CMSE 381 Final Project
Police Killings

Sandeep Kashyap

April 18, 2021

1 Introduction

The unnecessary killings of civilians by police is both a singular and a serious problem in the United States. Police shoot and kill about a thousand civilians each year, far in excess of any other fully developed nation, and the existing evidence suggests that at least half and perhaps as many as 80 percent of these killings are not necessary. This project aims to develop a model to predict which race is at the highest risk of falling prey to needless violence.

To make these predictions, we will utilize features such as race, location, population distribution and income levels.

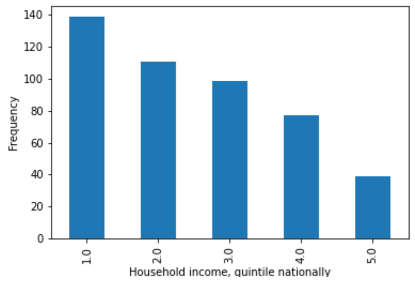
2 Related Work

Articles on the topic go into detail about the rate of police killings across the country. In a FiveThirtyEight article, the author Ben Cassleman states that in general police killing statistics are never accurate and are ”deeply flawed”. He talks about how police shootings more often than not happen in predominantly black neighborhoods which have fewer educational and job opportunities. But not all police killings fit this pattern. According to the report, a 57-year-old white man was shot in his home in Rockland County, New York where the median household income is more than 140K. There is also the case of the location of death i.e. the data shows where people died and not where they lived which led to a few cases such as a black woman who died in jail in Fairfax, Virgina which is one of the wealthiest tracts.

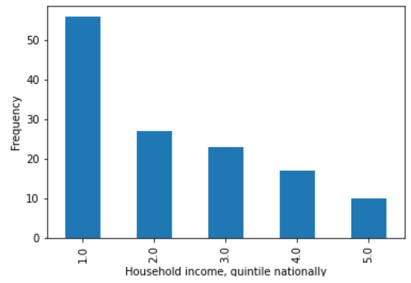
Another article on Nature also states that the ”police are disproportionately quick to shoot black civilians and those from other minority groups”, however various studies on the topic arrive at different conclusions on racial bias as a factor. Social scientists are now debating on including details such as the victim being armed or having a previous record, but one thing is for sure- ”police officers’ use of lethal force is much more common than previously thought”.

2.1 Regenerated Results

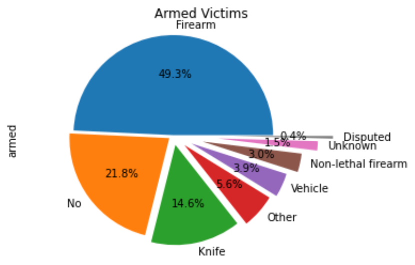
- Police killings tend to take place in neighborhoods that are poorer and blacker than the U.S. as a whole.
- 29.76 percent of the killings or 139 of the 467, took place in census tracts that are in the bottom 20 percent nationally in terms of household income.



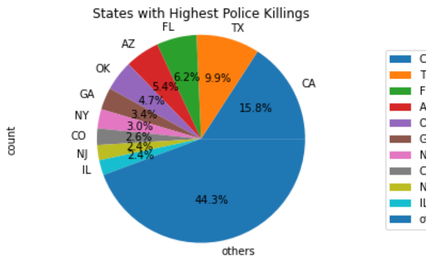
- A quarter of those killed by police died in tracts with majority-black populations
- 56 of the 136 African-Americans killed by police who are in the Guardian's database, more than 40 percent died in tracts in the poorest 20 percent nationally.



- Most of the victims in the Guardians's database were armed- Almost 50 percent had a firearm, 21.8 percent were unarmed and nearly 15 percent carried a knife.



- California with 15.8 percent, has the highest number of police killings in the country by a significant margin. Texas at 9.9 percent and Florida at 6.2 percent round out the top three.



3 Dataset

The main data source for this project is the public dataset for police killings nationally compiled by The Guardian. The dataset includes 467 entries each with 34 features.

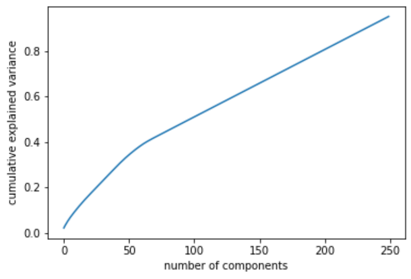
For the initial preprocessing, I inspected each feature of the dataset and made the following changes:

1. Set the missing values to zero where appropriate.
2. Converted certain numerical columns from string to float.
3. Removed irrelevant and uninformative features, for ex. street address, longitude, latitude.
4. Split the data into training and testing sets.

I performed One Hot Encoding on the dataset because many models cannot work with categorical data directly and need to be converted into numbers.

3.1 Principal Component Analysis

After doing One Hot Encoding on the dataset, I ended up with 416 columns. So to reduce the number of columns and still get an accurate model, I performed a PCA on the One Hot Encoded dataset and transformed the training and testing feature sets. The PCA model explained 95 percent variance using 250 components. These transformed datasets were then used in the Logistical Regression model and SVM model.



4 Methods

4.1 Random Forest Classifier

Random forest classifier is a supervised learning algorithm used for both classification and regression. This classifier gives the prediction result of the random samples with the most votes. First all the object type columns were label encoded using the preprocessing module from sklearn. I considered One Hot Encoding for this model, but upon futher research found that by one-hot encoding a categorical variable, we are inducing sparsity into the dataset which is undesirable. Then the features and labels dataset was created and the data was split into training and testing data. Finally we fit the training data and labels to the model and get the accuracy and classification report.

4.2 Logistic Regression

Logistic Regression is a supervised learning algorithm used mostly for classifying rather than regression. In this particular case, I used multi class logistic regression which follows an ”one vs all” algorithm. This model basically works by choosing all features one by one while putting the rest of them into a second class and running a binary logistic regression. The feature which has the highest probability is the result. To actually apply the logistic regression model, I applied One Hot Encoding to the dataset and then Principal Component Analysis on the training and testing feature set from derived from it. Finally we fit the training data and labels to the model and get the accuracy and classification report.

4.3 Support Vector Machine

Support Vector Machine is a supervised learning algorithm mostly for classification but also regression. This algorithm tries to find the optimal hyperplane which can be used to classify new data points. Unlike other models which learn the differences between classes, the SVM algorithm looks for similar examples across classes which are the support vectors. Finally we fit the training data and labels to the model and get the accuracy and classification report. I tried hyperparameter tuning by changing values of C and different types of kernels and I got the best accuracy for C = 0.5 and linear kernel.

5 Results

The accuracy rates, precision, F1 score and recall were used to evaluate the trained models. Training sets (327 examples) and testing sets (109 examples) were used to choose the best-performing models within each category. The test set, containing 109 entries, was used to provide an unbiased prediction, with the final models trained on both training and testing splits. Results for the testing and training splits respectively are given below:

Model Name	*Accuracy*	*Precison*	*Recall*	*F1 Score*
Random Forest Classifiers	0.73	0.70	0.73	0.71
Logistic Regression	0.66	0.66	0.66	0.65
Support Vector Machines	0.67	0.65	0.67	0.64

Model Name	*Accuracy*	*Precison*	*Recall*	*F1 Score*
Random Forest Classifiers	1.00	1.00	1.00	1.00
Logistic Regression	0.97	0.97	0.97	0.97
Support Vector Machines	0.99	0.99	0.99	0.99

The Random Forest Classifier performed the best out of all the models with an accuracy rate of 73 percent. It had a multi feature classification. The training accuracy in this case was 100 percent.

The next two models were Logistic Regression and Support Vector Machine. In both these cases, the data had to be One Hot Encoded as the models cannot deal with categorical data.

The Logistic Regression Classifier had an accuracy rate of 66 percent. Here also I calculated the training accuracy which was 97 percent. I tried different regression penalties but that did not work as both lasso regression and ridge regression ended up with almost the same accuracy rate. I suspect that this might be because l1 regression tries to reduce the number of features in an already reduced feature dataset due to the PCA.

The Support Vector Machine had an accuracy of 67 percent, almost the same as Logistic regression. In this model, the training accuracy was 99 percent. I tuned hyperparameters such as C and the kernel type in an effort to get a better accuracy. I found that the model is far more accurate on a linear kernel than a rbf kernel. The value of C had little effect on the accuracy.

What is interesting across all three algorithms is that training accuracy was far greater than the testing accuracy which is proof of overfitting. I took the weighted F1 score because it accounts for the sample size of each class unlike other averages.

6 Conclusion and Future Work

This project aimed to find the race that would most likely fall victim to police killings based on various factors such as location, income levels, poverty rates, age and gender. Machine learning models like Random Forest Classifier, Logistic Regression and Support Vector Machines were employed to try and get the best results. Initial results were promising but the fact that the training accuracy is 100 percent in Random Forest Classifier and greater that 95 percent in the other two models, speaks to severe overfitting which means that entries may just be chalked up to chance. Still, among the models, the Random Forest Classifier performed the best and may point to better results if the original dataset were bigger. The dataset was relatively small when compared to its features which led to overfitting and therefore causing the majority of inaccuracy.

For future work on this project, I would like to make use of the full Guardian dataset and in doing so also attempt a neural network and use cross validation. I would also like to explore other predictive questions that may arise from features in other datasets.

7 References

1. Casselman, Ben. "Where Police Have Killed Americans In 2015." FiveThirtyEight, 3 June 2015, https://fivethirtyeight.com/features/where-police-have-killed-americans-in-2015/

2. Analysis of Police Fatal Shootings in the U.S. 2020. databricks.com, https://databricks.com/blog/2020/11/16/fatal-force-exploring-police-shootings-with-sql-analytics.html

3. Brownlee, Jason. "How to One Hot Encode Sequence Data in Python." Machine Learning Mastery, 11 July 2017, https://machinelearningmastery.com/how-to-one-hot-encode-sequence-data-in-python/.

4. Karim, Raimi. "Intuitions on L1 and L2 Regularisation." Medium, 5 Oct. 2020, https://towardsdatascience.com/intuitions-on-l1-and-l2-regularisation-235f2db4c261.

5. Scikit-Learn: Machine Learning in Python — Scikit-Learn 0.24.1 Documentation. https://scikit-learn.org/stable/.