Crime Rate Estimation by Communities in U.S.

Kwanheum Cho ID: 2014147553

Computer Science, Yonsei Univ.

Abstract

In this paper, I explain my process about crime rate estimation by Machine Learning techniques. Used Data set[1] is combined socio-economic data from the ’90 Census, law enforcement data from the Law Enforcement Management and Admin Stats survey(LEMAS), and crime data from the 1995 FBI UCR. There are many attributes so i select meaningful variables that affect my goal. And also generate new variables by feature engineering. Through a great deal of effort, finally I got fine features and improve results

I. Introduction

From a Criminological point of view, there are three kinds of way to explain cause of crime(Biological, Psychological, and Sociological). Among them i focus on sociological aspect. Everyone belongs to their own community and each community has different charisteristics. I reasonably think that several features can affect crime rate positively or negatively. For example, it is generally known as that poor communities, like slum, tend to have high crime rate. It means crime rate is related to the economic factor. and also there exist another factors .
So I think the regression model is appropriate to make relation between several factors and crime rate.

II. Data Set

Data Set is already filtered by creator. It doesn’t include unrelated attributes from his point of view. There are 5-non-predicive variables, 124 independent variables and 18 dependent variables. I preprocess the data in my point of view. First, remove variables that have many missing values. For instance, Number of Police Car, Plice officers per population, and …etc are removed(total 23). The reason of these missing values are limitation of LEMAS survey.
Second, remove 16-dependent variables except 2 because former things are included to latter things so it is improper to include both of them. Remaining things are ViolentCrimesPerPopulation and nonViolentCrimesPerPopulation.
And sum two of them as one feature named as ‘CrimesPerPopulation’ which will be estimated later.
Lastly, I don’t use 5-non-predictive variables, like state, communityCode etc. At the result, remaining meaningful variables are 102.

III. Overall Approach

A. Regression model

My development tool is jupyter notebook and mainly using scikit learn[2] modules to test the data.
First, I use Linear Regression model and preprocessed data. Because of lots of variables, model has varying variable coefficients. In Figure.1, I find some variable mainly affect the result and lots of others cannot affect.
So I think another regression like Ridge or Lasso to regularize outlier.
And I adopt R2 score and adjusted R2 score to check model’s accuracy.
If I use more and more independent variables, regression model tends to increase R2 score[3] so I use adjusted R2 score at the same time to compare model that have different independent variables.

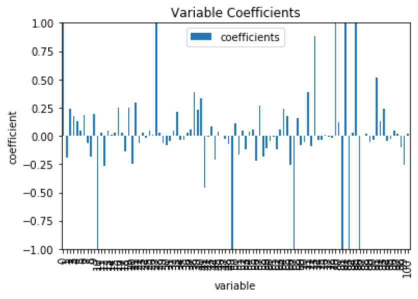


Figure 1. LinearRegression model’s coefficients

Next, I try Ridge Regression model. Ridge is different with Linear Regression in point of cost function[4]. In Figure.2, we can see a penalty that lambda multiply with square of coefficient(w). As a result it can regularizes the coefficient. After test several lambda values I select 0.01 and in Figure.3 Coefficients are regularized. Also obtain R2 and adjusted R2 score. Lastsly, I try Lasso Regression model. Lasso is also different with Linear Regression and similar with Ridge Regression in point of cost function[5]. In Figure.4, We can see a penalty that labmda multiply with absolute value of coefficient(w). After test several lambda values I select 0.00001 and in Figure.5 Coefficients are regularized. And also obtain R2 and adjusted R2 score.

$$\sum_{i=1}^{M}(y_i-\hat{y}_i)^2=\sum_{i=1}^{M}\left(y_i-\sum_{j=0}^{p}w_j\times x_{ij}\right)^2+\lambda\sum_{j=0}^{p}w_j^2$$

Figure 1. RidgeRegression cost function

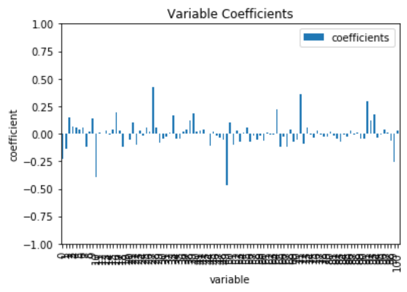


Figure 3. RidgeRegression model's coefficients

$$\sum_{i=1}^{M}(y_i-\hat{y}_i)^2=\sum_{i=1}^{M}\left(y_i-\sum_{j=0}^{p}w_j\times x_{ij}\right)^2+\lambda\sum_{j=0}^{p}|w_j|$$

Figure 4. LassoRegression cost function

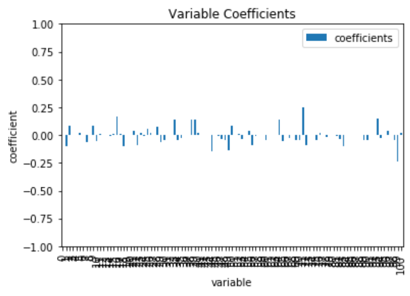


Figure 5. LassoRegression model's coefficients

In Figure.6, Comparing three model's accuracy, we can think Linear model is a little bit overfitted to train data and Lasso model is comparatively well trained because train and test score's gap is lowest. and Adj R2 score at test category is mainly meaningful, we can assume Lasso is good to this data because it have highest that score among them.

Model type	Linear		Ridge		Lasso	
case	Train	Test	Train	Test	Train	Test
R2 score	0.63953	0.60479	0.63347	0.60999	0.62504	0.61872
Adj R2	0.61930	0.58261	0.61290	0.58811	0.60401	0.59733

Figure 6. three model's accuracy

B. Feature Engineering

Step1, from data set, I select improper variables which is duplicate or unrelated in my perspective(total 9) and except from test. For example, 'persUrban'(=number of people living in urban) which is duplicated with 'pctUrban'(=percentage of people living in urban) and 'medNumBedrm'(=median number of bedrooms) which would not affect crime rate. After test , in Figure7 , All three model's accuracy improve.

Model type		Linear		Ridge		Lasso	
case		Train	Test	Train	Test	Train	Test
prior	R2 score	0.63953	0.60479	0.63347	0.60999	0.62504	0.61872
	Adj R2	0.61930	0.58261	0.61290	0.58811	0.60401	0.59733
After (step1)	R2 score	0.6348	0.61625	0.62775	0.61539	0.62050	0.62218
	Adj R2	0.61626	0.59673	0.60881	0.59582	0.60121	0.60297

Figure 7. three model's accuracy after step1.

Step2, try to generate new features. First, I think more homeless people, more crimes occur. And there are features 'NumStreet'(=number of homeless people in the street) and 'NumInShelters'(number of people in homeless shelters). So create 'HomelessPct' feature same as {(NumStreet + NumInShelters)/Population}.
Second, make economic features name as 'economicGapPerRace'. According to data, We can divide communities by race(white, black, asian and hispanic) and there are features 'racePerCap'(=per capita income for mentioned 4 race), 'perCapInc'(=per capita income), 'pctrace'(=percentage of population for mentioned 4 race). So create 'economicGapPerRace' feature same Figure8. Next, make another community economic feature that realated with housing cost and income. There are 'medRentpctHousInc'(=median gross rent as a percentage of household income) , 'medOwnCostpct'(= median owners cost as a percentage of household income – owners with a mortgage), and 'medOwnCostPctW0'(= median owners cost as a percentage of household income without a mortgage). And create 'HousingCostPerInc' feature same as {(medRentpctHousInc+medOwnCostpct+medOwnCostPctW0)}/3.

whiteIncFactor = whitePerCap / perCapInc , blackIncFactor = blackPerCap / perCapInc
AsianIncFactor = AsianPerCap / perCapInc, HispIncFactor = HispPerCap / perCapInc

whiteCommInc = whiteIncFactor * pctWhite, blackCommInc = blackIncFactor * pctBlack
AsianCommInc = AsianIncFactor * pctAsian, HispCommInc = HispIncFactor * pctHisp
TotalCommInc = whiteCommInc + blackCommInc + AsianCommInc + HispCommInc

whiteComm = whiteCommInc / TotalCommInc, blackComm = blackCommInc / TotalCommInc
AsianComm = AsianCommInc / TotalCommInc, HispComm = HispCommInc / TotalCommInc

economicGapPerRace = |whiteComm-blackComm| + |whiteComm-AsianComm| + |whiteComm-HispComm| + |blackComm-AsianComm| + |blackComm-HispComm| + |AsianComm-HispComm|

Figure 8. equation for 'economicGapPerRace'

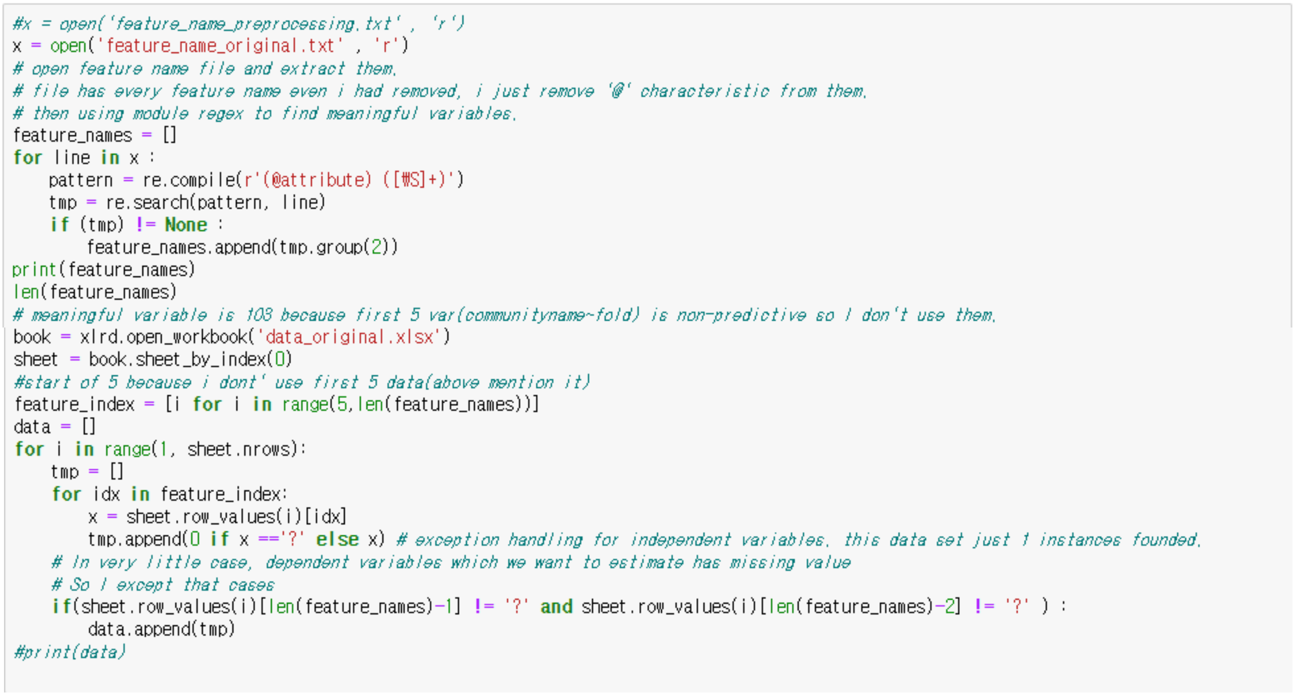
After test, in figure 9, we can see all three model's accuracy improve further. Particularly in linear model, adjusted R2 score changes 0.58261 to 0.61630 and gap between Train data and Test data is get lower because of removing unrelated/duplicated data, and making new meaningful features. That is also the same reason why all three model has similar accuracy. But in this process, I suppose Ridge's lambda as 0.01 and Lasso's lambda as 0.00001. It is very low and it seems like almost same as Linear model. Therefore I think if we properly select features and make significant things then performance will be similar in all of them.

Model type		Linear		Ridge		Lasso	
case		Train	Test	Train	Test	Train	Test
Prior (step1)	R2 score	0.6348	0.61625	0.62775	0.61539	0.62050	0.62218
	Adj R2	0.61626	0.59673	0.60881	0.59582	0.60121	0.60297
After (step2)	R2 score	0.63214	0.63265	0.62533	0.63339	0.61916	0.63609
	Adj R2	0.61577	0.61630	0.60865	0.61708	0.60221	0.61990

Figure 9. three model's accuracy after step2.

IV. Source Code

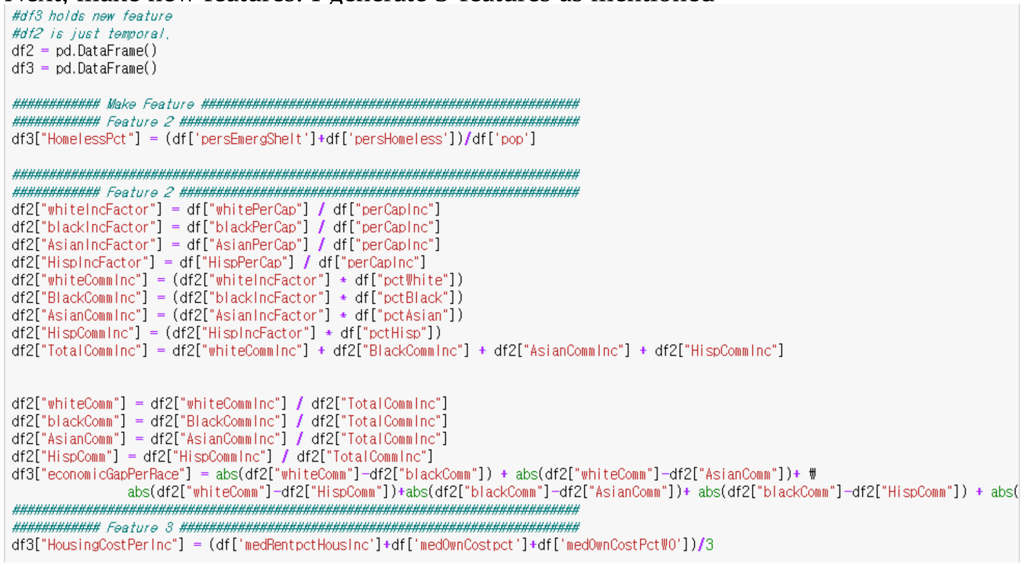
First, I extract data from original data file. As I mentioned in data set part, already remove unnecessary data in file



```python
#x = open('feature_name_preprocessing.txt' , 'r')
x = open('feature_name_original.txt' , 'r')
# open feature name file and extract them.
# file has every feature name even i had removed, i just remove '@' characteristic from them.
# then using module regex to find meaningful variables.
feature_names = []
for line in x :
    pattern = re.compile(r'(@attribute) ([₩S]+)')
    tmp = re.search(pattern, line)
    if (tmp) != None :
        feature_names.append(tmp.group(2))
print(feature_names)
len(feature_names)
# meaningful variable is 103 because first 5 var(communityname~fold) is non-predictive so I don't use them.
book = xlrd.open_workbook('data_original.xlsx')
sheet = book.sheet_by_index(0)
#start of 5 because i dont' use first 5 data(above mention it)
feature_index = [i for i in range(5,len(feature_names))]
data = []
for i in range(1, sheet.nrows):
    tmp = []
    for idx in feature_index:
        x = sheet.row_values(i)[idx]
        tmp.append(0 if x =='?' else x) # exception handling for independent variables. this data set just 1 instances founded.
    # In very little case, dependent variables which we want to estimate has missing value
    # So I except that cases
    if(sheet.row_values(i)[len(feature_names)-1] != '?' and sheet.row_values(i)[len(feature_names)-2] != '?' ) :
        data.append(tmp)
#print(data)
```

Figure 10. from FeatureEngineering.ipynb

Next, make new features. I generate 3-features as mentioned



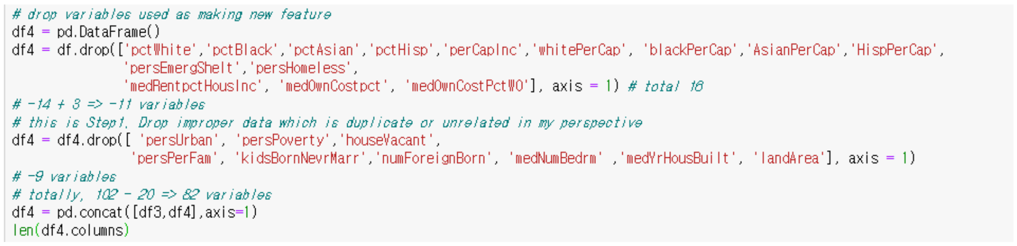
```python
#df3 holds new feature
#df2 is just temporal.
df2 = pd.DataFrame()
df3 = pd.DataFrame()

############ Make Feature ##########################################################
############ Feature 2 #############################################################
df3["HomelessPct"] = (df['persEmergShelt']+df['persHomeless'])/df['pop']

###################################################################################
############ Feature 2 #############################################################
df2["whiteIncFactor"] = df["whitePerCap"] / df["perCapInc"]
df2["blackIncFactor"] = df["blackPerCap"] / df["perCapInc"]
df2["AsianIncFactor"] = df["AsianPerCap"] / df["perCapInc"]
df2["HispIncFactor"] = df["HispPerCap"] / df["perCapInc"]
df2["whiteCommInc"] = (df2["whiteIncFactor"] * df["pctWhite"])
df2["BlackCommInc"] = (df2["blackIncFactor"] * df["pctBlack"])
df2["AsianCommInc"] = (df2["AsianIncFactor"] * df["pctAsian"])
df2["HispCommInc"] = (df2["HispIncFactor"] * df["pctHisp"])
df2["TotalCommInc"] = df2["whiteCommInc"] + df2["BlackCommInc"] + df2["AsianCommInc"] + df2["HispCommInc"]


df2["whiteComm"] = df2["whiteCommInc"] / df2["TotalCommInc"]
df2["blackComm"] = df2["BlackCommInc"] / df2["TotalCommInc"]
df2["AsianComm"] = df2["AsianCommInc"] / df2["TotalCommInc"]
df2["HispComm"] = df2["HispCommInc"] / df2["TotalCommInc"]
df3["economicGapPerRace"] = abs(df2["whiteComm"]-df2["blackComm"]) + abs(df2["whiteComm"]-df2["AsianComm"])+ ₩
                abs(df2["whiteComm"]-df2["HispComm"])+abs(df2["blackComm"]-df2["AsianComm"])+ abs(df2["blackComm"]-df2["HispComm"]) + abs(
###################################################################################
############ Feature 3 #############################################################
df3["HousingCostPerInc"] = (df['medRentpctHousInc']+df['medOwnCostpct']+df['medOwnCostPctWO'])/3
```

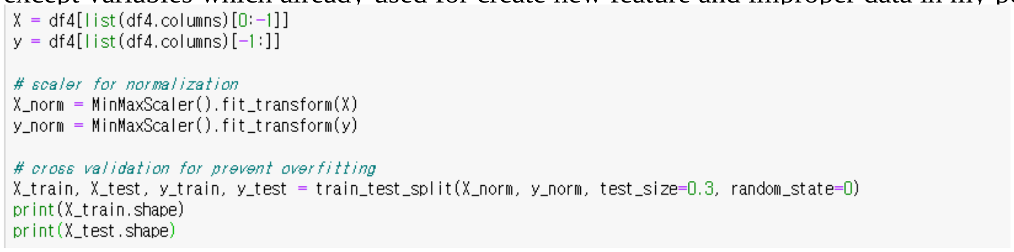
Figure 11. from FeatureEngineering.ipynb



```python
# drop variables used as making new feature
df4 = pd.DataFrame()
df4 = df.drop(['pctWhite','pctBlack','pctAsian','pctHisp','perCapInc','whitePerCap', 'blackPerCap','AsianPerCap','HispPerCap',
              'persEmergShelt','persHomeless',
              'medRentpctHousInc', 'medOwnCostpct', 'medOwnCostPctWO'], axis = 1) # total 16
# -14 + 3 => -11 variables
# this is Step1. Drop improper data which is duplicate or unrelated in my perspective
df4 = df4.drop([ 'persUrban', 'persPoverty','houseVacant',
               'persPerFam', 'kidsBornNevrMarr','numForeignBorn', 'medNumBedrm' ,'medYrHousBuilt', 'landArea'], axis = 1)
# -9 variables
# totally, 102 - 20 => 82 variables
df4 = pd.concat([df3,df4],axis=1)
len(df4.columns)
```

Figure 12. from FeatureEngineering.ipynb

except variables which already used for create new feature and improper data in my perspective

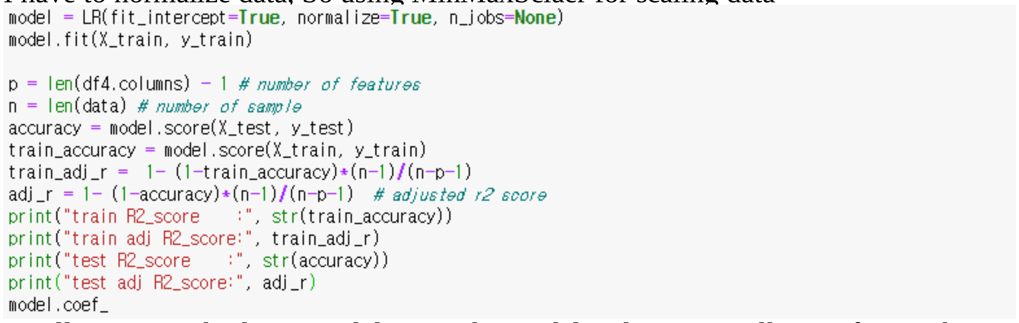


```python
X = df4[list(df4.columns)[0:-1]]
y = df4[list(df4.columns)[-1:]]

# scaler for normalization
X_norm = MinMaxScaler().fit_transform(X)
y_norm = MinMaxScaler().fit_transform(y)

# cross validation for prevent overfitting
X_train, X_test, y_train, y_test = train_test_split(X_norm, y_norm, test_size=0.3, random_state=0)
print(X_train.shape)
print(X_test.shape)
```

I have to normailze data, So using MinMaxSclaer for scaling data



```python
model = LR(fit_intercept=True, normalize=True, n_jobs=None)
model.fit(X_train, y_train)

p = len(df4.columns) - 1 # number of features
n = len(data) # number of sample
accuracy = model.score(X_test, y_test)
train_accuracy = model.score(X_train, y_train)
train_adj_r =  1- (1-train_accuracy)*(n-1)/(n-p-1)
adj_r = 1- (1-accuracy)*(n-1)/(n-p-1)  # adjusted r2 score
print("train R2_score    :", str(train_accuracy))
print("train adj R2_score:", train_adj_r)
print("test R2_score    :", str(accuracy))
print("test adj R2_score:", adj_r)
model.coef_
```

Finally, using scikit learn module train the model and test. It is all same for another two models

V. Conclusion

There are lots of variables in data set. And they are almost related to crime rate, so it is difficult to make new innovative feature. But I try to make a new one based on communities charicteristic and create 3-features which would be effective in my opinion. And also they achieve a little bit improvement on regression model whether or not it fell short of my expectation. I hope if there are more raw data, then I can generate another creatvie features. I learn the hard way that Feature Engineering is not easy and takes huge effort.

VI. References

[1] Dua, D. and Graff, C. (2019). UCI Machine Learning Repository [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.

[2] Pedregosa, Fabian, et al. "Scikit-learn: Machine learning in Python." *the Journal of machine Learning research* 12 (2011): 2825-2830.

[3] Helland, Inge S. "On the interpretation and use of R2 in regression analysis." *Biometrics* (1987): 61-69.

[4] Hoerl, Arthur E., and Robert W. Kennard. "Ridge regression: Biased estimation for nonorthogonal problems." *Technometrics* 12.1 (1970): 55-67.

[5] Tibshirani, Robert. "Regression shrinkage and selection via the lasso." *Journal of the Royal Statistical Society: Series B (Methodological)* 58.1 (1996): 267-288.