# hw6

*Kunle Lawal, Anubhav Rana, Mihir Tulpule, Ali Mujtaba Lakdawala*

```
crime_data <- read.table("uscrime.txt", header = TRUE)

orig_crimemod <- lm(Crime ~ ., data = crime_data)
summary(orig_crimemod)
```

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

We conduct a PCA of our crime data in order to reduce the variables used in our analysis.

```
crimepca <- prcomp(crime_data[,-16], scale.=TRUE, center = TRUE)

summary(crimepca)
```

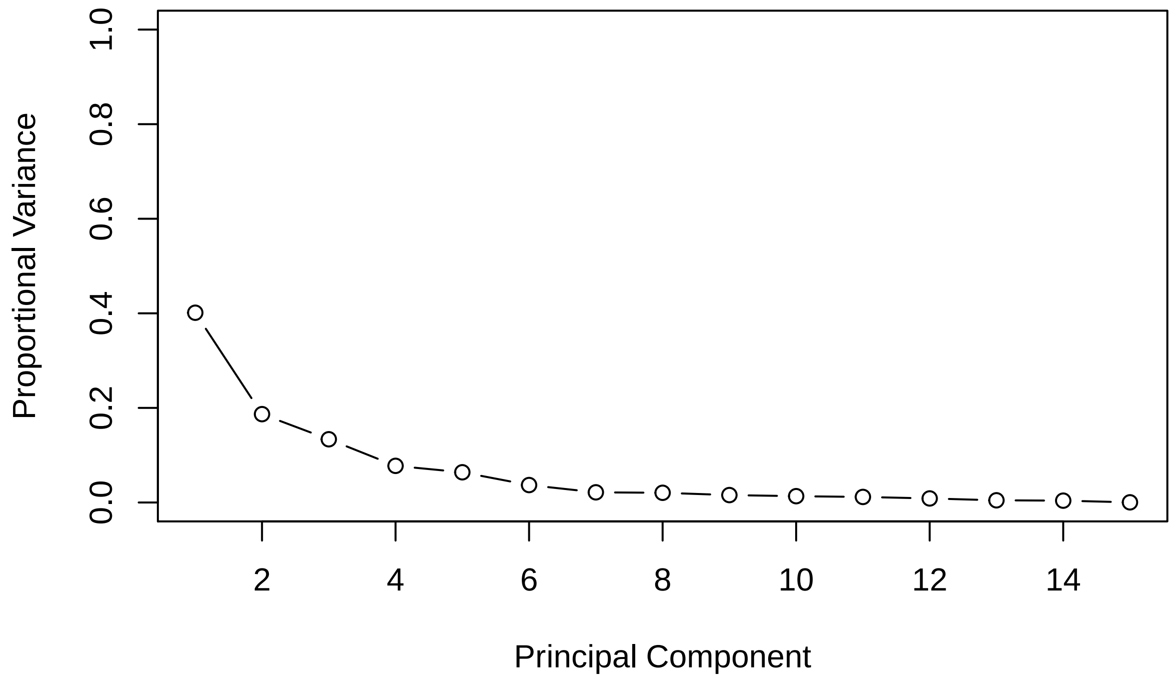
```
## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6
## Standard deviation     2.4534 1.6739 1.4160 1.07806 0.97893 0.74377
## Proportion of Variance 0.4013 0.1868 0.1337 0.07748 0.06389 0.03688
## Cumulative Proportion  0.4013 0.5880 0.7217 0.79920 0.86308 0.89996
##                            PC7     PC8     PC9    PC10    PC11    PC12
```

```
## Standard deviation     0.56729 0.55444 0.48493 0.44708 0.41915 0.35804
## Proportion of Variance 0.02145 0.02049 0.01568 0.01333 0.01171 0.00855
## Cumulative Proportion  0.92142 0.94191 0.95759 0.97091 0.98263 0.99117
##                           PC13   PC14    PC15
## Standard deviation     0.26333 0.2418 0.06793
## Proportion of Variance 0.00462 0.0039 0.00031
## Cumulative Proportion  0.99579 0.9997 1.00000
```

From our crime pca model we would like to choose the most important components. For this we plot the variances.

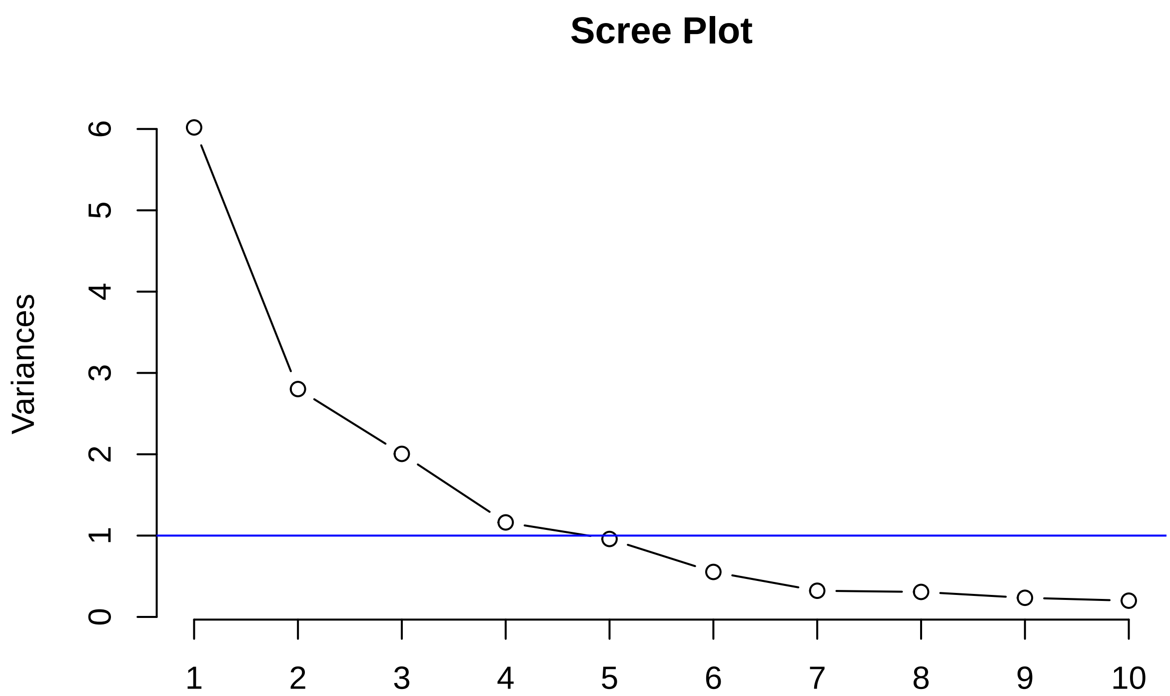
```r
variance <- crimepca$sdev^2

prop_variance <- variance/sum(variance)
plot(prop_variance,
     xlab = "Principal Component",
     ylab = "Proportional Variance",
     ylim = c(0,1) , type= "b")
```




The variance plot does not confirm certainly which is the best choice between 4,5 or 6 components. We investigated further using a screeplot which provides additional insight into the best number of components to chose.

```r
screeplot(crimepca, main = "Scree Plot", type = "line")
abline(h=1, col="blue")
```




From the screeplot we can identify that the best number of components is 5. Thus we continue to build our new linear model using the top 5 Principle Components.

```
# We first gather the components we need.
comb_crime <- cbind(crimepca$x[,1:4],crime_data[,16])
colnames(comb_crime) <- c("PC1", "PC2", "PC3", "PC4", "Crime")

# Then create a model using those components.
pcacrimemod <- lm(Crime ~., data = as.data.frame(comb_crime))
summary(pcacrimemod)
```

```
##
## Call:
## lm(formula = Crime ~ ., data = as.data.frame(comb_crime))
##
## 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
```

Now that we have our linear model in terms of the components of the PCA we would like to explain it back in terms of the regression coefficents of our original model.

For this calculation we need the following:

```
# Matrix of Eigenvectors:
beta <- pcacrimemod$coefficients[2:5]
VBeta <- crimepca$rotation[1:4] * beta
VBeta
```

```
##       PC1       PC2       PC3       PC4
## -19.80686  23.18919   8.55645  21.43341
```

```
beta0 <- pcacrimemod$coefficients[1]
beta0
```

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

```
adjustedBeta0=beta0+t(as.matrix(crimepca$center))%*%(VBeta/crimepca$scale)
```

```
## Warning in VBeta/crimepca$scale: longer object length is not a multiple of
## shorter object length
```

```
adjustedBeta0
```

```
##          [,1]
## [1,] 1654.785
```

```
adjustedbeta=VBeta/crimepca$scale
```

```
## Warning in VBeta/crimepca$scale: longer object length is not a multiple of
## shorter object length
```

```
adjustedbeta
```

```
##             M            So            Ed           Po1           Po2
##  -15.76021133   48.41418685    7.64856641    7.21203049   -7.08366327
##            LF           M.F           Pop            NW            U1
##  573.82212615    2.90370381    0.56298255   -1.92619703 1286.23177193
##            U2        Wealth          Ineq          Prob          Time
##   10.13143195    0.02221288   -4.96461461 1019.88922734    1.20736231
```

```
X = crime_data[,-ncol(crime_data)]
y_hat = as.matrix(X) %*% as.matrix(adjustedbeta) + adjustedBeta0[1,1]
```

Now that we have our converted coefficients we can use that to calculate the prediction of our new data point.

```
new_data_point = 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)

crime_prediction <- sum(
  adjustedbeta[1] %*% new_data_point$M,
  adjustedbeta[2] %*% new_data_point$So,
  adjustedbeta[3] %*% new_data_point$Ed,
  adjustedbeta[4] %*% new_data_point$Po1,
  adjustedbeta[5] %*% new_data_point$Po2,
  adjustedbeta[6] %*% new_data_point$LF,
  adjustedbeta[7] %*% new_data_point$M.F,
  adjustedbeta[8] %*% new_data_point$Pop,
  adjustedbeta[9] %*% new_data_point$NW,
  adjustedbeta[10] %*% new_data_point$U1,
  adjustedbeta[11] %*% new_data_point$U2,
  adjustedbeta[12] %*% new_data_point$Wealth,
  adjustedbeta[13] %*% new_data_point$Ineq,
  adjustedbeta[14] %*% new_data_point$Prob,
  adjustedbeta[15] %*% new_data_point$Time,
  adjustedBeta0
)

# Our manual prediction of crime.
crime_prediction
```

```
## [1] 2459.895
```

We get a prediction of 2459 which is much higher than our prediction of 1155 from the linear regression model. As a result of the r-squared of the PCA model is 0.24 which is super low compared to the original linear regression model. Showing that the original linear model was superior.

We can use another method to find the prediction as well rather than calculating it manually.

For this method we can convert the data point into its predicted pca components and then use that new pca data point in our pca linear model. The benefit of this method would be that we are able to access the prediction interval as well.

```
pred_df <- data.frame(predict(crimepca, new_data_point))
head(pred_df)
```

```
##        PC1       PC2      PC3       PC4       PC5      PC6        PC7
## 1 1.224044 -2.767641 0.533605 -1.146837 -1.206098 2.333343 -0.1535916
##         PC8      PC9       PC10       PC11     PC12      PC13      PC14
## 1 -1.391625 1.460274 -0.4525158 -0.3466498 1.663782 -1.811307 -2.174071
##       PC15
## 1 1.288675
```

```
pred <- predict(pcacrimemod, pred_df, interval = 'prediction')
pred
```

```
##        fit      lwr      upr
## 1 1112.678 396.1274 1829.228
```

The prediction is 1112. Which is similar to the original prediction and confidence interval.