군집분석 2

2 k-평균군집

k-평균군집(k-means clustering)은 원하는 군집 수만큼(k개) 초기값을 지정하고, 각 개체(데이터)를 가까운 초기값에 할당하여 군집을 형성한 뒤, 각 군집의 평균을 재계산하여 초기값을 갱신한다. 갱신된 값에 대해 위의 할당과정을 반복하여 k개의 최종군집을 형성한다.

k-평균군집의 절차(알고리즘)는 다음과 같다.

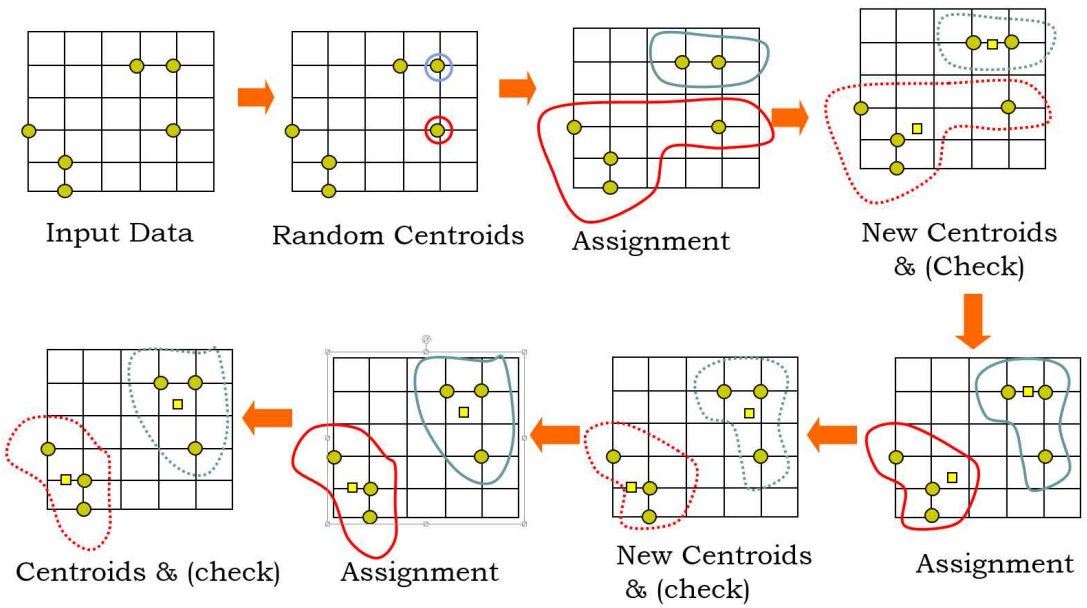
단계1. 초기 (군집의) 중심으로 k개의 객체를 임의로 선택한다.
단계2. 각 자료를 가장 가까운 군집 중심에 할당한다.
단계3. 각 군집 내의 자료들의 평균을 계산하여 군집의 중심을 갱신(update)한다.
단계4. 군집 중심의 변화가 거의 없을 때(또는 최대 반복수)까지 단계2와 단계3를 반복한다.

위의 단계2는 자료들의 군집의 중심점(평균)으로부터의 오차제곱합

$$E=\sum_{i=1}^{k}\sum_{x\in C_i}(x-\bar{x}_i)^2,\quad \bar{x}_i=\frac{1}{n}\sum_{x\in C_i}x$$

이 최소가 되도록 각 자료를 할당하는 과정이다.

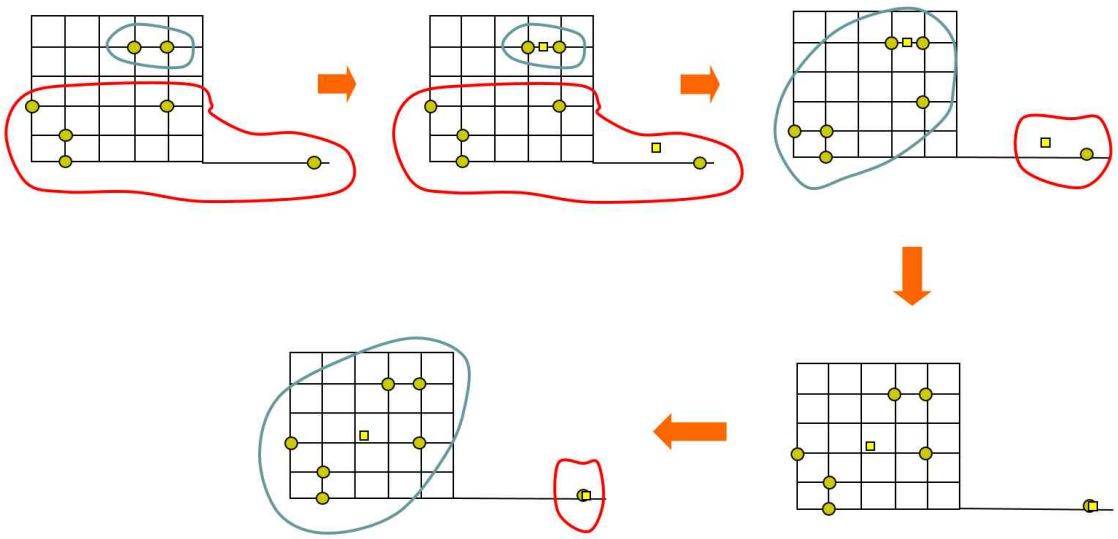
아래의 그림은 k-평균군집의 절차를 나타낸 것이다.



k-평균군집에서 군집의 수(k)는 미리 정해 주어야 하며, k-개의 초기 중심값은 임의로 선택될 수 있으나, 자료값 중에서 무작위로 선택하는 것이 보다 편리할 것이다. 다만 초기 중심점들은 서로 멀리 떨어져 있는 것이 바람직하며, 초기값에 따라 군집 결과가 크게 달라질 수 있다. 또한 k-평균군집은 군집의 매 단계마다 군집 중심으로부터의 오차제곱합을 최소화하는 방향으로 군집을 형성해나가는(부분 최적화를 수행하는) 탐욕적(greedy) 알고리즘으로 간주될 수 있으며, 안정된 군집은 보장하나 전체적으로 최적이라는 것은 보장하지 못한다.

k-평균군집은 알고리즘이 단순하며, 빠르게 수행되며 계층적 군집보다 많은 양의 자료를 다룰 수 있으며, 평균 등 거리 계산에 기반하므로 모든 변수가 연속적이어야 한다. 단점으로는 잡음이나 이상점에 영향을 많이 받으며(군집의 중심을 계산하는 과정에서), 볼록한 형태가 아닌(non-convex) 군집(예를 들어, U-형태의 군집)이 존재할 경우에는 성능이 떨어진다.

아래의 그림은 k-평균군집이 이상치 자료에 대해 민감하게 반응하는 과정을 보여준다.



이상치 자료에 민감한 k-평균군집의 단점을 보완하기 위해 군집을 형성하는 매 단계 마다 평균 대신 중앙값을 사용하는 k-중앙값(k-medoids)군집을 사용할 수 있다(R에서 k-중앙값 군집은 pam() 함수를 이용한다). k-평균군집을 수행하기 전 탐색적 자료분석을 통해 이상치를 미리 제거하는 것도 좋은 방법이다.

R에서 kmeans() 함수는 k-평균군집을 수행한다. 이 때, 임의로 선택되는 초기값에 따라 결과가 달라지는 것을 없애기 위해서는 set.seed()를 사용한다. 또한 nstart= 옵션은 다중(multiple)의 초기값에 대해 k-평균군집을 수행하고 그 가운데 최적의 결과를 제시해 준다(종종 nstart=25를 추천한다).

계층적 군집과는 달리 k-평균군집은 군집의 수를 미리 정해주어야 한다. 패키지 {Nbclust}를 통해 적절한 군집의 수에 대한 정보를 얻을 수 있다. 군집 수에 따른 집단 내

제곱합(within-groups sum of squares)의 그래프를 그려보는 것도 군집 수를 정하는데 도움이 된다. 이 그래프는 아래의 함수를 이용하여 그릴 수 있다.

```
> wssplot <- function(data, nc=15, seed=1234){
               wss <- (nrow(data)-1)*sum(apply(data,2,var))
               for (i in 2:nc){
                    set.seed(seed)
                    wss[i] <- sum(kmeans(data, centers=i)$withinss)}
                plot(1:nc, wss, type="b", xlab="Number of Clusters",
                     ylab="Within groups sum of squares")}
```

위 함수에서 data는 수치형의 자료이며, nc는 고려할 군집의 최대 수, seed는 난수 발생 초기값이다.

아래의 [예제 1]은 kmeans() 함수를 이용하여 k-평균군집을 수행한다.

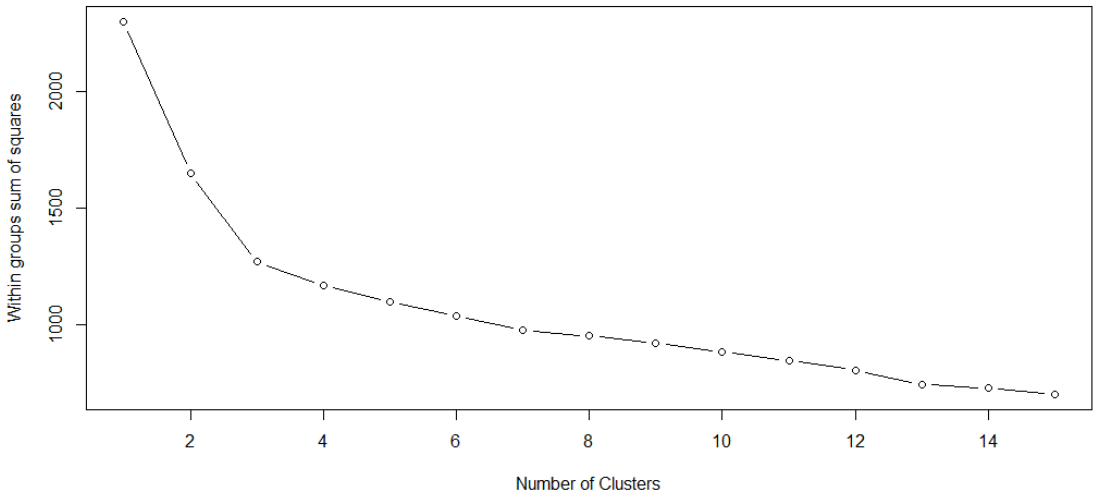
[예제 1] 분석에 사용되는 자료는 패키지 {rattle}에서 제공하는 178개 이탈리안 와인에 대해 13가지의 화학적 성분을 측정한 자료이다.

```
> data(wine, package="rattle")
> head(wine)
  Type Alcohol Malic  Ash Alcalinity Magnesium Phenols Flavanoids Nonflavanoids
1    1   14.23  1.71 2.43       15.6       127    2.80       3.06          0.28
2    1   13.20  1.78 2.14       11.2       100    2.65       2.76          0.26
3    1   13.16  2.36 2.67       18.6       101    2.80       3.24          0.30
4    1   14.37  1.95 2.50       16.8       113    3.85       3.49          0.24
5    1   13.24  2.59 2.87       21.0       118    2.80       2.69          0.39
6    1   14.20  1.76 2.45       15.2       112    3.27       3.39          0.34
  Proanthocyanins Color  Hue Dilution Proline
1            2.29  5.64 1.04     3.92    1065
2            1.28  4.38 1.05     3.40    1050
3            2.81  5.68 1.03     3.17    1185
4            2.18  7.80 0.86     3.45    1480
5            1.82  4.32 1.04     2.93     735
6            1.97  6.75 1.05     2.85    1450
```

변수의 측정 단위(또는 범위)가 매우 다르므로 군집분석을 수행하기 전에 scale() 함수를 이용하여 표준화를 수행하고, 적절한 군집 수를 정하기 위해 앞서 소개된 wssplot() 함수를 수행한다.

```
> df <- scale(wine[-1])
```

```
> wssplot(df)
```

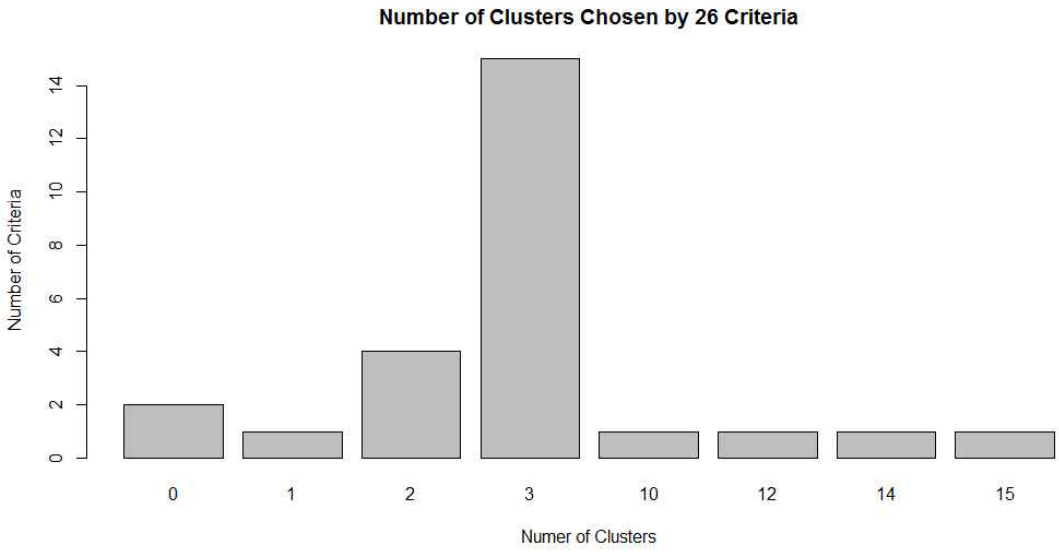


군집수 3에서 오차제곱합이 크게 감소되었음을 확인할 수 있다. 군집수의 결정은 아래와 같이 Nbclust{Nbclust}를 이용할 수도 있다.

```
> library(NbClust)
> set.seed(1234)
> nc <- NbClust(df, min.nc=2, max.nc=15, method="kmeans")
> table(nc$Best.n[1,])

 0  1  2  3 10 12 14 15
 2  1  4 15  1  1  1  1

> barplot(table(nc$Best.n[1,]),
          xlab="Numer of Clusters", ylab="Number of Criteria",
          main="Number of Clusters Chosen by 26 Criteria")
```



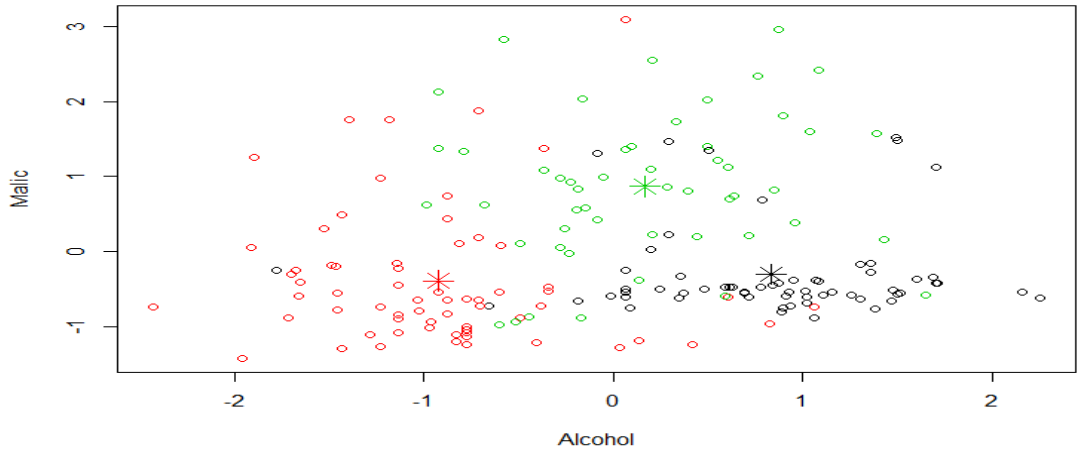
최적의 군집수를 정하기 위해 사용되는 지수(총 30개 중 여기서는 26개의 지수가 계산됨) 가운데 15개의 지수가 3을 최적의 군집수로 투표(majority voting)한 결과를 보여준다.

군집의 수(k)를 3으로 하여 kmeans()를 수행한 결과는 다음과 같다. 각 군집의 크기와 중심값을 보여준다. 군집 결과의 시각화는 plot() 함수를 이용한다.

```
> set.seed(1234)
> fit.km <- kmeans(df, 3, nstart=25)
> fit.km$size
[1] 62 65 51

> fit.km$centers
  Alcohol Malic   Ash Alcalinity Magnesium Phenols Flavanoids Nonflavanoids
1    0.83 -0.30  0.36      -0.61     0.576   0.883      0.975         -0.561
2   -0.92 -0.39 -0.49       0.17    -0.490  -0.076      0.021         -0.033
3    0.16  0.87  0.19       0.52    -0.075  -0.977     -1.212          0.724
  Proanthocyanins Color   Hue Dilution Proline
1           0.579  0.17  0.47     0.78    1.12
2           0.058 -0.90  0.46     0.27   -0.75
3          -0.778  0.94 -1.16    -1.29   -0.41

> plot(df, col=fit.km$cluster)
> points(fit.km$center, col=1:3, pch=8, cex=1.5)
```



각 군집별로 변수의 요약값을 측정단위의 척도로 나타내면 다음과 같다.

```
> aggregate(wine[-1], by=list(cluster=fit.km$cluster), mean)
```

cluster	Alcohol	Malic	Ash	Alcalinity	Magnesium	Phenols	Flavanoids
1	1	14	1.8	2.4	17	106	2.8	3.0
2	2	12	1.6	2.2	20	88	2.2	2.0
3	3	13	3.3	2.4	21	97	1.6	0.7

	Nonflavanoids	Proanthocyanins	Color	Hue	Dilution	Proline
1	0.29	1.9	5.4	1.07	3.2	1072
2	0.35	1.6	2.9	1.04	2.8	495
3	0.47	1.1	7.3	0.67	1.7	620

다음은 k-평균군집의 결과에 대한 정오분류표를 제시한다.

```
> ct.km <- table(wine$Type, fit.km$cluster)
> ct.km
    1  2  3
  1 59  0  0
  2  3 65  3
  3  0  0 48
```

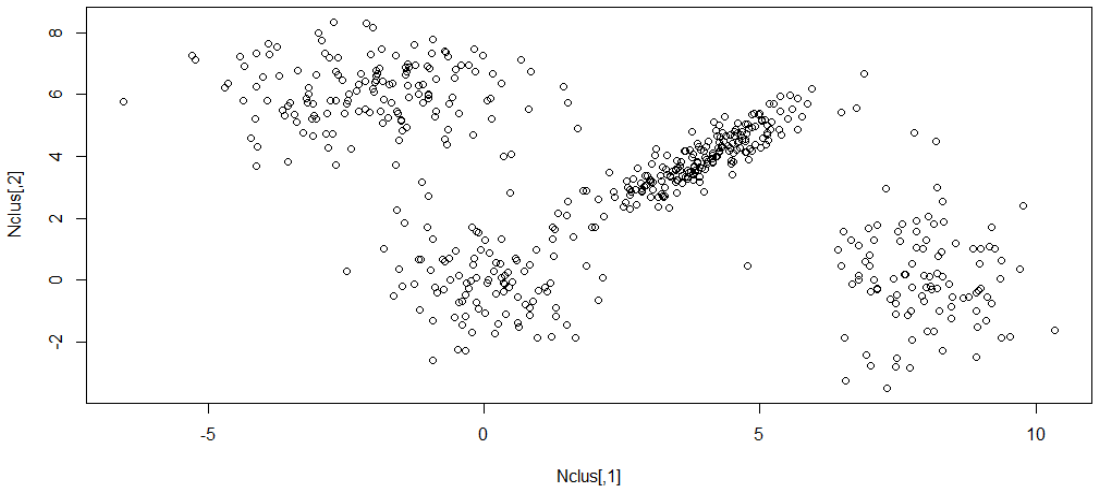
패키지 {flexclust}의 randIndex() 함수를 이용하면 실제 와인의 종류(Type)와 군집간의 일치도(agreement)를 나타내는 수정된 순위 지수(adjusted rank index)를 구할 수 있다. 여기서 수정된의 의미는 우연에 의해 발생되는 경우를 고려한 값이다. 이 지수는 -1(no agreement)과 1(perfect agreement) 사이의 값을 가진다.

```
> library(flexclust)
> randIndex(ct.km)
      ARI
0.897495
```

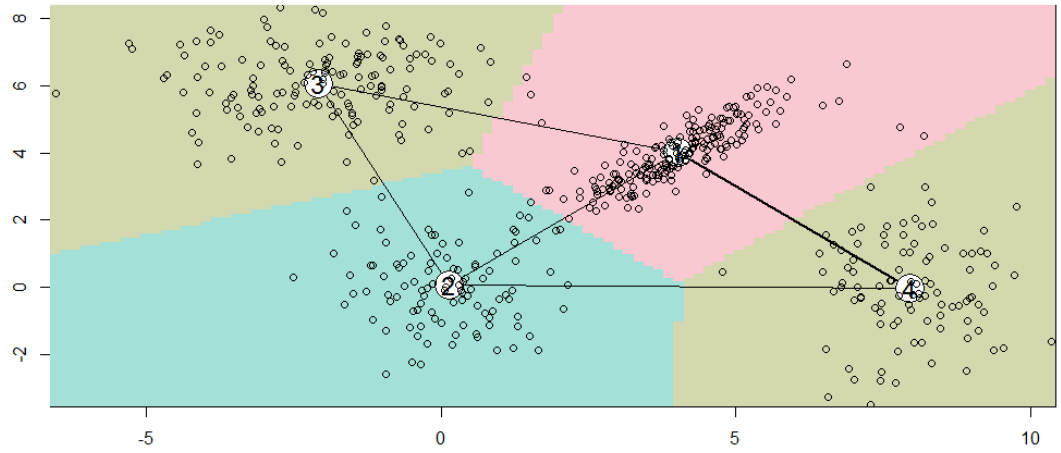
다음 [예제 2]는 패키지 {flexclust}의 kcca() 함수를 이용하여 k=평균군집을 수행한다. 패키지 {flexclust}는 다양한 시각화 기능을 제공한다.

[예제 2] Nclus는 {flexclust} 패키지에서 제공하는 데이터로써, 서로 다른 4개의 이변량 정규분포로부터 발생된 난수로 구성된 자료이다.

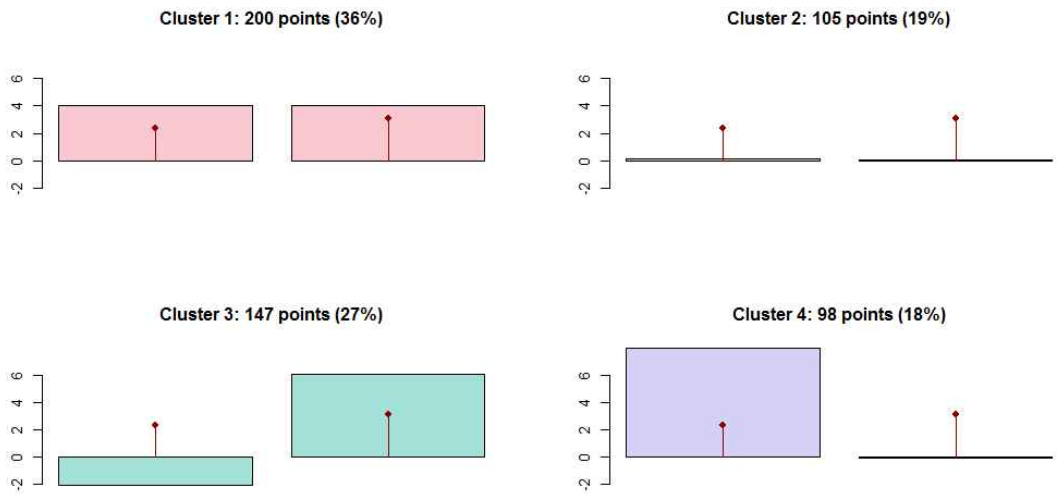
```
> library(flexclust)
> data("Nclus")
> plot(Nclus)
```




```
> cl <- kcca(Nclus, k=4, family=kccaFamily("kmeans"))
> image(cl)
> points(Nclus)
```

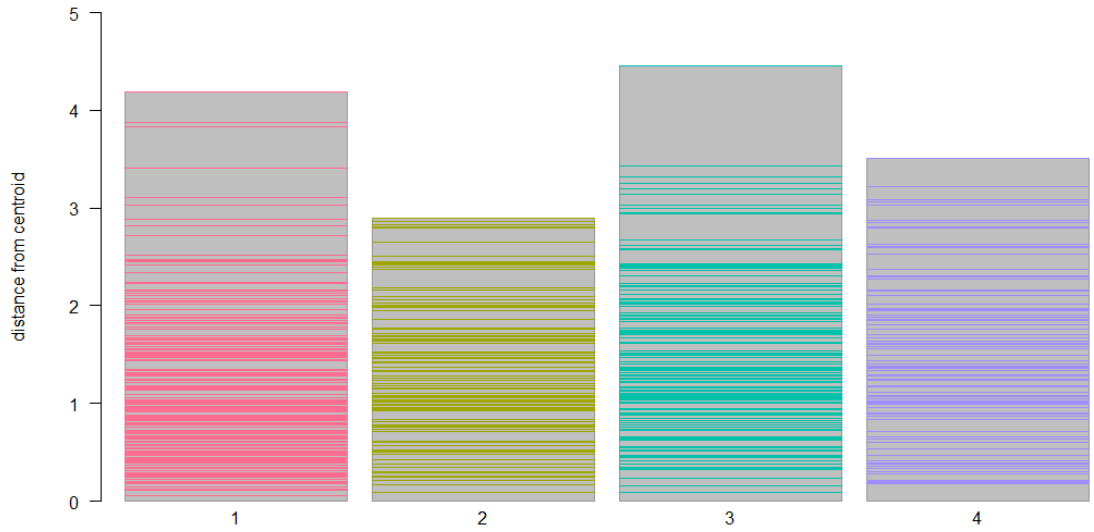


> barplot(cl)



위 그림은 각 군집의 (변수별) 중심이 전체 군집의 중심(상자 안의 막대)으로부터 얼마나 벗어나 있는지를 나타낸다.

> stripes(cl)



위 그림은 줄무늬를 이용하여 각 군집내의 자료들이 해당 군집의 평균으로부터 얼마나 떨어져 있는지를 나타낸다.

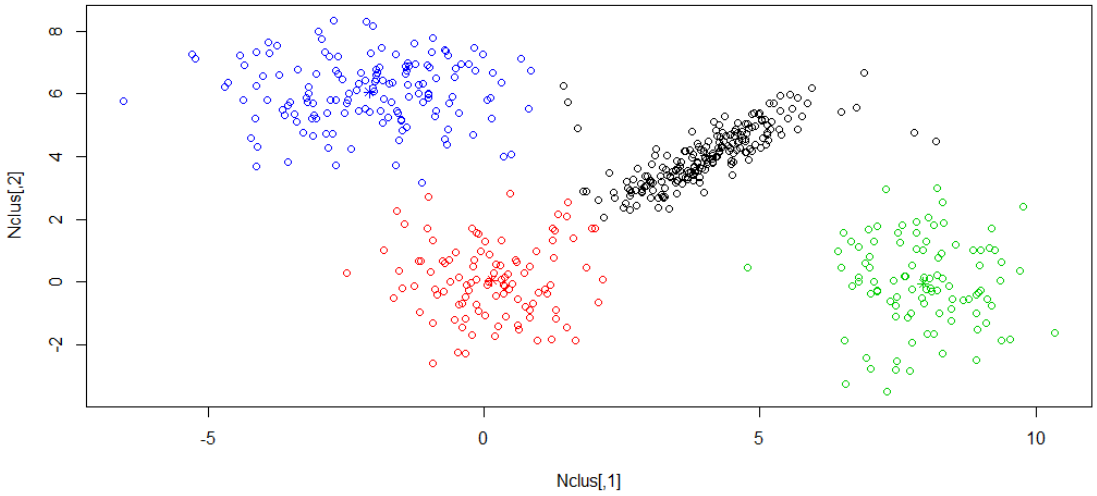
R 패키지 {flexclust}의 kcca() 함수는 k-중심군집(k-centroids clustering)을 수행한다. family= 옵션을 이용하여 “kmeans", "kmedians", "angle", "jaccard" 또는 ”ejaccard" 방법을 이용할 수 있으며, 여기서는 kmeans 옵션을 사용하였다. kcca() 함수의 적용 결과는 image{graphics}, barplot{graphics}, barchart{lattice}, stripes{flexclust} 함수 등을 통해 시각화 될 수 있다.

다음 [예제 3]은 패키지 {cclust}의 cclust() 함수를 이용하여 k-평균군집을 수행한다. cclust()는 convex clustering을 수행하는 함수로 패키지 {flexclust}에도 동일한 이름의 함수가 있으며, 두 함수의 기능은 동일하다.

[예제 3] [예제 2]와 동일한 자료를 사용하여 분석한다.

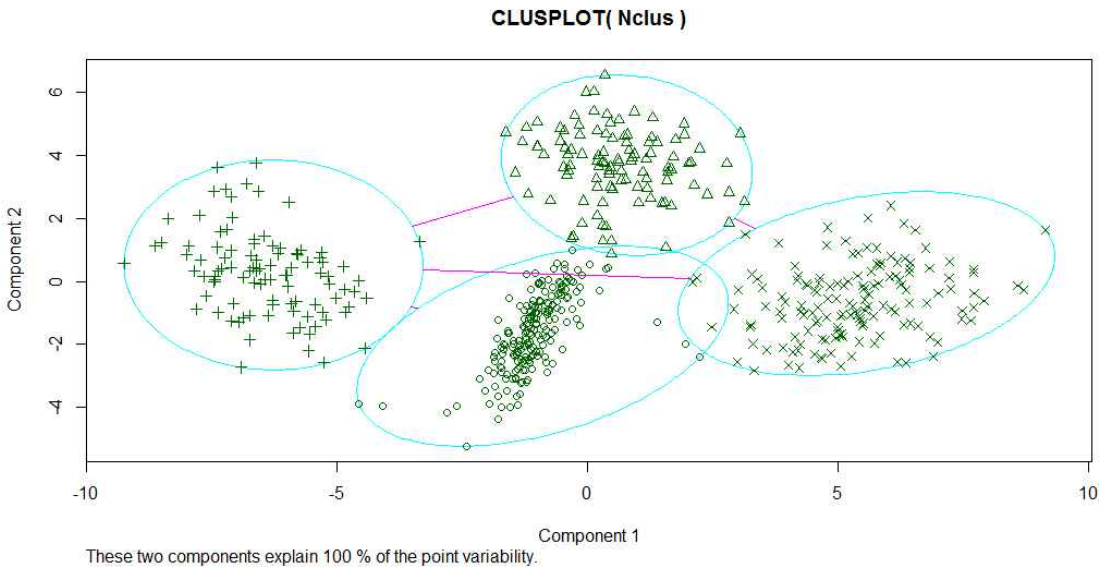
```
> library(cclust)
> cl.1 <- cclust(Nclus, 4, 20, method="kmeans")
> plot(Nclus, col=cl.1$cluster)
> points(cl.1$center, col = 1:4, pch = 8, cex=1.5)
```

cclust() 함수에서 method= 옵션에는 "kmeans", "hardcl", "neuralgas"가 있다. 이 가운데 “kmeans"는 MacQueen(1967)의 고전적인 kmeans 알고리즘을 사용하며, ”hardcl"은 hard competitive learning방법 사용하며, “neuralgas"은 이와 유사한 neural gas 알고리즘(Martinez 등, 1993)을 사용한다.



R 패키지 {cluster}의 clusplot() 함수는 2차원의 군집 그래프를 그려주는 함수이다. clusplot을 이용하면 군집의 반경과 관계까지 확인하여 볼 수 있다.

```
> library(cluster)
> clusplot(Nclus, cl.1$cluster)
```



k-평균군집을 수행하는 R 함수에는 kmeans{stats}, kcca{flexclust}, cclust{flexclust}, cclust{cclust}, Kmeans{amap} 등이 있다. Kmeans{amap}은 kmeans{stats}와 그 사용법이 유사하다. k-평균군집과 유사한 k-중앙값군집(k-medoids clusting)은 pam() 함수를 통해 수행할 수 있다. pam은 partitioning around medoids를 의미한다.