-*- coding: utf-8 -*-

"""Global Terrorism Database EDA .ipynb

Automatically generated by Colab.

Original file is located at

https://colab.research.google.com/drive/1S6tG_UZcjLW7FcN3obxKKPewqtQMqiP8

Project Name - Global terrorism dataset EDA

Project Type - EDA

Contribution - Kshitij Dubey

Project Summary -

Terrorism poses an enduring global challenge, transcending borders, cultures, and ideologies. The Global Terrorism Analysis project was initiated to unravel the complexities of this menace and formulate effective countermeasures. The project delved into a comprehensive exploration of data, employing various analyses to identify factors contributing to terrorism globally, methodologies used in specific regions, and the devastating consequences.

The project commenced by identifying key variables and cleaning the data. Essential features, contributing significantly to terrorism patterns, were carefully selected. A "casualties" column was created by combining killed and wounded to provide a nuanced understanding of the human toll of these incidents.

Impact variables, such as the number of casualties, suicide attacks, successful attacks, and overall attack count, were identified for studying the factors contributing to terrorism. The analytical approach included univariate, bivariate, and multivariate analyses using visualization tools like line plots, bar plots, combined line-bar charts, pie charts, and map visualizations.

The comprehensive analysis began with a global overview to understand broad terrorism trends across years, regions, nations, terrorist organizations, attack and weapon types, and targets. Local Analysis focused on nations and organizations with high terrorist activity, particularly Iraq, Afghanistan, Pakistan, India, Taliban, and ISIL.

The project identified patterns and trends, revealing that terrorism disproportionately affects developing and underdeveloped countries. A surge in attacks occurred in the 21st century in the Middle East, South Asia, and Sub-Saharan Africa, with preferred methods being bombings and explosions, followed by firearms. Private citizens and properties were targeted more than military and government institutions.

The analysis of specific terrorist organizations, such as Taliban and ISIL, highlighted their attack methods, geographical regions of interest, and specific targets. Conclusions emphasized the need for proactive prevention strategies based on the project's insights, ranging from identifying high-risk regions to understanding organizational structures.

In summary, the Global Terrorism Analysis project aims to foster a safer global environment by comprehending the factors contributing to terrorism. Its data-driven insights empower policymakers and security agencies to design proactive strategies, providing a roadmap for prevention rather than reactive response.

GitHub Link -

Provide your GitHub Link here.

Problem Statement

The Global Terrorism Analysis project seeks to address the escalating global threat of terrorism by providing actionable intelligence through comprehensive analysis of historical data, enabling informed decision-making, and fostering international collaboration for the formulation and implementation of effective prevention and mitigation strategies.

Define Your Business Objective?

Embark on a multifaceted exploration to cultivate an exhaustive comprehension of global terrorism dynamics. This objective involves conducting in-depth data analysis and exploration, delving into intricate details to uncover not only the patterns exhibited by terrorist activities but also the underlying drivers and contributing factors that fuel their emergence. Through rigorous examination of historical data, the aim is to discern nuanced trends, recognize evolving patterns, and identify the intricate web of factors that influence the occurrence and escalation of terrorism on a global scale. This comprehensive understanding serves as the bedrock for informed decision-making, enabling stakeholders to grasp the complexities of the threat landscape and devise targeted strategies to effectively mitigate and counteract terrorism.

Let's Begin !

1. Know Your Data

Import Libraries
"""

# Data Download
import gdown

# Data Manipulation and Analysis
import pandas as pd

```
import numpy as np

# Visualization
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.graph_objects as go  # Added Plotly
import plotly.express as px  # Added Plotly
import folium
from folium.plugins import HeatMap

# Missing Data Visualization
import missingno as msno

# Text and Number Formatting
import inflect

# Geospatial Data
import geopandas as gpd

# Web Requests
import requests

# Google Colab Integration
from google.colab import drive

# Suppress Warnings
```

```
import warnings

warnings.filterwarnings('ignore')


"""### Dataset Loading"""


# Load Dataset

drive.mount('/content/drive')


# Loading the file with a different encoding

file_path = '/content/drive/MyDrive/Colab Notebooks/Almabetter/Capstone Projects /EDA on GTD/Global Terrorism Data.csv'

df = pd.read_csv(file_path, encoding='ISO-8859-1')


"""### Dataset First View"""


# Dataset First Look

df.head()


"""### Dataset Rows & Columns count"""


# Dataset Rows & Columns count

print(f'Number of rows in the dataset: {df.shape[0]}')

print(f'Number of columns  in the dataset: {df.shape[1]}')


"""### Dataset Information"""
```

```
# Dataset Info
df.info(verbose = True, show_counts = True)


"""#### Duplicate Values"""


# Dataset Duplicate Value Count
print(f'Duplicated rows in the dataset: {df.duplicated().sum()}')


"""#### Missing Values/Null Values"""


# Calculate and display the total number of missing values in the dataset
total_missing_values = df.isna().sum().sum()
print(f'There are {total_missing_values} missing values in the dataset\n')


# Identify and display columns with missing values
columns_with_missing_values = df.isna().sum()
missing_values_per_column = columns_with_missing_values[columns_with_missing_values !=
0]
print('Columns with missing values:')
print(missing_values_per_column)


# Calculate the number of missing values for each column
missing_values_count = df.isnull().sum()


# Filter columns with missing values (count > 0)
missing_values_count = missing_values_count[missing_values_count > 0]
```

```
# Sort the missing values in descending order
missing_values_count = missing_values_count.sort_values(ascending=False)

# Create a bar chart using Plotly
fig = go.Figure()

fig.add_trace(go.Bar(
    x=missing_values_count.index,
    y=missing_values_count.values,
    marker_color='skyblue'
))

# Update layout for appearance
fig.update_layout(
    title='Number of Missing Values per Column',
    xaxis_title='Columns',
    yaxis_title='Number of Missing Values',
    xaxis=dict(showline=True, showgrid=False, showticklabels=True, linecolor='black',
linewidth=2, tickfont=dict(size=8)),
    yaxis=dict(showline=True, showgrid=False, showticklabels=True, linecolor='black',
linewidth=2, tickfont=dict(size=10)),
    template="simple_white",
    showlegend=False
)

fig.show()
```

"""### What did you know about your dataset?

Upon initial inspection of the dataset, the following observations are made:

* The dataset comprises 181,691 rows and 135 columns. Notably, one of these

columns is "eventid," serving as the primary key for the dataset.

* No duplicate data is present in the dataset, and all values are unique.

* Some columns exhibit missing values, primarily stemming from markers or counters, which may result in null values when not applicable.

* Due to the dataset's extensive width, a strategic approach will be employed for Data Analysis. This involves selecting only essential columns based on intelligent criteria. Null values will be addressed within this subset of chosen features.

2. Understanding Your Variables

"""

```
# Dataset Columns
for col in df.columns:
    print(col)
```

```
# Dataset Describe
df.describe(include='all')
```

```
"""As we can see, there are a total of 135 columns present in the dataset. For further analysis, we will be focusing only on those columns that have more than 80% of the data present in them."
"""

# Calculate the percentage of missing values for each column
missing_percentage = (df.isnull().sum() / len(df)) * 100

# Select columns with less than or equal to 20% missing data
selected_columns = missing_percentage[missing_percentage <= 20].index

#Printing the columns
selected_columns

# Define the necessary columns for Exploratory Data Analysis (EDA)
selected_columns = [
    'eventid', 'iyear', 'imonth', 'iday', 'country_txt', 'region_txt', 'provstate',
    'city', 'latitude', 'longitude', 'success', 'suicide', 'attacktype1_txt',
    'targtype1_txt', 'gname', 'weaptype1_txt', 'nkill', 'nwound', 'summary', 'motive'
]

# Create a new DataFrame with only the selected columns
eda_df = df[selected_columns].copy()

# Rename columns appropriately
column_rename_mapping = {
    'iyear': 'year', 'imonth': 'month', 'iday': 'day', 'country_txt': 'country',
```

```
    'region_txt': 'region', 'gname': 'group', 'provstate': 'state',
    'attacktype1_txt': 'attack_type', 'targtype1_txt': 'target',
    'weaptype1_txt': 'weapon_type', 'nkill': 'killed', 'nwound': 'wounded'
}

eda_df.rename(columns=column_rename_mapping, inplace=True)

"""Among the columns selected earlier, the following are reserved for reference purposes or for studying specific cases. They are not the primary focus of the Exploratory Data Analysis (EDA):

- month
- day
- provstate
- city
- latitude
- longitude
- summary
- motive
"""

# Calculate the total number of missing values in the new dataset
total_missing_values = eda_df.isna().sum().sum()
print(f'There are {total_missing_values} missing values in the new dataset.\n')

# Display columns with missing values and their respective counts
missing_values_by_column = eda_df.isna().sum()
```

```
columns_with_missing_values = missing_values_by_column[missing_values_by_column != 0]

if not columns_with_missing_values.empty:
   print('The columns which have missing values are:')
   print(columns_with_missing_values)
else:
   print('No columns have missing values.')

# Display the number of columns in the dataset and list their names
print(f'The dataset contains {len(eda_df.columns)} columns. Here are the column names:')
print(*eda_df.columns, sep='\n')

# Dataset Describe for the modified dataset which will be used for EDA
eda_df.describe()
```

"""The descriptive statistics for the selected columns reveal interesting patterns in the dataset. Notably, the dataset spans from 1970 to 2017, with various incidents recorded worldwide. The mean latitude and longitude suggest a global distribution of events. The majority of incidents have low casualty counts, as indicated by the relatively low mean values for killed and wounded. Additionally, the success rate of attacks is predominantly high, with a mean close to 0.89. However, it's important to note the wide range of values, as seen in the maximum counts for killed and wounded. Overall, these statistics provide a snapshot of the dataset's temporal and spatial distribution, as well as insights into the severity of incidents.

Variables Description

The Global Terrorism Dataset (GTD) encompasses comprehensive information on approximately 200,000 terrorist events worldwide, spanning numerous years. In this analysis, we focus on a subset of key features, each offering unique insights into the nature and characteristics of these incidents. The selected variables for our Exploratory Data Analysis (EDA) provide a nuanced understanding of the dataset, covering aspects such as temporal details, geographical coordinates, attack success rates, suicide incidents, attack types, target types, perpetrating groups, weapon types, and casualty figures. All the info is taken from the [this](https://drive.google.com/file/d/1gqw8DxWLAmQ4u3GbSvDWrrRjh9IPt0IV/view?usp=drive_link) document provided by University of Maryland's [site](https://www.start.umd.edu/gtd/) for GTD

The following are **"primary variables"**, i.e., variables which are primarily being used for the EDA.

* Eventid

>Numeric Variable

Incidents from the GTD follow a 12-digit Event ID system. •

First 8 numbers – date recorded “yyyymmdd”. •

Last 4 numbers – sequential case number for the given day (0001, 0002 etc). This is

“0001” unless there is more than one case occurring on the same date.

* Year

> Numeric Variable.

This field contains the year in which the incident occurred. In the case of incident(s) occurring

over an extended period, the field will record the year when the incident was initiated.

* Countery

> Categorical Variable.

This field identifies the country or location where the incident occurred. Separatist regions, such as Kashmir, Chechnya, South Ossetia, Transnistria, or Republic of Cabinda, are coded as part of the “home” country.

In the case where the country in which an incident occurred cannot be identified, it is coded as

“Unknown.”

* Region

>This field identifies the region in which the incident occurred. The regions are divided into the following 12 categories, and dependent on the country coded for the case:

  * 1. North America
  * 2. Central America & Caribbean
  * 3. South America
  * 4. East Asia
  * 5. Southeast Asia
  * 6. South Asia
  * 7. Central Asia
  * 8. Western Europe
  * 9. Eastern Europe
  * 10. Middle East & North Africa
  * 11. Sub-Saharan Africa
  * 12. Australasia & Oceania

* Attack Type

> This field captures the general method of attack and often reflects the broad class of tactics used. It consists of nine categories, which are defined below. Up to three attack types can be recorded for each incident. Typically, only one attack type is recorded for each incident unless the attack is comprised of a sequence of events.

  * 1. ASSASSINATION
  * 2. ARMED ASSAULT
  * 3. BOMBING/EXPLOSION
  * 4. HIJACKING

* 5. HOSTAGE TAKING (BARRICADE INCIDENT)

* 6. HOSTAGE TAKING (KIDNAPPING)

* 7. FACILITY / INFRASTRUCTURE ATTACK

* 8. UNARMED ASSAULT

* 9. UNKNOWN

* Success

> Categorical Variable.

Success of a terrorist strike is defined according to the tangible effects of the attack. Success is not judged in terms of the larger goals of the perpetrators. For example, a bomb that exploded in a building would be counted as a success even if it did not succeed in bringing the building down or inducing government repression.

* 1 = "Yes"

* 0 = "No"

* Suicide

>Categorical Variable. This variable is coded “Yes” in those cases where there is evidence that the perpetrator did not intend to escape from the attack alive.

* 1 = "Yes"

* 0 = "No"

* Target

>Categorical Variable. The target/victim type field captures the general type of target/victim. When a victim is attacked specifically because of his or her relationship to a particular person, such as a prominent figure, the target type reflects that motive.

* 1. BUSINESS

* 2. GOVERNMENT

* 3. POLICE

* 4. MILITARY
* 5. ABORTION RELATED
* 6. AIRPORTS & AIRCRAFT
* 7. GOVERNMENT (DIPLOMATIC)
* 8. EDUCATIONAL INSTITUTION
* 9. FOOD OR WATER SUPPLY
* 10. JOURNALISTS & MEDIA
* 11. MARITIME (INCLUDES PORTS AND MARITIME FACILITIES)
* 12. NGO
* 13. OTHER
* 14. PRIVATE CITIZENS & PROPERTY
* 15. RELIGIOUS FIGURES/INSTITUTIONS
* 16. TELECOMMUNICATION
* 17. TERRORISTS/NON-STATE MILITIAS
* 18. TOURISTS
* 19. TRANSPORTATION (OTHER THAN AVIATION
* 20. UNKNOWN
* 21. UTILITIES
* 22. VIOLENT POLITICAL PARTIES

* Group

> This field contains the name of the group that carried out the attack. In order to ensure consistency in the usage of group names for the database, the GTD database uses a standardized list of group names that have been established by project staff to serve as a reference for all subsequent entries.

* Weapon Type

Up to four weapon types are recorded for each incident. This field records the general type of weapon used in the incident. It consists of the following categorie:

 * 1. Biological

 * 2. Chemical

 * 3. Radiological

 * 4. Nuclear

 * 5. Firearms

 * 6. Explosives

 * 7. Fake Weapons

 * 8. Incendiary

 * 9. Melee

 * 10. Vehicle

 * 11. Sabotage Equipment

 * 12. Other

 * 13. Unknown

* Killed

>Numeric Variable.

This field stores the number of total confirmed fatalities for the incident. The number includes all victims and attackers who died as a direct result of the incident. Where there is evidence of fatalities, but a figure is not reported or it is too vague to be of use, this field remains blank.

Check Unique Values for each variable.

"""

Check Unique Values for each variable.

```
vars = ['eventid', 'year', 'country', 'success', 'suicide', 'attack_type', 'target', 'group',
'weapon_type']

for var in vars:

  to_print = 5

  suff = f'etc.. {eda_df[var].nunique()} values' if eda_df[var].nunique() > to_print else ''

  print(f'Unique values in "{var}" are: {sorted(eda_df[var].unique())[:to_print]} {suff}')


eda_df.head()
```

"""The dataset encompasses details on terrorist attacks spanning the years 1970 to 2017. It includes information about 3537 distinct organizations employing diverse methods like bombing and hijacking, as well as utilizing a variety of weapons such as explosives and firearms. The targeted locations vary widely, ranging from airports to educational institutions, with some incidents involving suicide attacks.

3. ***Data Wrangling***

Data Wrangling Code

In the preceding section, Data Wrangling efforts were initiated to streamline the extensive variable set. In this section, we delve into two key aspects:

Handling Missing Values:

Addressing the challenge of missing values in the dataset is our initial focus. Through practical analysis, decisions have been made, particularly in the context of two distinct sets of features: primary and secondary. Out of the total 21 columns, 8 are considered secondary, earmarked for potential use in a secondary Exploratory Data Analysis (EDA). For this reason, the missing values within these secondary columns are left untouched, preserving the integrity of the data.

Consequently, our attention narrows down to the remaining 8 columns, which are primary variables. Among these, only two columns—'killed' and 'wounded'—retain missing values, prompting the need for further consideration.

Creation of New Data Structures:

The second part of this section involves the generation of various new data structures (dataframes, series, etc.) derived from the original dataframe. These transformations are meticulously designed to facilitate a seamless transition into the realm of data visualization. The types of data extracted are thoughtfully explored and discussed in detail.

```
"""

# Checking missing values in 'killed' and 'wounded' columns
missing_values_killed_wounded = eda_df[['killed', 'wounded']].isna().sum()

print('Missing values in the "killed" and "wounded" columns:')
print(missing_values_killed_wounded)

# Calculating the impact of dropping rows with missing values in 'killed' and 'wounded'
total_rows = eda_df.shape[0]
rows_after_dropping = eda_df.dropna(subset=['killed', 'wounded']).shape[0]

lost_rows = total_rows - rows_after_dropping
percentage_lost_rows = round((lost_rows * 100) / total_rows, 2)

print('\nNumber of rows lost after dropping null values:', end=' ')
```

```python
print(f'{lost_rows} or {percentage_lost_rows}%')


import plotly.graph_objects as go


def line_plots(to_plot_dfs, names=None, figure_=True, markers=True):
    '''
    Returns a figure with a line plot using Plotly.
    Inputs are:
    - to_plot_dfs: the dataframes to plot, stored in a list
    - names: the names of each plot, also as a list
    - figure_: a boolean to indicate whether the figure or the data to plot is to be returned
    - markers: a boolean to indicate whether markers should be included in the plot
    '''
    to_plot = [
        go.Scatter(
            x=df.index,
            y=df.values,
            mode='lines',
            marker=dict(symbol='circle', size=8) if markers else None,
            name=name
        )
        for df, name in zip(to_plot_dfs, names)
    ]

    if figure_:
        fig = go.Figure(data=to_plot)
```

```
        return fig

    return to_plot

# Data preparation
yearly_df = eda_df['year'].value_counts().sort_index()
yearly_dropna_df = eda_df.dropna(subset=['killed',
'wounded'])['year'].value_counts().sort_index()
names = ['With NA of killed and wounded', 'Dropping NA of killed and wounded']

# Create the figure
fig = line_plots([yearly_df, yearly_dropna_df], names, True)

# Setting the layout for background color and spines
fig.update_layout(
    title='Yearly terrorism events - with and without dropping NA of killed and wounded',
    yaxis_title='Number of terrorist events',
    xaxis_title='Year',
    autosize=False,
    width=1300,
    height=600,
    template="simple_white",
    xaxis=dict(showline=True, showgrid=False, showticklabels=True, linecolor='black',
linewidth=2),
    yaxis=dict(showline=True, showgrid=False, showticklabels=True, linecolor='black',
linewidth=2),
    showlegend=True
```

)

fig.show()

"""* The analysis highlights a substantial data loss (~9.3%) resulting from the exclusion of null values in the 'killed' and 'wounded' columns. This loss is apparent in the corresponding decrease in terrorist activities across each year.

* Given that each row in the dataset represents a distinct terrorist event, any data loss significantly impacts the comprehensiveness of our analysis. The decision not to drop or tamper with these null values is grounded in the understanding that the absence of information in the 'killed' and 'wounded' columns is intentional. According to the provided documentation, these fields remain blank when the figures are either unreported or too vague to be of use.

* As a strategic approach, the entire dataset will be utilized when 'killed' and 'wounded' are not included in the analysis. Meanwhile, these null values will be consciously excluded when the same variables are utilized for Exploratory Data Analysis (EDA). This ensures a nuanced exploration of the dataset while preserving the integrity of the information available.

Data Wrangling Code

"""

Creating a new column for total casualties

eda_df['casualties'] = eda_df['killed'] + eda_df['wounded']

Simplifying the 'weapon_type' categories for better clarity

eda_df['weapon_type'] = eda_df['weapon_type'].replace(

    'Vehicle (not to include vehicle-borne explosives, i.e., car or truck bombs)', 'Vehicle'

)

```
# Displaying the updated distribution of 'weapon_type'
weapon_type_distribution = eda_df['weapon_type'].value_counts()

# Outputting the results
print(weapon_type_distribution)

# Creating a DataFrame for the number of terrorist events per year
terrorist_events_per_year = eda_df['year'].value_counts().sort_index()

# Creating a DataFrame grouped by year and aggregating other variables by addition
yearly_aggregated = eda_df.groupby('year').sum()

# Displaying the aggregated DataFrame as a sample
yearly_aggregated.head()

def create_summary_data(data, column):
    '''
    Create summary data for a given column:
    - Count of occurrences in the dataset.
    - Aggregated data by grouping the dataset based on the column.

    Parameters:
    - data: DataFrame containing the dataset.
    - column: Name of the column for which summary data is generated.
```

```
    Returns:
    - count_df: DataFrame with counts of occurrences for each unique value in the column.
    - sum_df: DataFrame with aggregated data by grouping based on the column.
    '''
    count_df = data[column].value_counts()
    sum_df = data.groupby(column).sum()
    return count_df, sum_df

# Creating DataFrames for number of terrorist events and aggregated variables for each category
region_df, region_sum = create_summary_data(eda_df, 'region')
country_df, country_sum = create_summary_data(eda_df, 'country')
group_df, group_sum = create_summary_data(eda_df, 'group')
attack_df, attack_sum = create_summary_data(eda_df, 'attack_type')
weapon_df, weapon_sum = create_summary_data(eda_df, 'weapon_type')
target_df, target_sum = create_summary_data(eda_df, 'target')

# Visualizing the count of terrorist events per region
region_df.head()

region_sum.head()

# Creating a DataFrame with the number of events for each weapon type, focusing on the top 3 attack types
attack_weapon_df = eda_df.groupby(['attack_type', 'weapon_type']).count()['eventid']

# Sorting and selecting the top 3 attack types based on the count of events
```

top3attack_weapon = attack_weapon_df.loc[attack_df[:3].index].groupby('attack_type', group_keys=False).apply(lambda x: x.sort_values(ascending=False))

top3attack_weapon = top3attack_weapon.to_frame().rename(columns={'eventid': 'number of events'})

Displaying a sample of the DataFrame

top3attack_weapon.loc[attack_df[:2].index]

Creating a DataFrame with the number of events for each attack type, focusing on the top 5 targets

target_attack_df = eda_df.groupby(['target', 'attack_type']).count()['eventid']

Selecting the top 5 targets based on the count of events

top5target_attack = target_attack_df.loc[target_df[:5].index].groupby('target', group_keys = False).apply(lambda x: x.sort_values(ascending = False))

top5target_attack = top5target_attack.to_frame().rename(columns = {'eventid': 'number of events'})

Visualizing the dataframe sample

top5target_attack.loc[target_df[:2].index]

Top 4 countries with terrorist activities

top_countries = country_df.head(4).index

Unmanipulated dataframes of specific countries stored in a dictionary

country_dict = {country: eda_df[eda_df['country'] == country] for country in top_countries}

Storing dataframes of specific countries grouped by year in a dictionary

country_yearly_sum = {country: eda_df[eda_df['country'] == country].groupby('year').sum() for country in top_countries}

Storing data of specific countries consisting of the number of terrorist events with each attack type, weapon type, etc in dictionaries

country_attack_dict = {country: eda_df[eda_df['country'] == country]['attack_type'].value_counts() for country in top_countries}

country_weapon_dict = {country: eda_df[eda_df['country'] == country]['weapon_type'].value_counts() for country in top_countries}

country_target_dict = {country: eda_df[eda_df['country'] == country]['target'].value_counts() for country in top_countries}

country_group_dict = {country: eda_df[eda_df['country'] == country]['group'].value_counts() for country in top_countries}

country_city_dict = {country: eda_df[eda_df['country'] == country]['city'].value_counts() for country in top_countries}

Visualizing one of the dictionaries as a sample

print(country_attack_dict['Iraq'].head())

Storing all data in a list

country_data = [country_dict, country_yearly_sum, country_attack_dict, country_weapon_dict, country_target_dict, country_group_dict, country_city_dict]

Selecting the top 4 known terrorist groups and adding specific groups

selected_groups = list(group_df.drop('Unknown').head(4).index) + ['Boko Haram', 'Al-Qaida', 'Communist Party of India - Maoist (CPI-Maoist)', 'Maoists', 'Sikh Extremists']

Unmanipulated dataframes of specific countries stored in a dictionary

group_dict = {group: eda_df[eda_df['group'] == group] for group in selected_groups}

Storing dataframes of specific terrorist groups grouped by year in a dictionary

group_yearly_sum = {group: eda_df[eda_df['group'] == group].groupby('year').sum() for group in selected_groups}

Storing data of specific terrorist groups consisting of the number of terrorist events with each attack type, weapon type, etc in dictionaries

group_attack_dict = {group: eda_df[eda_df['group'] == group]['attack_type'].value_counts() for group in selected_groups}

group_weapon_dict = {group: eda_df[eda_df['group'] == group]['weapon_type'].value_counts() for group in selected_groups}

group_target_dict = {group: eda_df[eda_df['group'] == group]['target'].value_counts() for group in selected_groups}

group_country_dict = {group: eda_df[eda_df['group'] == group]['country'].value_counts() for group in selected_groups}

group_city_dict = {group: eda_df[eda_df['group'] == group]['city'].value_counts() for group in selected_groups}

Visualizing one of the dictionaries as a sample

sample_group = 'Taliban'

print(f"{sample_group}: Top 5 attack types:\n\n{group_attack_dict[sample_group].head()}")

Storing all data in a list

group_data = [group_dict, group_yearly_sum, group_attack_dict, group_weapon_dict, group_target_dict, group_country_dict, group_city_dict]

"""### What all manipulations have you done and insights you found?

In this section, we meticulously prepared the data for visualization and analysis through several key manipulations:

1. Casualties Column: Created a 'casualties' column by combining the 'killed' and 'wounded' variables. This consolidation simplifies the visualization of the overall impact of terrorism.

2. Grouping by Categories: Formed dataframes by grouping categorical variables, counting event occurrences, and summing numerical variables. This approach provides a structured overview of the data.

3. Bivariate Analysis: Conducted bivariate analysis by creating dataframes for the most used weapon types corresponding to the top 5 attack types. This analysis helps identify patterns and trends in terrorist activities.

4. Target-Specific Analysis: Similarly, investigated the most used attack methods for specific terrorist targets. This analysis offers insights into the tactics employed for particular objectives.

Additionally, we delved into studying specific local countries of interest and high-impact terrorist organizations. Separate dataframes were generated for each, examining characteristics such as attack types, targeted entities, and motives. This granular exploration aimed at understanding the unique features and motivations of individual terrorist groups and gaining insights into specific countries of interest.

Through these manipulations, we aim to unravel meaningful patterns, motives, and geographical trends, contributing to a comprehensive understanding of terrorism dynamics.

4. Data Vizualization, Storytelling & Experimenting with charts : Understand the relationships between variables

Introduction

The analysis of the dataset is structured into three main sections, each consisting of multiple sub-sections:

1. Global Analysis:

* Holistic examination of the entire dataset, covering all countries and terrorist organizations.

* Focus on understanding general characteristics of terrorist attacks and their impact on the population.

* Commonly used "Impact" metrics include the total number of terrorist events, casualties (killed and wounded), number of suicide attacks, and successful attacks.

* Analyses includes:

   * Annual analysis

   * Demographic (regional) analysis,

   * Demographic (national) analysis,

   * General analysis of terrorist organizations,

   * Analysis of attack types, weapon types

   * Analysis of target types.

* Observations from plots are recorded as KEY TAKEAWAYS .

2. Local Analysis:

* Concentrated analysis on a subset of the dataset, focusing on specific countries of interest.

* Countries selected based on hot-spots of terrorist activities, as identified in heatmaps and plots.

* Countries under consideration:

   * Iraq

   * Afghanistan

* Pakistan

* India.

3. Terrorist Organizations Analysis:

* Selected based on the criteria of notoriety, popularity, and the impact of terrorism, an in-depth analysis was conducted on two specific terrorist organizations:

* Taliban

* The Islamic State of Iraq and the Levant (ISIL).

* Brief analysis of additional organizations:

* Al-Qaida

* Shining Path

* Boko Haram

1. Global Analysis A Comprehensive Study

1. Annual Analysis

1. Graphs
"""

```
# Plotting the distribution of terrorist events by year

fig = go.Figure()


# Add a line plot for yearly terrorism events

fig.add_trace(go.Scatter(x=yearly_df.index, y=yearly_df.values, mode='lines', name='Yearly Terrorism Events'))
```

```
# Customize the layout
fig.update_layout(
    title='Annual Terrorism Events',
    yaxis_title='Number of Terrorist Events',
    xaxis_title='Year',
    xaxis=dict(showline=True, showgrid=False, showticklabels=True, linecolor='black',
linewidth=2, tickfont=dict(size=8)),
    yaxis=dict(showline=True, showgrid=False, showticklabels=True, linecolor='black',
linewidth=2, tickfont=dict(size=10)),
    autosize=True,
    template="simple_white"
)

# Show the plot
fig.show()

# Create a DataFrame for the number of casualties by year
cas_cols = ['killed', 'wounded', 'casualties']
casualties_by_year = yearly_aggregated[cas_cols].sort_values(cas_cols, ascending=False).head()

# Display the top rows of the DataFrame
casualties_by_year

# Plotting the number of killed and wounded by year
fig = go.Figure()
```

```
# Add line plots for the number of people killed and wounded
for var in cas_cols[:-1]:
    fig.add_trace(go.Scatter(x=yearly_aggregated.index, y=yearly_aggregated[var], mode='lines',
name=f'Number of {var}'))

# Customize the layout
fig.update_layout(
    title='Annual Casualties Due to Terrorism',
    yaxis_title='Number of People',
    xaxis_title='Year',
    xaxis=dict(showline=True, showgrid=False, showticklabels=True, linecolor='black',
linewidth=2, tickfont=dict(size=8)),
    yaxis=dict(showline=True, showgrid=False, showticklabels=True, linecolor='black',
linewidth=2, tickfont=dict(size=10)),
    autosize=True,
    template="simple_white"
)

# Show the plot
fig.show()

# Plot of number of people killed, wounded, and total casualties
for var in cas_cols:
    text = f'people {var}' if var != 'casualties' else var

    # Line plots
```

```
    line_traces = line_plots([yearly_aggregated[var]], [f'Number of {text}'], figure_=False, markers
= False)

    # Bar plot for average number of people per event
    avg_trace = go.Bar(
        x=yearly_aggregated.index,
        y=(yearly_aggregated[var] / yearly_df),
        yaxis='y2',
        name=f'Average {text} per event',
        marker=dict(opacity=0.8, color='#bdcf32')
    )

    # Create the figure
    fig = go.Figure(data=[*line_traces, avg_trace])

    # Update the layout
    fig.update_layout(
        title=f'Annual Total and Average Number of {text}',
        xaxis_title='Year',
        yaxis_title='Number of people',
        xaxis=dict(showline=True, showgrid=False, showticklabels=True, linecolor='black',
linewidth=2, tickfont=dict(size=8)),
        yaxis=dict(showline=True, showgrid=False, showticklabels=True, linecolor='black',
linewidth=2, tickfont=dict(size=10)),
        yaxis2=dict(title='Avg number of people', overlaying='y', side='right'),
        autosize=True,
        template="simple_white"
```

```
    )

    # Show the plot
    fig.show()

# Creating a DataFrame for the number of successful and suicide events by year
success_suicide_by_year = yearly_aggregated[['success', 'suicide']]
top_success_suicide_by_year = success_suicide_by_year.sort_values(['success', 'suicide'],
ascending=False)

# Displaying the top successful and suicide events by year DataFrame
top_success_suicide_by_year.head()

# Plotting the successful and suicide events by year
texts = ['successful events', 'suicide events']
vars = ['success', 'suicide']

for text, var in zip(texts, vars):
    to_plot_df = [yearly_aggregated[var], yearly_df] if var == 'success' else
[yearly_aggregated[var]]
    yrange = [78, 100] if var == 'success' else [0, 12]

    # Creating line traces for the number of events
    line_traces = go.Scatter(x=yearly_aggregated.index, y=to_plot_df[0], mode='lines',
name=f'Number of {text}')

    # Creating a bar trace for the percentage of events
```

```
    success_trace = go.Bar(
        x=yearly_aggregated.index,
        y=(yearly_aggregated[var] / yearly_df * 100),
        yaxis='y2',
        name=f'Percentage of {text}',
        marker=dict(opacity=0.8, color='#bdcf32')
    )

    # Creating the figure
    fig = go.Figure(data=[line_traces, success_trace])

    # Updating layout
    fig.update_layout(
        title=f'Annual {text}',
        xaxis_title='Year',
        yaxis_title='Number of events',
        yaxis2=dict(title='Percentage', overlaying='y', side='right', range=yrange),
        autosize=True,
        template="simple_white"
    )

    # Displaying the figure
    fig.show()

"""##### 2. Why picked the specific chart?
```

* To visualize temporal trends and changes over time, line plots were chosen for their simplicity and effectiveness. These plots connect data points chronologically, providing a clear representation of temporal progression.

* In cases where displaying percentages or averages alongside temporal line plots was necessary, box plots with adjusted opacity were employed. This choice aimed to prevent cluttering and enhance interpretability.

3. What is/are the insight(s) found from the chart?

This chart serves as a foundational overview of the temporal trends within the dataset. Parameters such as killed, wounded, casualties, success, and suicide are integral metrics for gauging the impact of terrorist attacks and are consistently utilized throughout the subsequent charts.

* The first plot illustrates the trajectory of terrorist events over the years. Notably, there is a gradual increase until 1992, followed by a decline until 2004. Subsequently, there is a sharp upswing in terrorist activities, peaking in 2014, and then a considerable decrease until 2017.

* In tandem with this trend, the second plot reflects a similar pattern for the number of individuals killed and wounded. However, a local peak is observed in 2001, a year with a lower count of terrorist events. This anomaly suggests a high number of casualties in a few attacks, potentially attributed to the infamous 9/11 incident in New York—a point to be further explored in upcoming charts.

* The third plot overlays the average number of killed/wounded/casualties per event annually with the total count each year. Notably, the average casualties per event reach a peak in 2001, followed by 2004. The former aligns with the impact of the 9/11 attacks. Although the number of events increases, the casualties per attack substantially decrease post-2004.

* The fourth plot, akin to the third, focuses on events categorized as successful or involving suicide attacks. Both the count and percentage of successful attacks witness a significant decline in the 21st century. Conversely, the percentage of suicide attacks experiences a manifold increase, a rarity in the 20th century.

> Success is defined based on the tangible effects of the attack, distinct from the larger goals/motives of the perpetrators.

KEY TAKEAWAYS:

* Exponential increase in the number of terrorist attacks in the 21st century, coupled with casualties.

* Noteworthy surge in the number and percentage of suicide attacks in the 21st century.

* The year 2001 registers a peak in casualties despite a lower number of attacks, likely due to the 9/11 attacks in NYC.

4. Will the gained insights help creating a positive impact?

The insights from this section provide valuable information on the temporal trends, casualties, success rates, and the rise of suicide attacks in global terrorism. Understanding these patterns can inform risk assessments, security measures, and geopolitical considerations for businesses operating in regions prone to terrorism, enhancing overall strategic planning and crisis management.

2. Demographic (Regional) Analysis

1. Graphs
"""

```
# Plotting region-wise distribution of terrorist attacks, suicide attacks, and casualties
texts = ['terrorist events', 'suicide terrorist events', 'casualties']

for var, text in zip([None, 'suicide', 'casualties'], texts):
    region_plot_df = region_df if var is None else region_sum[var].sort_values(ascending=False)

    # Creating a bar trace for region-wise distribution
    region_plot = go.Bar(x=region_plot_df.index, y=region_plot_df.values,
marker=dict(color='#bdcf32'))

    # Creating the figure
    fig = go.Figure(data=[region_plot])

    # Updating layout
    fig.update_layout(
        title=f'Region-wise distribution of {text}',
        xaxis_title='Region',
        yaxis_title=f'Number of {text.split(" ")[-1]}',
        autosize=True,
        template="simple_white"
    )

    # Displaying the figure
    fig.show()
```

```
import plotly.graph_objects as go


def multi_cat_lineplot(names_toplot, toplot_dict, title, var=''):
    '''
    Plots multiple line charts representing different categories' trends over time.


    Parameters:
    - names_toplot (list): Names of categories to be plotted.
    - toplot_dict (dict): Dictionary with category names as keys and corresponding data as values.
    - category_title (str): Type of variable represented by categories (e.g., "regions", "groups").
    - metric (str): The specific metric to visualize (e.g., "casualties", "suicide"). If empty, displays
the number of events.


    Returns:
    - None: Displays the generated plot.
    '''


    # Checking if the plot is for the total number of events, or others such as casualties or suicide
events
    events = True if var == '' else False
    if events:
        categ_toplot = list({k: toplot_dict[k] for k in names_toplot}.values())
    else:
        categ_toplot = list({k: toplot_dict[k][var] for k in names_toplot}.values())


    # Plot
    fig = line_plots(categ_toplot, names_toplot, markers=False)
```

```
    var = 'terrorism' if var == '' else var

    var = var + ' events' if var != 'casualties' else var

    fig.update_layout(

        title=f'Annual {var} across different {title}',

        yaxis_title=f'Number of {var}',

        xaxis_title='Year',

        autosize=True,

        template="simple_white"

    )

    fig.show()


# Plotting region-wise distribution of terrorist attacks across each year

region_names = eda_df['region'].unique()[::-1]

region_names_tp = [region for region in region_names if region not in ['East Asia', 'Central Asia', 'Australasia & Oceania']]

region_year_dict = {region: eda_df[eda_df['region'] == region]['year'].value_counts().sort_index() for region in region_names}


multi_cat_lineplot(region_names_tp, region_year_dict, 'regions')


# Plotting region-wise distribution of casualties across each year

region_annual_sum_dict = {region: eda_df[eda_df['region'] == region].groupby('year').sum() for region in region_names}

multi_cat_lineplot(region_names_tp, region_annual_sum_dict, 'regions', 'casualties')


# Extracting data for the 9/11 attacks

df_911 = eda_df[
```

```
    (eda_df['region'] == 'North America') &
    (eda_df['year'] == 2001) &
    (eda_df['month'] == 9) &
    (eda_df['day'] == 11)
][cas_cols + ['region', 'country', 'city', 'target', 'attack_type']]

# Calculating total casualties for the 9/11 attacks and for North America in 2001
total_casualties_911 = int(df_911['casualties'].sum())
total_casualties_NA_2001 = int(
    eda_df[
        (eda_df['region'] == 'North America') &
        (eda_df['year'] == 2001)
    ]['casualties'].sum()
)

# Displaying the results
print(f"Total casualties in 9/11 attacks = {total_casualties_911}")
print(f"Total casualties in North America in 2001 = {total_casualties_NA_2001}\n")

# Displaying the details of the 9/11 attacks
df_911.head()

# Plotting region-wise distribution of suicide events across each year
multi_cat_lineplot(region_names_tp, region_annual_sum_dict, 'regions', 'suicide')

"""##### 2. Why did you pick the specific chart?
```

* To illustrate the cumulative distribution of total events, casualties, and suicide attacks across regions, bar plots are employed. In cases where the temporal aspect is significant, line plots are reintroduced. This choice stems from the distinct characteristics of these plot types.

* Line plots are adept at highlighting trends over different values of the x-coordinate, crucial in temporal analyses. On the other hand, bar plots are better suited for independent analysis of categories, offering a clearer representation when examining the cumulative distribution across regions over all time.

3. What is/are the insight(s) found from the chart?

* In the initial visualization, the Middle East, South Asia, and Sub-Saharan Africa stand out as regions with the highest number of terrorist events, casualties, and suicide attacks. South America exhibits notable terrorist activity but with fewer casualties.

* Examining the temporal distribution of terrorist events reveals significant attacks in Western Europe during the latter half of the 20th century, followed by Central and Southern America. The 21st century witnesses a sharp increase in terrorist activities in the Middle East, South Asia, and African regions.

* This temporal trend is also evident in the yearly plot of casualties. However, a remarkable peak in 2001 in North America aligns with the hypothesis from the previous chart and can be attributed to the four attacks in NYC on 9/11. Nearly all the casualties in North America in 2001 result from this specific terrorist incident.

* The final plot underscores a shift in the nature of terrorist attacks. While the 20th century saw numerous attacks, suicide attacks were relatively rare. However, in the 21st century, regions like South Asia, the Middle East, and Africa witness an increase in suicide attacks.

KEY TAKEAWAYS:

* The Middle East, South Asia, and Sub-Saharan Africa are terrorism hotspots, with a pronounced surge from the 21st century onwards. These regions also experience a significant presence of suicide attacks, a phenomenon less prevalent in other regions.

* The 9/11 terrorist attack had a profound impact, particularly in terms of casualties. This single incident, comprising four attacks, resulted in nearly 19,000 casualties, with a third of them fatal.

4. Will the gained insights help creating a positive impact?

The insights from this section are crucial for businesses operating in regions identified as terrorism hotspots. Understanding the geographical distribution, temporal trends, and the prevalence of suicide attacks provides valuable context for risk assessment and security planning, enabling businesses to tailor strategies for enhanced safety and resilience in these specific regions.

3.Demographic (National) Analysis

1.Graphs
"""

```
# Plotting the distribution of terrorist events by country (top 20)
texts = ['terrorist events', 'casualties', 'suicide terrorist events']

for var, text in zip([None, 'casualties', 'suicide'], texts):
    country_plot_df = country_df.head(20) if var is None else
country_sum[var].sort_values(ascending=False).head(20)
    country_plot = go.Bar(x=country_plot_df.index, y=country_plot_df.values, marker=
dict(color='#bdcf32'))
```

```python
    # Creating the figure
    fig = go.Figure(data=[country_plot])

    # Updating layout
    fig.update_layout(
        title=f'Country-wise distribution of {text} (top 20)',
        xaxis_title='Country',
        yaxis_title=f'Number of {text.split(" ")[-1]}',
        autosize=True,
        template="simple_white"
    )

    # Displaying the figure
    fig.show()

def plot_world(data_frame, title):
    '''
    Plots a choropleth map using Plotly Express.

    Parameters:
    - data_frame: DataFrame with index as country names and values as the weights.
    - title: Title of the map as a string.
    '''
    fig = px.choropleth(data_frame, locations=data_frame.index, locationmode='country names',
                        color=data_frame.values, hover_name=data_frame.index,
```

```python
                    projection='natural earth', color_continuous_scale="bluyl")
    fig.update_layout(title=title,autosize=True,template="simple_white")
    fig.show()

# Plotting the choropleth map by number of terrorist events
plot_world(country_df, 'Map of Number of Terrorist Events')

# Annual terrorism events across top 10 countries
country_names = list(country_df.head(10).index)
country_year_dict = {country: eda_df[eda_df['country'] ==
country]['year'].value_counts().sort_index() for country in country_names}

multi_cat_lineplot(country_names, country_year_dict, 'countries')

# Plotting the choropleth map by number of suicide events
country_sui = country_sum['suicide'].sort_values(ascending = False)
plot_world(country_sui, 'Map of number of suicide events')

# Annual suicide events across top 10 countries
country_names = list(country_sum['suicide'].sort_values(ascending=False).head(10).index)
country_yr_sum_dict = {country: eda_df[eda_df['country'] == country].groupby('year').sum() for
country in country_names}

multi_cat_lineplot(country_names, country_yr_sum_dict, 'countries', 'suicide')

# Plotting the choropleth map by number of Casualties
country_cas = country_sum['casualties'].sort_values(ascending = False)
```

plot_world(country_cas, 'Map of number of casualties')

Annual terrorism casualties across top 10 countries

country_names = list(country_sum['casualties'].sort_values(ascending=False).head(10).index)

country_yr_sum_dict = {country: eda_df[eda_df['country'] == country].groupby('year').sum() for country in country_names}

multi_cat_lineplot(country_names, country_yr_sum_dict, 'countries', 'casualties')

"""##### 2. What is/are the insight(s) found from the chart?

* Similar to the previous chart, line plots have been utilized to visualize the number of events, casualties, and suicide attacks across each country. Line plots are employed with a temporal factor.

* Given the large number of countries, it's impractical to display all of them in the line plots alone. Therefore, choropleth plots have been employed to visualize the global distribution on a map.

3. Will the gained insights help creating a positive business impact?

* Building on the insights gained from the previous chart, Iraq, Afghanistan, Pakistan, and India emerge as prominent terrorist hotspots, all situated in the top regions of the Middle East and South Asia.

* This observation is further emphasized in the choropleth heatmap. Notably, the United States records a relatively high number of casualties, reaching around 24,000. However, a staggering 80% of these casualties stem from a single incident.

* Extending the analysis from the prior chart, the 20th century witnessed heightened terrorist activities in the United Kingdom, El Salvador, and Peru. Contrastingly, the 21st century experienced a surge in terrorist attacks in Iraq, Pakistan, Afghanistan, and India, a trend also evident in the yearly distribution of casualties.

KEY TAKEAWAYS:

* Iraq, Pakistan, Afghanistan, and India emerge as significant terrorist hotspots in the 21st century, characterized by a high frequency of attacks and casualties. These countries will be subject to focused study in subsequent sections.

* Suicide attacks were notably prevalent as a method of assault in these regions.

In the 20th century, the United Kingdom, El Salvador, and Peru were the primary terrorist hotspots.

4. Will the gained insights help creating a positive impact?

The insights gained provide a valuable understanding of the prominent terrorist hotspots, patterns of attacks, and casualties in specific regions. This knowledge can contribute to informed decision-making in areas of conflict, aiding policymakers, security agencies, and researchers in developing targeted strategies for conflict resolution, peace-building, and counter-terrorism efforts.

4. Analysis of Terrorist Organisations

Data preparation

"""

Head of dataset of terrorist groups with most attacks

```
group_df.head()

"""Due to the overwhelming majority of events being attributed to Unknown groups, they will be set aside for the purpose of visualization.

##### 1.Graphs
"""

# Plotting the terrorist group-wise distribution of total terrorist events, suicide attacks, and casualties
texts = ['terrorist events', 'suicide terrorist events', 'casualties']
for var, text in zip([None, 'suicide', 'casualties'], texts):
    group_plot_df = group_df[1:].head(20) if var is None else group_sum[var].sort_values(ascending=False)[1:].head(20)
    group_plot = go.Bar(x=group_plot_df.index, y=group_plot_df.values, marker= dict(color='#bdcf32'))

    # Creating the figure
    fig = go.Figure(data=group_plot)

    # Updating layout
    fig.update_layout(
        title=f'Terrorist group-wise distribution of {text}',
        xaxis_title='Terrorist group',
        yaxis_title=f'Number of {text.split(" ")[-1]}',
        autosize=True,
        template="simple_white"
```

```
    )


    # Displaying the figure
    fig.show()


# Creating a dataframe with the success percentage of the top 20 terrorist groups relative to the number of attacks
var = 'success'
groups_toplot = group_sum[var].sort_values(ascending=False)[1:].head(20)


# Calculating success percentage
group_success_pct = (groups_toplot / group_df[groups_toplot.index] * 100).round(2)


# Sorting and displaying the top 10 groups with the highest success percentage
group_success_pct.sort_values(ascending=False).to_frame().rename(columns={0: 'Percentage of success'}).head(10)


"""Defining success is necessary to assess and quantify the effectiveness of terrorist groups by understanding the proportion of successful attacks relative to their overall activities, providing insights into their operational capabilities and impact.


"""


def pct_success_plot(df_toplot, pct_df, var, pct_range):
    '''
    Plots a bar-line overlay plot for each category of the input dataframe, with the bars indicating
    number of successful events and the lines indicating percentage of successful events
```

```
    Parameters:
    - df_toplot (pd.Series): Dataframe with successful events as values and the classes as index.
    - pct_df (pd.DataFrame): Second dataframe with percentage of success for the same classes.
    - var (str): The type of class ("group", "attack type", etc).
    - pct_range (list): The axis range of the percentage plot as a list.

    Returns:
    None
    '''
    bar_trace = go.Bar(x=df_toplot.index, y=df_toplot, yaxis='y2', name='Total success events',
                  marker=dict(opacity=0.8, color='#bdcf32'))
    line_trace = line_plots([pct_df], names=['Percentage of success'], figure_=False,
markers=False)

    fig = go.Figure(data=[*line_trace, bar_trace])
    fig.update_layout(title=f'Percentage and Total successful events for each {var}',
                  xaxis_title=var,
                  yaxis_title='Percentage of success',
                  yaxis2=dict(title='Total success events', overlaying='y', side='right'),
                  yaxis_range=pct_range,
                  autosize=True,
                  template="simple_white")
    fig.show()

# Genrating graph for visualizing success rate
```

pct_success_plot(groups_toplot, group_success_pct, 'group', [70, 100])

Yearly terrorism events of the top 10 groups

group_names = list(group_df[1:].head(10).index)

group_year_dict = {group: eda_df[eda_df['group'] == group]['year'].value_counts().sort_index() for group in group_names}

multi_cat_lineplot(group_names, group_year_dict, 'groups')

print(group_names)

Analyzing yearly terrorism metrics (casualties, success, suicide events) of the top 10 terrorist groups in each respective field

for var in ['casualties', 'success', 'suicide']:

  # Extracting the top 10 groups based on the specified metric

  group_names = list(group_sum[var].sort_values(ascending=False)[1:].head(10).index)

  # Creating a dictionary with yearly aggregated data for each top group

  group_yr_sum_dict = {group: eda_df[eda_df['group'] == group].groupby('year').sum() for group in group_names}

  # Plotting line charts for each top group based on the metric

  multi_cat_lineplot(group_names, group_yr_sum_dict, 'groups', var)

"""##### 2. Why did you pick the specific chart?

* Employing a consistent approach, bar plots were employed to illustrate casualties, the count of attacks, and suicide attacks for each group. Line plots were introduced when considering the temporal aspect.

* To enhance readability, the number of successful events and their percentage were amalgamated in a single graph, necessitating a combination of line and bar plots.

3. What is/are the insight(s) found from the chart?

* In the first plot, the Taliban and ISIL emerge as the leading terrorist groups, exhibiting the highest number of casualties, suicide attacks, and overall attacks. Notably, Shining Path and FMNL, while having a significant number of attacks, experienced fewer casualties. Al-Qaeda, despite not ranking in the top 20 groups by the number of attacks, incurred substantial casualties, primarily due to the 9/11 attacks, which alone claimed approximately 20,000 lives.

* The yearly distribution of terrorist attacks indicates heightened terrorist activities in the United Kingdom, El Salvador, and Peru. Specific groups such as the Irish Republican Army (IRA), Shining Path, and FMNL played significant roles in these regions.

* The IRA sought to end British rule in Northern Ireland, reflecting in both charts. FMNL, stemming from the civil war in El Salvador, and Shining Path, a far-left guerrilla terrorist group aiming to overthrow the Peruvian government in the late 20th century, were key contributors.

* The 21st century witnessed the sudden rise of the Taliban and ISIL, the latter emerging as late as 2013. Countries such as Iraq, Pakistan, Afghanistan and India were most affected by these groups, which, along with Boko Haram, frequently employed suicide attacks as their methodology.

KEY TAKEAWAYS:

- Taliban and ISIL are the most prominent terrorist organizations with the maximum impact in terms of casualties and the number of terrorist attacks. These two groups will be specifically studied in upcoming sections.

- In the late half of the 20th century, the countries identified as terrorist hotspots (United Kingdom, El Salvador and Peru.) featured the IRA, FMNL, and Shining Path as the respective terrorist groups. This is further analyzed in the final section, focusing on two of these groups.

4. Will the gained insights help creating a positive impact?

The gained insights offer a comprehensive understanding of the prominent terrorist organizations, their historical context, and geographical impact. This knowledge can contribute to more effective counter-terrorism measures, international cooperation, and focused efforts to address the root causes of terrorism, potentially leading to a positive impact on global security and stability.

5. Analysis of Attack & Weapon Types

1. Graphs

```
"""


# Function to plot bar charts with numeric values on bars
def plot_bar_chart_with_values(df, title, x_label, y_label, color='#bdcf32'):
    bar_chart = go.Bar(x=df.index, y=df.values, marker=dict(color=color))
    annotations = [dict(x=xi, y=yi, text=str(yi), showarrow=False, yanchor='bottom') for xi, yi in zip(df.index, df.values)]


    fig = go.Figure(data=[bar_chart])


    fig.update_layout(
        title=title,
```

```
        xaxis_title=x_label,
        yaxis_title=y_label,
        annotations=annotations,
        autosize=True,
        template="simple_white"
    )

    fig.show()

# Plotting attack type distribution with numeric values
plot_bar_chart_with_values(attack_df, 'Distribution of Attack Types', 'Attack Type', 'Number of Attacks')

# Plotting weapon type distribution with numeric values
plot_bar_chart_with_values(weapon_df, 'Distribution of Weapon Types', 'Weapon Type', 'Number of Attacks')

# Filtering data for suicide attacks
suicide_data = eda_df[eda_df['suicide'] == 1]

# Creating dataframes for attack type distribution in suicide attacks
suicide_attack_df = suicide_data['attack_type'].value_counts()

# Creating dataframes for weapon type distribution in suicide attacks
suicide_weapon_df = suicide_data['weapon_type'].value_counts()

# Function to plot bar charts with numeric values on bars for suicide attacks
```

```
def plot_bar_chart_with_values(df, title, x_label, y_label, color='#bdcf32'):
    bar_chart = go.Bar(x=df.index, y=df.values, marker=dict(color=color))
    annotations = [dict(x=xi, y=yi, text=str(yi), showarrow=False, xanchor='center',
yanchor='bottom') for xi, yi in zip(df.index, df.values)]

    fig = go.Figure(data=[bar_chart])

    fig.update_layout(
        title=title,
        xaxis_title=x_label,
        yaxis_title=y_label,
        annotations=annotations,
        autosize=True,
        template="simple_white"
    )

    fig.show()

# Plotting attack type distribution in suicide attacks with numeric values
plot_bar_chart_with_values(suicide_attack_df, 'Distribution of Attack Types in Suicide Attacks',
'Attack Type', 'Number of Suicide Attacks')

# Plotting weapon type distribution in suicide attacks with numeric values
plot_bar_chart_with_values(suicide_weapon_df, 'Distribution of Weapon Types in Suicide
Attacks', 'Weapon Type', 'Number of Suicide Attacks')

# Plotting success events and percentage of success for attack types
```

```
attack_success = attack_sum['success'].sort_values(ascending=False)
attack_success_pct = attack_success / attack_df[attack_success.index] * 100
pct_success_plot(attack_success, attack_success_pct, 'attack type', [30, 100])

# Plotting success events and percentage of success for weapon types
weapon_success = weapon_sum['success'].sort_values(ascending=False)
weapon_success_pct = weapon_success / weapon_df[weapon_success.index] * 100
pct_success_plot(weapon_success, weapon_success_pct, 'weapon type', [30, 100])

# Function to plot bar charts for casualties per attack or weapon type
def plot_casualties_per_category(df, category, title, color='#bdcf32'):
    casualties_df = df['casualties'].sort_values(ascending=False)
    fig = go.Figure(data=go.Bar(x=casualties_df.index, y=casualties_df.values,
marker=dict(color=color)))

    fig.update_layout(
        title=f'Casualties per {title}',
        xaxis_title=title,
        yaxis_title='Number of Casualties',
        autosize=True,
        template="simple_white"
    )

    fig.show()

# Plotting casualties per attack type
```

```
plot_casualties_per_category(attack_sum, 'Attack Type', 'attack type')

# Plotting casualties per weapon type
plot_casualties_per_category(weapon_sum, 'Weapon Type', 'weapon type')

# Proportions of weapon types used for top 3 attack types
top3_attacks = {index[0] for index in top3attack_weapon.index}

for var in top3_attacks:
    # Select top 6 weapon types, as others are considered insignificant
    top6_weapon_types = top3attack_weapon.loc[var].nlargest(6, 'number of events')

    # Calculate percentage on the fly
    top6_weapon_types['percentage'] = (top6_weapon_types['number of events'] /
top6_weapon_types['number of events'].sum()) * 100

    # Create a bar graph
    fig = go.Figure(data=go.Bar(x=top6_weapon_types.index, y=top6_weapon_types['number of
events'],
                    marker=dict(color='#bdcf32')))

    # Update layout
    fig.update_layout(title=f'Proportions of each weapon type for {var} attacks',
              xaxis_title='Weapon Type',
              yaxis_title='Number of Events',
              autosize=False,
              width=900,
```

```
            height=550,
            template="simple_white")

# Add annotations over the bars
for i, value in enumerate(top6_weapon_types['number of events']):
    fig.add_annotation(
        x=top6_weapon_types.index[i],
        y=value,
        text=f'{top6_weapon_types["percentage"].iloc[i]:.2f}%',  # Include percentage value
        showarrow=False,
        font=dict(color='black', size=10),
        yshift=5
    )

fig.show()

"""Considering the nearly one-to-one correspondence observed between the top attack types and weapon types (e.g., Bombing/Explosion aligns with Explosives, Armed Assault aligns with Firearms, etc.), the evaluation primarily focuses on attack types. Only the top 6 weapon types are taken into consideration for assessment, as lower values are deemed to be insignificant."
"""


# Yearly terrorism events across different attack types
attack_yr_dict = {attack_type: eda_df[eda_df['attack_type'] == attack_type]['year'].value_counts().sort_index() for attack_type in list(attack_df.index)}
multi_cat_lineplot(list(attack_df.index), attack_yr_dict, 'Attack Types')
```

Yearly terrorism casualties and successful events across different attack types

for var in ['casualties', 'success']:

attack_yr_sum_dict = {attack_type: eda_df[eda_df['attack_type'] == attack_type].groupby('year').sum() for attack_type in list(attack_df.index)}

multi_cat_lineplot(list(attack_df.index), attack_yr_sum_dict, 'attack types', var)

"""##### 2. Why did you pick the specific chart?

- In alignment with a consistent visualization strategy, bar charts were selected to portray casualties, the frequency of attacks, and suicide attacks for various parameters. This provides a clear and uniform representation across different aspects.

- For a comprehensive overview of success metrics, a combined approach of line and bar plots was employed to depict the count and percentage of successful events, ensuring a holistic understanding of the data's temporal and categorical dimensions.

3. What is/are the insight(s) found from the chart?

- It's evident that roughly 50% of terrorist attacks are executed using Bombing/Explosives, making Explosives the predominant weapon, accounting for nearly half of all attacks.

- In the context of suicide attacks, Explosives are almost exclusively used, with a substantial 94% of suicide attacks involving them.

- When considering success rates, Assassinations exhibit a relatively lower success rate (75%) compared to Bombings and Armed Assaults. This aligns with the previously defined success criteria in the dataset.

- The close association between attack type and weapon type is reaffirmed in the subsequent chart, revealing that nearly all Bombing and Armed Assault attacks involve the use of Explosives and Firearms.

- Bombings and Armed Assaults have consistently remained the top choices for terrorists in both the 20th and 21st centuries.

- The aftermath of 9/11 is apparent in the yearly plot of casualties, with a sudden peak observed in the Hijacking method of terrorist attacks.

KEY TAKEAWAYS:

- Bombings and Armed Assaults stand out as preferred methods of terrorist attacks, with a significant overlap in suicide attacks involving the former.

- The one-to-one relationship between attack type and weapon type highlights the primary focus on the former for this project.

4. Will the gained insights help creating a positive business impact?

Understanding that Bombing/Explosives is the predominant attack method, followed by Firearms, offers crucial insights for devising effective counterterrorism strategies. This awareness empowers security agencies to implement targeted detection and prevention measures, thereby minimizing the impact of these attacks.

6. Analysis of Target Types

1. Graphs
"""

```
# Proportions of different targets of terrorists

fig = go.Figure(data = go.Pie(values = target_df.values, labels = target_df.index))

fig.update_layout(title = f'Proportions of each target of terrorists', autosize = True)

fig.show()


# Different targets of suicide attacks

sui_df = target_sum['suicide'].sort_values(ascending = False)

fig = go.Figure(data = go.Pie(values = sui_df.values, labels = sui_df.index))

fig.update_layout(title = f'Proportions of targets of suicide attacks', autosize =True)

fig.show()


# Plotting success events and percentage of success for different targets of terrorists

target_success = target_sum['success'].sort_values(ascending=False)

target_success_pct = target_success / target_df[target_success.index] * 100


# Call the function to create a bar-line overlay plot

pct_success_plot(target_success, target_success_pct, 'target', [30, 100])


# Plotting the total casualties per target

target_cas_df = target_sum['casualties'].sort_values(ascending=False)

fig = go.Figure(data=go.Bar(x=target_cas_df.index, y=target_cas_df.values,
marker=dict(color='#bdcf32')))

fig.update_layout(title='Casualties per Target', xaxis_title='Target', yaxis_title='Number of
Casualties',

                autosize=True, template="simple_white")

fig.show()
```

```
# Yearly number of terrorist events across top 10 targets
target_names = list(target_df.head(10).index)
target_yr_dict = {target: eda_df[eda_df['target'] == target]['year'].value_counts().sort_index()
for target in target_names}

# Plotting
multi_cat_lineplot(target_names, target_yr_dict, 'Targets')

# Annual terrorism casualties, success events, and suicide events across top 10 targets
for var in ['casualties', 'success', 'suicide']:
    target_names = list(target_sum[var].sort_values(ascending=False).head(10).index)
    target_yr_sum_dict = {target: eda_df[eda_df['target'] == target].groupby('year').sum() for
target in target_names}
    multi_cat_lineplot(target_names, target_yr_sum_dict, 'Targets', var)

# Visualizing the distribution of attack types for top 5 targets
top5_targets = {index[0] for index in top5target_attack.index}
for target in top5_targets:
    fig = go.Figure(data=go.Pie(values=top5target_attack.loc[target]['number of events'],
                                labels=top5target_attack.loc[target].index))
    fig.update_layout(title=f'Distribution of Attack Types for {target} Target',
                      autosize=True, template="simple_white")
    fig.show()
```

"""Based on the one-to-one correspondence established between attack types and weapon types in studied earlier, only one of them will be analyzed in conjunction with the target type. In this case, the attack type is selected for the top 5 targets of terrorists.

2. Why did you pick the specific chart?

- The chosen charts, including bar plots and line plots, follow a consistent methodology, providing clarity in comparing variables across different contexts.

- Pie charts were employed to depict proportions, offering a quick and intuitive understanding of the distribution, especially for variables like attack types on specific targets.

- The visualizations aim to provide a comprehensive understanding of terrorist activities by analyzing the proportions of target types, success rates, and the interplay between attack types and weapon types across the most frequently targeted institutions. This holistic approach facilitates insights into the diverse strategies employed by terrorist organizations against different target types.

3. What is/are the insight(s) found from the chart?

- The predominant targets for terrorist attacks include Private Citizens and Properties, Military, Police, Government, and Business institutions. This trend is reflected in both the frequency of attacks and the resulting casualties. Suicide attacks, on the other hand, are more concentrated on Military and Police targets, followed by Private Properties and Government institutions.

- In terms of success rates, terrorist groups achieve a high success percentage (90% or more) across various target types, except for Government institutions where success rates are comparatively lower. This pattern persists in temporal distributions, with notable spikes in casualties, particularly in the year 2001 attributed to the 9/11 attacks targeting Private Citizens and Properties.

- Examining top attack types for common terrorist targets reveals specific trends:

  - Military experiences Bombings and Armed Assaults.
  - Government faces a combination of Bombings, Assassinations, and Armed Assaults.
  - Business targets are overwhelmingly subjected to Bombings.
  - Police encounters a mix of Bombings and Armed Assaults.
  - Private Citizens endure a combination of Bombings and Armed Assaults.

KEY TAKEAWAYS:

- Terrorists frequently target Private Citizens and Properties, Military, and Police, with Government and Business institutions also falling victim to attacks.

4. Will the gained insights help creating a positive business impact?

Recognizing the prevalent targets and attack patterns provides valuable insights for enhancing security measures and devising targeted counter-terrorism strategies. By understanding the preferred targets and tactics of terrorist groups, authorities can tailor preventive measures, allocate resources efficiently, and strengthen the protection of vulnerable sectors. Ultimately, these insights empower security agencies and policymakers to proactively mitigate the impact of terrorist activities and contribute to the overall safety of communities and critical institutions.

2. Local Analysis

> Building upon the insights obtained in the Demographic (National) Analysis section, a closer examination will be directed towards Iraq, Afghanistan, Pakistan, and India. These countries, identified with the highest incidence of terrorist activities, are chosen for an in-depth analysis to glean more nuanced and focused insights.

Basic functions for making graphs

```
"""

def generate_plots(cg_name, plot_type, cg_dict, cg_yearly_sum, cg_attack_dict,
cg_weapon_dict, cg_target_dict, cg_gc_dict, cg_city_dict):
    '''
    Generate specific plots for a particular group/country, including:
    (a) Annual total and successful events
    (b) Annual casualties
    (c) Annual events across top 5 countries/groups
    (d) Proportions of each attack type
    (e) Proportions of each weapon type
    (f) Proportions of each target type
    (g) Percentage of suicide attacks
    (h) Top 10 cities of terrorist activity in the country (only if the plot_type is country)

    Parameters:
    - cg_name (str): The country/group name.
    - plot_type (str): The plot type ('country' or 'group').
    - cg_dict (dict): Dictionary containing data for terrorist events.
    - cg_yearly_sum (dict): Dictionary containing annual summaries.
    - cg_attack_dict (dict): Dictionary containing attack type data.
    - cg_weapon_dict (dict): Dictionary containing weapon type data.
    - cg_target_dict (dict): Dictionary containing target type data.
    - cg_gc_dict (dict): Dictionary containing data for top 5 countries/groups.
    - cg_city_dict (dict): Dictionary containing city-wise terrorist activity data.
```

```
    Returns:
    None
    '''

    # Yearly total and successful events
    title_text = 'in ' + cg_name if plot_type == 'country' else 'by ' + cg_name
    cg_events = cg_dict[cg_name]['year'].value_counts().sort_index()
    cg_success = cg_yearly_sum[cg_name]['success'].sort_index()
    fig = line_plots([cg_events, cg_success], ['Number of terrorist events', 'Number of successful
events'])
    fig.update_layout(title = f'Annual terrorist events and successful terrorist events {title_text}',
yaxis_title = 'Number of events', xaxis_title = 'Year',
                      autosize=True,template="simple_white")
    fig.show()

    # Yearly casualties
    yr_kw_list = [cg_yearly_sum[cg_name][var] for var in ['killed', 'wounded']]
    fig = line_plots(yr_kw_list, ['Number of people killed', 'Number of people wounded'])
    fig.update_layout(title = f'Annual casualties {title_text} due to terrorism', yaxis_title =
'Number of people', xaxis_title = 'Year',
                      autosize=True,template="simple_white")
    fig.show()

    # Yearly terrorist attacks across top 5 countries/groups
    var = 'country' if plot_type == 'group' else 'group'
    cg_names = list(cg_gc_dict[cg_name].head(5).index)
```

```
    cg_year = {cou_group: cg_dict[cg_name][cg_dict[cg_name][var] ==
cou_group]['year'].value_counts().sort_index() for cou_group in cg_names}

    multi_cat_lineplot(cg_names, cg_year, f'{inflect.engine().plural(var)} (top 5) {title_text}')


    # Proportions of each attack type, weapon type and target type in the country or by the group

    awt_list = [cg_gc_dict[cg_name][:10], cg_attack_dict[cg_name], cg_weapon_dict[cg_name],
cg_target_dict[cg_name]]

    texts = [f'of top 10 {inflect.engine().plural(var).capitalize()}', 'Attack Type', 'Weapon Type',
'Target Type']

    for df, text in zip(awt_list, texts):

        fig = go.Figure(data = go.Pie(values = df.values, labels = df.index))

        fig.update_layout(title = f'Proportions of each {text} on total terrorist events {title_text}',
autosize=True )

        fig.show()


    # Proportions of suicide attacks

    suicide_plot = cg_dict[cg_name]['suicide'].map({0: 'Non-suicide attack', 1: 'Suicide
attack'}).value_counts()

    fig = go.Figure(data = go.Pie(values = suicide_plot.values, labels = suicide_plot.index))

    fig.update_layout(title = f'Percentage of suicide attacks {title_text}', autosize=True)

    fig.show()


    # Top 10 cities in terms of terrorist attacks in the country

    if plot_type == 'country':

        top10_cities = cg_city_dict[cg_name].head(10)

        fig = go.Figure(data = go.Bar(x = top10_cities.index, y = top10_cities.values,
marker=dict(color='#bdcf32')))
```

```
    fig.update_layout(title = f'Top 10 cities with terrorist activities in {cg_name}', xaxis_title = 'City', yaxis_title = 'Number of terrorist events',
                      autosize=True,template="simple_white")
    fig.show()


def generate_heatmap(cg_name, country_dict, geojson_path, cities=False, zoom_level=5):
  '''
  Generates a geographic heatmap of terrorist attacks in the country or by the group, with optional markers for the top 5 cities.

  Parameters:
  - cg_name (str): The country/group name.
  - country_dict (dict): Dictionary containing the complete dataset of the specific country/group.
  - geojson_path (str): Path to the GeoJSON file.
  - cities (bool): Boolean for whether the cities should be plotted or not (default is False).
  - zoom_level (int): Initial zoom state (default is 5).

  Returns:
  None
  '''

  # Top 5 cities with terrorist activities
  top5_cities = country_dict[cg_name]['city'].value_counts().head(5).index
  if 'Unknown' in top5_cities:
    top5_cities = country_dict[cg_name]['city'].value_counts().head(5).drop('Unknown').index
```

```
    top5_city_data = country_dict[cg_name].groupby('city').first()[['latitude',
'longitude']].loc[top5_cities]
    coords = [(row['latitude'], row['longitude']) for _, row in top5_city_data.iterrows()]
    coordinates = (top5_city_data.head(1)['latitude'].values[0],
top5_city_data.head(1)['longitude'].values[0])

    # Create a map centered around the country/city with top activity
    country_map = folium.Map(location=coordinates, zoom_start=zoom_level, tiles=None,
control_scale=True,
                             prefer_canvas=True, background_color='white', scrollWheelZoom=False)

    # Load GeoJSON data for country boundaries
    gdf = gpd.read_file(geojson_path)
    state_style = {'fillColor': 'white', 'color': 'black', 'weight': 1, 'fillOpacity': 1}
    folium.GeoJson(gdf, style_function=lambda feature: state_style).add_to(country_map)

    # Marking the coordinates of the top 5 cities
    if cities:
        c = 1
        for label, coord in zip(top5_cities, coords):
            folium.Marker(location=coord, icon=folium.Icon(color='red'), popup=f'{c}.
{label}').add_to(country_map)

    # Plotting the heatmap on the map
    country_map.add_child(HeatMap(data=country_dict[cg_name][['latitude',
'longitude']].dropna(),
                                  radius=10, blur=8, max_zoom=1))
```

    # Display the map
    display(country_map)

"""### **7. Iraq**

1. Graphs
"""

# Generating basic study plots for Iraq
generate_plots("Iraq", "country", *country_data)

#Link for geojson file
geojson_path='https://github.com/wmgeolab/geoBoundaries/raw/905b0ba/releaseData/gbOpen/IRQ/ADM2/geoBoundaries-IRQ-ADM2_simplified.geojson'
# Generate a heatmap for Iraq
generate_heatmap('Iraq', country_dict, geojson_path, cities=True, zoom_level=7)

"""##### 2. Why did you pick the specific chart?

- In addition to the barplots, lineplots, and pie charts employed earlier, this section introduces a geographical heatmap with markers for the top-5 cities in the selected country, showcasing the areas with the highest frequency of terrorist attacks.

- Utilizing the folium library for geographical heatmaps offers an interactive interface, allowing zooming and panning. The inclusion of markers with pop-ups on a simple click ensures a compact and user-friendly plot. Additionally, the heatmap dynamically updates at various zoom levels, providing insights into different levels of regional attack distribution.

3. What is/are the insight(s) found from the chart?

Key observations from the plots include:

- Iraq's Shift in Terrorist Activities: Iraq witnessed a substantial surge in terrorist activities primarily in the 21st century, contrasting with minimal occurrences in the 20th century. This shift is evident in the sharp rise in casualties, notably in 2007 (notable for the Yazidi community bombings during the Iraq war) and 2014 (marked by 31,000 casualties from around 4,000 attacks), coinciding with the onset of the Iraqi war and the territorial expansion of ISIL.

- Dominance of Unknown Groups and Rise of ISIL: Around 75% of terror attacks in Iraq were perpetrated by unknown groups, but the landscape changed significantly with the emergence of ISIL in 2014. ISIL became a prominent terror group, contributing significantly to the increased number of attacks and casualties in the region.

- Prevalent Attack Types and Targets: The preferred attack types in Iraq are Bombings and Armed Assaults, corresponding to the frequent use of Explosives and Firearms as weapons. The main targets of terror groups in Iraq include Private Citizens & Properties, Police, Military, Government, and Business, in descending order.

- Significance of Suicide Attacks: Suicide attacks constituted a noteworthy aspect of Iraqi terror incidents, with approximately 1 in every 10 attacks being suicide bombings.

- Hotspot Cities: Baghdad stands out as the predominant hotspot for terrorist attacks in Iraq, followed by Mosul, Kirkuk, and Baqubah. These cities are visually represented in the heatmap, highlighting the frequency of terrorist incidents across the map of Iraq.

4. Will the gained insights help creating a positive impact?

The insights garnered provide a foundation for targeted counter-terrorism strategies, emphasizing the evolving nature of threats in Iraq. Security measures can focus on mitigating the specific risk factors identified, including the rise of ISIL, prevalent attack types, and high-risk target areas. Additionally, understanding the geographical concentration of terrorist activities enables more effective deployment of resources and proactive security measures in identified hotspot cities.

8. Afghanistan

1. Graphs

"""

```
# Generating basic study plots for Afghanistan

generate_plots('Afghanistan', 'country',*country_data)


# Link for geojson file

geojson_path='https://github.com/wmgeolab/geoBoundaries/raw/905b0ba/releaseData/gbOpen/AFG/ADM0/geoBoundaries-AFG-ADM0_simplified.geojson'

# Generate a heatmap for Afghanistan

generate_heatmap('Afghanistan', country_dict, geojson_path, cities=True, zoom_level=6)
```

"""##### 2. Why did you pick the specific chart?

As previously detailed in the Iraq's sub-section, these graphs contains the necessary information for effective data representation.

3. What is/are the insight(s) found from the chart?

From the visualizations, the following observations can be made:

- Afghanistan, like Iraq, experienced a minimal presence of terrorist attacks in the 20th century and witnessed a surge in terrorist activities in the 21st century, as evident from the rise in casualties.

- The majority of terror attacks in Afghanistan are attributed to the Taliban, constituting approximately 58% of the incidents, marking their ascendancy in the late 1990s and early 2000s. Some attacks remain unclaimed, contributing to the category of Unknown terrorist groups.

- Similar to Iraq, the prevalent attack types in Afghanistan are Bombings and Armed Assaults, aligning with the frequent use of Explosives and Firearms as the most employed weapon types.

- The primary targets of terror groups in Afghanistan are the Police, followed by Private Citizens & Properties, Military, and Government.

- Suicide attacks are a recurring feature in Afghani terror incidents, constituting about 1 in every 10 attacks.

- Kabul and Kandahar emerge as prominent terrorist hotspots within Afghanistan, followed by other cities such as Jalalabad and Ghazni. These cities are also highlighted in the heatmap, illustrating the frequency of terrorist attacks across the Afghan map.

4. Will the gained insights help creating a positive impact?

The insights derived from Afghanistan's analysis can contribute to informed decision-making and strategic planning in counter-terrorism efforts. Understanding the evolving patterns of terrorist activities, key perpetrators like the Taliban, prevalent attack types, and target priorities can aid in crafting more effective security measures and response strategies. Additionally, the focus on specific hotspots, such as Kabul and Kandahar, provides valuable intelligence for targeted interventions, potentially minimizing the impact of terrorist incidents and fostering a safer environment.

9. Pakistan

1. Graphs
"""

```
# Generating basic study plots for Pakistan
generate_plots("Pakistan", "country", *country_data)

# Link for geojson file
geojson_path = 'https://github.com/wmgeolab/geoBoundaries/raw/905b0ba/releaseData/gbOpen/PAK/ADM0/geoBoundaries-PAK-ADM0_simplified.geojson'
# Generate a heatmap for Pakistan
generate_heatmap('Pakistan', country_dict, geojson_path, cities=True, zoom_level=6)
```

"""##### 2. Why did you pick the specific chart?

Same reseons as mentioned in earlier sections

3. What is/are the insight(s) found from the chart?

Observing the data visualizations:

- Pakistan experienced a surge in terrorist activities primarily in the 21st century, with a notable peak in 1995 during the 20th century, attributed to the Karachi car bombings. The casualties plot reflects this trend, with a local peak in 1987 due to significant events.

- The majority of terror attacks are attributed to unknown groups (around 80%), with a subsequent rise in Tehrik-i-Taliban Pakistan (TTP) from the early 2000s onwards.

- Bombings and armed assaults remain the most prevalent attack types, aligning with the common use of explosives and firearms as weapons.

- The primary targets of terror groups in Pakistan include Private Citizens & Properties, Police, Military, Government, and Business, in that sequence.

- Unlike Iraq and Afghanistan, suicide attacks are less frequent in Pakistani terror incidents, accounting for about 3% of total attacks.

- Karachi emerges as the city most affected by terrorist attacks in Pakistan, followed by Peshawar, Quetta, and Lahore, as illustrated in the heatmap depicting attack frequency within the country.

4. Will the gained insights help creating a positive impact?

The insights gained provide a nuanced understanding of Pakistan's terrorist landscape, enabling focused counter-terrorism efforts. Identification of key actors like Tehrik-i-Taliban Pakistan (TTP), predominant attack types, and target priorities informs security strategies. Recognizing the lower incidence of suicide attacks helps allocate resources effectively. Moreover, hotspot identification, particularly in Karachi, aids targeted security measures for enhanced public safety.

10. India

```
##### 1. Graphs
"""


# Generating basic study plots for India
generate_plots("India", "country", *country_data)


# Link for geojson file
geojson_path=
"https://github.com/wmgeolab/geoBoundaries/raw/de71fe3/releaseData/gbOpen/IND/ADM0/
geoBoundaries-IND-ADM0_simplified.geojson"
# Generate a heatmap for India
generate_heatmap('India', country_dict, geojson_path, cities=True, zoom_level=5)


"""> Considering the extensive landmass of India in comparison to the previous three countries,
we intend to generate individual graphs for top 3 terrorist group (excluding "Unknown") to
enhance our understanding."""


# List of terrorist groups for which a heatmap is generated
groups = ['Communist Party of India - Maoist (CPI-Maoist)', 'Maoists', 'Sikh Extremists']


# Iterate through the list of groups and generate a heatmap if the group data is available for
India; added this because a lot of activity by Maoists is Nepal
for group in groups:
    if group in group_dict and 'group' in group_dict[group].columns and 'India' in
group_dict[group]['country'].values:
        # Print the name of the group in bold to give the heading
        print('\033[1m' + '-' * 80 + '\n' + 'Attacks by '+ group + '\n' + '-' * 80 + '\033[0;0m')
```

    # Generate a heatmap for the selected group in India with cities marked, using specified zoom level

    generate_heatmap(group, group_dict, geojson_path, cities=True, zoom_level=5)

"""##### 2. Why did you pick the specific chart?

The reason of using the types of graphs is same as explained earlier sections

3. What is/are the insight(s) found from the chart?

Analyzing the plots reveals distinct patterns in India's terrorist landscape:

- While the 21st century witnessed a surge in terrorist activities, notable incidents occurred in the late 1980s and early 1990s, with a significant spike post the 2000s.

- The casualty plot reflects an unusual peak in 2006, attributed to the Mumbai train explosions by Lashkar-e-Taiba (LeT), causing a substantial toll. However, the latter half of the plot indicates a decrease in casualties despite a higher number of attacks.

- The majority of attacks are attributed to unknown groups (~45%), with Communist and Maoist entities featuring prominently, particularly from 2005 onward. Sikh Extremists contributed significantly in the 1980s.

- Predominant attack types involve Bombings and Armed Assaults, aligning with the prevalent use of Explosives and Firearms as weapons.

- Target priorities for terror groups include Private Citizens & Properties, Police, Government, Business, and Military, in that sequence.

- India exhibits a relatively lower rate of suicide attacks (0.5%) compared to other countries in the study.

- Srinagar and Imphal emerge as the cities most affected by terrorist attacks, followed by New Delhi and Amritsar. The heatmap visually underscores these regions, further detailing the activities of top terrorist groups in India.

4. Will the gained insights help creating a positive impact?

The gained insights from the analysis of terrorist activities in India provide a foundation for targeted and effective counter-terrorism measures. Understanding historical trends, prevalent attack types, and key targets allows for the development of nuanced strategies. The identification of regions with heightened terrorist activities, as depicted in the heatmap, enables focused security efforts to mitigate risks and enhance public safety. Moreover, recognizing the lower incidence of suicide attacks in India allows for tailored preventive measures, contributing to a more comprehensive and proactive security approach.

3. Analysis of specific terrorist organisations

> As highlighted in the Analysis of Terrorist Organizations, two distinct groups, namely the Taliban and ISIL, demonstrated significant terrorist activities, particularly in the 21st century. Similar to the previous section focusing on countries, these two groups are specifically selected for further in-depth analysis.

11. Taliban

1. Graphs
"""

Generating basic study plots for Taliban

generate_plots('Taliban', 'group', *group_data)

Link for geojson file

geojson_path='https://github.com/Stefie/geojson-world/raw/master/countries.geojson'

Generate a heatmap for activity of Taliban

generate_heatmap('Taliban', group_dict, geojson_path, cities = True, zoom_level = 5)

"""##### 2. Why did you pick the specific chart?

As using the same functions for genrating the graphs the reason also remains the same

3. What is/are the insight(s) found from the chart?

From the visualizations, we can infer that:

- The rise of the Taliban to prominence is evident from the late 1990s, as seen in the increasing total number of terrorist attacks during this period. This trend is mirrored in the casualties plot.

- The Taliban predominantly operates within Afghanistan, with 99.3% of its attacks occurring within the country. This aligns with the temporal trend observed in the Afghanistan country analysis in the previous section.

- The common methods employed by the Taliban for terrorist attacks include Bombings and Armed Assaults/Firearms.

- The primary targets for the Taliban are the Police, followed by the Military, Private Citizens & Property, and Government institutions.

- Approximately 10% of their terrorist attacks involve suicide attacks.

- The heatmap depicting their areas of terrorist activity confirms their almost exclusive focus on Afghanistan, with minimal activity in a few areas of Pakistan.

4. Will the gained insights help creating a positive impact?

The insights gained from the analysis of the Taliban's activities provide a foundation for targeted counter-terrorism efforts. By understanding their predominant methods, preferred targets, and geographic focus, security measures can be tailored to address specific vulnerabilities. This knowledge enables more effective resource allocation and strategic planning to mitigate the impact of the Taliban's activities, fostering a safer environment in the affected regions.

12. Islamic State of Iraq and the Levant (ISIL)

1. Graphs
"""

```
# Generating basic study plots for Islamic State of Iraq and the Levant (ISIL)
generate_plots('Islamic State of Iraq and the Levant (ISIL)', 'group', *group_data)

# Link for geojson file
geojson_path='https://github.com/Stefie/geojson-world/raw/master/countries.geojson'
# Generate a heatmap for activity of ISIL
```

generate_heatmap('Islamic State of Iraq and the Levant (ISIL)', group_dict, geojson_path, cities = True, zoom_level = 5)

"""##### 2. Why did you pick the specific chart?

Reason these graphs is as same as explained earlier

3. What is/are the insight(s) found from the chart?

By the studying the graphs, we can conclude:

- ISIL emerged as a terrorist group in 2013, and their activities and casualties increased until 2017, making them the most active terrorist group according to the latest available data. The majority of their operations occur in Iraq (~86%) and Syria (~10%), with a smaller proportion (1.5%) in Turkey.

- ISIL primarily employs bombings and explosives in their attacks, with only 10% involving firearms. Private citizens and properties are their most targeted category, followed by military, police, and business institutions.

- Approximately 25% of ISIL's attacks utilize suicide tactics. The geographical heatmap confirms their high activity in Iraq, Syria, and certain parts of Turkey. Additionally, it's noteworthy that a discernible proportion of their attacks is focused on coastal regions of various nations.

4. Will the gained insights help creating a positive impact?

The insights gained, particularly regarding ISIL's activities, can contribute to enhancing coastal region security in various nations. Understanding their predominant attack methods and target preferences allows for more targeted and effective counter-terrorism measures. Additionally, this knowledge aids in strengthening international collaboration to address the specific challenges posed by ISIL in Iraq, Syria, and parts of Turkey, fostering a collective and strategic response to mitigate their impact.

13. Brief Analysis of Additional Organizations

Analysis of Specific Terrorist Organizations includes a brief examination of the following groups:

- Al-Qaida
- Shining Path
- Boko Haram

Al-Qaida
"""

```
# Visualizing countries attacked by Al-Qaida
group_name = 'Al-Qaida'
fig = go.Figure(data = go.Pie(values = group_country_dict[group_name].values, labels = group_country_dict[group_name].index))
fig.update_layout(title = f'Countries attacked by {group_name}', autosize = False, width = 1300, height = 550)
fig.show()

# Link for geojson file
geojson_path='https://github.com/Stefie/geojson-world/raw/master/countries.geojson'
# Generate a heatmap for activity of Al-Qaida
generate_heatmap('Al-Qaida', group_dict, geojson_path, cities = True, zoom_level = 3)
```

```
"""##### Shining Path"""

# Visualizing countries attacked by Shining Path
group_name = 'Shining Path (SL)'
fig = go.Figure(data = go.Pie(values = group_country_dict[group_name].values, labels = group_country_dict[group_name].index))
fig.update_layout(title = f'Countries attacked by {group_name}', autosize = False, width = 1300, height = 550)
fig.show()

# Link for geojson file
geojson_path='https://github.com/Stefie/geojson-world/raw/master/countries.geojson'
# Generate a heatmap for activity of Shining Path (SL)
generate_heatmap('Shining Path (SL)', group_dict, geojson_path, cities = True, zoom_level = 5)

"""##### Boko Haram"""

# Visualizing countries attacked by Boko Haram
group_name = 'Boko Haram'
fig = go.Figure(data = go.Pie(values = group_country_dict[group_name].values, labels = group_country_dict[group_name].index))
fig.update_layout(title = f'Countries attacked by {group_name}', autosize = False, width = 1300, height = 550)
fig.show()

# Link for geojson file
geojson_path='https://github.com/Stefie/geojson-world/raw/master/countries.geojson'
```

```
# Generate a heatmap for activity of Boko Haram

generate_heatmap('Boko Haram', group_dict, geojson_path, cities = True, zoom_level = 6)


"""##### What is/are the insight(s) found from the chart?


- Even though Al-Qaida is famous for the 9/11 attacks in NYC, they were most active in Pakistan and Afghanistan.


- Shining Path was almost exclusively active in Peru, working to overthrow the Peruvian government.


- Boko Haram was active in Nigeria and parts of Cameroon.


### **14. Brief Global Overview At a Glance**


##### 1. Graph
"""


# Replace 'nan' values in the 'killed' column with a '0' to avoid errors

eda_df['killed'] = eda_df['killed'].replace(np.nan, 0)


# Define hover dictionary to control hover information

hover_dict = {'killed': True, 'wounded': True, 'city': True, 'attack_type': True, 'weapon_type': True,

            'year': False, 'latitude': False, 'longitude': False}


# Create a scatter geo plot using Plotly Express
```

```
scatter_geo_fig = px.scatter_geo(
    eda_df[eda_df[['latitude', 'longitude', 'year']].notnull().all(axis=1)],
    lat='latitude', lon='longitude', color='attack_type',
    hover_name='country', size='killed',
    animation_frame='year', projection="natural earth", template='simple_white',
    hover_data={'killed': hover_dict['killed'], 'wounded': hover_dict['wounded'],
                'city': hover_dict['city'], 'attack_type': hover_dict['attack_type'],
                'weapon_type': hover_dict['weapon_type'],
                'latitude': False, 'longitude': False }
)

# Update traces for marker size and opacity
scatter_geo_fig.update_traces(marker=dict(size=20, opacity=0.7))

# Update layout with title
scatter_geo_fig.update_layout(title_text='Terrorist Attacks Over Time')

# Display the plot
scatter_geo_fig.show()

"""##### 2. Why did you pick the specific chart?
```

- Geographic Representation: The scatter geo plot is chosen because it provides a geographic representation of terrorist attacks over time. The plot uses latitude and longitude coordinates to position markers on a world map, allowing for a spatial understanding of the distribution of attacks.

- Temporal Analysis: The use of an animated scatter geo plot enables a temporal analysis, showing how terrorist attacks evolve over different years. This animated approach helps in visualizing patterns and trends over time.

5. Solution to Business Objective

What do you suggest to achieve Objective ?

- Formulate targeted counterterrorism strategies by leveraging insights into prevalent attack types and corresponding weapon use, focusing on elements like bombings/explosions and armed assaults to bolster security measures.

- Prioritize the safeguarding of specific targets, including private citizens and properties, military installations, and law enforcement, considering their recurrent targeting by terrorist groups.

- Direct resources towards regions characterized by a high frequency of terrorist activities, such as Iraq and Afghanistan, to effectively address localized security challenges.

- Develop tailored counter-measures against prominent terrorist entities like the Taliban and ISIL, informed by an understanding of their operational methods and geographical areas of activity as highlighted in the insights.

- Implement proactive measures to counter the increasing prevalence of suicide attacks, particularly in specific regions identified through the analysis.

- Promote international collaboration to facilitate intelligence-sharing, synchronize counterterrorism initiatives, and collectively combat global threats posed by terrorism.

Conclusion

In conclusion:

Feature Pre-processing:

1. The dataset presented significant challenges with numerous columns and a high percentage of missing values, particularly in columns related to casualties.

2. Careful selection of features for Exploratory Data Analysis (EDA) was performed, distinguishing primary and secondary features based on their relevance to EDA.

3. Missing values in columns representing the total number of people killed and wounded were not imputed or dropped due to vagueness in the dataset description.

Data Analysis:

1. The 21st century witnessed an exponential increase in terrorist attacks and casualties, particularly in the Middle East, South Asia, and Sub-Saharan Africa, peaking in 2014.

2. Bombings and Armed Assaults were prevalent attack methods, with a notable increase in the percentage of suicide attacks in specific regions during the same period.

3. Private Citizens and Properties, Military, and Police were the primary targets of terrorist attacks.

4. In the 20th century, the United Kingdom, El Salvador, and Peru were identified as terrorist hotspots, with prominent organizations like IRA, FMNL, and Shining Path.

5. In the 21st century, Iraq, Afghanistan, Pakistan, and India emerged as terrorism hotspots, experiencing the highest number of attacks, casualties, and suicide attacks.

6. ISIL stood out as the most active terrorist organization in Iraq, conducting over 4000 attacks since 2013, with a focus on suicide bombings in Baghdad.

7. The Taliban dominated terrorist activities in Afghanistan, primarily targeting Afghan Police and Military.

8. India faced Maoist and Communist attacks in the 21st century, notably in Naxalist-infested states like Jharkhand and Chhattisgarh.

9. The insights gained from this project offer actionable intelligence for security agencies, aiding in decision-making to prevent terrorist attacks.

Future Use:

- The EDA project serves as a template, providing custom-made functions for analyzing various countries and terrorist organizations with ease by adjusting country names as inputs.
"""