Predicting county-level crime statistics using machine learning

Jiaqi Chen, Xiaojie Pan, Anuj Patel, Guangzhe Zhu

Overview

In 1989, FBI initiated a crime statistics collection known as the National Incident-Based Reporting System (NIBRS), which is a national crime dataset containing incident-level data that allows for more detail and specificity. By using NIBRS, law enforcement agencies can have a more detailed perspective of crimes and better understand the context of crime in their respective jurisdictions. Moreover, because access to NIBRS is not restricted only to law enforcement, local policymakers and the public can also benefit from using NIBRS data. However, the structure of the NIBRS dataset is extremely complex. As a result of these features, there is no easy and accessible way for the end users to extract insights from NIBRS and make informed decisions. Therefore, the end goal of this project was to create a model that can predict potential crime in a given area at a given time that allows the end users to generate useful predictive insights according to their own perspectives.

Data Collection & Preparation

The NIBRS dataset is available on the FBI's Crime Data Explorer (CDE) web tool. In order to ensure the quality and completeness of the data used for our analysis, we limited our dataset to Virginia, a US state that has a high NIBRS participation rate, and the time scope of 2009-2017. We chose this range because the demographic data we wanted to incorporate into our project spans as far back as 2009, and because the most recent NIBRS dataset is from 2017. For the demographic data, we obtained county-level population numbers, median house prices, and employment-to-population ratios from the American Community Survey (ACS) 5-year estimates provided by the US Census Bureau from 2009-2017.

To prepare our data for model building, we first addressed the segmented structure of the NIBRS data. While NIBRS provides extensive details regarding the offenses, victims, arrestees, and possible motivations, we concluded that only details pertaining to the time, location, jurisdiction, and type of crime occurred could be used to build a crime prediction model and thus, excluding information that could be inaccurate or that couldn't be appropriately modeled. For example, one cannot assume the arrestees' information to be accurate as an arrest is not equivalent to a guilty judicial verdict. From the time information, we extracted the hour, day of week, date of month, and month of the offense as variables. From the location and jurisdiction information, we extracted the county in which the offense occurred. We also extracted from the jurisdiction information the number of officers employed in the county and at the time of the offense. Finally, we omitted any observations that had missing county or location information and joined our demographic information (population, housing prices, and employment) based on the location and time of the offense.

Methodology

Using H2O.ai's Python package to prepare and build our models, we split the dataset into train (40%), validation (30%) and test (30%) dataset. When looking for the optimal combination of hyperparameters in order to find the best model, we apply the gradient search hyperparameter selection procedure for random forest, gradient boosting, naive Bayes, and neural networks. For our random forest models: the number of trees could range from 1 to 100. The max depth could range from 1 to 11. For gradient boosting machine, the number of trees could range from 0 to 500 in intervals of 50. The max depth could range from 1 to 11. Sample rate could range from 1 to 11 by 0.1 for each step and so do column sample rate.

For the neural networks, we tuned the following parameters to optimize the classifier. Hidden layer structure identifies the number of layers and the number of nodes in each layer. We tried the following hidden layer structures: [160, 320], [90, 180], [320, 160, 80], [100], and [50, 50, 50, 50]. L1 could range from 0 to 0.1 and L2 could range 0 to 0.01. The L1 and L2 parameters are used to improve generalization add stability to the model. The input layer dropout ratio could range from 0 to 0.2 and is also used to improve generalization. For naive Bayes, Laplace can range from 0 to 10. All grid search processes were designed to stop after validation error did not decrease for 5 consecutive rounds. The remaining parameters were left at default values as implemented by H2O.ai.

Results

We first concentrated on our random forest models that used grid search for model tuning. The resulting best model showed that “incident hour” is the most important variable and “county” is the second (Figure 1). The log loss of training, validation, test metrics demonstrated values of 1.9345, 1.9318, and 1.9331 respectively. From Figure 2 (log loss versus number of trees), we observe that training errors are greater than validation errors, which means there is an underfitting problem in this model. Underfitting occurs when a model cannot fit training data well and is unable to summarize the new data. We then tried alternative models to compare.

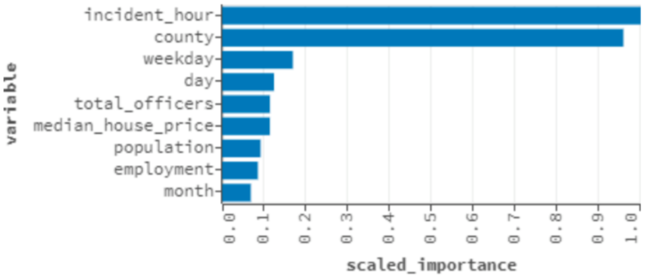

Figure 1: Random Forest Variable Importance

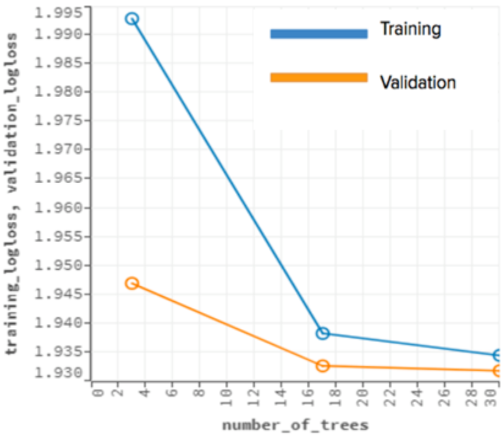

Figure 2: Random Forest Log Loss Scoring History

Gradient Boosting Machine (GBM)

Using the same sample data, we then implemented GBM. Like our random forest model, “county” and “incident hour” are still the two most important variables with “county” becoming the most important one (Figure 3). The best model after performing grid search on the hyperparameters resulted in log loss values of 1.8917, 1.9257, and 1.9271 for the training, validation, and test sets respectively. As shown in Figure 4 below, training errors are only slightly smaller than validation errors. In other words, although there is evidence of some overfitting, the extent of the overfitting is negligible compared to random forest. GBM also performs better compared to random forest with regards to the log loss values.

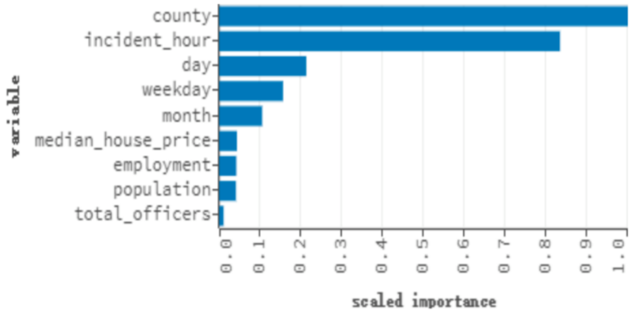

Figure 3: GBM Variable Importance

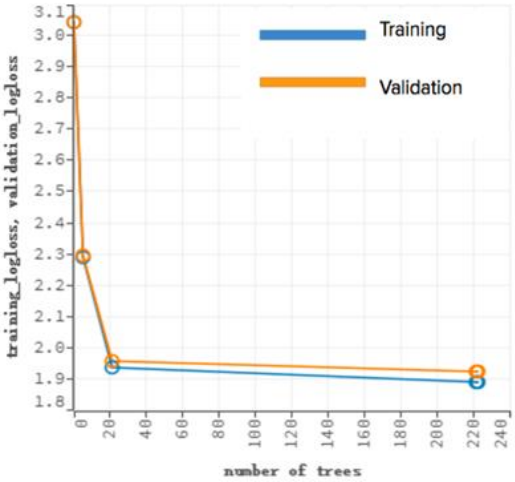

Figure 4: GBM Log Loss Scoring History

Neural Networks and Naive Bayes

To determine whether other machine learning models perform better than our GBM model, we also built neural networks and naive Bayes models. Table 1 summarizes the performance metrics of these models and we can observe that the log loss values did not improve relative to the random forest and GBM models.

Discussion

In this project, we evaluated the performance of models we built based on log loss values. Table 1 shows the summary of log loss for all four models mentioned above. Neural networks and naive Bayes result in larger log loss values compared to the first two models. Furthermore, we noted a serious underfitting problem in the random forest model, whereby the underlying algorithms cannot capture the pattern of data well. Therefore, we have chosen GBM as the preferred approach for our solution.

Table 1. Log Loss Metrics of Train, Validation, and Test Sets for All Models

Model	Train Log Loss	Validation Log Loss	Test Log Loss
Random Forest	1.9345	1.9318	1.9331
Gradient Boosting Machine(GBM)	1.8917	1.9257	1.9271
Neural Networks	2.1243	2.1134	2.1142
Naive Bayes	2.0529	2.0547	2.0566

Conclusion

In this project, we conducted the model that generates offense type prediction from the National Incident-Based Reporting System (NIBRS) dataset and complementary datasets collected from the United States Census Bureau. Since this is a supervised multi-layer dataset, the chosen target variable is the rate of offense types in the desired county regions. Our test trials highlighted important demographic information, such as population, housing prices, and employment, and several primary influential variables out of original 100+ variables, such as incident hour, day, year. After appropriately manipulating and preparing the dataset for model building, we pursued gradient boosting, random forest, naive Bayes, and neural networks models. Based on the results, the gradient boosting model is the best fit for our dataset. Our gradient boosting model has the lowest log loss among all the model. In summary, by learning from the dataset of labeled variables, we recommend the use of a gradient boosting model using grid search to optimize hyperparameters for the purpose of crime prediction.