Report: Global Terrorism Analysis Using Machine Learning

---

1. Introduction

1.1 Project Overview

This project focuses on analyzing global terrorism data using **Decision Trees** and **Random Forest algorithms**, including a **custom Random Forest implementation from scratch**. The primary objective is to predict **the success of terrorist attacks** based on historical data and visualize attack locations on an **interactive map**.

1.2 Dataset Description

- **Dataset Source**: Global Terrorism Database (50% sampled dataset for efficiency)

  (Note: The original dataset is also there in compressed form but the accuracy mentioned of that data is also mentioned since the file size is to big even to upload on Github, so used the above as sample data but large dataset can also be used as it has been saved in the form of .7z. the main large dataset can be extracted from there.)

- **Number of Features**: 135 (original dataset, reduced in preprocessing)
- **Selected Features**:
  - **Temporal**: `Year`, `Month`, `Day`
  - **Geographical**: `latitude`, `longitude`, `country_txt`, `region_txt`
  - **Attack Details**: `targtype1`, `attacktype1`, `weaptype1`
  - **Casualties**: `nkill`, `nwound` (combined into `Casualties`)
  - **Target Group**: `gname`
  - **Outcome**: `success` (target variable for classification)

---

2. Data Preprocessing

2.1 Handling Missing Values

- Numerical features (`nkill`, `nwound`) were **filled with 0**.
- Categorical features (`country_txt`, `region_txt`, `gname`) were **filled with the mode (most frequent value).**
- The dataset was reduced by filtering out **irrelevant records**.

2.2 Feature Engineering

- **New Feature:** `Casualties = nkill + nwound`
- Categorical features were **label encoded** (e.g., `gname` was converted to numerical values for model processing).

2.3 Handling Class Imbalance

- **Issue**: The dataset had an **imbalance** in the `success` column.
- **Solution**: Used **SMOTE (Synthetic Minority Over-sampling Technique)** to balance the dataset before training.

---

3. Machine Learning Models

3.1 Decision Tree Classifier

- Implemented a **custom Decision Tree algorithm and helper_functions** to make custom Random Forest Classifier work a
- Used **entropy** as the splitting criterion.
- Stopped splitting when:
  - A node was pure (only one class).
  - The depth reached the maximum threshold.

3.2 Random Forest Classifier

- **Implemented both:**
  1. **Scikit-learn's RandomForestClassifier**
  2. **Custom Random Forest (from scratch)**
- **How Random Forest Works:**
  - **Bootstrapping**: Random sampling of data to train each decision tree.
  - **Feature Subsampling**: Each tree is trained on a random subset of features.
  - **Aggregation**: The final prediction is based on a majority vote.
- **Comparison of Accuracy**:
  - **Sklearn Random Forest Classifier Accuracy**: Higher due to optimized implementation.
  - **Custom Random Forest Accuracy**: Slightly lower, demonstrating the manual logic of decision tree ensembles.

3.3 Model Evaluation Metrics

- **Accuracy Score**: Measures the overall correctness of the model.
- **F1 Score**: Provides a balance between precision and recall.

4.1 Accuracy and F1 Score Comparison

Results:

a. Using big dataset:

Model	Accuracy	F1 Score
Scikit-learn Random Forest	94.1%	94.1%
Custom Random Forest	87.8%	87.8%

b. Using sample dataset:

Model	Accuracy	F1 Score
Scikit-learn Random Forest	94.2%	94.2%
Custom Random Forest	85.3%	85.3%

4.2 Visualization & Mapping

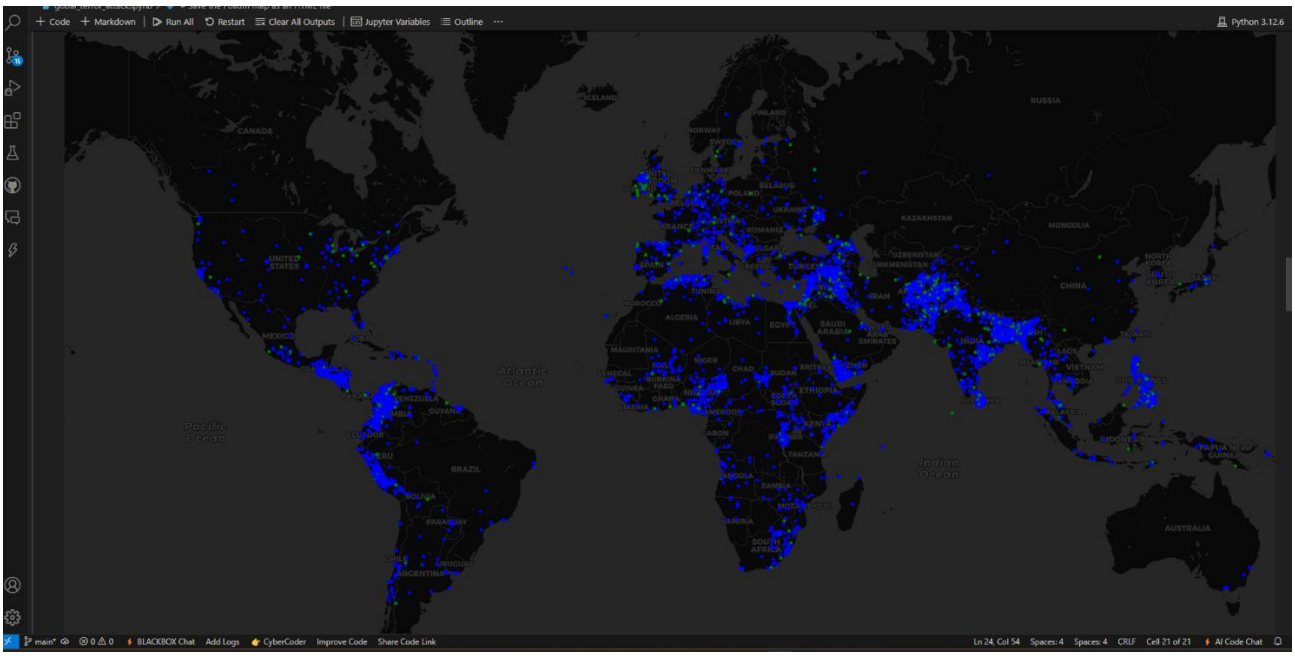
4.2.1 Data Visualization

- **Top 20 Terrorist Groups by Number of Attacks** were visualized using **bar plots**.
- **Casualties per Country** were displayed to show which countries suffered the most attacks.

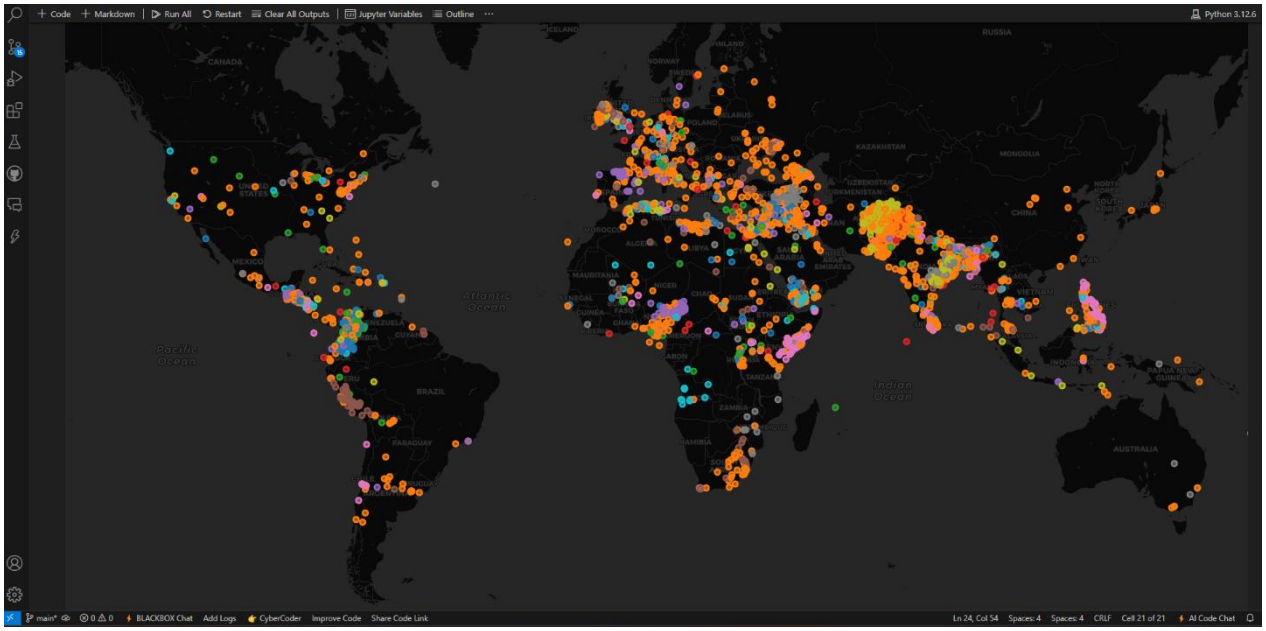
4.2.2 Interactive Map Visualization

- **Plotted global attack locations** using **Folium**.
- **Color Coding by Terrorist Group (`gname`)**:
  - Each **terrorist group was assigned a unique color**.
  - **Markers were displayed** at each attack location.
- **Map Popups Displayed:**
  - **Country**
  - **Terrorist Group (`gname`)**
  - **Predicted Attack Outcome (Successful/Unsuccessful)**

Below are the images of the observation made and also in order attain this code needs to be run as this consumes lots of space.
a.) Map visualization for attacks held with number of casualties.



b.) Predicted Terror groups with success of attempts



4.2.3 Downloading the Map Output

- **Saved the map as an HTML file (`terror_map.html`)** to allow offline viewing. Run the html file to view the plotted map.
- **Converted HTML to an image using Selenium (optional for visualization sharing).**

---

5. Conclusion & Future Improvements

5.1 Key Findings

- The **Scikit-learn Random Forest Classifier** provided the best accuracy.
- **Custom Random Forest implementation** demonstrated how decision trees work internally.
- **Mapping terrorist groups visually** provided valuable insights into attack patterns.

5.2 Future Enhancements

- **Hyperparameter Tuning**: Optimize `max_depth`, `min_samples_split`, etc.

- **Geospatial Clustering**: Use K-Means or DBSCAN for attack hotspot detection.
- **Time-Series Analysis**: Predict future attacks based on historical trends.
- **Ensemble Methods**: Try **Gradient Boosting (XGBoost, LightGBM)** for better predictive performance.

---

6. Final Thoughts

This project demonstrates the power of **machine learning in analyzing terrorism data**. By leveraging **decision trees, random forests, and visualization tools**, we provided **actionable insights into global terrorism trends**. The approach can be expanded for use in **security analysis and threat prediction models**.