# Project Final Paper:
## "Analyzing Crime Rates in America: Data Analytics Approach"

By

Bharadwaj Venkateswaran

For

Prof. Li Zeng, ISEN 613

# Table of Contents

# 1. Problem Statement

With the development of modern machine learning techniques and their availability across the globe, researchers are trying to apply it to a lot of fields. Crime Forecasting is one of those fields in which use of machine learning has recently developed but it still remains a largely untouched problem. While there are numerable factors that affect the crime rate in various locations, it does come down to a few critical factors that can be predicted with the help of data available from government agencies.

The purpose of this project is to convert the reactive approach of Law Enforcement agencies to a rather proactive approach based on the forecasting model developed using multiple linear regression or regression trees based on the dataset used in the project for various locations across the US.

# 2. Dataset

The dataset being used is an agglomeration of data points collected from multiple sources, bounded by a common timeline and are combined via programming. The given socio-economic data includes the 1990 Law Enforcement Management and Admin Stats (LEMAS) survey and crime data from 1995 FBI UCR.

The variables included in the dataset involve the community, such as the percent of the population considered urban, the median family income, involving law enforcement, such as per capita number of police officers, and percent of officers assigned to drug units. The crime attributes that could be predicted are the 8 crimes considered 'Index Crimes' by the FBI (Murders, Rape, Robbery, etc.), per capita (actually per 100,000 population) versions of each, and Per Capita Violent Crimes and Per Capita Nonviolent Crimes. The potential goal is to reduce the number of attributes from a total of 129 of which 125 are predictive attributes and 4 are non-predictive.

# 3. Approach

## 3.1 Backward Subset Selection

First, all the other potential output variables except *violentPerPop*, were removed from the data set. Then the dataset was divided into 70% training and 30% test data. The code is shown below:

```
#remove other predictor variables
crimeViolence<-crimeNonNA[,c(-120,-118:-103)]

set.seed(1)
#create the training data and test data. Training:Test ratio is 70:30
train<-sample(nrow(crimeViolence),(0.7)*nrow(crimeViolence))
crime.train<-crimeViolence[train,]
crime.test<-crimeViolence[-train,]
```

Then tried to perform subset selection method on violentPerPop (violent crime per population) after removing characteristic variables and response variables with target number of 20 variables.

```
> library(leaps)
>
> regfit.full = regsubsets(violentPerPop~.-State-communityname-murders-murdP
erPop-rapes-rapesPerPop-robberies-robbbPerPop-assaults-assaultPerPop-burglar
ies-burglPerPop-larcenies-larcPerPop-autoTheft-autoTheftPerPop-arsons-arsons
PerPop-violentPerPop-nonViolPerPop, data=crimeNonNA, nvmax=20, really.big=T)
Reordering variables and trying again:
```

However, as it was ‘big data’ with more than 100 attributes and 1000 observations, even after over 24-hours of waiting in workstation condition, the RStudio still showed running for the results. So, it was to use the simple backward subset selection method based on linear regression.

```
> model.back=step(lm(violentPerPop~.-State-communityname-murders-murdPerPop-rapes-rapesPerPop-robberies-
robbbPerPop-assaults-assaultPerPop-burglaries-burglPerPop-larcenies-larcPerPop-autoTheft-autoTheftPerPop
-arsons-arsonsPerPop-violentPerPop-nonViolPerPop, data=training), direction="backward")
Start:  AIC=15721.03
violentPerPop ~ (communityname + State + fold + pop + perHoush +
    pctBlack + pctWhite + pctAsian + pctHisp + `pct12-21` + `pct12-29` +
    `pct16-24` + pct65up + persUrban + pctUrban + medIncome +
    pctWwage + pctWfarm + pctWdiv + pctWsocsec + pctPubAsst +
    pctRetire + medFamIncome + perCapInc + whitePerCap + blackPerCap +
    NAperCap + asianPerCap + otherPerCap + hispPerCap + persPoverty +
    pctPoverty + pctLowEdu + pctNotHSgrad + pctCollGrad + pctUnemploy +
    pctEmploy + pctEmployMfg + pctEmployProfServ + pctOccupManu +
    pctOccupMgmt + pctMaleDivorc + pctMaleNevMar + pctFemDivorc +
    pctAllDivorc + persPerFam + pct2Par + pctKids2Par + `pctKids-4w2Par` +
    `pct12-17w2Par` + `pctWorkMom-6` + `pctWorkMom-18` + kidsBornNevrMarr +
    pctKidsBornNevrMarr + numForeignBorn + `pctFgnImmig-3` +
    `pctFgnImmig-5` + `pctFgnImmig-8` + `pctFgnImmig-10` + `pctImmig-3` +
    `pctImmig-5` + `pctImmig-8` + `pctImmig-10` + pctSpeakOnlyEng +
    pctNotSpeakEng + pctLargHousFam + pctLargHous + persPerOccupHous +
    persPerOwnOccup + persPerRenterOccup + pctPersOwnOccup +
    pctPopDenseHous + pctSmallHousUnits + medNumBedrm + houseVacant +
    pctHousOccup + pctHousOwnerOccup + pctVacantBoarded + pctVacant6up +
    medYrHousBuilt + pctHousWOphone + pctHousWOplumb + ownHousLowQ +
    ownHousMed + ownHousUperQ + ownHousQrange + rentLowQ + rentMed +
    rentUpperQ + rentQrange + medGrossRent + medRentpctHousInc +
    medOwnCostpct + medOwnCostPctWO + persEmergShelt + persHomeless +
    pctForeignBorn + pctBornStateResid + `pctSameHouse-5` + `pctSameCounty-5` +
    `pctSameState-5` + landArea + popDensity + pctUsePubTrans +
    pctOfficDrugUnit + murders + murdPerPop + rapes + rapesPerPop +
    robberies + robbbPerPop + assaults + assaultPerPop + burglaries +
    burglPerPop + larcenies + larcPerPop + autoTheft + autoTheftPerPop +
    arsons + arsonsPerPop + nonViolPerPop) - State - communityname -
    murders - murdPerPop - rapes - rapesPerPop - robberies -
    robbbPerPop - assaults - assaultPerPop - burglaries - burglPerPop -
    larcenies - larcPerPop - autoTheft - autoTheftPerPop - arsons -
    arsonsPerPop - violentPerPop - nonViolPerPop
```

After running the code, system tested several selections and it came down to 40 variables.

```
Step:  AIC=15628.35
violentPerPop ~ perHoush + pctBlack + persUrban + pctUrban +
    medIncome + pctWwage + pctWfarm + pctRetire + medFamIncome +
    perCapInc + asianPerCap + otherPerCap + pctPoverty + pctLowEdu +
    pctEmploy + pctEmployMfg + pctMaleDivorc + pctFemDivorc +
    pctAllDivorc + pctKids2Par + `pctWorkMom-18` + pctKidsBornNevrMarr +
    numForeignBorn + `pctImmig-5` + pctLargHous + persPerOccupHous +
    persPerRenterOccup + pctPersOwnOccup + pctPopDenseHous +
    houseVacant + pctHousOwnerOccup + pctVacantBoarded + pctVacant6up +
    rentLowQ + rentMed + medOwnCostpct + medOwnCostPctWO + persEmergShelt +
    pctForeignBorn + pctOfficDrugUnit

                      Df Sum of Sq       RSS   AIC
<none>                             158604072 15628
- `pctImmig-5`         1    243502 158847574 15628
- perHoush             1    284132 158888205 15629
- medOwnCostpct        1    303141 158907213 15629
- pctEmploy            1    326401 158930473 15629
- perCapInc            1    329389 158933462 15629
- persEmergShelt       1    383028 158987100 15630
- pctLargHous          1    405982 159010054 15630
- pctWfarm             1    446382 159050454 15630
- pctForeignBorn       1    585471 159189543 15631
- pctVacantBoarded     1    607580 159211652 15631
- medFamIncome         1    645264 159249336 15632
- pctLowEdu            1    649270 159253342 15632
- pctVacant6up         1    679787 159283859 15632
- pctOfficDrugUnit     1    722195 159326267 15632
- pctPopDenseHous      1    798125 159402197 15633
- otherPerCap          1    890215 159494287 15634
- asianPerCap          1    905789 159509861 15634
- pctPoverty           1    954697 159558769 15634
- pctFemDivorc         1   1000881 159604954 15635
- medIncome            1   1076085 159680157 15635
- medOwnCostPctWO      1   1101295 159705367 15636
- persPerRenterOccup   1   1104748 159708820 15636
- pctAllDivorc         1   1246176 159850248 15637
- rentLowQ             1   1267800 159871872 15637
- pctPersOwnOccup      1   1329812 159933884 15638
- pctUrban             1   1391914 159995986 15638
- persPerOccupHous     1   1407123 160011195 15638
- numForeignBorn       1   1473115 160077188 15639
- rentMed              1   1552309 160156381 15639
- pctHousOwnerOccup    1   1561018 160165090 15639
- pctWwage             1   1638371 160242443 15640
- pctEmployMfg         1   1733253 160337325 15641
- `pctWorkMom-18`      1   1838951 160443023 15642
- pctMaleDivorc        1   1882624 160486696 15642
- pctKids2Par          1   2155720 160759792 15644
- pctKidsBornNevrMarr  1   2201689 160805762 15645
- pctRetire            1   2236748 160840820 15645
- persUrban            1   2460671 161064743 15647
- houseVacant          1   3491974 162096046 15655
- pctBlack             1   4607057 163211130 15664
```

And the summary of linear regression model it obtained through backward subset selection method is provided on the next page.

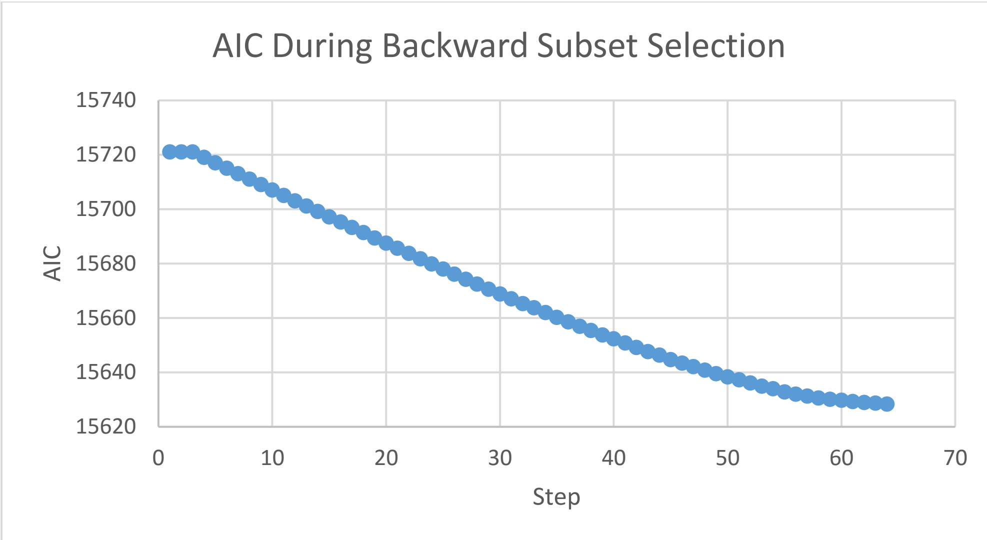
```
> summary(model.back)

Call:
lm(formula = violentPerPop ~ perHoush + pctBlack + persUrban +
    pctUrban + medIncome + pctWwage + pctWfarm + pctRetire +
    medFamIncome + perCapInc + asianPerCap + otherPerCap + pctPoverty +
    pctLowEdu + pctEmploy + pctEmployMfg + pctMaleDivorc + pctFemDivorc +
    pctAllDivorc + pctKids2Par + `pctWorkMom-18` + pctKidsBornNevrMarr +
    numForeignBorn + `pctImmig-5` + pctLargHous + persPerOccupHous +
    persPerRenterOccup + pctPersOwnOccup + pctPopDenseHous +
    houseVacant + pctHousOwnerOccup + pctVacantBoarded + pctVacant6up +
    rentLowQ + rentMed + medOwnCostpct + medOwnCostPctWO + persEmergShelt +
    pctForeignBorn + pctOfficDrugUnit, data = training)

Residuals:
     Min       1Q   Median       3Q      Max 
-1456.77  -177.02   -26.76   124.42  2341.13 

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)    
(Intercept)          3.432e+03  5.767e+02   5.951 3.43e-09 ***
perHoush            -1.545e+02  1.017e+02  -1.520 0.128856    
pctBlack             9.344e+00  1.527e+00   6.119 1.25e-09 ***
persUrban           -1.527e-03  3.415e-04  -4.472 8.43e-06 ***
pctUrban             1.028e+00  3.058e-01   3.363 0.000793 ***
medIncome           -2.159e-02  7.302e-03  -2.957 0.003160 ** 
pctWwage            -1.653e+01  4.531e+00  -3.649 0.000274 ***
pctWfarm             3.417e+01  1.794e+01   1.905 0.057044 .  
pctRetire           -1.616e+01  3.790e+00  -4.264 2.16e-05 ***
medFamIncome         1.532e-02  6.688e-03   2.290 0.022181 *  
perCapInc           -1.285e-02  7.853e-03  -1.636 0.102052    
asianPerCap          3.239e-03  1.194e-03   2.713 0.006752 ** 
otherPerCap          3.473e-03  1.291e-03   2.690 0.007242 ** 
pctPoverty          -1.004e+01  3.604e+00  -2.785 0.005423 ** 
pctLowEdu           -7.398e+00  3.221e+00  -2.297 0.021772 *  
pctEmploy            7.783e+00  4.779e+00   1.629 0.103618    
pctEmployMfg        -5.627e+00  1.499e+00  -3.753 0.000182 ***
pctMaleDivorc        2.482e+02  6.346e+01   3.912 9.65e-05 ***
pctFemDivorc         1.905e+02  6.678e+01   2.852 0.004413 ** 
pctAllDivorc        -4.112e+02  1.292e+02  -3.182 0.001495 ** 
pctKids2Par         -1.721e+01  4.112e+00  -4.186 3.04e-05 ***
`pctWorkMom-18`     -9.548e+00  2.470e+00  -3.866 0.000116 ***
pctKidsBornNevrMarr  4.259e+01  1.007e+01   4.230 2.50e-05 ***
numForeignBorn       2.170e-03  6.270e-04   3.460 0.000558 ***
`pctImmig-5`        -1.799e+01  1.279e+01  -1.407 0.159739    
pctLargHous         -1.788e+01  9.846e+00  -1.816 0.069534 .  
persPerOccupHous     7.068e+02  2.090e+02   3.382 0.000742 ***
persPerRenterOccup  -3.568e+02  1.191e+02  -2.996 0.002784 ** 
pctPersOwnOccup     -3.266e+01  9.936e+00  -3.287 0.001038 ** 
pctPopDenseHous      1.549e+01  6.082e+00   2.547 0.010985 *  
houseVacant          3.294e-02  6.183e-03   5.327 1.18e-07 ***
pctHousOwnerOccup    3.332e+01  9.355e+00   3.562 0.000382 ***
pctVacantBoarded     8.230e+00  3.704e+00   2.222 0.026447 *  
pctVacant6up        -2.281e+00  9.703e-01  -2.350 0.018899 *  
rentLowQ            -9.779e-01  3.047e-01  -3.210 0.001360 ** 
rentMed              1.097e+00  3.089e-01   3.552 0.000396 ***
medOwnCostpct       -8.507e+00  5.420e+00  -1.570 0.116751    
medOwnCostPctWO     -2.683e+01  8.970e+00  -2.992 0.002827 ** 
persEmergShelt       7.340e-02  4.160e-02   1.764 0.077910 .  
pctForeignBorn       9.151e+00  4.195e+00   2.181 0.029339 *  
pctOfficDrugUnit     9.661e+00  3.988e+00   2.423 0.015543 *  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 350.8 on 1289 degrees of freedom
Multiple R-squared:  0.6883,	Adjusted R-squared:  0.6787 
F-statistic: 71.17 on 40 and 1289 DF,  p-value: < 2.2e-16
```




It can be seen from graph above that AIC decreased continuously each step during stepwise backward selection. Through cross-validation approach a RMSE value of 122,105.7 was obtained.

```
> vali = vector()
> for (i in 1:10000){
+     set.seed(i)
+     indx = sample(nrow(crimeNonNA), training_number, replace=FALSE)
+     training = crimeNonNA[indx,]
+     test = crimeNonNA[-indx,]
+     vali = c(vali, mean((test$violentPerPop - predict(model.back, test))^2))}
> mean(vali)
[1] 122105.7
```

However, it would not be a good subset selection method as this selected variables' value is limited to when it is only used on building a linear regression model.

## 3.2 Lasso Regression

The output variable in focus is violentPerPop, which is the total number of violent crimes per 100K population. Ridge regression does have one obvious disadvantage. Unlike forward and backward stepwise selection, which will generally select models that involve just a subset of the variables, ridge regression will include all p predictors in the final model. Lasso regression is a great model that is similar to ridge regression, faster than the subset selection models, and in addition performs variable selection.

### 3.2.1 Methodology

After obtaining the training and test data, a model matrix *x* and vector *y* was created, since lasso regression expected the *x* to be a matrix and *y* to be a vector. These matrices contain the predictors and output variables respectively. The first variable in *x* is deleted because it consists of the intercept term. The code is as below:

```
#create a model matrix X and Y containing the predictors and outputs
x<-model.matrix (violentPerPop~.,crimeViolence)[,-1]
y<-crimeViolence$violentPerPop
```

To perform the lasso regression, **glmnet()** function from glmnet package is used.

```
#create a grid of values to iterate the lambda
grid <- 10^ seq (10,-2, length =100)
```

```
#run lasso regression using alpha = 1 and lambda = grid
lasso.mod =glmnet (x[train ,],y[train],alpha =1, lambda =grid)
```

The alpha value indicates the type regression: ridge (0) and lasso (1). Initially, a grid of values was created to iterate lambda (λ). In this case, *grid* is a variable consisting of sequence of power of 10. In the **glmnet()** function, the value of λ is assigned the variable *grid*.

```
> dim(coef(lasso.mod))
[1] 103 100
```

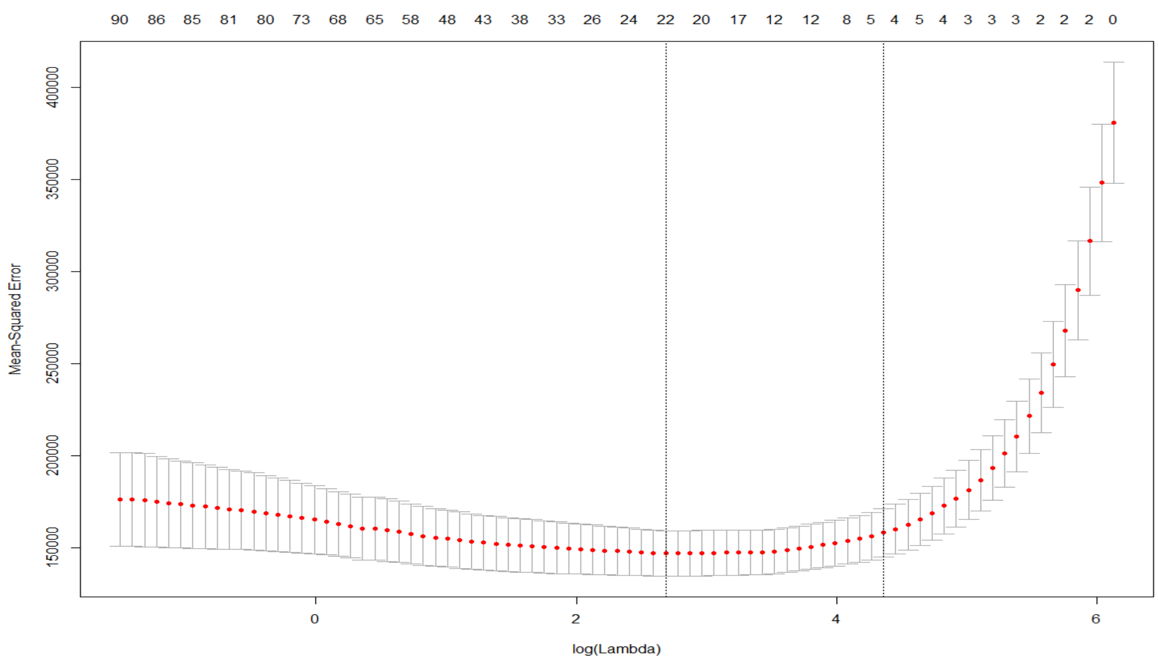
The resultant coefficients of *lasso.mod* is a 103x100 matrix. This means it has 103 rows with the number of variables including an intercept and 100 columns (one for each value of lambda). However, finding the best lambda is a cumbersome task, which leads to using cross validation has to be used to find the best lambda (λ). Function **cv.glmnet()** was used to run a cross validation. The code is as below:

```
#running cross validation using cv.glmnet
cv.crime.out<-cv.glmnet (x[train ,],y[train],alpha =1)
```

Once the cross validation ran, the output was plotted. The code is shown below:

```
#plot the cross validation output
plot(cv.crime.out)
```

The plot below gives the best value of λ that produces gives the least mean squared error. In this case, the best λ is the one corresponding to 22 degrees of freedom, which implies the variables have been reduced from 103 to 23 variables.




The best λ can be found out using the below code:

```
#get the best lambda
bestlam<-cv.crime.out$lambda.min
```

| Values | |
|---|---|
| bestlam | 14.6556082687358 |

The best λ value is approximately 14.6556. This λ value is used to predict the RMSE for the test data, which would correspond to the lowest amongst all the λ values. The test RMSE is around 127741.7, which is comparable to the test RMSE from the best subset selection. The code and output is shown below:

```
#using the bestlam predict the test MSE
lasso.pred<-predict (lasso.mod ,s=bestlam ,newx=x[-train,])
mean((lasso.pred - y[-train])^2)
```

```
> mean((lasso.pred - y[-train])^2)
[1] 127741.7
```

The model coefficients were evaluated using glmnet() function on the whole data with alpha = 1 and lambda = bestlam (14.6566). The code is shown below:

```
#predicting the coefficients
out=glmnet (x,y,alpha =1, lambda =bestlam)
lasso.coef=predict (out ,type ="coefficients",s=bestlam)[1:103,]
lasso.coef[lasso.coef!=0]
```

The *lasso.coef* contains the coefficients of all the variables in the model.

```
> lasso.coef
       (Intercept)                pop           perHoush           pctBlack           pctWhite           pctAsian            pctHisp
      1.843002e+03       0.000000e+00       0.000000e+00       2.666860e+00      -4.166381e+00       0.000000e+00       0.000000e+00
         `pct12-21`         `pct12-29`         `pct16-24`            pct65up           persUrban           pctUrban          medIncome
      0.000000e+00      -2.101424e+00       0.000000e+00       0.000000e+00       0.000000e+00       5.043544e-01       0.000000e+00
          pctWwage           pctWfarm            pctWdiv          pctWsocsec          pctPubAsst          pctRetire        medFamIncome
      0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00
        perCapInc        whitePerCap        blackPerCap            NAperCap         asianPerCap         otherPerCap          hispPerCap
      0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00       2.733715e-04       1.100905e-03       0.000000e+00
      persPoverty         pctPoverty          pctLowEdu         pctNotHSgrad         pctCollGrad         pctUnemploy          pctEmploy
      0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00
      pctEmployMfg  pctEmployProfServ        pctOccupManu        pctOccupMgmt      pctMaleDivorc       pctMaleNevMar       pctFemDivorc
     -1.791161e+00       0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00
      pctAllDivorc         persPerFam            pct2Par         pctKids2Par      `pctKids-4w2Par`     `pct12-17w2Par`       `pctWorkMom-6`
      0.000000e+00       0.000000e+00       0.000000e+00      -9.636614e+00       0.000000e+00       0.000000e+00       0.000000e+00
    `pctWorkMom-18`   kidsBornNevrMarr  pctKidsBornNevrMarr     numForeignBorn      `pctFgnImmig-3`      `pctFgnImmig-5`      `pctFgnImmig-8`
     -3.151782e+00       0.000000e+00       5.514086e+01       0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00
   `pctFgnImmig-10`        `pctImmig-3`        `pctImmig-5`        `pctImmig-8`       `pctImmig-10`      pctSpeakOnlyEng       pctNotSpeakEng
      0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00
     pctLargHousFam        pctLargHous     persPerOccupHous     persPerOwnOccup   persPerRenterOccup      pctPersOwnOccup      pctPopDenseHous
      0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00       8.465181e+00
   pctSmallHousUnits         medNumBedrm         houseVacant         pctHousOccup    pctHousOwnerOccup     pctVacantBoarded        pctVacant6up
      1.066477e-01       0.000000e+00       5.897551e-03      -4.405190e+00       0.000000e+00       7.643565e+00       0.000000e+00
     medYrHousBuilt      pctHousWOphone      pctHousWOplumb          ownHousLowQ          ownHousMed         ownHousUperQ        ownHousQrange
      0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00
         rentLowQ            rentMed         rentUpperQ          rentQrange        medGrossRent    medRentpctHousInc        medOwnCostpct
      0.000000e+00       0.000000e+00       0.000000e+00       1.488381e-01       0.000000e+00       4.717190e-01       0.000000e+00
    medOwnCostPctWO      persEmergShelt        persHomeless      pctForeignBorn     pctBornStateResid     `pctSameHouse-5`    `pctSameCounty-5`
     -1.072224e+01       0.000000e+00       0.000000e+00       7.427340e-01      -6.516433e-03       0.000000e+00       0.000000e+00
    `pctSameState-5`           landArea          popDensity       pctUsePubTrans      pctOfficDrugUnit
      0.000000e+00       0.000000e+00       0.000000e+00       0.000000e+00       7.598575e+00
```

This included the coefficients with value equal to zero. However, as mentioned at the beginning of this section, lasso regression's advantage over the ridge regression is its ability to perform variable selection and at the same time provide an accurate model with respect to the bias-variance trade-off. We can obtain the non-zero variables by sub-setting the *lasso.coef* variable. The final set of variables is shown below.

```
> lasso.coef[lasso.coef!=0]
       (Intercept)           pctBlack           pctWhite          `pct12-29`            pctUrban         asianPerCap         otherPerCap
      1.843002e+03       2.666860e+00      -4.166381e+00      -2.101424e+00        5.043544e-01        2.733715e-04        1.100905e-03
      pctEmployMfg      pctMaleDivorc         pctKids2Par     `pctWorkMom-18` pctKidsBornNevrMarr     pctPopDenseHous   pctSmallHousUnits
     -1.791161e+00       2.750502e+01      -9.636614e+00      -3.151782e+00        5.514086e+01        8.465181e+00        1.066477e-01
       houseVacant       pctHousOccup    pctVacantBoarded          rentQrange   medRentpctHousInc     medOwnCostPctWO      pctForeignBorn
      5.897551e-03      -4.405190e+00       7.643565e+00        1.488381e-01        4.717190e-01       -1.072224e+01        7.427340e-01
   pctBornStateResid   pctOfficDrugUnit
     -6.516433e-03       7.598575e+00
```

## 4. Final Model Selection for the crime dataset

For the final model, we have to edit the data to contain only the output variable and the predictor variables from the lasso regression since it is more efficient than the subset selection method, both backward and forward. In addition, ridge regression does not aid in variable selection although it is an accurate model.

```
#creating a vector with the new variables whose coefficients are not 0
newVariables<-names(lasso.coef[lasso.coef!=0][-1])
newVariables<-c("pctBlack","pctWhite","pct12-29","pctUrban","asianPerCap","otherPerCap","pctEmployMfg",
                "pctMaleDivorc","pctKids2Par","pctWorkMom-18","pctKidsBornNevrMarr","pctPopDenseHous",
                "pctSmallHousUnits","houseVacant","pctHousOccup","pctVacantBoarded","rentQrange","medRentpctHousInc",
                "medOwnCostPctWO","pctForeignBorn","pctBornStateResid","pctOfficDrugUnit","violentPerPop")
crimeViolenceFinalData<-crimeViolence[newVariables]
```

```
> str(crimeViolenceFinalData)
'data.frame':	1901 obs. of  23 variables:
 $ pctBlack           : num  1.37 0.8 0.74 2.51 1.6 ...
 $ pctWhite           : num  91.8 95.6 94.3 95.7 96.6 ...
 $ pct12-29           : num  21.4 21.3 25.9 32.9 27.4 ...
 $ pctUrban           : num  100 100 100 100 100 100 100 100 100 100 ...
 $ asianPerCap        : int  27101 20074 15528 10753 12639 8802 8011 5770 9107 9546 ...
 $ otherPerCap        : int  5115 5250 5954 7192 21852 7428 5332 7320 5267 8330 ...
 $ pctEmployMfg       : num  14.6 12.3 15.9 14.3 14 ...
 $ pctMaleDivorc      : num  3.67 4.23 10.1 11.4 5.97 ...
 $ pctKids2Par        : num  90.2 85.3 78.8 69.8 79.8 ...
 $ pctWorkMom-18      : num  58.9 62.4 74.2 70.5 72.8 ...
 $ pctKidsBornNevrMarr: num  0.36 0.24 0.88 1.58 1.18 4.66 1.64 4.71 2.47 7.44 ...
 $ pctPopDenseHous    : num  0.39 1.01 2.03 2.11 1.47 ...
 $ pctSmallHousUnits  : num  11.1 23.6 47.5 53.2 47.4 ...
 $ houseVacant        : int  64 240 544 5119 566 2051 1562 5606 1807 714 ...
 $ pctHousOccup       : num  98.4 97.2 95.7 91.8 95.1 ...
 $ pctVacantBoarded   : num  3.12 0 0.92 2.09 1.41 6.39 0.45 5.64 2.77 3.78 ...
 $ rentQrange         : int  316 205 150 134 361 139 146 177 142 190 ...
 $ medRentpctHousInc  : num  23.8 27.6 24.1 26.4 24.4 26.3 25.2 29.6 23.8 33 ...
 $ medOwnCostPctWO    : num  14 12.5 11.6 11.7 12.5 12.2 12.8 13 12.9 11.8 ...
 $ pctForeignBorn     : num  10.66 8.3 5 1.49 9.19 ...
 $ pctBornStateResid  : num  53.7 77.2 44.8 64.3 77.3 ...
 $ pctOfficDrugUnit   : num  0 0 0 0 0 0 0 6.57 0 0 ...
 $ violentPerPop      : num  41 128 219 443 227 ...
```

This dataset was used to perform linear regression and regression tree.

### 4.1 Linear Regression

The model was run against all the predictor variables on the training dataset. The code is shown below:

```
#linear regression model on the complete set of variables, performed on the training data
lm.fit<-lm(violentPerPop~.,data=crimeViolenceFinalData[train,])
summary(lm.fit)
```

The summary gives the various summaries of the result of the linear regression.

```
> summary(lm.fit)

Call:
lm(formula = violentPerPop ~ ., data = crimeViolenceFinalData[train,
    ])

Residuals:
     Min       1Q   Median       3Q      Max
-1507.29  -190.93   -35.36   130.19  2455.02

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)          1.687e+03  4.468e+02   3.776 0.000166 ***
pctBlack             8.192e+00  2.378e+00   3.445 0.000590 ***
pctWhite             8.289e-01  2.118e+00   0.391 0.695656
`pct12-29`          -5.596e+00  2.159e+00  -2.592 0.009654 **
pctUrban             5.385e-01  2.581e-01   2.086 0.037143 *
asianPerCap          2.137e-03  1.177e-03   1.815 0.069731 .
otherPerCap          3.307e-03  1.293e-03   2.557 0.010656 *
pctEmployMfg        -4.245e+00  1.487e+00  -2.854 0.004379 **
pctMaleDivorc        3.653e+01  6.775e+00   5.392 8.26e-08 ***
pctKids2Par         -9.023e+00  2.836e+00  -3.182 0.001498 **
`pctWorkMom-18`     -4.770e+00  1.835e+00  -2.599 0.009444 **
pctKidsBornNevrMarr  5.614e+01  8.873e+00   6.327 3.43e-10 ***
pctPopDenseHous      1.315e+01  3.803e+00   3.457 0.000565 ***
pctSmallHousUnits    4.519e-01  1.343e+00   0.336 0.736631
houseVacant          5.982e-03  1.591e-03   3.761 0.000177 ***
pctHousOccup        -6.612e+00  2.586e+00  -2.557 0.010682 *
pctVacantBoarded     6.732e+00  3.466e+00   1.942 0.052295 .
rentQrange           2.923e-01  1.555e-01   1.880 0.060329 .
medRentpctHousInc    6.134e+00  4.451e+00   1.378 0.168364
medOwnCostPctWO     -3.313e+01  8.217e+00  -4.033 5.83e-05 ***
pctForeignBorn       2.771e+00  2.734e+00   1.013 0.311016
pctBornStateResid    4.128e-01  8.239e-01   0.501 0.616411
pctOfficDrugUnit     8.519e+00  4.040e+00   2.109 0.035144 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 361.5 on 1307 degrees of freedom
Multiple R-squared:  0.6644,    Adjusted R-squared:  0.6587
F-statistic: 117.6 on 22 and 1307 DF,  p-value: < 2.2e-16

> lm.pred<-predict(lm.fit,crimeViolenceFinalData[-train,])
> mean((lm.pred-crimeViolenceFinalData[-train,]$violentPerPop)^2)
[1] 127357
```

The above output gives significance values attached to each of the coefficients of the variables. The model's F-statistic is < 2.2e-16 which suggests the model is significant. The multiple R-square is 0.6644 and the adjusted R-square is 0.6587. The RMSE of the base model is 127,357. The endeavor of this analysis was to find a model that takes into consideration bias variance trade-off, multi-collinearity and improving adjusted R-square.

The first step was to check for correlation between the variables and variance inflation factor (VIF). This is useful in checking for multi-collinearity present in the model. The **cor()** function was used to create a correlation matrix between the variables of the dataset and **vif()** function was used to calculate the VIF values for each variable.

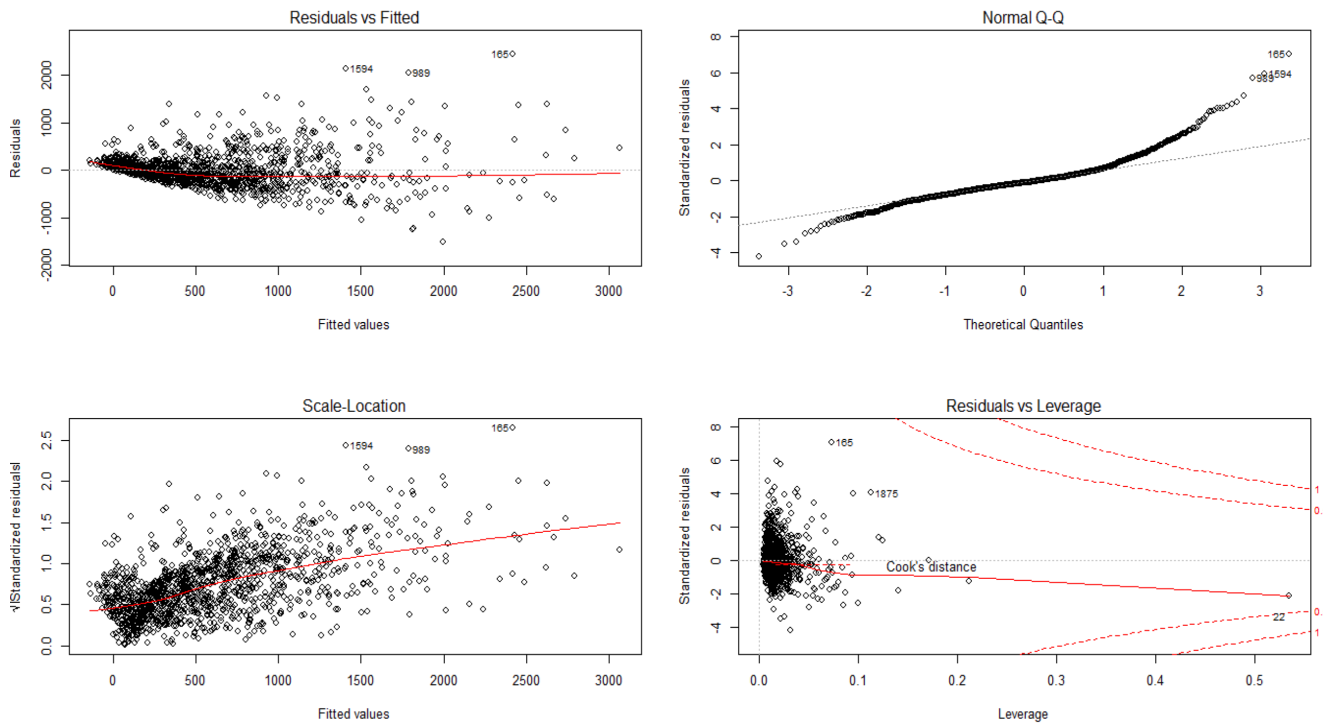
```
#correlation matrix
cor(crimeViolenceFinalData)
```

```
                      pctBlack    pctWhite    pct12-29     pctUrban    asianPerCap   otherPerCap  pctEmployMfg pctMaleDivorc pctKids2Par
pctBlack            1.000000000 -0.79612281  0.12170807 -0.001108497 -0.1079634624 -0.089816523 -0.0167747211    0.39851223 -0.73022541
pctWhite           -0.796122809  1.00000000 -0.22060335 -0.048983008  0.1421262378  0.101630404  0.0330027766   -0.33658011  0.69943035
pct12-29            0.121708069 -0.22060335  1.00000000 -0.110376102 -0.2867100401 -0.132139504 -0.1161581698   -0.05865820 -0.21258113
pctUrban           -0.001108497 -0.04898301 -0.11037610  1.000000000  0.0826915977  0.136647402 -0.1027644712   -0.06292618  0.10426066
asianPerCap        -0.107963462  0.14212624 -0.28671004  0.082691598  1.0000000000  0.150449248  0.0008427735   -0.21571294  0.27059974
otherPerCap        -0.089816523  0.10163040 -0.13213950  0.136647402  0.1504492476  1.000000000 -0.0398265231   -0.17813689  0.24801178
pctEmployMfg       -0.016774721  0.03300278 -0.11615817 -0.102764471  0.0008427735 -0.039826523  1.0000000000    0.04880237 -0.02425829
pctMaleDivorc       0.398512233 -0.33658011 -0.05865820 -0.062926182 -0.2157129432 -0.178136891  0.0488023678    1.00000000 -0.71154415
pctKids2Par        -0.730225407  0.69943035 -0.21258113  0.104260662  0.2705997407  0.248011782 -0.0242582854   -0.71154415  1.00000000
pctWorkMom-18       0.093261430  0.15319060  0.02467978 -0.063588039 -0.0982906320  0.009030479  0.0907257482    0.10590772  0.02553232
pctKidsBornNevrMarr 0.805987508 -0.79869868  0.24755725  0.012412963 -0.1949514656 -0.160074201  0.0413104084    0.47703650 -0.85907269
pctPopDenseHous     0.106760074 -0.59611933  0.25574512  0.046818216 -0.1556918591 -0.116378860  0.0449163617    0.10595398 -0.33138574
pctSmallHousUnits   0.230687625 -0.36829823  0.28453013 -0.025579618 -0.2484797130 -0.190387391 -0.0627126101    0.57164732 -0.63612379
houseVacant         0.178274385 -0.19405335  0.02866199  0.120718521 -0.0479523642 -0.040598164 -0.1176446047    0.18287795 -0.21789493
pctHousOccup       -0.176977377  0.10976775  0.05426459  0.143687623  0.0702500609  0.096097537  0.3017781816   -0.31802374  0.32512759
pctVacantBoarded    0.471390664 -0.44124639  0.06571658  0.018833126 -0.1246136282 -0.118070615 -0.0096422638    0.31807409 -0.51140007
rentQrange         -0.148904031  0.07771463 -0.16395706  0.226358768  0.2556612917  0.211353859 -0.1113090020   -0.39521112  0.33055177
medRentpctHousInc   0.184448871 -0.33177946  0.29995579 -0.023941261 -0.1310458147 -0.052407676 -0.2755652208    0.09422160 -0.36317165
medOwnCostPctWO     0.216758656 -0.05770524 -0.08660812 -0.019351974  0.0229028198  0.032855567  0.0365239984   -0.05822100 -0.12829139
pctForeignBorn     -0.092416524 -0.38803015  0.07031523  0.275071971  0.0469447689  0.065197471 -0.0296992810   -0.11492294 -0.03369781
pctBornStateResid   0.090748436  0.11677719 -0.04831816 -0.226344705 -0.0523206557 -0.100206038  0.3418906347   -0.05125950 -0.06276262
pctOfficDrugUnit    0.252640272 -0.25849976  0.07947460  0.224584768 -0.0591209638 -0.022221617 -0.0759063865    0.22033626 -0.28928097
violentPerPop       0.625339003 -0.67719557  0.10993216  0.073954613 -0.1331356171 -0.100414879 -0.0576949244    0.51663043 -0.72815910
                   pctWorkMom-18 pctKidsBornNevrMarr pctPopDenseHous pctSmallHousUnits  houseVacant pctHousOccup pctVacantBoarded   rentQrange
pctBlack             0.093261430          0.80598751      0.10676007        0.23068762  0.178274385  -0.17697738      0.471390664 -0.148904031
pctWhite             0.153190599         -0.79869868     -0.59611933       -0.36829823 -0.194053349   0.10976775     -0.441246392  0.077714626
pct12-29             0.024679778          0.24755725      0.25574512        0.28453013  0.028661994   0.05426459      0.065716577 -0.163957056
pctUrban            -0.063588039          0.01241296      0.04681822       -0.02557962  0.120718521   0.14368762      0.018833126  0.226358768
asianPerCap         -0.098290632         -0.19495147     -0.15569186       -0.24847971 -0.047952364   0.07025006     -0.124613628  0.255661292
otherPerCap          0.009030479         -0.16007420     -0.11637886       -0.19038739 -0.040598164   0.09609754     -0.118070615  0.211353859
pctEmployMfg         0.090725748          0.04131041      0.04491636       -0.06271261 -0.117644605   0.30177818     -0.009642264 -0.111309002
pctMaleDivorc        0.105907721          0.47703650      0.10595398        0.57164732  0.182877953  -0.31802374      0.318074095 -0.395211121
pctKids2Par          0.025532317         -0.85907269     -0.33138574       -0.63612379 -0.217894932   0.32512759     -0.511400072  0.330551766
pctWorkMom-18        1.000000000         -0.07798889     -0.38638258       -0.02817173 -0.070989669   0.04356815     -0.175937801 -0.125645390
pctKidsBornNevrMarr -0.077988894          1.00000000      0.39601169        0.46515535  0.242113540  -0.17647692      0.555808175 -0.145358721
pctPopDenseHous     -0.386382580          0.39601169      1.00000000        0.42269478  0.091344755  -0.03062160      0.195483438 -0.035763760
pctSmallHousUnits   -0.028171735          0.46515535      0.42269478        1.00000000  0.189122644  -0.31762911      0.168056075 -0.235016663
houseVacant         -0.070989669          0.24211354      0.09134476        0.18912264  1.000000000  -0.19098349      0.222099655 -0.001808993
pctHousOccup         0.043568154         -0.17647692     -0.03062160       -0.31762911 -0.190983491   1.00000000     -0.159310997  0.133240348
pctVacantBoarded    -0.175937801          0.55580817      0.19548344        0.16805608  0.222099655  -0.15931100      1.000000000 -0.115316371
rentQrange          -0.125645390         -0.14535872     -0.03576376       -0.23501666 -0.001808993   0.13324035     -0.115316371  1.000000000
medRentpctHousInc   -0.201289041          0.34073451      0.34459176        0.33895331  0.081046531  -0.21338804      0.155632784  0.150771779
medOwnCostPctWO     -0.013441573          0.20630820     -0.14367608        0.05736692  0.014706191  -0.14314070      0.177622287  0.159421108
pctForeignBorn      -0.351858095          0.15043728      0.72823854        0.33393437  0.088702672   0.09253142     -0.046217517  0.303191573
pctBornStateResid    0.133704735          0.02592819     -0.24781275       -0.21465234 -0.097702619   0.17938707      0.158786006 -0.218091613
pctOfficDrugUnit    -0.047004403          0.31197280      0.11025193        0.22979470  0.240768533  -0.07297227      0.203275365  0.014780519
violentPerPop       -0.141478711          0.73961847      0.40294564        0.46610650  0.288353604  -0.25561333      0.475130968 -0.115718678
                    medRentpctHousInc medOwnCostPctWO pctForeignBorn pctBornStateResid pctOfficDrugUnit violentPerPop
pctBlack                   0.18444887      0.21675866    -0.09241652        0.09074844       0.25264027    0.62533900
pctWhite                  -0.33177946     -0.05770524    -0.38803015        0.11677719      -0.25849976   -0.67719557
pct12-29                   0.29995579     -0.08660812     0.07031523       -0.04831816       0.07947460    0.10993216
pctUrban                  -0.02394126     -0.01935197     0.27507197       -0.22634471       0.22458477    0.07395461
asianPerCap               -0.13104581      0.02290282     0.04694477       -0.05232066      -0.05912096   -0.13313562
otherPerCap               -0.05240768      0.03285557     0.06519747       -0.10020604      -0.02222162   -0.10041488
pctEmployMfg              -0.27556522      0.03652400    -0.02969928        0.34189063      -0.07590639   -0.05769492
pctMaleDivorc              0.09422160     -0.05822100    -0.11492294       -0.05125950       0.22033626    0.51663043
pctKids2Par               -0.36317165     -0.12829139    -0.03369781       -0.06276262      -0.28928097   -0.72815910
pctWorkMom-18             -0.20128904     -0.01344157    -0.35185810        0.13370474      -0.04700440   -0.14147871
pctKidsBornNevrMarr        0.34073451      0.20630820     0.15043728        0.02592819       0.31197280    0.73961847
pctPopDenseHous            0.34459176     -0.14367608     0.72823854       -0.24781275       0.11025193    0.40294564
pctSmallHousUnits          0.33895331      0.05736692     0.33393437       -0.21465234       0.22979470    0.46610650
houseVacant                0.08104653      0.01470619     0.08870267       -0.09770262       0.24076853    0.28835360
pctHousOccup              -0.21338804     -0.14314070     0.09253142        0.17938707      -0.07297227   -0.25561333
pctVacantBoarded           0.15563278      0.17762229    -0.04621752        0.15878601       0.20327536    0.47513097
rentQrange                 0.15077178      0.15942111     0.30319157       -0.21809161       0.01478052   -0.11571868
medRentpctHousInc          1.00000000      0.08646156     0.29886841       -0.22887179       0.10982599    0.31135051
medOwnCostPctWO            0.08646156      1.00000000    -0.02413511        0.17190435       0.05127715    0.05246134
pctForeignBorn             0.29886841     -0.02413511     1.00000000       -0.47535475       0.12789469    0.20164666
pctBornStateResid         -0.22887179      0.17190435    -0.47535475        1.00000000      -0.11705261   -0.08666947
pctOfficDrugUnit           0.10982599      0.05127715     0.12789469       -0.11705261       1.00000000    0.31912127
violentPerPop              0.31135051      0.05246134     0.20164666       -0.08666947       0.31912127    1.00000000
```

In addition to this, the variance inflation factor (VIF) values can be calculated.

```
vif(lm.fit)
         pctBlack          pctWhite        `pct12-29`          pctUrban        asianPerCap         otherPerCap
        11.521016         12.633484          1.710365          1.341629           1.215820            1.125621
     pctEmployMfg     pctMaleDivorc       pctKids2Par   `pctWorkMom-18` pctKidsBornNevrMarr    pctPopDenseHous
         1.447163          3.737126         11.767610          1.428727           7.980885            5.361674
pctSmallHousUnits       houseVacant      pctHousOccup  pctVacantBoarded          rentQrange  medRentpctHousInc
         3.606784          1.169145          1.717957          1.704862            1.732170           1.693879
  medOwnCostPctWO    pctForeignBorn pctBornStateResid  pctOfficDrugUnit
         1.451586          5.900580          2.008470          1.283560
```

From the data above, one can see that pctBlack, pctWhite, pctKids2Par, pctKidsBornNevrMarr have high VIF values. From the correlation matrix we can observe that pctBlack and pctWhite are highly correlated and pctWhite has a high p-value. Hence, pctWhite can be removed from the model. It is also observed that pctKidsBornNevrMarr and pctKids2Par have high VIF values and they are highly correlated. We can remove pctKids2Par. Similarly, any variable with high p-value were removed. Before performing the next iteration of the regression analysis, the model was tested for outliers and leverage points.




```
> outlierTest(lm.fit)
      rstudent unadjusted p-value Bonferonni p
165   7.189727         1.0894e-12   1.4489e-09
1594  6.038750         2.0214e-09   2.6885e-06
989   5.797469         8.4324e-09   1.1215e-05
167   4.770054         2.0487e-06   2.7247e-03
1983  4.402528         1.1573e-05   1.5392e-02
707   4.304371         1.7996e-05   2.3935e-02
821  -4.263531         2.1569e-05   2.8687e-02
706   4.158305         3.4153e-05   4.5423e-02
```

It can be observed that observations 165, 1594, and 989 are outliers and have influence in the OLS fit and observation 22 is a high leverage point. These points were deleted and the new model was run.

```
> summary(lm.fit2)

Call:
lm(formula = violentPerPop ~ . - pctWhite - asianPerCap - medRentpctHousInc -
    pctSmallHousUnits - pctHousOccup - pctVacantBoarded - pctOfficDrugUnit -
    pctForeignBorn - pctBornStateResid, data = crimeViolenceFinalData2[train,
    ])

Residuals:
     Min       1Q   Median       3Q      Max
-1367.84  -172.34   -31.79   120.14  1729.53

Coefficients:
                      Estimate Std. Error t value Pr(>|t|)
(Intercept)          1.627e+03  2.636e+02   6.173 8.91e-10 ***
pctBlack             8.374e+00  1.277e+00   6.557 7.89e-11 ***
`pct12-29`          -4.962e+00  1.709e+00  -2.904 0.003748 **
pctUrban             6.567e-01  2.208e-01   2.974 0.002998 **
otherPerCap          2.348e-03  1.137e-03   2.065 0.039149 *
pctEmployMfg        -3.555e+00  1.215e+00  -2.927 0.003478 **
pctMaleDivorc        2.901e+01  5.581e+00   5.199 2.33e-07 ***
pctKids2Par         -1.215e+01  2.240e+00  -5.423 6.97e-08 ***
`pctWorkMom-18`     -6.061e+00  1.615e+00  -3.753 0.000182 ***
pctKidsBornNevrMarr  4.371e+01  8.061e+00   5.423 6.98e-08 ***
pctPopDenseHous      1.581e+01  2.003e+00   7.893 6.18e-15 ***
houseVacant          1.994e-02  2.410e-03   8.273 3.18e-16 ***
rentQrange           4.174e-01  1.294e-01   3.226 0.001286 **
medOwnCostPctWO     -2.350e+01  6.965e+00  -3.374 0.000764 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 334.9 on 1313 degrees of freedom
  (3 observations deleted due to missingness)
Multiple R-squared:  0.664,	Adjusted R-squared:  0.6606
F-statistic: 199.6 on 13 and 1313 DF,  p-value: < 2.2e-16
```

```
> lm.pred2<-predict(lm.fit2,crimeViolenceFinalData2[-train,])
> mean((lm.pred2-crimeViolenceFinalData2[-train,]$violentPerPop)^2)
[1] 146042.6
```

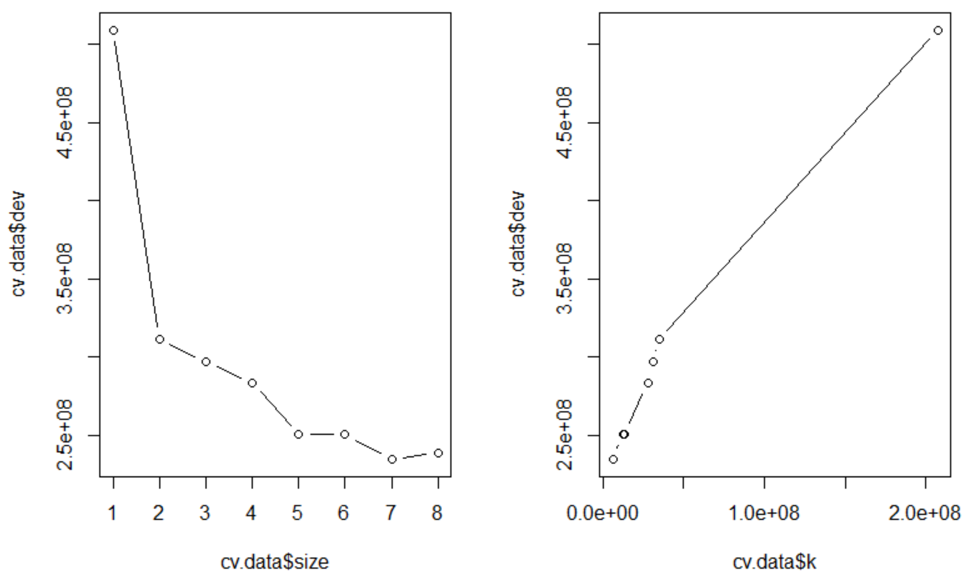
It can be observed that all the variables in the model are now significant and there has been an increase in the adjusted R-squared as well, which suggests that it is a better fit. However, there was also an increase in the RMSE for this model.

## 4.2 Regression Trees

Regression trees are easy to explain and are easily interpretable. Trees can be displayed graphically with the help of plot functions. In the analysis of the dataset, it is consider using a pruned tree using a cross-validation approach. The code is given below:

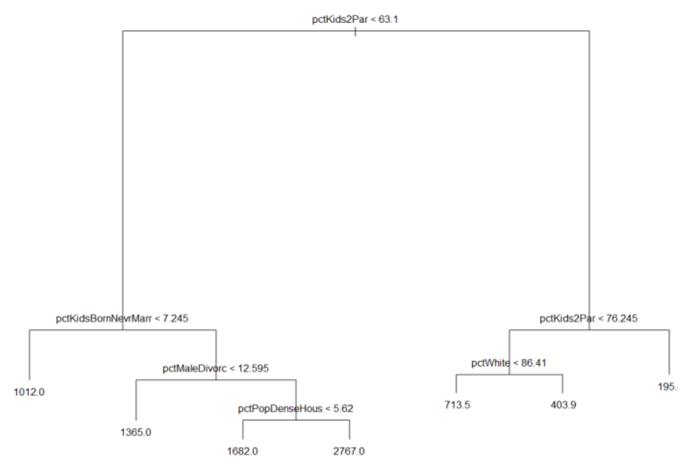
```
set.seed(10)
tree.data<-tree(violentPerPop~.,crimeViolenceFinalData1[train,])
cv.data =cv.tree(tree.data,FUN=prune.tree)
par(mfrow =c(1,2))
plot(cv.data$size ,cv.data$dev ,type="b")
plot(cv.data$k ,cv.data$dev ,type="b")
```

The corresponding plot is shown below:




```
prune.data =prune.tree (tree.data,best = 7)
par(mfrow =c(1,1))
plot(prune.data)
text(prune.data,pretty=0,xpd=TRUE)
tree.pred1=predict(prune.data,crimeViolenceFinalData1[-train,])
mean((tree.pred1-crimeViolenceFinalData1[-train,]$violentPerPop)^2)
> mean((tree.pred1-crimeViolenceFinalData1[-train,]$violentPerPop)^2)
[1] 166975.5
```

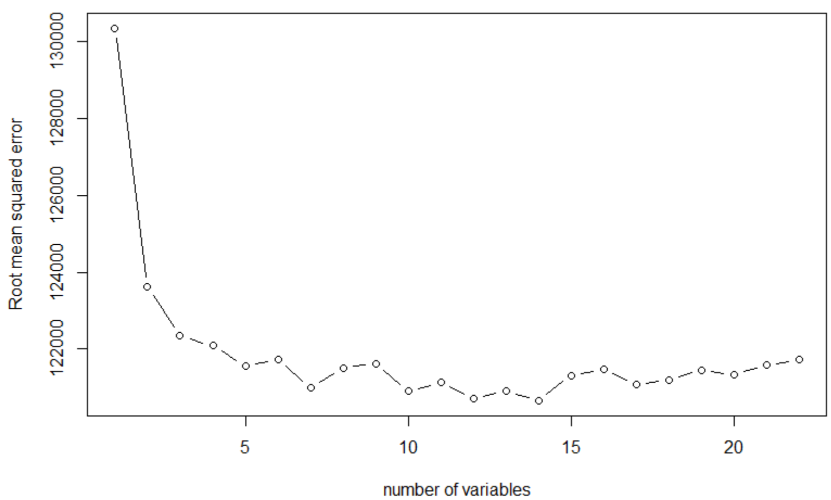
The model was run for tree size 7 and the mean was calculated. The RMSE of this model was higher than that of the linear regression model. Next, the random forest model was applied to check model efficiency.




## 4.3 Random Forests

Random forests give models that have low variance. Another model that was considered during this analysis was bagging. Though bagging improves the accuracy over using a single tree, it is hard to interpret the model. Random forest on the other hand is a very efficient learning method. It builds on the idea of bagging, but it de-correlates the trees, thereby reducing the variance of the whole model. For each tree, each time a split in a tree is considered, a random sample of *m* predictors is chosen as split candidates from among the *p* predictors. We calculate the value of *m* through iteration using the code below:

```
library(randomForest)
set.seed(5)
par(mfrow = c(1,1))
result<-NULL
for(k in 1:22){
  rf.model<-randomForest(violentPerPop~.,crimeViolenceFinalData1[train,],mtry=k,importance=TRUE,ntree=1501)
  pred.value<-predict(rf.model,crimeViolenceFinalData1[-train,])
  error<-mean((pred.value-crimeViolenceFinalData1[-train,]$violentPerPop)^2)
  result<-rbind(result,c(k,error))
}
plot(result,xlab="number of variables",ylab="misclassification",type="b")
```



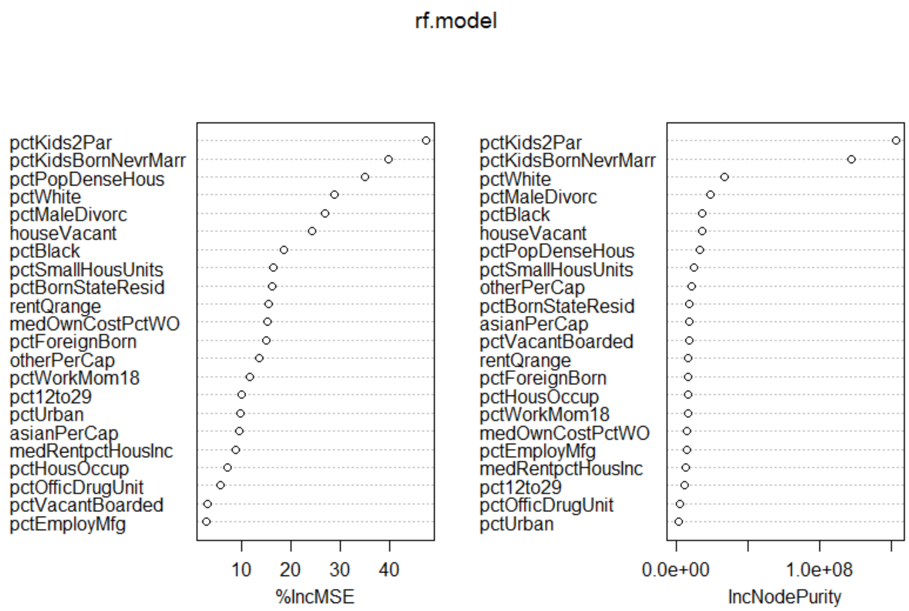

```
> result
      [,1]     [,2]
 [1,]    1 130347.1
 [2,]    2 123613.4
 [3,]    3 122348.4
 [4,]    4 122087.4
 [5,]    5 121563.9
 [6,]    6 121728.2
 [7,]    7 120987.9
 [8,]    8 121512.8
 [9,]    9 121621.5
[10,]   10 120902.8
[11,]   11 121139.1
[12,]   12 120702.4
[13,]   13 120907.2
[14,]   14 120660.7
[15,]   15 121318.3
[16,]   16 121479.3
[17,]   17 121074.1
[18,]   18 121196.8
[19,]   19 121460.4
[20,]   20 121338.5
[21,]   21 121580.7
[22,]   22 121721.1
```

From the graph above and the predictors' result table, it can be inferred that mtry=14 gives the least test error. The code below with mtry=14 gives RMSE as 120,993 which is far better than the pruned tree.

```
> set.seed(1111)
> rf.model<-randomForest(violentPerPop~.,crimeViolenceFinalData1[train,],mtry=14,importance=TRUE,ntree=1501)
> pred.value<-predict(rf.model,crimeViolenceFinalData1[-train,])
> mean((pred.value-crimeViolenceFinalData1[-train,]$violentPerPop)^2)
[1] 120993.3
```

We can get the variable importance using the **varImpPlot()** and the **importance()** function. The importance plots are shown below:

```
> importance(rf.model)
                      %IncMSE IncNodePurity
pctBlack            18.596190      18203135
pctWhite            28.775650      33598239
pct12to29            9.937725       5907633
pctUrban             9.631237       1885728
asianPerCap          9.427520       8874904
otherPerCap         13.553577      10690219
pctEmployMfg         2.752074       7267758
pctMaleDivorc       26.849688      23989813
pctKids2Par         47.423263     152828219
pctWorkMom18        11.634477       8042306
pctKidsBornNevrMarr 39.749917     122037269
pctPopDenseHous     35.042508      16782115
pctSmallHousUnits   16.321099      12394226
houseVacant         24.321962      18151784
pctHousOccup         6.997586       8191062
pctVacantBoarded     3.141162       8824431
rentQrange          15.467322       8657224
medRentpctHousInc    8.751142       6805535
medOwnCostPctWO     15.139890       7358574
pctForeignBorn      14.988458       8520580
pctBornStateResid   16.246287       9170241
pctOfficDrugUnit     5.722735       2587255
```




It can be interpreted that pctKids2Par, pctKidsBornNevrMarr, pctPoPDenseHous, pctWhite, pctMaleDivorc, houseVacant, and pctBlack are important variables to be considered when predicting violent crime per population (violentPerPop).

## 5. Conclusion

The final variables obtained from the lasso regression has 23 variables including the intercept term. Some of the variables are pctUrban (percentage of people living in areas classified as urban), pctBlack (percentage of population that is African American), pctWhite (percentage of population that is Caucasian) pctKidsBornNeverMar (percentage of kids born to never married), pctPersDenseHous (percent of persons in dense housing (more than 1 person per room), pctHousLess3BR (percent of housing units with less than 3 bedrooms), housevacant (number of vacant households), pctHousOccup (percent of housing occupied), and pctOfficDrugUnit (percent of officers assigned to drug units). Based on our everyday life and intuition, with information from news channels and the internet, it can be observed that these variables influence crimes in our cities.

However, lasso regression does not give the significance attached to each variable with respect to the output and the data at hand. This can be evaluated using other regression models such as linear regression or regression trees, which gives a better understanding of how the predictor variables influence the output.

Of the three models we have to choose a model that can predict the violent crimes accurately. The pruned tree has the highest test error among the three models chosen for final model prediction. The pruned tree and the linear regression models are easy to interpret when compared to the random forest. Though both the variables and the model are significant in the multiple linear regression model, the test error rate is really high when compared to the random forest model. This could be because of high variance that is inherent in the linear regression model. Yet, random forests approach has the lowest test error rate. However, this comes at the price of model interpretability.

Overall, the RMSE of regression trees was over 128,000 while RMSE for linear regression was 127,357. The RMSE for backward stepwise selection was 122,106 while random forest had the lowest RMSE of

120,993. Thus this data analysis that violentPerPop (violent crime per population) is most heavily influenced by pctBlack, pctWhite, pct12to29, and pctUrban variables.

## 6. Reference

Chen, H., W. Chung, J.J. Xu, G. Wang, Y. Qin, and M. Chau. "Crime Data Mining: A General Framework and Some Examples." Computer 37.4 (2004): 50-56. Wiley. Web. 20 Apr. 2016.

Levy, Francesca. "America's Safest Cities." Forbes. Forbes Magazine, n.d. Web. 27 Apr. 2016.

McMillen, Daniel P., “Issues in Spatial Data Analysis” Journal of Regional Science, Vol. 50, Issue 1. (2009):119-141.Wiley.Web. 20 Apr. 2016.

*Source for the dataset*:

Creator: Michael Redmond; Computer Science; La Salle University; Philadelphia, PA, 19141, USA

Culled from 1990 US Census, 1995 US FBI Uniform Crime Report, 1990 US Law Enforcement Management and Administrative Statistics Survey, available from ICPSR at U of Michigan.