CHAPTER 4

Visualization of Principal Components

OBJECTIVE

The primary objective of this chapter is to present various techniques for visualizing several key statistics produced by the Principal Component Analysis (PCA). The first section will focus on the scree and cumulative variance plots. These plots depict the percent of total variance that each synthetic variable accounts for. Next, I will present score plots, which you will use to display clusters of data defined by PC scores, in two-dimensional graphs. You will also learn how to create loading and variable contribution plots, which depict the influence that each original variable has on the principal components retained for analysis. These important visualization techniques are an integral part of any PCA project, and all analysts must know them.

Contents

4.1 *Introduction* 84
4.2 *Useful Packages* 87
4.3 *Scree & Cumulative Variance Plots* 89
4.4 *PCA Score Plot* 98
4.5 *Loadings and Variable Contribution Plots* 103
4.6 *Concluding Remarks* 112

4.1 Introduction

In this chapter, I will present important graphs often used to visualize various aspects of principal component analysis. I will use the Breast Cancer Wisconsin (BCW) dataset for illustrating these graphs. The BCW data set contains breast tumor diagnosis data, where each of the 569 records represents the image of a breast mass aspirate. The 32 attributes associated with each record are the following:

1) ID number (the aspirate identifier),
2) Diagnosis (M=malignant, B=benign),
3) 30 additional attributes are obtained as follows:

   The `mean`, `standard error`, and the `worst` (i.e. the average of the 3 largest values) of each of the following 10 characteristics are calculated:

   a) `radius` (mean of distances from center to points on the perimeter),
   b) `texture` (standard deviation of gray-scale values),
   c) `perimeter`,
   d) `area`,
   e) `smoothness` (local variation in radius lengths),
   f) `compactness` (perimeter^2/area $-$ 1.0),
   g) `concavity` (severity of concave portions of the contour),
   h) `concave points` (number of concave portions of the contour),
   i) `symmetry`,
   j) `fractal dimension` ("coastline approximation" - 1).

You may download this data set in a `.csv` text format using the link https://bit.ly/422TkFs, or in Excel format with the link https://bit.ly/40qp1YZ. Note that the CSV and Excel files do not include column labels. Therefore, column labels need to be assigned programmatically as shown in Program 4.1, to be discussed in the next paragraph. You can also get more information about this data set in section 1.5 of chapter #1, or by visiting the web page https://bit.ly/45cKE19.

The following 18-line R script (Program 4.1) performs PCA based on the centered and standardized `WDBC` dataset[1]. Line `#01` loads some third-party packages that I will discuss in section 4.2. For now, simply install these packages

[1]You may download this script using the link: https://bit.ly/467SmsE.

(including the `pacman` package) if they are not already installed, in order to make the script work.

A key part of this program is the reading of the input dataset `wdbc.xlsx` in lines `#03-#12`. I first define the working directory `wdir` in line `#05`, where the input dataset is located. Then the `read.xlsx()` function from the `xlsx` package is used in lines `#06` and `#07` to read the input data into the `wdbc.df` data frame. In lines `#08-#12`, I create all 32 variable names and attach them to the data frame.

Program 4.1. Reading the BCW input dataset and performing the PCA

```
01 pacman::p_load(factoextra,tidyverse,xlsx)
02
03 #-- Reading the input dataset
04
05 wdir <- "C:\\Users\\...\\PCA-Using-R\\datasets\\breast-cancer-data"
06 wdbc.df <- read.xlsx(file=paste0(wdir,"\\wdbc.xlsx"),
07                       sheetName = "wdbc",header = FALSE)
08 features <- c("radius","texture","perimeter","area","smoothness",
09               "compactness","concavity","concave_points","symmetry",
10               "fractal_dimension")
11 names(wdbc.df) <- c("id","diagnosis",paste0(features,"_mean"),
12                   paste0(features,"_se"),paste0(features,"_worst"))
13
14 #-- Performing the PCA
15
16 input.df <- select(wdbc.df,-c(id,diagnosis))
17 wdbc.pca <- prcomp(input.df, center=TRUE, scale=TRUE)
18 round(summary(wdbc.pca)$importance,4)
```

End of Script

Note that `id`, and `diagnosis` are 2 variables in the `wdbc.df` data frame that are not quantitative. Therefore, both must be excluded from the data frame before PCA can be performed. That task is accomplished in line `#16`. The resulting data frame `input.df` without the 2 categorical variables is used as argument in function `prcomp()` of line `#17` to perform the PCA. This operation produced the list object `wdbc.pca`, which is summarized in line `#18`. This summary is presented in Output 4.1.1 and shows the 30 principal components along with the proportion of variance they account for and the associated cumulative proportions.

It follows from Output 4.1.1 that PC_1 alone explains about 44.27% of total variance, whereas this proportion decreases to 18.97% for PC_2. There is a steady decrease of this proportion for subsequent principal components to the point where it reaches almost 0% for PC #30. However, both PC_1 and PC_2 explain cumulatively about 63.24% of total variance. This cumulative proportion increases with the addition of more PCs and will reach 100% with PC_{30}.

Output 4.1.1. PCA summary of the WDBC dataset

```
> round(summary(wdbc.pca)$importance,4)
                            PC1    PC2    PC3    PC4    PC5    PC6
Standard deviation     3.6444 2.3857 1.6787 1.4074 1.2840 1.0988
Proportion of Variance 0.4427 0.1897 0.0939 0.0660 0.0550 0.0402
Cumulative Proportion  0.4427 0.6324 0.7264 0.7924 0.8473 0.8876
                            PC7    PC8    PC9   PC10   PC11   PC12
Standard deviation     0.8217 0.6904 0.6457 0.5922 0.5421 0.5110
Proportion of Variance 0.0225 0.0159 0.0139 0.0117 0.0098 0.0087
Cumulative Proportion  0.9101 0.9260 0.9399 0.9516 0.9614 0.9701
                           PC13   PC14   PC15   PC16   PC17   PC18
Standard deviation     0.4913 0.3962 0.3068 0.2826 0.2437 0.2294
Proportion of Variance 0.0080 0.0052 0.0031 0.0027 0.0020 0.0018
Cumulative Proportion  0.9781 0.9833 0.9865 0.9892 0.9911 0.9929
                           PC19   PC20   PC21   PC22   PC23   PC24
Standard deviation     0.2224 0.1765 0.1731 0.1656 0.1560 0.1344
Proportion of Variance 0.0016 0.0010 0.0010 0.0009 0.0008 0.0006
Cumulative Proportion  0.9945 0.9956 0.9966 0.9975 0.9983 0.9989
                           PC25   PC26   PC27   PC28   PC29   PC30
Standard deviation     0.1244 0.0904 0.0831 0.0399 0.0274 0.0115
Proportion of Variance 0.0005 0.0003 0.0002 0.0000 0.0000 0.0000
Cumulative Proportion  0.9994 0.9997 0.9999 1.0000 1.0000 1.0000
```

End of Output

Output 4.1.2 on the other hand, shows you the structure of the PCA list object `wdbc.pca`, which contains a total of 5 elements. For example, `sdev` is a 30-element numeric vector whose first few values are also displayed. Next, `rotation` is numeric matrix. It has 2 dimensions as well as named rows and columns. The first 2 row names are `radius_mean` and `texture_mean`, whereas the first 4 column names are `PC1-PC4`. The remaining 3 list elements are `center`, a 30-element named numeric vector[2], `scale`, a 30-element named numeric vec-

[2] A vector is named when each of its elements has a name.

tor, and x, a 569 × 30 numeric matrix of PC scores, with named columns and unnamed rows. I will further discuss the content of these list elements in subsequent sections.

Output 4.1.2. Structure of the list object "wdbc.pca" from Program 4.1

```
> str(wdbc.pca)
List of 5
 $ sdev    : num [1:30] 3.64 2.39 1.68 1.41 1.28 ...
 $ rotation: num [1:30, 1:30] -0.219 -0.104 -0.228 -0.221 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : chr [1:30] "radius_mean" "texture_mean"  ...
  .. ..$ : chr [1:30] "PC1" "PC2" "PC3" "PC4" ...
 $ center  : Named num [1:30] 14.1273 19.2896 91.969  ...
  ..- attr(*, "names")= chr [1:30] "radius_mean" ...
 $ scale   : Named num [1:30] 3.524 4.301 24.299 351.9141 ...
  ..- attr(*, "names")= chr [1:30] "radius_mean" ...
 $ x       : num [1:569, 1:30] -9.18 -2.39 -5.73 -7.12 -3.93 ...
  ..- attr(*, "dimnames")=List of 2
  .. ..$ : NULL
  .. ..$ : chr [1:30] "PC1" "PC2" "PC3" "PC4" ...
 - attr(*, "class")= chr "prcomp"
```

End of Output

4.2 Useful Packages

In chapter #3, I used the 2 functions `prcomp()` and `princomp()` to perform Principal Component Analysis (PCA). These 2 functions are provided by the R built-in "stats" package, which does not require any installation. Although there are a few other third-party packages that can perform PCA, what the "stats" package offers is sufficient to meet our needs. However, this package cannot produce good graphs. Therefore, I will appeal to third-party packages to obtain the graphs that I want.

To visualize PCA results, I recommend using the `factoextra` package. You can access this package's online documentation with the link https://rpkgs.datanovia.com/factoextra/index.html. Note that this package provides a series of functions for visualizing not only PCA outputs, but also the outputs of

many other multivariate analysis techniques such as Correspondence Analysis, Factor Analysis and more. I will only focus on PCA functions in this chapter.

As with any other R package, before you can use `factoextra`, you need to install it, then load it to your R session. Here are the 2 ways that you can install this package:

- From CRAN:
  At the console, type the command: `install.packages("factoextra")`.

- Alternatively, you can install the development version from GitHub[3]. This option requires that you first install the package `devtools` using the `install.packages()` function if not installed yet. Then run the command, `devtools::install_github("kassambara/factoextra")`.

  I would recommend using the development version, as it usually has more features. Since it may not have been tested as much as the CRAN version, you should use it with some caution.

In addition to the `factoextra` package, I will need the 2 packages `tidyverse` and `xlsx`. `Tidyverse` is a major R package, used for advanced statistical analysis, which you cannot ignore if you are going to do any serious work with R. Know that `tidyverse` is actually a major collection of open source packages for the R programming language. Its use in this chapter is rather limited. I will use it here primarily for its graphics capabilities. If you are interested in learning more about `tidyverse`, one of the best resources is the link https://www.tidyverse.org/. I will also need the `xlsx` package, mainly for manipulating Excel files.

After installing a package, you must load it to your R session before it can be used. Loading a package to an R session is generally done with the `library()` function as `library(PackageName)`. If you need to load several packages, you must repeat the same call of the `library()` function multiple times. To avoid this repetition, I used the `p_load()` function in line `#01` of Program 4.1 to load 3 different packages at once. But this function is defined in the third-party package `pacman`, which must first be installed. Since `p_load()` is the only function from this package that I want to use, I decided not to load the entire package. Instead I used the :: operator to call the `p_load()` function as follows: `pacman::p_load()`.

[3]To learn more about the use of GitHub with R, read the book "Using R for Excel Analysts" by Gwet (2022).

4.3 Scree & Cumulative Variance Plots

A scree plot displays how much variation in the data is captured by each principal component. The scree plot can either display the variance absolute values such as in Figure 4.1, or the proportion of total variance explained by each individual principal component such as in Figure 4.2.

Figure 4.1 is a scree plot associated with the PCA of section 4.1. Although there are 30 PCs, I decided to display the first 20 only, as the variance of each of the remaining components is very small. As a matter of fact, only the first 6 PCs have a variance that exceeds 1. Consequently, as of PC_7 all principal components have a variance smaller than 1, which is the variance of each standardized original variable. For further analysis, it is recommended to use no more than the first 6 PCs. This scree plot is a steep curve that bends quickly after the 6^{th} PC and flattens out. The variance values used in Figure 4.1 can be obtained by taking the squared standard deviations of Output 4.1.1.

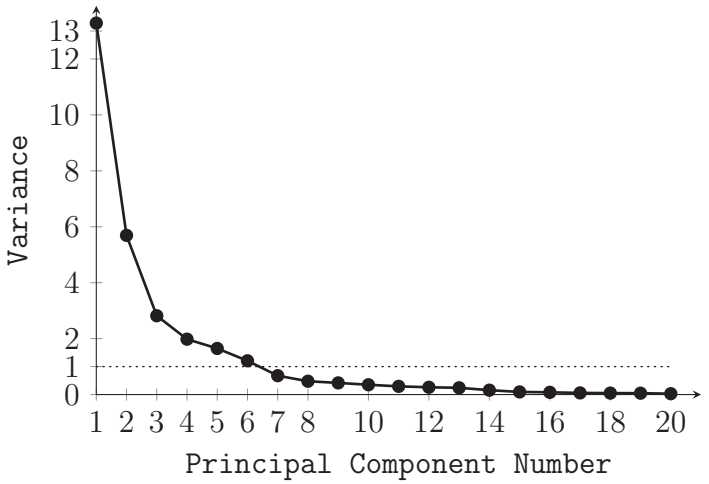

Figure 4.1: Screeplot of the principal component score variances showing the absolute value of variance explained, for the BCW dataset

Figure 4.2 is a special scree plot, which is a combination of the line and bar charts. Moreover, the vertical axis shows the proportion of total variance explained by each PC. Both figures 4.1 and 4.2 display the same information that is presented differently. Therefore using one of the other is a matter of preference.

Figure 4.3 is the cumulative variance plot. It shows the PC number on the

horizontal axis, and the cumulative percent of total variance (CPTV) on the vertical axis. PC #4 for example, is associated with a CPTV of 0.8. That is, the first 4 PCs explain about 80% of total variation in your data. Any analysis you conduct based on the first 4 PCs will capture 80% of the total variation in your dataset. It follows from this graph that as you include more PCs, the CPTV curve flattens out and remains close to its upper bound of 1.

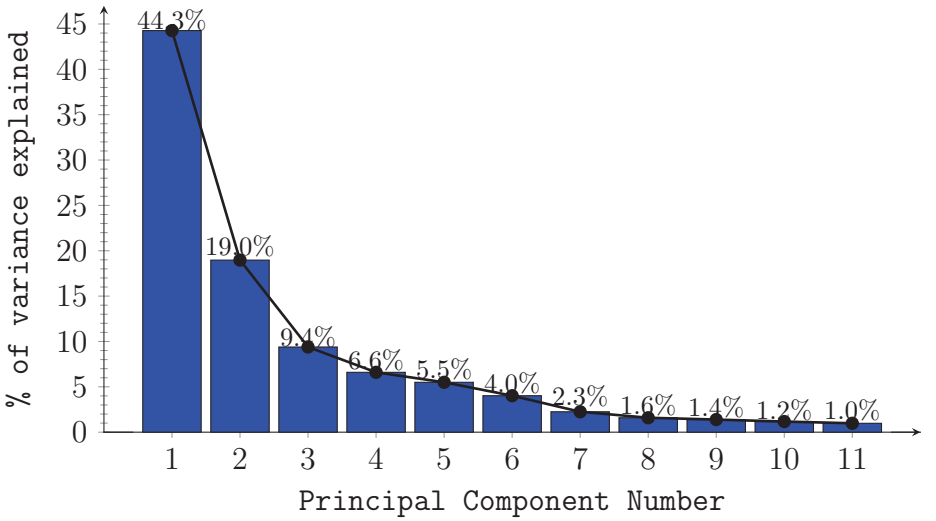

Figure 4.2: Scree plot showing of the percentage of the variance explained

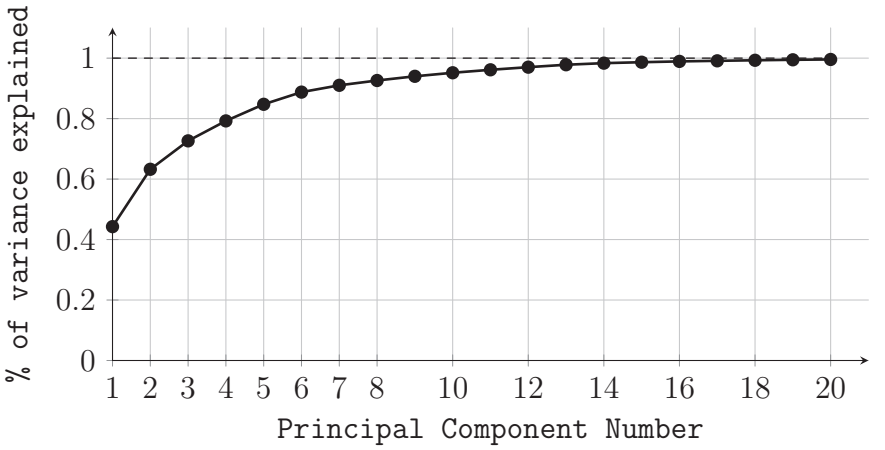

Figure 4.3: Cumulative variance plot for the BCW dataset