Principal Component Analysis

```r
# Clear environment
rm(list = ls())


library(tidyverse)
library(DAAG)
library(boot)
library(GGally)

#setwd("/Users/Ryan/Desktop/DS")

crime_data = read.table("uscrime.txt.",
                        sep="",
                        fill=FALSE,
                        strip.white=TRUE,
                        header = TRUE)

#test data
crime_test <- 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 = 0.04,
                         Time = 39.0)
```

Loading in the four packages that will be used throughout the problem. Next is setting the working directory and reading in the crime data that was give to us. The crime test data is the information about the city which we are trying to predict the crime rate for with PCA. Once we build the model, we are trying to predict the crime rate given those values about the city and then see how well our model does compared to the cross validated model.

Cross Validation

```r
lm_model <- lm(Crime ~ M + Ed + Po1 + U2 + Ineq + Prob,
               data = crime_data)

summary(lm_model)
```

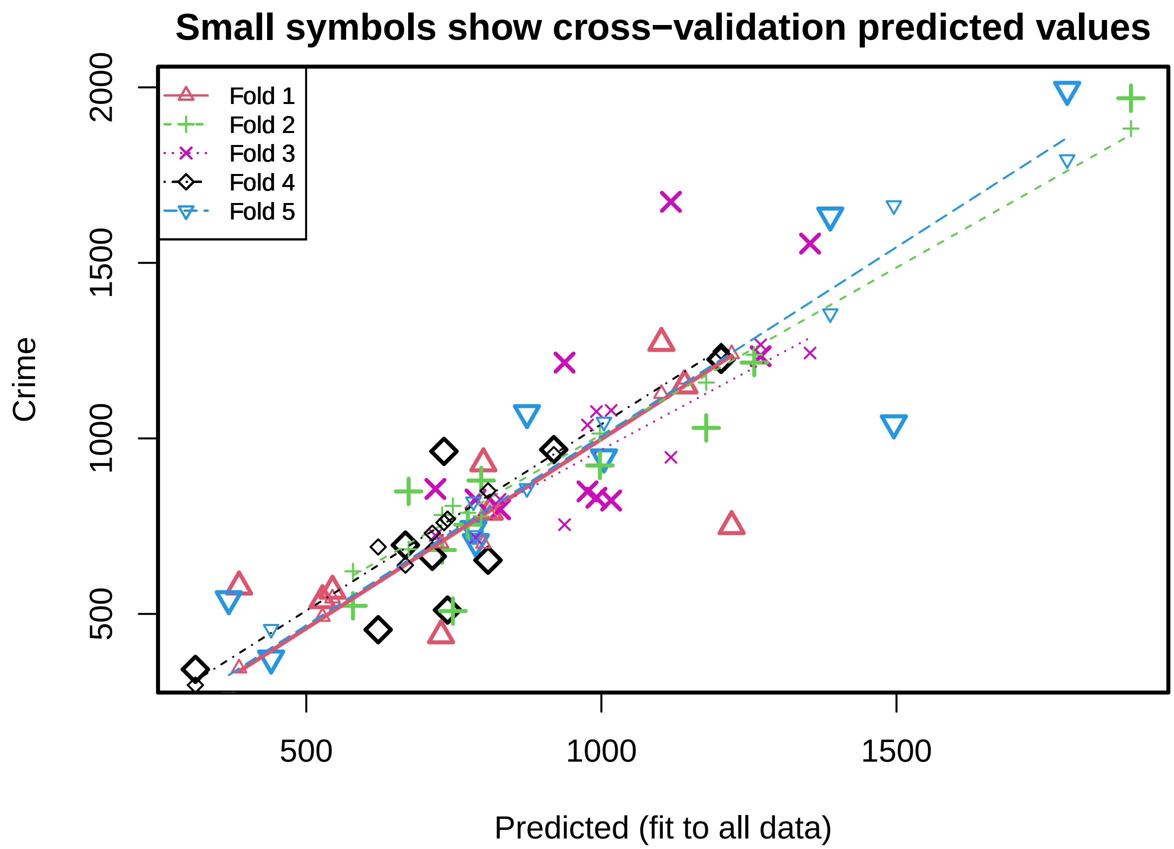
```
##
## Call:
## lm(formula = Crime ~ M + Ed + Po1 + U2 + Ineq + Prob, data = crime_data)
```

```
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -470.68  -78.41  -19.68  133.12  556.23
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5040.50     899.84  -5.602 1.72e-06 ***
## M             105.02      33.30   3.154  0.00305 **
## Ed            196.47      44.75   4.390 8.07e-05 ***
## Po1           115.02      13.75   8.363 2.56e-10 ***
## U2             89.37      40.91   2.185  0.03483 *
## Ineq           67.65      13.94   4.855 1.88e-05 ***
## Prob        -3801.84    1528.10  -2.488  0.01711 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 200.7 on 40 degrees of freedom
## Multiple R-squared:  0.7659, Adjusted R-squared:  0.7307
## F-statistic: 21.81 on 6 and 40 DF,  p-value: 3.418e-11
```

```r
#cross validate
cv_model <- cv.lm(crime_data, lm_model, m=5)
```

```
## Analysis of Variance Table
##
## Response: Crime
##           Df  Sum Sq Mean Sq F value  Pr(>F)
## M          1   55084   55084    1.37 0.24914
## Ed         1  725967  725967   18.02 0.00013 ***
## Po1        1 3173852 3173852   78.80 5.3e-11 ***
## U2         1  217386  217386    5.40 0.02534 *
## Ineq       1  848273  848273   21.06 4.3e-05 ***
## Prob       1  249308  249308    6.19 0.01711 *
## Residuals 40 1611057   40276
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
## Warning in cv.lm(crime_data, lm_model, 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   810.83 386 527.4 800 1221  728 1102 544.4 1140.8
## cvpred      785.36 345 492.2 701 1240  702 1127 544.7 1168.2
## Crime       791.00 578 539.0 929  750  439 1272 566.0 1151.0
## CV residual   5.64 233  46.8 228 -490 -263  145  21.3  -17.2
##
## Sum of squares = 439507    Mean square = 48834    n = 9
##
## fold 2
## Observations in test set: 10
##                  4     6  12    25     28  32     34  41   44   46
## Predicted   1897.2 730.3 673 579.1 1259.0 774  997.5 796 1178  748
## cvpred      1882.7 781.8 684 621.4 1238.3 788 1013.9 778 1159  808
## Crime       1969.0 682.0 849 523.0 1216.0 754  923.0 880 1030  508
## CV residual   86.3 -99.8 165 -98.4  -22.3 -34  -90.9 102 -129 -300
##
## Sum of squares = 181038    Mean square = 18104    n = 10
##
## fold 3
## Observations in test set: 10
##                  5    8   9   11    15   23   37  39   43   47
## Predicted   1269.8 1354 719 1118 828.3  938  992 787 1017  976
## cvpred      1266.8 1243 724  946 826.3  754 1077 717 1080 1038
```

```
## Crime       1234.0 1555 856 1674 798.0 1216  831 826  823  849
## CV residual  -32.8  312 132  728 -28.3  462 -246 109 -257 -189
##
## Sum of squares = 1033612    Mean square = 103361    n = 10
##
## fold 4
## Observations in test set: 9
##                7   13    14     20    24    27    30   35   45
## Predicted   733  739 713.6 1203.0 919.4 312.2 668.0  808  622
## cvpred      760  770 730.1 1247.9 953.7 297.2 638.9  851  691
## Crime       963  511 664.0 1225.0 968.0 342.0 696.0  653  455
## CV residual 203 -259 -66.1  -22.9  14.3  44.8  57.1 -198 -236
##
## Sum of squares = 213398    Mean square = 23711    n = 9
##
## fold 5
## Observations in test set: 9
##                 2    10   16    21   26   29    31   33  42
## Predicted   1388 787.3 1004 783.3 1789 1495 440.4  874 369
## cvpred      1356 723.7 1047 819.7 1795 1664 456.6  858 261
## Crime       1635 705.0  946 742.0 1993 1043 373.0 1072 542
## CV residual  279 -18.7 -101 -77.7  198 -621 -83.6  214 281
##
## Sum of squares = 650990    Mean square = 72332    n = 9
##
## Overall (Sum over all 9 folds)
##    ms
## 53586
```

```r
# We can calculate the R-squared values directly.
# R-squared = 1 - SSEresiduals/SSEtotal
# total sum of squared differences between data and its mean
sse <- 48203 * nrow(crime_data)
## total sum of squares
sst <- sum((crime_data$Crime - mean(crime_data$Crime))^2)
# mean squared error
rsq <- 1 - sse / sst
rsq
```

```
## [1] 0.671
```

```r
predict1 <- predict(lm_model, crime_test)
predict1 #1304
```

```
##    1
## 1304
```

```r
AIC(lm_model)
```

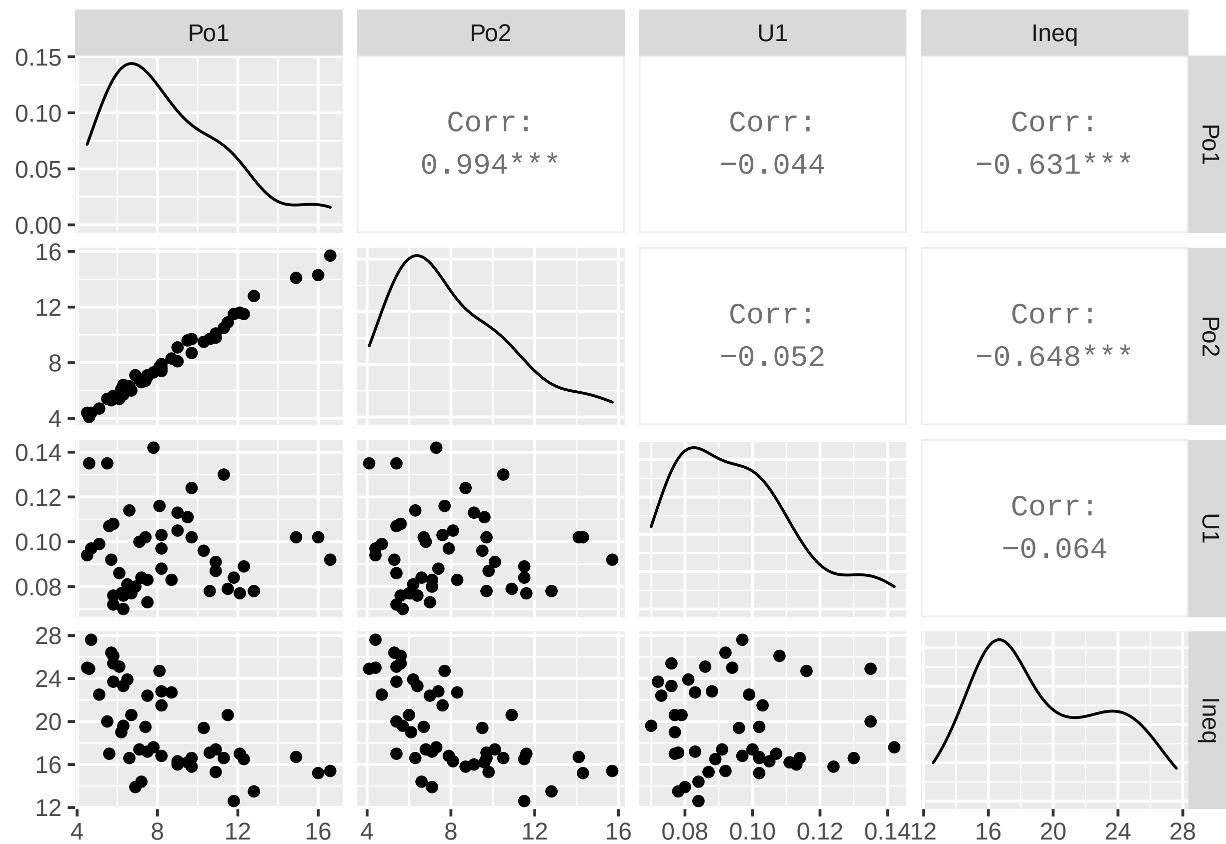
```
## [1] 640
```

The R-squared for the model is 0.671. This was the best performing model that I made from last week. We are still potentially overfitting with this model though so it'll be interesting to see the new model. The second test of fit is the AIC value which models with lower values are more accurate. Again, this model had the lowest value for AIC so I would say that it is the best of the lm models for predicting crime rate.

PCA

Using the same crime data set as the previous question, I will apply Principal Component Analysis and then create a regression model using the first few principal components. Then after that I will compare the quality of the PCA model with the cross validate model.

```
#Correlation in the data
ggpairs(crime_data, columns = c('Po1', 'Po2', 'U1', 'Ineq'))
```



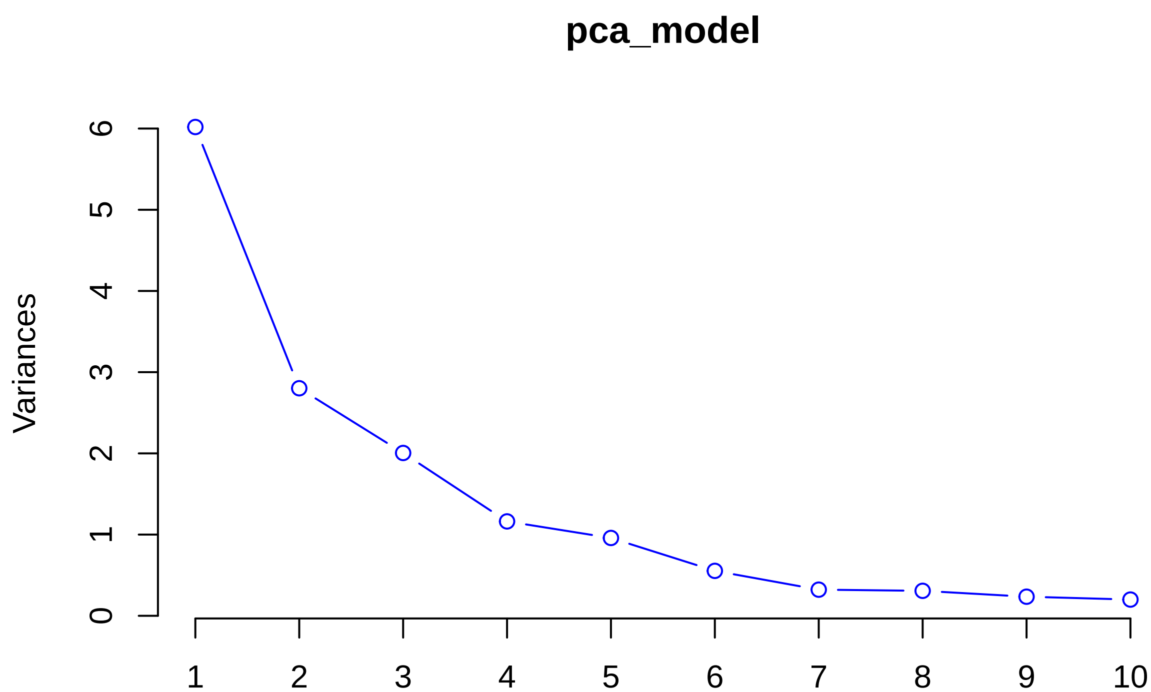

```
pca_model <- prcomp(crime_data[,1:15], scale = TRUE)
summary(pca_model)
```

```
## Importance of components:
##                           PC1   PC2   PC3    PC4    PC5    PC6    PC7    PC8
## Standard deviation     2.453 1.674 1.416 1.0781 0.9789 0.7438 0.5673 0.5544
## Proportion of Variance 0.401 0.187 0.134 0.0775 0.0639 0.0369 0.0214 0.0205
## Cumulative Proportion  0.401 0.588 0.722 0.7992 0.8631 0.9000 0.9214 0.9419
##                            PC9   PC10   PC11    PC12    PC13   PC14    PC15
## Standard deviation     0.4849 0.4471 0.4191 0.35804 0.26333 0.2418 0.06793
```

```
## Proportion of Variance 0.0157 0.0133 0.0117 0.00855 0.00462 0.0039 0.00031
## Cumulative Proportion  0.9576 0.9709 0.9826 0.99117 0.99579 0.9997 1.00000
```

```r
#a lot of variance is from the first 5 predictors

#plot the variances of each of the principal component
screeplot(pca_model, type = 'lines', col = 'blue')
```



The GGpairs function was shown during the Monday lecture and so I used it also to look at the correlation between the predictors. From the graph it is clear that Po1 and Po2 have a strong correlation between the two. Ineq had a strong correlation with Wealth, Po1, and Po2 so that could be problematic. Lets run the pca function on the dataset but make sure you don't include crime data and scale it. Screeplot was shown also during the Monday lecture, which it plots the variances of each of the principal components. From the graph it is obvious that first principal component has the biggest varinance and then pc 2, pc3, pc4, . . . So, from my model I am going to choose the first 5 predictors since they account for the majority of the variance.

```r
#obtain the 5 principal components from result matrix
#since they composed of the most variance
principal_comp <- pca_model$x[,1:5]
# now create a new matrix with components and crime response
pca_matrix <- cbind(principal_comp, crime_data[,16])

#lm model using first 5 pc
pca_lm_model <- lm(V6 ~.,
                   data = as.data.frame(pca_matrix))
summary(pca_lm_model)
```

```
##
## Call:
## lm(formula = V6 ~ ., data = as.data.frame(pca_matrix))
##
## 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
```

First let's make a matrix of the first five principal components that we will use to make our model. In order to make our model we also need the crime rate data so let's combine our matrix with that column. Then we can run the Linear Regression model on the pca which will hopefully make a more accurate prediction that the LM model from last week. Looking at the summary, we see that pca model has a lower R^2 but let's compute a more accurate R^2 that isnt scaled.

```
k = 5

#beta zero or the intercept
intercept <- pca_lm_model$coefficients[1]
intercept
```

```
## (Intercept)
##         905
```

```
#betas; slopes from scaled PCA regression
betas <- pca_lm_model$coefficients[2:(1+k)]
betas
```

```
##    PC1    PC2    PC3    PC4    PC5
##   65.2  -70.1   25.2   69.4 -229.0
```

```
#pca_model$roatation is the matrix of eigenvectors
#a_j=b_k*v_jk
#b:coefficients
#v: rotation matrix
#j: original factors
#k: principal components
# %*% matrix multiplication
```

```r
alpha <- pca_model$rotation[,1:k]%*%betas

#unscale alpha by dividng by the scale
alpha_unscaled <- alpha/pca_model$scale
alpha_unscaled
```

```
##            [,1]
## M       4.84e+01
## So      7.90e+01
## Ed      1.78e+01
## Po1     3.95e+01
## Po2     3.99e+01
## LF      1.89e+03
## M.F     3.67e+01
## Pop     1.55e+00
## NW      9.54e+00
## U1      1.59e+02
## U2      3.83e+01
## Wealth  3.72e-02
## Ineq    5.54e+00
## Prob   -1.52e+03
## Time    3.84e+00
```

```r
beta0_unscaled <- intercept - sum(alpha*pca_model$center/pca_model$scale)#unscaled intercept
beta0_unscaled
```

```
## (Intercept)
##       -5934
```

```r
#model y = ax + b
y <-  as.matrix(crime_data[,1:15])%*%alpha_unscaled + beta0_unscaled
#Calculate the R^2 error using the equations from last week
rss2 <- sum((y - crime_data[,16]) ^ 2)  ## residual sum of squares
rsq2 <- 1 - rss2/sst
rsq2 # R-squared of PCA
```

```
## [1] 0.645
```

```r
AIC(pca_lm_model)
```

```
## [1] 658
```

All of this above code relates to unscaling the data and computing the R^2 of the pca model. The equation a_j=b_k*v_jk is used to compute the aplha. We need to take the eigenvectors and times by the beta to get our alpha. Then we want to unscale the alpha so we can use it to compute the R^2. We have to unscale the beta in order to compute our model y=mx+b. Once we have the unscaled and calculate the y then we can compute the R^2. Our R^2 is 0.645 which is lower than our previous 0.671 but cross validated model could have overfitted the data. Our AIC score is worse than CV model at 658 and previously it was 640. So, our lm model using PCA did predict worse than CV model.

```r
#Predict crime rate with pca model
pred_df <- data.frame(predict(pca_model, crime_test))
predict2 <- predict(pca_lm_model, pred_df)
predict2
```

```
##    1
## 1389
```

Finally, we are able to predict our crime rate which resulted in 1389. Last week we got 1304 so a little bit lower than our PCA model. It looks like scaling the data and using the principle component analysis might have gave us a more accurate predicion than standard lm. Even though our accuracy is lower it seems that we are not overfitting the data like CV model. This test was on a pretty small sample size so it would be interesting to see the results of the scaling with a large data set.