PCA on Arrests data

PCA on the USArrests data set, which is part of the base R package. The rows of the data set contain the 50 states, in alphabetical order.

```
states =row.names(USArrests )
states
```

```
##  [1] "Alabama"        "Alaska"         "Arizona"        "Arkansas"
##  [5] "California"     "Colorado"       "Connecticut"    "Delaware"
##  [9] "Florida"        "Georgia"        "Hawaii"         "Idaho"
## [13] "Illinois"       "Indiana"        "Iowa"           "Kansas"
## [17] "Kentucky"       "Louisiana"      "Maine"          "Maryland"
## [21] "Massachusetts"  "Michigan"       "Minnesota"      "Mississippi"
## [25] "Missouri"       "Montana"        "Nebraska"       "Nevada"
## [29] "New Hampshire"  "New Jersey"     "New Mexico"     "New York"
## [33] "North Carolina" "North Dakota"   "Ohio"           "Oklahoma"
## [37] "Oregon"         "Pennsylvania"   "Rhode Island"   "South Carolina"
## [41] "South Dakota"   "Tennessee"      "Texas"          "Utah"
## [45] "Vermont"        "Virginia"       "Washington"     "West Virginia"
## [49] "Wisconsin"      "Wyoming"
```

```
names(USArrests)
```

```
## [1] "Murder"   "Assault"  "UrbanPop" "Rape"
```

The columns of the data set contain the four variables

```
apply(USArrests , 2, mean)
```

```
##   Murder  Assault UrbanPop     Rape
##    7.788  170.760   65.540   21.232
```

Means are different for each of the columns

```
apply(USArrests , 2, var)
```

```
##     Murder    Assault   UrbanPop       Rape
##   18.97047 6945.16571  209.51878   87.72916
```

Different variances across each of the columns. Hence the need to scale the variables. Now calculating the PCA using scale = TRUE

```
pr.out =prcomp (USArrests , scale =TRUE)
```

By default, the prcomp() function centers the variables to have mean zero. By using the option scale=TRUE, we scale the variables to have standard deviation one.

```
names(pr.out)
```

```
## [1] "sdev"     "rotation" "center"   "scale"    "x"
```

The center and scale components correspond to the means and standard deviations of the variables that were used for scaling prior to implementing PCA.

```
pr.out$center
```

```
##   Murder  Assault UrbanPop     Rape
##    7.788  170.760   65.540   21.232
```

```r
pr.out$scale
```

```
##    Murder   Assault  UrbanPop      Rape
##  4.355510 83.337661 14.474763  9.366385
```

The rotation matrix provides the principal component loadings; each column of pr.out$rotation contains the corresponding principal component loading vector.

```r
pr.out$rotation
```

```
##                 PC1        PC2        PC3         PC4
## Murder   -0.5358995  0.4181809 -0.3412327  0.64922780
## Assault  -0.5831836  0.1879856 -0.2681484 -0.74340748
## UrbanPop -0.2781909 -0.8728062 -0.3780158  0.13387773
## Rape     -0.5434321 -0.1673186  0.8177779  0.08902432
```

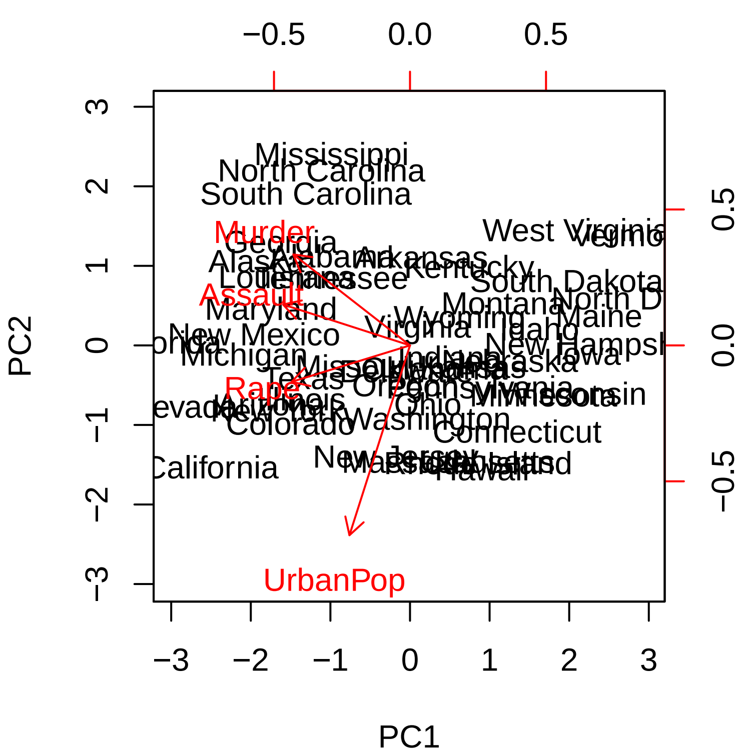
We see that there are four distinct principal components. This is to be expected because there are in general min(n ??? 1, p) informative principal components in a data set with n observations and p variables. We do not need to explicitly multiply the data by the principal component loading vectors in order to obtain the principal component score vectors. Rather the 50 ×4 matrix x has as its columns the principal component score vectors. That is, the kth column is the kth principal component score vector.

```r
dim(pr.out$x )
```

```
## [1] 50  4
```

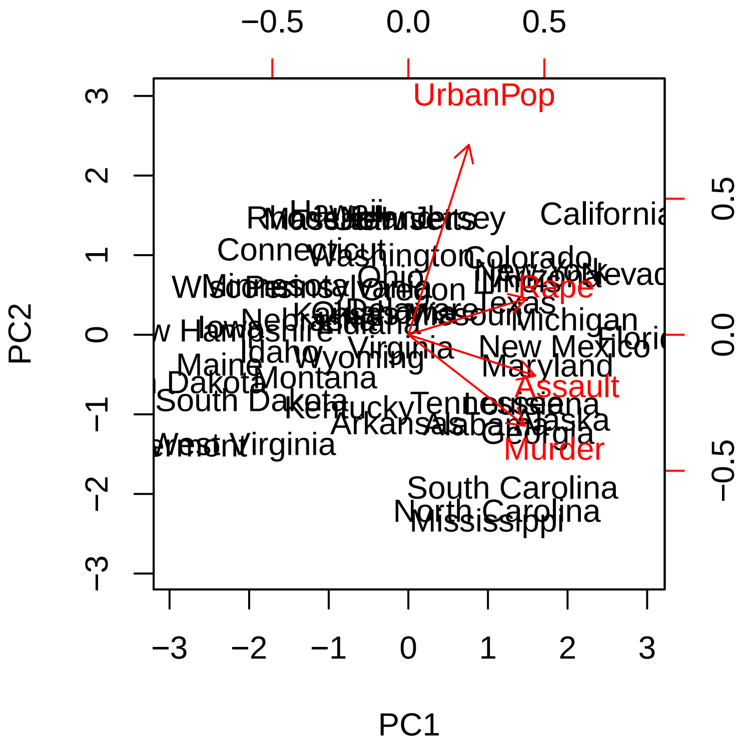
Plotting the first two principal components as follows:-

```r
biplot (pr.out , scale =0)
```



Changing the sign of the components

```
pr.out$rotation=-pr.out$rotation
pr.out$x=-pr.out$x
biplot (pr.out , scale =0,expand = 1)
```



Accessing the standard deviations

```
pr.out$sdev
```

```
## [1] 1.5748783 0.9948694 0.5971291 0.4164494
```

Accessing the variances

```
pr.var =pr.out$sdev ^2
pr.var
```

```
## [1] 2.4802416 0.9897652 0.3565632 0.1734301
```

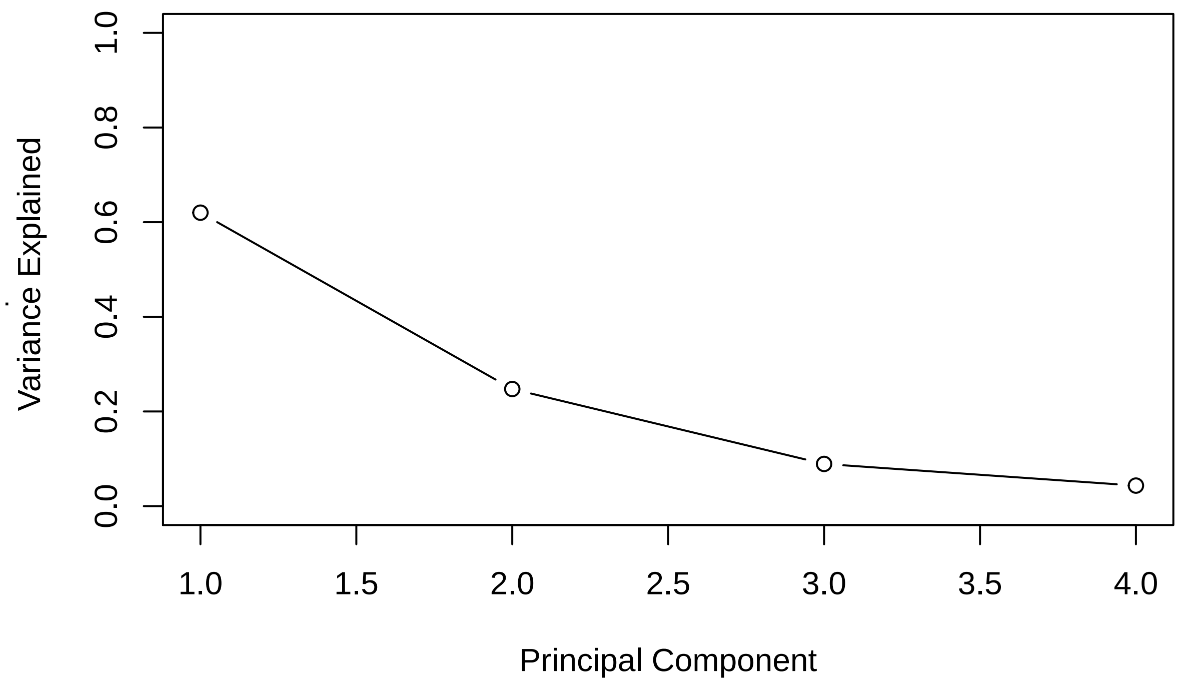
To compute the proportion of variance explained by each principal component, we simply divide the variance explained by each principal component by the total variance explained by all four principal components:

```
pve=pr.var/sum(pr.var )
pve
```

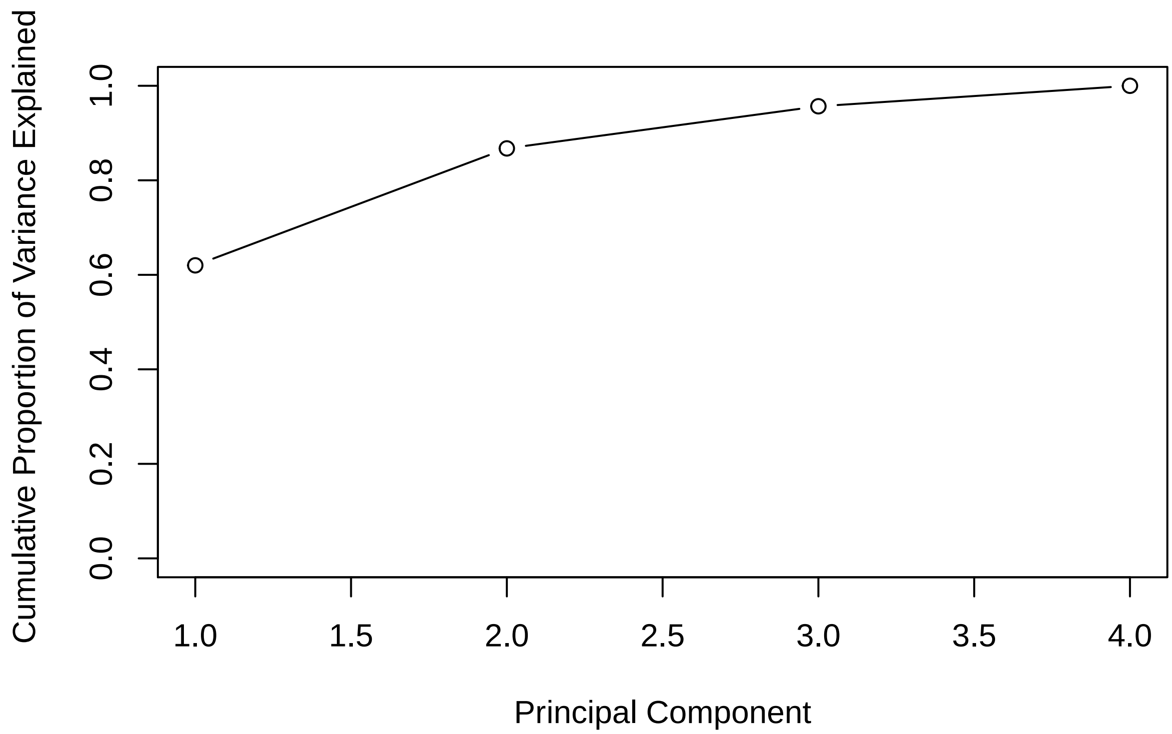
```
## [1] 0.62006039 0.24744129 0.08914080 0.04335752
```

We can plot the PVE explained by each component, as well as the cumulative PVE, as follows:

```
plot(pve , xlab=" Principal Component ", ylab=" Proportion of
Variance Explained ", ylim=c(0,1) ,type="b")
```




```
plot(cumsum (pve ), xlab=" Principal Component ", ylab ="
Cumulative Proportion of Variance Explained ", ylim=c(0,1) ,
type="b")
```


Cumulative Proportion of Variance Explained
0.0
0.2
0.4
0.6
0.8
1.0
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Principal Component