Homework 3 - Principal Component Analysis

Daniel Carpenter

August 2022

Table of contents

Check list:

- 1 (iv - v)

Packages

```r
library(tidyverse) # get tidverse for piping
library(skimr)
library(knitr)
library(scales)
require(lubridate)

library(mlbench)       # Glass data
library(ggbiplot)      # biplots
library(corrplot)
library(caret)
```

1. Glass Data

Get and Clean Data

```r
data(Glass)

# Remove duplicates
Glass <- Glass[!duplicated(Glass), ]
```

(a) Mathematics of PCA

`i.` Create the correlation matrix of all the numerical attributes in the `Glass` data and store the results in a new object `corMat`

```r
skimmed <- skim(Glass)

# Notice one factor data, for variable `type`
skimmed$skim_type
```

```
 [1] "factor"  "numeric" "numeric" "numeric" "numeric" "numeric" "numeric"
 [8] "numeric" "numeric" "numeric"
```

```r
# Get only numeric data
GlassNumeric <- Glass %>% select(where(is.numeric))
```

```r
# Create correlation matrix using only numeric data type
corMat <- cor(GlassNumeric)
```

ii. Compute the eigenvalues and eigenvectors of `corMat`.

Eigenvalues

```r
# prcomp(corMat)
eigenValues = eigen(corMat)$values
eigenValues
```

```
[1] 2.510152168 2.058169337 1.407484057 1.144693344 0.914768873 0.528593040
[7] 0.370262639 0.064267543 0.001608997
```

Eigenvectors

```r
eigenVectors = eigen(corMat)$vectors
eigenVectors
```

```
            [,1]        [,2]        [,3]       [,4]        [,5]        [,6]
 [1,]  0.5432231 -0.28911804 -0.08849541  0.1479796  0.07670808 -0.11455615
 [2,] -0.2676141 -0.26909913  0.36710090  0.5010669 -0.14626769  0.55790564
 [3,]  0.1093261  0.59215502 -0.02295318  0.3842440 -0.11610001 -0.30585293
 [4,] -0.4269512 -0.29636272 -0.32602906 -0.1488756 -0.01720068  0.02014091
 [5,] -0.2239232  0.15874450  0.47979931 -0.6394962 -0.01763694 -0.08850787
 [6,] -0.2156587  0.15305116 -0.66349177 -0.0733491  0.30154622  0.24107648
 [7,]  0.4924367 -0.34678973  0.01380151 -0.2743430  0.18431431  0.14957911
 [8,] -0.2516459 -0.48262056 -0.07649040  0.1299431 -0.24970936 -0.65986429
 [9,]  0.1912640  0.06089167 -0.27223834 -0.2252596 -0.87828176  0.24066617
             [,7]        [,8]        [,9]
 [1,] -0.08223530  0.75177166 -0.02568051
 [2,] -0.15419352  0.12819398  0.31188932
 [3,]  0.20691746  0.07799332  0.57732740
 [4,]  0.69982052  0.27334224  0.19041178
 [5,] -0.20945417  0.38077660  0.29747147
 [6,] -0.50515516  0.11064442  0.26075531
 [7,]  0.09984144 -0.39885229  0.57999243
 [8,] -0.35043794 -0.14497643  0.19853265
 [9,] -0.07120579  0.01650505  0.01459278
```

`iii.` Use `prcomp` to compute the principal components of the `Glass` attributes (make sure to use the scale option).

```
# Using only numeric data
pc.glass <- prcomp(GlassNumeric, scale = TRUE)
pc.glass
```

```
Standard deviations (1, .., p=9):
[1] 1.58434597 1.43463213 1.18637433 1.06990343 0.95643550 0.72704404 0.60849210
[8] 0.25351044 0.04011231

Rotation (n x k) = (9 x 9):
          PC1         PC2         PC3        PC4         PC5         PC6
RI -0.5432231  0.28911804 -0.08849541 -0.1479796  0.07670808 -0.11455615
Na  0.2676141  0.26909913  0.36710090 -0.5010669 -0.14626769  0.55790564
Mg -0.1093261 -0.59215502 -0.02295318 -0.3842440 -0.11610001 -0.30585293
Al  0.4269512  0.29636272 -0.32602906  0.1488756 -0.01720068  0.02014091
Si  0.2239232 -0.15874450  0.47979931  0.6394962 -0.01763694 -0.08850787
K   0.2156587 -0.15305116 -0.66349177  0.0733491  0.30154622  0.24107648
Ca -0.4924367  0.34678973  0.01380151  0.2743430  0.18431431  0.14957911
Ba  0.2516459  0.48262056 -0.07649040 -0.1299431 -0.24970936 -0.65986429
Fe -0.1912640 -0.06089167 -0.27223834  0.2252596 -0.87828176  0.24066617
           PC7         PC8         PC9
RI -0.08223530 -0.75177166 -0.02568051
Na -0.15419352 -0.12819398  0.31188932
Mg  0.20691746 -0.07799332  0.57732740
Al  0.69982052 -0.27334224  0.19041178
Si -0.20945417 -0.38077660  0.29747147
K  -0.50515516 -0.11064442  0.26075531
Ca  0.09984144  0.39885229  0.57999243
Ba -0.35043794  0.14497643  0.19853265
Fe -0.07120579 -0.01650505  0.01459278
```

`iv.` Compare the results from (ii) and (iii) - Are they the same? Different? Why?

- The eigenvalues differ
- The eigenvectors are the same in absolute value, but the signs are the opposite within each value of the vectors
- Why do they differ? Past `ii` uses the correlation matrix; the principal component analysis (`ii`) uses the covariance matrix, which is a scaled, or *normalized*, version of the correlation matrix.

v. Using R demonstrate that principal components 1 and 2 from (iii) are orthogonal. (Hint: the inner product between two vectors is useful in determining the angle between the two vectors)

```
PC1.glass <- pc.glass$x[,1]
PC2.glass <- pc.glass$x[,2]

angle <- acos( sum(PC1.glass*PC2.glass) / ( sqrt(sum(PC1.glass * PC1.glass)) * sqrt(sum(PC

angle
```

```
[1] 1.570796
```

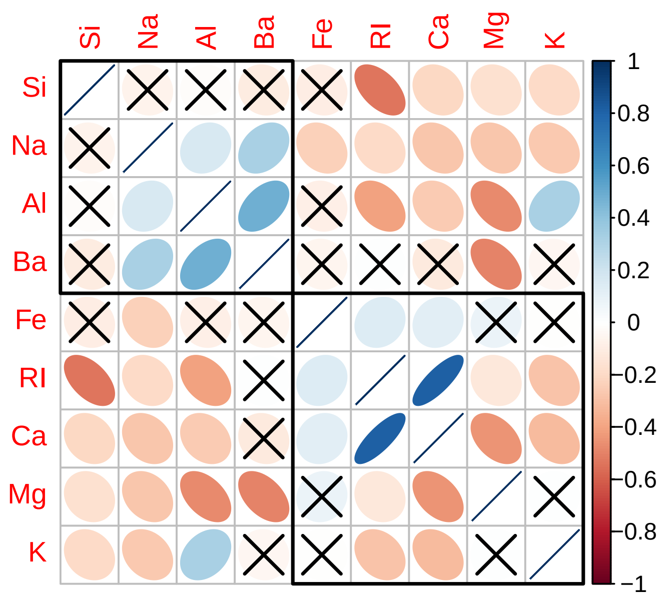
(b) Applications of PCA

i. Create a visualization of the corMat correlation matrix (i.e., a heatmap or variant).

- `corrplot` options.

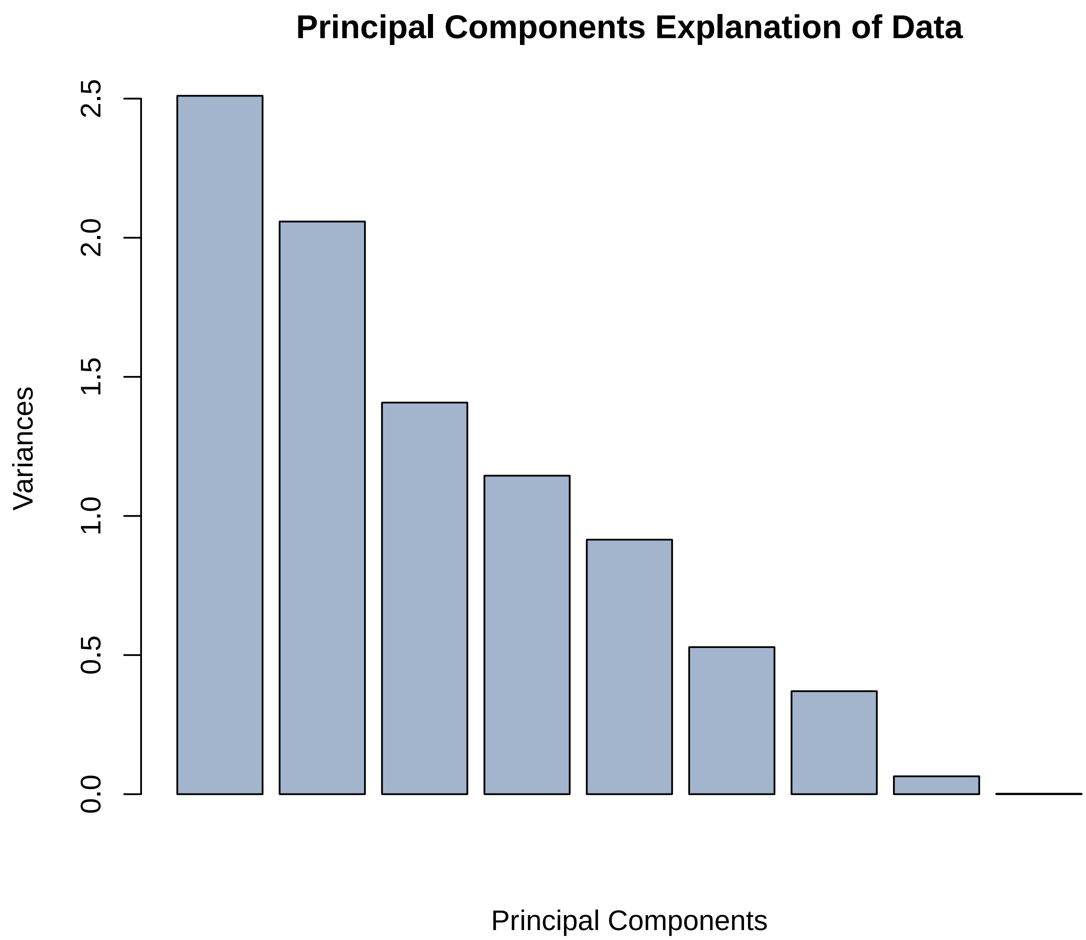
```r
testRes = cor.mtest(GlassNumeric, conf.level = 0.90)


# Correlation matrix to show spread and significance
corrplot(corMat,
         p.mat      = testRes$p, # Significance 'x' marks
         sig.level = 0.10,       # ""             levels
         order      = 'hclust',  # Clustering
         addrect    = 2,
         method     = 'ellipse') # Show spread and direction
```



ii. Provide visualizations of the principal component analysis results from the Glass data. Consider incorporating the glass type to group and color your biplot.

```r
# First show the spread of the components
plot(pc.glass,
     main = 'Principal Components Explanation of Data',
     xlab = 'Principal Components',
     col  = 'lightsteelblue3'
     )
```



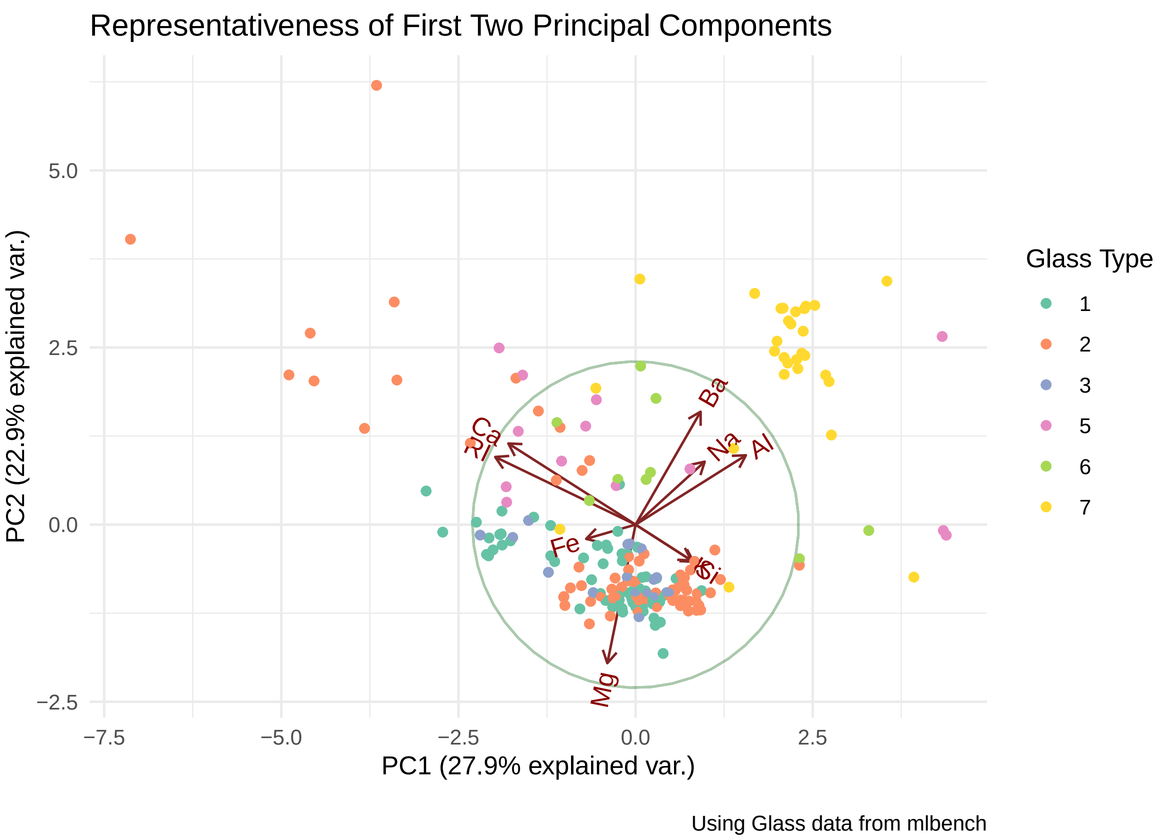
```r
# NExt show the biplots
ggbiplot(pc.glass,
         obs.scale    = 1,
         var.scale    = 1,
         varname.size = 4,
         labels.size  = 10,
         circle       = TRUE,
         group        = Glass$Type#,
         # ellipse      = TRUE
         ) +

  # Titles and caption
  labs(title   = 'Representativeness of First Two Principal Components',
       caption = '\nUsing Glass data from mlbench') +

  # Add color to points by glass type
  geom_point(aes(colour=Glass$Type), size = 1) +

  # Categorical palette on glass type
  scale_color_brewer(name = 'Glass Type',
                     palette = 'Set2', type = 'qual') +

  theme_minimal() # the theme
```



iii. Provide an interpretation of the first two prinicpal components the Glass data.

- Both PC1 and PC2 represent roughly half (50%) of the cumulative proportion of variance (see summary below)
- PC1 best explains `Fe`, `K`, and `Si` glass types, since they lie closest to parallel with the x axis
- PC2 best represents `Ba`, and `Mg`, since they lie close to parallel with the y axis.
- Other variables appear to be explained by both principal components, since they are near a 45 degree angle.

Summary of cumulative proportion located here

```r
summary(pc.glass)
```

```
Importance of components:
                          PC1    PC2    PC3    PC4    PC5     PC6     PC7
```

```
Standard deviation     1.5843 1.4346 1.1864 1.0699 0.9564 0.72704 0.60849
Proportion of Variance 0.2789 0.2287 0.1564 0.1272 0.1016 0.05873 0.04114
Cumulative Proportion  0.2789 0.5076 0.6640 0.7912 0.8928 0.95154 0.99268
                           PC8     PC9
Standard deviation     0.25351 0.04011
Proportion of Variance 0.00714 0.00018
Cumulative Proportion  0.99982 1.00000
```

iv. Based on the PCA results, do you believe that you can effectively reduce the dimension of the data? If so, to what degree? If not, why?

- Given the cumulative proportions above, it is clear that the first two principal components capture only half (~50%) of the variation in the original data. We could compare that to a coin flip, or a random chance.
- However, the *first four* PC's capture roughly 80%. This cuts the number of variables in half, which is impressive.
- Note that if your `q` threshold was set to 95%, then this analysis would not perform well, since all but one of the PC's capture 95% of the variation in the actual data.

(c) Application of LDA

i. Since the Glass data is grouped into various labeled glass types we can consider linear discriminant analysis (LDA) as another form of dimension reduction. Use the lda method from the MASS package to reduce the Glass data dimensionality.

ii. How would you interpret the first discriminant function, LD1?

iii. Use the ldahist function from the MASS package to visualize the results for LD1 and LD2. Comment on the results.

2. Principal components for dimension reduction

3. Housing data dimension reduction and exploration