Post 02: Application and Visualization on multivariate data analysis in R

Yawen Sun

11/27/2017

If you try to reproduce this post on your laptop, please install the following packages if you have not done so:

install.packages("readr")

install.packages("dplyr")

install.packages("ggplot2")

install.packages("corrplot")

```
# packages
library(readr)     # importing data
library(dplyr)     # data wrangling
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)  # graphics
library(corrplot) # correlogram
```

```
## corrplot 0.84 loaded
```

```
# please set your own path to the file if you try to reproduce this on your own
setwd("/Users/sunsebrina/Documents/fall 2017courses/stat133/stat133-hws-fall17/post02")
```

Introduction:

In the lecture, the professor gave us an introduction on principal components analysis(PCA). We did exercises on PCA in hw03 and I wrote about introduction to PCA in my post 01. In this post 02, I will first explore general ways to explore connections between rows and columns of a data set. Thenk, I will expand more on the application and visualization of principal components analysis(PCA).

Motivation:

PCA is a hard and interesting topic for me. I am confused about but also interested in how to use this method to help us analyze data. Thus, I will explore more on this topic in this post.

Data exploration:

If you have not read my post 01, I will give another similar introduction with more example codes on data analyzation here:

Given a data set, a table, we can analyze it in two perspectives: one is objects – rows of the table, the other is variables – the columns of the table. We can study relationship among column variables and similarity between individual objects to explore a specific data set.

```
# read input data, teams
teams <- data.frame(read_csv(file = "data/nba2017-teams.csv"))
```

```
## Parsed with column specification:
## cols(
##   team = col_character(),
##   experience = col_integer(),
##   salary = col_double(),
##   points3 = col_integer(),
##   points2 = col_integer(),
##   free_throws = col_integer(),
##   points = col_integer(),
##   off_rebounds = col_integer(),
##   def_rebounds = col_integer(),
##   assists = col_integer(),
##   steals = col_integer(),
##   blocks = col_integer(),
##   turnovers = col_integer(),
##   fouls = col_integer(),
##   efficiency = col_double()
## )
```

```
str(teams)
```

```
## 'data.frame':    30 obs. of  15 variables:
##  $ team        : chr  "ATL" "BOS" "BRK" "CHI" ...
##  $ experience  : int  93 63 52 58 66 128 62 74 55 101 ...
##  $ salary      : num  90.9 91.9 65.5 92.1 100.2 ...
##  $ points3     : int  626 985 738 565 808 1012 754 868 631 982 ...
##  $ points2     : int  2254 2183 1950 2162 2102 2107 1803 2347 2638 2540 ...
##  $ free_throws : int  1373 1536 1381 1330 1499 1355 1042 1471 1140 1447 ...
##  $ points      : int  7759 8857 7495 7349 8127 8605 6910 8769 8309 9473 ...
##  $ off_rebounds: int  807 744 608 865 634 727 534 867 908 759 ...
##  $ def_rebounds: int  2537 2698 2546 2416 2636 2639 2109 2646 2838 2854 ...
##  $ assists     : int  1784 2069 1593 1746 1805 1760 1291 2009 1731 2475 ...
##  $ steals      : int  601 617 547 605 550 475 538 538 574 779 ...
##  $ blocks      : int  354 341 372 339 320 299 266 285 310 551 ...
##  $ turnovers   : int  1136 1037 1152 952 827 1005 703 1097 932 1171 ...
##  $ fouls       : int  1329 1686 1522 1275 1198 1318 1280 1458 1467 1565 ...
##  $ efficiency  : num  140 148 148 139 145 ...
```

To find relationship among column variables, we can calculate correlations between each two variables. We can do that by computing a correlation matrix.

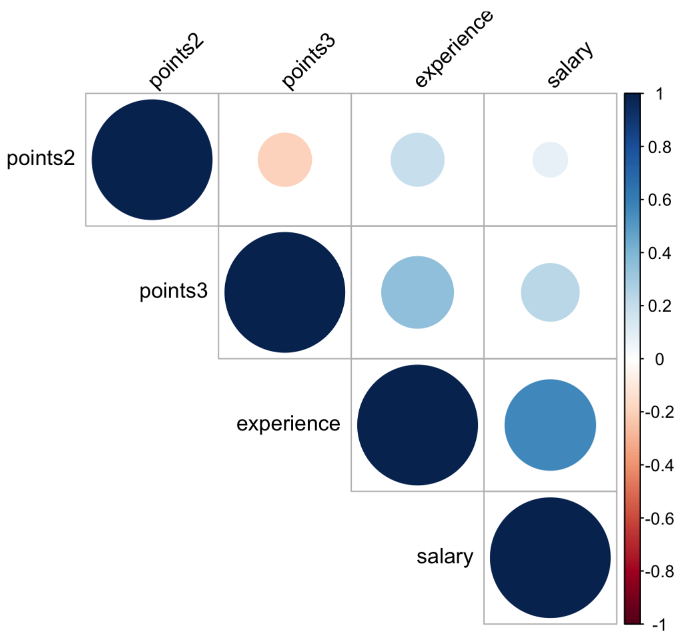
```
# dat includes columns of experience, salary, points3, and points2 in teams
dat <- teams[,c(2,3,4,5)]
# compute correlation matrix of dat
matrix <- cor(dat)
round(matrix, 2)
```

```
##            experience salary points3 points2
## experience       1.00   0.57    0.36    0.19
## salary           0.57   1.00    0.23    0.08
## points3          0.36   0.23    1.00   -0.20
## points2          0.19   0.08   -0.20    1.00
```

We could use the function corrplot(), which takes the correlation matrix as the first argument. The second argument (type="upper") is used to display only the upper triangular of the correlation matrix.

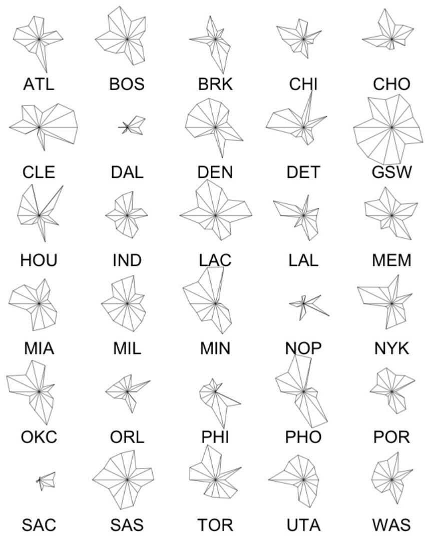
Circles in blue color represents positive correlationsand negative correlations are displayed in red color. Color intensity and the size of the circle are proportional to the correlation coefficients. In the right side of the correlogram, the legend color shows the correlation coefficients and the corresponding colors.

```
#install.packages("corrplot")

# Use corrplot() to create a correlogram:
corrplot(matrix, type = "upper", order = "hclust",
         tl.col = "black", tl.srt = 45)
```



To find out resemblance among individuals(rows), we can draw star plots of objects. The star plot is a classic method of displaying multivariate data. Each star represents a single observation. See an example of a star plot of NBA teams:

```r
# star plot of the teams
stars(teams[ ,-1], labels = teams$team)
```



To summarize the systematic variation of the variables, we see methods including calculating correlations, standardizing the variables, or using correlations of transformed and standardized variables.

What is PCA?

A better way to study the relationship of variables is to use principal componets analysis(PCA), a multivariate method and to look at principal components (PCs): linear combinations of the original variables.

We often see very high-dimensional data, so we use PCA as an unsupervised dimensionaly reduction technique to reduce dimensionality of data. A lower-dimensional representation of data is useful because we can usually only visualize data with 2 or 3 dimensions, and lower dimensions can help reduce computational load.

For example, in the case of nba2017-teams.csv data set, we have 15 variables describing different aspects of information and performances of each team. We can carry out a principal component analysis to discover whether we can have most of the variation between teams using fewer new variables, which are “principal components” – linear combinations of all or some of the 15 original variables.

Given a matrix of data points, PCA finds one or more orthogonal directions that capture the largest amount of variance in the data, so PCA can reduce the dimensionality of a data set while keeping as much as possible of the variation present in the data. Intuitively, the directions with less variance contain less information and may be discarded without introducing too much error.

Here is a general overview of steps of calculation, we will do these steps one by one later in the examples part. First, we subtract the mean to make the data points zero-mean. Then, we transform the original variables into a smaller set of new variables that summarize the variation in data – the principal componets, which can be seen as linear combinations of the original variables. After diagonaliztion, we do eigenvalue decomposition(EVD) of data.

Application of PCA on NBA teams data:

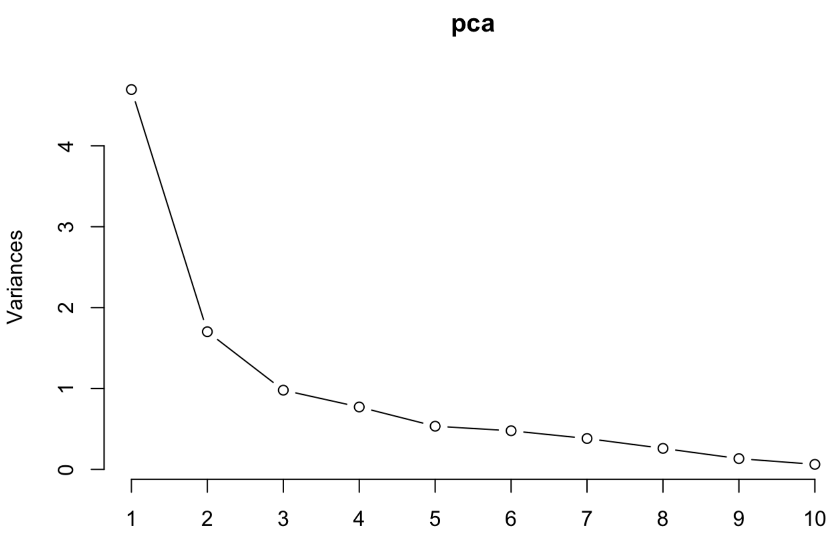
In R, there are many packages and functions that can be used to apply PCA (see more in the dicussion part later). In this post, I will stick to the function prcomp from the stats package. ###Perform a principal components analysis (PCA) Notice that I did not do standardise data here. Many people would rescale (taking log transfromation, for example) the data prior to the application of PCA.

```
# select columns we are interested in to do pca
part <- select(teams, points3, points2, free_throws, free_throws, off_rebounds, def_rebounds, assists, steals, blo
cks, turnovers, fouls)

# do a pca
pca <- prcomp(part, scale. = TRUE)

# see importance of components
summary(pca)
```

```
## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5    PC6
## Standard deviation     2.1670 1.3046 0.98971 0.87848 0.73081 0.6914
## Proportion of Variance 0.4696 0.1702 0.09795 0.07717 0.05341 0.0478
## Cumulative Proportion  0.4696 0.6398 0.73774 0.81491 0.86832 0.9161
##                            PC7     PC8     PC9    PC10
## Standard deviation     0.61823 0.51016 0.36550 0.25039
## Proportion of Variance 0.03822 0.02603 0.01336 0.00627
## Cumulative Proportion  0.95434 0.98037 0.99373 1.00000
```

We use the plot method here to return a picture of the variances associated with the PCs. In this case, we retain 10 PCs and we can see that the first two PCs show most of the variability in our nba teams data. We can also see that the most obvious change in slope in the scree plot occurs at component 3, and thus we should definitely keep the first two components.

```
# plot method
plot(pca, type = "l")
```



We can use the predict() function if we have some new data and want to predict values of their PCs. Just for illustration, we can pretend the last 3 rows of our data has just arrived and we would like to see what is their PCs values:

```
# Predict PCs
predict(pca,
        newdata=tail(part, 3))
```

```
##           PC1        PC2        PC3         PC4         PC5        PC6
## 28  0.6469827  1.3120040  0.4496369 -0.03210040 -0.04584683 -1.5195812
## 29  0.7307586 -0.1550934  0.5154001 -0.27881211  1.22136639  0.2553563
## 30 -0.1093009  0.2091070 -0.4485501  0.02343454 -1.12340751  0.0488935
##           PC7        PC8        PC9        PC10
## 28 -0.5137515  0.5910190  0.2346611  0.20892464
## 29 -0.4137574 -0.2504524  0.3884404 -0.02919999
## 30  0.3468959 -0.5059521 -0.3708573  0.23806925
```

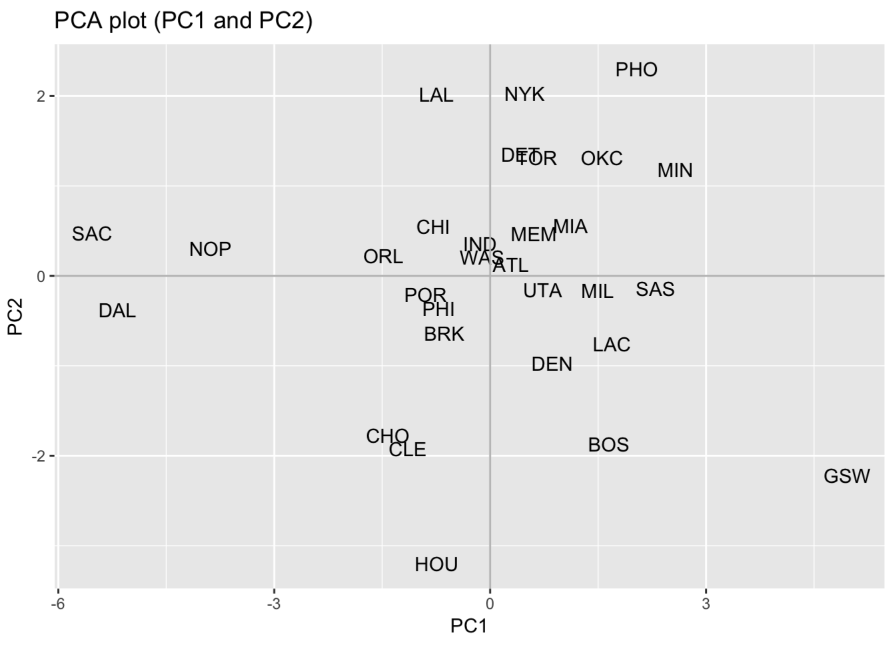
```
# keep first two PCs
pc <- data.frame(
  PC1 = pca$x[,1],
  PC2 = pca$x[,2],
  name = teams$team
)
pc
```

```
##           PC1        PC2 name
## 1   0.2883171  0.1281265  ATL
## 2   1.6475677 -1.8678932  BOS
## 3  -0.6378694 -0.6410895  BRK
## 4  -0.7889514  0.5491124  CHI
## 5  -1.4213891 -1.7716179  CHO
## 6  -1.1429197 -1.9254795  CLE
## 7  -5.1770470 -0.3771922  DAL
## 8   0.8628216 -0.9755539  DEN
## 9   0.4228059  1.3520635  DET
## 10  4.9580722 -2.2173199  GSW
## 11 -0.7434842 -3.2031420  HOU
## 12 -0.1393098  0.3561238  IND
## 13  1.6926408 -0.7550453  LAC
## 14 -0.7449230  2.0200116  LAL
## 15  0.6071090  0.4667924  MEM
## 16  1.1154708  0.5570744  MIA
## 17  1.4939629 -0.1637954  MIL
## 18  2.5754284  1.1769429  MIN
## 19 -3.8867632  0.3023898  NOP
## 20  0.4804728  2.0259452  NYK
## 21  1.5554071  1.3170619  OKC
## 22 -1.4831168  0.2204544  ORL
## 23 -0.7149664 -0.3641317  PHI
## 24  2.0387934  2.2997473  PHO
## 25 -0.8965058 -0.2071566  POR
## 26 -5.5291364  0.4742780  SAC
## 27  2.2990719 -0.1427248  SAS
## 28  0.6469827  1.3120040  TOR
## 29  0.7307586 -0.1550934  UTA
## 30 -0.1093009  0.2091070  WAS
```

We are using the first two PCs to get a scatterplot of the teams here.

The scatterplot shows the first principal component on the x-axis, and the second principal component on the y-axis. This plot is taken from my post01. Feel free to read more on that to explore PCA.

```
# PCA scatterplot
ggplot(data = pc, aes(x = PC1, y = PC2)) +
  geom_text(aes(label = name)) +
  ggtitle("PCA plot (PC1 and PC2)") +
  geom_hline(aes(yintercept = 0), color ="gray") +
  geom_vline(aes(xintercept = 0), color ="gray")
```



Another multivariate analysis: linear discriminant analysis (LDA)

The purpose of principal component analysis is to find the best low-dimensional representation of the variation in a multivariate data set. By carrying out a principal component analysis on our teams data, we found that most of the variation in the variables between different teams can be captured using the first two principal componets.

Another way to analyze multivariate data is linear discriminant analysis (LDA), which gives the best possible separation between variables of the data. We will explore this method in our post as well.

We can use linear discriminant analysis (LDA) by “lda()” function from the R “MASS” package.

Discussion:

There are many pca functions and packages in R:

prcomp(), princomp(), PCA(), etc.

There are also other visualization function for PCA, including ggbiplot().

Conclusion:

Take home message:

There are many ways to analyze data in R. We can use PCA, a dimentinality reduction method (eg:reduce a data set with dimention 4 to dimention 2), or LDA, a way to characterizes or separates many classes of objects, to help us better explore the data.

References:

1. “data/nba2017-teams.csv”, which is from hw03
2. Intro to PCA, lecture slide: https://github.com/ucb-stat133/stat133-fall-2017/blob/master/slides/15-principal-components1.pdf
3. Principal Component Analysis note from Berkeley cs189 course website: http://www.eecs189.org/static/notes/n9.pdf
4. Venables, W. N., Brian D. R. Modern applied statistics with S-PLUS. Springer-verlag. (Section 11.1)
5. Using R for Multivariate Analysis: http://little-book-of-r-for-multivariate-analysis.readthedocs.io/en/latest/src/multivariateanalysis.html
6. Principal Component Analysis – PCA in R: https://www.youtube.com/watch?v=mzx6CtdQJ4o
7. Brief introduction: http://www.itl.nist.gov/div898/handbook/pmc/section5/pmc55.htm
8. A guide to correlation matrix in R: http://www.sthda.com/english/wiki/correlation-matrix-a-quick-start-guide-to-analyze-format-and-visualize-a-correlation-matrix-using-r-software

9: Computing and visualizing PCA in R: https://www.r-bloggers.com/computing-and-visualizing-pca-in-r/