Homework 6

Q 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)!)

##Import data and use R function prcomp for PCA (Scaling)

```
set.seed(1)
uscrime<-read.table('uscrime.txt',header=T)
pca<-prcomp(uscrime[,1:15],scale. = T)  # excluding the response variable
pca
```

```
## Standard deviations (1, .., p=15):
##  [1] 2.45335539 1.67387187 1.41596057 1.07805742 0.97892746 0.74377006
##  [7] 0.56729065 0.55443780 0.48492813 0.44708045 0.41914843 0.35803646
## [13] 0.26332811 0.24180109 0.06792764
##
## Rotation (n x k) = (15 x 15):
##                PC1         PC2          PC3         PC4         PC5
## M      -0.30371194  0.06280357  0.1724199946 -0.02035537 -0.35832737
## So     -0.33088129 -0.15837219  0.0155433104  0.29247181 -0.12061130
## Ed      0.33962148  0.21461152  0.0677396249  0.07974375 -0.02442839
## Po1     0.30863412 -0.26981761  0.0506458161  0.33325059 -0.23527680
## Po2     0.31099285 -0.26396300  0.0530651173  0.35192809 -0.20473383
## LF      0.17617757  0.31943042  0.2715301768 -0.14326529 -0.39407588
## M.F     0.11638221  0.39434428 -0.2031621598  0.01048029 -0.57877443
## Pop     0.11307836 -0.46723456  0.0770210971 -0.03210513 -0.08317034
## NW     -0.29358647 -0.22801119  0.0788156621  0.23925971 -0.36079387
## U1      0.04050137  0.00807439 -0.6590290980 -0.18279096 -0.13136873
## U2      0.01812228 -0.27971336 -0.5785006293 -0.06889312 -0.13499487
## Wealth  0.37970331 -0.07718862  0.0100647664  0.11781752  0.01167683
## Ineq   -0.36579778 -0.02752240 -0.0002944563 -0.08066612 -0.21672823
## Prob   -0.25888661  0.15831708 -0.1176726436  0.49303389  0.16562829
## Time   -0.02062867 -0.38014836  0.2235664632 -0.54059002 -0.14764767
##                 PC6         PC7         PC8         PC9        PC10        PC11
## M      -0.449132706 -0.15707378 -0.55367691  0.15474793 -0.01443093  0.39446657
## So     -0.100500743  0.19649727  0.22734157 -0.65599872  0.06141452  0.23397868
## Ed     -0.008571367 -0.23943629 -0.14644678 -0.44326978  0.51887452 -0.11821954
## Po1    -0.095776709  0.08011735  0.04613156  0.19425472 -0.14320978 -0.13042001
## Po2    -0.119524780  0.09518288  0.03168720  0.19512072 -0.05929780 -0.13885912
## LF      0.504234275 -0.15931612  0.25513777  0.14393498  0.03077073  0.38532827
## M.F    -0.074501901  0.15548197 -0.05507254 -0.24378252 -0.35323357 -0.28029732
## Pop     0.547098563  0.09046187 -0.59078221 -0.20244830 -0.03970718  0.05849643
## NW      0.051219538 -0.31154195  0.20432828  0.18984178  0.49201966 -0.20695666
```

```
## U1      0.017385981 -0.17354115 -0.20206312  0.02069349  0.22765278 -0.17857891
## U2      0.048155286 -0.07526787  0.24369650  0.05576010 -0.04750100  0.47021842
## Wealth -0.154683104 -0.14859424  0.08630649 -0.23196695 -0.11219383  0.31955631
## Ineq    0.272027031  0.37483032  0.07184018 -0.02494384 -0.01390576 -0.18278697
## Prob    0.283535996 -0.56159383 -0.08598908 -0.05306898 -0.42530006 -0.08978385
## Time   -0.148203050 -0.44199877  0.19507812 -0.23551363 -0.29264326 -0.26363121
##              PC12        PC13        PC14          PC15
## M       0.16580189 -0.05142365  0.04901705  0.0051398012
## So     -0.05753357 -0.29368483 -0.29364512  0.0084369230
## Ed      0.47786536  0.19441949  0.03964277 -0.0280052040
## Po1     0.22611207 -0.18592255 -0.09490151 -0.6894155129
## Po2     0.19088461 -0.13454940 -0.08259642  0.7200270100
## LF      0.02705134 -0.27742957 -0.15385625  0.0336823193
## M.F    -0.23925913  0.31624667 -0.04125321  0.0097922075
## Pop    -0.18350385  0.12651689 -0.05326383  0.0001496323
## NW     -0.36671707  0.22901695  0.13227774 -0.0370783671
## U1     -0.09314897 -0.59039450 -0.02335942  0.0111359325
## U2      0.28440496  0.43292853 -0.03985736  0.0073618948
## Wealth -0.32172821 -0.14077972  0.70031840 -0.0025685109
## Ineq    0.43762828 -0.12181090  0.59279037  0.0177570357
## Prob    0.15567100 -0.03547596  0.04761011  0.0293376260
## Time    0.13536989 -0.05738113 -0.04488401  0.0376754405
```

##Examine the variance distribution between PCs

```r
library(GGally)
```

```
## Loading required package: ggplot2
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

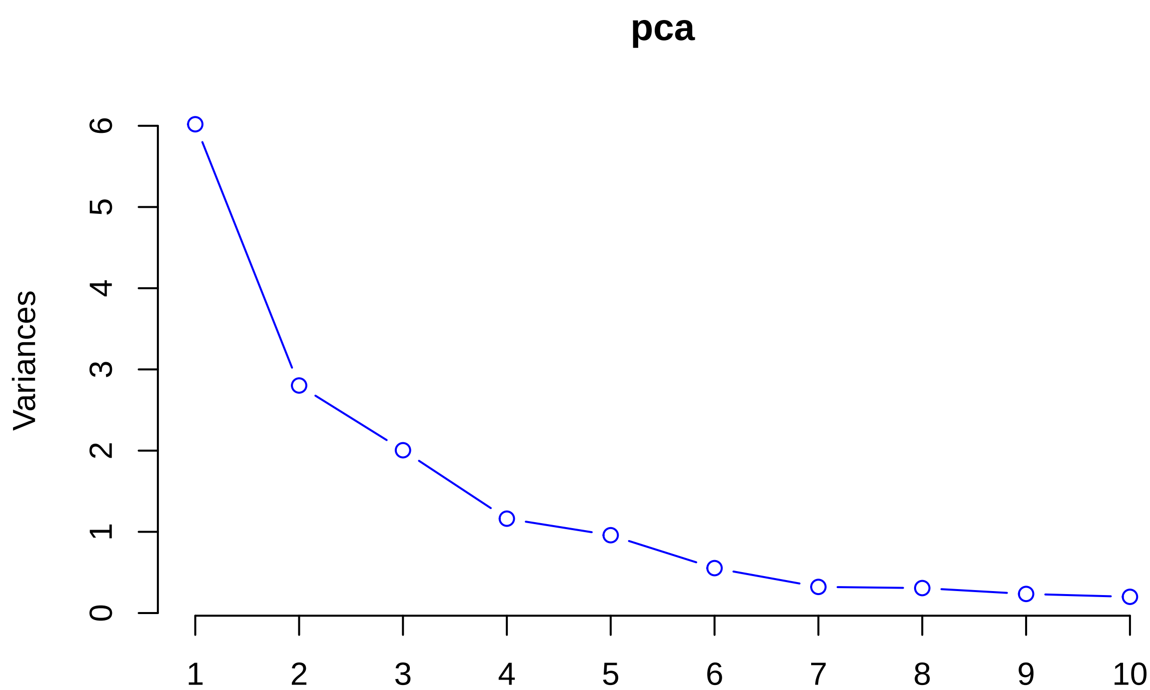
```r
summary(pca)
```

```
## 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
cor(pca$x) # no colinearity error found
```

```
##                PC1           PC2           PC3           PC4           PC5
## PC1   1.000000e+00 -1.273307e-16 -1.825724e-16  2.298165e-16 -3.391074e-16
## PC2  -1.273307e-16  1.000000e+00 -5.694249e-16  3.269637e-16 -8.335299e-16
## PC3  -1.825724e-16 -5.694249e-16  1.000000e+00  1.177395e-16 -1.906912e-16
## PC4   2.298165e-16  3.269637e-16  1.177395e-16  1.000000e+00 -9.226708e-17
## PC5  -3.391074e-16 -8.335299e-16 -1.906912e-16 -9.226708e-17  1.000000e+00
## PC6  -1.459722e-16  4.219478e-16 -7.520921e-16  1.547542e-16  6.022076e-17
## PC7   3.976873e-16  1.540007e-16  2.035710e-16 -5.123996e-16  1.854410e-16
## PC8   6.388541e-16 -6.173812e-17 -7.165046e-17 -1.070185e-15 -6.433073e-16
## PC9   2.470077e-16 -3.807073e-16 -5.893441e-17  4.569382e-16  5.766527e-16
## PC10 -8.449048e-17 -4.552839e-16 -1.456269e-16  3.273781e-16  1.209745e-16
## PC11  1.213205e-16  2.045710e-17  8.169971e-18 -8.690871e-17  1.034889e-15
## PC12  1.662919e-16  1.097279e-16 -5.546615e-16 -5.863430e-16  1.159214e-15
## PC13  1.070330e-16 -8.302804e-16  9.079977e-16  4.193459e-16  2.700256e-16
## PC14  9.443813e-16 -6.262505e-16 -5.086062e-16  1.699532e-16 -1.210316e-17
## PC15  3.677245e-15  3.390845e-15 -3.874069e-15  2.292428e-15  3.579062e-17
##                PC6           PC7           PC8           PC9          PC10
## PC1  -1.459722e-16  3.976873e-16  6.388541e-16  2.470077e-16 -8.449048e-17
## PC2   4.219478e-16  1.540007e-16 -6.173812e-17 -3.807073e-16 -4.552839e-16
## PC3  -7.520921e-16  2.035710e-16 -7.165046e-17 -5.893441e-17 -1.456269e-16
## PC4   1.547542e-16 -5.123996e-16 -1.070185e-15  4.569382e-16  3.273781e-16
## PC5   6.022076e-17  1.854410e-16 -6.433073e-16  5.766527e-16  1.209745e-16
## PC6   1.000000e+00 -2.663864e-16 -1.213255e-16  6.943245e-16  2.552376e-16
## PC7  -2.663864e-16  1.000000e+00  1.364129e-15 -6.240791e-16 -5.487255e-16
## PC8  -1.213255e-16  1.364129e-15  1.000000e+00  3.245495e-16 -1.844524e-16
## PC9   6.943245e-16 -6.240791e-16  3.245495e-16  1.000000e+00 -1.337589e-15
## PC10  2.552376e-16 -5.487255e-16 -1.844524e-16 -1.337589e-15  1.000000e+00
## PC11 -1.090780e-16  1.020271e-16 -7.028380e-16  4.432449e-16  2.883589e-16
## PC12 -2.098893e-17  3.723676e-16  4.344960e-17 -2.141621e-16 -4.547243e-16
## PC13 -8.364523e-16 -2.010077e-16 -2.310523e-16 -2.007507e-16  4.375205e-16
## PC14  3.280329e-16 -1.651300e-16 -1.885520e-16  8.629211e-16  1.261895e-16
## PC15 -2.651521e-15 -5.196469e-16 -1.627361e-16  3.828687e-15  1.382630e-16
##               PC11          PC12          PC13          PC14          PC15
## PC1   1.213205e-16  1.662919e-16  1.070330e-16  9.443813e-16  3.677245e-15
## PC2   2.045710e-17  1.097279e-16 -8.302804e-16 -6.262505e-16  3.390845e-15
## PC3   8.169971e-18 -5.546615e-16  9.079977e-16 -5.086062e-16 -3.874069e-15
## PC4  -8.690871e-17 -5.863430e-16  4.193459e-16  1.699532e-16  2.292428e-15
## PC5   1.034889e-15  1.159214e-15  2.700256e-16 -1.210316e-17  3.579062e-17
## PC6  -1.090780e-16 -2.098893e-17 -8.364523e-16  3.280329e-16 -2.651521e-15
## PC7   1.020271e-16  3.723676e-16 -2.010077e-16 -1.651300e-16 -5.196469e-16
## PC8  -7.028380e-16  4.344960e-17 -2.310523e-16 -1.885520e-16 -1.627361e-16
## PC9   4.432449e-16 -2.141621e-16 -2.007507e-16  8.629211e-16  3.828687e-15
## PC10  2.883589e-16 -4.547243e-16  4.375205e-16  1.261895e-16  1.382630e-16
## PC11  1.000000e+00  1.555555e-16  3.969289e-16 -6.800922e-16  3.893464e-16
## PC12  1.555555e-16  1.000000e+00  1.184215e-16 -1.287411e-16 -3.548408e-16
## PC13  3.969289e-16  1.184215e-16  1.000000e+00  4.443130e-16 -2.885221e-15
## PC14 -6.800922e-16 -1.287411e-16  4.443130e-16  1.000000e+00 -3.562487e-16
## PC15  3.893464e-16 -3.548408e-16 -2.885221e-15 -3.562487e-16  1.000000e+00
```

```r
##We can use the screeplot function to plot the variances of each of the principal components.
screeplot(pca,type='line',col='blue')
```



From the graph above, we can tell the marginal benefits start diminishing from PC4,as the cumulative proportion of variance for PC 1:4 accounts for 79.9%, which indicates how impactful these 4 PCs are for the whole dataset.

##Input the combined data into lm function

```
pc<-pca$x[,1:4]
crimepc<-cbind(pc,uscrime[,16])
model1<-lm(V5~.,data=as.data.frame(crimepc))
summary(model1)
```

```
##
## Call:
## lm(formula = V5 ~ ., data = as.data.frame(crimepc))
##
## 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
```

Rotate back the data to the original

```
intercept<-model1$coefficients[1]
beta<-model1$coefficients[2:5]
alpha<-pca$rotation[,1:4]%*%beta #Dot product for matrix multiplication
alpha
```

```
##              [,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
```

```
intercept
```

```
## (Intercept)
##    905.0851
```

##Unscale the coefficients and adjust the intercept

Since, mathematically, the equation is Xs=(X-Mu)/SIGMAx, we should do the following transformation: alpha_unscale= alpha/SIGMA intercept_adjusted=intercept-(alpha*Mu1/SIGMA1 + alpha*Mu2/SIGMA2 + ...+alpha*Mu15/SIGMA15)

```
alpha_unscale<-alpha/sapply(uscrime[,1:15],sd)
intercept_adjusted<-intercept-sum(alpha*sapply(uscrime[,1:15],mean)/sapply(uscrime[,1:15],sd))
alpha_unscale
```

```
##                [,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
```

```r
intercept_adjusted
```

```
## (Intercept)
##    1666.485
```

##Make prediction using the updated coef and intercept.

```r
test_data<-data.frame(CM=14.0,So=0,Ed=10.0,Po1=12.0,Po2=15.5,LF=0.64,M.F=94.0, Pop=150, NW=1.1, U1=0.12
prediction= sum(t(alpha_unscale)*test_data) + intercept_adjusted
prediction
```

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

```
##'Based on the given parameters from Q 8.2, the prediction is  1113'
```

##Comparing the models bulit with PCA and without PCA.

```r
#create a model without PCA
model_original<-lm(Crime~.,data=uscrime)
summary(model_original)
```

```
##
## Call:
## lm(formula = Crime ~ ., data = uscrime)
##
## 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
```

Model1 - With PCA: Multiple R-squared: 0.3091, Adjusted R-squared: 0.2433

Model_original- Without PCA Multiple R-squared: 0.8031, Adjusted R-squared: 0.7078

By comparing these 2 models, we found both Multiple R^2 and Adjusted R^2 of the Model PCA are lower than the model with all factors. There might be some unseen misleading information and i decided to do CV.LM.

##Use cv.lm to check if the result changes.

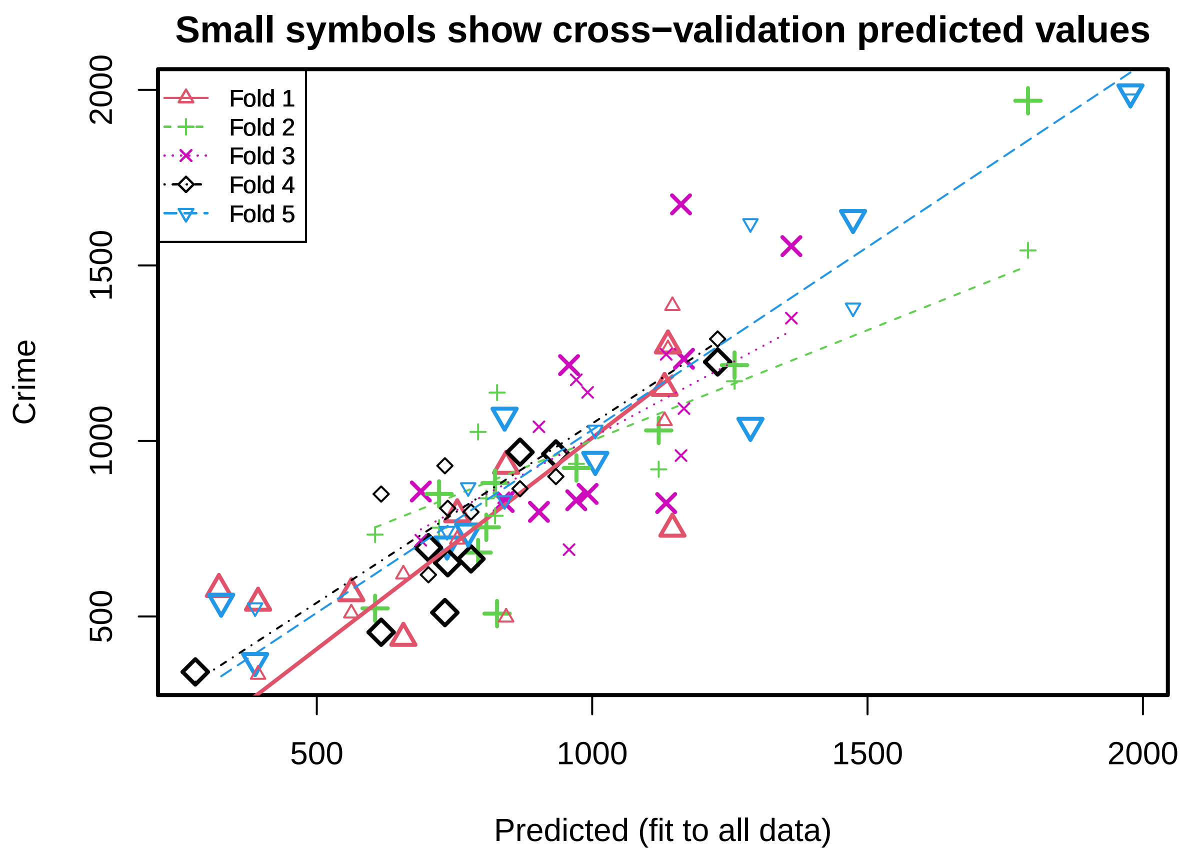
```
#first, look at the original model with cv.lm
library(DAAG)
```

```
## Loading required package: lattice
```

```
cvmodel_orignial<-cv.lm(uscrime,model_original,m=5)
```

```
## Analysis of Variance Table
##
## Response: Crime
##           Df  Sum Sq Mean Sq F value  Pr(>F)
## M          1   55084   55084    1.26  0.2702
## So         1   15370   15370    0.35  0.5575
## Ed         1  905668  905668   20.72 7.7e-05 ***
## Po1        1 3076033 3076033   70.38 1.8e-09 ***
## Po2        1  153024  153024    3.50  0.0708 .
## LF         1   61134   61134    1.40  0.2459
## M.F        1  111000  111000    2.54  0.1212
## Pop        1   42649   42649    0.98  0.3309
## NW         1   14197   14197    0.32  0.5728
## U1         1    7065    7065    0.16  0.6904
## U2         1  269663  269663    6.17  0.0186 *
## Wealth     1   34748   34748    0.79  0.3795
## Ineq       1  547423  547423   12.52  0.0013 **
## Prob       1  222620  222620    5.09  0.0312 *
## Time       1   10304   10304    0.24  0.6307
## Residuals 31 1354946   43708
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Warning in cv.lm(uscrime, model_original, m = 5):
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate
```




```
##
## fold 1
## Observations in test set: 9
##                  1   3  17  18   19   22     36    38     40
## Predicted    755.0 322 393 844 1146  657 1137.6 562.7 1131.5
## cvpred       719.5 227 334 497 1385  620 1261.6 509.1 1057.1
## Crime        791.0 578 539 929  750  439 1272.0 566.0 1151.0
## CV residual   71.5 351 205 432 -635 -181   10.4  56.9   93.9
##
## Sum of squares = 804291    Mean square = 89366    n = 9
##
## fold 2
## Observations in test set: 10
##                 4    6    12   25     28    32    34    41   44   46
## Predicted    1791  793 722.0  606 1258.5 807.8 971.5 823.7 1121  827
## cvpred       1543 1026 752.8  733 1170.1 836.6 934.6 786.7  919 1138
## Crime        1969  682 849.0  523 1216.0 754.0 923.0 880.0 1030  508
## CV residual   426 -344  96.2 -210   45.9 -82.6 -11.6  93.3  111 -630
```

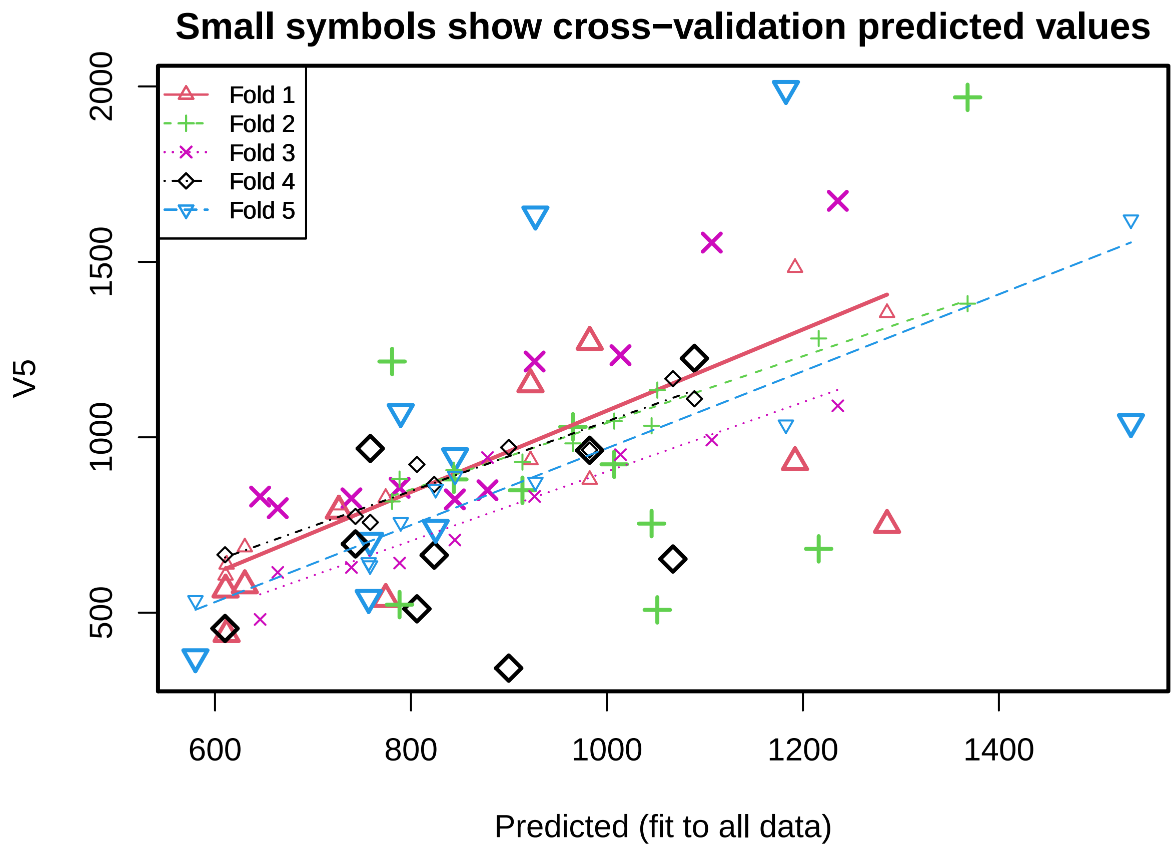
```
##
## Sum of squares = 779686    Mean square = 77969    n = 10
##
## fold 3
## Observations in test set: 10
##                 5    8   9   11   15   23   37    39   43   47
## Predicted   1167 1362 689 1161  903  958  971 839.3 1134  992
## cvpred      1092 1350 717  958 1040  690 1174 838.2 1247 1138
## Crime       1234 1555 856 1674  798 1216  831 826.0  823  849
## CV residual  142  205 139  716 -242  526 -343 -12.2 -424 -289
##
## Sum of squares = 1310071    Mean square = 131007    n = 10
##
## fold 4
## Observations in test set: 9
##                 7   13   14      20  24  27    30   35   45
## Predicted   934.2  733  780 1227.8 869 279 702.7  738  617
## cvpred      898.5  929  797 1290.4 864 227 618.7  808  849
## Crime       963.0  511  664 1225.0 968 342 696.0  653  455
## CV residual  64.5 -418 -133  -65.4 104 115  77.3 -155 -394
##
## Sum of squares = 410147    Mean square = 45572    n = 9
##
## fold 5
## Observations in test set: 9
##                 2    10     16   21     26   29   31   33  42
## Predicted   1474 736.5 1005.7  775 1977.4 1287  388  841 326
## cvpred      1380 743.3 1031.4  868 1975.1 1620  525  831 113
## Crime       1635 705.0  946.0  742 1993.0 1043  373 1072 542
## CV residual  255 -38.3  -85.4 -126   17.9 -577 -152  241 429
##
## Sum of squares = 688401    Mean square = 76489    n = 9
##
## Overall (Sum over all 9 folds)
##    ms
## 84949
```

```
#Then, the one with PCA
cvmodel_pca<-cv.lm(as.data.frame(crimepc),model1,m=5)
```

```
## Analysis of Variance Table
##
## Response: V5
##           Df  Sum Sq Mean Sq F value Pr(>F)
## PC1        1 1177568 1177568   10.40 0.0024 **
## PC2        1  633037  633037    5.59 0.0227 *
## PC3        1   58541   58541    0.52 0.4760
## PC4        1  257832  257832    2.28 0.1387
## Residuals 42 4753950  113189
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Warning in cv.lm(as.data.frame(crimepc), model1, m = 5):
```

```
##
##  As there is >1 explanatory variable, cross-validation
##  predicted values for a fold are not a linear function
##  of corresponding overall predicted values.  Lines that
##  are shown for the different folds are approximate
```




```
##
## fold 1
## Observations in test set: 9
##                   1    3   17   18   19   22   36    38   40
## Predicted     726.3  630  774 1192 1286  612  982 610.7  922
## cvpred        806.4  687  828 1483 1355  638  879 606.3  935
## V5            791.0  578  539  929  750  439 1272 566.0 1151
## CV residual -15.4 -109 -289 -554 -605 -199  393 -40.3  216
##
## Sum of squares = 1010591    Mean square = 112288    n = 9
##
## fold 2
## Observations in test set: 10
##                 4    6    12   25   28   32   34    41     44   46
## Predicted     1368 1216 913.8  788  781 1046 1007 843.8  965.3 1051
## cvpred        1381 1282 929.4  881  817 1033 1046 906.3  982.7 1134
## V5            1969  682 849.0  523 1216  754  923 880.0 1030.0  508
## CV residual    588 -600 -80.4 -358  399 -279 -123 -26.3   47.3 -626
##
```

```
## Sum of squares = 1487411    Mean square = 148741    n = 10
##
## fold 3
## Observations in test set: 10
##                  5    8   9    11  15   23  37  39  43    47
## Predicted     1014 1107 788 1236 664  926 646 739 845 878.1
## cvpred         950  992 642 1090 615  831 481 629 707 942.3
## V5            1234 1555 856 1674 798 1216 831 826 823 849.0
## CV residual    284  563 214  584 183  385 350 197 116 -93.3
##
## Sum of squares = 1149649    Mean square = 114965    n = 10
##
## fold 4
## Observations in test set: 9
##                     7   13   14   20  24   27    30   35   45
## Predicted     982.362  806  824 1089 758  900 743.3 1067  610
## cvpred        963.673  923  865 1110 757  971 774.4 1167  665
## V5            963.000  511  664 1225 968  342 696.0  653  455
## CV residual    -0.673 -412 -201  115 211 -629 -78.4 -514 -210
##
## Sum of squares = 977599    Mean square = 108622    n = 9
##
## fold 5
## Observations in test set: 9
##                  2    10    16   21   26   29   31   33   42
## Predicted      927 758.2 845.0  825 1183 1535  580  790  757
## cvpred         873 634.6 889.8  852 1036 1620  535  758  643
## V5            1635 705.0 946.0  742 1993 1043  373 1072  542
## CV residual    762  70.4  56.2 -110  957 -577 -162  314 -101
##
## Sum of squares = 1986093    Mean square = 220677    n = 9
##
## Overall (Sum over all 9 folds)
##     ms
## 140667
```

#We calculate the R^2 for the original one first.

```r
sse_org<-84949 *nrow(uscrime)
sst<-sum((uscrime$Crime-mean(uscrime$Crime))^2)
rsqa<-1-sse_org/sst
rsqa
```

```
## [1] 0.42
```

#Then, calculate the rest

```r
crimepc_data<-as.data.frame(crimepc)
sse_pca<-140667*nrow(crimepc_data)
rsqa_pca<-1-sse_pca/sst
rsqa_pca
```

```
## [1] 0.0392
```

#Compare the AIC score as well:

```
AIC(model1)
```

```
## [1] 687
```

```
AIC(model_original)
```

```
## [1] 650
```

From the R^2 and AIC comparison, we got even lower R^2 for the model built with PCA. it seems that we are not better off when using PCA.

##Try bulit model with 5 PCAs

```
pc2<-pca$x[,1:5]
crimepc2<-cbind(pc2,uscrime[,16])
model2<-lm(V6~.,data=as.data.frame(crimepc2))
summary(model2)
```

```
##
## Call:
## lm(formula = V6 ~ ., data = as.data.frame(crimepc2))
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -420.8 -185.0   12.2  146.2  447.9
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    905.1       35.6   25.43  < 2e-16 ***
## PC1             65.2       14.7    4.45  6.5e-05 ***
## PC2            -70.1       21.5   -3.26   0.0022 **
## PC3             25.2       25.4    0.99   0.3272
## PC4             69.4       33.4    2.08   0.0437 *
## PC5           -229.0       36.8   -6.23  2.0e-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:  0.645,  Adjusted R-squared:  0.602
## F-statistic: 14.9 on 5 and 41 DF,  p-value: 2.45e-08
```

R^2 for the model2 (with 5 PCAs) nearly doubled comparing with model1(with 4 PCAs), comparing these 2 models, we believe if we want to go with PCA to elimante the colinearity and other error, the model with 5 PCAs is a better option.