Global Terrorism Analysis

A Data Analysis Report

By Ahmed Solayman – ITI Data Engineering Track 2024

Introduction

This project is a part of ITI data engineering track 2024.

In the next few pages, you'll find some interesting insights on how terrorism spread all over the world from 1970 to 2017.

Data Exploration

Performance comparison between Pandas and Dask

In this section I compare pandas and Dask libraries in loading data in terms of memory usage and time consumption.

Pandas:

```
# Load dataset using Pandas
start_time = time.time()
df = pd.read_csv(file_path, encoding=result['encoding'])
pandas_load_time = time.time() - start_time
pandas_memory_usage = get_memory_usage()
```

Dask:

```
# Load dataset using Dask
start_time = time.time()
dask_df = dd.read_csv(file_path, encoding=result['encoding'])
dask_load_time = time.time() - start_time
dask_memory_usage = get_memory_usage()
```

\

Results:

```
Pandas Load Time: 2.3142383098602295 seconds
Pandas Memory Usage: 594.4921875 MB
Dask Load Time: 0.039183378219604449 seconds
Dask Memory Usage: 597.640625 MB
```

It’s clear that Dask is far superior in terms of loading the data, but the two libraries are almost the same in terms of memory usage.

Let’s dive into the dataset!

Choosing needed columns only

```
Index(['eventid', 'iyear', 'imonth', 'iday', 'approxdate', 'extended',
       'resolution', 'country', 'country_txt', 'region',
       ...
       'addnotes', 'scite1', 'scite2', 'scite3', 'dbsource', 'INT_LOG',
       'INT_IDEO', 'INT_MISC', 'INT_ANY', 'related'],
      dtype='object', length=135)
```

```python
# Choosing only the columns that are needed

df1 = df[
    ['iyear', 'imonth', 'iday', 'country_txt', 'extended',
     'success', 'suicide' ,'region_txt', 'city', 'latitude',
     'longitude', 'attacktype1_txt', 'targtype1_txt', 'natlty1_txt',
     'gname','nkill', 'nwound', 'weaptype1_txt']
     ]
```

```
Index(['iyear', 'imonth', 'iday', 'country_txt', 'extended', 'success',
       'suicide', 'region_txt', 'city', 'latitude', 'longitude',
       'attacktype1_txt', 'targtype1_txt', 'natlty1_txt', 'gname', 'nkill',
       'nwound', 'weaptype1_txt'],
      dtype='object') (181691, 18)
```

	iyear	imonth	iday	country_txt	extended	success	suicide	region_txt	city	latitude	longitude	attacktype1_txt	targtype1_txt	natlty1_txt	gname	nkill	nwound	weapty
0	1970	7	2	Dominican Republic	0	1	0	Central America & Caribbean	Santo Domingo	18.456792	-69.951164	Assassination	Private Citizens & Property	Dominican Republic	MANO-D	1.0	0.0	U
1	1970	0	0	Mexico	0	1	0	North America	Mexico city	19.371887	-99.086624	Hostage Taking (Kidnapping)	Government (Diplomatic)	Belgium	23rd of September Communist League	0.0	0.0	U
2	1970	1	0	Philippines	0	1	0	Southeast Asia	Unknown	15.478598	120.599741	Assassination	Journalists & Media	United States	Unknown	1.0	0.0	U
3	1970	1	0	Greece	0	1	0	Western Europe	Athens	37.997490	23.762728	Bombing/Explosion	Government (Diplomatic)	United States	Unknown	NaN	NaN	E
4	1970	1	0	Japan	0	1	0	East Asia	Fukouka	33.580412	130.396361	Facility/Infrastructure Attack	Government (Diplomatic)	United States	Unknown	NaN	NaN	In
5	1970	1	1	United States	0	1	0	North America	Cairo	37.005105	-89.176269	Armed Assault	Police	United States	Black Nationalists	0.0	0.0	

Handling Nulls and duplicates

I looked up the percentage of null values in the data frame, I found that the null values were below 10% which leaded to the decision to drop all null values.

```
iyear             0.000000
imonth            0.000000
iday              0.000000
country_txt       0.000000
extended          0.000000
success           0.000000
suicide           0.000000
region_txt        0.000000
city              0.239417
latitude          2.507554
longitude         2.508104
attacktype1_txt   0.000000
targtype1_txt     0.000000
natlty1_txt       0.858050
gname             0.000000
nkill             5.676120
nwound            8.977330
weaptype1_txt     0.000000
dtype: float64
```

```python
df2 = df1.dropna()
(df2.isnull().sum()/len(df2))*100
df2.reset_index(drop=True, inplace=True)
```

Dropping the duplicate rows was the next thing to do.

```python
df2 = df2.drop_duplicates()
df2.reset_index(drop=True, inplace=True)
```

Handing Unknown Values

My next approach was to check all unique values in each column to determine if there were any misleading or unknown values

```
iyear [1970 1971 1972 1973 1974 1975 1976 1977 1978 1979 1980 1981 1986 1982
 1983 1984 1985 1987 1988 1989 1990 1991 1992 1994 1995 1996 1997 1998
 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012
 2013 2014 2015 2016 2017]
imonth [ 7  0  1  2  3  4  5  6  8  9 10 11 12]
iday [ 2  0  1  3  6  8  9 12 13 14 15 19 20 21 22 25 26 27 28 30 31  4  7 11
 16 17 18 23 24  5 10 29]
country_txt ['Dominican Republic' 'Mexico' 'Philippines' 'United States' 'Uruguay'
 'Italy' 'Guatemala' 'Venezuela' 'West Germany (FRG)' 'Switzerland'
 'Brazil' 'Egypt' 'Argentina' 'Lebanon' 'Japan' 'Jordan' 'Turkey'
 'Paraguay' 'East Germany (GDR)' 'United Kingdom' 'Greece' 'Nicaragua'
 'Belgium' 'Netherlands' 'Canada' 'Iran' 'Australia' 'Pakistan' 'Spain'
 'Ethiopia' 'Sweden' 'South Yemen' 'Cambodia' 'Israel' 'Poland' 'Panama'
 'West Bank and Gaza Strip' 'Ireland' 'India' 'Austria' 'France'
 'South Vietnam' 'Colombia' 'Brunei' 'Zaire'
 "People's Republic of the Congo" 'Portugal' 'Algeria' 'El Salvador'
 'Thailand' 'Haiti' 'Morocco' 'Cyprus' 'Afghanistan' 'Peru' 'Chile'
 'Yugoslavia' 'Ecuador' 'New Zealand' 'Zambia' 'Malaysia' 'Bolivia'
 'Singapore' 'Botswana' 'Kuwait' 'Jamaica' 'Chad' 'North Yemen' 'Syria'
 'South Korea' 'United Arab Emirates' 'South Africa' 'Kenya' 'Iraq'
 'Somalia' 'Sri Lanka' 'Namibia' 'Bahamas' 'Nigeria' 'Barbados'
 'Costa Rica' 'Taiwan' 'Bangladesh' 'Mauritania' 'Djibouti' 'Indonesia'
 'Rhodesia' 'Soviet Union' 'Angola' 'Guyana' 'Mozambique' 'Myanmar'
 'Tunisia' 'Denmark' 'Uganda' 'Honduras' 'Norway' 'Lesotho' 'Tanzania'
 'Gabon' 'Libya' 'Trinidad and Tobago' 'Saudi Arabia' 'Bahrain'
...
weaptype1_txt ['Unknown' 'Firearms' 'Explosives' 'Incendiary' 'Chemical' 'Melee'
 'Sabotage Equipment'
 'Vehicle (not to include vehicle-borne explosives, i.e., car or truck bombs)'
 'Fake Weapons' 'Radiological' 'Other' 'Biological']
```

As per the below snapshot, there were a decent number of “Unknown” values, so I decided to drop these values if their percentage is less than 30%.

```
Column 'city' contains 5380 'Unknown' values out of 147850
Column 'attacktype1_txt' contains 5409 'Unknown' values out of 147850
Column 'targtype1_txt' contains 3965 'Unknown' values out of 147850
Column 'gname' contains 71799 'Unknown' values out of 147850
Column 'weaptype1_txt' contains 10640 'Unknown' values out of 147850
Cleaned DataFrame shape: (128838, 18)
```

Handling date columns

```
Number of zeroes in 'imonth' column: 6
Number of zeroes in 'iday' column: 385
```

```
'''
to decide whether we're keeping or dropping the zero values in iday and imonth
columns, Iwe need to check the som of the columns where day or month has zero
in respect with the approx date (we need to use the original df)

'''
```

```
'''
Decisions:
drop the iday column as we don't need the exact day of the attack
fill the zero values in imonth column with the mode of the column
'''
```

Data type Conversions

```
# Convert 'latitude' and 'longitude' to string type

df2['latitude'] = df2['latitude'].astype(str)
df2['longitude'] = df2['longitude'].astype(str)
```

```
# Convert 'nkill' and 'nwound' to integer type
df2['nkill'] = df2['nkill'].fillna(0).astype('int64')
df2['nwound'] = df2['nwound'].fillna(0).astype('int64')

print(df2.dtypes)
```

```
iyear               int64
imonth              int64
country_txt        object
extended            int64
success             int64
suicide             int64
region_txt         object
city               object
latitude           object
longitude          object
attacktype1_txt    object
targtype1_txt      object
natlty1_txt        object
gname              object
nkill               int64
nwound              int64
weaptype1_txt      object
dtype: object
```

Cleaned Dataset Creation

In this section I renamed some of the columns to be more descriptive and then exported the cleaned dataset as csv file.

```
Index(['iyear', 'imonth', 'country_name', 'extended', 'success', 'suicide',
       'region', 'city', 'latitude', 'longitude', 'attack_type', 'target_type',
       'nationality_of_target', 'group_name', 'number_of_kills',
       'number_of_wounds', 'weapon_type'],
      dtype='object')
```

```python
df2.shape
```

```
(128838, 17)
```

```python
df2.to_csv('cleaned_globalterrorismdb_0718dist.csv', index=False)
```

Python

```python
df2.head(10)
```

Python

	iyear	imonth	country_name	extended	success	suicide	region	city	latitude	longitude	attack_type	target_type	nationality_of_target	group_name	number_of_kills	nu
0	1970	1	United States	0	1	0	North America	Cairo	37.005105	-89.176269	Armed Assault	Police	United States	Black Nationalists	0	
1	1970	1	Uruguay	0	0	0	South America	Montevideo	-34.891151	-56.187214	Assassination	Police	Uruguay	Tupamaros (Uruguay)	0	
2	1970	1	United States	0	1	0	North America	Oakland	37.791927	-122.225906	Bombing/Explosion	Utilities	United States	Unknown	0	
3	1970	1	United States	0	1	0	North America	Madison	43.076592	-89.412488	Facility/Infrastructure Attack	Military	United States	New Year's Gang	0	
4	1970	1	United States	0	1	0	North America	Madison	43.07295	-89.386694	Facility/Infrastructure Attack	Government (General)	United States	New Year's Gang	0	
5	1970	1	United States	0	0	0	North America	Baraboo	43.4685	-89.744299	Bombing/Explosion	Military	United States	Weather Underground, Weathermen	0	

Data Analysis and Visualization

Numerical columns basic statistics

	extended	success	suicide	number_of_kills	number_of_wounds
count	128838.000000	128838.000000	128838.000000	128838.000000	128838.000000
mean	0.022734	0.900255	0.042713	2.215177	3.706515
std	0.149055	0.299661	0.202209	10.273461	40.576605
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	1.000000	0.000000	0.000000	0.000000
50%	0.000000	1.000000	0.000000	0.000000	0.000000
75%	0.000000	1.000000	0.000000	2.000000	2.000000
max	1.000000	1.000000	1.000000	1384.000000	8191.000000

```
'Mean values'
number_of_kills     2.215177
number_of_wounds    3.706515
dtype: float64
'Median values'
array([0., 0.])
'Standard Deviation values'
number_of_kills     10.273422
number_of_wounds    40.576448
dtype: float64
```

In summary, the average kill count per incident is more than 2, and the wound count is more than 3.

So, on average, terrorism incidents lead to around 5 casualties per attack, a disappointment of the world we live in.

Also, the spread of the counts from average is rather big, meaning that there were incidents with huge difference for the average 2 kills and 5 wounds.

The median is zero as there’s a large number of zero values.

Categorical columns most frequent values

```
'Most frequent values in categorical columns:'
{'attack_type': 'Bombing/Explosion',
 'city': 'Baghdad',
 'country_name': 'Iraq',
 'group_name': 'Unknown',
 'imonth': '5',
 'iyear': '2014',
 'latitude': '33.303566',
 'longitude': '44.371773',
 'nationality_of_target': 'Iraq',
 'region': 'Middle East & North Africa',
 'target_type': 'Private Citizens & Property',
 'weapon_type': 'Explosives'}
```

With this it's clear that North Africa and The Middle east (Mostly Iraq) were heavily targeted by terrorism. Understandable of course, due to the waging wars that sprung in the last 50 years.

With the most group name appearing is Unknown (anonymous attacker), I looked for the second most responsible group. As Iraq is the highest country bleeding from terrorism, 'Taliban' were the second highest group.

```
Second most frequent value in the 'group_name' column: Taliban
```

The most common attack type is bombing by far, makes total sense!

```
'Most common attack types:'

attack_type
Bombing/Explosion                   66966
Armed Assault                       33443
Assassination                       14967
Facility/Infrastructure Attack       7456
Hostage Taking (Kidnapping)          4215
Name: count, dtype: int64
```

What has happened in 2014?

The highest year with number of attacks is 2014, I looked up online what happened in that year and I found that in 2014, according to a study U.S. Dept of state, the deadliest terrorist attack since 9/11 2001 happened in Iraq. Along with a very deadly "fighting season" in Afghanistan and some incidents around Syria and Somalia as well.

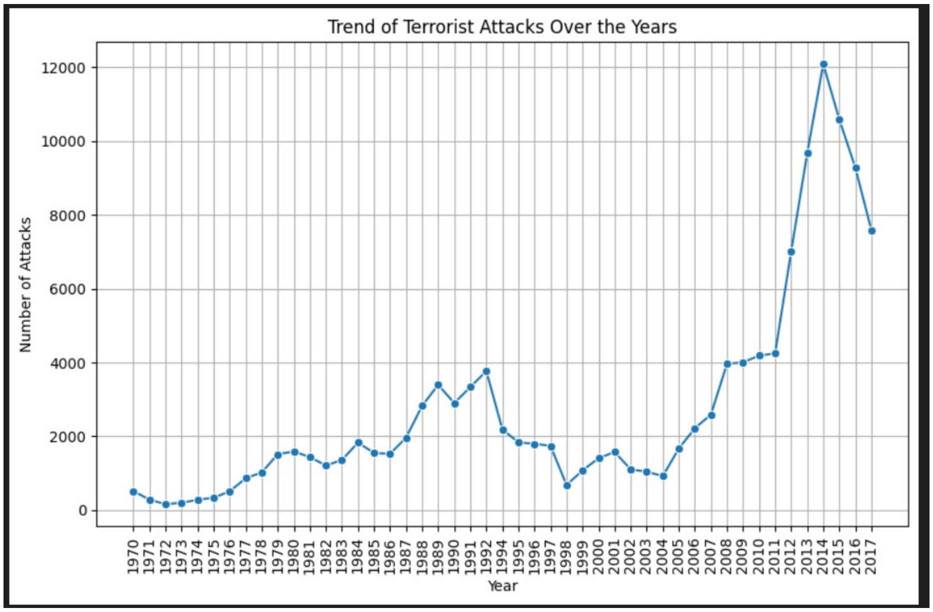
A very sad year for the middle east!

```
Top five years with the highest number of attacks:
iyear
2014    12093
2015    10591
2013     9673
2016     9283
2017     7589
Name: count, dtype: int64
```

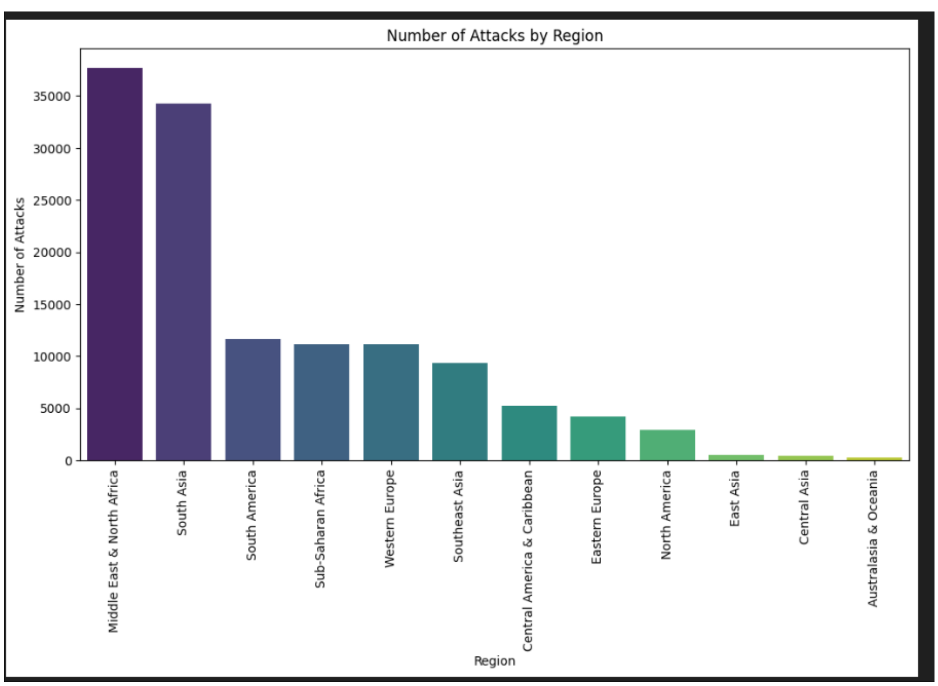
- The months with the most terrorist attacks and combined casualties (deaths and injuries) were May, June, and July.
    - In particular, the high number of attacks in May coincides with the peak of spring "fighting season" in Afghanistan, where attacks increased more than 107% between February and May.
    - Contributing to the high number of fatalities in June, the Islamic State of Iraq and the Levant (ISIL) carried out an attack on Badush prison in Mosul, Iraq on June 10, 2014, which resulted in the deaths of 670 Shia prisoners. As of the end of 2014, this was the deadliest terrorist attack worldwide since September 11, 2001.
    - Also in June, there were five attacks in which more than 50 people were kidnapped. Three took place in Iraq, one in Somalia, and one in Syria. In August, four attacks (three in Iraq and one in Nigeria) involved the abduction of more than 50 people.
    - The exceptionally high number of hostages reported in December is largely a result of the attack on the Army Public School in Peshawar, Pakistan. Assailants from Tehrik-i-Taliban Pakistan held more than 500 individuals hostage during a siege that killed at least 150 people.
- More than 6,200 of the 32,700 people killed in 2014 (19%) were perpetrators of terrorist attacks. Perpetrators were killed intentionally in suicide attacks, accidentally while attempting to carry out attacks, or by security forces or victims responding to attacks.

More insights

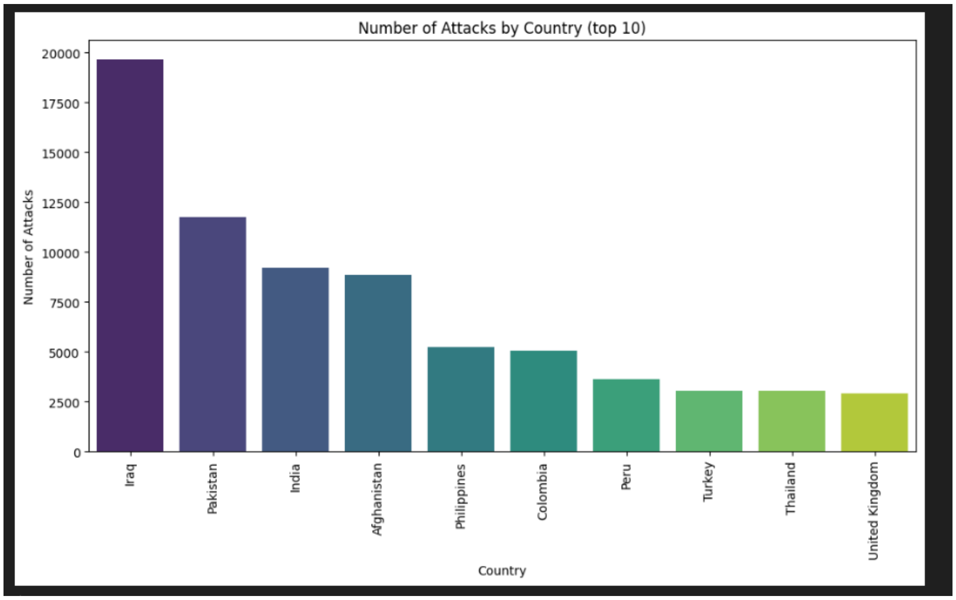
The trend shows a very disturbing rise of terrorist attacks in the first quarter of the 21st century, with a peak number of attacks happening in 2014 as previously stated.



Most affected regions were Africa, Middle east and South Asian by far, and the least is Australia.

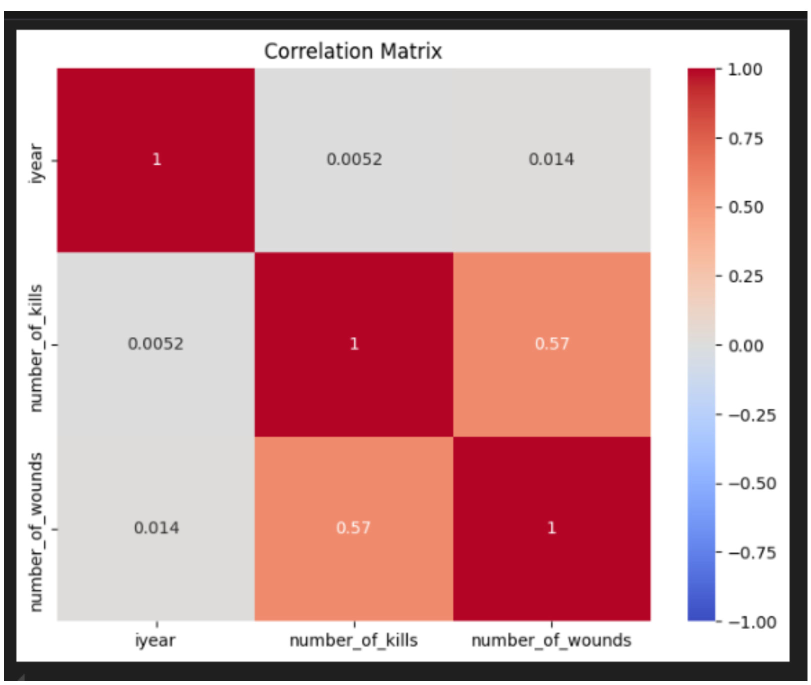


Expectdly, the highest 10 countries with the highest number of attacks were mostly from the middle east and South Asia.

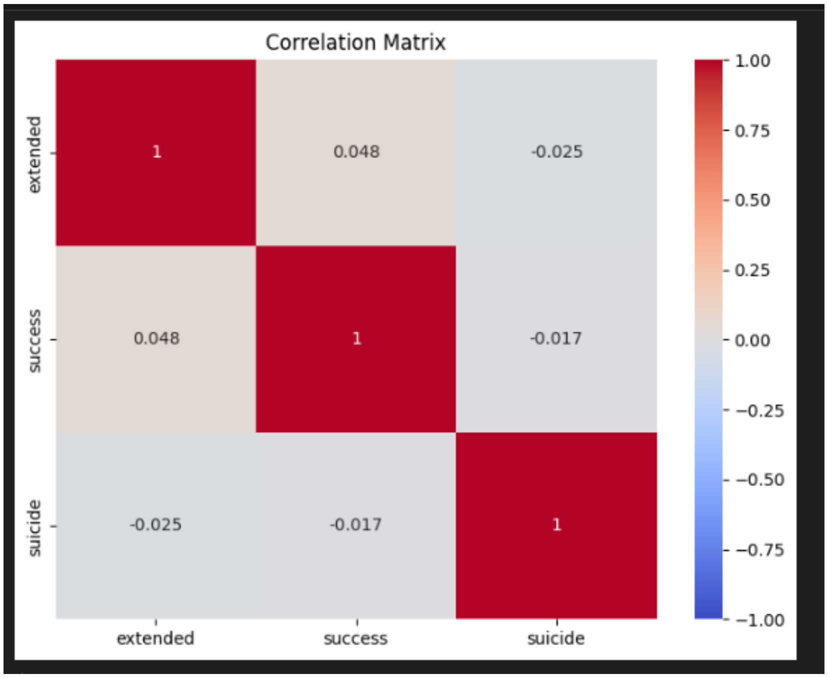


Correlations and Relationships

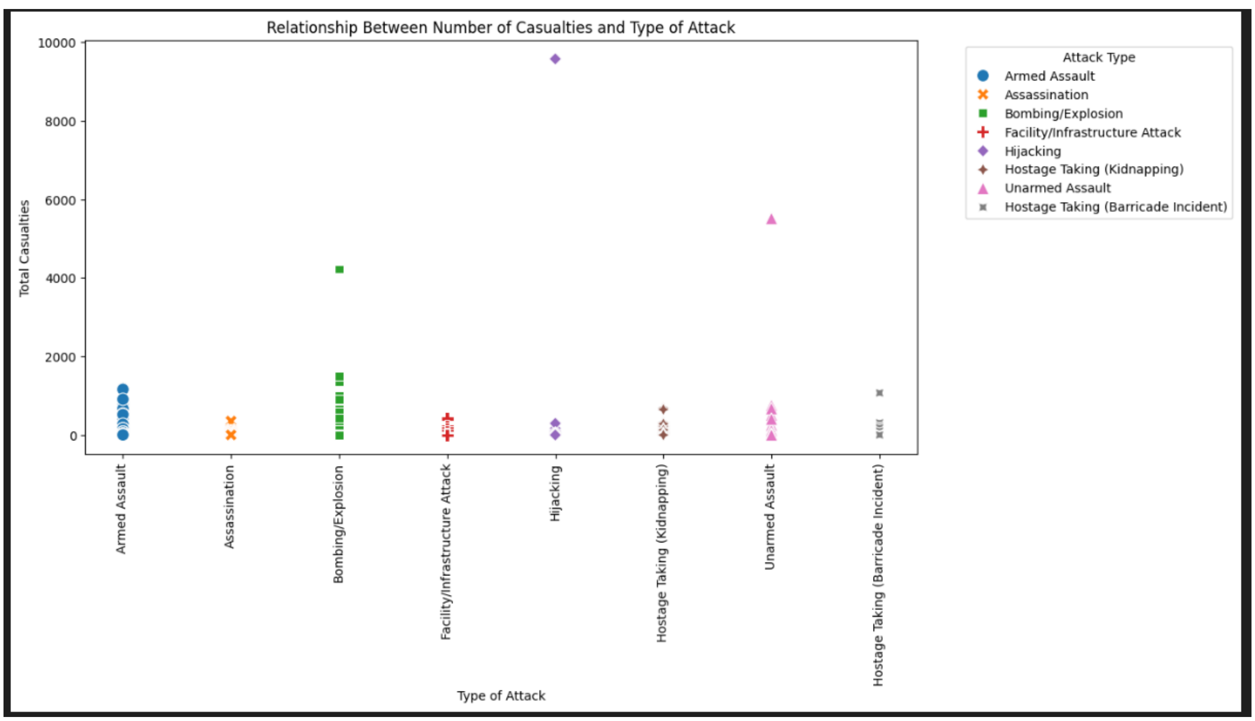
There was a positive correlation between the number of kills and the number of wounds and almost no correlation with the year!



Also, there's almost no correlation between commiting suicide, the success of the attack, and the extnsion of it.



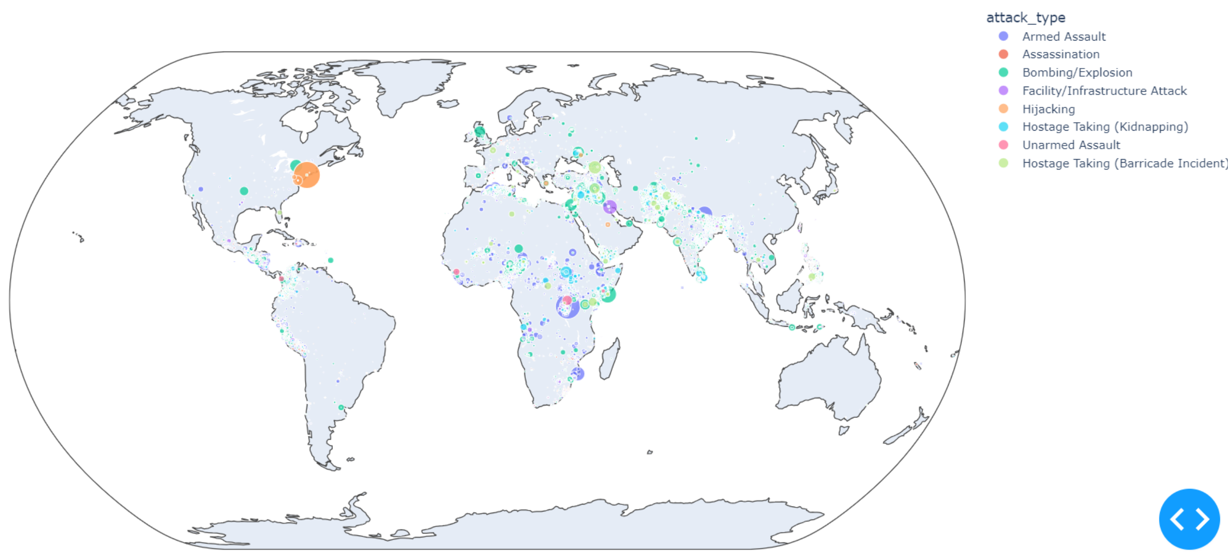
The figure below shows the spread of number of casualties (kills + wounds) and the type of attack.



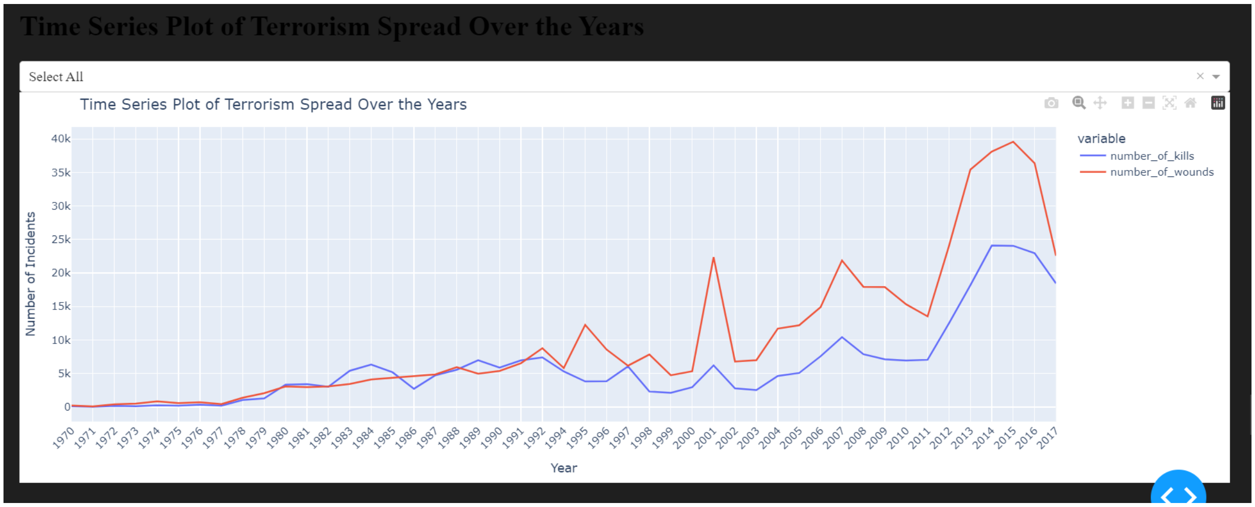
Spread of terrorism over time and land

Egypt had its share of terrorism attacks over the years, with a peak number of kills = 311. This happened in 2017 in the Sinai Mosque dreadful incident.

The distribution of attacks all over the world in over the years.



The spread of number of kills and wounds over the years



This supports the correlation that was found in the correlation section.\

Conclusion

The 21st century was a very intense in terms of terrorism, the disease that sprung from hate, extremism and the pursuit of power!

With these types of analyses, we can look for patterns and trends that can help us identify the characteristics of terrorism in the hope that we can reduce its effect.