다변량자료분석 및 실습 Lab 7

서울대학교 통계학과 2017-11362 박건도

2021년 12월 01일

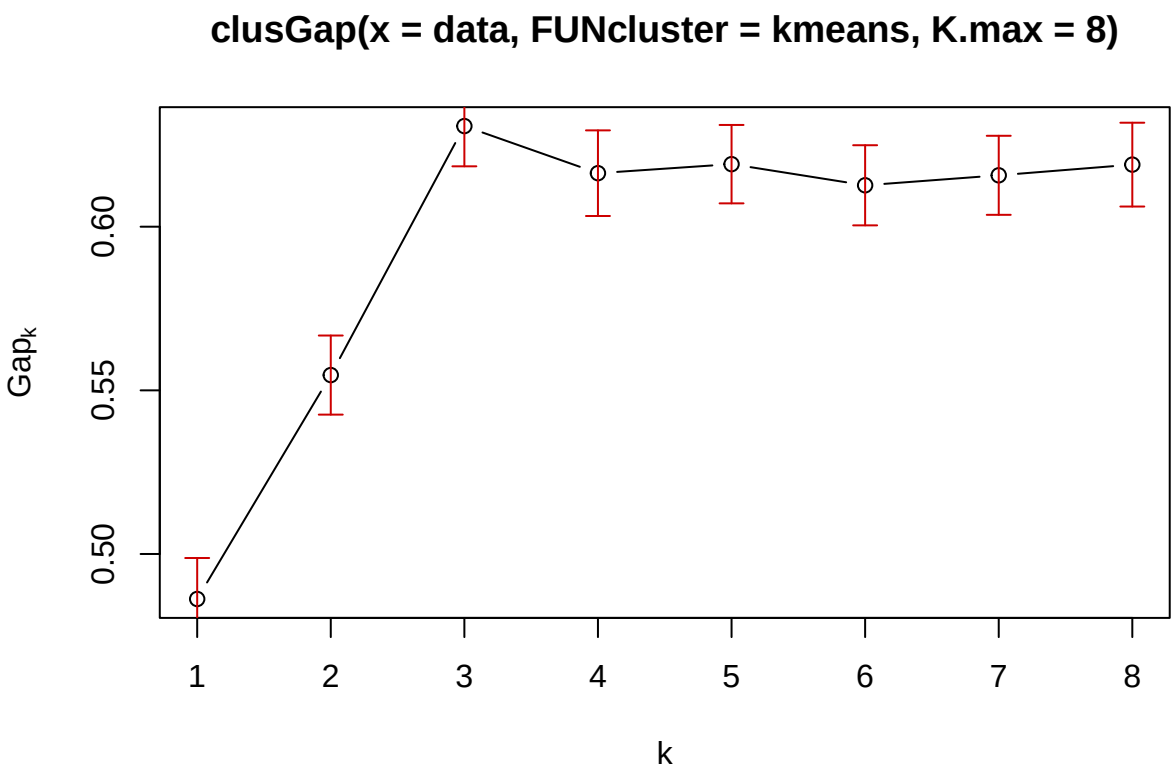
1. Load the dataset “rattle.data::wine”. Standardize the data.

```
data <- rattle.data::wine[,-1]
data <- scale(data)
```

2. Use the last 12 variables (just excluding “Type”).

a) determine the number of clusters.

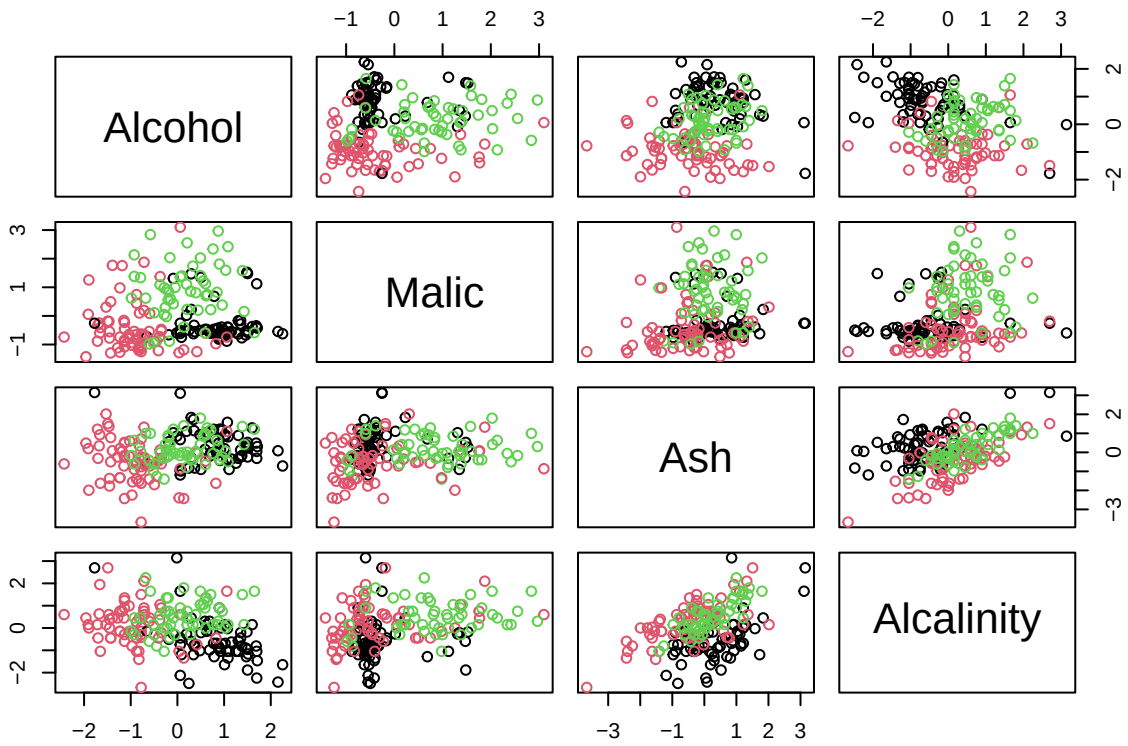
```
gap <- clusGap(data, FUN = kmeans, K.max = 8)
plot(gap)
```



From gap statistics, we can determine K = 3.

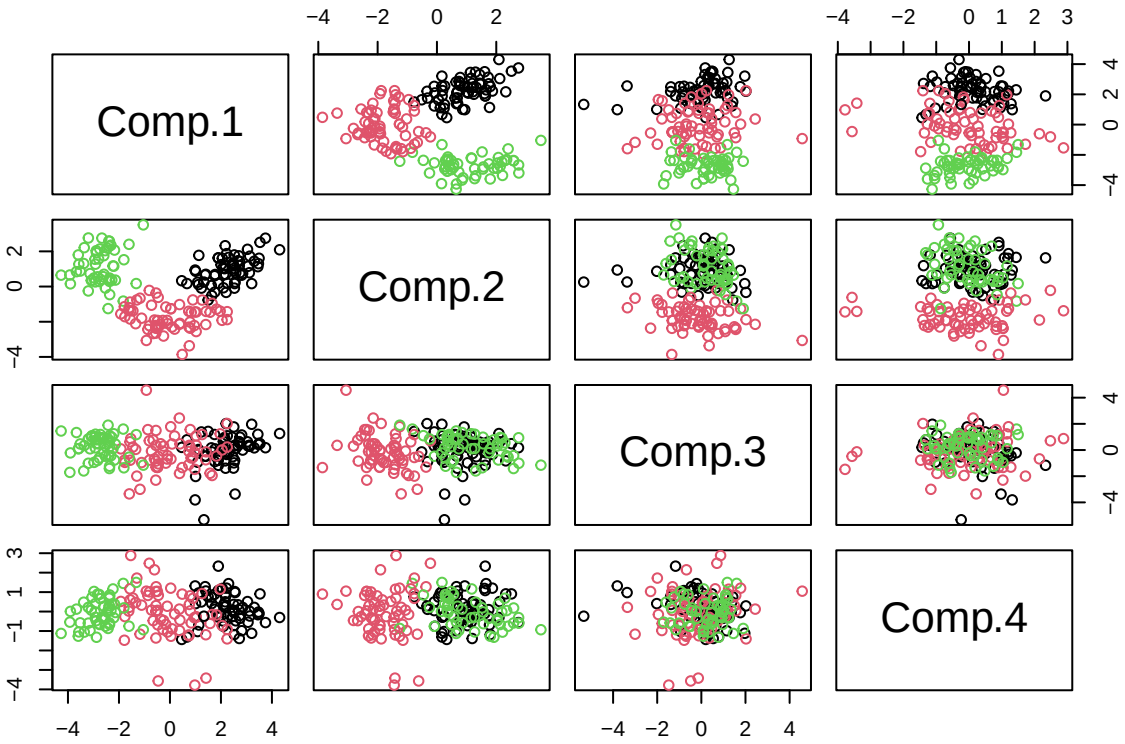
b) visualize your clustering results (on a 2-D coordinate)

```
k <- 3
kmeansobj <- kmeans(data,k)
pairs(data[,1:4], col = kmeansobj$cluster)
```



12개의 variable을 모두 plot 하면 너무 그림이 많아져서 4개만 그려보았다. PCA를 통해 상위 4개의 결과를 보면 아래와 같다.

```
pairs(princomp(data)$scores[,1:4], col = kmeansobj$cluster)
```



c) create a confusion table, comparing the clustering results with true labels.

```
predicted <- as.factor(kmeansobj$cluster)
confusionMatrix(predicted, rattle.data::wine[,1])
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  1  2  3
##          1 59  3  0
##          2  0 65  0
##          3  0  3 48
##
## Overall Statistics
##
##                Accuracy : 0.9663
##                  95% CI : (0.9281, 0.9875)
##     No Information Rate : 0.3989
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.9491
##
```

```
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3
## Sensitivity            1.0000   0.9155   1.0000
## Specificity            0.9748   1.0000   0.9769
## Pos Pred Value         0.9516   1.0000   0.9412
## Neg Pred Value         1.0000   0.9469   1.0000
## Prevalence             0.3315   0.3989   0.2697
## Detection Rate         0.3315   0.3652   0.2697
## Detection Prevalence   0.3483   0.3652   0.2865
## Balanced Accuracy      0.9874   0.9577   0.9885
```

꽤 잘 분류하는 것을 알 수 있다.

3. Repeat the E and M steps until the estimates do not change substantially. How many iterations do you need for the change in pi1 is less than 10e-5?

```
E.step <- function(theta, X) {# theta = list(pi1, mu1, sigma1, pi2, mu2, sigma2)
  pi1_X <- theta[[1]] * dmvnorm(X, mean = theta[[2]], sigma = theta[[3]])
  pi2_X <- theta[[4]] * dmvnorm(X, mean = theta[[5]], sigma = theta[[6]])
  pi1X <-  pi1_X / (pi1_X + pi2_X)
  pi1X
}

M.step <- function(pi1X, X){
  theta <- list()
  theta[[2]] <- apply(X, 2, weighted.mean, w = pi1X)
  cX1 <- apply(X, 1, function(x) x - theta[[2]])
  theta[[3]] <- cX1 %*% diag(pi1X) %*% t(cX1) / sum(pi1X)

  theta[[5]] <- apply(X, 2, weighted.mean, w = 1 - pi1X)
  cX2 <- apply(X, 1, function(x) x - theta[[5]])
  theta[[6]] <- cX2 %*% diag(1-pi1X) %*% t(cX2) / sum(1 - pi1X)

  theta[[1]] <- sum(pi1X) / n
```

```
  theta[[4]] <- 1 - theta[[1]]
  theta
}

# initial guess for theta
n <- nrow(data)
kmeansobj <- kmeans(data,2)

mu1 <- colMeans(data[kmeansobj$cluster == 1,])
mu2 <- colMeans(data[kmeansobj$cluster == 2,])
S1 <- cov(data[kmeansobj$cluster == 1,])
S2 <- cov(data[kmeansobj$cluster == 2,])
pi1 <- kmeansobj$size[1] / n
pi2 <- 1 - pi1
theta <- list(pi1, mu1, S1, pi2, mu2, S2)

# EM algorithm
cnt = 0
while (TRUE) {
  cnt <- cnt + 1
  pi1X <- E.step(theta, data)
  new_theta <- M.step(pi1X, data)
  if (abs(theta[[1]] - new_theta[[1]]) < 1e-5) {
    break
  }
  theta <- new_theta
}
cnt
```

```
## [1] 65
```

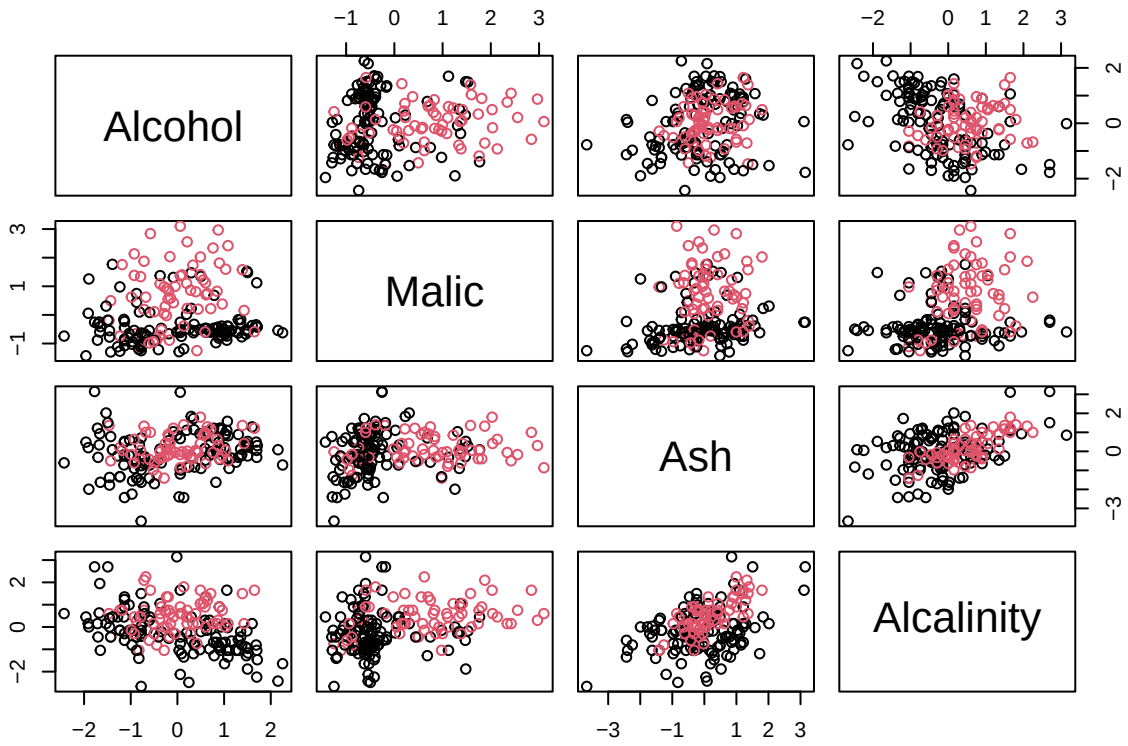
4. Report the clustering result by providing the class probabilities for each observation.

```
paste0('pi1: ', theta[[1]], ', pi2: ', theta[[4]])
```

```
## [1] "pi1: 0.646602198503158, pi2: 0.353397801496842"
```

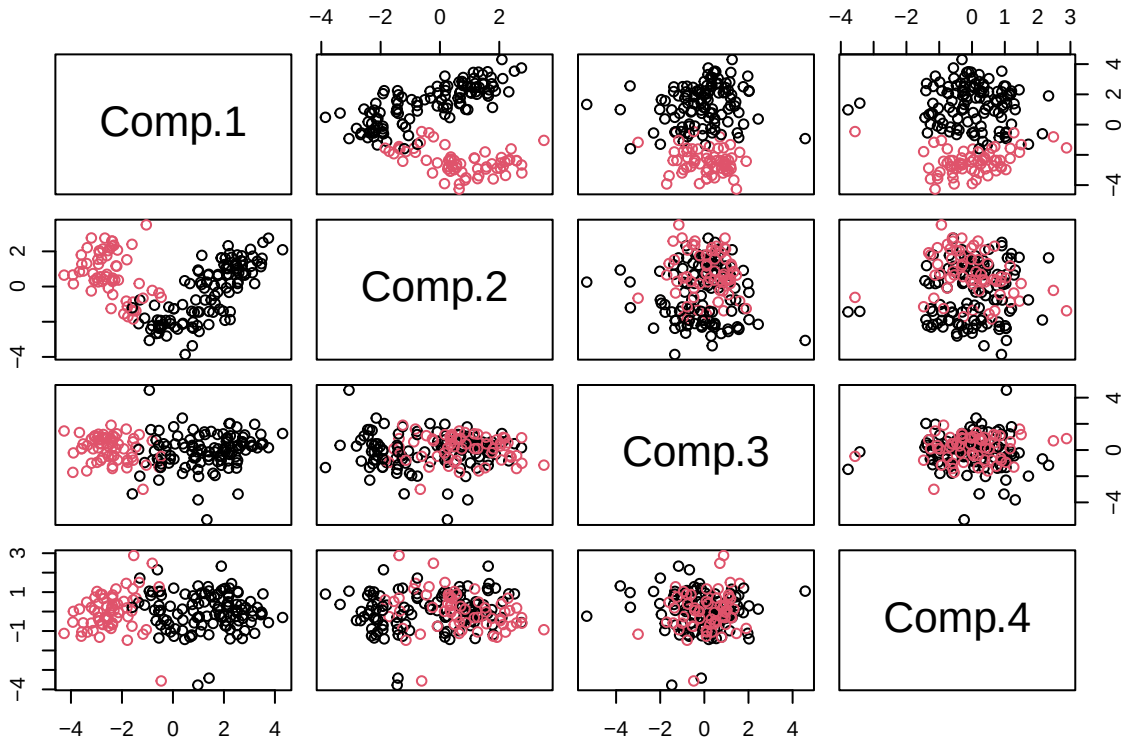
```
pi1X <- E.step(theta, data)
predicted <- ifelse(pi1X > 0.5, 1, 2)

pairs(data[,1:4], col = predicted)
```



PCA를 통한 그림은 다음과 같다.

```
pairs(princomp(data)$scores[,1:4], col = predicted)
```


Comp.1
Comp.2
Comp.3
Comp.4
−4 −2 0 2
−4 −2 0 1 2 3
−4 0 2 4
−4 0 2
−4 0 4
−4 −1 1 3
−4 −2 0 2 4
−4 −2 0 2 4