# ISyE 6501 HW6

September 17, 2019

## Question 9.1

### Conclusion

- Below is a lot of code and output, so we'll put the conclusion down first. Our new model expressed in the original variables is:
  -16.93076M + 21.34368So + 12.82972Ed + 21.35216Po1 + 23.08832Po2 -346.5657LF
  -8.293097M.F + 1.046216Pop + 1.500994NW -1509.935U1 + 1.688367U2 + 0.0400119Wealth
  -6.902022Ineq + 144.9493Prob -0.9330765Time + 1666.485
- Also, the PCA with 4 principal components performed much worse than a simple linear regression (R-squared values of 0.2433 vs 0.7078, respectively).

### Procedure

- First let's import the data.

```
rm(list = ls())
set.seed(123)

crime_data = read.table("uscrime.txt", header = TRUE, stringsAsFactors = FALSE)
crime_mod1 = lm(Crime ~ ., data = crime_data)
summary(crime_mod1)
```

```
##
## 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
```

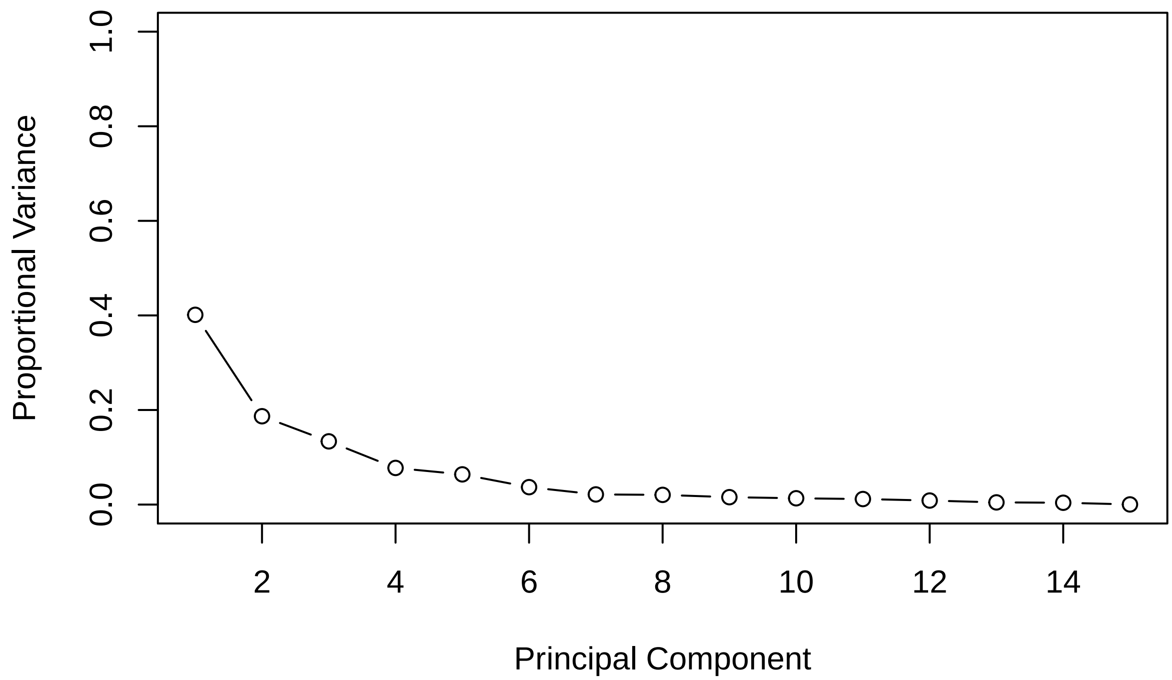
- Then let's run a pca analysis and summarize the analysis:

```
crime_pca = prcomp(~., crime_data[,-16], scale.=TRUE, center = TRUE)
summary(crime_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
```

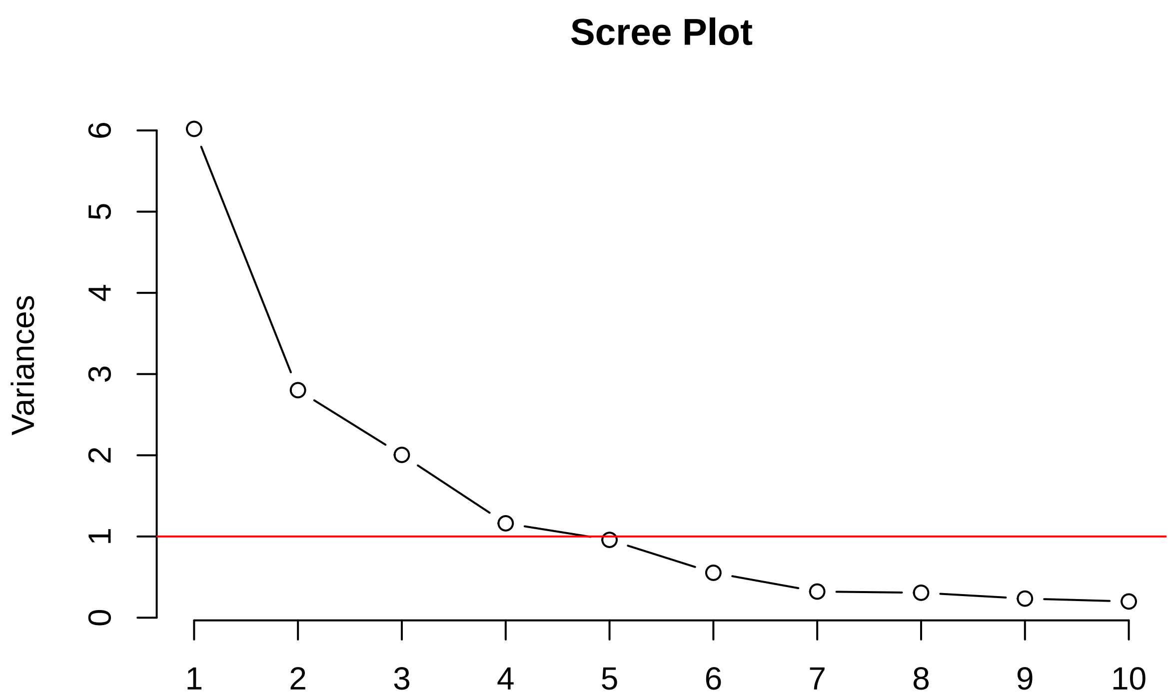
- It's helpful to plot the variances against the proportional variances of the pca.

```
variance = crime_pca$sdev^2
prop_variance <- variance/sum(variance)
plot(prop_variance,  xlab = "Principal Component", ylab = "Proportional Variance",
     ylim = c(0,1) , type= "b")
```




- Further, we can look into a Scree Plot, which will help us determine how many components to utilize.

```
screeplot(crime_pca, main = "Scree Plot", type = "line")
abline(h=1, col="red")
```




- It looks like we should choose between 4 and 5 components. Let's go with 4 and then generate the new linear model using those components.

```
top_pcs = cbind(crime_pca$x[,1:4],crime_data[,16])
colnames(top_pcs) = c("PC1", "PC2", "PC3", "PC4", "Crime")
head(top_pcs)
```

```
##          PC1        PC2         PC3         PC4 Crime
## 1 -4.199284 -1.0938312 -1.11907395  0.67178115   791
## 2  1.172663  0.6770136 -0.05244634 -0.08350709  1635
## 3 -4.173725  0.2767750 -0.37107658  0.37793995   578
## 4  3.834962 -2.5769060  0.22793998  0.38262331  1969
## 5  1.839300  1.3309856  1.27882805  0.71814305  1234
## 6  2.907234 -0.3305421  0.53288181  1.22140635   682
```

```r
# Now create a linear model using these principal components.
crime_lm = lm(Crime ~., data = as.data.frame(top_pcs))
summary(crime_lm)
```

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

- We now have linear model that used the first 4 principal components. We see that the adjusted r-squared is 0.2433. Now let's retrieve the coefficients of the the principal components.

```r
beta_0 = crime_lm$coefficients[1]
beta_i = crime_lm$coefficients[2:5]
alpha_i = crime_pca$rotation[,1:4] %*% beta_i

# we convert the alpha to adjust for scaling.

adjusted_alpha = alpha_i/sapply(crime_data[,1:15],sd)
adjusted_beta0 = beta_0 - sum(alpha_i*sapply(crime_data[,1:15],mean)/sapply(crime_data[,1:15],sd))

t(adjusted_alpha)
```

```
##              M       So       Ed      Po1      Po2        LF       M.F      Pop
## [1,] -16.93076 21.34368 12.82972 21.35216 23.08832 -346.5657 -8.293097 1.046216
##            NW        U1       U2    Wealth      Ineq     Prob       Time
## [1,] 1.500994 -1509.935 1.688367 0.0400119 -6.902022 144.9493 -0.9330765
```

```r
adjusted_beta0
```

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

- This gives us the new model in terms of the original variables. In other words, the model is: -16.93076M + 21.34368So + 12.82972Ed + 21.35216Po1 + 23.08832Po2 -346.5657LF -8.293097M.F + 1.046216Pop + 1.500994NW -1509.935U1 + 1.688367U2 + 0.0400119Wealth -6.902022Ineq + 144.9493Prob -0.9330765Time + 1666.485
- If we compare this a relatively simpler linear regression model, we see the PCA model heavily underperforms relative to the full model (which has an adjusted R-squared value of 0.7078), despite using the same amount of data. This is because we left out some of the PCs. Had we specified all the PCs, estimates would be the same of regression.
- On the other hand, we note that we obtain a good fit with a much lower number of regressors, which is because PCA takes out the multicollinearity between the variables and creates linear combinations that are orthogonal to each other.
- Overall, we clearly see the pros/cons of PCA. It reduces a problem's dimensionality and is particularly helpful when the original predictors are strongly correlated. However, it does not reduce the data requirements and therefore the associated costs of collecting data.

```r
lm_mod = lm(Crime ~., data = crime_data)
summary(lm_mod)
```

```
##
## 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
```

- Comparing our model to the one in HW5, the model "only" explained 48.55% of the variation in Crime.

```r
# Remove observations 4 and 26 (likely outliers).
data = read.delim("uscrime.txt", header=TRUE)
data = data[-c(4,26),]
# re-code 'Prob' to percentage-points into 'Prob2'
Prob2 = data$Prob*100
data = cbind(data, Prob2)
data = within(data, rm(Prob))
# re-order data and isolate PCA data
data = data[c(1:14, 16, 15)]
pca = data[c(1:15)]

model = prcomp(pca, center=TRUE, scale=TRUE)

# run model
model3 = lm(Crime ~ Wealth + M + factor(So) + Po1 + Prob2, data = data)
summary(model3)
```

```
##
## Call:
## lm(formula = Crime ~ Wealth + M + factor(So) + Po1 + Prob2, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -484.62 -133.79   28.25  106.08  552.80
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -5.062e+02  7.988e+02  -0.634  0.53000
## Wealth      -7.509e-03  7.989e-02  -0.094  0.92560
## M            6.792e+01  4.111e+01   1.652  0.10652
## factor(So)1  1.524e+02  1.115e+02   1.367  0.17949
## Po1          7.795e+01  2.283e+01   3.414  0.00151 **
## Prob2       -4.866e+01  2.038e+01  -2.387  0.02191 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 242.6 on 39 degrees of freedom
## Multiple R-squared:  0.4855, Adjusted R-squared:  0.4196
## F-statistic: 7.361 on 5 and 39 DF,  p-value: 6.111e-05
```

```r
summary(model3)$r.squared
```

```
## [1] 0.4855126
```