To Identify Terrorist Events using Event Triggers

Imports

↳ 10 cells hidden

Global Objects

↳ 2 cells hidden

Exploratory Data Analysis

↳ 19 cells hidden

Prepare data for classification

↳ 12 cells hidden

Classification

Naive Bayes Classifier

```
Naive = naive_bayes.MultinomialNB()
Naive.fit(Train_X_Tfidf,Train_Y)
predictions_NB = Naive.predict(Test_X_Tfidf)
```

```
print(classification_report(predictions_NB, Test_Y, zero_division=0))
```

```
              precision    recall  f1-score   support

           0       0.00      0.38      0.00        52
           1       0.00      0.00      0.00         0
           2       1.00      0.53      0.69     34613
           3       0.00      0.50      0.00         4
           4       0.00      0.00      0.00         0
           5       0.00      0.00      0.00         0
           6       0.00      0.00      0.00         0
           7       0.00      0.00      0.00         0
           8       0.00      0.00      0.00         0

    accuracy                           0.53     34669
   macro avg       0.11      0.16      0.08     34669
weighted avg       1.00      0.53      0.69     34669
```

```
print(precision_score(predictions_NB, Test_Y, zero_division=0, average='weighted'))
print(recall_score(predictions_NB, Test_Y, zero_division=0, average='weighted'))
print(f1_score(predictions_NB, Test_Y, zero_division=0, average='weighted'))
```

```
0.9972933914033522
0.5259742132741065
0.6881800306620117
```

SVM Classifier

```
SVM = svm.SVC(C=1.0, kernel='linear', degree=3, gamma='auto')
SVM.fit(Train_X_Tfidf,Train_Y)
predictions_SVM = SVM.predict(Test_X_Tfidf)
```

```
print(classification_report(predictions_SVM, Test_Y, zero_division=0))
```

```
                precision    recall  f1-score   support

             0       0.00      0.33      0.01        73
             1       0.00      0.67      0.00         3
             2       1.00      0.53      0.69     34587
             3       0.00      0.17      0.00         6
             4       0.00      0.00      0.00         0
             5       0.00      0.00      0.00         0
             6       0.00      0.00      0.00         0
             7       0.00      0.00      0.00         0
             8       0.00      0.00      0.00         0

      accuracy                           0.53     34669
     macro avg       0.11      0.19      0.08     34669
  weighted avg       1.00      0.53      0.69     34669
```

```
print(precision_score(predictions_SVM, Test_Y, zero_division=0, average='weighted'))
print(recall_score(predictions_SVM, Test_Y, zero_division=0, average='weighted'))
print(f1_score(predictions_SVM, Test_Y, zero_division=0, average='weighted'))
```

```
0.9955619514045915
0.5255992385127926
0.6873267249805707
```

```
SVM = svm.SVC(C=1.0, kernel='rbf', degree=3, gamma='auto')
SVM.fit(Train_X_Tfidf,Train_Y)
predictions_SVM = SVM.predict(Test_X_Tfidf)
```

```
print(classification_report(predictions_SVM, Test_Y, zero_division=0))
```

```
                precision    recall  f1-score   support

             0       0.00      0.00      0.00         0
             1       0.00      0.00      0.00         0
             2       1.00      0.53      0.69     34669
             3       0.00      0.00      0.00         0
             4       0.00      0.00      0.00         0
             5       0.00      0.00      0.00         0
             6       0.00      0.00      0.00         0
             7       0.00      0.00      0.00         0
             8       0.00      0.00      0.00         0

      accuracy                           0.53     34669
     macro avg       0.11      0.06      0.08     34669
  weighted avg       1.00      0.53      0.69     34669
```

```
print(precision_score(predictions_SVM, Test_Y, zero_division=0, average='weighted'))
print(recall_score(predictions_SVM, Test_Y, zero_division=0, average='weighted'))
print(f1_score(predictions_SVM, Test_Y, zero_division=0, average='weighted'))
```

```
1.0
0.525916524849289
0.6893123133340894
```

Random Forest

```
RF = RandomForestClassifier()
RF.fit(Train_X_Tfidf,Train_Y)
predictions_RF = RF.predict(Test_X_Tfidf)
```

```
print(classification_report(predictions_RF, Test_Y, zero_division=0))
```

	precision	recall	f1-score	support
0	0.02	0.29	0.04	662
1	0.00	0.11	0.01	88
2	0.98	0.53	0.68	33751
3	0.01	0.27	0.02	77
4	0.00	0.00	0.00	5
5	0.00	0.00	0.00	2
6	0.00	0.21	0.01	58
7	0.00	0.00	0.00	5
8	0.00	0.10	0.00	21
accuracy			0.52	34669
macro avg	0.11	0.17	0.09	34669
weighted avg	0.95	0.52	0.67	34669

```
print(precision_score(predictions_RF, Test_Y, zero_division=0, average='weighted'))
print(recall_score(predictions_RF, Test_Y, zero_division=0, average='weighted'))
print(f1_score(predictions_RF, Test_Y, zero_division=0, average='weighted'))
```

```
0.9501501172265656
0.5198592402434451
0.6670890797870292
```

XGBoost Classifier

```
XGB = XGBClassifier()
XGB.fit(Train_X_Tfidf,Train_Y)
predictions_XGB = XGB.predict(Test_X_Tfidf)
```

```
print(classification_report(predictions_XGB, Test_Y, zero_division=0))
```

	precision	recall	f1-score	support
0	0.00	0.50	0.00	8
1	0.00	1.00	0.00	3
2	1.00	0.52	0.69	34641
3	0.00	0.24	0.00	17
4	0.00	0.00	0.00	0
5	0.00	0.00	0.00	0
6	0.00	0.00	0.00	0
7	0.00	0.00	0.00	0
8	0.00	0.00	0.00	0
accuracy			0.52	34669
macro avg	0.11	0.25	0.08	34669
weighted avg	1.00	0.52	0.69	34669

```
print(precision_score(predictions_XGB, Test_Y, zero_division=0, average='weighted'))
print(recall_score(predictions_XGB, Test_Y, zero_division=0, average='weighted'))
print(f1_score(predictions_XGB, Test_Y, zero_division=0, average='weighted'))
```

```
0.9985325953222426
0.523147480458046
0.6863119231052144
```

Classification N-Gram (2,3)

```
Tfidf_vect = TfidfVectorizer(ngram_range=(2,3))
Tfidf_vect.fit(gtd_view['summary_new'].astype(str))
Train_X_Tfidf = Tfidf_vect.transform(Train_X.astype(str))
Test_X_Tfidf = Tfidf_vect.transform(Test_X.astype(str))
```

Naive Bayes Classifier

```
Naive = naive_bayes.MultinomialNB()
Naive.fit(Train_X_Tfidf,Train_Y)
predictions_NB = Naive.predict(Test_X_Tfidf)
```

```
print(classification_report(predictions_NB, Test_Y, zero_division=0))
```

```
print(classification_report(predictions_NB, Test_Y, zero_division=0))
```

```
              precision    recall  f1-score   support

           0       0.00      0.27      0.00        15
           1       0.00      1.00      0.00         2
           2       1.00      0.53      0.69     34650
           3       0.00      1.00      0.00         2
           4       0.00      0.00      0.00         0
           5       0.00      0.00      0.00         0
           6       0.00      0.00      0.00         0
           7       0.00      0.00      0.00         0
           8       0.00      0.00      0.00         0

    accuracy                           0.53     34669
   macro avg       0.11      0.31      0.08     34669
weighted avg       1.00      0.53      0.69     34669
```

```
print(precision_score(predictions_NB, Test_Y, zero_division=0, average='weighted'))
print(recall_score(predictions_NB, Test_Y, zero_division=0, average='weighted'))
print(f1_score(predictions_NB, Test_Y, zero_division=0, average='weighted'))
```

```
0.9991233931675582
0.5259742132741065
0.6889559259017372
```

SVM Classifier

```
SVM = svm.SVC(C=1.0, kernel='linear', degree=3, gamma='auto')
SVM.fit(Train_X_Tfidf,Train_Y)
predictions_SVM = SVM.predict(Test_X_Tfidf)
```

```
print(classification_report(predictions_SVM, Test_Y, zero_division=0))
```

```
              precision    recall  f1-score   support

           0       0.01      0.33      0.02       265
           1       0.00      0.60      0.00         5
           2       0.99      0.53      0.69     34377
           3       0.00      0.44      0.01        16
           4       0.00      0.00      0.00         0
           5       0.00      0.00      0.00         0
           6       0.00      0.60      0.00         5
           7       0.00      0.00      0.00         0
           8       0.00      0.00      0.00         1

    accuracy                           0.52     34669
   macro avg       0.11      0.28      0.08     34669
weighted avg       0.98      0.52      0.68     34669
```

```
print(precision_score(predictions_SVM, Test_Y, zero_division=0, average='weighted'))
print(recall_score(predictions_SVM, Test_Y, zero_division=0, average='weighted'))
print(f1_score(predictions_SVM, Test_Y, zero_division=0, average='weighted'))
```

```
0.9844291965141213
0.5249935100522081
0.6824509616062843
```

```
SVM = svm.SVC(C=1.0, kernel='rbf', degree=3, gamma='auto')
SVM.fit(Train_X_Tfidf,Train_Y)
predictions_SVM = SVM.predict(Test_X_Tfidf)
```

```
print(classification_report(predictions_SVM, Test_Y, zero_division=0))
```

```
                precision    recall  f1-score   support

             0       0.00      0.00      0.00         0
             1       0.00      0.00      0.00         0
             2       1.00      0.53      0.69     34669
             3       0.00      0.00      0.00         0
             4       0.00      0.00      0.00         0
             5       0.00      0.00      0.00         0
             6       0.00      0.00      0.00         0
             7       0.00      0.00      0.00         0
             8       0.00      0.00      0.00         0

      accuracy                           0.53     34669
     macro avg       0.11      0.06      0.08     34669
  weighted avg       1.00      0.53      0.69     34669
```

```
print(precision_score(predictions_SVM, Test_Y, zero_division=0, average='weighted'))
print(recall_score(predictions_SVM, Test_Y, zero_division=0, average='weighted'))
print(f1_score(predictions_SVM, Test_Y, zero_division=0, average='weighted'))
```

```
1.0
0.525916524849289
0.6893123133340894
```

Random Forest

```
RF = RandomForestClassifier()
RF.fit(Train_X_Tfidf,Train_Y)
predictions_RF = RF.predict(Test_X_Tfidf)
```

```
print(classification_report(predictions_RF, Test_Y, zero_division=0))
```

```
                precision    recall  f1-score   support

             0       0.02      0.28      0.04       674
             1       0.00      0.12      0.01        72
             2       0.97      0.53      0.68     33752
             3       0.01      0.24      0.02        79
             4       0.00      0.00      0.00         5
             5       0.00      0.00      0.00         1
             6       0.00      0.14      0.01        65
             7       0.00      0.00      0.00         4
             8       0.00      0.06      0.00        17

      accuracy                           0.52     34669
     macro avg       0.11      0.15      0.08     34669
  weighted avg       0.95      0.52      0.67     34669
```

```
print(precision_score(predictions_RF, Test_Y, zero_division=0, average='weighted'))
print(recall_score(predictions_RF, Test_Y, zero_division=0, average='weighted'))
print(f1_score(predictions_RF, Test_Y, zero_division=0, average='weighted'))
```

```
0.949471273648215
0.5191092907208169
0.6665836940202605
```

XGBoost Classifier

```
XGB = XGBClassifier()
XGB.fit(Train_X_Tfidf,Train_Y)
predictions_XGB = XGB.predict(Test_X_Tfidf)
```

```
print(classification_report(predictions_XGB, Test_Y, zero_division=0))
```

	precision	recall	f1-score	support
0	0.00	0.33	0.00	6
1	0.00	1.00	0.00	1
2	1.00	0.53	0.69	34653
3	0.00	0.56	0.01	9
4	0.00	0.00	0.00	0
5	0.00	0.00	0.00	0
6	0.00	0.00	0.00	0
7	0.00	0.00	0.00	0
8	0.00	0.00	0.00	0
accuracy			0.53	34669
macro avg	0.11	0.27	0.08	34669
weighted avg	1.00	0.53	0.69	34669

```
print(precision_score(predictions_XGB, Test_Y, zero_division=0, average='weighted'))
print(recall_score(predictions_XGB, Test_Y, zero_division=0, average='weighted'))
print(f1_score(predictions_XGB, Test_Y, zero_division=0, average='weighted'))
```

```
0.9992651304303738
0.5260030574865153
0.6890151106249981
```