ORIE 4740 Final Project

Exploring the Deadliness of Potential Future Terrorist Attacks

Karim Arem (ka295), Maverick Lin (ml874)

May 2018

1 Abstract

Since the amount of terrorist attacks has dramatically increased over the past few decades, we were interested in figuring out which attributes of an attack would help predict the impact on civilians. Specifically, we wanted to predict the amount of total injuries/deaths for various attacks. Not surprisingly, explosives, bombs and dynamite were found to be among the most lethal types of weapons used. In addition, the type of institutions targeted (military and religious) gave an indication of how devastating an attack would be. What was surprising to us was that, East Asia was the region

where the most catastrophic events are bound to occur. Reflecting on our project, although we were only able to make use of 10% of our initial dataset due to the scarcity of terrorist data and incomplete information, we were able to create a respectable model to predict the total casualties. More in-depth research in this area could could help governments identify some of the most important sources of damage during attacks. Armed with such information, we hope that governments around the world take the necessary precautions to mitigate future damages and limit the lives lost.

2 Introduction

To keep our model simple, our main goal was to predict the number of people killed or severely injured. We used the Global Terrorism Database, managed by the National Consortium for START, which includes the most comprehensive data set of all terrorist attacks since 1970. For clarity, START defines terrorism as the "threatened or actual use of illegal force and violence by a non-state actor to attain a political, economic, religious, or social goal through fear, coercion, or intimidation." The data set includes 170,350 recorded incidents, each of them containing 135 variables describing the attack. These predictors include everything from the location of the event, to the amount of ransom that was requested, to the motive of the terrorist. After pre-processing, feature selection, and splitting data into training, validation and test sets, we fit numerous models to our data including Linear Regression, KN Regression, Regression Tree, Random Forest, and Gradient Boosting Regression Tree. We found that Gradient Boosting gave us the best model, with the lowest MSE and highest R^2 score.

3 Data Pre-Processing

Data pre-processing was probably the biggest challenge we faced. The dataset was extremely large and we had to determine which variables to include and which to toss. After exploring the data and applying some domain knowledge, we trimmed the 135 predictor variables into 8 predictors. Many of the predictors, such as ransom information, gun sub-types, etc. contained mostly NA values, which we elected to throw out. Our final predictor variables are included in the figure below:

Variable Name	Description	Type
Success	Failed or Successful Attack	Binary
Suicide	Suicide Attack	Binary
nperps	Number of Perpetrators	Numeric
region_txt	Location/Region of Attack	Categorical
attacktype1_txt	Type of Attack (hostage, hijacking, etc.)	Categorical
targtype1_txt	Type of Target (Business, Religious, etc.)	Categorical
propextent_txt	Extent of Property Damage	Categorical
weaptype1_txt	Type of Weapon Used	Categorical

Figure 1: Predictor Variables to be used in our analysis

After eliminating all events that contained NA values, we decided to filter our data to include the events where the number of perpetrators (nperps) is known. Many of the data points for nperps were -99, or ”Unknown”. We found that we obtained better results doing this rather than replacing them with the average. Our final pre-processed dataset featured 4,730 observations with 7 predictor variables.

Finally, we converted all of the categorical variables into dummy variables (in order to run the regression models) and we summed the nkill and nwound columns to give us what would be our y variable.

For our train-test split, we initially chose a 60/20/20 split, but we decided to opt for a 80/20 split since we were using cross-validation to estimate our test error and to tune our hyperparamters.

4 Analysis

Our game plan was to start with the simpler models and fit future models in order of higher complexity. For each model, if we were able to, we performed hyperparamter tuning using sklearn's GridCV to optimize certain parameters of the given model. We found that the tuned models performed much better and we provide a comparison of the default vs. tuned models below. In addition, we performed cross-validation to provide an estimate of our test error. For our error metric, we used Mean Squared Error (MSE) and Adjusted R^2. Finally, for each model, we decided to explore the features that the model deemed the most important.

4.1 Exploratory Data Analysis

To get a better idea of the distribution of our data, we plotted a histogram of our outcome variable in figure 2. It is clear that most of the data indicates that no one was hurt or killed. However, this was strange, since our data indicated that 96% of the attacks were successful. We decided to look more into the data and discovered that a good portion of our data (around 30%) was comprised of hostage attacks, hijackings, infrastructure takeovers etc... We decided to leave the data unchanged, as such information could be useful for our models to differentiate between lethal and non-lethal attacks.

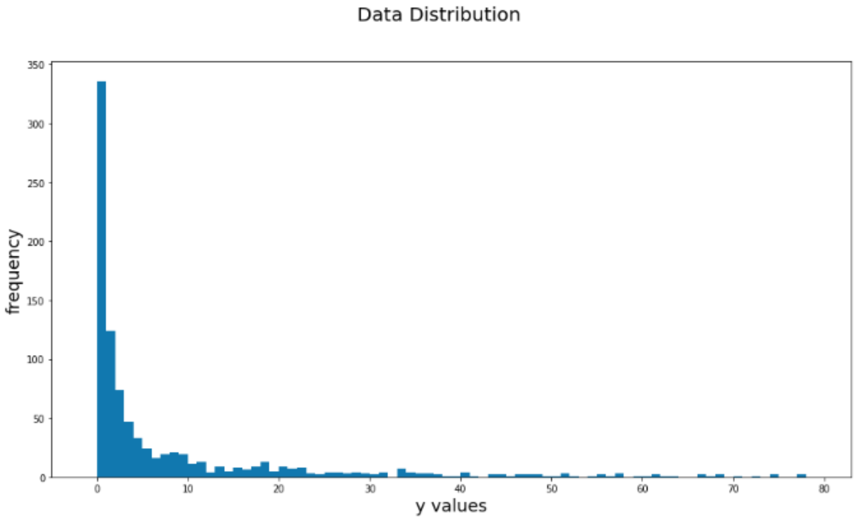
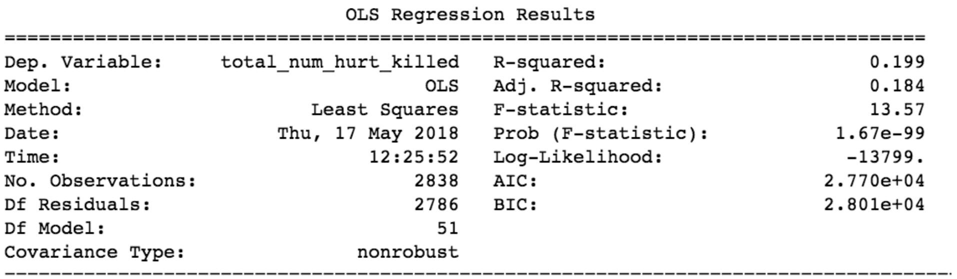


Figure 2: Distribution of the outcome variable

4.2 Linear Regression

Since our variable outcome was continuous, our first instinct was to try a simple multiple linear regression model. This model achieved a MSE estimate of 773.72 and an adjusted R^2 score of 0.184. The RMSE value of 27.82 seemed quite high, given that over 95% of our outcome values fell between 0 and 80.



```
                            OLS Regression Results
==============================================================================
Dep. Variable:     total_num_hurt_killed   R-squared:                       0.199
Model:                               OLS   Adj. R-squared:                  0.184
Method:                    Least Squares   F-statistic:                     13.57
Date:                   Thu, 17 May 2018   Prob (F-statistic):           1.67e-99
Time:                           12:25:52   Log-Likelihood:                -13799.
No. Observations:                   2838   AIC:                         2.770e+04
Df Residuals:                       2786   BIC:                         2.801e+04
Df Model:                             51
Covariance Type:               nonrobust
------------------------------------------------------------------------------
```

Figure 3: Summary of the Multiple Linear Regression model results

After dummifying the categorical values, we were left with over 50 variables. We were curious to see which predictors the model thought was significant, so we pulled up a summary table (Figure 3). The number of perpetrators (nperps), the target type (namely private citizens/property and religious figures/institutions, and whether the attack was a suicide were the most significant variables.

- **suicide**: p-value of10^{-3}
- **target type (religious)**: p-value of 0.009
- **suicide (private citizens/property)**: p-value of 0.0095
- **nperps**: p-value of 0.017

Since we had so many variables that offered no predictive power, we thought to use Lasso and Ridge Regression to perform feature selection/elimination. However, these methods ended up increasing our MSE (and adjusted R^2 and we realized that we needed more flexible models to capture the skewness in our data. The next algorithm we used was K-Neighbors Regression, which offered a bit more flexibility since we could change the K value.

4.3 K-Neighbors Regression

With K-Neighbors Regression, the models finds the closest K points to a given observation and outputs the average of the y values of the K neighbors. We originally started with a random K value of 5 and obtained a MSE of 632.60 and an adjusted R^2 of 0.04. Thus, using a more flexible model allowed us to decrease the MSE. In addition to using CV to estimate the test error, we used 3-fold CV to figure out best value of K. We used GridSearchCV to test K values between 1 and 100 and obtained that K=17 performed the best. It acheived an MSE of 529.47 and an adjusted R^2 of 0.19. In terms of both error metrics, this is a significant improvement from our Linear Regression model.

4.4 Decision Tree Regression

With Decision Tree Regression, we did not have high expectations, but we decided to use it as a baseline for the tree-based ensemble methods (namely Random Forest and Gradient Boosting). We knew that we would get high variance in our results within the MSE because our data is quite large while we also have a high number of features with high scatter in our data. One decision tree could not capture the complexity of our data. The MSE obtained was 598.45 and an adjusted R^2 value of 0.09; clearly not a great model, but the results were in line with our expectations.

4.5 Random Forest Regression

Random Forest was the natural go-to ensemble method. By taking numerous decision trees, we were able to significantly reduce variance while creating a model that would not overfit. The default Random Forest model obtained an MSE of 559.93 and an adjusted R^2 of 0.15. The default Random Forest model used 10 estimators, randomly picked predictor variables out of the total number of predictors. After tuning the number of estimators, the max depth, and the min sample leafs, we obtained a better model with an MSE of 522.03 and an adjusted R^2 of 0.21. Interestingly enough, this model that performed the best used all 53 predictor variables. Therefore, the Bagging model proved to be better than the more complex Random Forest model in this scenario.

4.6 Gradient Boosting Regression

We decided to choose Gradient Boosting Regression to learn more about ensemble methods. Just by using the default parameters, we obtained a MSE of 520.55 and an adjusted R^2 of 0.21; which was a clear winner off the bat (without any tuning). We therefore decided to try every possible number of predictor variables while trying different values for the learning rate and max depth. It turned out that our best model used 39 of the original predictors, had a learning rate of 0.1, and a max depth of 3. This dropped the MSE to 515.17 and raised the adjusted R^2 to 0.22, our best performing algorithm overall.

As mentioned earlier, we were interested in figuring which features were the most important in determining the total number of wounded/killed given the features we had. Figure 4 (below) shows us the most important/significant predictor variables in determining the deadliness of a given incident for our Gradient Boosting Model.

Not surprisingly, if an event is caused by a large number of perpetrators, a suicide bomber and/or if the target population is private citizens/property or religious figures, it is most likely that the event will result in a higher number of kills/injuries. A surprising statistic that we found was that if the attack took place in East Asia, it provided more predictive than any other region. Another surprising insight we found was that attacks with lower property damage (less than 1M USD worth) tended to be more of a predictor than attacks with higher property damage (over 1M USD but less than 1B USD worth). The same goes for attacks with catastrophic damage (greater than 1B USD), which held basically no predictive power. This seems quite counter-intuitive since one would think that the number of kills/injuries would be affected more by whether the event was catastrophic or not.

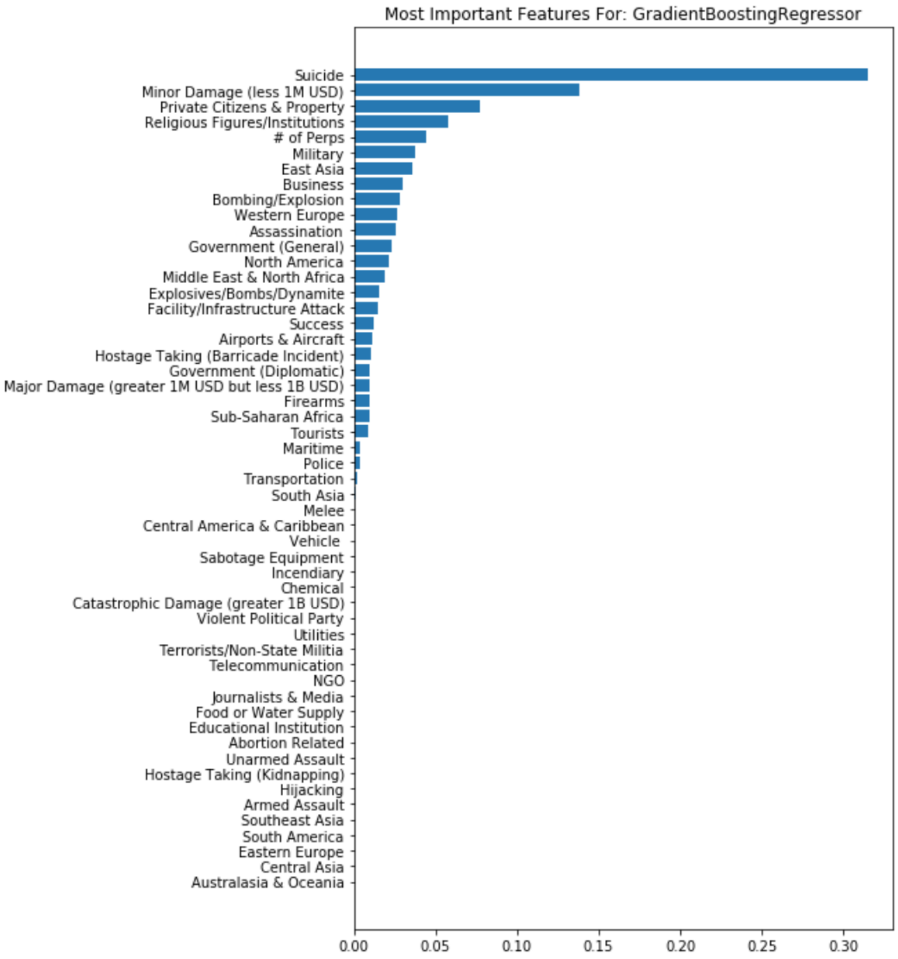


Figure 4: Significance of variables in predicting number of people hurt/killed

5 Conclusion

We realized from our testing of algorithms that as we went on, we needed a model that was more flexible from the initial thought of running a Linear Regression on the data. We were able to achieve decent results but however, could not realize the results that we wanted because of the sparsity of the

Comparison of Models: Before and After Tuning		
Model	Before	After
	(MSE, Adj R^2)	(MSE, Adj R^2)
Linear Regression	(775.06, 0.16)	-
KNN	(632.60, 0.04)	(529.47, 0.19)
Decision Tree	(597.7, 0.09)	-
Random Forest	(559.93, 0.15)	(540.13, 0.18)
Gradient Boosting	(520.30, 0.21)	(515.17, 0.22)

Figure 5: Model Comparison table

data. There are numerous predictors that we wanted to use that were unavailable to us. However, we realized that hyperparameter tuning was one of our biggest assets, in the sense that it helped increase the accuracy of our models. If we were to have continued with this experiment, one interesting idea would have been to implement a Natural Language Processing feature for every incident where we would look at some of the notes/summaries that were taken about each attack to obtain more in-depth knowledge of the keywords that are used to describe whether incidents were more or less deadly.

6 Bibliography

Data Source: https://www.kaggle.com/START-UMD/gtd

Documentation for scikit-learn: http://scikit-learn.org/stable/documentation.html

Documentation for Numpy: https://docs.scipy.org/doc/

Documentation for Pandas: https://pandas.pydata.org/pandas-docs/stable/