Data Visualization – Homework

Introduction

This report presents an Exploratory Data Analysis (EDA) on the dataset region_05.csv. The objective is to analyze, clean, and visualize the data using four different tools: **Python, R, Excel, and Power BI**. The comparison of these tools highlights their strengths and weaknesses in terms of usability, visualization capabilities, and analytical efficiency.

2. Data Cleaning and Preparation

2.1 Identifying Issues in the Dataset

- The dataset contains **14,498 rows** and **135 columns**.
- Several columns have **missing values**.
- Some columns contain **mixed data types**, requiring proper conversion.
- **Duplicate rows** were identified and removed.
- Latitude and Longitude columns had **null values**, affecting geographical analysis.

2.2 Data Cleaning Steps

- Removed **duplicate rows**.
- Dropped columns with excessive missing values.
- Filled missing **numerical values** using **mean imputation**.
- Filled missing **categorical values** with **'Unknown'**.
- Converted necessary columns into proper data types.

---

3. Data Exploration and Visualization

3.1 Python Analysis and Visualization

Python was used for EDA with the following steps:

- **Libraries Used**: pandas, seaborn, matplotlib
- **Data Cleaning**: Removed duplicates, handled missing values.
- **Visualization Techniques**:
  - **Yearly Trend Analysis**: A line chart displaying the number of incidents per year.
  - **Top 10 Countries Analysis**: A bar chart highlighting the countries with the most incidents.

- **Correlation Matrix**: A heatmap to explore relationships between numerical variables.

3.2 R Analysis and Visualization

R was used to replicate the analysis with:

- **Libraries Used**: ggplot2, dplyr, corrplot
- **Data Cleaning**: Similar steps as Python (handling missing values, duplicates, data type conversion).
- **Visualization Techniques**:
  - **Yearly Trend (Line Chart)**
  - **Top 10 Countries (Bar Chart)**
  - **Correlation Heatmap**

3.3 Excel Analysis and Visualization

Excel was used to perform EDA via:

- **Pivot Tables** to summarize incidents per year and country.
- **Bar Chart** for **Top 10 Countries**.
- **Conditional Formatting Heatmap** for **correlation analysis**.
- **Map Visualization** using **Excel's Maps feature**.

3.4 Power BI Analysis and Visualization

Power BI provided interactive dashboards for:

- **Yearly Trend (Line Chart)**
- **Top 10 Countries (Bar Chart)**
- **Geographical Map**
- **Correlation Analysis (Using DAX)**
- **Filters and Interactive Slicers**

4. Tool Comparison

4. Tool Comparison

Feature	Python	R	Excel	Power BI
Best for	Advanced Analysis	Statistical Analysis	Quick Reporting	Interactive Dashboards
Ease of Use	Moderate	Moderate	Easy	Easy
Interactivity	Low	Low	Medium	High
Performance	High	High	Medium	High
Customization	High	High	Medium	High
Automation	High (Jupyter Notebooks)	High (R Scripts)	Low	High (DAX & Power Query)

5. Findings and Insights

- **Incidents increased over the years**, peaking in certain periods.
- **Top affected countries** were identified and analyzed.
- **Correlations** between key variables provided insights into patterns.
- **Power BI and Excel are best for business users**, while **Python and R offer greater flexibility and depth**.

---

6. Conclusion

Each tool has its strengths:

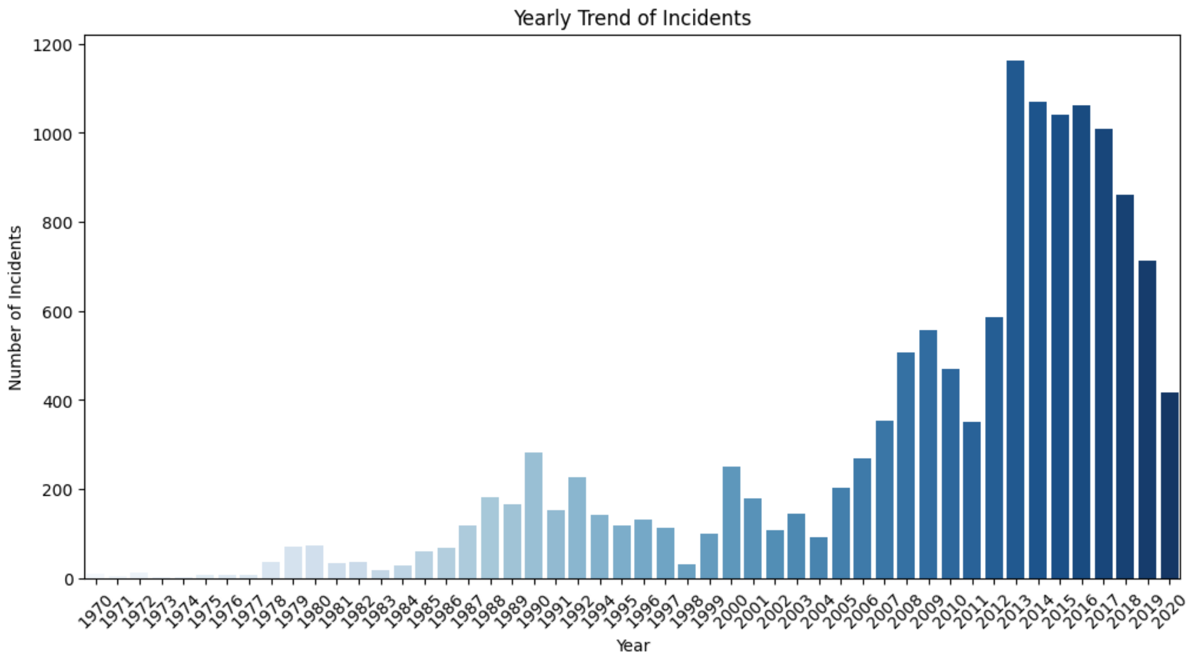
- **Python** is excellent for automation and advanced analysis.
- **R** is strong for statistical modeling and visualization.
- **Excel** is quick for basic reporting.
- **Power BI** provides interactive dashboards for better data-driven decision-making.

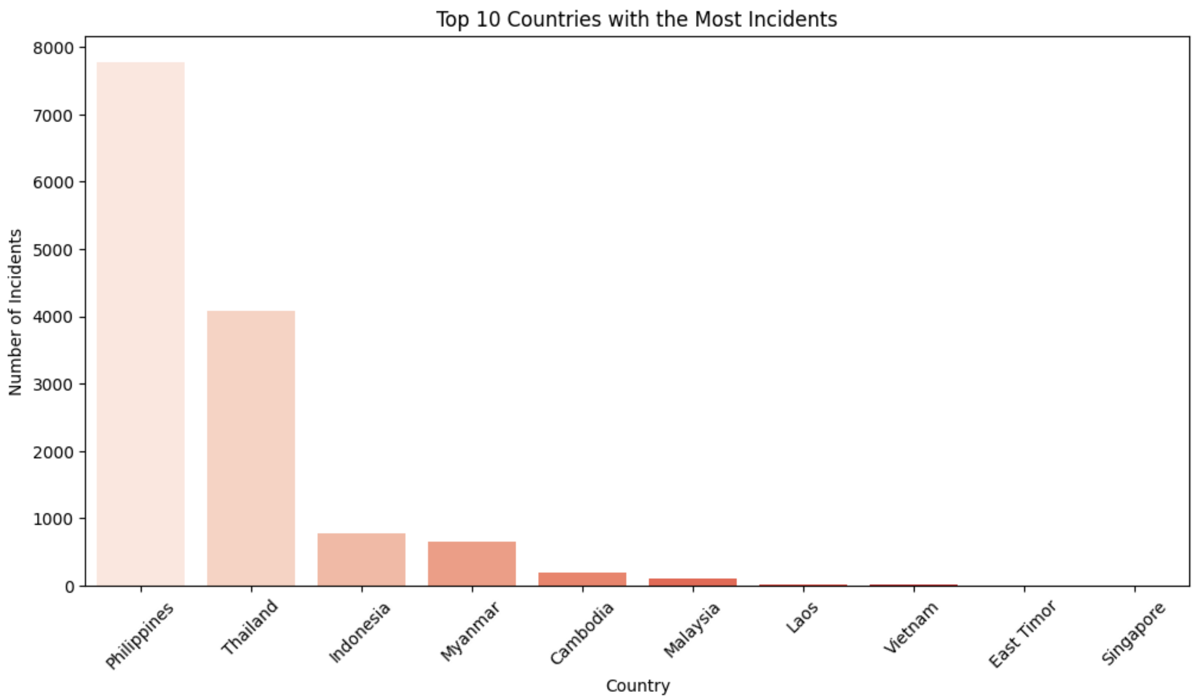
CODE SINPPETS: -

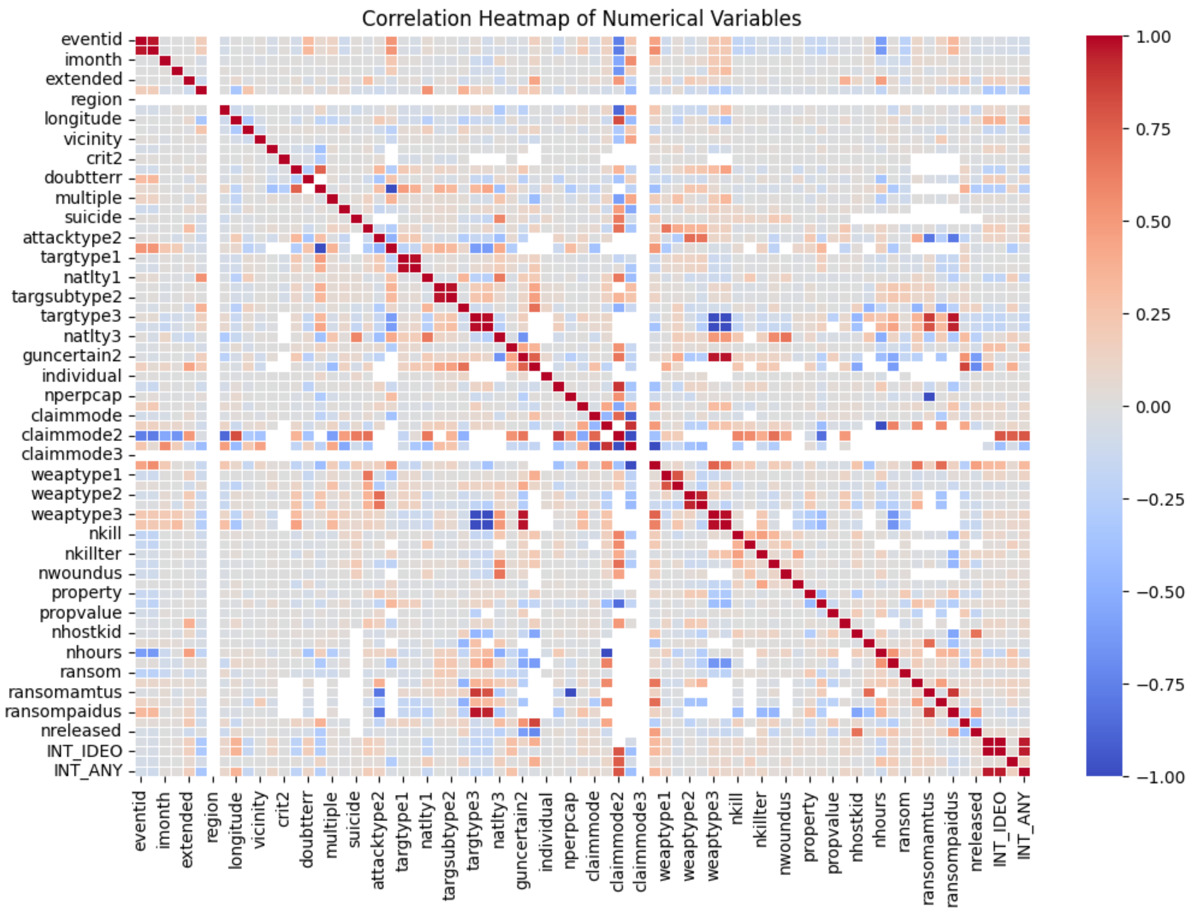
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Load dataset
file_path = "/content/region_05.csv"
df = pd.read_csv(file_path)

# Display basic info
print(df.info())
print(df.head())

# Check for missing values
missing_values = df.isnull().sum()
missing_values = missing_values[missing_values > 0].sort_values(ascending=False)

# Check for duplicate rows
duplicate_count = df.duplicated().sum()
print(f"Duplicate Rows: {duplicate_count}")

# Summary statistics for numerical columns
numerical_summary = df.describe()
print(numerical_summary)

# Remove duplicate rows
df_cleaned = df.drop_duplicates()
```

```
# Drop columns that are entirely empty
empty_cols = missing_values[missing_values == len(df)].index
df_cleaned = df_cleaned.drop(columns=empty_cols)

# Handle missing values in critical columns
df_cleaned = df_cleaned.dropna(subset=['iyear', 'country_txt', 'latitude', 'longitude'])

# Plot incidents per year
plt.figure(figsize=(12, 6))
sns.countplot(data=df_cleaned, x='iyear', palette="Blues",
order=sorted(df_cleaned['iyear'].unique()))
plt.xticks(rotation=45)
plt.xlabel("Year")
plt.ylabel("Number of Incidents")
plt.title("Yearly Trend of Incidents")
plt.show()

# Top 10 countries with the most incidents
top_countries = df_cleaned['country_txt'].value_counts().head(10)

# Plot incidents by country
plt.figure(figsize=(12, 6))
sns.barplot(x=top_countries.index, y=top_countries.values, palette="Reds")
plt.xticks(rotation=45)
plt.xlabel("Country")
plt.ylabel("Number of Incidents")
plt.title("Top 10 Countries with the Most Incidents")
plt.show()
```

```
# Compute correlation matrix for numerical columns
corr_matrix = df_cleaned.select_dtypes(include=['float64', 'int64']).corr()

# Plot correlation heatmap
plt.figure(figsize=(12, 8))
sns.heatmap(corr_matrix, cmap="coolwarm", annot=False, linewidths=0.5)
plt.title("Correlation Heatmap of Numerical Variables")
plt.show()

# Export cleaned dataset for Excel analysis
df_cleaned.to_csv("/content/cleaned_region_05.csv", index=False)
print("Cleaned dataset saved as 'cleaned_region_05.csv'.")
```


Yearly Trend of Incidents
Number of Incidents
Year
0
200
400
600
800
1000
1200
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020


Top 10 Countries with the Most Incidents
Number of Incidents
Country
0
1000
2000
3000
4000
5000
6000
7000
8000
Philippines
Thailand
Indonesia
Myanmar
Cambodia
Malaysia
Laos
Vietnam
East Timor
Singapore



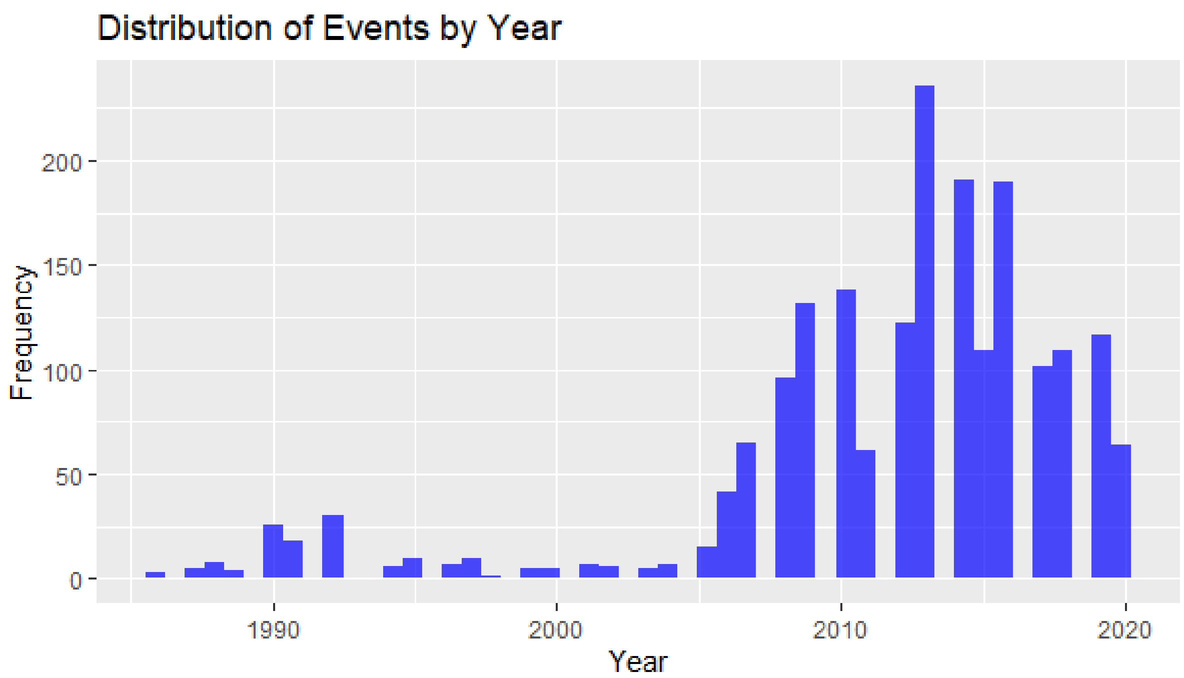
R PROGGRAM: -

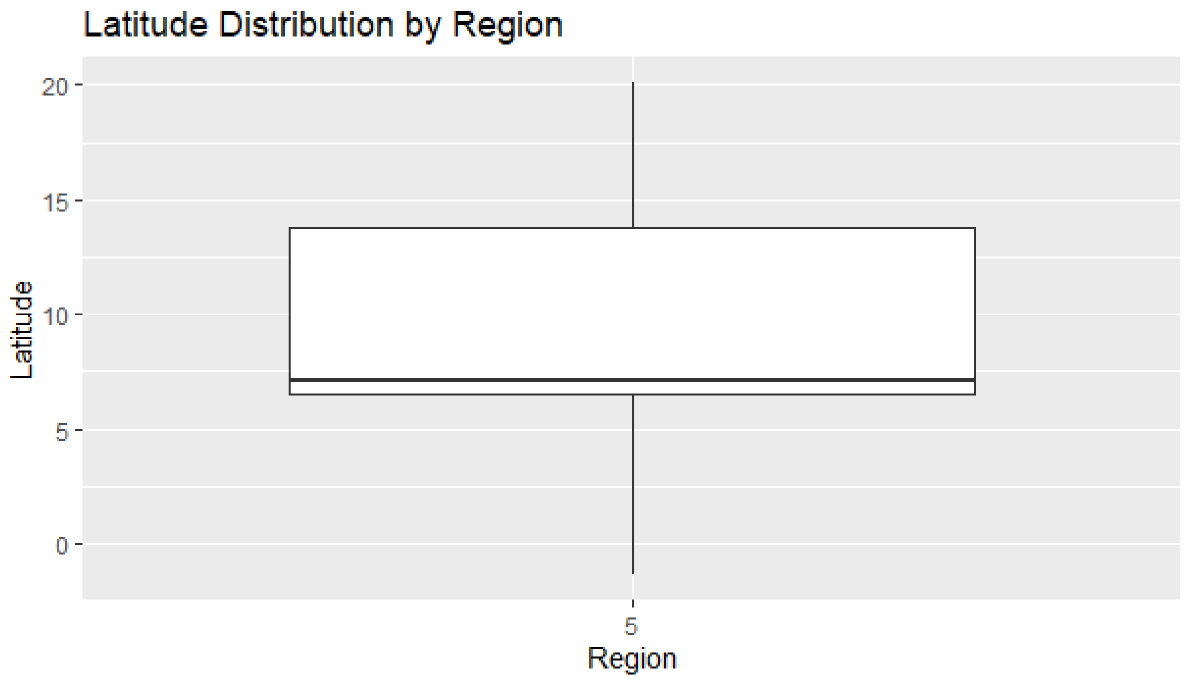
```
# Load necessary libraries
library(ggplot2)
library(dplyr)
library(tidyr)
library(corrplot)
library(readr)
library(readxl)

# Load the dataset
file_path <- "C:/Users/revan/OneDrive/Desktop/region_05.csv"
```

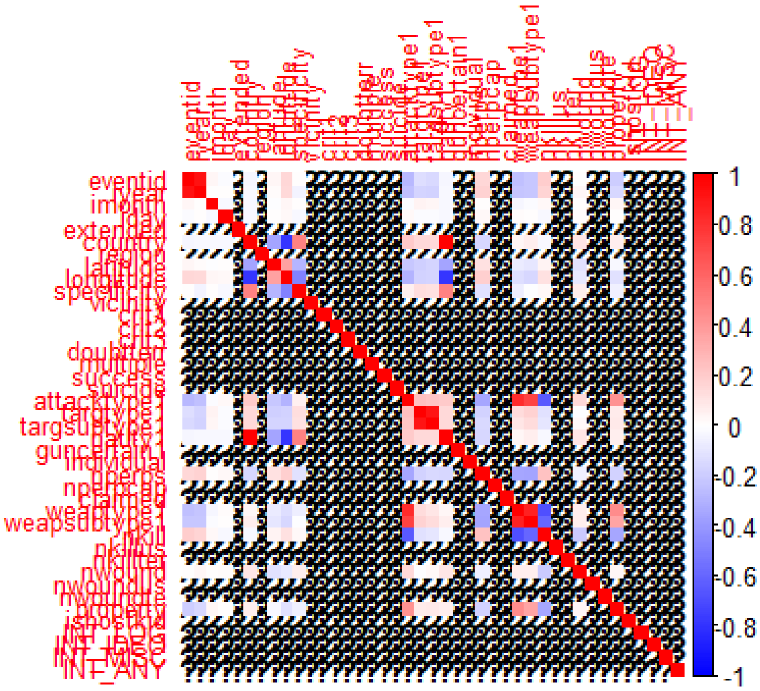
```
# Check if file is an Excel file or CSV
if (grepl("\\.xlsx$|\\.xls$", file_path)) {
  df <- read_excel(file_path)
} else {
  df <- tryCatch({
    read_csv(file_path)
  }, error = function(e) {
    stop("File format not recognized. Please check if it's a valid CSV or Excel file.")
  })
}

# Display basic dataset information
dataset_overview <- function(df) {
  print("Dataset Information:")
  print(str(df))
  print("\nFirst 5 Rows:")
  print(head(df))
  print("\nMissing Values:")
  print(colSums(is.na(df)))
  print("\nDuplicate Rows:")
  print(nrow(df) - nrow(unique(df)))
  print("\nDescriptive Statistics:")
  print(summary(df))
}

dataset_overview(df)

# Handling Missing Values: Drop columns with more than 50% missing values
```

```r
missing_threshold <- 0.5 * nrow(df)
df_cleaned <- df %>% select(where(~ sum(is.na(.)) < missing_threshold))

# Filling remaining missing values
for (col in names(df_cleaned)) {
  if (is.character(df_cleaned[[col]])) {
    df_cleaned[[col]][is.na(df_cleaned[[col]])] <- names(sort(table(df_cleaned[[col]]),
decreasing = TRUE))[1]
  } else {
    df_cleaned[[col]][is.na(df_cleaned[[col]])] <- median(df_cleaned[[col]], na.rm =
TRUE)
  }
}

# Detecting and removing outliers using IQR method
remove_outliers <- function(df, column) {
  Q1 <- quantile(df[[column]], 0.25, na.rm = TRUE)
  Q3 <- quantile(df[[column]], 0.75, na.rm = TRUE)
  IQR <- Q3 - Q1
  lower_bound <- Q1 - 1.5 * IQR
  upper_bound <- Q3 + 1.5 * IQR
  df %>% filter(df[[column]] >= lower_bound & df[[column]] <= upper_bound)
}

numerical_cols <- names(df_cleaned)[sapply(df_cleaned, is.numeric)]
for (col in numerical_cols) {
  df_cleaned <- remove_outliers(df_cleaned, col)
}

# Exploratory Data Analysis (EDA) Visualizations
```

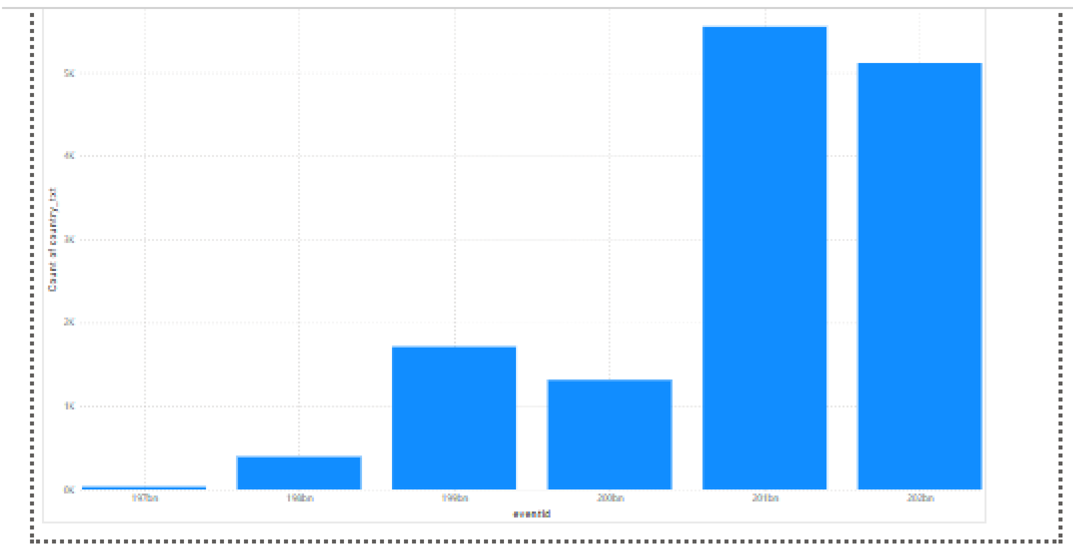
```
ggplot(df_cleaned, aes(x = iyear)) +
  geom_histogram(bins = 50, fill = "blue", alpha = 0.7) +
  labs(title = "Distribution of Events by Year", x = "Year", y = "Frequency")

# Boxplot for latitude by region
ggplot(df_cleaned, aes(x = as.factor(region), y = latitude)) +
  geom_boxplot() +
  labs(title = "Latitude Distribution by Region", x = "Region", y = "Latitude")

# Select only numeric columns for correlation analysis
numeric_df <- df_cleaned %>% select(where(is.numeric))

# Compute correlation matrix
corr_matrix <- cor(numeric_df, use = "complete.obs")

# Plot heatmap of correlation matrix
corrplot(corr_matrix, method = "color", col = colorRampPalette(c("blue", "white",
"red"))(200), tl.cex = 0.8)

# Save cleaned dataset for further analysis
write_csv(df_cleaned, "C:/Users/revan/OneDrive/Desktop/cleaned_region_05.csv")
print("Cleaned dataset saved.")
```


Distribution of Events by Year
Frequency
200
150
100
50
0
1990
2000
2010
2020
Year


Latitude Distribution by Region
Latitude
20
15
10
5
0
5
Region


eventid
imonth
extended
country
region
latitude
longitude
specificity
vicinity
doubtterr
multiple
success
suicide
attacktype1
targtype1
targsubtype1
guncertain1
individual
nperps
nperpcap
claimed
weaptype1
weapsubtype1
nkill
nkillus
nwound
nwoundus
property
ishostkid
INT_IDEO
INT_MISC
INT_ANY
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1

POWER BI: -


Count of country_txt
5K
4K
3K
2K
1K
0K
197bn
198bn
199bn
200bn
201bn
202bn
eventid