

10장 비지도학습

주성분분석

임요한

서울대학교

Aug, 2018

“비지도학습” 수업에서 다룰 내용

- 군집분석
 - 자율학습과 지도학습의 소개
 - K평균 군집분석
 - 계층군집화
- 주성분 분석
 - 주성분분석
 - 주성분회귀분석
- 인자분석

군집분석(clustering)

- 자율학습과 지도학습의 소개
- Kmeans 군집분석
- Kmeans++ 알고리즘
- 계층군집화

군집분석

- 자료를 동질적인 군집 혹은 부분으로 나누는 분석.
- 각 군집을 쉽게 설명하므로 전체를 설명하고자 한다.
- 군집분석의 예
 - 유방암 환자들의 자료를 이용해 알려지지 않은 암의 subtype을 찾으려고 한다.
 - 사람들의 자료를 이용하여 시장을 분할하고, 특정 광고에 잘 반응할 subgroup을 찾으려 한다.

자율학습(unsupervised Learning)

- 반응변수 Y 는 없고 변수들 X_1, \dots, X_p 만 있는 경우이다.
- 예측에는 관심이 없고, 변수들 사이의 특별한 관계가 있는가? 관측치들에 그룹이 있는가 등의 질문에 관심이 있다.
- 보통 탐색적 자료분석의 일부분이다.
- 자료분석의 결론이 성능이 좋은지 안좋은지 판단하기가 어렵고, 주관적인 측면이 강하다.

지도학습(supervised learning)

- 자료가 반응변수 Y 와 설명변수 X_1, \dots, X_p 로 구성되어 있다.
- 설명변수 X_1, \dots, X_p 가 주어져 있을 때, 반응변수 Y 의 예측에 목적이 있다.

K 평균(K means) 군집분석

군집 C_1, \dots, C_K 를 군집들이라 한다. 이는 다음의 조건을 만족한다.

$$\bigcup_{k=1}^K C_k = \{1, 2, \dots, n\} \quad C_i \cap C_j = \emptyset, \quad i \neq j.$$

목표

$$\sum_{k=1}^K W(C_k)$$

를 최소화하는 C_1, \dots, C_K 를 찾고자 한다.

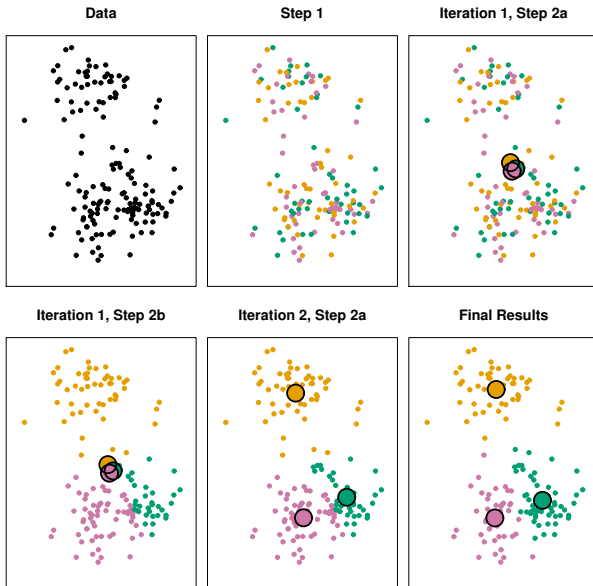
여기서 $W(C_k)$ 는 군집 C_k 의 변동을 측정하는 척도이다. 즉, C_k 의 동질성이 커지면 작아지는 값으로 대표적인 예로는

$$W(C_k) = \frac{1}{|C_k|} \sum_{i,j \in C_k} \|x_i - x_j\|^2$$

를 생각한다.

알고리즘

- 관측치들을 C_1, \dots, C_K 에 랜덤하게 할당한다.
- 군집이 바뀌지 않을 때까지 다음을 반복한다.
 - 각 군집마다 군집의 중앙(centroid)을 계산한다.
 - 모든 관측치를 가장 가까운 중심의 군집에 할당한다.



알고리즘의 수렴

위의 알고리즘은 반복이 진행될 때마다 $\sum_{k=1}^K W(C_k)$ 를 감소시킨다. 따라서, 이 알고리즘은 지역최적값(local optimum)으로 수렴한다. 지역 최적값으로 수렴하기 때문에 초기 조건을 달리하여 여러 번 돌리고 그 중 최소의 변동을 갖는 군집을 최종 결론으로 한다.

https://en.wikipedia.org/wiki/K-means_clustering

K-means ++

- 데이터 집합으로부터 임의의 데이터를 하나 선택하여 첫 번째 중심으로 설정한다.
- K개의 중심이 선택될 때 까지 다음의 단계를 반복한다.
 - 데이터 집합의 각 데이터에 대해서, 해당 데이터와 선택된 중심점들 중 가장 가까운 중심과의 거리 $D(x)$ 를 계산한다.
 - 확률이 $D(x)^2$ 에 비례하는 편중 확률 분포를 사용하여 임의의 데이터를 선택한 후, k-번째 중심으로 설정한다.
- 선택된 K개의 중심들을 초기 값으로 하여 K-평균 클러스터링을 수행한다.

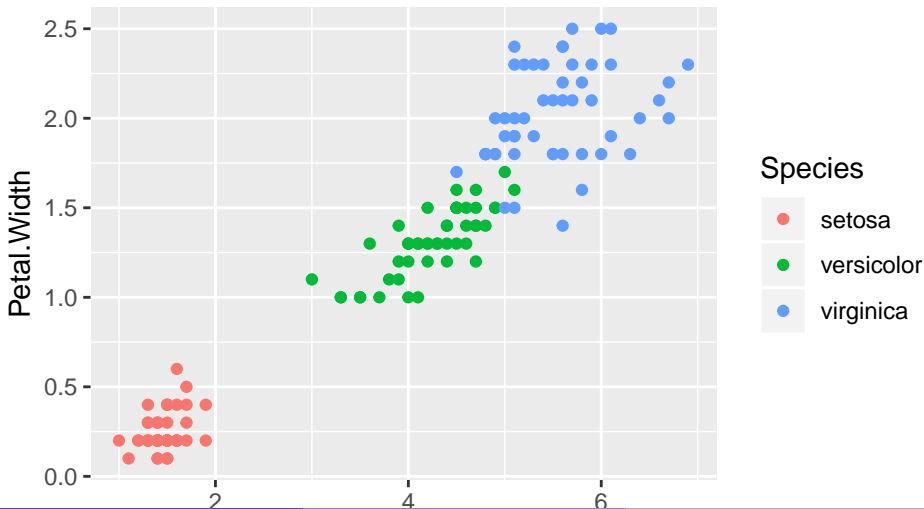
K-means: R code-1

```
library(datasets)
head(iris)
```

##	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
## 1	5.1	3.5	1.4	0.2	setosa
## 2	4.9	3.0	1.4	0.2	setosa
## 3	4.7	3.2	1.3	0.2	setosa
## 4	4.6	3.1	1.5	0.2	setosa
## 5	5.0	3.6	1.4	0.2	setosa
## 6	5.4	3.9	1.7	0.4	setosa

K-means: R code-2

```
library(ggplot2)
ggplot(iris, aes(Petal.Length, Petal.Width, color = Species)) + geom_point()
```



K-means: R code-3

```
set.seed(20)
irisCluster <- kmeans(iris[, 3:4], 3, nstart = 20)
irisCluster
```

```
## K-means clustering with 3 clusters of sizes 50, 52, 48
```

##

```
## Cluster means:
```

```
##      Petal.Length Petal.Width
```

```
## 1      1.462000      0.246000
```

```
## 2      4.269231      1.342308
```

```
## 3      5.595833      2.037500
```

##

```
## Clustering vector:
```

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

```
## [36] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 2 2
```

```
## [71] 2 2 2 2 2 2 2 3 2 2 2 2 2 3 2 2 2 2 2 2 2 2 2 2 2 2
```

```
## [106] 3 2 3 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 3 2 3 3 3 3 3 3 3
```

```
## [141] 3 3 3 3 3 3 3 3 3 3
```

##

Within cluster sum of squares by cluster: [1] 2.02200 13.05769 16.29167
(between_SS / total_SS = 94.3 %)

Available components:

- [1] "cluster" "centers" "totss" "withinss"
- [5] "tot.withinss" "betweenss" "size" "iter"
- [9] "ifault"

계층적 군집화

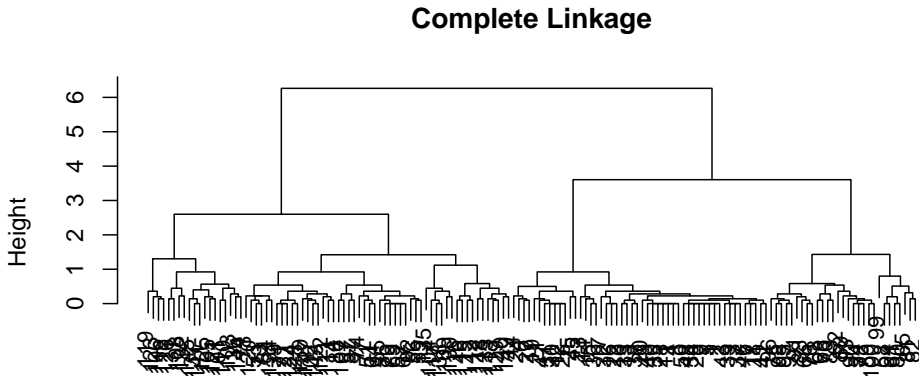
방법론 설명은 노트를 따라간다.

계층적군집화 Rcode-1

```
x=iris[,3:4]  
hc.complete=hclust(dist(x), method="complete")  
hc.average=hclust(dist(x), method="average")  
hc.single=hclust(dist(x), method="single")
```

계층적군집화 Rcode-2

```
plot(hc.complete,main="Complete Linkage", xlab="", sub="", cex=.9)
```



계층적군집화 Rcode-3

```
cutree(hc.complete, 2)
```

[illegible]

```
cutree(hc.complete, 3)
```

[illegible]

계층적군집화 Rcode-4

```
out.c=cutree(hc.complete, 3)
table(iris$Species,out.c)
```

```
##           out.c
##           1  2  3
## setosa      50  0  0
## versicolor  0 21 29
## virginica   0 50  0
```

주성분분석

방법론 설명은 노트를 따라간다.

알츠하이머자료

- diagnosis: “Impaired”, “Control”
- predictors: 130 variables
- 333 obs : 250 train samples, 83 test samples

```
load("AlzheimerDisease.RData")
head(predictors,n=1)
```

```
## ACE_CD143_Angiotensin_Converti ACTH_Adrenocorticotrophic_Hormon
## 1 2.0031 -1.386294
## Adiponectin_Alpha_1_Antichymotrypsin_Alpha_1_Antitrypsin
## 1 -5.360193 1.740466 -12.63136
## Alpha_1_Microglobulin_Alpha_2_Macroglobulin_Angiopoietin_2_ANG
## 1 -2.577022 -72.65029 1.06471
## Angiotensinogen_Apolipoprotein_A_IV_Apolipoprotein_A1_Apolipop
## 1 2.510547 -1.427116 -7.402052 -0
## Apolipoprotein_B_Apolipoprotein_CI_Apolipoprotein_CIII_Apolipop
## 1 -4.624044 -1.272966 -2.312635
## Apolipoprotein_E_Apolipoprotein_H_B_Lymphocyte_Chemoattractant
## 1 3.754521 -0.1573491 2.2969
## BMP_6_Beta_2_Microglobulin_Betacellulin_C_Reactive_Protein
## 1 -2.200744 0.6931472 34 -4.074542
## CD40 CD5L_Calbindin_Calcitonin CgA_Clusterin_A
## 1 -0.7964147 0.09531018 33.21363 1.386294 397.6536 3.55
## Complement_3_Complement_Factor_H_Connective_Tissue_Growth_Facto
## 1 -10.36305 3.573725 0.530628
```

주성분분석 Rcode-1

```
train.x=predictors[1:250,1:129]
test.x=predictors[251:333,1:129]
train.y=as.numeric(diagnosis[1:250])
test.y=as.numeric(diagnosis[251:333])
```


주성분분석 Rcode-2

```
pr.out=prcomp(train.x,scale=TRUE)
str(pr.out)
```

```
## List of 5
## $ sdev      : num [1:129] 5.98 3.83 2.33 2 1.98 ...
## $ rotation: num [1:129, 1:129] -0.1059 -0.0281 -0.1154 -0.0749 ...
## ..- attr(*, "dimnames")=List of 2
## .. ..$ : chr [1:129] "ACE_CD143_Angiotensin_Converti" "ACTH_Adre
## .. ..$ : chr [1:129] "PC1" "PC2" "PC3" "PC4" ...
## $ center    : Named num [1:129] 1.31 -1.556 0.285 -5.249 1.348 ...
## ..- attr(*, "names")= chr [1:129] "ACE_CD143_Angiotensin_Conver
## $ scale     : Named num [1:129] 0.552 0.267 0.452 0.67 0.365 ...
## ..- attr(*, "names")= chr [1:129] "ACE_CD143_Angiotensin_Conver
## $ x         : num [1:250, 1:129] -11.486 -3.087 0.565 -0.589 -11.5
## ..- attr(*, "dimnames")=List of 2
## .. ..$ : chr [1:250] "1" "2" "3" "4" ...
## .. ..$ : chr [1:129] "PC1" "PC2" "PC3" "PC4" ...
## - attr(*, "class")= chr "prcomp"
```

주성분분석 Rcode-3

```
pr.out$rotation
```

##		PC1	PC2	
##	ACE_CD143_Angiotensin_Converti	-0.105936157	-0.1292568290	-0.05
##	ACTH_Adrenocorticotropic_Hormon	-0.028129754	-0.0663231776	0.01
##	AXL	-0.115412879	-0.1318680364	-0.05
##	Adiponectin	-0.074905479	0.1259499772	0.04
##	Alpha_1_Antichymotrypsin	-0.129034627	0.1020186687	0.01
##	Alpha_1_Antitrypsin	-0.083273696	0.1196552718	0.06
##	Alpha_1_Microglobulin	-0.123397498	0.1258695666	0.02
##	Alpha_2_Macroglobulin	-0.115058728	-0.0061491878	0.10
##	Angiopietin_2_ANG_2	-0.115878753	-0.1065056764	0.02
##	Angiotensinogen	-0.021077009	-0.0859870509	0.00
##	Apolipoprotein_A_IV	-0.092736118	0.1290239700	-0.05
##	Apolipoprotein_A1	-0.101134352	0.1423530979	-0.09
##	Apolipoprotein_A2	-0.099531360	0.1722375502	-0.06
##	Apolipoprotein_B	-0.077015389	0.1272307761	0.07
##	Apolipoprotein_CI	-0.089880486	0.1514731907	-0.08

주성분분석 Rcode-4

```
dim(pr.out$x)
```

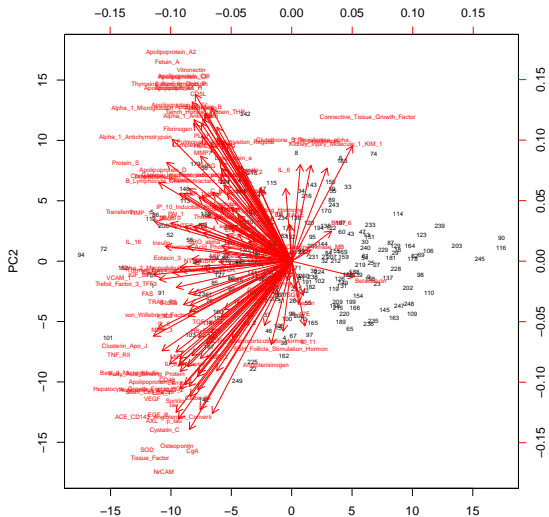
```
## [1] 250 129
```

```
str(pr.out$x)
```

```
## num [1:250, 1:129] -11.486 -3.087 0.565 -0.589 -11.561 ...  
## - attr(*, "dimnames")=List of 2  
## ..$ : chr [1:250] "1" "2" "3" "4" ...  
## ..$ : chr [1:129] "PC1" "PC2" "PC3" "PC4" ...
```

주성분분석 Rcode-5

```
biplot(pr.out,scale=0,cex=0.5)
```



주성분분석 Rcode-6

```
pr.out$sdev[1:10]
```

```
## [1] 5.977187 3.828614 2.334865 2.004566 1.979021 1.887169 1.8587
## [8] 1.630767 1.575265 1.453642
```

```
pr.var=pr.out$sdev^2
pr.var[1:10]
```

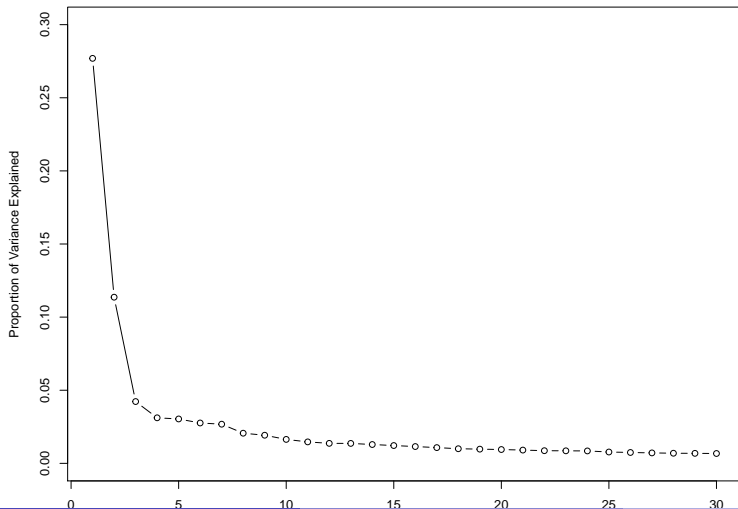
```
## [1] 35.726760 14.658288 5.451597 4.018287 3.916525 3.561407
## [8] 2.659402 2.481461 2.113075
```

```
pve=pr.var/sum(pr.var)
pve[1:10]
```

```
## [1] 0.27695163 0.11363014 0.04226044 0.03114951 0.03036066 0.027
## [7] 0.02678388 0.02061552 0.01923613 0.01638043
```

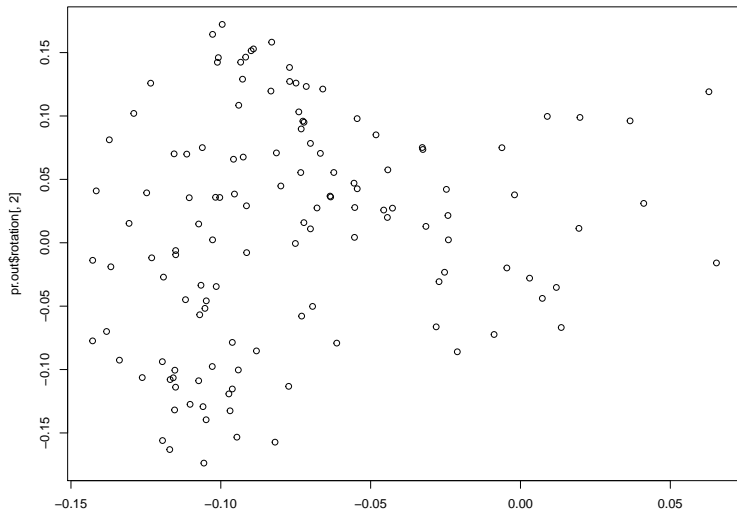
주성분분석 Rcode-7

```
plot(pve[1:30], xlab="Principal Component",  
     ylab="Proportion of Variance Explained", ylim=c(0,0.3),type='b')
```



주성분분석 Rcode-7

```
plot(pr.out$rotation[,1],pr.out$rotation[,2])
```



주성분회귀 Rcode-1

교재의 R code에서 다음의 부분이 중간에 추가되어야 한다.

```
library(ISLR)
Hitters=na.omit(Hitters)
x=model.matrix(Salary ~.,Hitters)[, -1]
y=Hitters$Salary
set.seed(1)
train=sample(1:nrow(x), nrow(x)/2)
test=(-train)
y.test=y[test]
```


주성분회귀 예제

- USArrests자료를 이용하고 가상 response에 대한 PCR을 생각하여 보자.
-

$$x_5 = x_2 + \text{rnorm}(50, 0, 1)$$

$$y = 1 + 1 * x_1 + 2 * x_2 + 3 * x_3 + 4 * x_4 + 5 * x_5$$

- 참값

$$[1, 1, 2, 3, 4, 5]$$

주성분회귀 Rcode-2

```
x0=USArrests
set.seed(1)
x5=USArrests[,2]+rnorm(50,0.01)
x=cbind(x0,x5)
x=scale(x)
y=1+1*x[1]+2*x[2]+3*x[3]+4*x[4]+5*x[5]+rnorm(50)
pr.out=prcomp(x,scale=T)
head(pr.out,n=1)
```

```
## $sdev
```

```
## [1] 1.835728367 1.007043291 0.618863150 0.482627383 0.006661972
```

주성분회귀 Rcode-3

```
pcscore=pr.out$x[,1:2] #2개의 principal component 사용
load=pr.out$rotation[,1:2]
plm=lm(unlist(y)~pcscore)
pbeta=plm$coef
betaest=rep(0,6)
betaest[1]=pbeta[1]
for(k in (2:6)){
  betaest[k]=pbeta[2]*load[k-1,1]+pbeta[3]*load[k-1,2]
}
for(k in (2:6)){
  betaest[k]=pbeta[2]*load[k-1,1]
}
betaest
```

```
## [1] 5.91125440 0.05836535 0.06580357 0.02569453 0.05522457 0.0657
```

주성분회귀 Rcode-4

```
library(pls)
```

```
##  
## Attaching package: 'pls'  
  
## The following object is masked from 'package:stats':  
##  
##      loadings
```

```
pcr.fit=pcr(unlist(y)~x,scale=T,ncomp=2)  
pcr.fit$coef
```

```
## , , 1 comps  
##  
##      unlist(y)  
## Murder    0.05836535  
## Assault   0.06580357  
## UrbanPop  0.02569453  
## Rape      0.05522457
```

, , 1 comps

Murder 0.05836535 \ Assault 0.06580357 \ UrbanPop 0.02569453 \ Rape
0.05522457 \ x5 0.06576019 \

, , 2 comps

Murder -6.346392e-05 \ Assault 4.315352e-02 \ UrbanPop 1.801896e-01 \ Rape
9.888428e-02 \ x5 4.325277e-02 \