PCA

Principal Component Analysis

Code

When faced with a large set of correlated variables, principal components allow us to summarize this set with a smaller number of representative variables that collectively explain most of the variability in the original set.

The principal component directions are directions in feature space along which the original data are highly variable.

Principal component analysis (PCA) refers to the process by which principal components are computed, and the subsequent use of these components in understanding the data.

PCA is an unsupervised approach, since it involves only a set of features X1, X2, . . . , Xp, and no associated response Y . Apart from producing derived variables for use in supervised learning problems, PCA also serves as a tool for data visualization.

Here, we perform 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.

Hide

```
data("USArrests")
str(USArrests)
```

```
'data.frame':   50 obs. of  4 variables:
 $ Murder  : num  13.2 10 8.1 8.8 9 7.9 3.3 5.9 15.4 17.4 ...
 $ Assault : int  236 263 294 190 276 204 110 238 335 211 ...
 $ UrbanPop: int  58 48 80 50 91 78 77 72 80 60 ...
 $ Rape    : num  21.2 44.5 31 19.5 40.6 38.7 11.1 15.8 31.9 25.8 ...
```

lets compute the mean and variance of each variable.

Hide

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

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

Hide

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

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

We see that there are on average three times as many rapes as murders, and more than eight times as many assaults as rapes.the variables also have vastly different variances.

The UrbanPop variable measures the percentage of the population in each state living in an urban area, which is not a comparable number to the number of rapes in each state per 100,000 individuals.

If we failed to scale the variables before performing PCA, then most of the principal components that we observed would be driven by the Assault variable, since it has by far the largest mean and variance.

Thus, it is important to standardize the variables to have mean zero and standard deviation one before performing PCA. We now perform principal components analysis using the prcomp() function.

Hide

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

Hide

```
pr.out$center
```

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

Hide

```
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.

Hide

```
pr.out$rotation
```

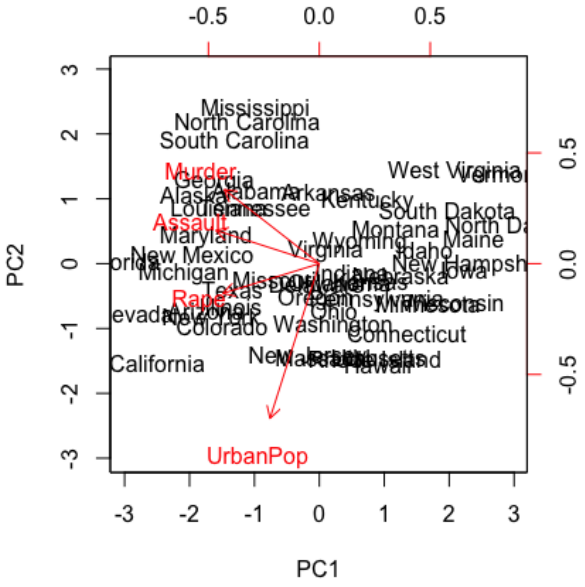
We see that there are four distinct principal components. In general min(n − 1, p) informative principal components in a data set with n observations and p variables.

Using the prcomp() function, we do not need to explicitly multiply the data by the principal component loading vectors in order to obtain the principal component score vectors. Matrix x has as its columns the principal component score vectors. That is, the kth column is the kth principal component score vector.

We can plot the first two principal components as follows

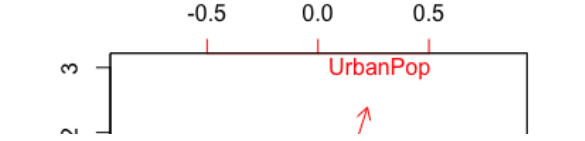
Hide

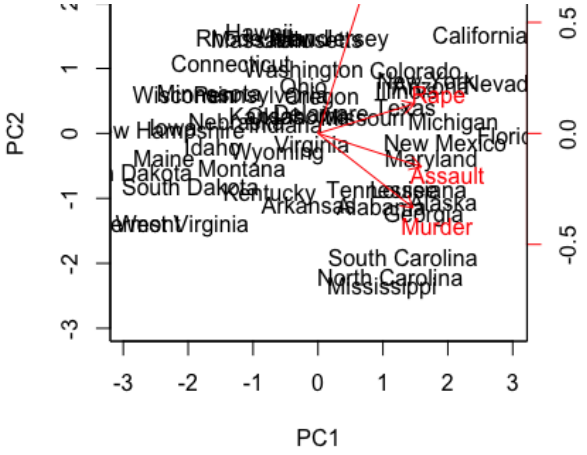
```
biplot(pr.out, scale=0)
```



Hide

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





The prcomp() function also outputs the standard deviation of each principal component.

Hide

```
pr.out$sdev
```

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

Hide

```
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:

Hide

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

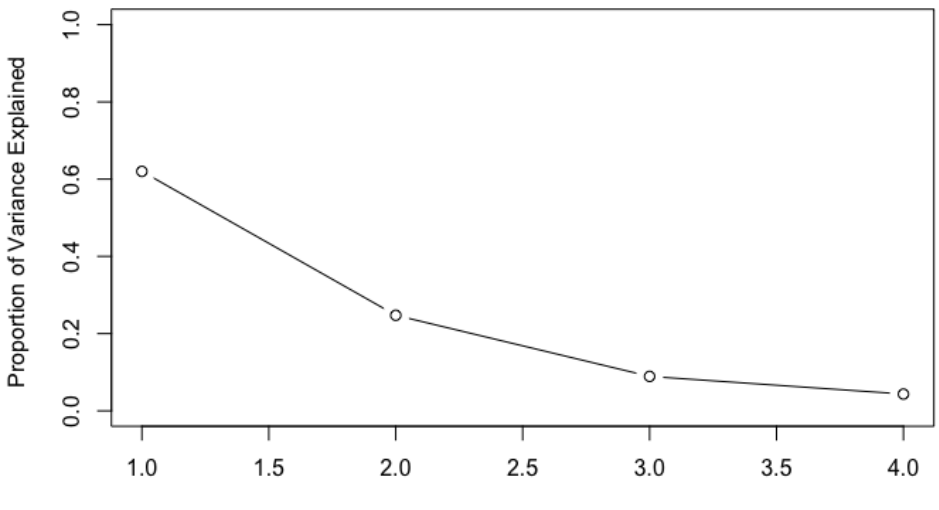
```
[1] 0.62006039 0.24744129 0.08914080 0.04335752
```

We see that the first principal component explains 62.0 % of the variance in the data, the next principal component explains 24.7 % of the variance, and so forth.

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

Hide

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



Principal Component

Hide

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