AndrewEaton_GTDChallenge

November 28, 2017

1 This purpose of this notebook is to predict what terrorist group may have been resposnible for a terrorist attack using the Global Terrorism Database.

This will be a multinomial classification problem. We will predict the terrorist group using the following statistical methods:

- Logistic Regression
- K Nearest Neighbors
- Random Forests

We will start off using train, test, split to narrow down which model is the best and then switch to cross validation.

Let's first import the data.

```
In [1]: import pandas as pd
        import numpy as np

        df = pd.read_excel(r'/Users/eatonaw/Downloads/globalterrorismdb_0617dist.xlsx')
        df.head()
```

```
Out[1]:         eventid  iyear  imonth  iday approxdate  extended resolution  country  \
        0  197000000001   1970       7     2        NaN         0        NaT       58
        1  197000000002   1970       0     0        NaN         0        NaT      130
        2  197001000001   1970       1     0        NaN         0        NaT      160
        3  197001000002   1970       1     0        NaN         0        NaT       78
        4  197001000003   1970       1     0        NaN         0        NaT      101

                  country_txt  region    ...     addnotes scite1 scite2  scite3  \
        0  Dominican Republic       2    ...          NaN    NaN    NaN     NaN
        1              Mexico       1    ...          NaN    NaN    NaN     NaN
        2         Philippines       5    ...          NaN    NaN    NaN     NaN
        3              Greece       8    ...          NaN    NaN    NaN     NaN
        4               Japan       4    ...          NaN    NaN    NaN     NaN

           dbsource  INT_LOG  INT_IDEO INT_MISC INT_ANY  related
```

```
0      PGIS       0       0      0      0      NaN
1      PGIS       0       1      1      1      NaN
2      PGIS      -9      -9      1      1      NaN
3      PGIS      -9      -9      1      1      NaN
4      PGIS      -9      -9      1      1      NaN

[5 rows x 135 columns]
```

Let's examine the shape, column names, and column types from the data.

```
In [2]: print(df.shape)
        print(df.columns)
        print(df.dtypes)
```

```
(170350, 135)
Index(['eventid', 'iyear', 'imonth', 'iday', 'approxdate', 'extended',
       'resolution', 'country', 'country_txt', 'region',
       ...
       'addnotes', 'scite1', 'scite2', 'scite3', 'dbsource', 'INT_LOG',
       'INT_IDEO', 'INT_MISC', 'INT_ANY', 'related'],
      dtype='object', length=135)
eventid                               int64
iyear                                 int64
imonth                                int64
iday                                  int64
approxdate                           object
extended                              int64
resolution                   datetime64[ns]
country                               int64
country_txt                          object
region                                int64
region_txt                           object
provstate                            object
city                                 object
latitude                            float64
longitude                           float64
specificity                         float64
vicinity                              int64
location                             object
summary                              object
crit1                                 int64
crit2                                 int64
crit3                                 int64
doubtterr                             int64
alternative                         float64
alternative_txt                      object
multiple                              int64
success                               int64
```

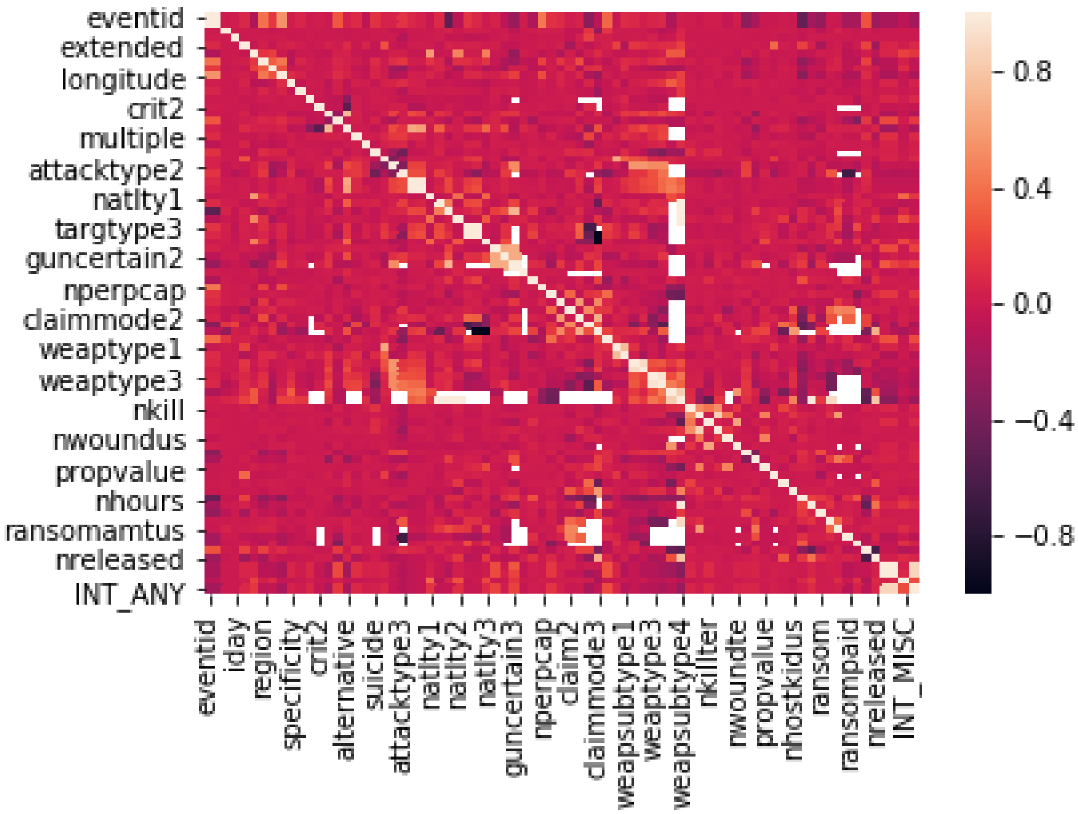
```
suicide                          int64
attacktype1                      int64
attacktype1_txt                 object
                                 ...
propextent                     float64
propextent_txt                  object
propvalue                      float64
propcomment                     object
ishostkid                      float64
nhostkid                       float64
nhostkidus                     float64
nhours                         float64
ndays                          float64
divert                          object
kidhijcountry                   object
ransom                         float64
ransomamt                      float64
ransomamtus                    float64
ransompaid                     float64
ransompaidus                   float64
ransomnote                      object
hostkidoutcome                 float64
hostkidoutcome_txt              object
nreleased                      float64
addnotes                        object
scite1                          object
scite2                          object
scite3                          object
dbsource                        object
INT_LOG                          int64
INT_IDEO                         int64
INT_MISC                         int64
INT_ANY                          int64
related                         object
Length: 135, dtype: object
```

Let's start to examine any correlations that may be present between different columns.

```
In [3]: import matplotlib.pyplot as plt
        import seaborn as sns
        %matplotlib inline

        corr = df.corr()
        sns.heatmap(corr)
```

```
Out[3]: <matplotlib.axes._subplots.AxesSubplot at 0x126c0e6d8>
```



Let's examine the data more closely by looking at the value counts for the target variable we are interested in (gname).

```
In [4]: print(df.gname.value_counts())
```

```
Unknown                                              78306
Taliban                                               6575
Shining Path (SL)                                     4551
Islamic State of Iraq and the Levant (ISIL)           4287
Farabundo Marti National Liberation Front (FMLN)      3351
Al-Shabaab                                            2683
Irish Republican Army (IRA)                           2669
Revolutionary Armed Forces of Colombia (FARC)         2481
New People's Army (NPA)                               2414
Kurdistan Workers' Party (PKK)                        2152
Boko Haram                                            2077
Basque Fatherland and Freedom (ETA)                   2024
Communist Party of India - Maoist (CPI-Maoist)        1766
Liberation Tigers of Tamil Eelam (LTTE)               1606
National Liberation Army of Colombia (ELN)            1483
Maoists                                               1424
```

```
Tehrik-i-Taliban Pakistan (TTP)                     1252
Palestinians                                        1124
Al-Qaida in the Arabian Peninsula (AQAP)             975
Nicaraguan Democratic Force (FDN)                    895
Houthi extremists (Ansar Allah)                      893
Manuel Rodriguez Patriotic Front (FPMR)              830
Sikh Extremists                                      714
Corsican National Liberation Front (FLNC)            639
Al-Qaida in Iraq                                     636
Donetsk People's Republic                            614
African National Congress (South Africa)             607
Separatists                                          571
Muslim extremists                                    561
Tupac Amaru Revolutionary Movement (MRTA)            557
                                                    ...
Bangsamoro National Liberation Army                    1
Mouhajiroune Brigade                                   1
Grupo Armado de Liberacion Argentina (GALA)            1
Opponents of Regime                                    1
God our Father Cult                                    1
Dashmesh Regiment                                      1
Pakistan Muslim League (PML)                           1
Separatist Clandestine Organization                    1
Politico-Military Revolutionary Command                1
Al-Sawaiq Brigade                                      1
Mozambique Rightest Rebels                             1
Extreme Right Commando Brigade                         1
Al-Sunni muslim sect                                   1
French Armed Islamic Front                             1
Martyrs of Saad Sayel                                  1
Organization Alliance of Cuban Intransigence           1
Robin Garcia Student Front                             1
The Nation's Army                                      1
Los Rastrojos (Colombia)                               1
The Great Serpent                                      1
Commander Gonzalo Southern Group                       1
Isatabu Freedom Movement (IFM)                         1
Front of French National Liberation                    1
Al-Fateh Al-Jadid                                      1
Latvian Republic Volunteer Troops                      1
Arakan Rohingy Islamic Front                           1
Anti-Imperialist Commando                              1
Pemuda Pancasila                                       1
Kabataang Makabayan (KM)                               1
Free Nasserite Revolutionaries                         1
Name: gname, Length: 3454, dtype: int64
```

As we can see, the gnames (our target) column contains a lot of unknowns and terrorist organizations that only committed one attack. It is not extremely useful to predict an unknown terrorist organziation and we need to have sufficient sample sizes (n>100 attacks). We will create a new data frame that contains a susbet of the data.

In [5]:
```
df2 = df[(df.gname != 'Unknown')]
df2 = df2.groupby("gname").filter(lambda x: len(x) >= 100)
print(df2.gname.value_counts())
df2.shape
```

```
Taliban                                             6575
Shining Path (SL)                                   4551
Islamic State of Iraq and the Levant (ISIL)         4287
Farabundo Marti National Liberation Front (FMLN)    3351
Al-Shabaab                                          2683
Irish Republican Army (IRA)                         2669
Revolutionary Armed Forces of Colombia (FARC)       2481
New People's Army (NPA)                             2414
Kurdistan Workers' Party (PKK)                      2152
Boko Haram                                          2077
Basque Fatherland and Freedom (ETA)                 2024
Communist Party of India - Maoist (CPI-Maoist)      1766
Liberation Tigers of Tamil Eelam (LTTE)             1606
National Liberation Army of Colombia (ELN)          1483
Maoists                                             1424
Tehrik-i-Taliban Pakistan (TTP)                     1252
Palestinians                                        1124
Al-Qaida in the Arabian Peninsula (AQAP)             975
Nicaraguan Democratic Force (FDN)                    895
Houthi extremists (Ansar Allah)                      893
Manuel Rodriguez Patriotic Front (FPMR)              830
Sikh Extremists                                      714
Corsican National Liberation Front (FLNC)            639
Al-Qaida in Iraq                                     636
Donetsk People's Republic                            614
African National Congress (South Africa)             607
Separatists                                          571
Muslim extremists                                    561
Tupac Amaru Revolutionary Movement (MRTA)            557
M-19 (Movement of April 19)                          555
                                                    ...
Khmer Rouge                                          160
Baloch Liberation Front (BLF)                        160
Garo National Liberation Army                        158
Hizbul Mujahideen (HM)                               157
Guerrilla Army of the Poor (EGP)                     157
Lashkar-e-Taiba (LeT)                                156
Islamic Salvation Front (FIS)                        153
Popular Front for the Liberation of Palestine (PFLP)     151
```

```
Barqa Province of the Islamic State                         147
Runda Kumpulan Kecil (RKK)                                  146
Islamic State of Iraq (ISI)                                 145
Democratic Revolutionary Alliance (ARDE)                    141
Tribesmen                                                   131
Guatemalan National Revolutionary Unity (URNG)              131
Corsican National Liberation Front- Historic Channel        128
Left-Wing Guerrillas                                        127
Muslim Brotherhood                                          125
Irish National Liberation Army (INLA)                       124
Lashkar-e-Jhangvi                                           122
Fuerzas Armadas de Liberacion Nacional (FALN)               120
Lashkar-e-Islam (Pakistan)                                  118
Revolutionary Organization of People in Arms (ORPA)         117
Animal Liberation Front (ALF)                               116
Montoneros (Argentina)                                      116
Free Aceh Movement (GAM)                                    116
November 17 Revolutionary Organization (N17RO)              112
Mujahedin-e Khalq (MEK)                                     112
Free Syrian Army                                            112
The Extraditables                                           109
United Popular Action Movement                              109
Name: gname, Length: 117, dtype: int64
```

```
Out[5]: (72171, 135)
```

We have now organized our data frame to not include the unknown terrorist groups and only include terrorist groups that have orchestrated over 100 attacks. Let's see what our null accuracy is:

There are 72,171 cases and the Taliban (group with most attacks) committed 6,575 of these attacks. Therefore, if we divide 72,171 by 6,575, we will get our null accuracy: 10.98%

Let's examine some of the features more carefully. Thankfully, a lot of them are already mapped as integers which is necessary for using scikit learn. The features I have decicded I am interested in are: iyear, imonth, country, region, suicide, attacktype1, targtype1, weaptype1, and random. Let's see if any of these have null values.

```
In [6]: print(df2.iyear.isnull().sum())
        print(df2.imonth.isnull().sum())
        print(df2.country.isnull().sum())
        print(df2.region.isnull().sum())
        print(df2.suicide.isnull().sum())
        print(df2.attacktype1.isnull().sum())
        print(df2.targtype1.isnull().sum())
        print(df2.weaptype1.isnull().sum())
        print(df2.ransom.isnull().sum())
```

```
0
0
0
0
0
0
0
0
33774
```

Ransom is the only feature that has any null values. Let's fill the null values with 0s.

```
In [7]: df2.ransom.fillna(0, inplace = True)
        df2.ransom.isnull().sum()
```

```
Out[7]: 0
```

Let's set up the data in a format recognizable by scikit learn so we can begin testing different models.

```
In [8]: from sklearn.model_selection import train_test_split
        X = df2[['iyear', 'imonth', 'country', 'region', 'suicide',\
            'attacktype1', 'targtype1', 'weaptype1', 'ransom']]
        y = df2.gname
        print(X.shape)
        print(y.shape)

        X_train, X_test, y_train, y_test = train_test_split(X, y, random_state = 42)
```

```
(72171, 9)
(72171,)
```

Let's start with logistic regression in its default state and use accuracy score to evaluate it.

```
In [9]: from sklearn.linear_model import LogisticRegression
        from sklearn.metrics import accuracy_score

        logreg = LogisticRegression()
        logreg.fit(X_train, y_train)
        y_pred = logreg.predict(X_test)
        accuracy = accuracy_score(y_pred, y_test)
        print(accuracy)
```

```
0.473535443108
```

Our accuracy is 47%, which is an improvement over the null accuracy.

Let's now try to use the k-nearest neighbors classifier using its default hyperparameters.

```
In [10]: from sklearn.neighbors import KNeighborsClassifier

         knn = KNeighborsClassifier()
         knn.fit(X_train, y_train)
         y_pred = knn.predict(X_test)
         print(accuracy_score(y_test, y_pred))
```

```
0.801141717009
```

Our accuracy score is now 80%, which is a strong improvement over using a logistic regression.

Let's next look at random forests, which is an ensemble of decision trees, using its default hyperparameters.

```
In [11]: from sklearn.ensemble import RandomForestClassifier

         clf = RandomForestClassifier()
         clf.fit(X_train, y_train)
         y_pred = clf.predict(X_test)
         print(accuracy_score(y_test, y_pred))
```

```
0.830903951671
```

Random forests give us an accuracy score of around 83%. This is the best model so far, when just looking at the default hyperparameters.

We can try to improve upon all of our models through a randomized cross-validation search of tuning the hyperparameters. This is computationally expensive, but I will provide the code for how it would be run.

```
In [ ]: from sklearn.model_selection import RandomizedSearchCV
```

```
In [ ]: param_dist = {'penalty' : ['l1', 'l2'],
                      'C' : np.arange(.00001, 1, .00001)}

        random_search = RandomizedSearchCV(logreg,\
            param_distributions = param_dist, n_iter = 30)

        random_search.fit(X, y)
        print(random_search.best_params_)
```

```
In [ ]: param_dist = {'n_neighbors' : range(0, 300, 5)}

        random_search = RandomizedSearchCV(knn,\
            param_distributions = param_dist, n_iter = 10)
```

```
        random_search.fit(X, y)
        print(random_search.best_params_)
```

```
In [ ]: param_dist = {'n_estimators' : range(1, 350, 10),
                      'max_features' : range(1, 10, )}

        random_search = RandomizedSearchCV(clf,\
            param_distributions = param_dist, n_iter = 10)

        random_search.fit(X, y)
        print(random_search.best_params_)
```

Another technique we could use to improve our models is ensembling. Bagging is a powerful tool that can improve classifiers. However, it will not have much of an effect on random forests, since random forests are produced from bagging decision trees.

Bagging is computationally expensive, but I will provide the code for how it could be carried out for logistic regression and k-nearest neighbors.

```
In [ ]: from sklearn.ensemble import BaggingClassifier

        logreg = LogisticRegression()
        bag = BaggingClassifier(base_estimator = logreg,\
            n_estimators = 5, max_samples = 0.5)
        bag.fit(X_train, y_train)
        y_pred = bag.predict(X_test)
        print(accuracy_score(y_pred, y_test))
```

```
In [ ]: knn = KNeighborsClassifier()
        bag = BaggingClassifier(base_estimator = knn,\
            n_estimators = 5, max_samples = 0.5)
        bag.fit(X_train, y_train)
        y_pred = bag.predict(X_test)
```