GLOBAL TERRORISM ANALYSIS

JASH SHETH
SOHAN SHIRODKAR
DISHEEN SOLANKI

INTRODUCTION

- Terrorism is a major problem in today's world.
- Every country in the world is developing some sort of preventive mechanism to protect their citizens from terrorist attacks
- Some factors lead to higher impact of terrorist activities, while some factors aren't associated with significant damage to the society.
- For better protection against terrorism, and for defense agencies to analyse the risks associated with terrorist attacks, we aim to provide a study that analyses the various factors leading to serious threats.
- To fight terrorism, we must first fight the factors leading to it. Our study analyses the performance of various machine learning algorithms to check the impact of certain factors on terrorist impact.

OBJECTIVES

- To analyse the impact of global terrorist attacks in general
- To analyse the risks associated in terms of number of casualties and wounded
- To perform a comparative study of various Machine Learning algorithms for regression analysis

MACHINE LEARNING PROCESS

- Getting the dataset
- Understanding the data
- Data preprocessing
- Model fitting
- Evaluating accuracy of models
- Comparison of various models

1. GETTING THE DATASET

- The dataset used is the freely available Global Terrorism Database from Kaggle.
- The dataset contains terrorist activity information from 1970 till 2017.
- The dataset originally had 182,000 rows and 135 columns.
- After refining the dataset with relevant and significant features, we formed a dataset of 57 features.

UNDERSTANDING THE DATA

The following output snippet shows the kind of data that the dataset consists of:

	iyear	extended	region_txt	latitude	longitude	attacktype1_txt	targtype1_txt	weaptype1_txt	nkill	nwound
146045	2015	0	South Asia	30.475091	69.365678	Armed Assault	Police	Firearms	1.0	0.0
171397	2017	0	South Asia	18.965838	83.451666	Facility/Infrastructure Attack	Transportation	Sabotage Equipment	34.0	61.0
52333	1992	0	Middle East & North Africa	38.403111	37.956069	Armed Assault	Military	Firearms	3.0	0.0
171476	2017	0	Middle East & North Africa	36.354145	43.143570	Unknown	Military	Unknown	NaN	NaN
59505	1995	0	Sub-Saharan Africa	-7.365016	20.815942	Armed Assault	Government (Diplomatic)	Firearms	0.0	2.0
7647	1979	0	Central America & Caribbean	12.432570	-86.881204	Assassination	Private Citizens & Property	Unknown	1.0	0.0
104155	2011	0	Middle East & North Africa	37.080181	41.220196	Armed Assault	Military	Firearms	1.0	1.0
165836	2016	0	Middle East & North Africa	18.223461	42.526051	Bombing/Explosion	Private Citizens & Property	Explosives	0.0	0.0
121050	2013	0	South Asia	31.823601	64.564087	Bombing/Explosion	Military	Explosives	2.0	1.0
67269	1997	0	Sub-Saharan Africa	-1.996892	29.706386	Armed Assault	Police	Firearms	5.0	0.0

UNDERSTANDING THE DATASET

- The dataset consists of information about all kinds of factors possibly involved in a terrorist activity.
- The weapons columns consists of type of weapon that is used, like Firearms, Explosives, etc.
- The region column consists of the area in which the activity took place, as some areas are more terrorist activity prone than others.
- The attacktype column shows the kind of attack that took place, as mass attacks and bombings can have higher impact.
- The target column just indicates who the attack was intended for.

UNDERSTANDING THE DATASET

- We combined the "nkill" and "nwounded" features into one, "impact" feature that would measure the impact that a terrorist activity would have.
- The impact feature is used as a target variable.
- All of the other features are used as input features to train the models

DATA PREPROCESSING

- 1. Removing Missing values
- 2. Removing irrelevant features
- 3. Removing insignificant values like "Unknown" types of weapons.
- 4. Checking correlation among features and combining heavily correlated features together
- 5. One-hot encoding for categorical data.
- 6. Splitting data into training and testing sets

MODEL FITTING

In this step, we have used various machine learning algorithms to predict the impact, ie number of casualties + wounded people a terrorist attack may have. We have used 6 regression algorithms for our process:

- Linear Regression
- Decision Tree Regression
- Random Forest Regression
- XG Boost Regressor
- Artificial Neural Network
- Support Vector Regression

LINEAR REGRESSION

- Linear Regression is a machine learning algorithm based on supervised learning.
- Linear regression is the most basic regression model, hence we used it in order to compare with other models
- Linear regression generally works better for a dataset with a trend shown, especially if the target feature is a linear function of the input features.
- Mean Absolute Error : 6.64
- r2 score: 0.074

$$Y = b_0 + b_1x_1 + b_2x_2 + ... + b_nx_n$$

DECISION TREE REGRESSION

- Decision tree algorithm falls under the category of the supervised learning. They can be used to solve both regression and classification problems.
- Decision tree uses the tree representation to solve the problem in which each leaf node corresponds to a class label and attributes are represented on the internal node of the tree.
- We have used the DecisionTreeRegressor class from sklearn.tree in order to implement the decision tree algorithm.
- Mean Absolute Error: 6.94
- r2 score: -0.419

RANDOM FOREST

- Random Forest is a flexible, easy to use machine learning algorithm that produces, even without hyper- parameter tuning, a great result most of the time.
- The "forest" it builds, is an ensemble of Decision Trees, most of the time trained with the "bagging" method. The general idea of the bagging method is that a combination of learning models increases the overall result.
- Since Random Forest uses a collection of Decision Trees for training, it is likely to perform better than Decision Tree.
- Mean Absolute Error: 6.22
- r2 score: 0.025

XGBOOST

- XGBoost is an ensemble learning method. Sometimes, it may not be sufficient to rely upon the results of just one machine learning model. Ensemble learning offers a systematic solution to combine the predictive power of multiple learners. The resultant is a single model which gives the aggregated output from several models.
- The beauty of this powerful algorithm lies in its scalability, which drives fast learning through parallel and distributed computing and offers efficient memory usage.
- Mean Absolute Error: 6.34
- r2 score: 0.1075

ARTIFICIAL NEURAL NETWORK

- We used a neural network mainly because of the size of the dataset. As the final input data had tens of thousands of training examples, ANN seemed a good model to use.
- The neural network we used was built using Tensorflow's high level API, Keras.
- The ANN consisted of 4 hidden layers, each consisting of 16 neurons.

```
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 16)                912
_________________________________________________________________
dense_2 (Dense)              (None, 16)                272
_________________________________________________________________
dense_3 (Dense)              (None, 16)                272
_________________________________________________________________
dense_4 (Dense)              (None, 16)                272
_________________________________________________________________
dense_5 (Dense)              (None, 1)                 17
=================================================================
Total params: 1,745
Trainable params: 1,745
Non-trainable params: 0
_________________________________________________________________
```

ARTIFICIAL NEURAL NETWORK - TRAINING

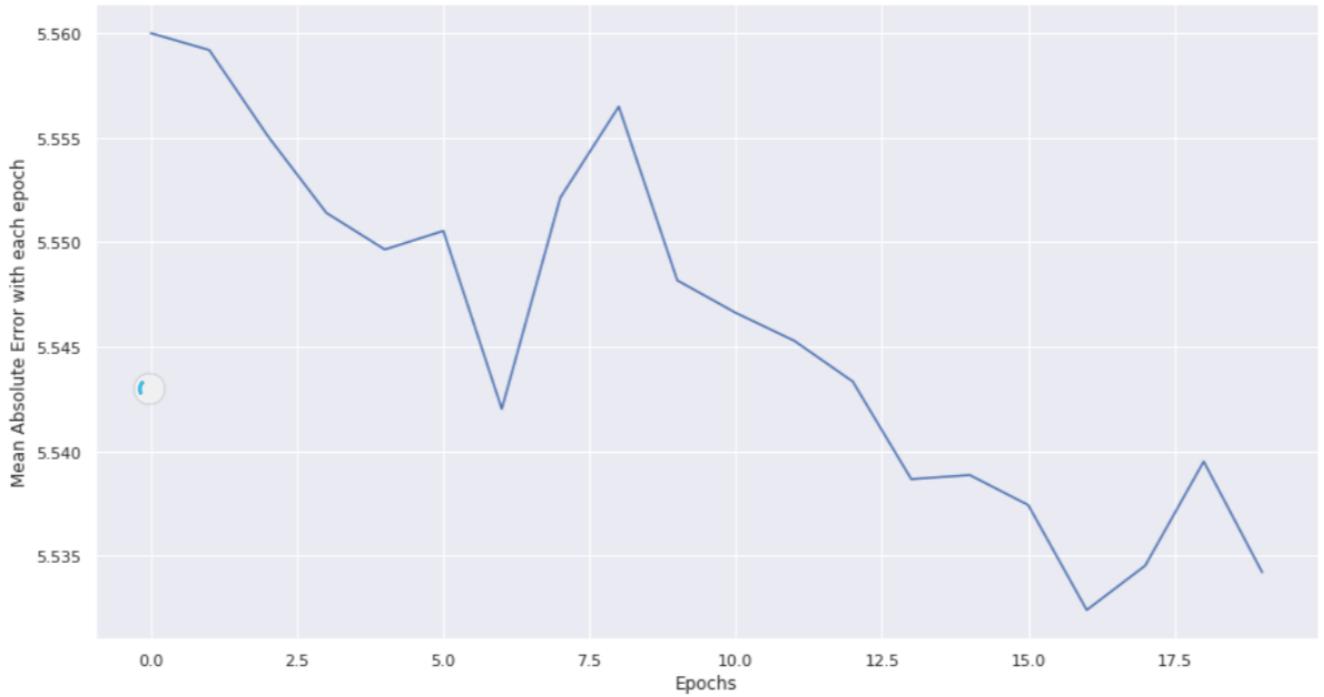
```
Epoch 1/10
72622/72622 [==============================] - 8s 115us/step - loss: 6.2897 - mean_absolute_error: 6.2897
Epoch 2/10
72622/72622 [==============================] - 7s 102us/step - loss: 7.4096 - mean_absolute_error: 7.4096
Epoch 3/10
72622/72622 [==============================] - 7s 102us/step - loss: 8.5928 - mean_absolute_error: 8.5928
Epoch 4/10
72622/72622 [==============================] - 7s 102us/step - loss: 5.7423 - mean_absolute_error: 5.7423
Epoch 5/10
72622/72622 [==============================] - 7s 102us/step - loss: 6.4869 - mean_absolute_error: 6.4869
Epoch 6/10
72622/72622 [==============================] - 7s 102us/step - loss: 5.6259 - mean_absolute_error: 5.6259
Epoch 7/10
72622/72622 [==============================] - 8s 110us/step - loss: 5.6142 - mean_absolute_error: 5.6142
Epoch 8/10
72622/72622 [==============================] - 8s 107us/step - loss: 5.9496 - mean_absolute_error: 5.9496
Epoch 9/10
72622/72622 [==============================] - 7s 101us/step - loss: 5.6064 - mean_absolute_error: 5.6064
Epoch 10/10
72622/72622 [==============================] - 7s 102us/step - loss: 5.6031 - mean_absolute_error: 5.6031
<keras.callbacks.History at 0x7f85458fc240>
```

ARTIFICIAL NEURAL NETWORK

- Mean Absolute Error: 5.36
- r2 score: -0.045

```
plt.plot(history.history['mean_absolute_error'])
plt.ylabel("Mean Absolute Error with each epoch")
plt.xlabel("Epochs")
```

```
Text(0.5, 0, 'Epochs')
```



SUPPORT VECTOR REGRESSOR

- The Support Vector Regression (SVR) uses the same principles as the SVM for classification, with only a few minor differences.
- First of all, because output is a real number it becomes very difficult to predict the information at hand, which has infinite possibilities. In the case of regression, a margin of tolerance (epsilon) is set in approximation to the SVM which would have already requested from the problem. But besides this fact, there is also a more complicated reason, the algorithm is more complicated therefore to be taken in consideration.
- This algorithm took the longest computational time, significantly greater than the others, but performed the best in terms of Mean Absolute Error.
- Mean Absolute Error: 5.16 (best)
- r2 score: 0.004

EVALUATING ACCURACY OF MODELS

Support Vector Regression performed the best among all the models

	Linear Regression	Support Vector Regression	Decision Tree	Random Forest	XG Boost	Neural Network
Mean Absolute Error	6.64	5.16	6.94	6.22	6.34	5.36
r2 score	0.074	0.004	-0.419	0.025	0.1075	-0.045
RMSE	12.44	12.90	15.41	12.77	12.22	13.22

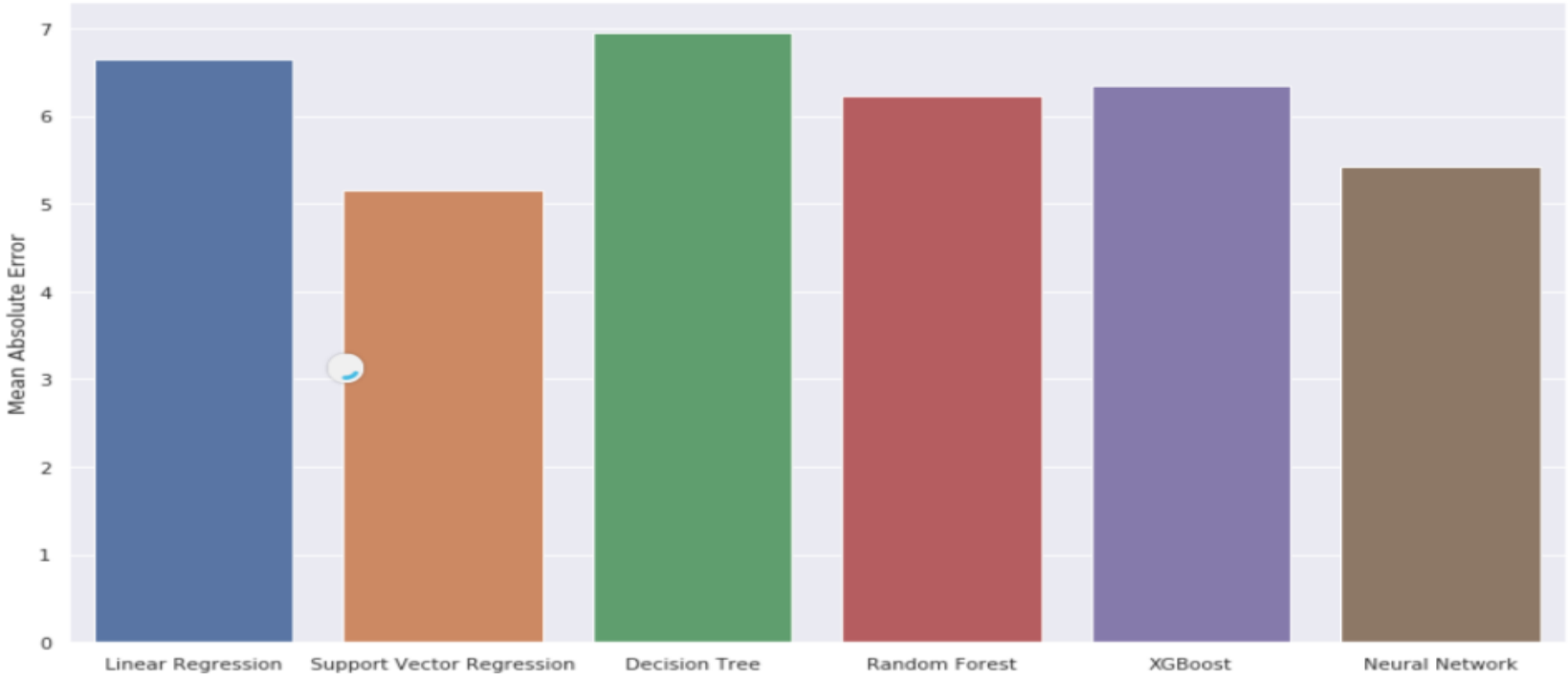
Table 4.4.1 Accuracy Metrics

COMPARISON OF VARIOUS MODELS

```
[100]
      sns.set(rc={'figure.figsize':(15,8)})
      plt.ylabel("Mean Absolute Error")

      sns.barplot(algorithms,scores)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f7c6cc5d278>
```



CONCLUSION

- Therefore, we analysed and studied the impact that various factors can have on the number of casualties due to a terrorist attack using popular regression algorithms.
- We also compared the difference in performance that each of the algorithms had on our dataset.
- We learnt that Support Vector Regression and the Artificial Neural Network perfomed best on this kind of dataset.