassination	Journalists & Media	1.0
3	1.970010e+11	1	Bombing/Explosion	Government (Diplomatic)	NaN
4	1.970010e+11	1	Facility/Infrastructure Attack	Government (Diplomatic)	NaN

This is how our data will look after we drop the unwanted values.

```
df5=df5.dropna()
df5
```

	eventid	success	attacktype1_txt	targtype1_txt	nkill
0	1.970000e+11	1	Assassination	Private Citizens & Property	1.0
1	1.970000e+11	1	Hostage Taking (Kidnapping)	Government (Diplomatic)	0.0
2	1.970010e+11	1	Assassination	Journalists & Media	1.0
5	1.970010e+11	1	Armed Assault	Police	0.0
6	1.970010e+11	0	Assassination	Police	0.0

DATA ANALYSIS

In order to analyse the data better ,we have imported these following packages :

- Numpy
- Pandas
- Matplotlib
- Seaborn
- Sklearn : linear_model
- Statsmodels
- Scipy
- Warnings

Analysis for Question 1: Which is the most terrorism affected country each decade?

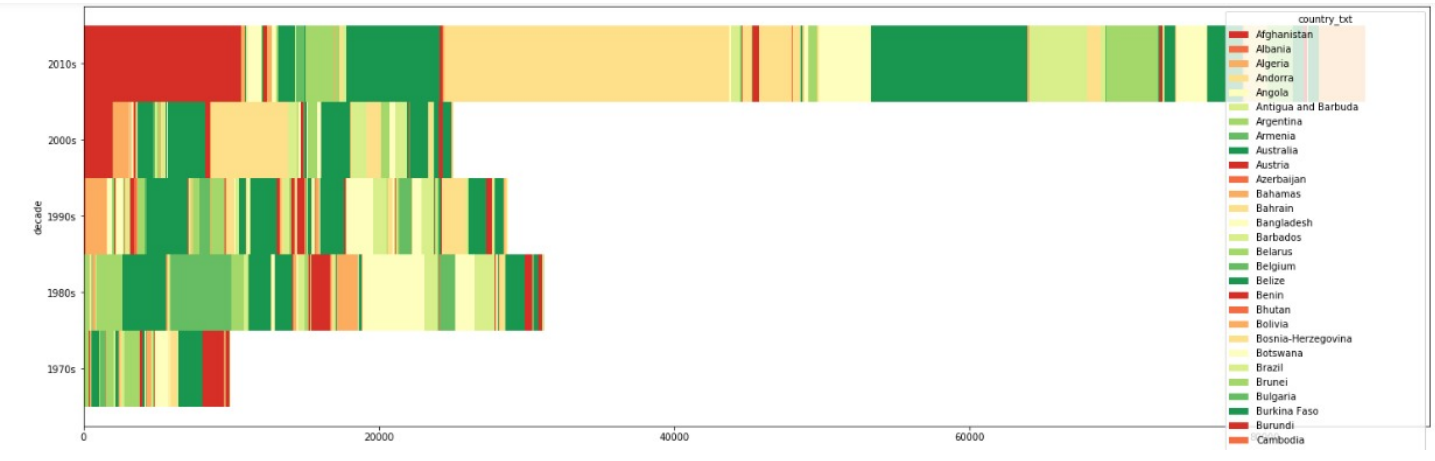
We first create the column decade. Which uses values from ‘iyear’ column and gives decade of that year. We achieve this using normalisation.

```
    decade = []
    for row in df['iyear']:
        if row < 1980:    decade.append('1970s')
        elif row < 1990:  decade.append('1980s')
        elif row < 2000:  decade.append('1990s')
        elif row < 2010:  decade.append('2000s')
        else:    decade.append('2010s')
    # Create a column from the list
    df1['decade'] = decade
```

```
df1['decade'].value_counts()
```

```
2010s    86815
1980s    31160
1990s    28762
2000s    25040
1970s     9914
Name: decade, dtype: int64
```

The graph now we get looks something like this :

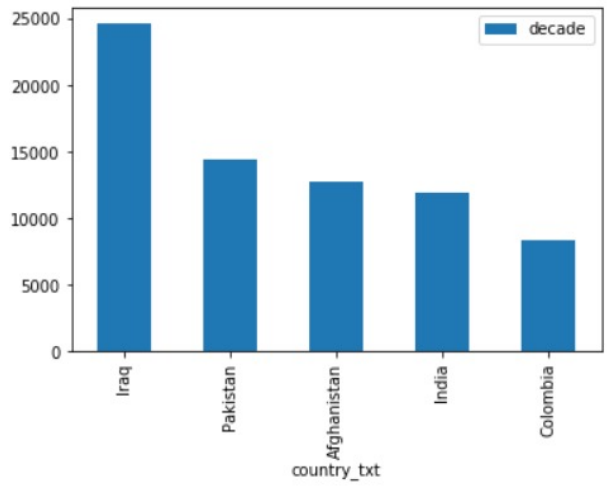


Since the graph can't be understood, we further segregate the data we have for each decade.

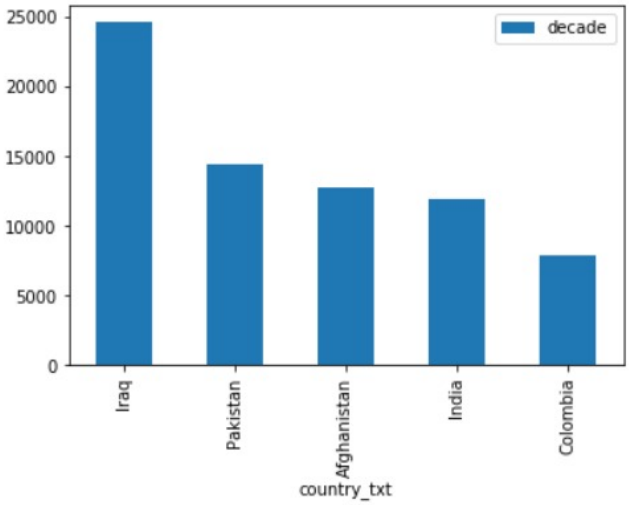
```
df70 = d12[d12['decade'] >= '1970s']
df701=df70.groupby('country_txt').count().sort_values('decade',ascending=False).head(5).reset_index()
df701
```

	country_txt	decade
0	Iraq	24636
1	Pakistan	14368
2	Afghanistan	12731
3	India	11960
4	Colombia	8306

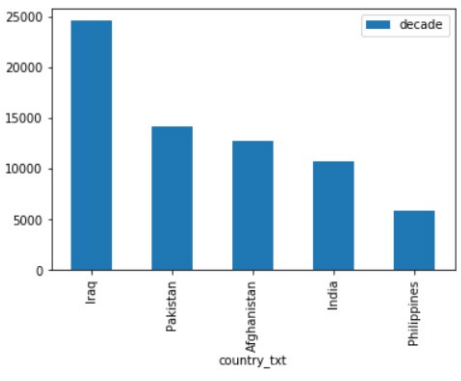
Now, we plot the bar-graph for these data for each decade. Here, we see the plot for the 1970s decade.



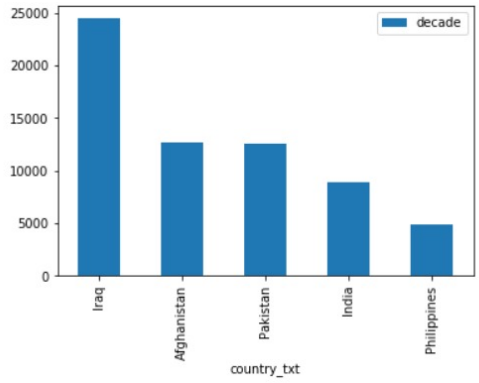
Decade 1970s



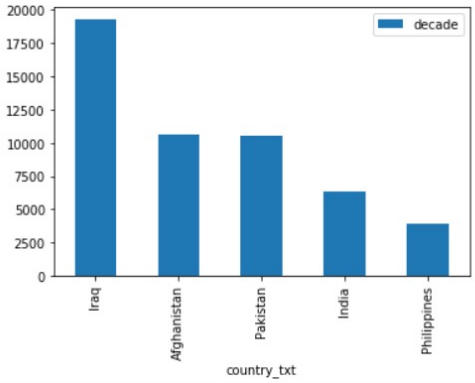
Decade 1980s



Decade 1990s



Decade 2000s



Decade 2010s

It can be observed from each graph that :

For 1970's, Iraq was the most affected country with 24636 attacks. Followed by Pakistan with 14368 attacks.

For 1980's, Iraq once again was the most affected country with 24630 attacks. Followed by Pakistan with 14351 attacks.

For 1990's, Iraq once again was the most affected country with 24600 attacks. Followed by Pakistan with 14155 attacks.

For 2000's, Iraq once again was the most affected country with 24475 attacks. Followed by Afghanistan with 12607 attacks.

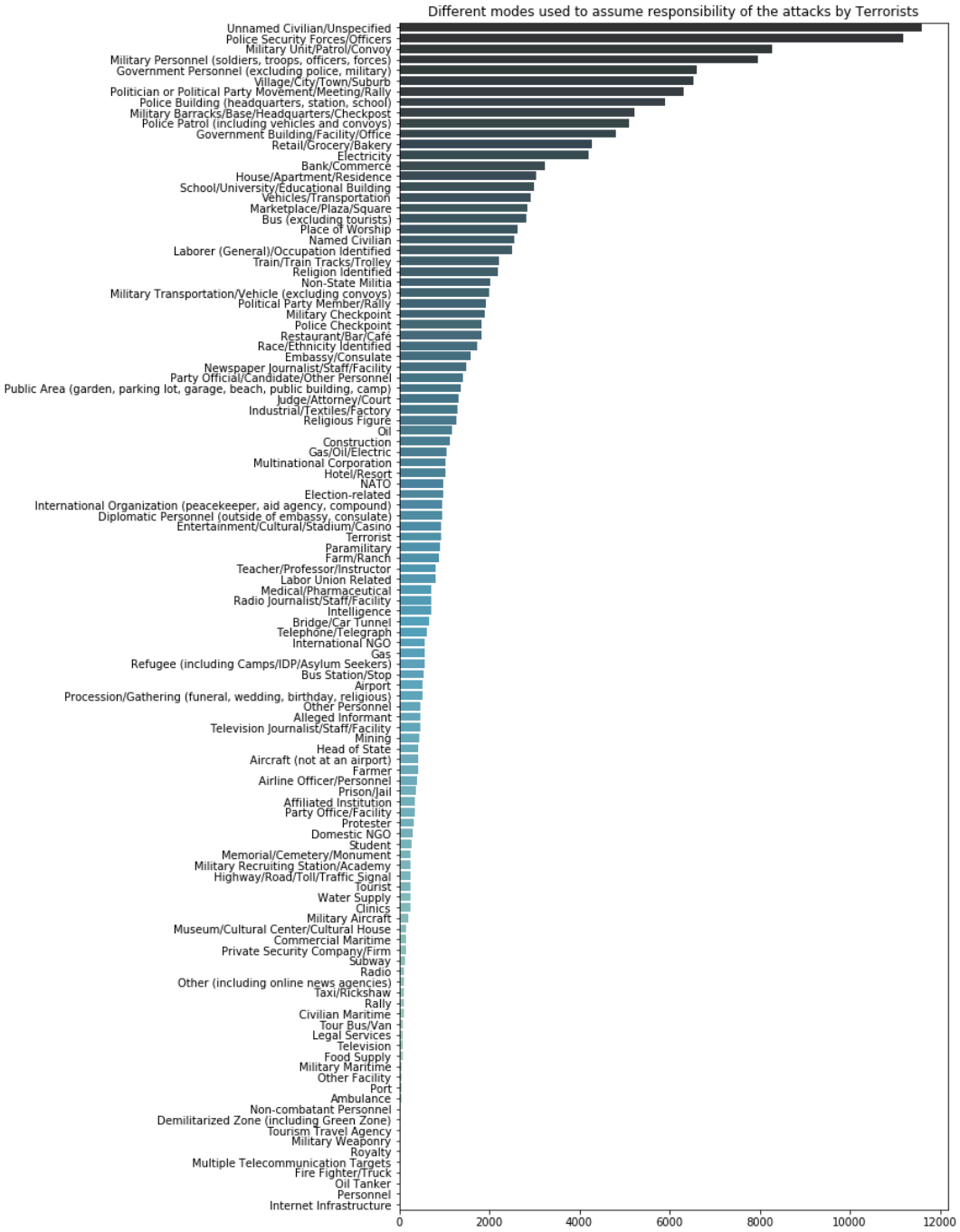
For 2010's, Iraq once again was the most affected country with 19286 attacks. Followed by Afghanistan with 10658 attacks.

It can be seen that Iraq remains on the top of all decades as most affected country. Pakistan and Afghanistan tussle for the second spot in each decade.

Analysis of Question 2: Which of the among is the most popular target type assassinated

To derive the target type assassinated from the whole dataset we again start by filtering data and only coordinating with required variables.

Now we plot the graph to display the data derived.



Analysis of Question 3: Determine the most active terrorist group's activity across the year, weapon used, and kills caused by it.

Since, our data has lot's of Unknown gang names, it's better to filter out these "Unknown" entries.

```
df3= df3[df3.gname != "Unknown"]
df3
```

	eventid	gname	weapsubtype1_txt	nkill
0	1.970000e+11	MANO-D	NaN	1.0
1	1.970000e+11	23rd of September Communist League	NaN	0.0
5	1.970010e+11	Black Nationalists	Unknown Gun Type	0.0
6	1.970010e+11	Tupamaros (Uruguay)	Automatic or Semi-Automatic Rifle	0.0
8	1.970010e+11	New Year's Gang	Molotov Cocktail/Petrol Bomb	0.0
9	1.970010e+11	New Year's Gang	Gasoline or Alcohol	0.0

Now, we count and see which group has most number of terror attacks :

```
df3a['gname'].value_counts()
```

```
Taliban                                                    6006
Islamic State of Iraq and the Levant (ISIL)                4282
Shining Path (SL)                                          3665
Farabundo Marti National Liberation Front (FMLN)           2396
New People's Army (NPA)                                    2368
Al-Shabaab                                                 2308
Irish Republican Army (IRA)                                2196
```

Since, there are 23 entries, a pie-chart with so many entries will not come out well. Hence, we will plot the same data with the enrties which have killed 1000+ people and treating rest all as others.

To get sum of all other values,

```
sumOthers = df3TK.iloc[6:24]
sumOthers.nkill.sum()
```

```
1982.0
```

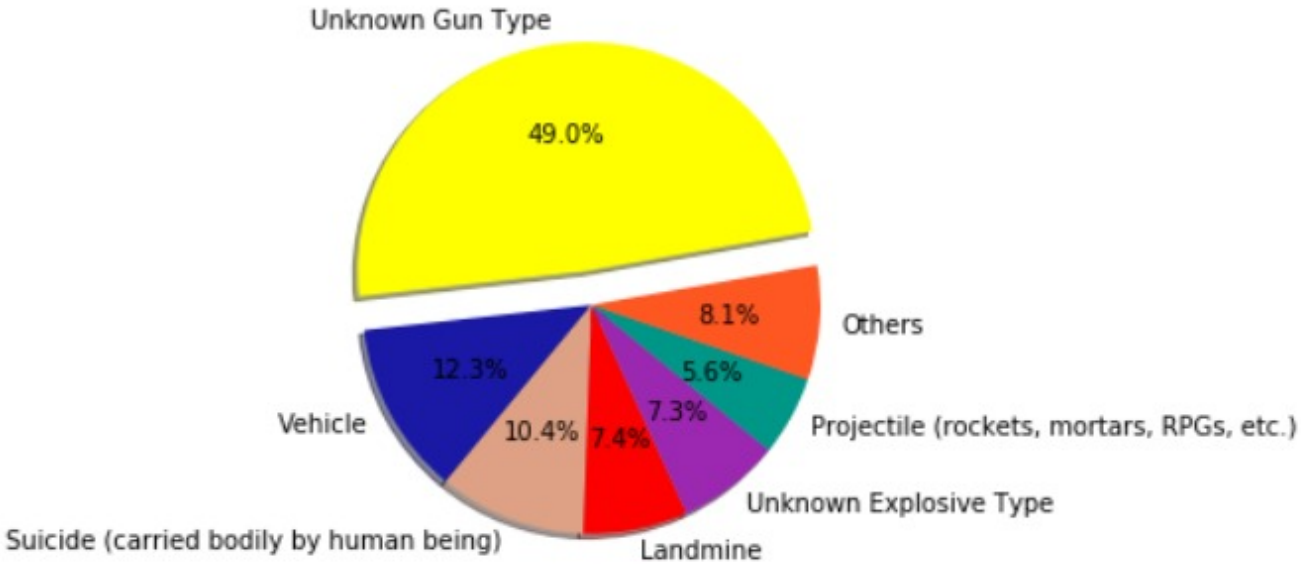
Now, we segregate the data before plotting the piechart.

```
df3TalKill = df3Tal.filter(['weapsubtype1_txt','nkill'], axis=1)
```

```
df3TK=df3TalKill.groupby('weapsubtype1_txt').sum().sort_values('nkill',ascending=False).reset_index()
df3TK
```

	weapsubtype1_txt	nkill
0	Unknown Gun Type	12054.0
1	Vehicle	3015.0
2	Suicide (carried bodily by human being)	2560.0
3	Landmine	1821.0
4	Unknown Explosive Type	1785.0
5	Projectile (rockets, mortars, RPGs, etc.)	1390.0

The resultant pie-chart :



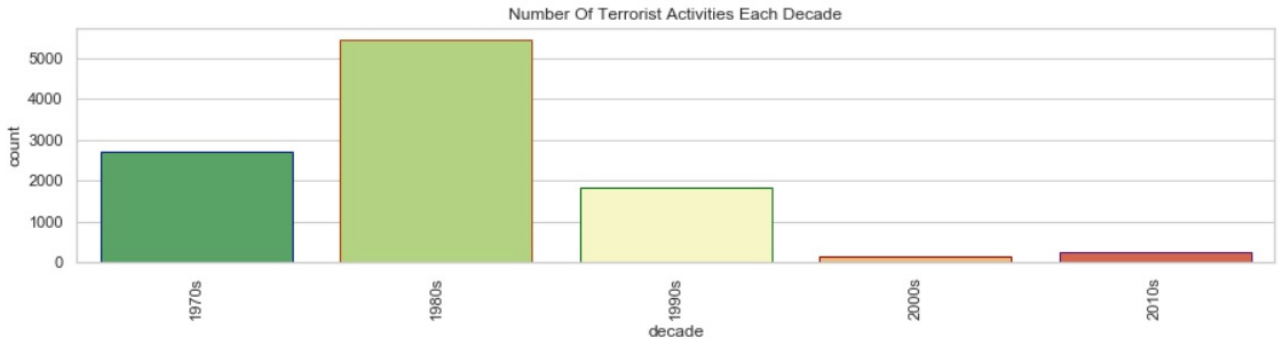
Analysis of Question 4: What is economical loss of each decade and its comparison based on criticality of the attack.

To determine this, we first create the Decade variable like we did in question 1. This is how our data looks after adding decade :

```
df4a
```

	eventid	iyear	country_txt	propvalue	crit1	decade
7	1.970010e+11	1970	United States	22500.0	1	1970s
8	1.970010e+11	1970	United States	60000.0	1	1970s
10	1.970010e+11	1970	United States	0.0	1	1970s
11	1.970010e+11	1970	United States	305.0	1	1970s
14	1.970010e+11	1970	United States	2000000.0	1	1970s
19	1.970010e+11	1970	United States	17000.0	1	1970s

Later we plot a bar graph representing the economical loss of each decade.



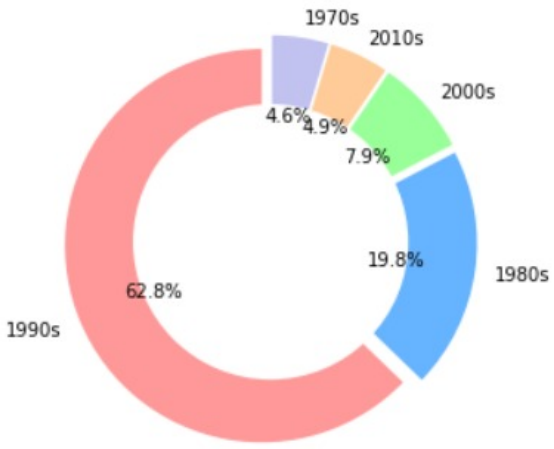
Now, we group-by “decade” sum and sort it using “propvalue” to get a filtered data for decade and property’s value.

```
df4ab=df4a.groupby('decade').sum().sort_values('propvalue',ascending=False).reset_index()
```

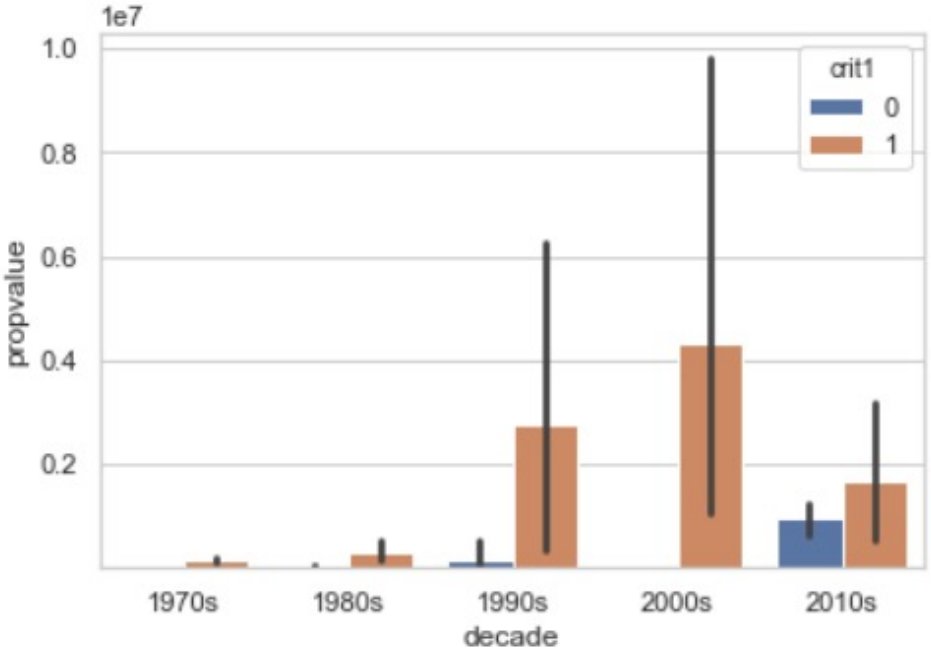
```
df4tot = df4ab.filter(['decade','propvalue'], axis=1)
df4tot
```

	decade	propvalue
0	1990s	5.113051e+09
1	1980s	1.615999e+09
2	2000s	6.401500e+08
3	2010s	3.979782e+08
4	1970s	3.770212e+08

Now, we plot the pie-chart to show us percentage distribution of property loss during each decade. It shows 1990s experienced the most loss in property i.e. 62.8%.



Now, to answer our question, To compare the property damage per decade based on criticality of the attack, we are using “crti1”, which has values 1 for highly critical and 0 for less critical.



With inclusion of criticality, we can see that now 2000s have the most economical loss.

Analysis of Question 5: Regression to see success of terrorist attack

Here we try to determine success rate of the terrorist attacks in various countries.

First, we create dummies to perform logistic regression.

```
df5reg = pd.get_dummies(df5[['success', 'attacktype1_txt', 'targtype1_txt','nkill']], drop_first = True)
```

```
df5reg
```

	success	nkill	attacktype1_txt_Assassination	attacktype1_txt_Bombing/Explosion	attacktype1_txt_Facility/Infrastructure Attack	attacktype1_txt_Hijacking	atta
0	1	1.0	1	0	0	0	
1	1	0.0	0	0	0	0	
2	1	1.0	1	0	0	0	
5	1	0.0	0	0	0	0	
6	0	0.0	1	0	0	0	
7	1	0.0	0	1	0	0	

After performing the regression, the coefficients and intercepts can be seen as below :

```
from sklearn import linear_model

reg = linear_model.LogisticRegression()

results = reg.fit(X = df5reg, y = df5['success'] )
```

```
C:\Users\Shraddha\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:432: FutureWarning: Default solver will be chang
ed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
```

```
print(results.coef_)
```

```
[[15.78232305  0.02006355 -1.03703794 -0.72578015 -0.35251527 -0.14953044
   0.03353308 -0.07237935 -0.2041371  -0.715178   -0.23758543 -0.29322164
  -0.22262372 -0.03269212 -0.41716541 -0.5727952  -0.18999141 -0.05124356
  -0.59875827 -0.0351435  -0.02312874 -0.47874742 -0.30789898 -0.18750798
  -0.03872165 -0.09225434 -0.02750905 -0.33861417 -1.22224108 -0.05875831
  -0.19320954]]
```

```
print(results.intercept_)
```

```
[-5.68156089]
```

Now we calculate the odd ratio to further understand dependency of each variables on success.

```
valuesR = np.append(results.intercept_, results.coef_)

namesR = np.append('intercep', df5reg.columns)

table = pd.DataFrame( valuesR, index = namesR, columns = ['coef'])
table['odd_ratio'] = np.exp(table['coef'])

table
```

	coef	odd_ratio
intercep	-5.681561	3.408234e-03
success	15.782323	7.147856e+06
nkill	0.020064	1.020266e+00
attacktype1_txt_Assassination	-1.037038	3.545032e-01
attacktype1_txt_Bombing/Explosion	-0.725780	4.839469e-01
attacktype1_txt_Facility/Infrastructure Attack	-0.352515	7.029178e-01
attacktype1_txt_Hijacking	-0.149530	8.611122e-01
attacktype1_txt_Hostage Taking (Barricade Incident)	0.033533	1.034102e+00
attacktype1_txt_Hostage Taking (Kidnapping)	-0.072379	9.301780e-01
attacktype1_txt_Unarmed Assault	-0.204137	8.153506e-01
attacktype1_txt_Unknown	-0.715178	4.891050e-01
targtype1_txt_Airports & Aircraft	-0.237585	7.885295e-01
targtype1_txt_Business	-0.293222	7.458568e-01
targtype1_txt_Educational Institution	-0.222624	8.004160e-01
targtype1_txt_Food or Water Supply	-0.032692	9.678365e-01
targtype1_txt_Government (Diplomatic)	-0.417165	6.589119e-01
targtype1_txt_Government (General)	-0.572795	5.639469e-01
targtype1_txt_Journalists & Media	-0.189991	8.269662e-01
targtype1_txt_Maritime	-0.051244	9.500473e-01
targtype1_txt_Military	-0.598758	5.494935e-01
targtype1_txt_NGO	-0.035144	9.654669e-01
targtype1_txt_Other	-0.023129	9.771367e-01
targtype1_txt_Police	-0.478747	6.195590e-01
targtype1_txt_Private Citizens & Property	-0.307899	7.349896e-01
targtype1_txt_Religious Figures/Institutions	-0.187508	8.290225e-01
targtype1_txt_Telecommunication	-0.038722	9.620185e-01
targtype1_txt_Terrorists/Non-State Militia	-0.092254	9.118732e-01
targtype1_txt_Tourists	-0.027509	9.728659e-01
targtype1_txt_Transportation	-0.338614	7.127574e-01
targtype1_txt_Unknown	-1.222241	2.945693e-01
targtype1_txt_Utilities	-0.058758	9.429346e-01
targtype1_txt_Violent Political Party	-0.193210	8.243092e-01

As we know in Logistic r regression, if the response variable increases by x units then the predictor variable will Increases by b1x times where b1 is the coefficient. So if the response variables increases by one unit then out predictor variables variables such as 'attacktype_1' , 'targettype_1' and 'nkill' increases by their respective coefficient amount of times.

As a result we can see that If 'nkill' increases by 1 unit, 'success' increases by 0.025 times.

Now, we perform logistic regression for iyear, success, propvalue, country , crit1 , crit2 , crit3 and nkill.

```
model = smf.logit('success ~ iyear + propvalue + nkill+country+crit1+crit2+crit3', data = dflr).fit()

print(model.summary())
```

```
Optimization terminated successfully.
         Current function value: 0.143748
         Iterations 17
                           Logit Regression Results
==============================================================================
Dep. Variable:                success   No. Observations:                36906
Model:                          Logit   Df Residuals:                    36898
Method:                           MLE   Df Model:                            7
Date:                Fri, 13 Dec 2019   Pseudo R-squ.:                 0.03781
Time:                        22:43:42   Log-Likelihood:                -5305.2
converged:                       True   LL-Null:                       -5513.6
Covariance Type:            nonrobust   LLR p-value:                5.557e-86
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      4.1968      5.740      0.731      0.465      -7.053      15.446
iyear         -0.0012      0.003     -0.435      0.663      -0.007       0.004
propvalue   3.923e-05    4.9e-06      8.005      0.000    2.96e-05    4.88e-05
nkill          0.1492      0.017      9.025      0.000       0.117       0.182
country       -0.0004      0.000     -1.258      0.208      -0.001       0.000
crit1          0.6586      0.216      3.050      0.002       0.235       1.082
crit2          0.9505      0.335      2.838      0.005       0.294       1.607
crit3         -0.2702      0.111     -2.442      0.015      -0.487      -0.053
==============================================================================
```

Now we can see that our odd ratios for variables are :

```
odds_ratio = np.exp(model.params)

odds_ratio
```

```
Intercept    66.473844
iyear         0.998763
propvalue     1.000039
nkill         1.160914
country       0.999648
crit1         1.932022
crit2         2.586992
crit3         0.763264
dtype: float64
```

Prediction

To understand and perform prediction, we will create interacting variables.

We created 2 such variables :

1. Involving crit1 & iyear
2. Involving crit1, crit2 and crit3.

```
#adding interacting variables,
dflr['crit1*iyear'] = dflr.crit1*dflr.iyear
dflr['crit1*crit2*crit3'] = dflr.crit1*dflr.crit2*dflr.crit3

dflr.head()
```

	iyear	success	propvalue	country	crit1	crit2	crit3	nkill	crit1*iyear	crit1*crit2*crit3
7	1970	1	22500.0	217	1	1	1	0.0	1970	1
8	1970	1	60000.0	217	1	1	1	0.0	1970	1
10	1970	0	0.0	217	1	1	0	0.0	1970	0
11	1970	1	305.0	217	1	1	1	0.0	1970	1
14	1970	1	2000000.0	217	1	1	1	0.0	1970	1

Now, we create different models which we will test for regression and performance, the 5 models which we are using are :

m1 = success, nkill, iyear, propvalue, country, crit1, crit3 and crit1*iyear

m2 = success, nkill, iyear, crit1*iyear, country, crit1, crit2 and crit1*crit2*crit3

m3 = success, nkill, iyear, propvalue, country, crit2, crit3

m4 = success, nkill, iyear, propvalue, crit1, crit2, crit3, crit1*iyear and crit1*crit2*crit3

m5 = success, nkill, propvalue, country, crit1, crit2, crit3 and crit1*crit2*crit3

The ANOVA deviance table for our model shows :

```
   df_residuals  resid_stddev  df  deviance
0         36898           NaN   7       NaN
1         36898           NaN   7       NaN
2         36899           NaN   6       NaN
3         36898           NaN   7       NaN
4         36899           NaN   6       NaN
```

Now, we create predictor and response matrices,

```
#creating predictor and response matrices
y1,X1 = dmatrices('success ~ nkill + iyear + propvalue + country + crit1 + crit3+crit1*iyear',dflr)
y2,X2 = dmatrices('success ~ nkill + iyear + crit1*iyear + country + crit1 + crit2+crit1*crit2*crit3',dflr)
y3,X3 = dmatrices('success ~ nkill + iyear + propvalue + country + crit2 + crit3',dflr)
y4,X4 = dmatrices('success ~ nkill + iyear + propvalue + crit1 + crit2 + crit3+crit1*iyear+crit1*crit2*crit3',dflr)
y5,X5 = dmatrices('success ~ nkill + propvalue + country + crit1 + crit2 + crit3+crit1*crit2*crit3',dflr)
```

apply the fitting model :

```
lr = linear_model.LogisticRegression()

s1 = cross_val_score(lr, X1, y1, cv=5)
s2 = cross_val_score(lr, X2, y2, cv=5)
s3 = cross_val_score(lr, X3, y3, cv=5)
s4 = cross_val_score(lr, X4, y4, cv=5)
s5 = cross_val_score(lr, X5, y5, cv=5)
```

Now the accuracy for each model comes out to be following :

For M1 : 0.965697

For M2 : 0.965697

For M3 : 0.965697

For M4 : 0.965697 &

For M5 : 0.965697

Now, to improve the accuracy of our model we regularise our model on basis of logistic regression

```
response, predictors = dmatrices('success ~ iyear + propvalue + nkill+country+crit1+crit2+crit3+ crit1*crit2*crit3',
                                 data = dflr)
```

```
X_train, X_test, y_train, y_test = train_test_split(predictors,response,random_state=0)

lr = linear_model.LogisticRegression()

logmodel = lr.fit(X_train, y_train)
```

```
C:\Users\Shraddha\Anaconda3\lib\site-packages\sklearn\utils\validation.py:724: DataConversionWarning: A column-vector y was pas
sed when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

After applying the regression, our model shows increase in the efficiency to 0.9665113254578953. This can be seen from our model's accuracy for it's test & training data as follow :

```
#Accuracy of training dataset
print(lr.score(X_train,y_train))
```

```
0.9654250514830738
```

```
#Accuracy of testing dataset

print(lr.score(X_test,y_test))
```

```
0.9665113254578953
```

CONCLUSION

This project has **helped** us analyse the global terrorism acts all around the world. It helped us learn efficient ways to perform data cleaning and knowledge on how to segregate and filter out data. It made us learn how to differentiate and determine crucial data required for analysing from the bulk of raw data.

All 5 of our models achieved an accuracy of **0.965697.**

No much change even after regularisation of the modelsIt provided better understanding of data manipulation using multiple functions and methods which gave an expected appropriate result.

We applied statistical methods to data such as linear regression which helped us in processing relation between variable as how one variable change would affect multiple parameters in data and modify it.

CHALLENGES

Major challenges we faced were related to picking what all we require from the data. We spent around 2 days in finding the right approach to select proper data for our questions. We were losing out crucial datas while performing the basic data cleaning processes. So we cleaned our dataset for each questions separately with only whatever data was required.

Apart from that, we had to create columns wherever necessary. Sometimes operate only with 40,000 data entries out of 1,80,000+ entries.

Segregation was also an important factor as at places it was hard for us to analyse the plotted data because of abundance of data entries.

THE TEAM

Mukul Pathak

Shraddha Upadhyay

Mansa Veeramachineni

Disha Shah