Machine Learning Engineer Nanodegree

Frederico Guerra

December 2018

Contents

1 Introduction

1.1 Domain Background

Terrorism is usually understood as the use or threat of violence to further a political cause[1] around the world. The attacks are increasing since the 70 decade and it is being called Modern Terrorism with the rise of nationalist movements around the globe after The Second World War.

Although this growth, the Institute for Economics and Peace's[2] appointed that in the last few years the number of deaths and injuries has decreased 22% to 25,673 deaths in 2016 compared to the peak of terror activity in 2014 when over 32,500 people were killed. Still it is a gain, Daniel Brown[3] showed that more countries experienced at least one terrorism-related death in 2016 than in any other year 2001 with 77 countries affected.

1.2 Problem Statement

Although terrorism attacks are becoming less deadly in the past years, as can bee seen in Figure 1, the number of deaths is still high compared with last 40 years. Therefore, is important to know what are the main reasons associated with fatalities and death occurrence in these attacks. The purpose of this project is to build a binary classification model to predict whether occurred death or not given an terrorist attack. By building this classification model the expected outcomes of this project are:

- Predict what type attacks will originate death;
- Which features have most impact in fatalities existence;

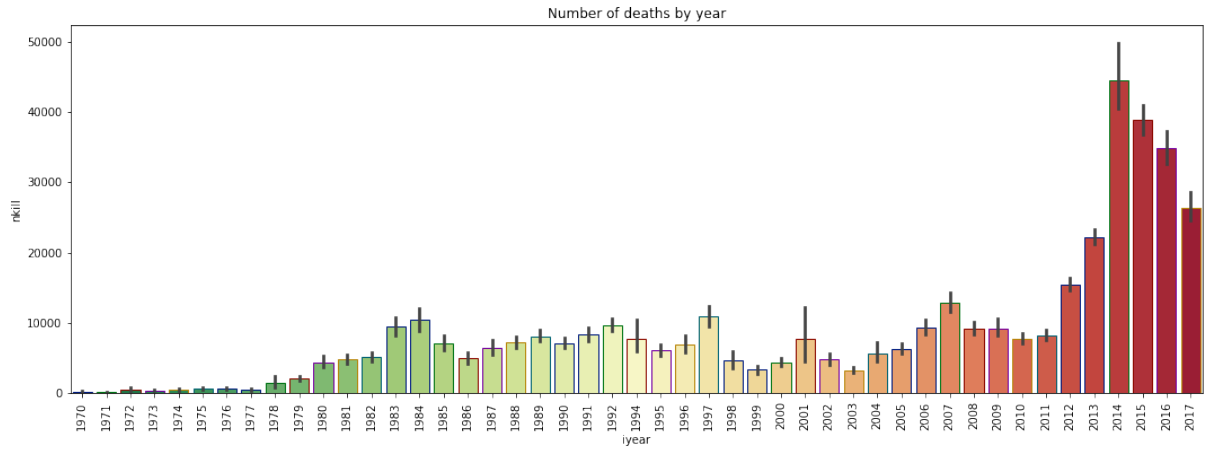

Figure 1: Number of deaths by year. (1970-2017)

[1] https://ourworldindata.org/terrorismdata-quality-definitions
[2] http://visionofhumanity.org/indexes/global-peace-index/
[3] https://www.businessinsider.com/global-terrorism-terrorism-increased-deaths-from-terrorism-decreased-2017-11

1.3 Evaluation Metrics

Considering that this is a binary classification problem, two metrics will be used to evaluate the model chosen: Confusion Matrix and Area Under the Curve (ROC curve). The main dataset will be splitted in other two for training and testing procedures.

1.3.1 Confusion Matrix

n = Test data size	**Predicted: NO**	**Predicted: YES**
Actual: NO		
Actual: YES		

To evaluate the model will be calculated its accuracy taking the number of attacks with death predictions (yes) by Total number of Samples:

$$accuracy = \frac{TruePositives + FalseNegatives}{TotalNumberofSamples}$$

where:

- **True Positives:** The cases in which will be predicted death (YES) and the actual output was also death (YES).
- **True Negatives:** The cases in which will be predicted no death (NO) and the actual output was no death (NO).
- **False Positives:** The cases in which will be predicted death (YES) and the actual output was no death (NO).
- **False Negatives:** The cases in which will be predicted no death (NO) and the actual output was death (YES).

1.3.2 Area Under the Curve

AUC of a classifier is equal to the probability that the classifier will rank a randomly chosen positive example higher than a randomly chosen negative example. To define AUC, should be understood about Sensitivity (True Positive Rate) and Specificity(False Positive Rate).

- **Sensitivity(True Positive Rate):** Number of deaths correctly predicted by the total number of deaths, in another words,

$$Sensitivity = \frac{TruePositives}{FalseNegatives + TruePositives}$$

- **Specificity(False Positive Rate):** Number of deaths mistakenly predicted by the total number of attacks without fatalities.

$$Specificity = \frac{FalsePositive}{FalsePositive + TrueNegative}$$

The values of TPR and FPR are computed continuously between [0,1] and drawn in a line chart. The area under this curve corresponds to the AUC score. Most closely this value is to 1, better is the model.

2 Analysis

2.1 Data Exploration

The Global Terrorism Database(GTD) is an open-source database including information and systematic data on domestic, transnational and international terrorist events around the world from 1970 to 2017. The dataset uded in this project will be caught from Kaggle API[4] At the moment, the dataset includes more than 180,000 cases and has a lot of information such as date, location, the weapons used, nature of the target, the number of casualties and - when identifiable - the group or individual responsible.
In total, there are 135 features and, as described in the GTD codebook[5], contains:

- Date
- Incident Information
- Incident Location
- Attack Information
- Weapon Information
- Target/Victim Information
- Perpetrator Information
- Casualties and Consequences
- Additional Information and Sources

The target variable death will be a boolean calculated column from the Total Number of Fatalities *nkill* feature in Casualties and Consequences group, where:

$$death = \begin{cases} 1, & if \quad nkill > 0 \\ 0, & otherwise \end{cases}$$

After a briefly exploratory analysis of nkill variable, can be inferred that there are 94% filled values - which corresponds to 171k valid data and in **48**% (83k) of them occurred at least one death. The other 6% data with missing values will be avoided.

[4]https://www.kaggle.com/START-UMD/gtd
[5]https://www.start.umd.edu/gtd/downloads/Codebook.pdf

2.2 Exploratory Visualization

Figure 2 shows the 105 features which contain missing values. Since there are many columns with high NAN percentage, in this analysis features with more than 50% will be dropped.

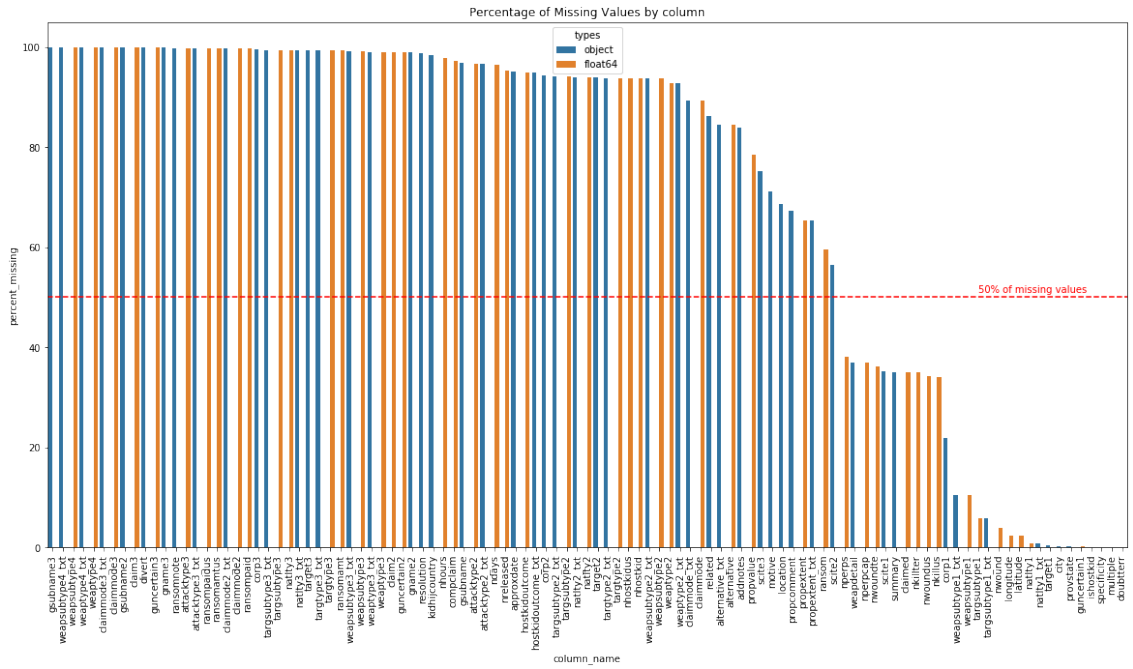

Figure 2: Percentage of missing values by column.

The top 3 regions more likely to happen deaths given an attack are The Sub-Saharian Africa, Middle East & North Africa and South Asia. The last two regions have the highest number of fatalities events while the remaining Asia and Africa have less occurrences.

Figure 4 shows that whether an act whose primary objective is to kill someone (Assassination) or is to cause physical harm or death directly to human beings by use of a firearm, incendiary, or sharp instrument (Armed Assault) this act is more likely to has fatalities.

The correlation matrix gives features relationship, Figure 5 shows that *attacktype1, weaptype1, weapsubtype1, success, nkillter and nwound* are most correlated columns with *death*.

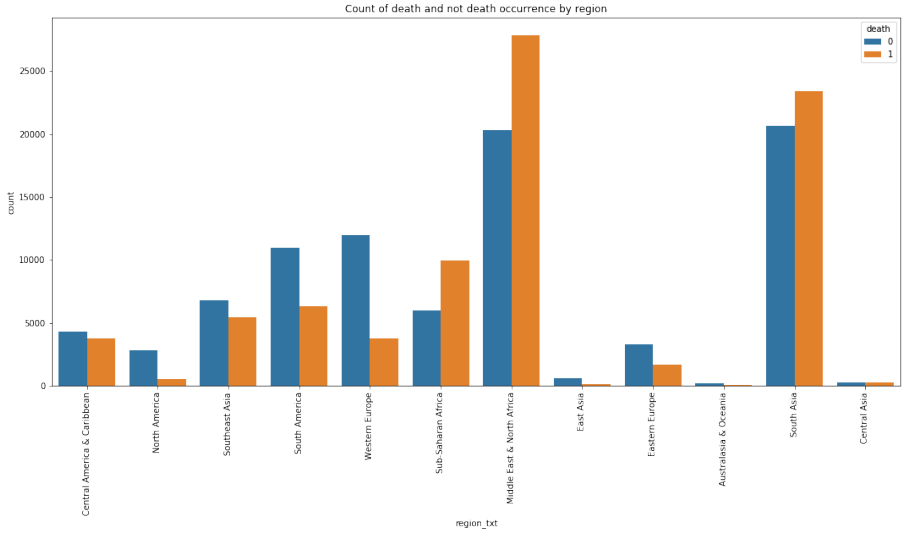

Figure 3: Count of death occurrence by region.

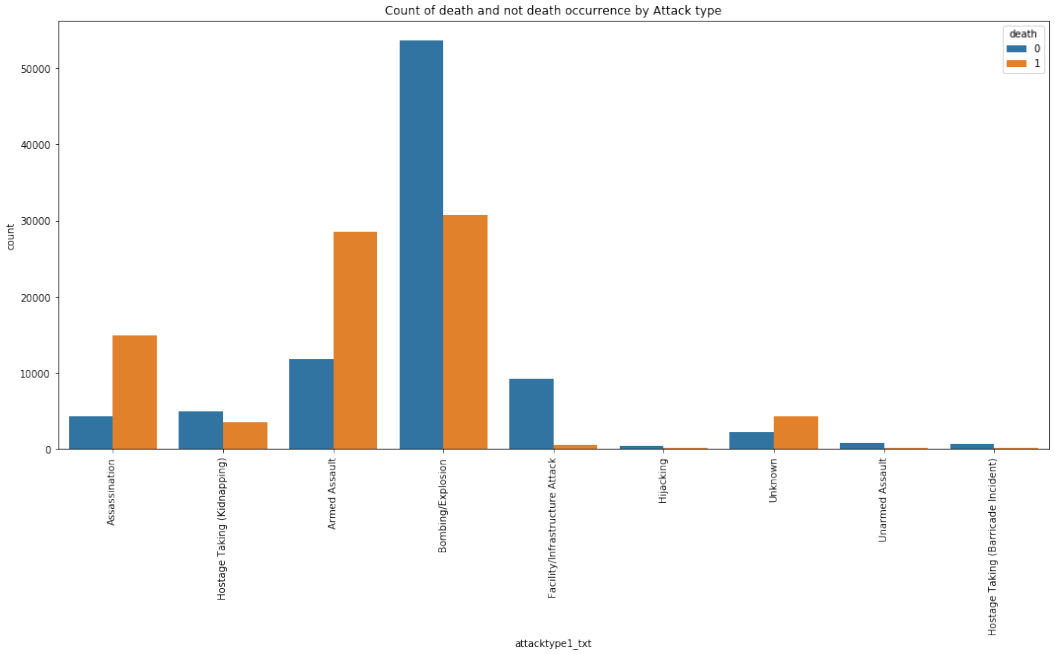

Figure 4: Count of death occurrence by attack type.

2.3 Algorithms and Techniques

Since it is a classification problem, the *sklearn-learn* library in python will be used to split the dataset, train and test three predefined models to find the best one.

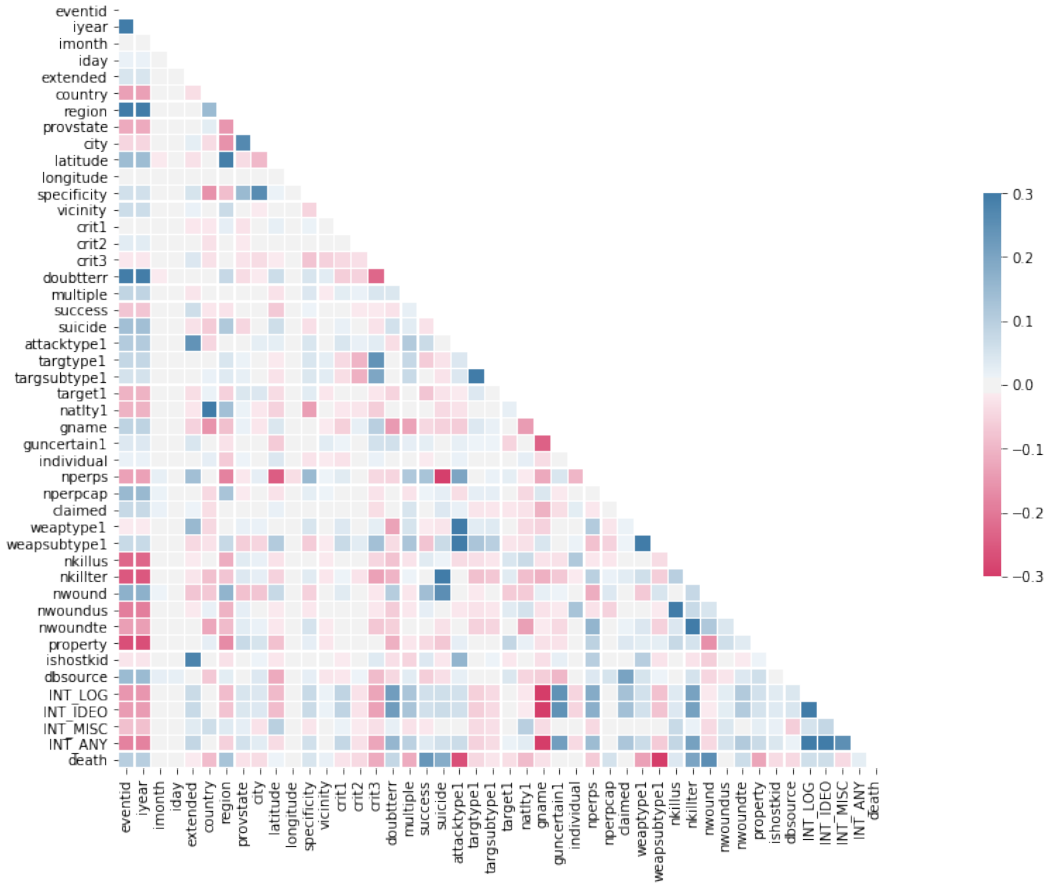

Figure 5: Correlation Matrix.

1. **AdaBoost Classifier.** AdaBoost is a mix of classifier models that pick them with different weights according to its impacts to the classification. It is fast, simple and versatile to implement, but its sensitive to noisy data and outliers.

2. **K Nearest Neighbors.** KNN is one of the simplest classification model in Machine Learning and widely used in problems in the industry since it is non-parametric, meaning, features do not need to have a specified distribution. In classification, an instance is defined by a majority vote of its K neighbours. Each defined sample has its coordinates (x,y) and a specified classification, the predictive sample is classified by the majority class of its K nearest neighbours based on its coordinates (x_0, y_0).

3. **Random Forest.** RFC is a collection of many decision trees working in parallel and independent learning decision rules. RFC runs efficiently on large data bases and handle a large ammount of input variables avoiding variable deletion. It gives useful internal estimates, such as error, correlation and variable importance. RFC is computationally lighter than other classification models like Adaboost and KNeighbors. Random Forest uses only a portion of the input features for each split which makes its train fast, beside it is relatively robust to outliers

and noise.

2.4 Benchmark Model

A benchmark model for this problem is a not tuned Decision Tree model classifier predefined by sklearn library. The table 1 shows its accuraccy and F_score.

	Decision Tree
accuracy	82.01%
fscore	81.36%

Table 1: Benchmark Decision Tree Evaluation

3 Methodology

3.1 Data Prepossessing

- **Treating Missing Values.** Given the high percentage of missing values (see Figure 2 on page 5) and their harmful effects to model performance, columns which contains percentage above 50% were dropped. The remaining features were splitted in two categories by type to fill their NAN values: object and float64. (Table 2)

type	**fillna**
object	*mode*
float64	*mean*

Table 2: NAN filling by column type on GTD database

- **Duplicated Features.** The GFD database has some features with text values and label encoding values in another one, i.e., the column *country_txt* identifies the country where the incident occurred, while *country* feature is the code of *country_txt*. As the model just understand codes as label, all columns with name *_txt* were avoided.

- **Categorical Features.** As machine learning models does not learn text columns, is required to encode these features. Due the high number of categorical features and their number of unique values, in this project is used LabelEncoder function from sklearn which transform one text column to numerical.

- **Numerical Features.** Different features in the data set have values in different ranges. It means that columns are more weighted than others and this divergence can cause an untruth learning to the model. To avoid this, columns with skewness coefficient above 0.75 is normalized through natural logarithm of one plus the input values:

$$x' = \log(1 + x) \tag{1}$$

3.2 Implementation

The detailed implementation is discussed as part of the Project Design showed on Figure 6. This step comprehend data cleaning and transformation, model selection, evaluation and tuning processes.

1. **Data cleaning and Transformation:**
   As mentioned above, columns with more than half of values missing is avoided and the remaining features are treated filling NAN values with mode and mean in categorical and numerical columns, respectively. Text columns is encoded using the *LabelEncoder()* function from *sklearn* library due the learn requirement of models in machine learning.

2. **Model Selection and Evaluation:**
   The GTD data set is splitted in train and test data sets following the proportion of 80% for 20%. After that, a decision tree benchmark model is building to get its *accuraccy* and *fbetascore* (beta = 0.5) from *sklearn.metrics*.

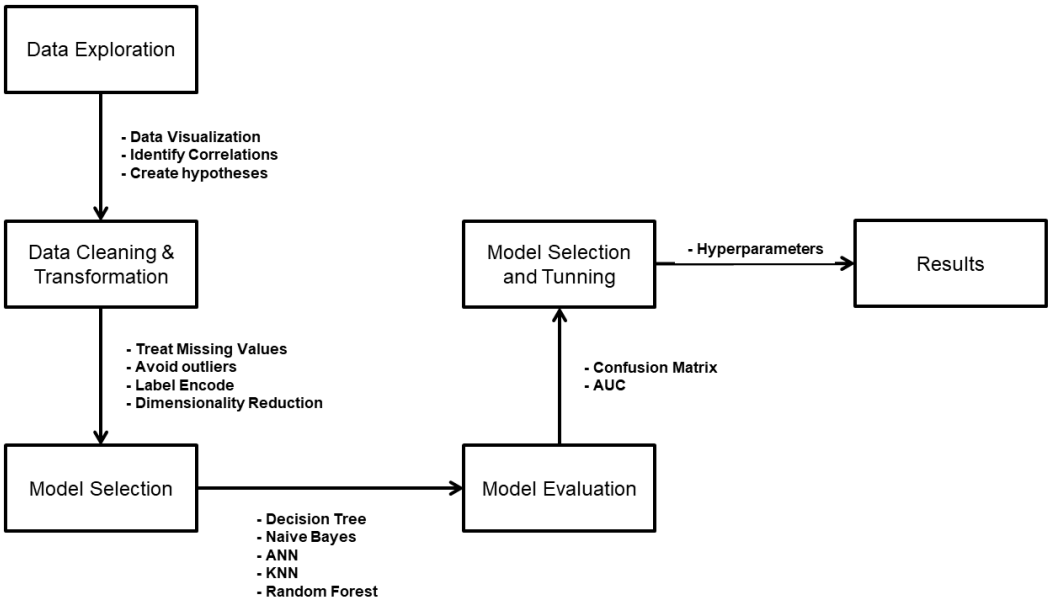

Figure 6: Project Pipeline

3.3 Refinement

Given the high number of columns, is used a Feature Selection from model with threshold of 0.2, it means that just features with importance higher than 0.2 on model prediction with continue in the analysis, it resulted in a reduction from 45 columns to 21.The RFC returned 85.79% and 84.83% of accuraccy and fscore after this dimensionality reduction.
The RandomizedSearchCV is used to test and find the best parameters for the model. See the parameters:

- bootstrap: [True, False];
- min_samples_leaf: [1, 2, 4];
- n_estimators: [2, 6, 11, 15, 20];
- min_samples_split: [2, 5, 10];
- max_features: ['auto', 'sqrt'];
- max_depth: [1, 3, 6, 8, 11, None];

4 Results

4.1 Model Evaluation and Validation

The decision tree benchmark model presented 82.01% and 81.36% of accuracy and fscore, the other three classifiers (K-Nearest Neighbours, Random Forest and Adaboost) is compared using their accuraccy, fbeta_score and training/testing times parameters. Random Forest Classifier(RFC) presented the lowest testing time, highest accuraccy (86.06%) and fscore (86.09%) due the either two and benchmark models.

Given the tested parameters above, the optimized Random Forest Classifier has:

- bootstrap: False;
- min_samples_leaf: 4;
- n_estimators: 20;
- min_samples_split: 5;
- max_features: 'auto';
- max_depth: None;

The final model pointed

	Benchmark Model	**KNN**	**Adaboost**	**Random Forest**	**Tuned Random Forest**
accuracy	82.01%	58.28%	82.59%	85.60%	**87.03%**
fscore	81.36%	58.28%	82.59%	85.62%	**86.50%**

Table 3: Model Evaluation

With the Random Forest Classifier trained on 100% of training data, a confusion matrix of the prediction test set can be seen below.

n = Test data size	**Predicted: NO**	**Predicted: YES**
Actual: NO	15585	2158
Actual: YES	2288	14245

Table 4: Final Confusion Matrix

These results indicate 86.84% of precision and 86.16% of recall, it means the final model correctly predicts death occurrence in 86.16% of the attacks with fatalities.

Figure 7 shows the ROC curve of the final random forest model. The ROC graph shows the performance of the classification model at all classification threshold, there are two parameters in this curve:

- True Positive Rate (Recall);
- False Positive Rate;

For each decision threshold are calculated both True positive and False positive rates and plotted as a point on the line green below. The AUC stands for Area Under the Curve and it is a measure of the entire two-dimensional area underneath the entire ROC curve ranging from [0.0] to [1,1].AUC provides an aggregate measure of performance across all possible classification thresholds. One way of interpreting AUC is as the probability that the model ranks a random positive example more highly than a random negative example.

AUC score ranges in value from 0 to 1, a model whose predictions are totally wrong has an AUC of 0, another one whose predictions are 100% correct has an AUC of 1. For this problem the AUC score is 0.95, which indicates that the final model has a good performance.

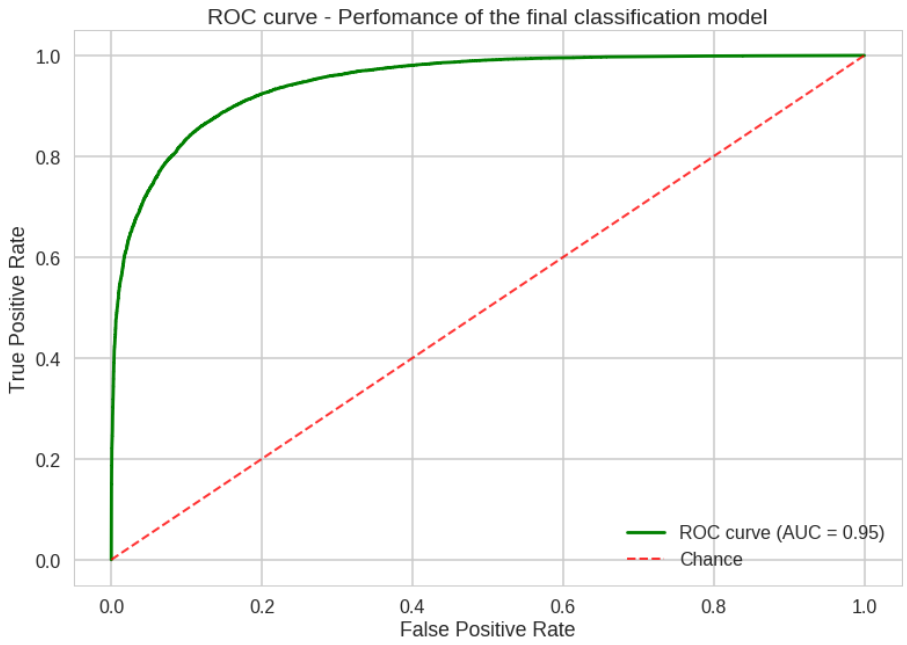

Figure 7: ROC curve and AUC score

5 Conclusion

As Figure 8 shows, the 5 most important features in the final RDC model is:

- **weaptype1**: This field records the general type of weapon used in the incident.
- **weapsubtype1**: This field records a more specific value for most of the Weapon Types identified.
- **nwound**: Records the number of confirmed non-fatal injuries to both perpetrators and victims.
- **attacktype1**: Captures the general method of attack and often reflects the broad class of tactics used.
- **success**: Success of a terrorist strike is defined according to the tangible effects of the attack. The definition of a successful attack depends on the type of attack. Essentially, the key question is whether or not the attack type took place.

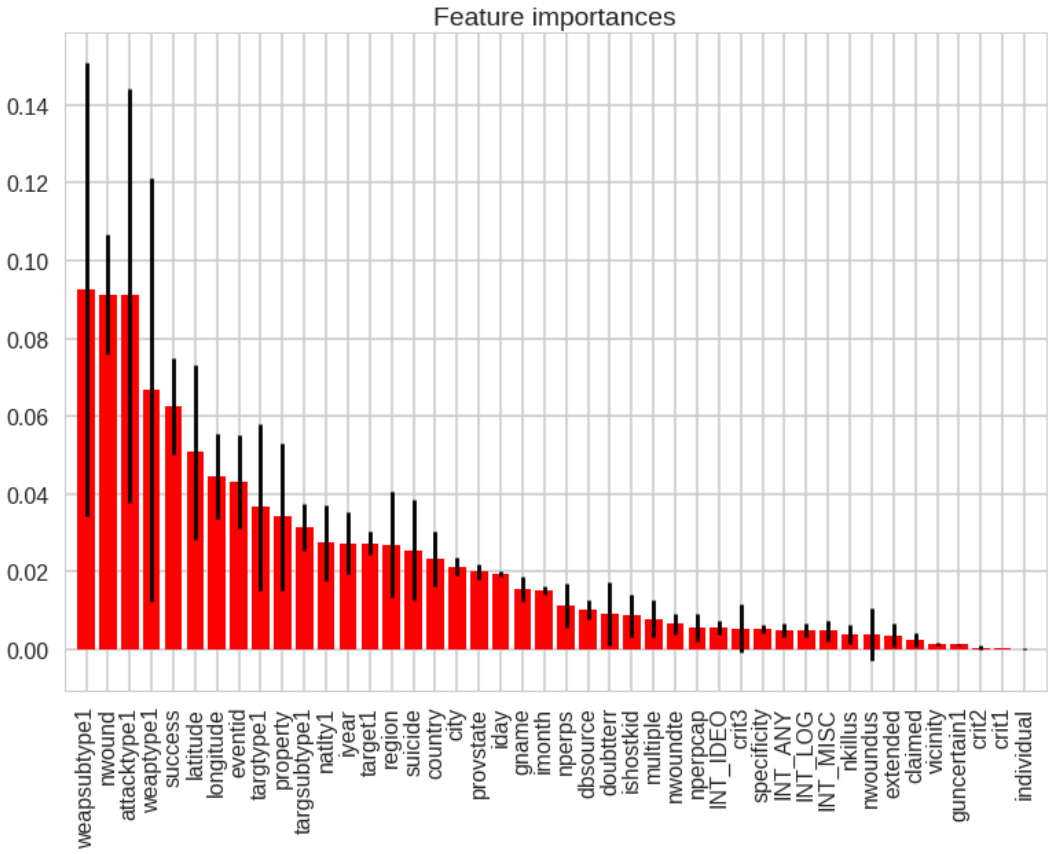

Figure 8: Feature Importances of the final Random Forest Classifier