Homework #6

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)!)

As per usual, call the libraries I will be using (via pacman) and import the dataset (via rio). Reorganized the data, putting 'crime' in the first column, so it's easier to call the regression functions. I ran the 'prcomp()' function on the predictors (with scaled = TRUE), and printed the summary, which allows me to see the Proportion of Variance for each Principle Component (how relevent each factor is)

```
#housekeeping
library(pacman)
pacman::p_load(rio, stats, pls, DAAG)

set.seed(123)
#import data
data <- import("D…/uscrime.txt")
#swap column to fit formula (so that crime is first column)
orData <- data[c(16, 1:15)]
pred = orData[-1] #predictors
crime = orData[1]
PCA = prcomp(~ ., pred, scale = TRUE)
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
```

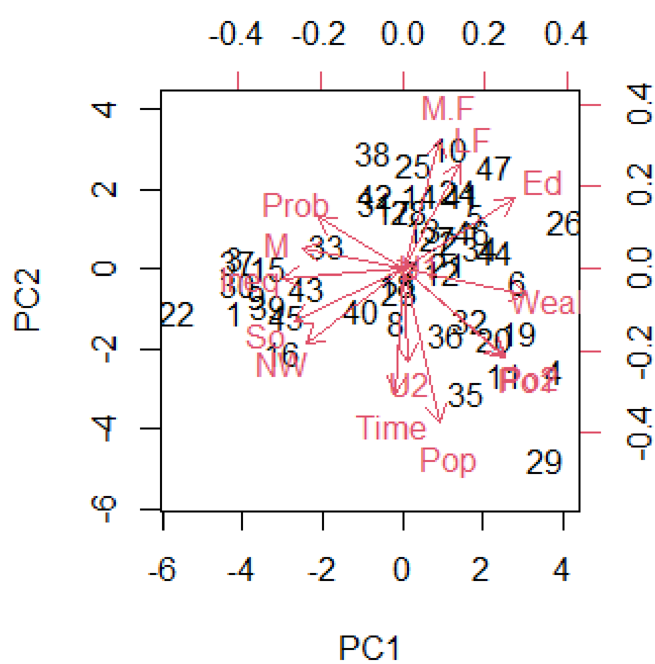
```
##                               PC15
## Standard deviation     0.06793
## Proportion of Variance 0.00031
## Cumulative Proportion  1.00000
```

```r
attributes(PCA)
```

```
## $names
## [1] "sdev"     "rotation" "center"   "scale"    "x"        "call"
##
## $class
## [1] "prcomp"
```

To help visualize the structure of the first two PC's, I can create a biplot for their Eigen vectors.

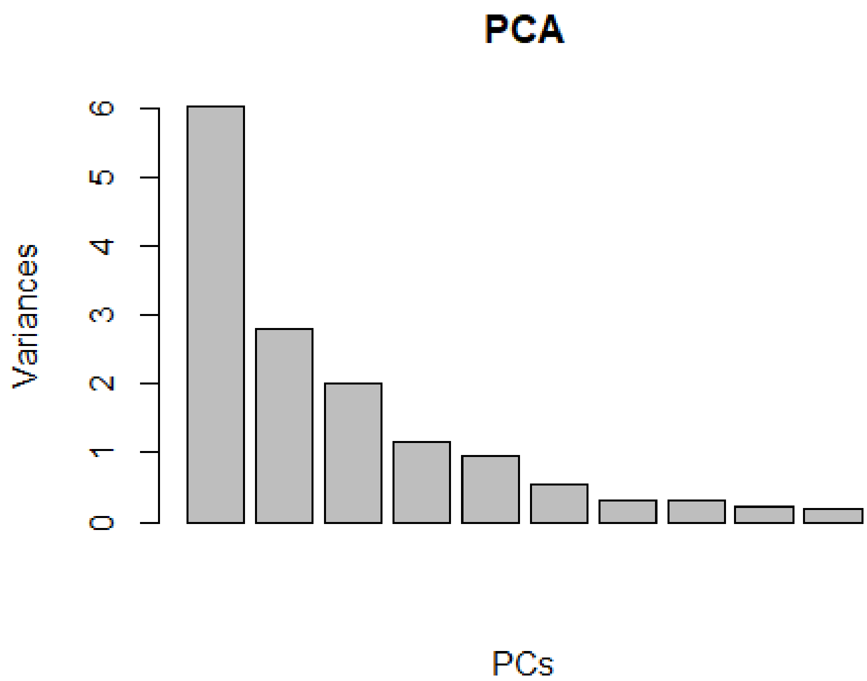
```r
biplot(PCA, scale = 0)
```



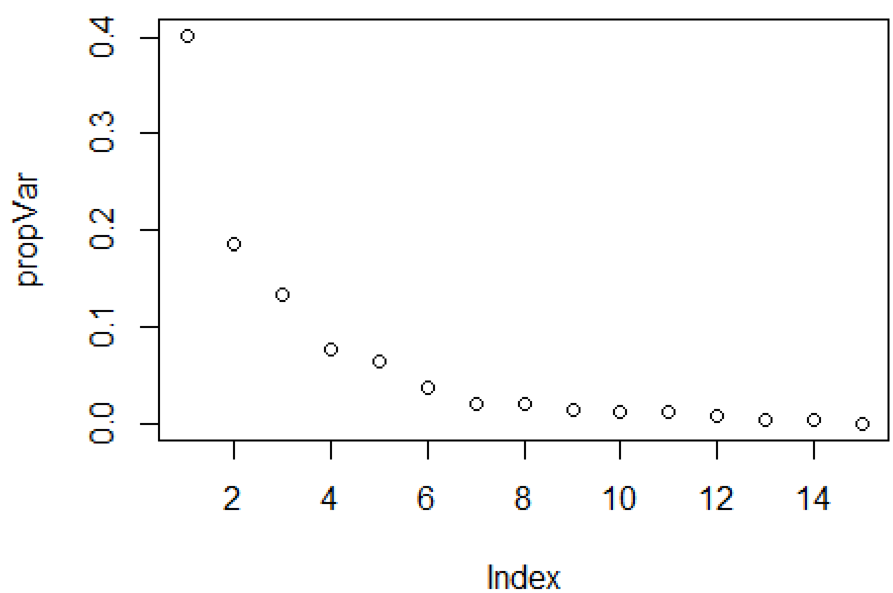
Judging from the biplot, PC1 seems to be a function with variables 'Wealth', 'Ineq', 'M', and 'So'. PC2 seems to be a function of 'Time', 'Pop', 'M.F', and 'L.F'. I've come to this conclusion because the lines that are most parallel to their respective axes have the largest variance in those scales.

I can, then, create a scree plot to chose the optimal amount of PC's to use in the model.

```r
screeplot(PCA, xlab = 'PCs')
```




```
var = PCA$sdev ^ 2 #variance
propVar = var / sum(var) #proportion of var
plot(propVar) #plots prop vs the PC number
```



Plotting the PoV VS. the PC number, there is a clear downward trend, showing diminishing returns. At this point, it's really up to the user to decide what value to go with. I chose 7 PC's, since the summary accoutns for ~92% of the values and because it looks like that's when the curve flattens.

```
x = 7 # number of pc

PCs = PCA$x[, 1:x]
PCdata = cbind(crime, PCs)

model = lm(Crime ~ ., PCdata)
summary(model)

##
## Call:
## lm(formula = Crime ~ ., data = PCdata)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -475.41 -141.65   34.73  137.25  412.32
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   905.09      34.21  26.454  < 2e-16 ***
## PC1            65.22      14.10   4.626 4.04e-05 ***
## PC2           -70.08      20.66  -3.392   0.0016 **
## PC3            25.19      24.42   1.032   0.3086
## PC4            69.45      32.08   2.165   0.0366 *
## PC5          -229.04      35.33  -6.483 1.11e-07 ***
## PC6           -60.21      46.50  -1.295   0.2029
## PC7           117.26      60.96   1.923   0.0617 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 234.6 on 39 degrees of freedom
## Multiple R-squared:  0.6882, Adjusted R-squared:  0.6322
## F-statistic:  12.3 on 7 and 39 DF,  p-value: 3.513e-08

variables = 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,
```

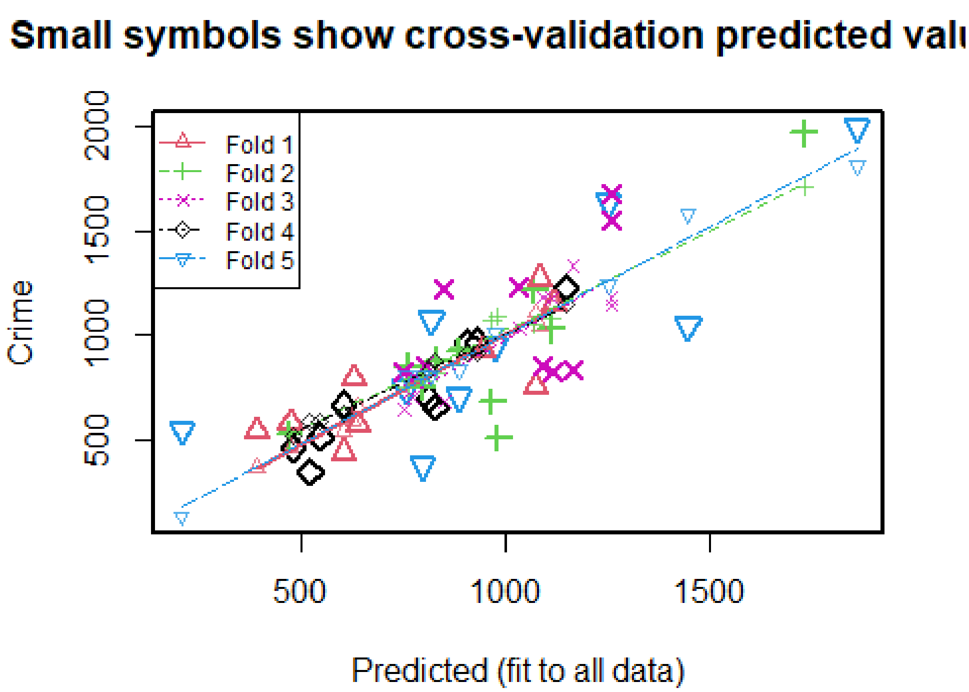
```r
  Wealth = 3200,
  Ineq = 20.1,
  Prob = 0.04,
  Time = 39.0
)

model$coefficients #coeff in pca

## (Intercept)          PC1          PC2          PC3          PC4          PC5
##   905.08511     65.21593    -70.08312     25.19408     69.44603   -229.04282
##         PC6          PC7
##   -60.21329    117.25590

betas = model$coefficients[-1]
beta0 = model$coefficients[1]


#convert the coefficients back into the de-scaled space
alphas = PCA$rotation[, 1:x] %*% betas

p_mean = sapply(pred, mean)
p_sd = sapply(pred, sd)
a_orig = alphas / p_sd
a_orig  #de-scaled coefficients

##                 [,1]
## M       5.523735e+01
## So      1.397571e+02
## Ed     -6.803836e+00
## Po1     4.458638e+01
## Po2     4.642432e+01
## LF      6.733809e+02
## M.F     4.440293e+01
## Pop     9.599076e-01
## NW      5.684940e+00
## U1     -1.027735e+03
## U2      2.441589e+01
## Wealth  2.883565e-02
## Ineq    1.245113e+01
## Prob   -5.170569e+03
## Time   -2.215095e+00

a0 = beta0 - sum(alphas * p_mean / p_sd)
a0 #de-scaled intercept

## (Intercept)
##   -5498.458

prediction = a0 + sum(a_orig * variables)  #equation of regression line
prediction
```

```
## (Intercept)
##    1230.418
```

In the snippet above, I ran the lm() function the first 7 PC's, de-scaled the coefficients, and inserted the new city's pre-defined variables' values. This gave a predicted crime rate of 1230 with an Adjusted R-Squared value of 0.6322. Looking back to question 8.2 from the previous homework, the predicted crime rate was 1304 and the R-Squared value was 0.7307 (with equation Crime ~ M + Ed + Ineq + Prob + U2 + Po1). This suggests that there might be a correlation in the data that is too high for the PCA model to overcome and that removing multiple predictors as in the last HW yields a better model. I can use cross-validation:

```
PClist = as.data.frame(PCA$x[, 1:x])
PCcv = cbind(crime, PClist)
model2 = lm(Crime ~ ., PCcv)
cv = cv.lm(PCcv, model2, m = 5)
```

```
## Warning in cv.lm(PCcv, model2, 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
## Predicted   628.7597 475.2375 394.7130 948.32205 1074.2348 604.7846
1085.223
## cvpred      590.9906 459.6297 365.7962 953.53166 1108.1353 538.2655
1047.276
## Crime       791.0000 578.0000 539.0000 929.00000  750.0000 439.0000
1272.000
## CV residual 200.0094 118.3703 173.2038 -24.53166 -358.1353 -99.2655
224.724
##                     38         40
## Predicted   641.38459 1121.75684
## cvpred      654.01612 1139.85653
## Crime       566.00000 1151.00000
## CV residual -88.01612   11.14347
##
## Sum of squares = 281103.1    Mean square = 31233.68    n = 9
##
## fold 2
## Observations in test set: 10
##                      4         6        12          25        28        32
## Predicted   1732.1969  969.5473 762.0269 472.536843 1072.9914 798.65545
## cvpred      1714.2867 1068.6814 746.9762 529.071468 1049.1614 777.72986
## Crime       1969.0000  682.0000 849.0000 523.000000 1216.0000 754.00000
## CV residual  254.7133 -386.6814 102.0238  -6.071468  166.8386 -23.72986
##                     34        41         44        46
## Predicted   888.26654 834.71933 1113.66782  983.4052
## cvpred      943.70217 819.76136 1076.79097 1086.5979
## Crime       923.00000 880.00000 1030.00000  508.0000
## CV residual -20.70217  60.23864  -46.79097 -578.5979
##
## Sum of squares = 594267.5    Mean square = 59426.75    n = 10
##
## fold 3
## Observations in test set: 10
##                      5         8          9        11        15         23
## Predicted   1036.3192 1261.9257 806.077864 1261.6800 775.01424  852.3868
## cvpred      1028.2487 1143.8494 854.810335 1174.0922 701.36874  685.1514
## Crime       1234.0000 1555.0000 856.000000 1674.0000 798.00000 1216.0000
## CV residual  205.7513  411.1506   1.189665  499.9078  96.63126  530.8486
##                     37       39        43        47
## Predicted   1167.0391 753.3714 1116.9070 1095.4323
## cvpred      1332.3513 644.4387 1176.3109 1181.5807
## Crime        831.0000 826.0000  823.0000  849.0000
## CV residual -501.3513 181.5613 -353.3109 -332.5807
##
## Sum of squares = 1272182    Mean square = 127218.2    n = 10
##
## fold 4
## Observations in test set: 9
```

```
##                         7         13         14         20         24         27
## Predicted   909.88199 547.63861 606.52255 1150.4003 933.01752  524.3022
## cvpred      917.32253 588.34528 621.68272 1144.7724 910.82539  582.3591
## Crime       963.00000 511.00000 664.00000 1225.0000 968.00000  342.0000
## CV residual  45.67747 -77.34528  42.31728   80.2276  57.17461 -240.3591
##                        30         35        45
## Predicted    813.5090  829.6430 481.84678
## cvpred       832.3003  876.8388 519.08221
## Crime        696.0000  653.0000 455.00000
## CV residual -136.3003 -223.8388 -64.08221
##
## Sum of squares = 150125.5    Mean square = 16680.61    n = 9
##
## fold 5
## Observations in test set: 9
##                         2        10          16         21         26         29
## Predicted   1253.7618  887.3683  977.18147 757.09445 1861.5139 1449.2364
## cvpred      1239.5313  836.5023 1012.99149 807.24919 1815.8644 1581.5253
## Crime       1635.0000  705.0000  946.00000 742.00000 1993.0000 1043.0000
## CV residual  395.4687 -131.5023  -66.99149 -65.24919  177.1356 -538.5253
##                        31        33        42
## Predicted    798.1198  821.0790 208.2992
## cvpred       805.9805  805.5864 129.9378
## Crime        373.0000 1072.0000 542.0000
## CV residual -432.9805  266.4136 412.0622
##
## Sum of squares = 932063.8    Mean square = 103562.6    n = 9
##
## Overall (Sum over all 9 folds)
##      ms
## 68717.9
```

```r
mn = mean(crime[, 1])
R2 = 1 - attr(cv, "ms") * nrow(orData) / sum((crime - mn) ^ 2)
R2
```

```
## [1] 0.5306241
```

I can see an R-Squared value of 0.5306241 for the PCA when using the first 7 PC's (compared to the 0.419759 from the CV model ran on all 15 predictors). And when reducing the number of predictors to the above formula (Crime ~ M + Ed + Ineq + Prob + U2 + Po1), I am given an R-Squared value of 0.638, which again shows that removing predictors for regression models is superior.