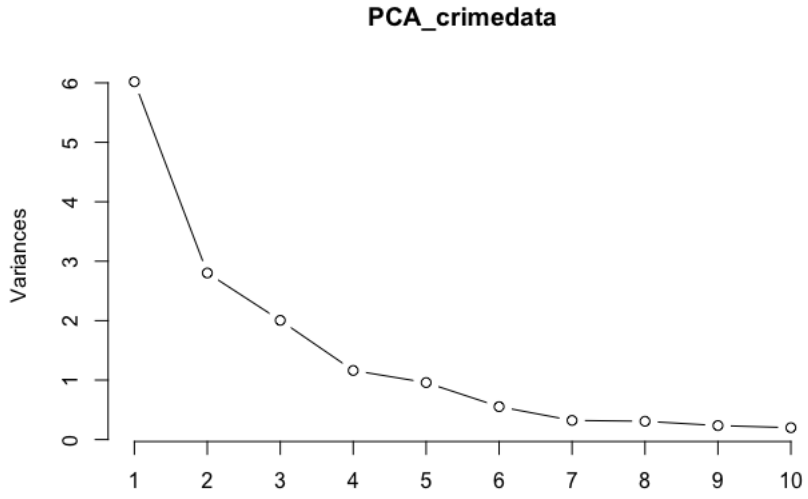
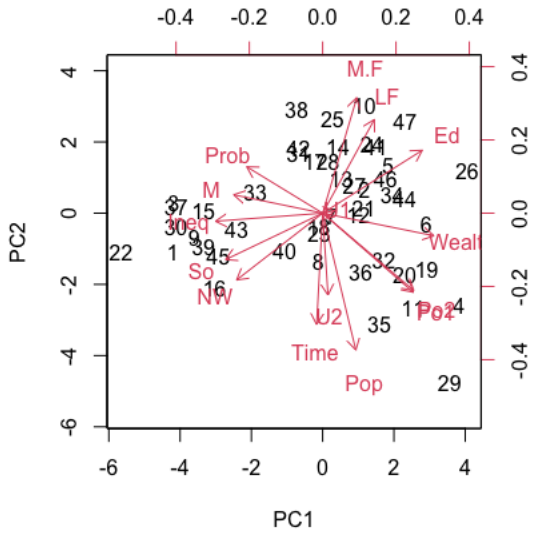
Question 9.1

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)!)

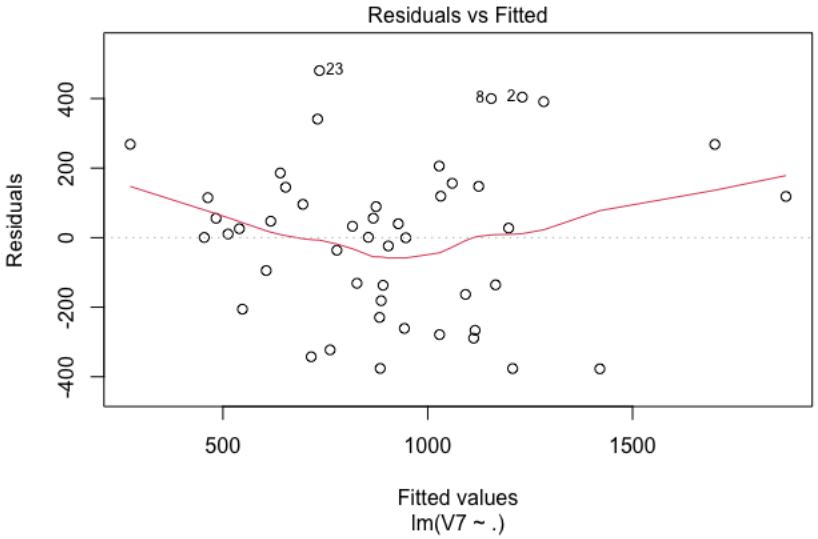
To create this regression model using the first few principal components, I started by setting the seed and loading the data into R. After this is when I applied PCA using the prcomp function. Since PCA is sensitive to scale, I scaled the data in the crime data set. I used all of the columns in the data set except the response one (the last column in the data set). I then ran a summary on the PCA to see what I was looking at number wise. Each component had some proportion of variance, when added up = 1. Now I had to pick only the principal components that contained majorly all the variances accounted for. To figure this out, I plotted my data to look almost like an elbow graph. Instead of distance on the Y-axis, this one had variances. The graph is shown below:



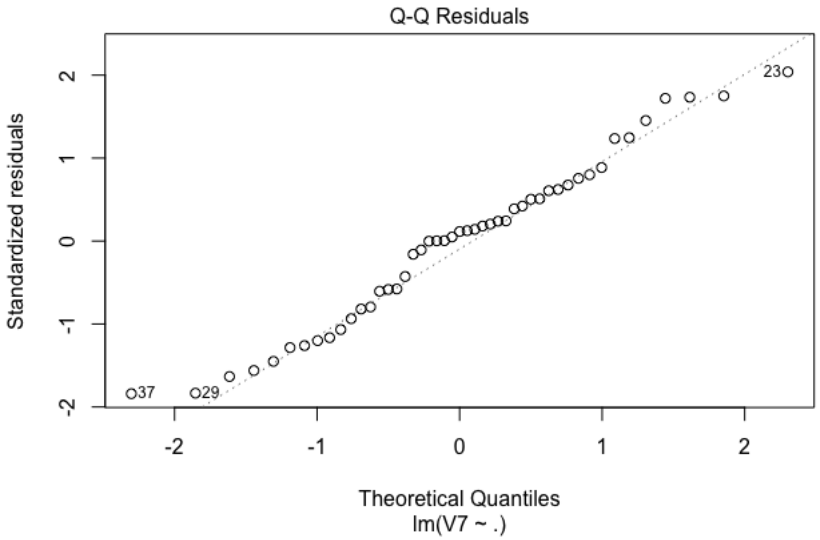
This graph shows that the first 6 or 7 (I chose 6) principal components account for most of the variance. To check the eigenvectors, I used the biplot function. The red lines show the eigenvectors. This just helps interpret the principal components. The graph is shown below:



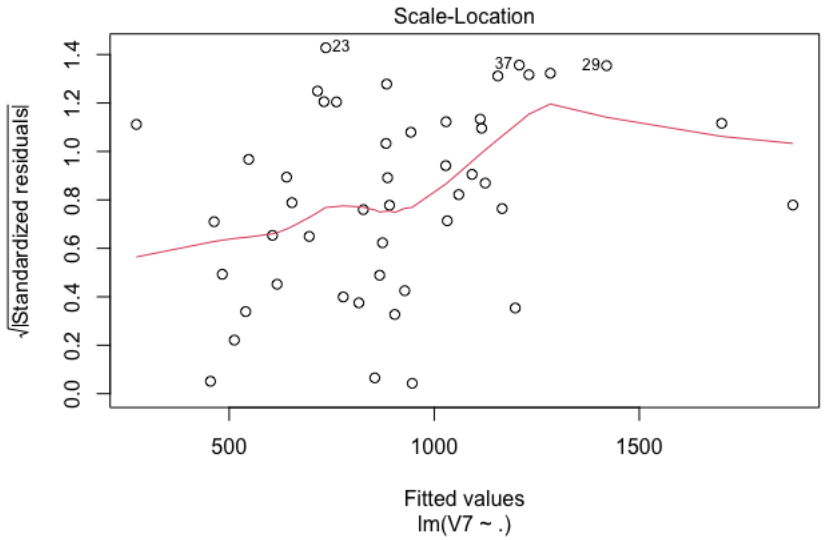
I then created another variable that only included the first 6 PC's. At this point is when I built my linear regression model. To do this, I used the cbind function to bind the 6 PC's to the crime data set, excluding the response column. I then converted it to a dataframe. This produced values for PC1 through PC6 using the respective data set. It also produced a prediction value "V7" for each row. Since I was forming a linear model, I had to form a linear equation where PC1 to PC6 were my independent variables. To fit my linear model, I used the lm function with "V7" as my prediction over the rest of the independent variable (PC1 – PC6) for the crime data set. I then plotted this model, which resulted in 4 different graphs. The "Results vs Fitted" graph shows that the residuals are distributed evenly for the most part across 0 Residuals (the horizontal length). Residuals in this case is the difference between what is in the observations in the data set and what is predicted.



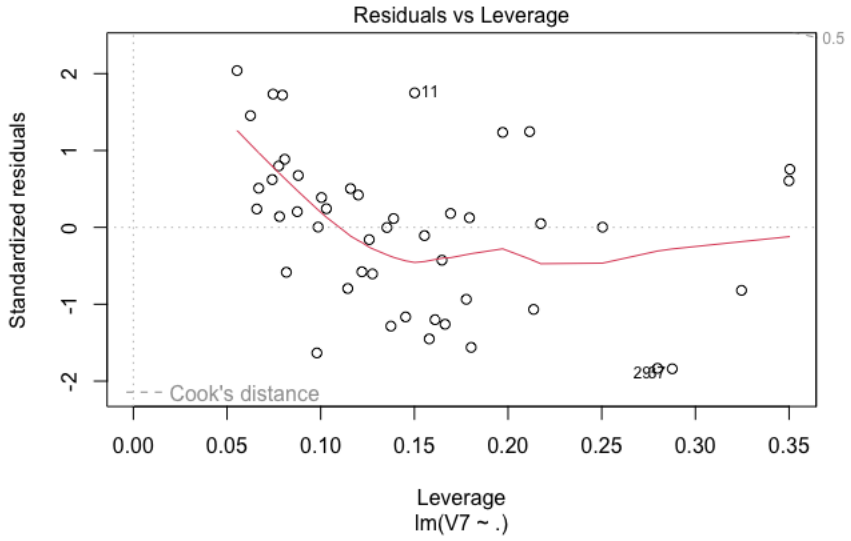
The “Q-Q Residuals” graph shows if the data is normally distributed or not. Between one standard deviation, it is shown that pretty much all the data is concentrated. It is not perfectly normally distributed, but for this test it is enough.



The “Scale-Location” graph is a scaled version of the “Residuals vs Fitted” graph. It standardizes the residuals to fit them. The standardized residuals are more sensitive to the data than the non-standardized ones, which is why the graph is different. Like the first graph, all the x values look to be independent of each other.



The “Residuals vs Leverage” graph shows that the points that are further away from the 0 line are high influential points. Cooks distance tells you if any point lies beyond the dashed 0.5 or 1 line, they could hurt the model if it is an outlier. This graph shows that there are no points that lie beyond that range, so there are likely no outliers that will hurt the model. If there are outliers that are beyond that range, the model will try to fit them into it, which hurts the pattern it is trying to form.



Looking at the 4 of these graphs, it shows that a linear regression model can be used. I then checked the structure of the model using the str function. This allowed me to see every piece of the model I will need is, like the coefficients, residuals, etc. I then took a summary of the model. This summary showed me that my adjusted R squared value was 0.6074. I now needed to figure out the linear equation for the model. Using the structure of the model, I extracted the coefficients and the intercept. My equation was { Crime = 905.0851 + 65.21593PC1 – 70.08312PC2 + 25.19408PC3 + 69.44603PC4 – 229.04282PC5 – 60.21329PC6 }. The issue is principal components are nothing but derived components derived from the original attributes, in this case the attributes from the crime data set. So, I needed to convert this back into my original attributes and then I will have my equation in terms of my original attributes. I can then use the equation to predict the crime rate. What I did to transform the PC’s into the original attributes was to multiply the PC coefficients by the rotation and then unscale. I checked my original data scaled data set without the response attribute, which showed the rotation for each of the variable’s representation wise of the principal component representation. I then multiplied the rotation by the coefficients to get the following scaled version of the original coefficients:

M	87.838105
So	43.899723
Ed	20.463867
Po1	123.111917
Po2	118.647767
LF	45.893299
M.F	112.612562
Pop	25.937634
NW	94.987694
U1	1.819916
U2	29.445920
Wealth	45.247340
Ineq	5.724056
Prob	-51.712898
Time	36.128815

Now I needed to unscale this. To unscale, I took this data and divided it by the standard deviation to get the following original coefficients:

M	6.989232e+01
So	9.165344e+01
Ed	1.829254e+01
Po1	4.142536e+01
Po2	4.243282e+01
LF	1.135641e+03
M.F	3.821603e+01
Pop	6.812930e-01

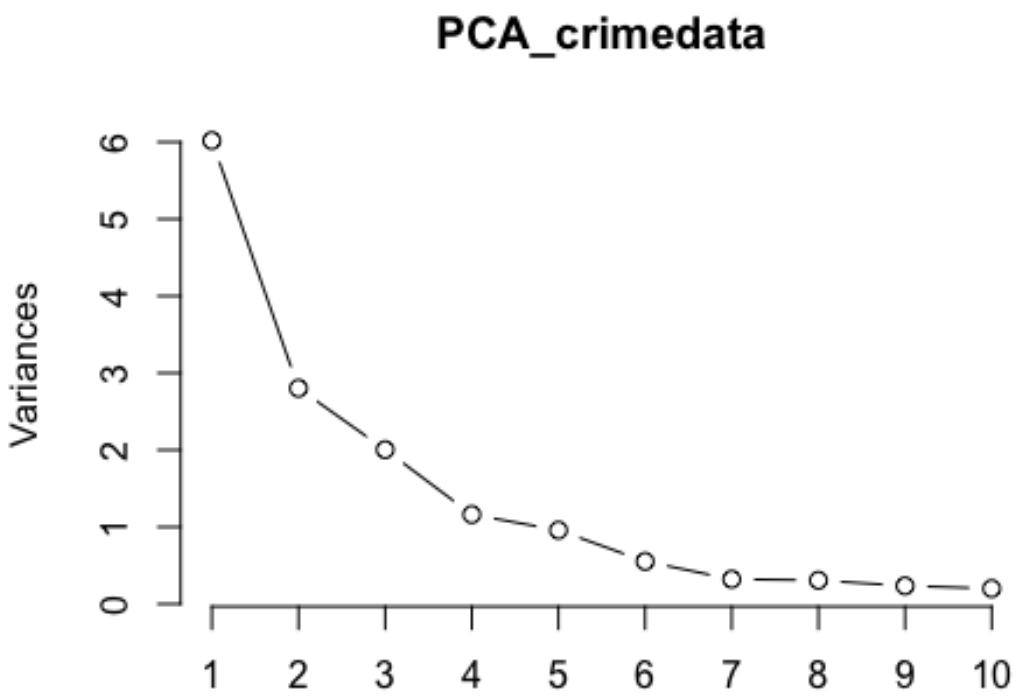
NW	9.237458e+00
U1	1.009450e+02
U2	3.486602e+01
Wealth	4.689284e-02
Ineq	1.434742e+00
Prob	-2.274397e+03
Time	5.097975e+00

Now, I also needed to get my intercept. To do this, I took the scaled intercept in the equation with the PC's and subtracted it from the sum of the alpha*mean/sd – alpha and mean are defined in the code. This got me an intercept value of -5923.647. So, now my linear equation would be { Crime = -5923.647 + 6.989232e+01M + 9.165344e+01So + 1.829254e+01Ed + 4.142536e+01Po1 + 4.243282e+01Po2 + 1.135641e+03LF + 3.821603e+01M.F + 6.812930e-01Pop + 9.237458e+00NW + 1.009450e+02U1 + 3.486602e+01U2 + 4.689284e-02Wealth + 1.434742e+00Ineq - 2.274397e+03Prob + 5.097975e+00Time } . To see how good this model is, I calculated the R squared value of it. To find the R squared value, I found the sum of squared errors using the values in the equation (coefficients and intercept). I then calculated the sum of squared totals. To find the R squared, I did 1 – sum of squares/sum of squared totals. The R squared value I got was 0.6586023.

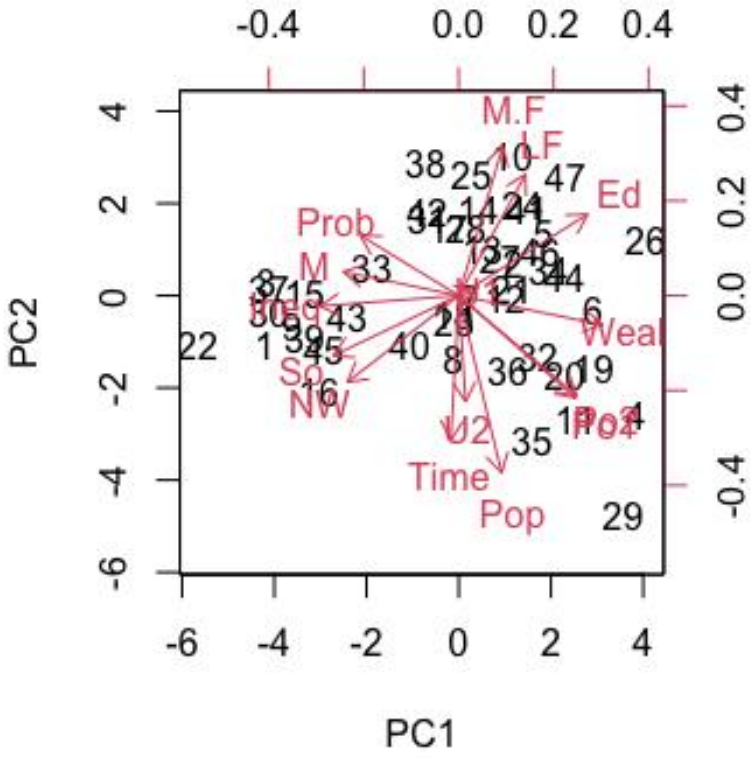
Compared to my solution to Question 8.2, this model produced a better final R squared value, meaning its quality was better, although not by much. The R squared value of my Question 8.2 model was 0.6339817 whereas this one produced the value 0.6586023. I would be interested to see if the difference would be any smaller or larger with a larger data set and not one with as few observations as this crime data set.

R Code and Output:

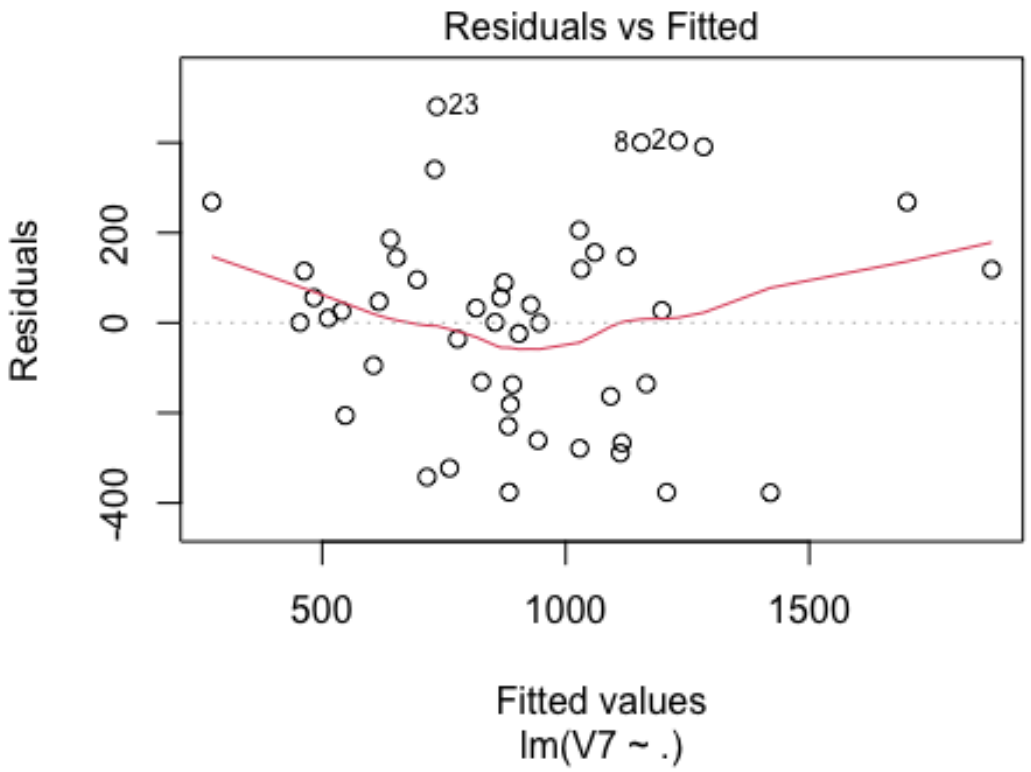
```
set.seed(123)
crimedata <- read.table("uscrime.txt", header = TRUE)

PCA_crimedata <- prcomp(crimedata[,-16], scale = TRUE)
summary(PCA_crimedata)

## Importance of components:
##                           PC1    PC2    PC3     PC4     PC5     PC6     PC
7
## Standard deviation     2.4534 1.6739 1.4160 1.07806 0.97893 0.74377 0.5672
9
## Proportion of Variance 0.4013 0.1868 0.1337 0.07748 0.06389 0.03688 0.0214
5
## Cumulative Proportion  0.4013 0.5880 0.7217 0.79920 0.86308 0.89996 0.9214
2
##                           PC8     PC9    PC10    PC11    PC12    PC13   P
C14
## Standard deviation     0.55444 0.48493 0.44708 0.41915 0.35804 0.26333 0.2
418
## Proportion of Variance 0.02049 0.01568 0.01333 0.01171 0.00855 0.00462 0.0
039
## Cumulative Proportion  0.94191 0.95759 0.97091 0.98263 0.99117 0.99579 0.9
997
##                           PC15
## Standard deviation     0.06793
## Proportion of Variance 0.00031
## Cumulative Proportion  1.00000

plot(PCA_crimedata, type = "l")
```

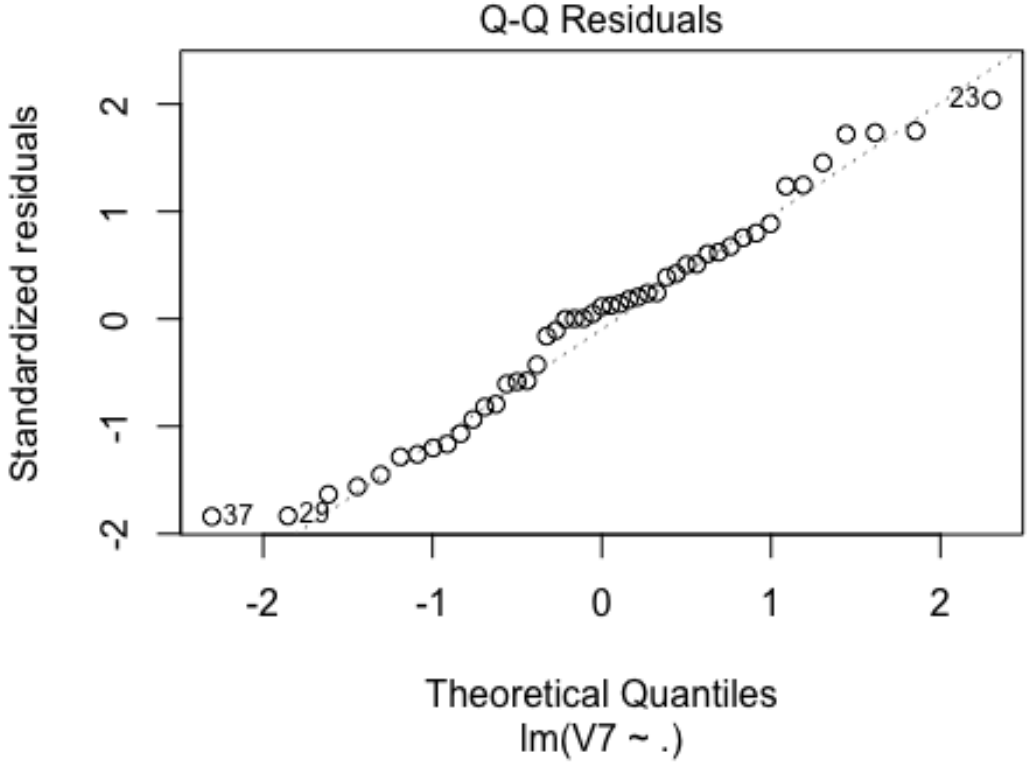


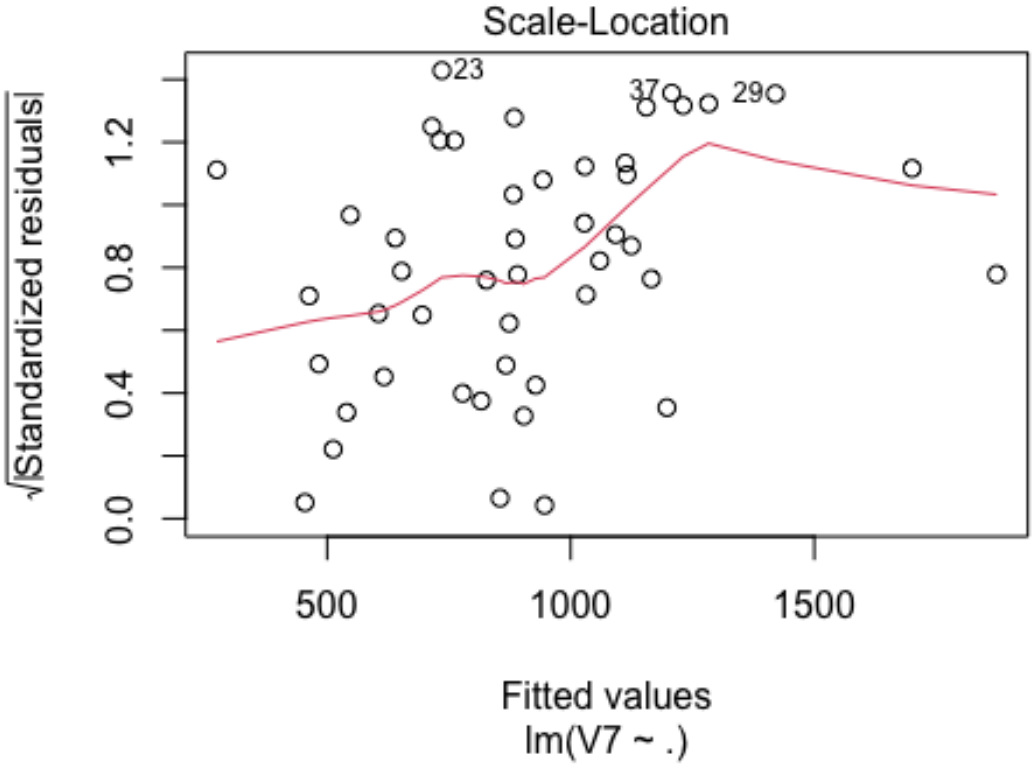

```
biplot(PCA_crimedata, scale = 0)
```

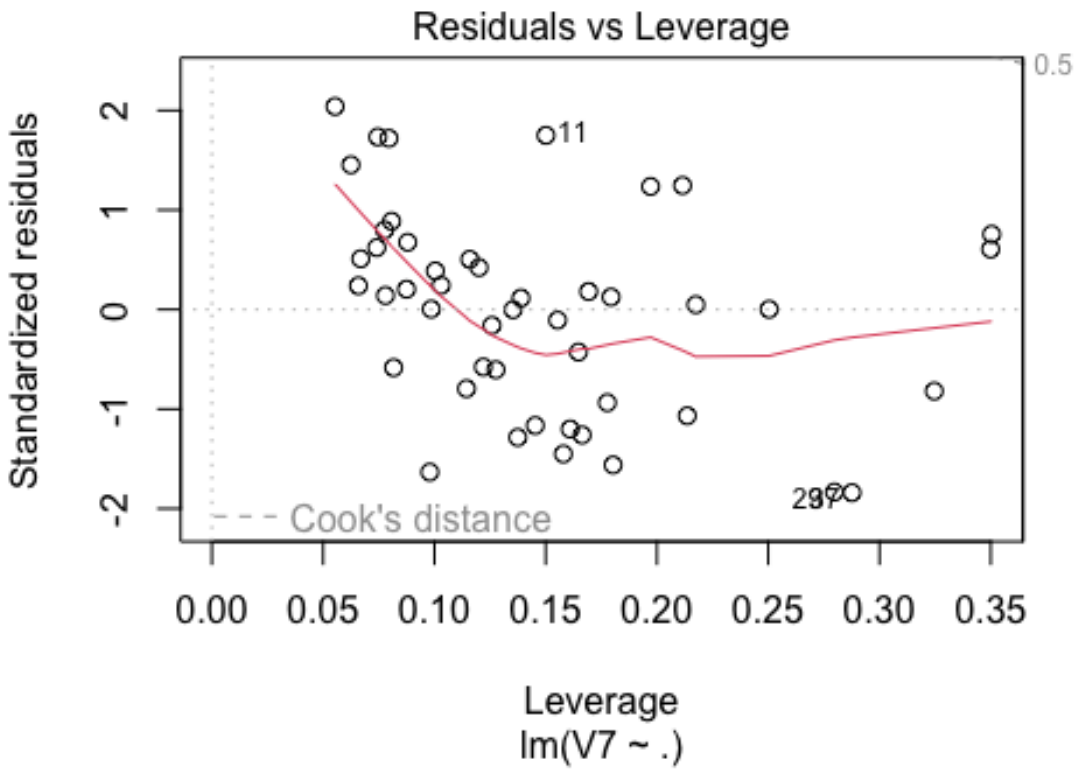



```
PCs <- PCA_crimedata$x[,1:6]

crime2 <- cbind(PCs, crimedata[,16])
crime2 <- as.data.frame(crime2)

model <- lm(V7~., data = crime2)
plot(model)
```


Residuals vs Fitted
23
8
20
Residuals
200
0
-400
500
1000
1500
Fitted values
lm(V7 ~ .)


Q-Q Residuals
Standardized residuals
-2
-1
0
1
2
37
29
23
-2
-1
0
1
2
Theoretical Quantiles
lm(V7 ~ .)


Scale-Location
√|Standardized residuals|
0.0
0.4
0.8
1.2
23
37
29
500
1000
1500
Fitted values
lm(V7 ~ .)




```
str(model)
```

```
## List of 12
##  $ coefficients : Named num [1:7] 905.1 65.2 -70.1 25.2 69.4 ...
##   ..- attr(*, "names")= chr [1:7] "(Intercept)" "PC1" "PC2" "PC3" ...
##  $ residuals    : Named num [1:47] 95.8 404.2 114.9 268.1 205.9 ...
##   ..- attr(*, "names")= chr [1:47] "1" "2" "3" "4" ...
##  $ effects      : Named num [1:47] -6205 -1085 796 -242 -508 ...
##   ..- attr(*, "names")= chr [1:47] "(Intercept)" "PC1" "PC2" "PC3" ...
##  $ rank         : int 7
##  $ fitted.values: Named num [1:47] 695 1231 463 1701 1028 ...
##   ..- attr(*, "names")= chr [1:47] "1" "2" "3" "4" ...
##  $ assign       : int [1:7] 0 1 2 3 4 5 6
##  $ qr           :List of 5
##   ..$ qr   : num [1:47, 1:7] -6.856 0.146 0.146 0.146 0.146 ...
##   .. ..- attr(*, "dimnames")=List of 2
##   .. .. ..$ : chr [1:47] "1" "2" "3" "4" ...
##   .. .. ..$ : chr [1:7] "(Intercept)" "PC1" "PC2" "PC3" ...
##   .. ..- attr(*, "assign")= int [1:7] 0 1 2 3 4 5 6
##   ..$ qraux: num [1:7] 1.15 1.1 1.05 1.03 1.08 ...
##   ..$ pivot: int [1:7] 1 2 3 4 5 6 7
##   ..$ tol  : num 1e-07
##   ..$ rank : int 7
```

```
##   ..- attr(*, "class")= chr "qr"
##  $ df.residual  : int 40
##  $ xlevels      : Named list()
##  $ call         : language lm(formula = V7 ~ ., data = crime2)
##  $ terms        :Classes 'terms', 'formula'  language V7 ~ PC1 + PC2 + PC3
+ PC4 + PC5 + PC6
##   .. ..- attr(*, "variables")= language list(V7, PC1, PC2, PC3, PC4, PC5,
PC6)
##   .. ..- attr(*, "factors")= int [1:7, 1:6] 0 1 0 0 0 0 0 0 0 1 ...
##   .. .. ..- attr(*, "dimnames")=List of 2
##   .. .. .. ..$ : chr [1:7] "V7" "PC1" "PC2" "PC3" ...
##   .. .. .. ..$ : chr [1:6] "PC1" "PC2" "PC3" "PC4" ...
##   .. ..- attr(*, "term.labels")= chr [1:6] "PC1" "PC2" "PC3" "PC4" ...
##   .. ..- attr(*, "order")= int [1:6] 1 1 1 1 1 1
##   .. ..- attr(*, "intercept")= int 1
##   .. ..- attr(*, "response")= int 1
##   .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
##   .. ..- attr(*, "predvars")= language list(V7, PC1, PC2, PC3, PC4, PC5, P
C6)
##   .. ..- attr(*, "dataClasses")= Named chr [1:7] "numeric" "numeric" "nume
ric" "numeric" ...
##   .. .. ..- attr(*, "names")= chr [1:7] "V7" "PC1" "PC2" "PC3" ...
##  $ model        :'data.frame':   47 obs. of  7 variables:
##   ..$ V7 : num [1:47] 791 1635 578 1969 1234 ...
##   ..$ PC1: num [1:47] -4.2 1.17 -4.17 3.83 1.84 ...
##   ..$ PC2: num [1:47] -1.094 0.677 0.277 -2.577 1.331 ...
##   ..$ PC3: num [1:47] -1.1191 -0.0524 -0.3711 0.2279 1.2788 ...
##   ..$ PC4: num [1:47] 0.6718 -0.0835 0.3779 0.3826 0.7181 ...
##   ..$ PC5: num [1:47] 0.0553 -1.1732 0.5413 -1.6447 0.0416 ...
##   ..$ PC6: num [1:47] 0.307 -0.583 0.719 0.729 -0.394 ...
##   ..- attr(*, "terms")=Classes 'terms', 'formula'  language V7 ~ PC1 + PC2
+ PC3 + PC4 + PC5 + PC6
##   .. .. ..- attr(*, "variables")= language list(V7, PC1, PC2, PC3, PC4, PC
5, PC6)
##   .. .. ..- attr(*, "factors")= int [1:7, 1:6] 0 1 0 0 0 0 0 0 0 1 ...
##   .. .. .. ..- attr(*, "dimnames")=List of 2
##   .. .. .. .. ..$ : chr [1:7] "V7" "PC1" "PC2" "PC3" ...
##   .. .. .. .. ..$ : chr [1:6] "PC1" "PC2" "PC3" "PC4" ...
##   .. .. ..- attr(*, "term.labels")= chr [1:6] "PC1" "PC2" "PC3" "PC4" ...
##   .. .. ..- attr(*, "order")= int [1:6] 1 1 1 1 1 1
##   .. .. ..- attr(*, "intercept")= int 1
##   .. .. ..- attr(*, "response")= int 1
##   .. .. ..- attr(*, ".Environment")=<environment: R_GlobalEnv>
##   .. .. ..- attr(*, "predvars")= language list(V7, PC1, PC2, PC3, PC4, PC5
, PC6)
##   .. .. ..- attr(*, "dataClasses")= Named chr [1:7] "numeric" "numeric" "n
umeric" "numeric" ...
##   .. .. .. ..- attr(*, "names")= chr [1:7] "V7" "PC1" "PC2" "PC3" ...
##  - attr(*, "class")= chr "lm"
```

```r
summary(model)
```

```
##
## Call:
## lm(formula = V7 ~ ., data = crime2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -377.15 -172.23   25.81  132.10  480.38
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   905.09      35.35  25.604  < 2e-16 ***
## PC1            65.22      14.56   4.478 6.14e-05 ***
## PC2           -70.08      21.35  -3.283  0.00214 **
## PC3            25.19      25.23   0.998  0.32409
## PC4            69.45      33.14   2.095  0.04252 *
## PC5          -229.04      36.50  -6.275 1.94e-07 ***
## PC6           -60.21      48.04  -1.253  0.21734
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 242.3 on 40 degrees of freedom
## Multiple R-squared:  0.6586, Adjusted R-squared:  0.6074
## F-statistic: 12.86 on 6 and 40 DF,  p-value: 4.869e-08
```

```r
intercept <- model$coefficients[1]
coefficents <- model$coefficients[2:7]
coefficents
```

```
##        PC1        PC2        PC3        PC4        PC5        PC6
##   65.21593  -70.08312   25.19408   69.44603 -229.04282  -60.21329
```

```r
intercept
```

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

```r
(PCA_crimedata)
```

```
## 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  0.394
46657
## So     -0.100500743  0.19649727  0.22734157 -0.65599872  0.06141452  0.233
97868
## Ed     -0.008571367 -0.23943629 -0.14644678 -0.44326978  0.51887452 -0.118
21954
## Po1    -0.095776709  0.08011735  0.04613156  0.19425472 -0.14320978 -0.130
42001
## Po2    -0.119524780  0.09518288  0.03168720  0.19512072 -0.05929780 -0.138
85912
## LF      0.504234275 -0.15931612  0.25513777  0.14393498  0.03077073  0.385
32827
## M.F    -0.074501901  0.15548197 -0.05507254 -0.24378252 -0.35323357 -0.280
29732
## Pop     0.547098563  0.09046187 -0.59078221 -0.20244830 -0.03970718  0.058
49643
## NW      0.051219538 -0.31154195  0.20432828  0.18984178  0.49201966 -0.206
95666
## U1      0.017385981 -0.17354115 -0.20206312  0.02069349  0.22765278 -0.178
57891
## U2      0.048155286 -0.07526787  0.24369650  0.05576010 -0.04750100  0.470
21842
## Wealth -0.154683104 -0.14859424  0.08630649 -0.23196695 -0.11219383  0.319
55631
## Ineq    0.272027031  0.37483032  0.07184018 -0.02494384 -0.01390576 -0.182
78697
## Prob    0.283535996 -0.56159383 -0.08598908 -0.05306898 -0.42530006 -0.089
78385
## Time   -0.148203050 -0.44199877  0.19507812 -0.23551363 -0.29264326 -0.263
63121
##                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
```

```
alpha <- PCA_crimedata$rotation[,1:6] %*% coefficents
alpha
```

```
##              [,1]
## M       87.838105
## So      43.899723
## Ed      20.463867
## Po1    123.111917
## Po2    118.647767
## LF      45.893299
## M.F    112.612562
## Pop     25.937634
## NW      94.987694
## U1       1.819916
## U2      29.445920
## Wealth  45.247340
## Ineq     5.724056
## Prob   -51.712898
## Time    36.128815
```

```
sd <- sapply(crimedata[,1:15], sd)
mean <- sapply(crimedata[,1:15], mean)
alpha_org <- alpha/sd
alpha_org
```

```
##                 [,1]
## M       6.989232e+01
## So      9.165344e+01
## Ed      1.829254e+01
## Po1     4.142536e+01
## Po2     4.243282e+01
## LF      1.135641e+03
## M.F     3.821603e+01
## Pop     6.812930e-01
## NW      9.237458e+00
## U1      1.009450e+02
## U2      3.486602e+01
## Wealth  4.689284e-02
## Ineq    1.434742e+00
## Prob   -2.274397e+03
## Time    5.097975e+00
```

```r
a0_org <- intercept - sum(alpha*mean/sd)
a0_org
```

```
## (Intercept)
##   -5923.647
```

```r
crime_value_predict <- as.matrix(crimedata[,-16]) %*% alpha_org + a0_org
SSE <- sum((crimedata[,16] - crime_value_predict)^2)
SSE
```

```
## [1] 2349133
```

```r
SStot <- sum((crimedata[,16] - mean(crimedata[,16]))^2)
SStot
```

```
## [1] 6880928
```

```r
Rsquared <- 1-SSE/SStot
Rsquared
```

```
## [1] 0.6586023
```