**1.1 Visualization of Terrorist Incidents by Country using D3.js**

**Design Choices:**

**Choropleth Map**

* The choropleth map is chosen as the primary visualization method due to its effectiveness in displaying spatial distributions. It allows users to quickly identify geographic areas that experience a higher frequency of terrorist incidents.
* This visualization enables immediate recognition of regions most affected by terrorism, facilitating discussions on security and policy implications.

**Color Scale**

* A sequential red color scale is utilized to indicate the severity of incidents. Darker shades represent higher incident counts, while lighter shades indicate fewer incidents.
* The use of red effectively communicates urgency, and the gradient provides a clear visual differentiation of incident prevalence across countries.

**Interactive Tooltips**

* Tooltips display specific incident counts for each country upon hovering. This feature provides detailed information without cluttering the map.
* Tooltips enhance the map's interactivity, allowing users to engage more deeply with the data by providing contextual information relevant to each country's incident count.

**Year Slider**

* A slider is implemented to filter incidents by year. This interactive element allows users to observe trends over time.
* The ability to filter data dynamically facilitates the analysis of temporal trends and potential correlations with significant historical events.

**Color Legend**

* A color legend is included to provide context for the color scale used in the map.
* The legend enhances the interpretability of the data, allowing users to understand the significance of color variations accurately.

**Instructions for Running the Code**

To view and interact with the choropleth map visualization,we need to setup environment like below

**Setup Instructions**

1. **Create the HTML File**:
   * Save the file with a .html extension, for example,d3\_assignment.html.
2. **Prepare the Data File**:
   * Ensure that the CSV file (region\_5.csv) is accessible. If it is stored locally on our computer, we may need to run a local server to serve the file correctly.
3. **Running a Local Server**:
   * Open a terminal or command prompt.
   * Navigate to the directory where we saved the HTML and CSV files.
   * Use the following command to start a local server (for Python 3):

**python3 -m http.server**

* + This will start a local server at http://localhost:8000.

1. **Accessing the Visualization**:
   * Open our web browser and enter the following URL:

**http://localhost:8000/d3\_assignment.html**

* + The choropleth map visualizing terrorist incidents will be displayed.

1. **Interacting with the Visualization**:
   * Utilize the year slider to filter incidents by year. Adjust the slider to explore how incident counts vary across different years.
   * Hover over countries to display tooltips containing the country name and incident count, and refer to the color legend for interpretation.

**Plots:**

A map of the world

Description automatically generated

**1.2 Top 10 Terrorist Groups Bar Chart:**

**OVERVIEW**

The goal of this project is to create an interactive bar chart using D3.js that displays the top 10 terrorist groups by incident count for a selected year. This chart is implemented in HTML and JavaScript and includes several interactive and animated features. Data is loaded from a CSV file, region\_5.csv, which contains incident data categorized by group name (gname) and year (iyear).

**PROJECT REQUIREMENTS**

1. Primary Functionalities

* Data Aggregation by Group: The gname column is used to aggregate incident counts for each group within the selected year.
* Dynamic Bar Chart: Displays the top 10 groups with the highest incident counts in a horizontal bar chart format.
* Year Selection Dropdown: Allows users to select a specific year, updating the chart to reflect the top 10 groups for that year.
* Animations: The chart includes smooth transitions for changes in year, with a progressive load effect on the bars.
* Color Scale: Bars are colored using a gradient based on incident count, creating a clear visual distinction.
* Hover Effect: A tooltip displays group names and incident counts on hover, with additional bar highlighting.

1. Additional Functionalities

* Error Handling: Displays an alert if a selected year contains fewer than 10 groups.
* Total Incident Count Display: Shows the total incident count for the selected year above the chart.
* Initial Load Animation: When the page is first loaded, bars animate in by increasing their width from zero to the actual value.

**INSTRUCTIONS TO RUN THE CODE**

To view the visualization, follow these steps:

1. Environment Setup: Ensure the index.html file and region\_5.csv are saved in the same project directory.
2. Open with Live Server:
   * If using Visual Studio Code, install the Live Server extension.
   * Right-click on index.html and select Open with Live Server.
3. Accessing the Visualization: The bar chart will load in the browser automatically.
   * Use the dropdown to select different years and observe the animation transitions.
   * Hover over each bar to view a tooltip with detailed information on the group and its incident count.

**CODE EXPLANATION**

1. Data Processing

Data is loaded and parsed from region\_5.csv using D3’s d3.csv() function. The data is filtered by the selected year, and groups are aggregated based on the count of incidents.

1. D3.js Visualization Components

* SVG and Chart Layout: An SVG container is created with specified margins. Horizontal bars are added with a color scale that dynamically changes based on incident count.
* Dropdown Menu: The dropdown menu for selecting the year is populated based on unique years found in the dataset.
* Dynamic Updates: updateChart(year) is the primary function that updates the chart each time a new year is selected.

1. Additional (Optional) Tasks Implementation

* Error Handling: If fewer than 10 groups are found for a selected year, an alert is displayed.
* Total Incident Count Display: The total incident count for each year is shown above the chart.
* Initial Load Animation: On page load, bars animate from a width of zero to their calculated width, creating a smooth load-in effect.

**DESIGN CHOICES**

1. **Color Scale (Yellow-Green-Blue Gradient):**

A d3.interpolateYlGnBu color scale was used to represent incident counts, where lighter shades indicate fewer incidents, and darker shades represent higher counts. This gradient aids in visually differentiating between groups with varying incident levels, making the visualization more accessible.

1. **Animated Bar Transitions:**

Bars grow horizontally from zero on the initial load with a progressive delay effect. This transition provides a smooth experience, emphasizing the rise of the top groups. This approach effectively highlights the groups with the highest number of incidents, enhancing visual engagement.

1. **Interactive Tooltip and Hover Effects:**

Hovering over each bar displays a tooltip with detailed information (group name and incidents), making the data more accessible without cluttering the chart. The color temporarily changes to red on hover, providing immediate visual feedback and guiding user focus to specific groups.

1. **Year Selection Dropdown:**

The dropdown menu allows users to filter data by year, supporting dynamic exploration of trends over time. This functionality enables comparison of incident counts for different groups across various years.

1. **Highlighting the Top Group:**

A pulse animation was applied to the top group to reinforce its significance as the group with the highest incident count. This visual differentiation draws attention to the leading group without detracting from other bars.

1. **Total Incident Count Display:**

Displaying the total incident count for the selected year provides additional context, helping users understand the scale of incidents relative to other years.

1. **Error Message for Fewer Than 10 Groups:**

An error message appears if fewer than 10 groups are available, ensuring the visualization accurately reflects the dataset and avoids user confusion.

**INSIGHTS GAINED**

1. **Comparative Analysis of Group Incidents Across Years:**

This design enables easy identification of the groups with the highest incidents in each selected year, facilitating comparative analysis of trends and highlighting years with particularly high activity for specific groups.

1. **Patterns of Incident Counts Over Time:**

Observing changes in the top groups across years reveals periods of intensified activity by certain groups, potentially reflecting shifts in geopolitical factors or regional unrest.

1. **Highlighting Extremes and Contextual Awareness:**

The gradient color scale allows for quick identification of groups with outlier values, while the total incident count provides context within each year, contributing to a better understanding of overall incident patterns.

These design decisions aim to create an intuitive and informative visualization that supports both a high-level overview and detailed analysis of terrorist incidents across years.

**OUTPUT**

A graph showing a number of people

Description automatically generated

A graph with numbers and text

Description automatically generated with medium confidence

**3. Plotly Visualization: Attack Types Over Time:**

**Code Submission Details**

Python Script: Contains the data preparation and visualization code. This script reads in the dataset, processes it into a usable format for the visualization, and constructs an interactive Plotly figure. The code also includes buttons to allow toggling between the different visualization views.

**Design Choices:**

**Stacked Area Chart:**

A stacked area chart was selected to represent the trends and distribution of different attack types over time. It allows for an easy comparison of how the frequency of each type changes year by year.

Each colored area in the chart represents a unique attack type, with the x-axis showing the year and the y-axis representing the number of incidents.

**Color and Styling:**

Each attack type is assigned a distinct color to differentiate areas and make it easier to identify each type visually.

Hover information is included for each data point, showing the year, attack type, and the count of incidents, which provides detailed information without crowding the visual.

**Interactive Buttons:**

Stacked View: The default view shows attack types stacked over time, making it easy to see how different types contribute to the total number of incidents each year.

Grouped View: This view separates each attack type into its own line, allowing a more direct comparison of each attack type's trend without the influence of stacking.

100% Stacked View: This view converts each year’s data into a percentage of total incidents, highlighting the proportional relationship of attack types and allowing users to observe shifts in dominance among attack types.

**Hovermode and Legend:**

The hover mode is set to `'x unified'`, which groups all hover information by year to provide a comprehensive view of all attack types in a given year.

A legend displays each attack type, allowing users to quickly identify the colors associated with each type.

**Insights Gained from the Visualization:**

**Trend Analysis**: This chart allows for a clear visualization of trends in attack types over time. Some types become more prevalent in certain years, while others may decline, reflecting possible changes in global or regional conflict dynamics.

**Shift in Dominant Attack Types:**The 100% stacked view effectively reveals the proportional shifts in attack types, showing how some types become relatively more or less common over time.

**Comparison Between Attack Types:** By switching to the grouped view, users can see individual trends more clearly, making it easier to identify fluctuations in specific attack types without the influence of stacking.

**Instructions for Running and Viewing the Code**

**1.Requirements:**

Required libraries: `pandas` and `plotly`. Install them using `pip install pandas plotly`.

**2. Data Preparation:**

Place the dataset file (e.g., `region\_5.csv`) in the same directory as the Python script.

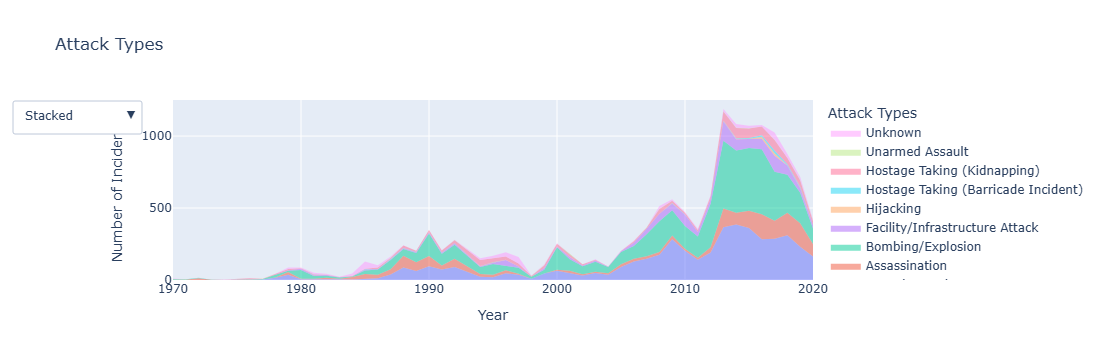
Modify the file path in the script if necessary to match your local setup.

**3. Running the Code:**

Run the Python script to generate the interactive Plotly chart. This will display the chart, allowing you to interact with it.

Interact with the buttons to toggle between different views, and use hover interactions to explore detailed information on each data point.

**Plotly graph:**



**Bokeh Visualization: Target Types and Casualties:**

**1.Importing Libraries**

• bokeh.io.curdoc, bokeh.layouts.column, bokeh.models: These Bokeh components are imported for constructing the interactive web-based plot and the layout structure.

• bokeh.plotting.figure: Used to create the figure for the scatter plot.

• bokeh.transform.factor\_cmap: Used to map categorical data (target types) to colors.

• bokeh.palettes.Category20: A palette with 20 colors to use for differentiating target types.

• pandas: Used to load and process the dataset

**2. Load and Process the Dataset**

• The dataset is loaded using pandas.read\_csv(). Make sure to replace the file path with the correct one for your system.

• The script ensures that the columns 'nkill' (number killed) and 'nwound' (number wounded) are treated as numeric values. Any non-numeric data is coerced to NaN and replaced with 0 using fillna(0).

• A new column, 'total\_casualties', is created to represent the total number of casualties for each incident (sum of killed and wounded).

**3. ColumnDataSource**

• A ColumnDataSource is a Bokeh object used to efficiently share data between plots and widgets. It is initialized with the entire dataset.

• This data source will be updated dynamically when the user interacts with the slider.

**4. Color Mapping**

• factor\_cmap is used to assign colors to each unique target type from the 'targtype1\_txt' column, using the Category20 color palette.

• The min(len(target\_types), 20) ensures that we do not use more colors than are available in the Category20 palette, which has a maximum of 20 colors.

**5.Scatter Plot**

The scatter plot is added to the figure using p.scatter(), where:

• 'nkill' and 'nwound' are used as x and y coordinates.

• 'scaled\_size' is a dynamically calculated size for the markers to reflect the scale of casualties.

• The colors are mapped based on target types, and transparency (alpha) is set to 0.8.

• legend\_field is used to label the legend with target types.

**6.Data Filtering:**

update\_data() is a callback function that filters the data based on the slider's value. It:

• Determines the range of years to display.

• Filters the data to only include incidents within the selected year range.

• Recalculates the scaled\_size of the markers for visibility.

• Updates the ColumnDataSource with the filtered data.

**7.Document Setup:**

• column() is used to layout the scatter plot and slider vertically.

• curdoc().add\_root() adds the layout to the Bokeh document, and curdoc().title sets the title of the document.

**RUN THE CODE:**

python3 -m bokeh serve --show bokehfile.py

**PLOT:**

A graph with many colored dots

Description automatically generated

**DESIGN CHOICES:**

1**. Dataset Selection:** The visualization uses data from the region\_5.csv file, which includes information about terrorist incidents, such as the number of people killed (nkill) and wounded (nwound). These values are aggregated into a total\_casualties metric to understand the overall impact of each incident better.

**2. Scatter Plot Design:**

**• Axes:** The scatter plot maps the number of people killed (nkill) on the x-axis and the number wounded (nwound) on the y-axis. This design helps visualize the correlation between fatalities and injuries, highlighting the severity of different incidents.

• **Point Size:** The size of each scatter point is scaled according to total\_casualties. This allows more significant incidents to stand out, drawing attention to events with higher casualties.

**• Colors:** A color map based on targtype1\_txt (target type) assigns unique colors to each target category. The use of Category20 ensures a vibrant and diverse palette, making it easy to differentiate between target types.

**• Hover Tool:** A hover tool provides additional information about each data point, displaying the target type and casualty counts for better context.

**3. Interactive Elements:**

• **Range Slider:** The year range slider allows users to filter data dynamically, providing the ability to explore trends over time. This enhances user engagement and facilitates the discovery of patterns or changes in attack severity and target preferences.

• **Zoom and Pan Tools:** Users can explore different areas of the plot using pan, wheel zoom, and box zoom tools, making the plot more navigable and interactive.