Crime Rate Prediction

Connor Secen, Tristan Marshall, and Richard Strouss

Abstract The use of machine learning techniques to predict crime rate has been common practice in many areas for several years. Being able to predict crime rate can help law enforcement focus on specific areas and manage resources more practically. However, these benefits can come at a cost, bringing in unwanted bias, resulting in certain races or socioeconomic group being flags as having a higher crime rate. This paper attempts to tackle these issues by comparing different machine learning technique to find the method that produces the highest accuracy, while also trying to remove bias from the system. Different datasets, one containing race data and one with race data removed, are compared using the same methods, to find if excluding bias heavy data can still yield acceptable prediction results.

Introduction

As the proliferation of data and computational capability increases, police departments increasingly use crime forecasting techniques to aid in resource management. This has sometimes been a contentious issue and is worthy of the attention of experts and non-experts alike. This paper attempts to navigate this contentious issue while providing some insight into the use of different techniques and their efficacy for crime rate prediction.

Background

The importance of having relevant features cannot be overstated in crime prediction. Many studies have reported their most relevant features when predicting crime rates. For example, Ingilevich and Ivanov found, in addition to population size, the number of bars, churches, and schools in an area were predictive of crime rates.[1] Alves, Ribiero, and Rodrigues found unemployment, illiteracy, and male population to be the most predictive features when they studied the crime rates of Brazilian cities.[2]

1. Ingilevich and Ivanov 2018.
2. Alves, Ribiero, and Rodrigues 2018.

Techniques

Just about every known machine learning technique has been used on crime prediction at some time and many techniques have been tested against each other. One study compared linear regression, logistic regression, and gradient boosted decision trees for predicting crime rights within different areas of Russian cities.[3] Between the three, linear regression actually had the best performance metrics. However, some predictions were considered impossible, such as negative crime rates, leading the researchers to judge their gradient boosted decision tree as the best technique for the purpose. It is conceivable that a technique could have been applied to prevent negative predictions and that would maintain relatively high performance. Both linear regression and decision trees have shown promise for crime rate prediction.

K-Nearest Neighbors has been a popular technique for use in crime rate prediction. However, results have been mixed. In some cases, KNN outperformed most other techniques including random forests, support vector machines, and naive bayes.[4] In others, KNN performed significantly worse when compared with other methods.[5]

Decision trees are commonly used for crime prediction, partly due their ability to handle non-linear, non-Gaussian data as well as their relative interpretability when compared to techniques such as neural networks.[6] The ID3 decision tree algorithm is a common simple decision tree implementation, but often ensemble techniques with multiple trees are used such as random forests, bagging, or boosting.

Other techniques that have been used include support vector machines and artificial neural networks. SVM has had mixed results and seems to often perform poorly compared to some other techniques.[7,8] Neural networks, especially deep neural networks, are often very accurate but have been criticized for the lack of interpretability.[9]

Ensembles

Ensemble techniques combine outputs from multiple models to make predictions that will, in theory at least, outperform any of the single models. Ensembles can be made of different types of models, such as SVM, naive Bayes, and decision trees, and have performed well.[10] Ensembles can also be made of the same or similar types of models. For instance, a random forest is an ensemble of decision trees, all made using slightly different datasets by selecting different features. Other useful ensemble

3. Ingilevich and Ivanov 2018.
4. Safat, Asghar, and Gillani 2021.
5. Abdulrahman and Abedalkhader 2017.
6. Alves, Ribiero, and Rodrigues 2018.
7. Safat, Asghar, and Gillani 2021.
8. Shamsuddin, Ali, and Alwee 2017.
9. Rudin 2019.
10. Kshatri et al. 2021.

methods that use the same classifier include baggin and boosting. In bagging, a random sample is chosen with replacement from the training set, on which a classifier is trained. Boosting is similar but each iteration of classifiers is biased to weigh those observations misclassified by previous classifiers more strongly. One popular boosting technique is called AdaBoost (for Adaptive Boosting). This technique is most often used for binary classifications but has been extended to work well for multi-class problems.[11,12,13] Most often this is done by turning a multi-class problem into an ensemble of binary class problems. AdaBoost as originally developed requires accuracies of at least 50

Data Preprocessing

To start out, the main dataset[14] was split into two datasets, one including all race related features and one excluding all race related features. This was done for comparison purposed to see we could still achieve the higher or at least similar accuracies without provide race related data. The crime rate (target value) was then converted from a continuous feature to a categorical feature to allow for the use of classification algorithms. This conversion was done by binning all the each value into one of ten categories separated by .1.

Many of the features in the dataset were correlated, so in order to reduce the dimensionality, a correlation was found between each feature and the target value. The features were then sorted by their correlation value progressively added, highest correlation to lowest, finding the accuracy for each new feature added.

Experiment

The experiment consists of comparing various classification algorithms, seeing which produces the best overall accuracy. It is broken down into four main methods, basic classification and three ensemble methods; voting, bagging, and boosting.

Basic Classification

To get a basic understanding of the data and it's predictive power, we start with a basic comparison between a logistic regression model and an ID3 decision tree model. Because there are a total of 11 classes that could be predicted, a one-vs-one method is used with the logistic regression algorithm. The methods are compared on how well they can predict on the race features included and excluded, using increasing number

11. Fleyeh and Davami 2013.
12. Eibl and Pfeiffer 2005.
13. Zhu et al. 2009.
14. Redmond 2009.

of features.

The logistic regression algorithm is performed using gradient descent. Equation 1 is the sigmoid function used in logistic regression. The derivative used in gradient descent for this method is equation 2.

$$J = \frac{1}{1 + e^{-z}} \tag{1}$$

$$dJ = \frac{1}{1 + e^{-z}}(1 - \frac{1}{1 + e^{-z}}) \tag{2}$$

Voting

The voting ensemble method is a combination of several weak classifiers. These classifiers include the logistic regression and decision tree classifiers that were used in the basic classification example along side a naive bayes and a k nearest neighbor classifier. For the knn algorithm, the value of $k = 4$ is used. The algorithm uses squared euclidean distance equation 3 as the distance function for determining how similar the examples are.

$$d(p, q) = \sum_{i=1}^{n} (q_i - p_i)^2 \tag{3}$$

Each weak classifier is trained on the exact same training set. Each classifier made a prediction for every instance in the validation set. Then for every instance all 4 predictions are compared and the mode is taken from these predictions. This mode is then the final prediction made by the voting ensemble method.

Bagging

The bagging ensemble method uses the ID3 decision tree as the weak classifier. From the training set of size N, N instance are selected at random with replacement. A decision tree is then trained on this new sub training set. This is repeated ten times, creating a total of ten different weak classifiers all trained on different data from the same initial training dataset.

After all of these weak classifiers are trained, each one is used to make a prediction on the validation set. Then between these ten predictions the mode prediction is selected as the overall prediction. This overall prediction is then the final prediction of the bagging ensemble method.

Boosting

Similar to bagging, the boosting algorithm used ID3 decision trees as weak classifiers and created ten classifiers ($C_i, i = 1...10$). However, each tree is limited to a depth of one. These trees are known as stumps. Each classifier is trained on a sample from the

initial training set (size=N), chosen with replacement. Different from bagging, each observation is given a weight determining it's probability of being included in the next sample. These are initialized equally for all observations as $\frac{1}{N}$. After each classifier is evaluated, the weights are updated so that incorrectly classified observations are more likely to be sampled, therefore becoming a greater proportion of the sample set for the next classifier. Each prediction will then use the results of all of these classifiers, weighted by a peformance measure β.

The first iteration begins the same as bagging, with a sample with replacement of size N and every observation having the same probability of being chosen. A stump (C_1) is trained on this sample, then used to predict the sample set on which it was just trained. The training error is calculated simply as the proportion of misclassified observations. A performance metric β is calculated from the error with K being the number of classes:

$$\beta = \log \frac{1 - error}{error} + \log(K - 1)$$

This metric is an update from that typically used in AdaBoost algorithms, $\log \frac{1-error}{error}$, and is based on the SAMME[15] method of extending AdaBoost from a binary classifier to multiclass. The addition of the $\log(K - 1)$ element prevents negative performance measures and ensures the updates move in the correct direction. AdaBoost performs poorly with the relatively high error rates of multiclass models and the addition of this element adjusts the metric taking into account the low expected accuracy for guessing, i.e. the expected performance from random chance, naively $\frac{1}{K}$. The stump is then used to predict the classes of the initial training set. The weight for each misclassified observation is adjusted by $weight * e^{\beta_1}$ and for each correctly classified observation by $weight * e^{-\beta_1}$. The weights are then normalized by dividing by the sum of the weights, ensuring they sum to one as should a probability distribution. The stump and its associated β are stored for later use in prediction.

The second iteration follows the same steps as the first with one key difference: since the observation weights have been adjusted, they are no longer equally likely to be chosen as part of the sample set. Instead, the probabilities are biased towards those observations misclassified by the previous stump. This causes these observations to be over-sampled and naturally puts a greater weight on the accuracy of predicting these observations in this iteration and potentially those to come. This is the only difference between the first iteration and all subsequent iterations of classifier C_i. For each iteration, the stump (C_i) and its β_i are stored.

To make predictions based on observations, each observation x must be passed through all of the trained weak classifiers C. The last step is to combine the outputs of the weak classifiers. This is done by linear combination of the β_i values for each k in

15. Zhu et al. 2009.

K classes. Thus, each prediction $\hat{Y}(x)$ is calculated as

$$\hat{Y}(x) = argmax_k \sum_{i=1}^{10} \beta_i * \mathbb{I}(C_i(x) = k)$$

This is the sum of all β_i values where observation x was predicted as k. The final prediction, $\hat{Y}(x)$, is given by assigning the class k for which this metric is highest.

Evaluation

Dataset

The dataset used in this project, "Communities and Crime", combines data from the 1990 US Census, law enforcement data from the 1990 LEMAS survey, and crime data from the 1995 FBI UCR. There are 128 total features, including the selected class feature violent crime rate. This dataset contains 1994 observations. Several of the features were incomplete and some were non-predictive, such as community name. These features were removed. With incomplete and non-predictive features removed, there were 102 features remaining. These make up the race-included dataset. The list of these features can be seen in appendix table 3 along with their signed correspondence to the target feature. With race-specific categories removed, there were 92 total features. These make up the race excluded dataset which can be seen in appendix table 2. The features themselves are already normalized and so z-scoring is not necessary. For a list of the features with their descriptions, see the communities.names file included with this project.

Results

Both datasets were fed through all seven classifiers implemented for the project. These include ID3 referred to as Decision Tree, Naïve Bayes, KNN, Logistic Regression, Voting Ensamble (which runs based on the first four classifiers), Bagging, and Adaboost. Further testing included reducing the features by keeping only the first N features with the features sorted in descending order based on the absolute value of their correspondence to the target feature. This was done because some features had very low correspondence values and could have negatively affected the capabilities of the classifiers. Each classifier was tested at every number of available features in the sorted order. The results of this can be seen in figures 1, 2, and 3. For comparison graphs of the race included and excluded results by classifier, run the accuracyGraphing.m script included with this project. Most classifiers will not be discussed on their own and so visuals were not included but are available in the aforementioned code. The last data point for the race included dataset was not included for the comparison data as it pulled focus from the main body of the data and results in a smaller relative scale of the main body of the data making it more difficult to visualize.

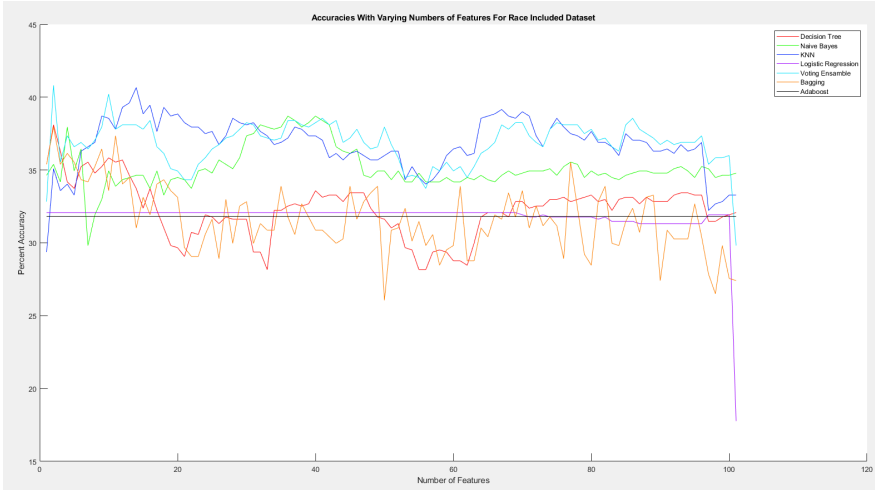

FIGURE 1. *Accuracies of Classifiers on Race Included Data with Feature Reduction Where Features are Added in Descending Order of Absolute Correspondence.*

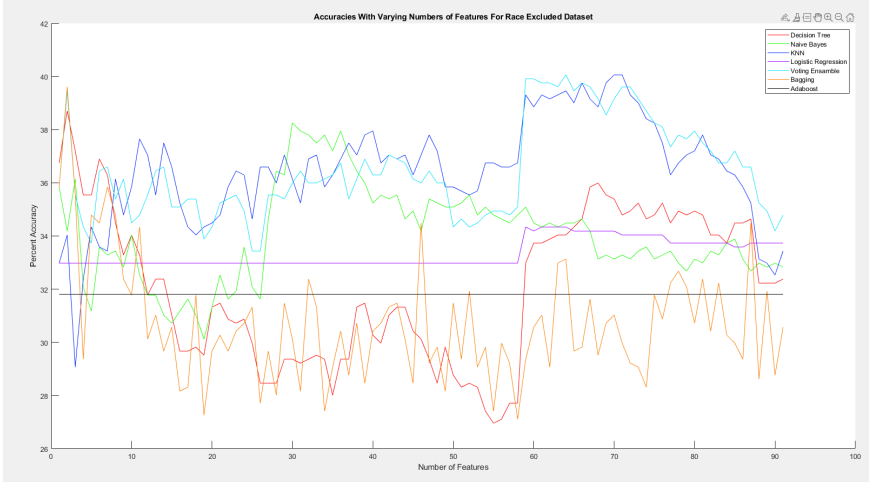

FIGURE 2. *Accuracies of Classifiers on Race Excluded Data with Feature Reduction Where Features are Added in Descending Order of Absolute Correspondence.*

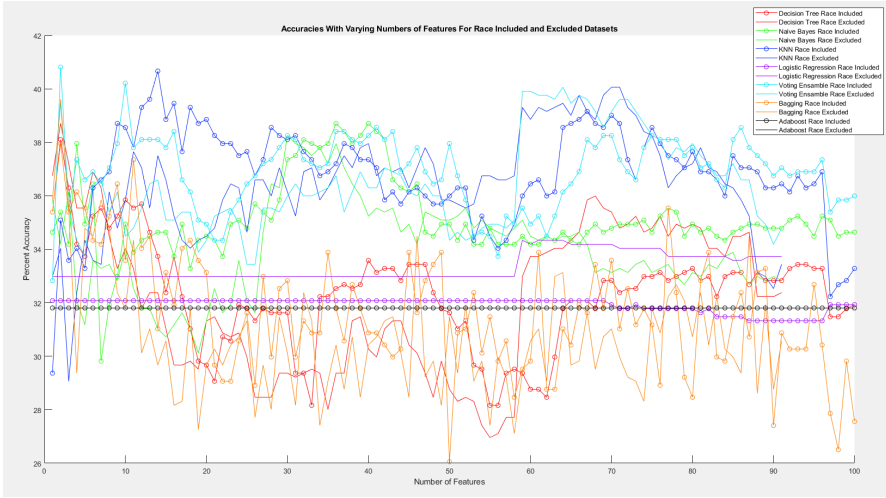

FIGURE 3. *Accuracies of Classifiers on Race Included and Excluded Data with Feature Reduction Where Features are Added in Descending Order of Absolute Correspondence for Comparison.*

Special Case – Logistic Regression and Adaboost.

Logistic Regression did not see as much variation in accuracy as the other classifiers. It stayed steady for most of the trials. A correspondence matrix for one of the data sets was generated. This table is visible below in table 1. Based on this confusion matrix, it is clear that the Logistic Regression classifier stuck to only the class buckets of 10% and 20% violent crime rates for that dataset. The Logistic Regression classifier was inspected but appeared to be functioning properly. The resulting conclusion is that this adherence to two bins is related to the content of the data. A similar result was observed with Adaboost.

		Predicted Class										
		0	10	20	30	40	50	60	70	80	90	100
Actual Class	0	0	98	4	0	0	0	0	0	0	0	0
	10	0	202	13	0	0	0	0	0	0	0	0
	20	0	105	17	0	0	0	0	0	0	0	0
	30	0	59	18	0	0	0	0	0	0	0	0
	40	0	33	16	0	0	0	0	0	0	0	0
	50	0	17	12	0	0	0	0	0	0	0	0
	60	0	12	3	0	0	0	0	0	0	0	0
	70	0	14	6	0	0	0	0	0	0	0	0
	80	0	7	4	0	0	0	0	0	0	0	0
	90	0	6	1	0	0	0	0	0	0	0	0
	100	0	9	8	0	0	0	0	0	0	0	0

TABLE 1. *Confusion Matrix for Logistic Regression with All Features Included.*

Discussion of Results

The first notable result is that none of the classifiers achieved better than 40.81% accuracy. Discussion of why this may be the case is included later in this section. The next notable result is that between the race included and excluded datasets, both were more accurate than the other at many points throughout the data depending on the classifier and number of features used. This suggests there is no major benefit to including race data and it can even be harmful to accuracy in many cases depending on the features and classifier used.

The major point of discussion is, why did the classifiers perform so poorly? Many methods were tried, and they all performed in the same approximate range of 26n% to 41% accuracy. With all of the accuracies in the same general range, it is worth taking a closer look at the data itself. While reviewing the data, it became clear that there is a sizeable potential for error. First, the margin of error on census data is 10%. This gives a wide margin for all features pulled from census data. Furthermore, there are features like "speaks English well" which are not well defined. This means the feature could be affected by subjective assessment. What is considerable as speaking English well could be completely different in one person's opinion versus another. Without any indication of a more objective backing for this feature, it is possible this feature, and others, could be subjective and thus have poor correlation between observations depending on who collected that feature for each observation. Furthermore, there are features which should not technically contain non-zero data, such as the "other" category for race. The other major race features White, Black, Asian, and Indian should cover heritage from all historically inhabited regions based on how they were described in the data. As a result, the "other" categories should not contain any people if the data is entirely accurate. Overall, the data is vulnerable to many potential sources of inaccuracy and could be inaccurate enough itself to severely reduce classification accuracy.

Conclusion

In conclusion, while the efforts in this paper did not reach the goal of an effective classifier for predicting violent crime rates based on area attributes, important results were achieved. The body of classifiers already attempted provide background for future attempts and the interpretation of the data provides insight into the faults of the data and how it may be improved for more successful future attempts. Furthermore, the comparison of the race included and excluded datasets demonstrates that the inclusion of race data in the classifiers does not significantly aid classification and instead fluctuates between minor increases and decreases in accuracy. The comparison results suggest that race data need not be included in future attempts; thus, addressing the issues surrounding the usage of such data in other crime classifiers. While the goal of the efforts described in this paper were not all reached, the knowledge gained will be beneficial for future works.

Future Work

The algorithms used in this project were all basic or weak classifiers. While ensemble methods were used to increase the performance of these classifiers, a possible future extension of this project could be building a more complex model, such as an artificial neural network. Neural networks have a proven track record for being able to learn complex functions that model data well and produce good predictive results. Implementing a neural network to predict crime rate would most likely be able to produce better results than the weak classifiers used in this project and could be more efficient that running multiple models for the same task. Alternatively, instead of using a classifier, the learning task could instead be a regression task, allowing for a more granular prediction and providing more options in terms are possible algorithms to use.

Another extension of this project could be using the same classification system used, but recollecting the data. Recollecting the data could provide many advantages. One, recollecting data would allow the addition of examples which would help to reduce overfitting to the training set. The features collected could also be more focused on features that seem to be better predictors. Collecting new features would also allow for more uncorrelated features to be added, which would reduce the overall number of features needed. Focusing on fewer, but more predictive features tends to yield better results and allows for more explainability.[16]

Appendix

TABLE 2. *Excluded Race Data, Names and Correspondence.*

Begin of Table	
FeatureNames	Correspondence
PctIlleg	0.737956549858663
pctWPubAsst	0.574665267716506
FemalePctDiv	0.556031877334695
TotalPctDiv	0.552777386598994
MalePctDivorce	0.525407284155456
PctPopUnderPov	0.521876541271772
PctUnemployed	0.504234642099599
PctHousNoPhone	0.48824345179297
PctNotHSGrad	0.483365881577958
PctVacantBoarded	0.482815798946848
PctHousLess3BR	0.47448990318547
NumIlleg	0.47102814202191
PctPersDenseHous	0.452900906934008
NumUnderPov	0.44758164147656
HousVacant	0.421395796121031

16. Rajpurohit 2017.

Continuation of Table 2	
FeatureNames	Correspondence
PctLess9thGrade	0.411095513368354
PctLargHouseFam	0.38347973476424
NumInShelters	0.375754152273157
population	0.367157378273526
PctWOFullPlumb	0.364453920508293
numbUrban	0.362897442225776
NumStreet	0.340276799399366
MedRentPctHousInc	0.325045261480456
MalePctNevMarr	0.30458292348832
PctNotSpeakEnglWell	0.300019042069333
PctOccupManu	0.295581227341957
PctLargHouseOccup	0.29480413944407
NumImmig	0.294239999945123
PctImmigRec10	0.291535996238266
PopDens	0.281389517379282
PctRecImmig10	0.264263148004992
PctRecImmig8	0.253197241155997
PersPerRentOccHous	0.248339786333025
PctImmigRec8	0.248135353362345
PctRecImmig5	0.248044621495733
PctRecentImmig	0.230768560562255
PctImmigRec5	0.216048975208697
LandArea	0.196799891154285
PctForeignBorn	0.194390437130095
PctImmigRecent	0.171877771659772
PctUsePubTrans	0.153828846739004
agePct12t29	0.153356688262718
PersPerFam	0.140651345116907
pctWSocSec	0.117977155410145
agePct16t24	0.0993466802092155
pctUrban	0.0820254198830155
PctSameCity85	0.0755579221611243
agePct65up	0.0671714539885565
MedOwnCostPctInc	0.0638466830557437
agePct12t21	0.0604772486579485
MedOwnCostPctIncNoMtg	0.0537633108125448
PctVacMore6Mos	0.0212828282527316
PctSameState85	-0.0194589369986818
PctWorkMomYoungKids	-0.0225353625616705
fold	-0.0325766614237816
householdsize	-0.0349225953359325
PersPerOccupHous	-0.0397391054134632
PctEmplManu	-0.0449060026616663
PctEmplProfServ	-0.0714826115385987
PctBornSameState	-0.0771589882642183
pctWRetire	-0.0984422239111715
MedYrHousBuilt	-0.10999439407448
PersPerOwnOccHous	-0.124415955996766
PctWorkMom	-0.150558260794525
pctWFarmSelf	-0.153124350628868
PctSameHouse85	-0.155388184652079
OwnOccHiQuart	-0.172122091464392
OwnOccMedVal	-0.190724208919619

Continuation of Table 2	
FeatureNames	Correspondence
OwnOccLowQuart	-0.210549811617313
state	-0.211397515395696
RentHighQ	-0.232290234665964
MedRent	-0.239863776841334
RentMedian	-0.240493738143995
PctSpeakEnglOnly	-0.241465900328619
RentLowQ	-0.25184693692818
pctWWage	-0.305500465833001
PctBSorMore	-0.314675178037531
PctHousOccup	-0.319009654590186
PctEmploy	-0.331642828194539
PctOccupMgmtProf	-0.339109225463785
perCapInc	-0.352084729607661
MedNumBR	-0.357384742320778
medIncome	-0.424220616726126
medFamInc	-0.439107738058735
PctHousOwnOcc	-0.47068258933402
PctPersOwnOccup	-0.525491144285445
pctWInvInc	-0.576323643921112
PctTeen2Par	-0.661581644430408
PctYoungKids2Par	-0.666058895934797
PctFam2Par	-0.706667469156986
PctKids2Par	-0.738423802070445
End of Table	

TABLE 3. *Excluded Race Data, Names and Absolute Value Correspondence.*

Begin of Table	
FeatureNames	Correspondence
PctKids2Par	0.738423802070445
PctIlleg	0.737956549858663
PctFam2Par	0.706667469156986
PctYoungKids2Par	0.666058895934797
PctTeen2Par	0.661581644430408
pctWInvInc	0.576323643921112
pctWPubAsst	0.574665267716506
FemalePctDiv	0.556031877334695
TotalPctDiv	0.552777386598994
PctPersOwnOccup	0.525491144285445
MalePctDivorce	0.525407284155456
PctPopUnderPov	0.521876541271772
PctUnemployed	0.504234642099599
PctHousNoPhone	0.48824345179297
PctNotHSGrad	0.483365881577958
PctVacantBoarded	0.482815798946848
PctHousLess3BR	0.47448990318547
NumIlleg	0.47102814202191
PctHousOwnOcc	0.47068258933402
PctPersDenseHous	0.452900906934008
NumUnderPov	0.44758164147656
medFamInc	0.439107738058735

Continuation of Table 3	
FeatureNames	Correspondence
medIncome	0.424220616726126
HousVacant	0.421395796121031
PctLess9thGrade	0.411095513368354
PctLargHouseFam	0.38347973476424
NumInShelters	0.375754152273157
population	0.367157378273526
PctWOFullPlumb	0.364453920508293
numbUrban	0.362897442225776
MedNumBR	0.357384742320778
perCapInc	0.352084729607661
NumStreet	0.340276799399366
PctOccupMgmtProf	0.339109225463785
PctEmploy	0.331642828194539
MedRentPctHousInc	0.325045261480456
PctHousOccup	0.319009654590186
PctBSorMore	0.314675178037531
pctWWage	0.305500465833001
MalePctNevMarr	0.30458292348832
PctNotSpeakEnglWell	0.300019042069333
PctOccupManu	0.295581227341957
PctLargHouseOccup	0.29480413944407
NumImmig	0.294239999945123
PctImmigRec10	0.291535996238266
PopDens	0.281389517379282
PctRecImmig10	0.264263148004992
PctRecImmig8	0.253197241155997
RentLowQ	0.25184693692818
PersPerRentOccHous	0.248339786333025
PctImmigRec8	0.248135353362345
PctRecImmig5	0.248044621495733
PctSpeakEnglOnly	0.241465900328619
RentMedian	0.240493738143995
MedRent	0.239863776841334
RentHighQ	0.232290234665964
PctRecentImmig	0.230768560562255
PctImmigRec5	0.216048975208697
state	0.211397515395696
OwnOccLowQuart	0.210549811617313
LandArea	0.196799891154285
PctForeignBorn	0.194390437130095
OwnOccMedVal	0.190724208919619
OwnOccHiQuart	0.172122091464392
PctImmigRecent	0.171877771659772
PctSameHouse85	0.155388184652079
PctUsePubTrans	0.153828846739004
agePct12t29	0.153356688262718
pctWFarmSelf	0.153124350628868
PctWorkMom	0.150558260794525
PersPerFam	0.140651345116907
PersPerOwnOccHous	0.124415955996766
pctWSocSec	0.117977155410145
MedYrHousBuilt	0.10999439407448
agePct16t24	0.0993466802092155

Continuation of Table 3	
FeatureNames	Correspondence
pctWRetire	0.0984422239111715
pctUrban	0.0820254198830155
PctBornSameState	0.0771589882642183
PctSameCity85	0.0755579221611243
PctEmplProfServ	0.0714826115385987
agePct65up	0.0671714539885565
MedOwnCostPctInc	0.0638466830557437
agePct12t21	0.0604772486579485
MedOwnCostPctIncNoMtg	0.0537633108125448
PctEmplManu	0.0449060026616663
PersPerOccupHous	0.0397391054134632
householdsize	0.0349225953359325
fold	0.0325766614237816
PctWorkMomYoungKids	0.0225353625616705
PctVacMore6Mos	0.0212828282527316
PctSameState85	0.0194589369986818
End of Table	

TABLE 4. *Included Race Data, Names and Correspondence.*

Begin of Table	
FeatureNames	Correspondence
PctIlleg	0.737956549858663
racepctblack	0.631263634659703
pctWPubAsst	0.574665267716506
FemalePctDiv	0.556031877334695
TotalPctDiv	0.552777386598994
MalePctDivorce	0.525407284155456
PctPopUnderPov	0.521876541271772
PctUnemployed	0.504234642099599
PctHousNoPhone	0.48824345179297
PctNotHSGrad	0.483365881577958
PctVacantBoarded	0.482815798946848
PctHousLess3BR	0.47448990318547
NumIlleg	0.47102814202191
PctPersDenseHous	0.452900906934008
NumUnderPov	0.44758164147656
HousVacant	0.421395796121031
PctLess9thGrade	0.411095513368354
PctLargHouseFam	0.38347973476424
NumInShelters	0.375754152273157
population	0.367157378273526
PctWOFullPlumb	0.364453920508293
numbUrban	0.362897442225776
NumStreet	0.340276799399366
MedRentPctHousInc	0.325045261480456
MalePctNevMarr	0.30458292348832
PctNotSpeakEnglWell	0.300019042069333
PctOccupManu	0.295581227341957
PctLargHouseOccup	0.29480413944407
NumImmig	0.294239999945123

Continuation of Table 4	
FeatureNames	Correspondence
racePctHisp	0.293050218125069
PctImmigRec10	0.291535996238266
PopDens	0.281389517379282
PctRecImmig10	0.264263148004992
PctRecImmig8	0.253197241155997
PersPerRentOccHous	0.248339786333025
PctImmigRec8	0.248135353362345
PctRecImmig5	0.248044621495733
PctRecentImmig	0.230768560562255
PctImmigRec5	0.216048975208697
LandArea	0.196799891154285
PctForeignBorn	0.194390437130095
PctImmigRecent	0.171877771659772
PctUsePubTrans	0.153828846739004
agePct12t29	0.153356688262718
PersPerFam	0.140651345116907
pctWSocSec	0.117977155410145
agePct16t24	0.0993466802092155
pctUrban	0.0820254198830155
PctSameCity85	0.0755579221611243
agePct65up	0.0671714539885565
MedOwnCostPctInc	0.0638466830557437
agePct12t21	0.0604772486579485
MedOwnCostPctIncNoMtg	0.0537633108125448
racePctAsian	0.0376217867813556
PctVacMore6Mos	0.0212828282527316
OtherPerCap	0
PctSameState85	-0.0194589369986818
PctWorkMomYoungKids	-0.0225353625616705
fold	-0.0325766614237816
householdsize	-0.0349225953359325
PersPerOccupHous	-0.0397391054134632
PctEmplManu	-0.0449060026616663
PctEmplProfServ	-0.0714826115385987
PctBornSameState	-0.0771589882642183
indianPerCap	-0.0908538163486274
pctWRetire	-0.0984422239111715
MedYrHousBuilt	-0.10999439407448
PersPerOwnOccHous	-0.124415955996766
PctWorkMom	-0.150558260794525
pctWFarmSelf	-0.153124350628868
PctSameHouse85	-0.155388184652079
AsianPerCap	-0.155591664727308
OwnOccHiQuart	-0.172122091464392
OwnOccMedVal	-0.190724208919619
whitePerCap	-0.209272201390512
OwnOccLowQuart	-0.210549811617313
state	-0.211397515395696
RentHighQ	-0.232290234665964
MedRent	-0.239863776841334
RentMedian	-0.240493738143995
PctSpeakEnglOnly	-0.241465900328619
HispPerCap	-0.244552895666062

Continuation of Table 4	
FeatureNames	Correspondence
RentLowQ	-0.25184693692818
blackPerCap	-0.275391085713885
pctWWage	-0.305500465833001
PctBSorMore	-0.314675178037531
PctHousOccup	-0.319009654590186
PctEmploy	-0.331642828194539
PctOccupMgmtProf	-0.339109225463785
perCapInc	-0.352084729607661
MedNumBR	-0.357384742320778
medIncome	-0.424220616726126
medFamInc	-0.439107738058735
PctHousOwnOcc	-0.47068258933402
PctPersOwnOccup	-0.525491144285445
pctWInvInc	-0.576323643921112
PctTeen2Par	-0.661581644430408
PctYoungKids2Par	-0.666058895934797
racePctWhite	-0.684769576271543
PctFam2Par	-0.706667469156986
PctKids2Par	-0.738423802070445
End of Table	

TABLE 5. *Included Race Data, Names and Absolute Value Correspondence.*

Begin of Table	
FeatureNames	Correspondence
PctKids2Par	0.738423802070445
PctIlleg	0.737956549858663
PctFam2Par	0.706667469156986
racePctWhite	0.684769576271543
PctYoungKids2Par	0.666058895934797
PctTeen2Par	0.661581644430408
racepctblack	0.631263634659703
pctWInvInc	0.576323643921112
pctWPubAsst	0.574665267716506
FemalePctDiv	0.556031877334695
TotalPctDiv	0.552777386598994
PctPersOwnOccup	0.525491144285445
MalePctDivorce	0.525407284155456
PctPopUnderPov	0.521876541271772
PctUnemployed	0.504234642099599
PctHousNoPhone	0.48824345179297
PctNotHSGrad	0.483365881577958
PctVacantBoarded	0.482815798946848
PctHousLess3BR	0.47448990318547
NumIlleg	0.47102814202191
PctHousOwnOcc	0.47068258933402
PctPersDenseHous	0.452900906934008
NumUnderPov	0.44758164147656
medFamInc	0.439107738058735
medIncome	0.424220616726126
HousVacant	0.421395796121031

Continuation of Table 5	
FeatureNames	Correspondence
PctLess9thGrade	0.411095513368354
PctLargHouseFam	0.38347973476424
NumInShelters	0.375754152273157
population	0.367157378273526
PctWOFullPlumb	0.364453920508293
numbUrban	0.362897442225776
MedNumBR	0.357384742320778
perCapInc	0.352084729607661
NumStreet	0.340276799399366
PctOccupMgmtProf	0.339109225463785
PctEmploy	0.331642828194539
MedRentPctHousInc	0.325045261480456
PctHousOccup	0.319009654590186
PctBSorMore	0.314675178037531
pctWWage	0.305500465833001
MalePctNevMarr	0.30458292348832
PctNotSpeakEnglWell	0.300019042069333
PctOccupManu	0.295581227341957
PctLargHouseOccup	0.29480413944407
NumImmig	0.294239999945123
racePctHisp	0.293050218125069
PctImmigRec10	0.291535996238266
PopDens	0.281389517379282
blackPerCap	0.275391085713885
PctRecImmig10	0.264263148004992
PctRecImmig8	0.253197241155997
RentLowQ	0.25184693692818
PersPerRentOccHous	0.248339786333025
PctImmigRec8	0.248135353362345
PctRecImmig5	0.248044621495733
HispPerCap	0.244552895666062
PctSpeakEnglOnly	0.241465900328619
RentMedian	0.240493738143995
MedRent	0.239863776841334
RentHighQ	0.232290234665964
PctRecentImmig	0.230768560562255
PctImmigRec5	0.216048975208697
state	0.211397515395696
OwnOccLowQuart	0.210549811617313
whitePerCap	0.209272201390512
LandArea	0.196799891154285
PctForeignBorn	0.194390437130095
OwnOccMedVal	0.190724208919619
OwnOccHiQuart	0.172122091464392
PctImmigRecent	0.171877771659772
AsianPerCap	0.155591664727308
PctSameHouse85	0.155388184652079
PctUsePubTrans	0.153828846739004
agePct12t29	0.153356688262718
pctWFarmSelf	0.153124350628868
PctWorkMom	0.150558260794525
PersPerFam	0.140651345116907
PersPerOwnOccHous	0.124415955996766

Continuation of Table 5	
FeatureNames	Correspondence
pctWSocSec	0.117977155410145
MedYrHousBuilt	0.10999439407448
agePct16t24	0.0993466802092155
pctWRetire	0.0984422239111715
indianPerCap	0.0908538163486274
pctUrban	0.0820254198830155
PctBornSameState	0.0771589882642183
PctSameCity85	0.0755579221611243
PctEmplProfServ	0.0714826115385987
agePct65up	0.0671714539885565
MedOwnCostPctInc	0.0638466830557437
agePct12t21	0.0604772486579485
MedOwnCostPctIncNoMtg	0.0537633108125448
PctEmplManu	0.0449060026616663
PersPerOccupHous	0.0397391054134632
racePctAsian	0.0376217867813556
householdsize	0.0349225953359325
fold	0.0325766614237816
PctWorkMomYoungKids	0.0225353625616705
PctVacMore6Mos	0.0212828282527316
PctSameState85	0.0194589369986818
OtherPerCap	0
End of Table	

Github Repository

https://github.com/TnMarshall/CS613Final.git

References

Abdulrahman, N., and W. Abedalkhader. 2017. KNN Classifier and Naive Bayes Classifier for Crime Prediction in San Francisco Context. *Internation Journal of Database Management Systems* 9 (4).

Alves, L., H. Ribiero, and F. Rodrigues. 2018. Crime Prediction Through Urban Metrics and Statistical Learning. *Physica A* 55:435–443.

Eibl, G., and K-P. Pfeiffer. 2005. Multiclass Boosting for Weak Classifiers. *Journal of Machine Learning Research* 6:189–210.

Fleyeh, H., and E. Davami. 2013. Multiclass Adaboost Based on an Ensemble of Binary AdaBoosts. *American Journal of Intelligent Systems* 3 (2):57–70.

Ingilevich, V., and S. Ivanov. 2018. Crime Rate Prediction in the Urban Environment Using Social Factors. *Procedia Computer Science* 136:472–478.

Kshatri, Sapna Singh, Deepak Singh, Bhavana Narain, Surbhi Bhatia, Mohammad Tabrez Quasim, and G. R. Sinha. 2021. An Empirical Analysis of Machine Learning Algorithms for Crime Prediction Using Stacked Generalization: An Ensemble Approach. *IEEE Access* 9:67488–67500.

Rajpurohit, Anmol. 2017. Must-Know: Why it may be better to have fewer predictors in Machine Learning models? Blog. Available at <https://www.kdnuggets.com/2017/04/must-know-fewer-predictors-machine-learning-models.html>. Accessed 5 June 2021.

Redmond, Michael. 2009. Communities and Crime Data Set. Dataset. Available at <https://archive.ics.uci.edu/ml/datasets/Communities+and+Crime>. Accessed 9 May 2021.

Rudin, C. 2019. Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead. *Nature Machine Intelligence* 1:206–215.

Safat, W., S. Asghar, and S. Gillani. 2021. Empirical Analysis for Crime Prediction and Forecasting Using Machine Learning and Deep Learning Techniques. *IEEE Access* 9:70080–70094.

Shamsuddin, Nurul Hazwani Mohd, Nor Azizah Ali, and Razana Alwee. 2017. An overview on crime prediction methods. Paper presented at the 2017 6th ICT International Student Project Conference (ICT-ISPC), 1–5.

Zhu, J., S. Rosset, H. Zou, and T. Hastie. 2009. Multi-Class AdaBoost. *Statistics and its Interface* 2 (3):349–360.

Authors

Connor Secen is a first year graduate student at Drexel University, pursuing his masters in Artificial Intelligence and Machine Learning. He can be reached at cas598@drexel.edu.

Tristan Marshall is a fourth year undergraduate and graduate student at Drexel University, pursuing his bachelors in Electrical Engineering and his masters in Robotics and Autonomy. He can be reached at tlm428@drexel.edu

Richard Strouss is a data analyst and second year graduate student at Drexel University, pursuing his masters in Data Science. He can be reached at richard.j.strouss@drexel.edu