Week 6 Assignment - Principle Component Analysis (PCA)

Omer Farooq (EDx ID: mfarooq4)

02/18/2020

Table of Contents

QUESTION 9.1

Using the same crime data set uscrime.txt as in Question 8.2, apply Principal Component Analysis and then create a regression model using the first few principal components. Specify your new model in terms of the original variables (not the principal components), and compare its quality to that of your solution to Question 8.2. You can use the R function prcomp for PCA. (Note that to first scale the data, you can include scale. = TRUE to scale as part of the PCA function. Don't forget that, to make a prediction for the new city, you'll need to unscale the coefficients (i.e., do the scaling calculation in reverse)!)

The factor values we used in Q8.2 to predict the Crime value are below for reference

- M = 14.0
- So = 0
- Ed = 10.0
- Po1 = 12.0
- Po2 = 15.5
- LF = 0.640
- M.F = 94.0
- Pop = 150
- NW = 1.1
- U1 = 0.120
- U2 = 3.6
- Wealth = 3200
- Ineq = 20.1
- Prob = 0.04
- Time = 39.0

Before jumping into the solution, I wrote down the steps needed to get to the solution. This help us track our solution as we go through it. Disclaimer: I got to these steps based on several discussions that happened in Piazza posts this week.

1. Perform PCA on scaled data. PCA R function prcomp does it all (necessary axis transformations to maximize the variance in the data explained by the least amount of principal components). It also has a parameter to scale the data which is an important step of the process.

2. Identify the PCs using the plot.
3. Build the regression model using the first few principle components (PCs). This will give us the regression coefficients based on PCs for scaled data.
4. Perform trace back steps (i.e. unscale) to get the coefficients of the linear model back in terms of the original predictors.

5. Perform prediction using the unscaled coefficients from the model based on PCs.
6. Compare the prediction from step 5 to the prediction from Q8.2 which did not use PCA.

First, I loaded the data.

```
#setting the seed so that results are the same at every run
set.seed(101)

#loading data
crimedata <- read.delim("data_9.1/uscrime.txt")

#quick glance at the data
head(crimedata)

##      M So   Ed  Po1  Po2    LF   M.F Pop   NW    U1  U2 Wealth Ineq     Prob
## 1 15.1  1  9.1  5.8  5.6 0.510  95.0  33 30.1 0.108 4.1   3940 26.1 0.084602
## 2 14.3  0 11.3 10.3  9.5 0.583 101.2  13 10.2 0.096 3.6   5570 19.4 0.029599
## 3 14.2  1  8.9  4.5  4.4 0.533  96.9  18 21.9 0.094 3.3   3180 25.0 0.083401
## 4 13.6  0 12.1 14.9 14.1 0.577  99.4 157  8.0 0.102 3.9   6730 16.7 0.015801
## 5 14.1  0 12.1 10.9 10.1 0.591  98.5  18  3.0 0.091 2.0   5780 17.4 0.041399
## 6 12.1  0 11.0 11.8 11.5 0.547  96.4  25  4.4 0.084 2.9   6890 12.6 0.034201
##      Time Crime
## 1 26.2011   791
## 2 25.2999  1635
## 3 24.3006   578
## 4 29.9012  1969
## 5 21.2998  1234
## 6 20.9995   682
```

I built a dataframe of the provided predictors in Q8.2 to use for prediction later on.

```
#data frame with data we need to predict crime for
predictdata <-data.frame(M = 14.0,So = 0, Ed = 10.0, Po1 = 12.0, Po2 =
15.5,LF = 0.640, M.F = 94.0, Pop = 150, NW = 1.1, U1 = 0.120, U2 = 3.6,
Wealth = 3200, Ineq = 20.1, Prob = 0.040,Time = 39.0)
```

Before I jumped into models, I checked the pearson correlation matrix of the Crimes data. Value is 1 and -1 indicate positive and negative correlation where 0 indicates no correlation. This will help her see if the factors have any correlation b/w themselves. As we know, PCA helps with:

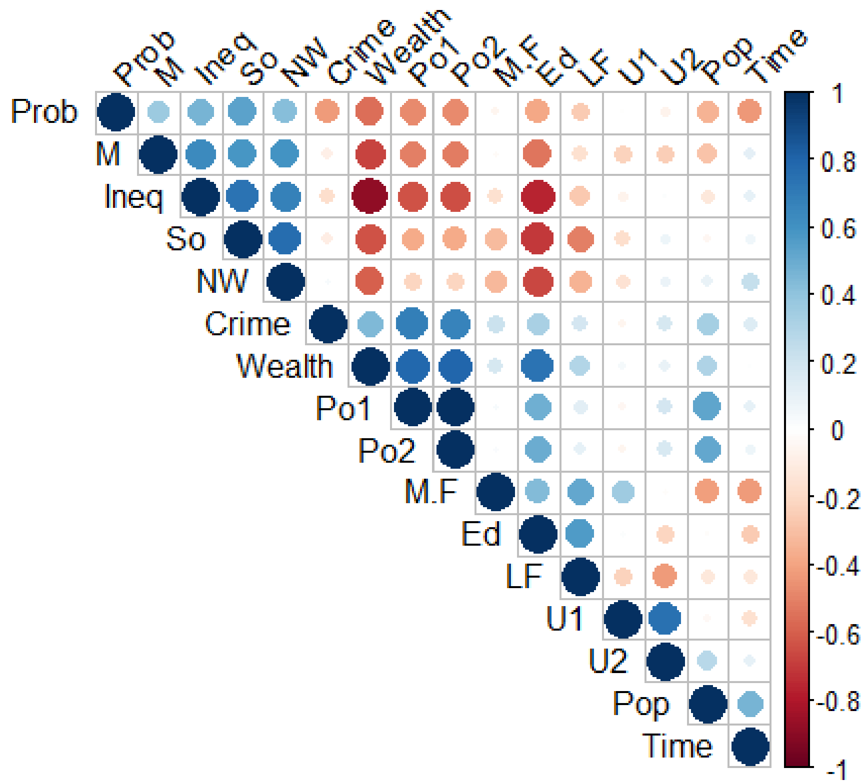
- Remove correlation
- Reduce the number of factors by factor extraction

We can see from the visual that several factors have very strong positive or negative correlation (e.g. Ineq/Wealth and Ineq/Eq have neg correlation whereas Po1/Po2 have high positive correlation).

```
library(corrplot) #for correlation plot

#pearson correlation matrix
corrmat <- cor(crimedata)
round(corrmat, 2)
```

```
##            M    So    Ed   Po1   Po2    LF   M.F   Pop    NW    U1    U2 Wealth
## M       1.00  0.58 -0.53 -0.51 -0.51 -0.16 -0.03 -0.28  0.59 -0.22 -0.24  -0.67
## So      0.58  1.00 -0.70 -0.37 -0.38 -0.51 -0.31 -0.05  0.77 -0.17  0.07  -0.64
## Ed     -0.53 -0.70  1.00  0.48  0.50  0.56  0.44 -0.02 -0.66  0.02 -0.22   0.74
## Po1    -0.51 -0.37  0.48  1.00  0.99  0.12  0.03  0.53 -0.21 -0.04  0.19   0.79
## Po2    -0.51 -0.38  0.50  0.99  1.00  0.11  0.02  0.51 -0.22 -0.05  0.17   0.79
## LF     -0.16 -0.51  0.56  0.12  0.11  1.00  0.51 -0.12 -0.34 -0.23 -0.42   0.29
## M.F    -0.03 -0.31  0.44  0.03  0.02  0.51  1.00 -0.41 -0.33  0.35 -0.02   0.18
## Pop    -0.28 -0.05 -0.02  0.53  0.51 -0.12 -0.41  1.00  0.10 -0.04  0.27   0.31
## NW      0.59  0.77 -0.66 -0.21 -0.22 -0.34 -0.33  0.10  1.00 -0.16  0.08  -0.59
## U1     -0.22 -0.17  0.02 -0.04 -0.05 -0.23  0.35 -0.04 -0.16  1.00  0.75   0.04
## U2     -0.24  0.07 -0.22  0.19  0.17 -0.42 -0.02  0.27  0.08  0.75  1.00   0.09
## Wealth -0.67 -0.64  0.74  0.79  0.79  0.29  0.18  0.31 -0.59  0.04  0.09   1.00
## Ineq    0.64  0.74 -0.77 -0.63 -0.65 -0.27 -0.17 -0.13  0.68 -0.06  0.02  -0.88
## Prob    0.36  0.53 -0.39 -0.47 -0.47 -0.25 -0.05 -0.35  0.43 -0.01 -0.06  -0.56
## Time    0.11  0.07 -0.25  0.10  0.08 -0.12 -0.43  0.46  0.23 -0.17  0.10   0.00
## Crime  -0.09 -0.09  0.32  0.69  0.67  0.19  0.21  0.34  0.03 -0.05  0.18   0.44
##         Ineq  Prob  Time Crime
## M       0.64  0.36  0.11 -0.09
## So      0.74  0.53  0.07 -0.09
## Ed     -0.77 -0.39 -0.25  0.32
## Po1    -0.63 -0.47  0.10  0.69
## Po2    -0.65 -0.47  0.08  0.67
## LF     -0.27 -0.25 -0.12  0.19
## M.F    -0.17 -0.05 -0.43  0.21
## Pop    -0.13 -0.35  0.46  0.34
## NW      0.68  0.43  0.23  0.03
## U1     -0.06 -0.01 -0.17 -0.05
## U2      0.02 -0.06  0.10  0.18
## Wealth -0.88 -0.56  0.00  0.44
## Ineq    1.00  0.47  0.10 -0.18
## Prob    0.47  1.00 -0.44 -0.43
## Time    0.10 -0.44  1.00  0.15
## Crime  -0.18 -0.43  0.15  1.00
```

```r
#plotting the correlation matrix
corrplot(cormat, type = "upper", order = "hclust",
         tl.col = "black", tl.srt = 45)
```



Step 1 & 2 - Perform PCA & Identify PCs

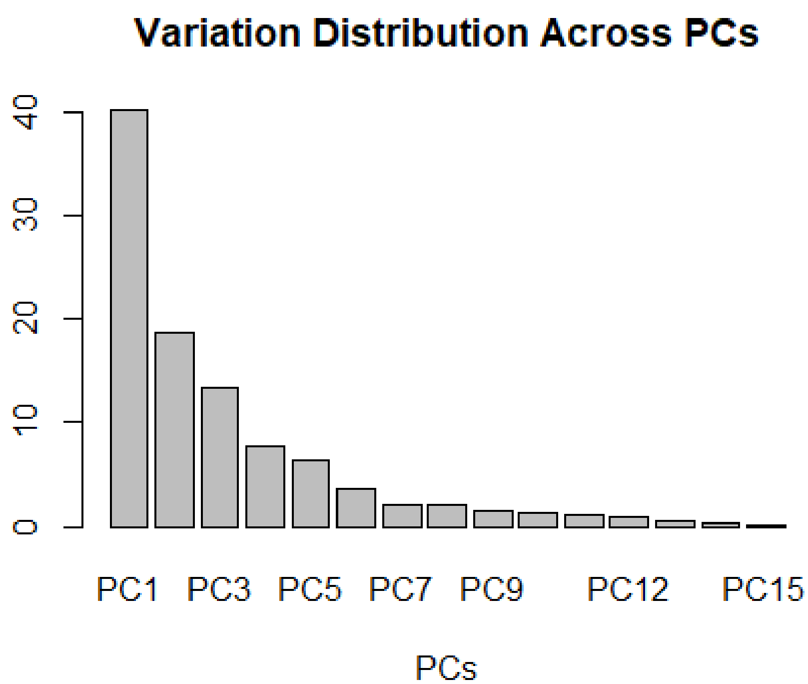
I performed the PCA on the scaled crime data below and plotted the variation measure of each PC in a bar graph. Variaton measure of a PC is Sum of Sqaured Distances of data point projection on the PC dimension from the center divided by the n-1 where n is the number of records. It essentially described how much a PC represents the variation in the data.

The plots clearly showed the 1st 4 PCs covered ~80% of the data variation. And 1st 5 PCs covered ~89% of the variation. 4 of 5 PCs would be good to use in the regression model moving forward.

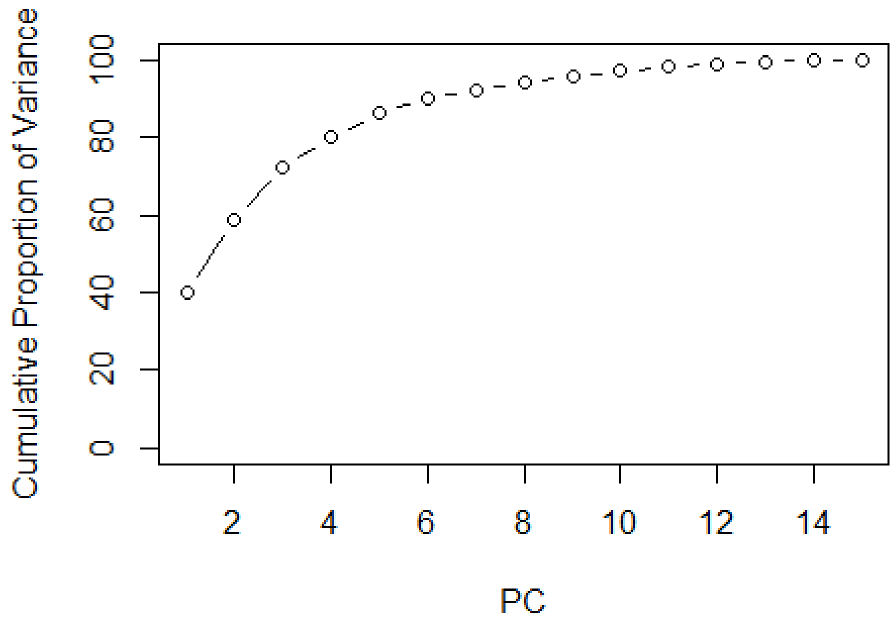
```r
pca_crime <- prcomp(crimedata[,1:15], scale. = TRUE)
pca_crime_sum <- summary(pca_crime)
pca_crime_sum

## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6     PC7
## Standard deviation     2.4534 1.6739 1.4160 1.07806 0.97893 0.74377 0.56729
## Proportion of Variance 0.4013 0.1868 0.1337 0.07748 0.06389 0.03688 0.02145
## Cumulative Proportion  0.4013 0.5880 0.7217 0.79920 0.86308 0.89996 0.92142
##                            PC8     PC9    PC10    PC11    PC12    PC13   PC14
## Standard deviation     0.55444 0.48493 0.44708 0.41915 0.35804 0.26333 0.2418
## Proportion of Variance 0.02049 0.01568 0.01333 0.01171 0.00855 0.00462 0.0039
## Cumulative Proportion  0.94191 0.95759 0.97091 0.98263 0.99117 0.99579 0.9997
```

```
##                              PC15
## Standard deviation     0.06793
## Proportion of Variance 0.00031
## Cumulative Proportion  1.00000
```

```r
#plot of the proportion of variance of each PC
barplot((pca_crime_sum$importance[2,])*100,
        main="Variation Distribution Across PCs",
        xlab="PCs")
```




```r
#plot of cumulative sum of variance
plot(pca_crime_sum$importance[3,]*100,
     xlab = "PC",
     ylab = "Cumulative Proportion of Variance",
     ylim = c(0,100),
     type = "b")
```



Step 3 - Build Regression Model Using Principle Components

I built two models, first with 4 PCs and then with 5. First, I had to bind together the PCs with the Crime Rate column to build the dataset to be used. I did not scale the crime rate column b/c we are trying to predict those values, scaling it would affect our prediction.

```
#binding together the 1st 4 PCs with the crime rate column.
PC_crime_data4 <- as.data.frame(cbind(pca_crime$x[,1:4], crimedata[,16]))

#4 PCs regression model
four_PC_model <- lm(V5~., data = PC_crime_data4)
summary(four_PC_model)

##
## Call:
## lm(formula = V5 ~ ., data = PC_crime_data4)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -557.76 -210.91  -29.08  197.26  810.35
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   905.09      49.07  18.443  < 2e-16 ***
## PC1            65.22      20.22   3.225  0.00244 **
## PC2           -70.08      29.63  -2.365  0.02273 *
## PC3            25.19      35.03   0.719  0.47602
```

```
## PC4           69.45      46.01   1.509  0.13872
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 336.4 on 42 degrees of freedom
## Multiple R-squared:  0.3091, Adjusted R-squared:  0.2433
## F-statistic: 4.698 on 4 and 42 DF,  p-value: 0.003178
```

```
#binding together the 1st 5 PCs with the crime rate column.
PC_crime_data5 <- as.data.frame(cbind(pca_crime$x[,1:5], crimedata[,16]))

#5 PCs regression model
five_PC_model <- lm(V6~., data = PC_crime_data5)
summary(five_PC_model)
```

```
##
## Call:
## lm(formula = V6 ~ ., data = PC_crime_data5)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -420.79 -185.01   12.21  146.24  447.86
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   905.09      35.59  25.428  < 2e-16 ***
## PC1            65.22      14.67   4.447 6.51e-05 ***
## PC2           -70.08      21.49  -3.261  0.00224 **
## PC3            25.19      25.41   0.992  0.32725
## PC4            69.45      33.37   2.081  0.04374 *
## PC5          -229.04      36.75  -6.232 2.02e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 244 on 41 degrees of freedom
## Multiple R-squared:  0.6452, Adjusted R-squared:  0.6019
## F-statistic: 14.91 on 5 and 41 DF,  p-value: 2.446e-08
```

Step 4 - Get coefficients in terms of the original predictors.

This step involved,

- First getting the coefficients for the PCA regression (intercept and 4 PC coefficients)
- Next, these coefficients were transformed to orginal 15 variables by matrix multiplication with the rotation matrixfrom the PCA output
- Lastly, these coefficients had to be unscaled so that these could be used for prediction. Unscaling was explained in this link (https://stats.stackexchange.com/questions/74622/converting-standardized-betas-back-to-original-variables). The formula shown below told me that following two formulas could be used to unscale the coefficients:

1. Original Coefficients = Scaled Coefficients / Standard Deviation
2. Original Intercept = Scaled Intercept - (Scaled Coefficients X Mean / Standard Deviation)

$$\hat{Y} = \left(\hat{\beta}_0 - \sum_{j=1}^{k} \hat{\beta}_j \frac{\bar{x}_j}{S_j}\right) + \sum_{j=1}^{k} \left(\frac{\hat{\beta}_j}{S_j}\right) x_j$$

Regression equation for scaled data

```
#PCA coefficients from 4 PC regression model
four_intercept <- four_PC_model$coefficients[1]
four_coef <- four_PC_model$coefficients[2:5]

four_intercept

## (Intercept)
##    905.0851

four_coef

##       PC1       PC2       PC3       PC4
##  65.21593 -70.08312  25.19408  69.44603

#transforming coefficents for the original SCALED variables
four_coef_all <- pca_crime$rotation[,1:4] %*% four_coef
four_coef_all

##                [,1]
## M      -21.277963
## So      10.223091
## Ed      14.352610
## Po1     63.456426
## Po2     64.557974
## LF     -14.005349
## M.F    -24.437572
## Pop     39.830667
## NW      15.434545
## U1     -27.222281
## U2       1.425902
## Wealth  38.607855
## Ineq   -27.536348
## Prob     3.295707
## Time    -6.612616

#Unscaling the coefficients
four_orig_coef <- four_coef_all/sapply(crimedata[,1:15],sd)
four_orig_intercept <- four_intercept -
sum(four_coef_all*sapply(crimedata[,1:15],mean)/sapply(crimedata[,1:15],sd))
```

```
four_orig_intercept

## (Intercept)
##    1666.485

four_orig_coef

##                  [,1]
## M        -16.9307630
## So        21.3436771
## Ed        12.8297238
## Po1       21.3521593
## Po2       23.0883154
## LF      -346.5657125
## M.F       -8.2930969
## Pop        1.0462155
## NW         1.5009941
## U1     -1509.9345216
## U2         1.6883674
## Wealth     0.0400119
## Ineq      -6.9020218
## Prob     144.9492678
## Time      -0.9330765

#PCA coefficients from 5 PC regression model
five_intercept <- five_PC_model$coefficients[1]
five_coef <- five_PC_model$coefficients[2:6]

five_intercept

## (Intercept)
##    905.0851

five_coef

##       PC1        PC2        PC3        PC4        PC5
##   65.21593  -70.08312   25.19408   69.44603 -229.04282

#transforming coefficents for the original SCALED variables
five_coef_all <- pca_crime$rotation[,1:5] %*% five_coef
five_coef_all

##              [,1]
## M       60.794349
## So      37.848243
## Ed      19.947757
## Po1    117.344887
## Po2    111.450787
## LF      76.254902
## M.F    108.126558
## Pop     58.880237
```

```
## NW      98.071790
## U1       2.866783
## U2      32.345508
## Wealth  35.933362
## Ineq    22.103697
## Prob   -34.640264
## Time    27.205022
```

```
#Unscaling the coefficients
five_orig_coef <- five_coef_all/sapply(crimedata[,1:15],sd)
five_orig_intercept <- five_intercept -
sum(five_coef_all*sapply(crimedata[,1:15],mean)/sapply(crimedata[,1:15],sd))

five_orig_intercept
```

```
## (Intercept)
##   -5933.837
```

```
five_orig_coef
```

```
##                  [,1]
## M       4.837374e+01
## So      7.901922e+01
## Ed      1.783120e+01
## Po1     3.948484e+01
## Po2     3.985892e+01
## LF      1.886946e+03
## M.F     3.669366e+01
## Pop     1.546583e+00
## NW      9.537384e+00
## U1      1.590115e+02
## U2      3.829933e+01
## Wealth  3.724014e-02
## Ineq    5.540321e+00
## Prob   -1.523521e+03
## Time    3.838779e+00
```

Step 5 - Prediction Crime for the new city using variable values from Q8.2 based on PCA Model

In this step, I used the coefficients from the 4 and 5 PCA models to make predictions using the variable values we were given in Q8.2 last week. 4 PCA model predicts crime rateof 1113 whereas 5 PCA model predicts 1289. Both values are plausible when we look at the summary of the crime field in the data which has min of 342, max of 1993 and mean of 905. The predicted values are close to the 3rd quartile.

```
#prediction for 4 PCA Model
four_prediction <- four_orig_intercept +
sum(data.frame(mapply('*',predictdata,four_orig_coef)))
four_prediction
```

```
## (Intercept)
##    1112.678
```

```r
#prediction for 5 PCA Model
five_prediction <- five_orig_intercept +
sum(data.frame(mapply('*',predictdata,five_orig_coef)))
five_prediction
```

```
## (Intercept)
##    1388.926
```

```r
summary(crimedata$Crime)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   342.0   658.5   831.0   905.1  1057.5  1993.0
```

Step 6 - Compare the prediction from step 5 to the prediction from Q8.2 which did not use PCA

Lastly, to compare to the results from Q8.2 of last week, I recreated the simple regression model based on all data which predicted the crime rate for the new values to be 155. This is value it not correct due to overfitting. My best model from last week was k-fold CV model and it's prediction being 641 which iscloser to the 1st quartile of the crime values.

Results of both PCA regression (i.e. 1112 and 1388) and regression without PCA (641) are both within the range but close to different quartiles of the crimes data range but both models suffer from overfitting (indicated by high r-squared).

```r
set.seed(101) #to keep output consistent

#simple regression using lm()
model1 <- lm(Crime~. , data=crimedata)
summary(model1)
```

```
##
## Call:
## lm(formula = Crime ~ ., data = crimedata)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -395.74  -98.09   -6.69  112.99  512.67
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5.984e+03  1.628e+03  -3.675 0.000893 ***
## M            8.783e+01  4.171e+01   2.106 0.043443 *
## So          -3.803e+00  1.488e+02  -0.026 0.979765
## Ed           1.883e+02  6.209e+01   3.033 0.004861 **
## Po1          1.928e+02  1.061e+02   1.817 0.078892 .
## Po2         -1.094e+02  1.175e+02  -0.931 0.358830
## LF          -6.638e+02  1.470e+03  -0.452 0.654654
```

```
## M.F          1.741e+01  2.035e+01   0.855 0.398995
## Pop         -7.330e-01  1.290e+00  -0.568 0.573845
## NW           4.204e+00  6.481e+00   0.649 0.521279
## U1          -5.827e+03  4.210e+03  -1.384 0.176238
## U2           1.678e+02  8.234e+01   2.038 0.050161 .
## Wealth       9.617e-02  1.037e-01   0.928 0.360754
## Ineq         7.067e+01  2.272e+01   3.111 0.003983 **
## Prob        -4.855e+03  2.272e+03  -2.137 0.040627 *
## Time        -3.479e+00  7.165e+00  -0.486 0.630708
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 209.1 on 31 degrees of freedom
## Multiple R-squared:  0.8031, Adjusted R-squared:  0.7078
## F-statistic: 8.429 on 15 and 31 DF,  p-value: 3.539e-07
```

```
#predicting crime value
predict(model1, predictdata)
```

```
##        1
## 155.4349
```

```
#k-fold cv regression
library(caret)
```

```
#building model 4
subsets <- c(1:15)
ctrl <- rfeControl(functions = lmFuncs, method = "cv", number = 10, verbose =
FALSE)
model4 <- rfe(crimedata[,-16], crimedata[,16], sizes = subsets, rfeControl =
ctrl)
model4$results
```

```
##    Variables     RMSE  Rsquared      MAE   RMSESD RsquaredSD     MAESD
## 1          1 369.5842 0.4027672 300.4540 124.5373  0.2633520  89.92889
## 2          2 351.4894 0.2864740 280.7087 126.8533  0.3333725  91.60052
## 3          3 372.2303 0.2365345 296.2431 127.4563  0.2693231  94.51961
## 4          4 363.9416 0.3030541 294.2948  94.4162  0.3308304  72.00409
## 5          5 327.3928 0.4314587 273.7917 111.0290  0.3125395  84.95545
## 6          6 329.9003 0.4325974 274.7243 109.4702  0.2905849  86.99893
## 7          7 277.5265 0.5440652 231.6929 111.9335  0.3432121  97.34323
## 8          8 287.8287 0.5645258 238.4586 130.3624  0.3830956 105.68070
## 9          9 244.5158 0.5998315 204.8487 119.5225  0.3163053  94.67931
## 10        10 235.0783 0.6205952 192.3008 120.8504  0.3113745  97.34578
## 11        11 232.3357 0.6202498 191.2804 111.3571  0.3132973  90.44976
## 12        12 251.8811 0.5630041 204.0425 113.4631  0.3261040  93.40328
## 13        13 264.3894 0.5281037 217.0579 129.1861  0.3307715 112.87886
## 14        14 264.9621 0.5460892 213.7700 135.8352  0.3401456 115.24794
## 15        15 282.8904 0.5332446 227.3037 125.5100  0.3104522 106.87101
```

```r
#model suggest best predictors (it's suggest 11 predictors)
predictors(model4)

##  [1] "U1"   "Prob" "LF"   "Po1"  "Ed"   "U2"   "Po2"  "So"   "M"    "Ineq"
## [11] "M.F"

#predicting crime value
predict(model4,predictdata)

##        1
## 641.0715
```