HW7 Solutions

Dante Wu

2023-03-04

```
pr = read.table("/Users/wugaoyu/Desktop/PhD/TA/Winter 2023/Places_Rated.txt")
pr = pr[-10]
colnames(pr) = c("Climate and Terrain", "Housing","Health Care & the Environment","Crime","Transportati
head(pr)
```

```
##   Climate and Terrain Housing Health Care & the Environment Crime
## 1                 521    6200                           237   923
## 2                 575    8138                          1656   886
## 3                 468    7339                           618   970
## 4                 476    7908                          1431   610
## 5                 659    8393                          1853  1483
## 6                 520    5819                           640   727
##   Transportation Education The Arts Recreation Economics
## 1           4031      2757      996       1405      7633
## 2           4883      2438     5564       2632      4350
## 3           2531      2560      237        859      5250
## 4           6883      3399     4655       1617      5864
## 5           6558      3026     4496       2612      5727
## 6           2444      2972      334       1018      5254
```

Scale raw data

```
pr_scaled = scale(pr)
head(pr_scaled)
```

```
##      Climate and Terrain     Housing Health Care & the Environment       Crime
## [1,]          -0.1467824 -0.89992576                    -0.9458990 -0.10654981
## [2,]           0.3002069 -0.08743661                     0.4688539 -0.21014653
## [3,]          -0.5854941 -0.42241020                    -0.5660393  0.02504601
## [4,]          -0.5192735 -0.18386205                     0.2445273 -0.98292201
## [5,]           0.9955236  0.01946986                     0.6652643  1.46140045
## [6,]          -0.1550600 -1.05965660                    -0.5441052 -0.65533240
##      Transportation  Education   The Arts Recreation  Economics
## [1,]     -0.1234045 -0.1804514 -0.4641863 -0.5458150  1.9434730
## [2,]      0.4637042 -1.1748623  0.5198122  0.9729596 -1.0838164
## [3,]     -1.1570466 -0.7945547 -0.6276834 -1.2216511 -0.2539168
## [4,]      1.8418937  1.8208394  0.3240034 -0.2834024  0.3122592
## [5,]      1.6179379  0.6580957  0.2897530  0.9482037  0.1859300
## [6,]     -1.2169979  0.4897628 -0.6067885 -1.0248417 -0.2502283
```

Analyze scaled data Apply R function from package `psych`

```
library(psych)
pca.scaled = prcomp(pr_scaled, scale = F, center = F)
```

If you didn't scale the data in advance, you can also do this within `prcomp`:

```
pca.scaled = prcomp(pr_scaled, scale = T)
```

See proportion and cumulative proportion of total variance explained by different PCs.

```
summary.pca = summary(pca.scaled)
summary.pca$importance
```

```
##                              PC1      PC2     PC3       PC4       PC5       PC6
## Standard deviation     1.846156 1.101806 1.06840 0.9596446 0.8679199 0.7940793
## Proportion of Variance 0.378700 0.134890 0.12683 0.1023200 0.0837000 0.0700600
## Cumulative Proportion  0.378700 0.513590 0.64042 0.7427400 0.8264400 0.8965000
##                              PC7      PC8     PC9
## Standard deviation     0.7021736 0.563949 0.34699
## Proportion of Variance 0.0547800 0.035340 0.01338
## Cumulative Proportion  0.9512800 0.986620 1.00000
```

```
pov = summary.pca$importance[2,]
cpov = summary.pca$importance[3,]
```

You can also get eigenvalues and eigenvectors from `prcomp` object:

```
evals.scaled = pca.scaled$sdev^2
evals.scaled
```

```
## [1] 3.4082918 1.2139762 1.1414791 0.9209178 0.7532849 0.6305619 0.4930477
## [8] 0.3180385 0.1204021
```

```
evecs.scaled = pca.scaled$rotation
evecs.scaled
```

```
##                                    PC1        PC2          PC3         PC4
## Climate and Terrain          0.2064140  0.2178353 -0.689955982  0.13732125
## Housing                      0.3565216  0.2506240 -0.208172230  0.51182871
## Health Care & the Environment 0.4602146 -0.2994653 -0.007324926  0.01470183
## Crime                        0.2812984  0.3553423  0.185104981 -0.53905047
## Transportation               0.3511508 -0.1796045  0.146376283 -0.30290371
## Education                    0.2752926 -0.4833821  0.229702548  0.33541103
## The Arts                     0.4630545 -0.1947899 -0.026484298 -0.10108039
## Recreation                   0.3278879  0.3844746 -0.050852640 -0.18980082
## Economics                    0.1354123  0.4712833  0.607314475  0.42176994
##                                     PC5         PC6         PC7         PC8
## Climate and Terrain          -0.3691499  0.37460469 -0.08470577 -0.36230833
## Housing                       0.2334878 -0.14163983 -0.23063862  0.61385513
## Health Care & the Environment -0.1032405 -0.37384804  0.01386761 -0.18567612
## Crime                        -0.5239397  0.08092329  0.01860646  0.43002477
## Transportation                0.4043485  0.46759180 -0.58339097 -0.09359866
## Education                    -0.2088191  0.50216981  0.42618186  0.18866756
## The Arts                     -0.1050976 -0.46188072 -0.02152515 -0.20398969
## Recreation                    0.5295406  0.08991578  0.62787789 -0.15059597
## Economics                    -0.1596201  0.03260813 -0.14974066 -0.40480926
##                                        PC9
## Climate and Terrain           0.0013913515
## Housing                       0.0136003402
## Health Care & the Environment -0.7163548935
## Crime                        -0.0586084614
## Transportation                0.0036294527
```

```
## Education                         0.1108401911
## The Arts                          0.6857582127
## Recreation                       -0.0255062915
## Economics                         0.0004377942
```

```r
pov.scaled = evals.scaled/sum(evals.scaled)
pov.scaled
```

```
## [1] 0.37869909 0.13488624 0.12683102 0.10232420 0.08369832 0.07006243 0.05478308
## [8] 0.03533761 0.01337801
```

```r
cpov.scaled = cumsum(pov.scaled)
cpov.scaled
```

```
## [1] 0.3786991 0.5135853 0.6404163 0.7427405 0.8264389 0.8965013 0.9512844
## [8] 0.9866220 1.0000000
```

The third way is to find eigenvalues and eigenvectors manually.

```r
S = cov(pr_scaled)
eigen(S)
```

```
## eigen() decomposition
## $values
## [1] 3.4082918 1.2139762 1.1414791 0.9209178 0.7532849 0.6305619 0.4930477
## [8] 0.3180385 0.1204021
##
## $vectors
##             [,1]       [,2]         [,3]        [,4]       [,5]        [,6]
##  [1,] -0.2064140  0.2178353  0.689955982  0.13732125  0.3691499 -0.37460469
##  [2,] -0.3565216  0.2506240  0.208172230  0.51182871 -0.2334878  0.14163983
##  [3,] -0.4602146 -0.2994653  0.007324926  0.01470183  0.1032405  0.37384804
##  [4,] -0.2812984  0.3553423 -0.185104981 -0.53905047  0.5239397 -0.08092329
##  [5,] -0.3511508 -0.1796045 -0.146376283 -0.30290371 -0.4043485 -0.46759180
##  [6,] -0.2752926 -0.4833821 -0.229702548  0.33541103  0.2088191 -0.50216981
##  [7,] -0.4630545 -0.1947899  0.026484298 -0.10108039  0.1050976  0.46188072
##  [8,] -0.3278879  0.3844746  0.050852640 -0.18980082 -0.5295406 -0.08991578
##  [9,] -0.1354123  0.4712833 -0.607314475  0.42176994  0.1596201 -0.03260813
##              [,7]        [,8]          [,9]
##  [1,]  0.08470577  0.36230833 -0.0013913515
##  [2,]  0.23063862 -0.61385513 -0.0136003402
##  [3,] -0.01386761  0.18567612  0.7163548935
##  [4,] -0.01860646 -0.43002477  0.0586084614
##  [5,]  0.58339097  0.09359866 -0.0036294527
##  [6,] -0.42618186 -0.18866756 -0.1108401911
##  [7,]  0.02152515  0.20398969 -0.6857582127
##  [8,] -0.62787789  0.15059597  0.0255062915
##  [9,]  0.14974066  0.40480926 -0.0004377942
```

```r
evals = eigen(S)$values
pov.scaled = evals / sum(evals)
pov.scaled
```

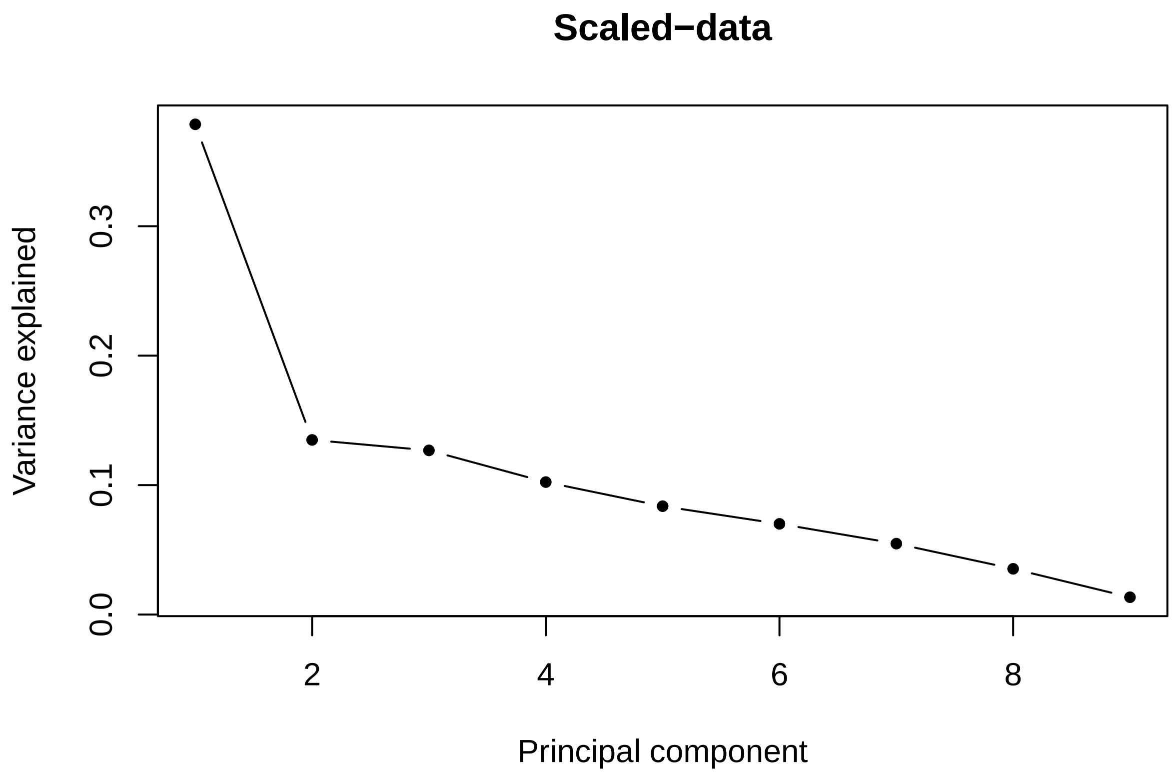
```
## [1] 0.37869909 0.13488624 0.12683102 0.10232420 0.08369832 0.07006243 0.05478308
## [8] 0.03533761 0.01337801
```

```r
cpov.scaled = cumsum(pov.scaled)
cpov.scaled
```

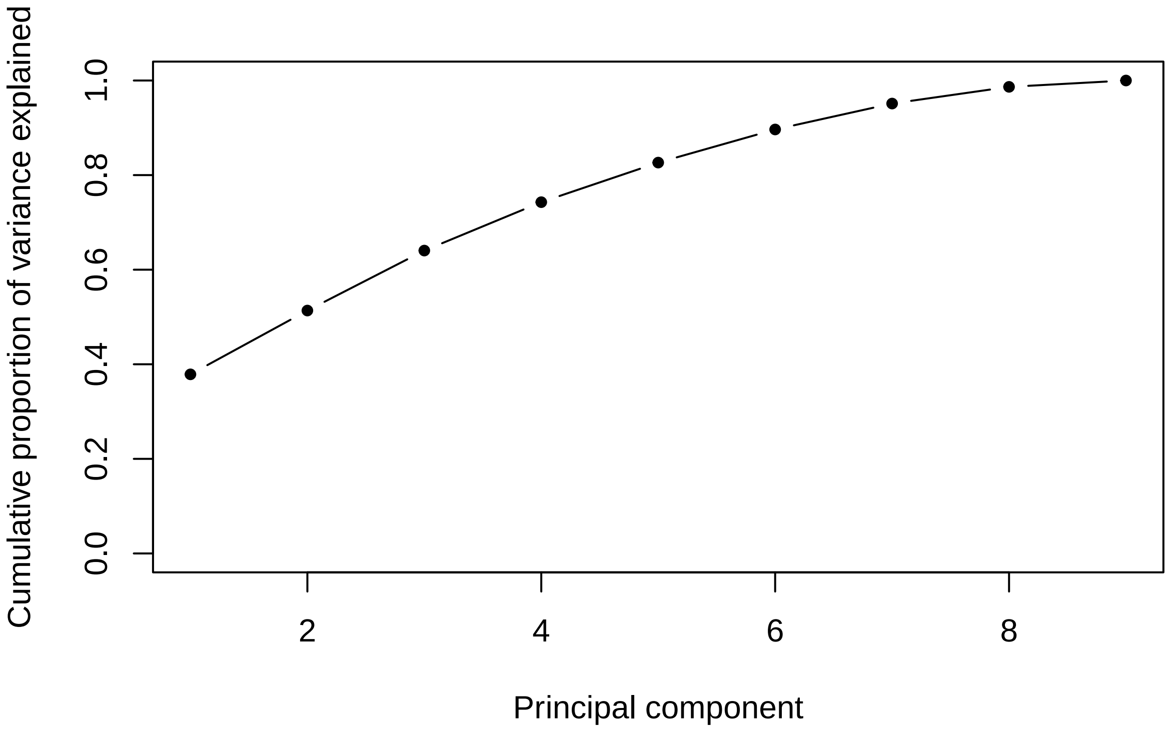
```
## [1] 0.3786991 0.5135853 0.6404163 0.7427405 0.8264389 0.8965013 0.9512844
## [8] 0.9866220 1.0000000
```

Draw scree plot and cumulative plot for PCA of scaled data

```
#par(mfrow=c(1,2))
plot(pov, pch = 20, type = "b", ylab = "Variance explained", xlab = "Principal component", main = "Scal
```

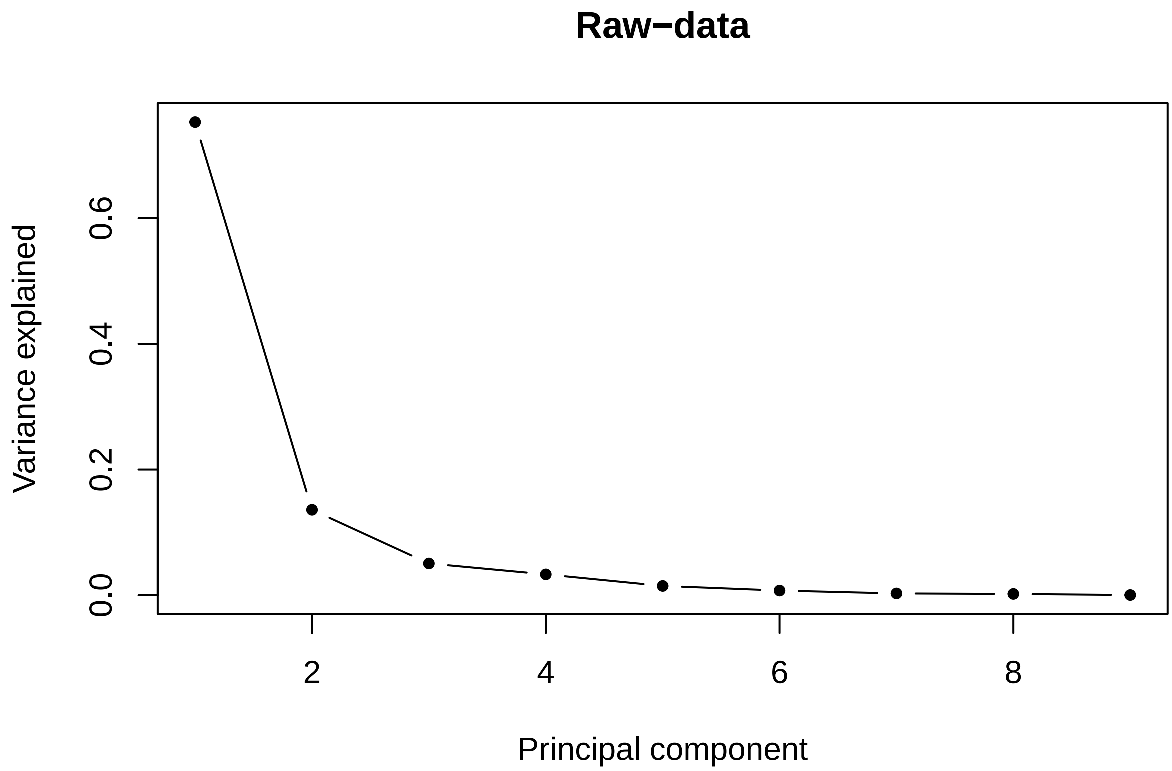



```
plot(cpov, pch = 20, type = "b", ylab = "Cumulative proportion of variance explained",xlab = "Principal
```

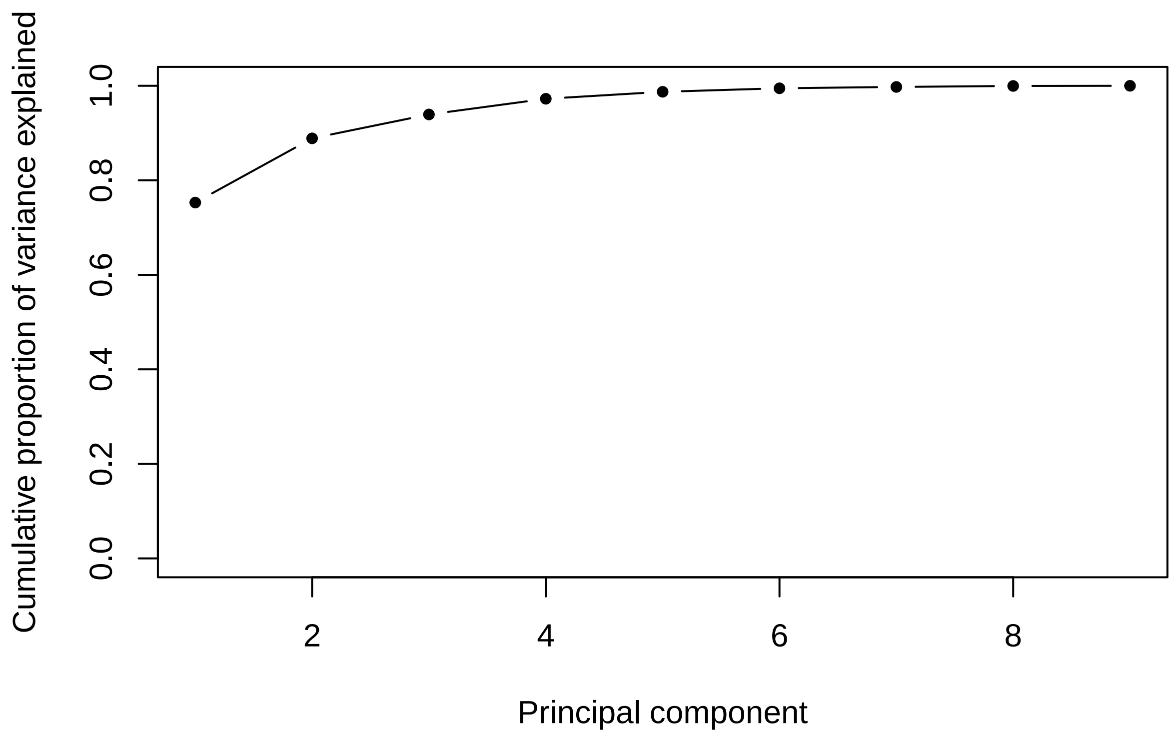


Repeat the same procedure for raw data

```
pca.raw= prcomp(pr, scale = F)
summary_pca.raw = summary(pca.raw)
pov.raw = summary_pca.raw$importance[2,]
cpov.raw = summary_pca.raw$importance[3,]
#par(mfrow=c(1,2))
plot(pov.raw, pch = 20, type = "b", ylab = "Variance explained",xlab = "Principal component",main = "Raw
```




```
plot(cpov.raw, pch = 20, type = "b", ylab = "Cumulative proportion of variance explained",xlab = "Princ
```



Also, remember to include eigenvalues and eigenvectors in your homework.

Determine the number of principal components

Make the Cumulative proportion of variance explained larger than certain threshold, e.g. 0.75, 0.8, 0.9, etc.

For example:

```
k = which.max(cpov.scaled >= 0.8)
k
```

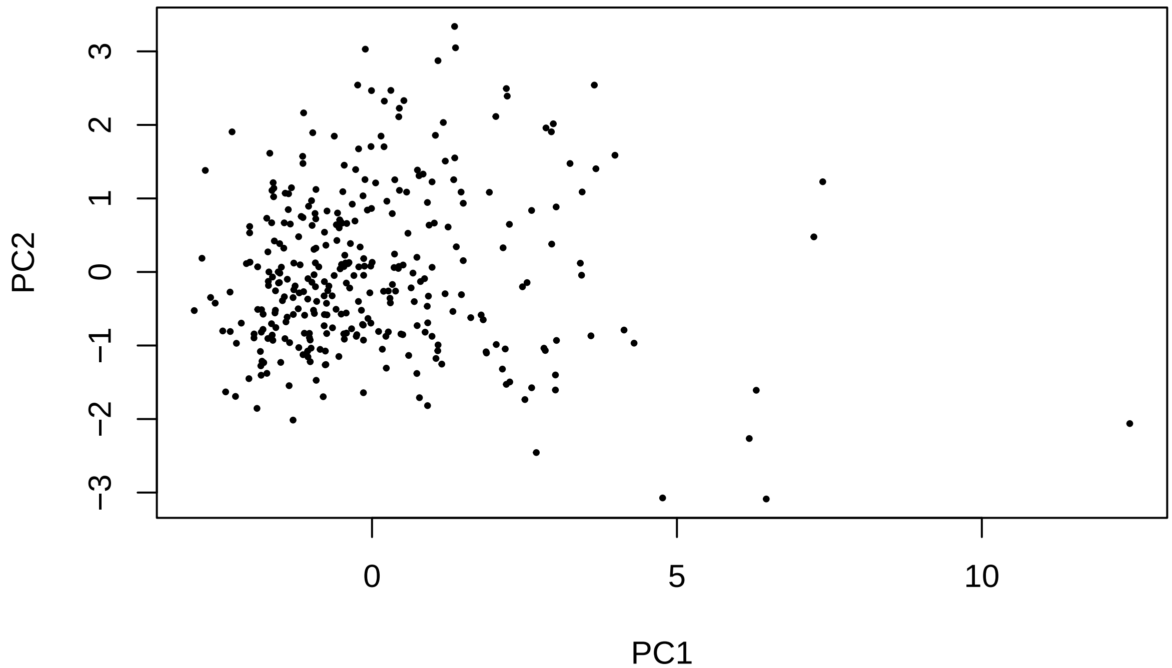
```
## [1] 5
```

So we choose the first 5 principle components.

Do projection

Get projection results from `prcomp` object

```
pcs = pca.scaled$x
plot(x = pcs[,1], y = pcs[,2], cex = 0.5, pch = 20, ylab = "PC2", xlab = "PC1")
```

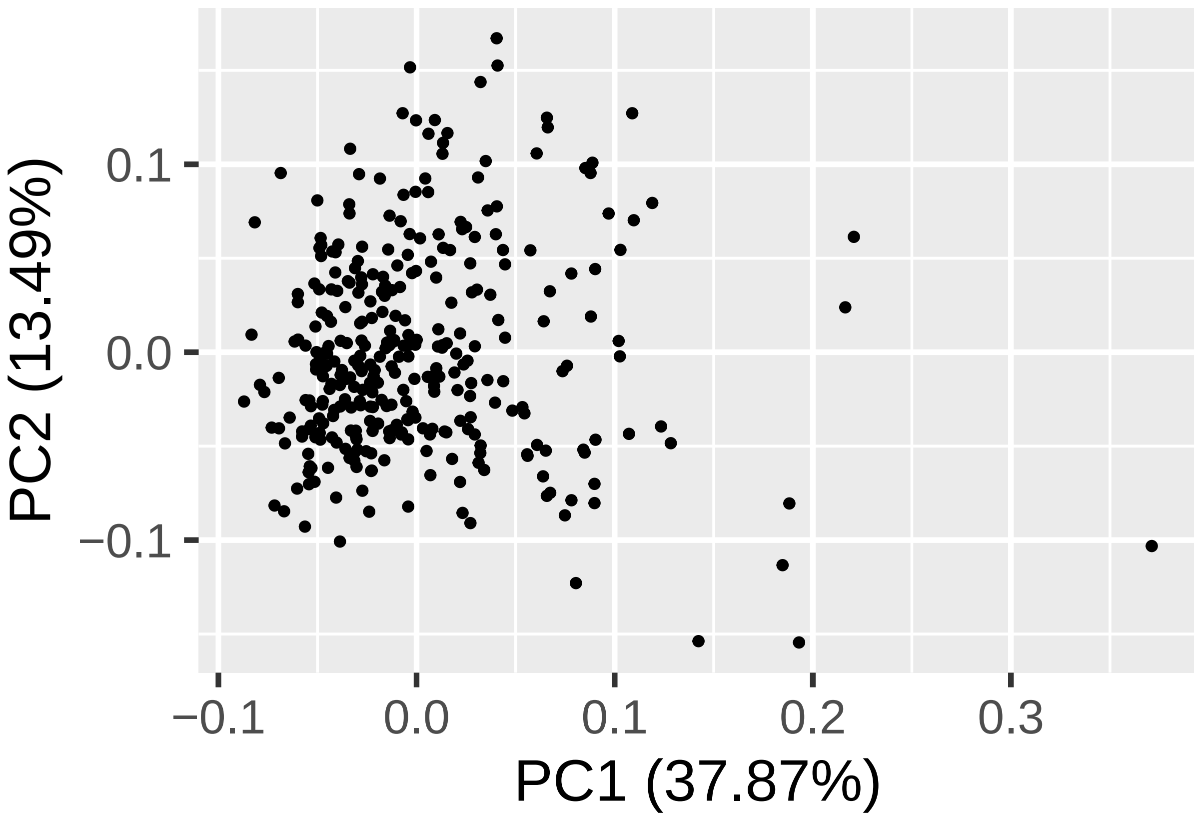


You can also use `autoplot`:

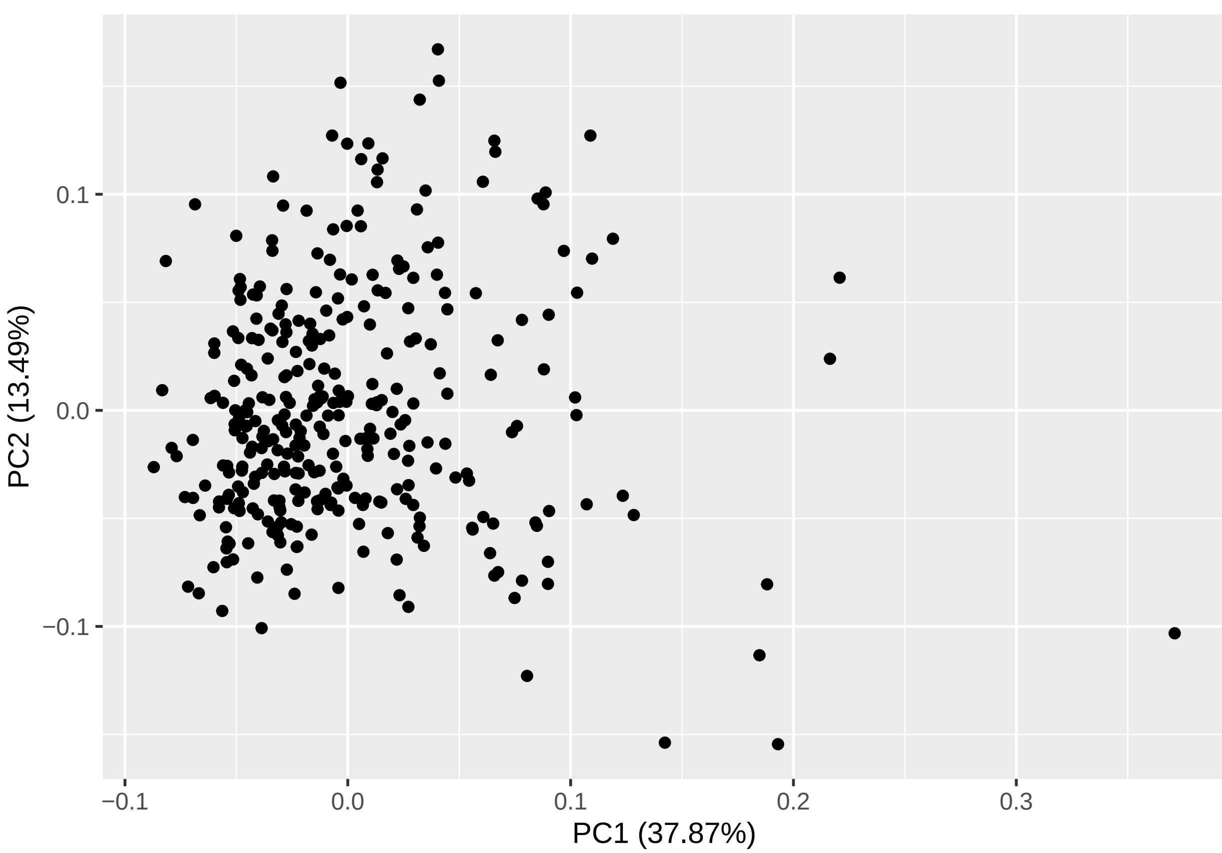
```
library(ggfortify)
```

```
## Loading required package: ggplot2

## Warning in register(): Can't find generic `scale_type` in package ggplot2 to
## register S3 method.

##
## Attaching package: 'ggplot2'

## The following objects are masked from 'package:psych':
##
##     %+%, alpha
```

```
plot_0 = autoplot(pca.scaled, data = pr, color = 'black')
plot_0+ theme_grey(base_size = 22)
```


PC2 (13.49%)
0.1
0.0
−0.1
−0.1
0.0
0.1
0.2
0.3
PC1 (37.87%)

```
plot_0
```


PC2 (13.49%)
0.1
0.0
−0.1
−0.1
0.0
0.1
0.2
0.3
PC1 (37.87%)

Compute projection manually; Loading vectors are estimated by eigenvector

```
Vec_1 = eigen(cov(pr_scaled))$vectors[,1]
Vec_2 = eigen(cov(pr_scaled))$vectors[,2]

PC1 = as.numeric(pr_scaled %*% Vec_1)
PC2 = as.numeric(pr_scaled %*% Vec_2)
plot(x = PC1, y = PC2, cex = 0.5, pch = 20, ylab = "PC2", xlab = "PC1")
```