RoboCop: Crime Classification and Prediction in San Francisco

John Cherian and Mitchell Dawson

December 11, 2015

Abstract

In this paper, we employ machine learning and other statistical techniques to the problems of classifying and predicting crimes in San Francisco. Drawing upon existing research in the field to approach these two problems, we employ Random Forest and VAR(p) models, respectively. For the classification problem, our results across all 39 crime categories demonstrate the difficulty of the fully-specified crime classification problem, as we achieve a maximum 39-way classification accuracy of 31.84%. Although our results are perhaps inappropriate for daily or weekly use in any police organization, the time series model performs adequately at forecasting crime incident averages in the coming weeks and months. With more data and the use of a time series model already developed by these authors for discrete time series, our results might be improved upon further.

1 Introduction

Criminal activity is inevitably a part of urban life. All city dwellers therefore have an interest in the improvement of our understanding of crime and its patterns. Police departments in particular could use this improved understanding to more effectively allocate their resources and better serve their communities. Knowing the spatial and temporal patterns of criminal activity would allow police to deploy the right officers where and when they are most needed, and being able to predict criminal activity would allow them to anticipate and combat surges in crime.

The goal of our project is twofold: first, to able to classify using machine learning techniques - in particular, a Random Forest model - the type of criminal incident (e.g. theft, DUI, prostitution) given data about the incident's location and time of occurrence; and second, to able to predict increases and decreases in the the rates of specific categories of crime given past rates using a VAR(p) time series model.

2 Related Work

The literature on crime classification is highly limited when compared to most applied machine learning problems. However, a few papers shed some light on the topic. For example, Bogomolov's MIT paper [1] on crime classification contained a number of critical insights for our work. The most important of those was the superiority of the Random Forest classifier for this problem. This assertion (which was confirmed by empirical analyses of various off-the-shelf classifiers) was also corroborated by another paper from Nasridinov, et.al. [2], which advocated the use of decision trees.

Other papers helped us figure out how to construct our feature set. Although the Wang et.al. paper is focused on finding patterns in crime data, it suggested that location and time data by itself would be inadequate for any serious crime analysis problem. For instance, Wang et.al.'s pattern analysis suggests that the locations and times of burglars tends to vary significantly over time [3]. In contrast, the Bogomolov paper suggested that demographic data constituted a particularly useful set of features for his crime classification problem [1].

On the issue of time series analysis, a couple different papers suggested that an ARIMA family model would likely give us reasonable, though fragile results [4][5]. However, the papers we were able to find that were focused on forecasting homicide crime rates, etc. nearly always used univariate time series models with exogenous variables. We felt that this was a potential area for improvement because a multivariate model could take correlations between different types of crimes into account.

3 Dataset

Our dataset consists of criminal incidents drawn from the San Francisco Police Department's Crime Incident Reporting system, and was made available through a Kaggle competition [6]. The dataset contains 878049 examples, each consisting of a timestamp (date and time of day), one of 39 crime categories (what we wanted to predict), a short description of the incident, the day of week, the police district in which the incident occurred, the resolution of the incident, the address, the longitude, and the latitude. We ignored the description and resolution fields as both did not appear in the competition's test set. We ignored the address field, as it was difficult to work with and redundant given the police district, longitude, and latitude fields. We also did not use the competition's test set, as it did not contain crime category information. Additionally, we gathered demographic data such as per capita income, racial composition, and median age from the 2010 American Community Survey (ACS) to enhance our feature set. Specifically, for each criminal incident in the data set we used the given coordinates to find demographic data associated with the Census block in which the incident occurred.

For the time series analysis, we aggregated the data on a biweekly basis because data from every other week was removed from the provided training set so that Kaggle could do out-of-sample testing. We then constructed time series covering 321 two-week periods for each crime class with a total number of crimes exceeding 2000. See Figure 1 for an example.

4 Methods

4.1 Classification

The primary machine learning method we employed for classification was an ensemble method called Random Forest. A Random Forest is defined as follows [7]:

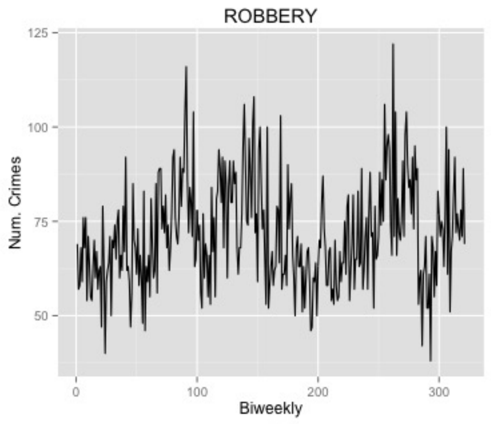

Figure 1: Robbery time series

A collection of decision trees $\{T_b\}$ is constructed from B random samples drawn from the training set with replacement. Each decision tree is grown by repeating the following steps for each node in the tree until we reach the limits on the size of the tree:

Randomly select m out of the p feature variables. Split the decision tree on the variable that maximizes the Gini impurity associated with the variable. For reference, the Gini impurity is a measure of how homogeneous the set is. It is defined as follows (f_i is the fraction of elements labeled i in the set):

$$I_g(f) = \sum_{i=1}^{m} f_i(1 - f_i)$$

After the decision trees are constructed, the Random Forest classifies the test data based on whichever class received the most votes from the B decision trees.

4.2 Time Series

The time series model we employed was the Vector Auto-Regressive model with lag p, or VAR(p).

$$N_t = \omega + \sum_{j=1}^{p} A_j N_{t-j}$$
$$N_t \in \mathbb{R}^k, A_j \in \mathbb{R}^{k\times k}$$

Using the MTS library in R[8], the time series model was fit using least-squares, i.e. the parameters ω, A_j are chosen to minimize $(N_t - \omega - \sum_{j=1}^{p} A_j N_{t-j})^2$.

There are two particularly important assumptions made by this model. The first is that the data is weakly stationary, i.e. the following statements must hold:

$$E[N_t] = \mu \qquad \text{For all } t$$
$$Cov(N_t, N_{t-j}) = \Gamma_j \qquad \text{For all } t$$

An Augmented Dickey-Fuller unit root test for stationarity[9], however, reveals that no crime class time series exhibits any significant deviations from stationarity. The critical value observed for each categorical time series is significant at the 5 percent level if it exceeds 2.86. Though some do come close, none of the critical values exceed that threshold.

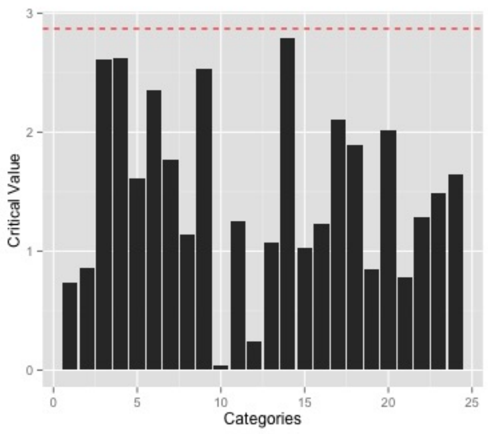

Figure 2: Critical values from Dickey-Fuller tests on each time series

It is assumed that the residuals are drawn from a $\mathcal{N}(0, \sigma^2)$ distribution. While the residuals are likely not "truly" Normal, the average p-value of notoriously oversensitive Shapiro-Wilk tests of normality performed on the residuals of each time series is 0.1093, well above the 0.05 or 0.01 commonly used as a threshold for significance.

For the detrended time series model, we "detrend" the time series data by computing the linear least-squares fit for each class time series and subtracting the resulting function from the data. This is implemented in the PRACMA package in R[10].

5 Results

5.1 Classification

5.1.1 Establishing a Baseline

For classification, we use as our accuracy measure the number of correctly labeled examples divided by the total number of examples. Unless specified otherwise, all test accuracies were estimated using 6-fold cross validation.

We first ran a number of standard classification algorithms on our data, including the Random Forest classifier as recommended in the literature. The results are shown below in Table 1. The training accuracies were calculated on the entire dataset.

Table 1: Accuracies of 4 Classification Algorithms

Algorithm	**Training Accuracy**	**Test Accuracy**
Logistic Regression	20.00%	19.99%
SVM	$< 70\%$	$\approx 25\%$
Naive Bayes	22.74%	22.73%
Random Forest	84.68%	29.31%

The results for Logistic Regression and Naive Bayes were little better than the accuracy of the naive classifier that always chooses the most common crime category (about 19.92%). We found that the SVM took much too long to run on any reasonably sized subset of the data, so the accuracies for SVM above are rough estimates based on very small subsets. The results for Random Forest were somewhat promising compared to the others, however.

5.1.2 Tuning Random Forest

Our next step was to attempt to tune the hyperparameters of Random Forest to reduce overfitting. The hyperparameter most relevant to this task was the maximum depth of the decision trees of which the Random Forest is composed. We systematically estimated the test accuracy for different maximum depths while holding the number of trees constructed constant at 60. The value that gave the best test accuracy was 21. Figure 3 shows how accuracy varied with changes in maximum depth. We performed a similar search for the hyperparameter that determines the number of individual decision trees constructed by the Random Forest. We found that both test and training accuracy monotonically increased as we increased number of trees, until we reached a point where training the model became computationally infeasible with our available resources. We therefore chose 60, a value on the upper end of the range for which we could train our model in a reasonable amount of time. With these hyperparameters set at these values, we achieved an estimated test accuracy of 31.70% and a training accuracy of 74.70%. The confusion matrix produced by this model appears in Figure 4.

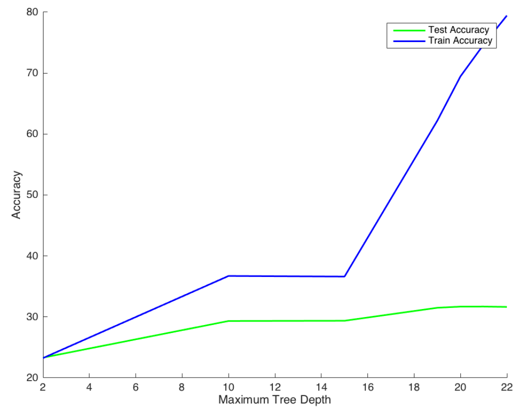

Figure 3: Model accuracy vs. tree depth

5.1.3 Adding Demographic Data

While tuning the hyperparameters of the Random Forest somewhat mitigated the overfitting of the model, the bias of the model remained high. We next sought to the reduce this bias by gathering demographic data from the U.S. Census' ACS and using this new data to expand our feature set. At first, we aggregated the data by hand from a published profile of San Francisco neighborhoods from the 2005-2009 ACS [12]. Adding these demographic features to our dataset did nothing to enhance the accuracy of our model.

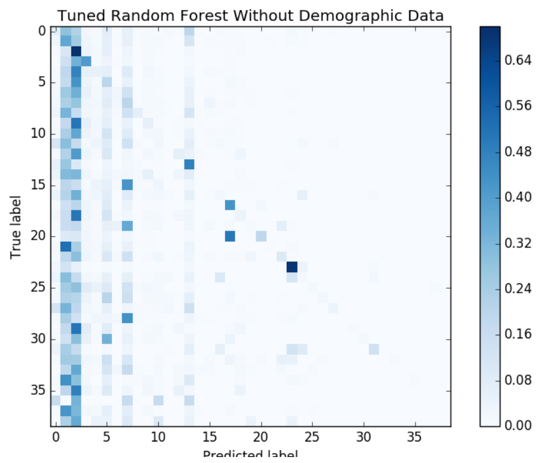

Figure 4: Confusion matrix w/o demographic data

At the poster session, we were alerted to the existence of the U.S. Census' ACS API [13] by the other group working with the same San Francisco crime dataset, and we used this to gather more accurate, fine-grained data (the specifics of these data are detailed in the Dataset Section above). Using these new demographic features improved the test accuracy of our model very slightly, resulting in an estimated test accuracy of 31.84%. The training accuracy dropped slightly to 71.65%.

5.1.4 Adding Higher Order Features

We next attempted to reduce the bias of our model by building higher order features - such as longitude^2 or time of day$\times$ latitude - and adding them to our feature set. Because of the large size of our dataset and consequent long training times with our initial set of features, we chose to only work with the most important features as measured by the Gini importance, which is a measure of the informativeness of a feature based on how much Gini impurity is decreased by nodes that split on that feature [14]. Latitude, longitude, and time of day were the 3 most important features. For a given degree n, we calculated all higher order features of degree less than or equal to n involving those three features and added them to our feature set. Figure 5 below shows how the estimated test accuracy varied with the degree n. To make this computationally feasible, we removed the demographic data and used a smaller (20) number of trees when estimating the accuracies.

The variations were small, but the estimated test accuracy tended to decrease as n increased. The slight increase in accuracy for $n = 2$, and indeed all the variations in accuracy, may have been due to the randomness of the Random Forest, especially since we used a smaller number of decision trees in the forest, increasing the variance of the forest's outputs.

5.2 Time Series

5.2.1 Motivations

The VAR(p) model is uniquely capable of finding trends in this set of crime data. Auto-correlation function (ACF)

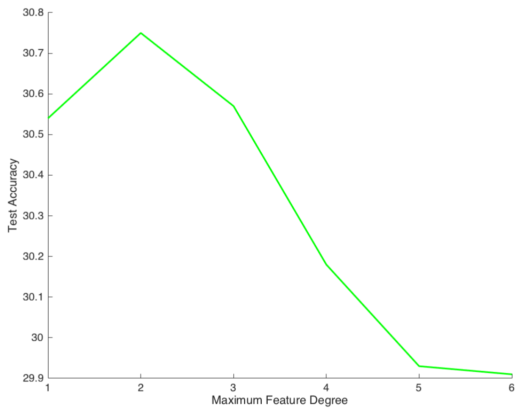

Figure 5: Test accuracy vs maximum feature degree

and cross-correlation (CCF) plots between the different categories of crimes revealed significant statistical connections that the coefficient matrix in a multivariate time series model like VAR(p) can capture. Figure 6 below shows two particularly informative plots. For example, in this case, the

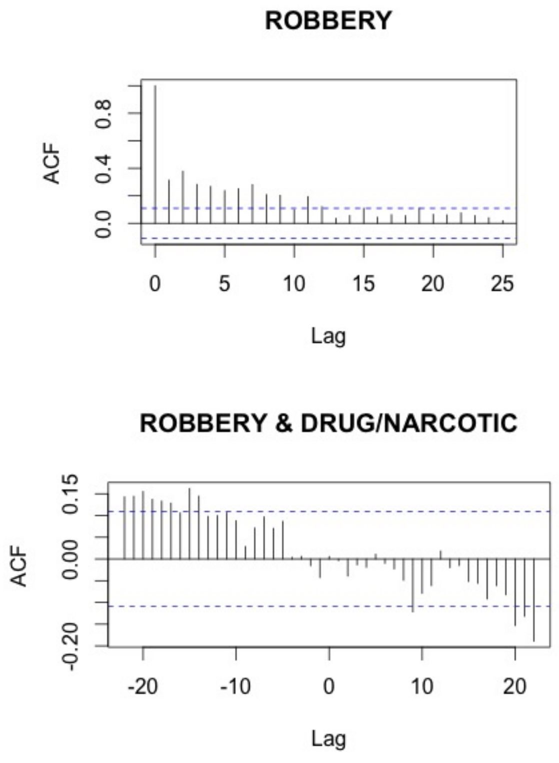

Figure 6: ACF and CCF plots

cross-correlation function between robberies and drug crimes might imply that an increase in drug arrests precedes a decrease in robberies. This and other similar correlations are identified and exploited by the matrix coefficients of the VAR(p) model.

5.2.2 Model Selection

Choosing p for the VAR(p) is a feature selection problem. However, in this case, instead of using cross-validation, we fit the model with different values of p, and calculated the model AIC (Akaike Information Criterion) and BIC (Bayesian Information Criterion). As we can in Figure 7, p = 1 appears to be optimal. Although each of the time series passes system-

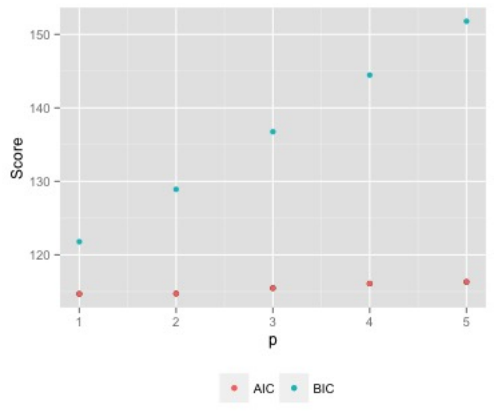

Figure 7: AIC/BIC

atic stationarity tests, it is evident that certain series exhibit a trend over time. To account for this, we also fit the model to detrended data (the process of detrending is described in further detail in the Methods section).

5.2.3 Fit Results

What follows are two charts of the 20-week forecast (with a 95 percent confidence interval) of the unmodified and detrended model trained on "ROBBERY" time series data. These are Figure 8 and Figure 9, respectively. As one can see, this model is unable to capture the week-to-week variation, but does adequately predict the underlying trend.

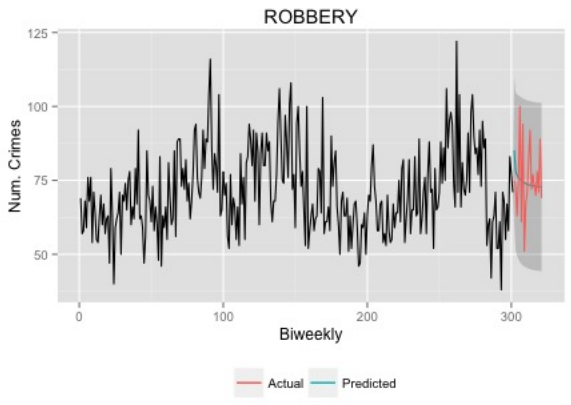

Figure 8: Forecast on unmodified data

In Table 2, we present the complete fit results for the original time series and the detrended time series model. For error data, we use normalized root-mean-square error (nRMSE)

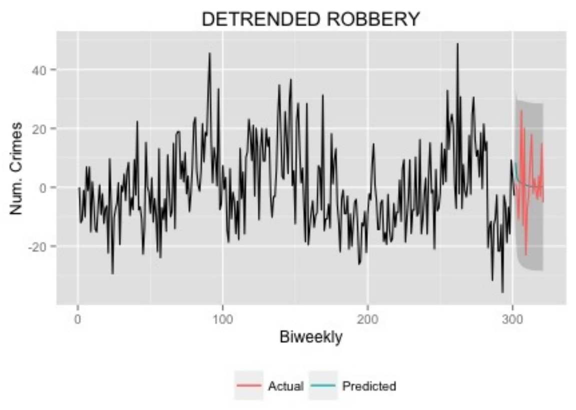

Figure 9: Forecast on detrended data

from cross-validation on a sliding window of size 100. To calculate nRMSE, the root-mean-squared-error for each time series is normalized by the standard deviation of the observed data.

Table 2: Time Series Errors (nRMSE)

Name	**Unmodified**	**Detrended**
Assault	4.6578	4.6295
Burglary	6.0505	5.9080
Disorderly Cond.	5.6276	5.6068
DUI	5.0666	4.9672
Drugs	6.3099	6.0101
Drunkenness	4.9411	4.9605
Forgery	7.7357	7.1147
Fraud	4.8329	4.8197
Kidnapping	5.0244	4.9380
Larceny	6.0363	6.2834
Missing Person	4.7970	4.7875
Non-Criminal	6.5049	6.6939
Other Offenses	5.8062	5.8096
Prostitution	5.8834	5.4976
Robbery	5.5518	5.6401
Secondary Codes	5.0086	4.8826
Sex Offenses (For.)	4.6517	4.6294
Stolen Property	5.3639	5.2323
Suspicious Occur.	5.2512	5.3832
Trespassing	4.7736	4.7754
Vandalism	5.3609	5.3498
Vehicle Theft	8.6704	8.9123
Warrants	5.1641	5.1969
Weapon Laws	4.6688	4.6602

6 Conclusion

In this paper, we used machine learning techniques to classify the type of crime that occurred given data on where and when it occurred and time series analysis to predict future patterns of crime given counts of past instances of crime. Specifically, we use a Random Forest model with added demographic data about the location of a crime's occurrence to achieve a 39-way classification accuracy of 31.84%. We believe Random Forest is better suited to this problem than other classification algorithms, as it makes no assumptions about the structure of the data. This is important since our dataset is highly noisy.

We used a VAR(p) time series model to achieve adequate predictive results for many crime categories. We believe that using the MINGARCH(p, q) model, which assumes the counts are drawn from a Poisson distribution (more appropriate for crime statistics), that John developed in his summer research would give more useful and accurate results. However, the currently implemented version of the model is insufficiently optimized for a larger dataset like ours. Also, adding data from cities like Chicago would help train more accurate, higher lag time series models.

References

[1] A. Bogomolov, et. al. Once Upon a Crime: Towards Crime Prediction from Demographics and Mobile Data. *MIT*, 2014. PDF.

[2] A. Nasridinov, S. Ihm, and Y. Park. A Decision Tree-Based Classification Model for Crime Prediction. *Information Technology Convergence*, 531-538, 2013. PDF.

[3] T. Wang, C. Rudin, D. Wagner, and R. Sevieri. Learning to Detect Patterns of Crime. *European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2013. PDF.

[4] Pepper, James V. "Forecasting Crime: A City-level Analysis." *Understanding Crime Trends: Workshop Report* (2007): 177-209. *Google Books*. Web. 11 Dec. 2015.

[5] Klepinger, Daniel H., and Joseph G. Weis. "Projecting Crime Rates: An Age, Period, and Cohort Model Using ARIMA Techniques." *Journal of Quantitative Criminology* 1.4 (1985): 387-416. *SpringerLink*. Web. 11 Dec. 2015.

[6] San Francisco Crime Classification Data. Kaggle. *https://www.kaggle.com/c/sf-crime/data.*

[7] Hastie, Trevor, Robert Tibshirani, and J. H. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* New York: Springer, 2009. Print.

[8] Tsay, Ruey S. *Multivariate Time Series (MTS)*. Program documentation. *R Project*. Vers. 0.33. The R Foundation, 11 Feb. 2015. Web. 11 Dec. 2015.

[9] Pfaff, Bernhard, and Matthieu Stigler. *urca*. Program documentation. *R Project*. Vers. 1.2-8. The R Foundation, 5 June 2013. Web. 11 Dec. 2015.

[10] Borchers, Hans Werner. *PRACMA*. Program documentation. *R Project*. Vers. 1.8.8. The R Foundation, 28 Oct. 2015. Web. 11 Dec. 2015.

[11] *Scikit-learn*. Program documentation. *Scikit-learn.org*. Vers. 0.17. Scikit-learn Developers, n.d. Web. 11 Dec. 2015.

[12] *San Francisco Neighborhoods Socio-Economic Profiles*. American Community Survey 2005-2009. San Francisco Planning Department, May 2011. *http://empowersf.org/wp-*

content/uploads/2014/03/SFProfilesByNeighborhood-SF-Planning-Dept..pdf. PDF.

[13] U.S. Census' American Community Survey API. *http://www.census.gov/data/developers/data-sets/acs-survey-5-year-data.html*.

[14] Louppe, Gilles, et al. Understanding variable importances in forests of randomized trees. *Advances in Neural Information Processing Systems*. 2013.