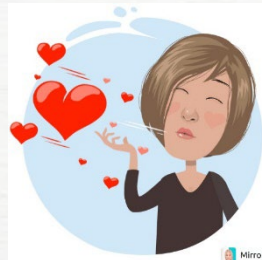




Chapter10. 군집분석



목차

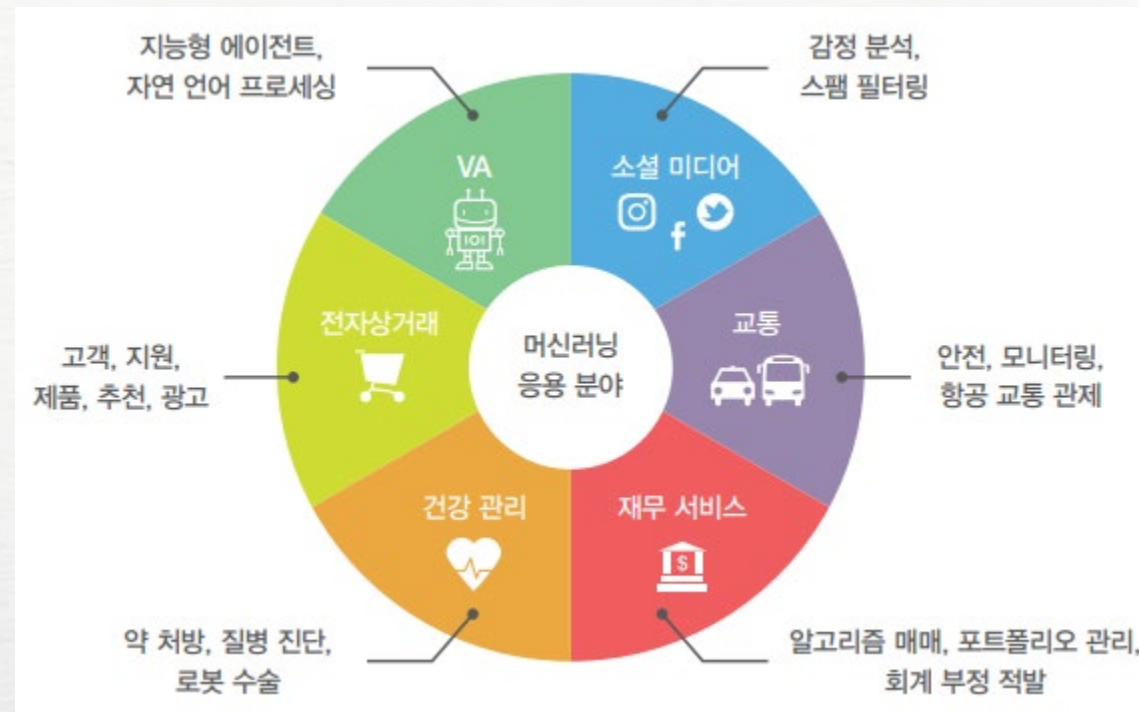
주제!

- I. 군집화와 분류의 개요
- II. 소스분석문제
- III. k-평균 군집
- IV. K-최근접 이웃 분류
- V. 응용1
- VI. 응용2

I. 군집화와 분류의 개요

1. 머신러닝(Machine Learning)의 등장

- 머신러닝은 방대한 데이터를 컴퓨터가 스스로 분석하고 학습하여 유용한 정보를 얻어내거나 미래를 예측하기 위한 예측 모델을 만들어내는 기술
- 머신러닝의 대표적 기술 → 군집화(clusterig)와 분류(classification)



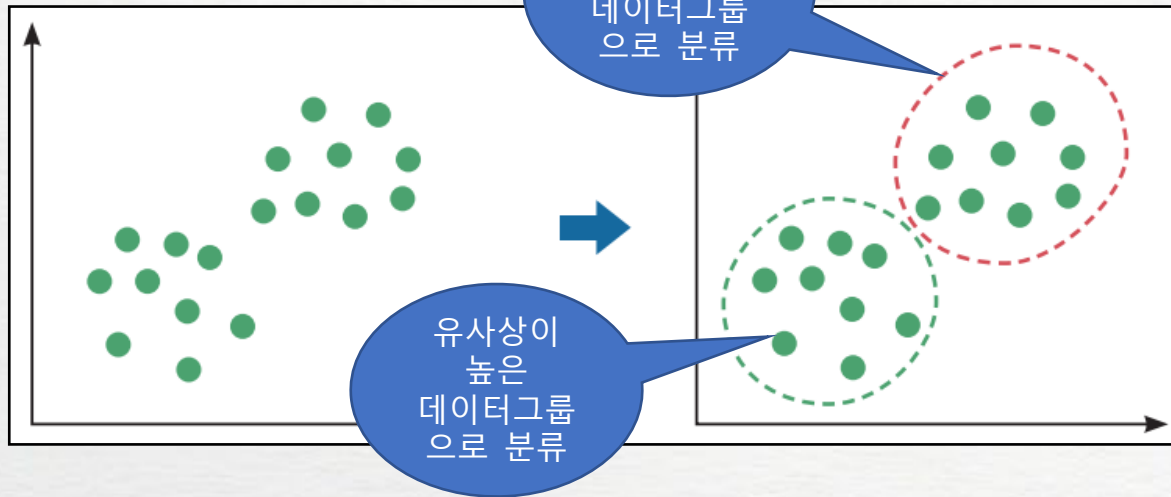
머신러닝의 응용 분야

2. 군집과 분류의 개념

군집화(clustering)

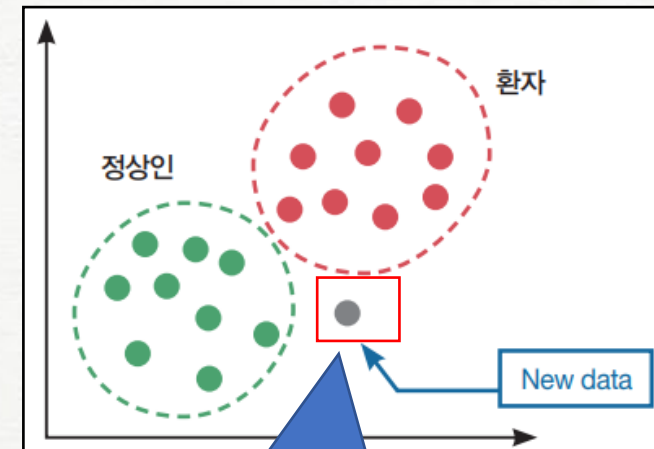
주어진 대상 데이터들을 유사성이 높은 것끼리 묶어주는 기술

=> 묶음 → 군집 cluster, 범주 category, 그룹 group, class 등 다양한 용어로 부름



분류(classification)

그룹 group, class의 형태로 알려진 데이터들이 있을 때 그룹을 모르는 어떤 데이터에 대해 어느 그룹에 속하는지를 예측하는 기술

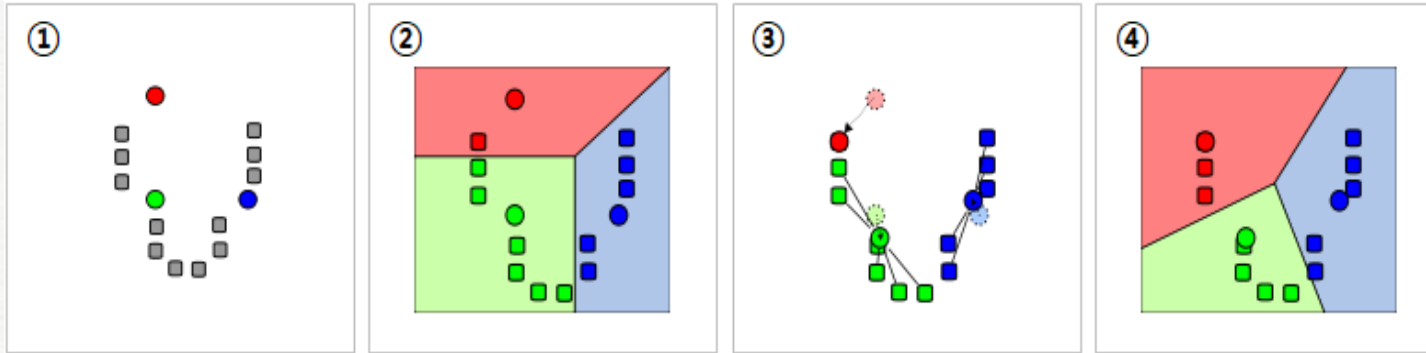


기존에 [정상인]과 [환자]로 분류된 정보를 이용하여 새로운 사용자는 자신과 비슷한 사용자가 속한 그룹을 찾음
-> 새로운 사용자도 그룹으로 분류됨

3. k-평균 군집

3.1 정의

k-평균 군집(k-Means Clustering)은 주어진 데이터를 k개의 군집으로 묶는 알고리즘으로 가장 대표적인 군집 방법



<k-평균 군집 실행 과정, wikipedia>

- ① 초기 k 평균값(위의 그림에서 k=3, 빨강, 파랑, 녹색원)은 데이터 개체 중에서 랜덤하게 선택
- ② k를 중심으로 각 데이터의 개체들은 가장 가까이 있는 평균값을 기준으로 묶음
- ③ k개 군집의 각 중심점을 기준으로 평균값이 재조정
- ④ k개 군집의 각 중심점과 각 군집 내 개체들 간의 거리의 제곱합이 최소가 될때까지 ②와 ③의 과정을 반복

3. k-평균 군집

3.2 시뮬레이션

[K-means clustering, starting with 4 left-most points \(shabal.in\)](http://shabal.in)

K-means clustering.

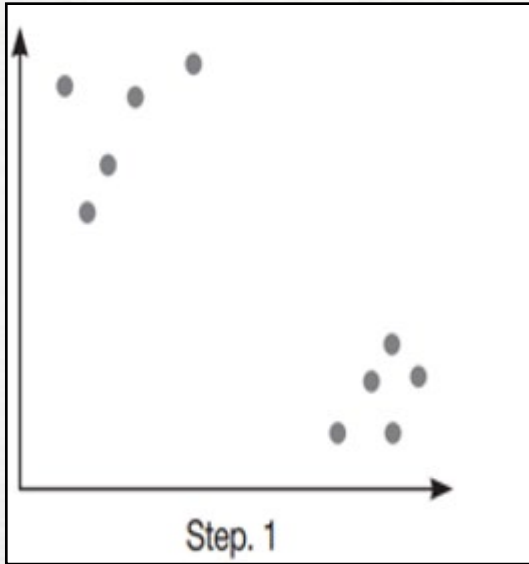
The data points. Click the picture to continue.



3. k-평균 군집

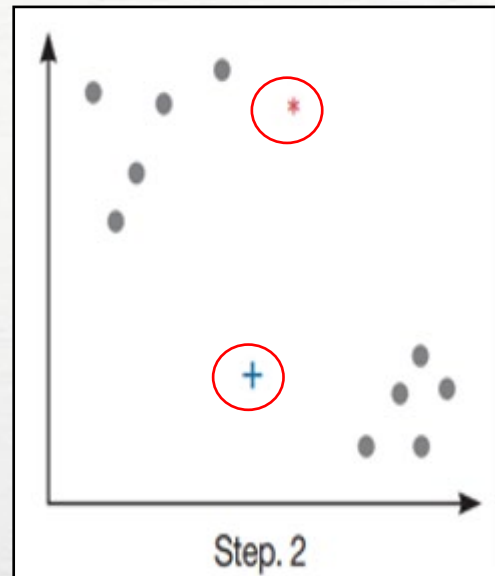
3.3 군집화 과정

군집의 개수 k 가 2인 경우를 가정하여 주어진 데이터에서 2개의 군집을 찾는 과정



1단계

대상 데이터셋을 준비한다. 이때 산점도 상의 점 하나가 관측값 하나를 의미



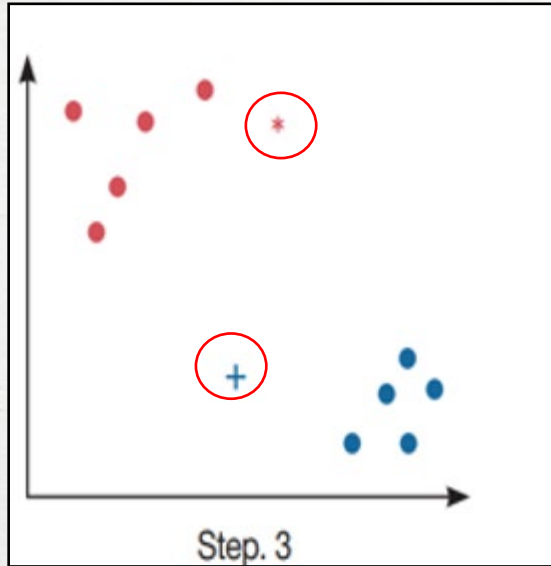
2단계

산점도 상에 임의의 점 2개(* 와 +)를 만든다. 이 2개의 점은 나중에 군집이 완성되었을 때 각 군집의 중심점이 된다. 따라서 군집의 개수만큼 임의의 점을 생성

3. k-평균 군집

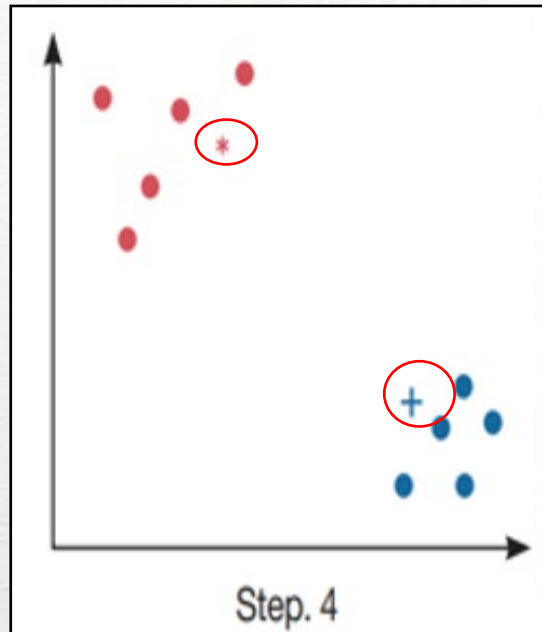
3.3 군집화 과정

군집의 개수 k 가 2인 경우를 가정하여 주어진 데이터에서 2개의 군집을 찾는 과정



3단계

산점도 상의 점들 하나하나와 임의의 점 2개와의 거리를 계산하여 두 점 중 가까운 쪽으로 군집을 형성. 그 결과 그래프의 왼쪽 위의 점들은 (*) 군집으로, 오른쪽 아래의 점들은 (+) 군집으로 묶임

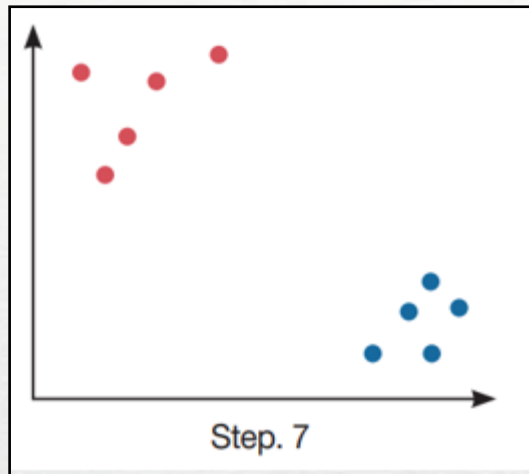
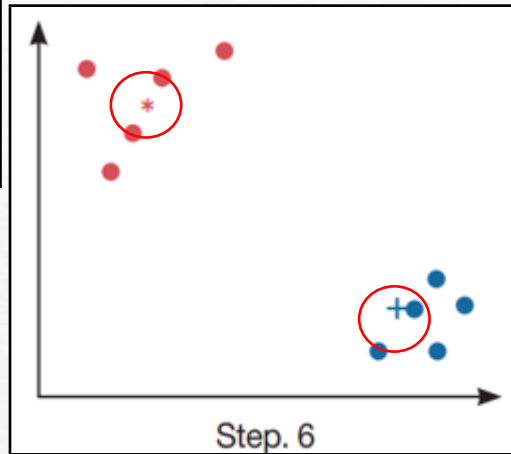
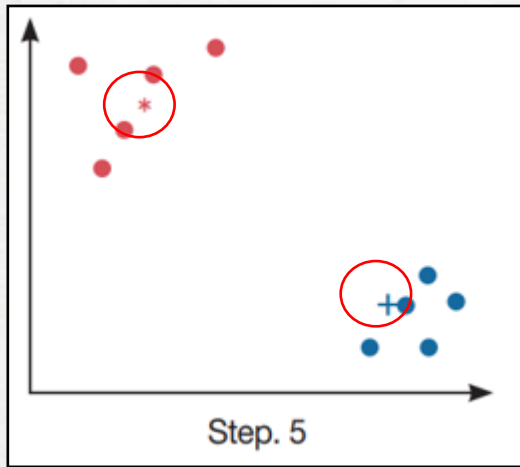


4단계

두 개의 군집에서 중심점을 다시 계산(*와 +도 포함하여 계산). (*)의 위치와 (+)의 위치를 새로 계산한 중심점의 위치로 이동

3. k-평균 군집

3.3 군집화 과정



5단계

4단계의 과정을 반복

6단계

(*)와 (+)의 위치가 더 이상 변동되지 않으면 군집의 중심점에 도달했으므로 반복을 중단

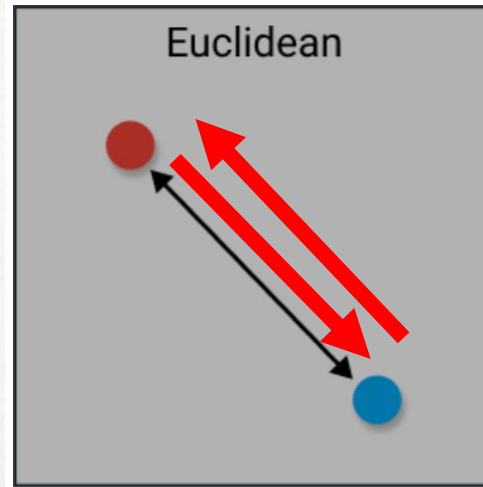
7단계

마지막으로 (*)와 가까운 점들은 (*) 군집으로, (+)와 가까운 점들은 (+) 군집으로 표시. 군집화를 종료

3. k-평균 군집

3.4 유클리디안 거리

k-평균 군집화의 방법을 정리하면 먼저 군집의 중심점을 찾고, 다른 점들은 거리가 가장 가까운 중심점의 군집에 속하는 것으로 결정

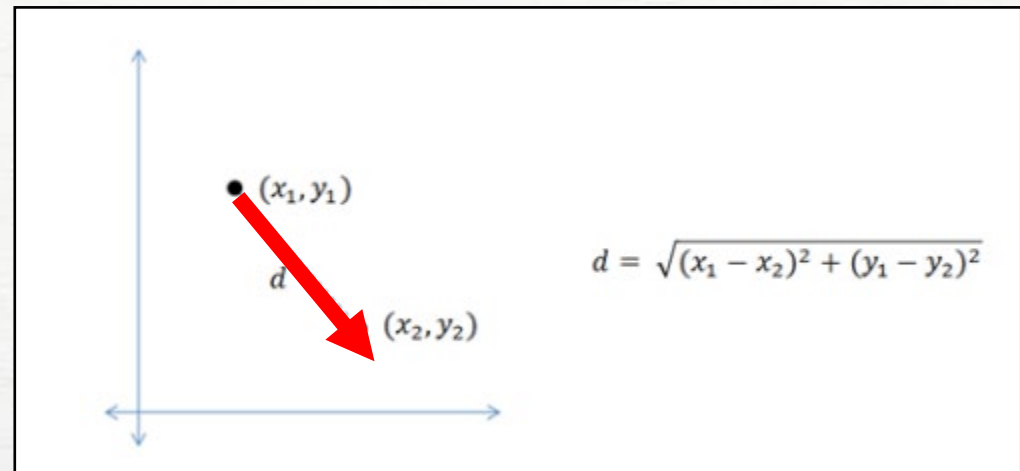
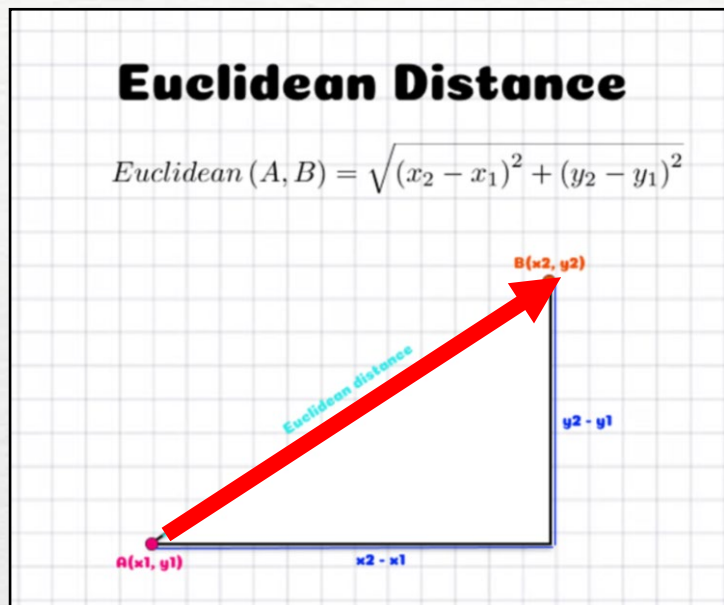


3. k-평균 군집

3.4 유클리디안 거리

- 유클리디안 거리를 이용하면 n차원 상의 점 p, q의 거리는 다음과 같이 계산

$$d(p, q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \dots + (p_n - q_n)^2} = \sqrt{\sum_{i=1}^n (p_i - q_i)^2}$$



3. k-평균 군집

3.4 유클리디안 거리

학생	키	시력
A	180	1.2
B	170	0.9

A, B 학생의 키와 시력

$$\begin{aligned}d(A, B) &= \sqrt{(180 - 170)^2 + (1.2 - 0.9)^2} \\&= \sqrt{(10)^2 + (0.3)^2} \\&= \sqrt{100 + 0.09}\end{aligned}$$

■한계점: 거리 계산에 있어서 키의 값은 많이 반영되는데(100), 시력은 거리 계산에 있어서 거의 영향을 미치지 못함(0.09)=>표준화하는 방법이 있음

=> 자료의 범위가 큰 변수가 거리 계산에 있어서 더 많은 영향을 미칠 수밖에 없다는 의미

4. iris데이터 k-평균 군집화 코드 분석

4.1 군집화 코드

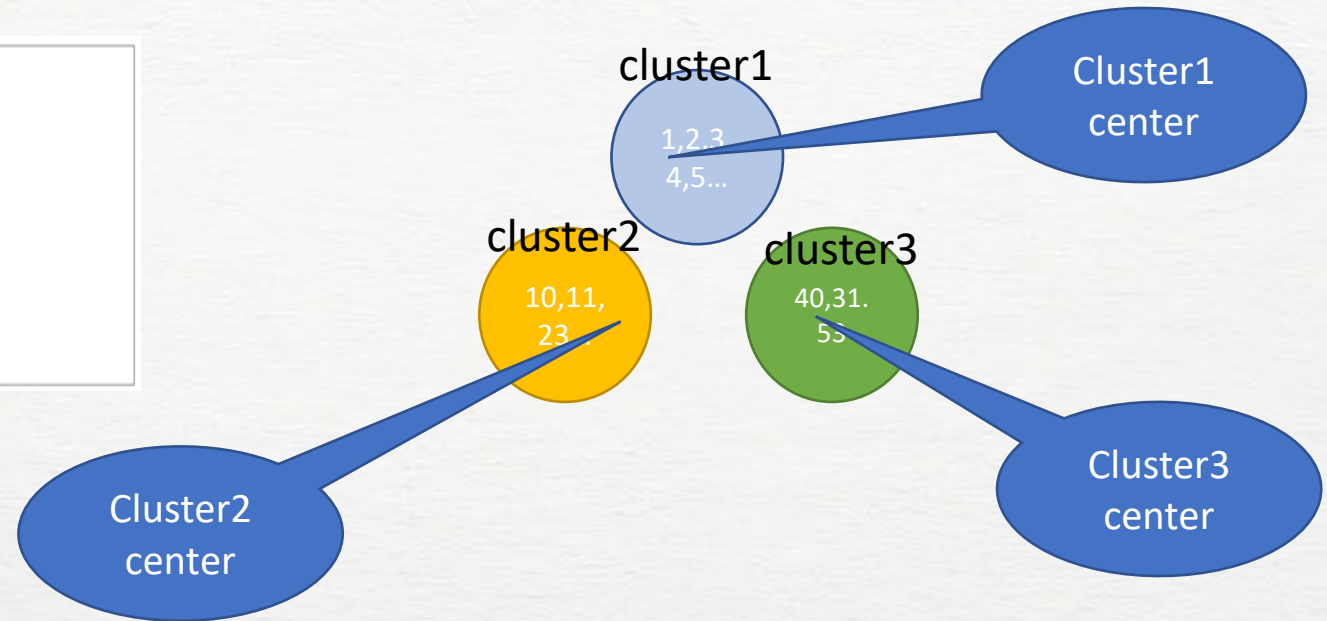
```
# Sepal.Length, Sepal.Width, Petal.Length, Petal.Width 4개의 변수  
mydata <- iris[,1:4] # 데이터 준비-4개의 변수,[,]는 모든행  
  
# mydata를 3개의 군집으로 군집화  
fit <- kmeans(x=mydata, centers=3)  
  
fit$cluster # 각 데이터에 대한 군집 번호  
fit$centers # 각 군집의 중심점 좌표
```

4개의 변수를 사용하여 비슷한
꽃 끼리 군집을 만들

사용하지 않음

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

- `x=mydata`
매개변수 x에는 군집화의 대상이 되는 데이터셋을 지정한다.
- `centers=3`
매개변수 centers에는 군집의 개수를 입력한다.



4. iris데이터 k-평균 군집화 코드 분석

4.2 그래프그리기

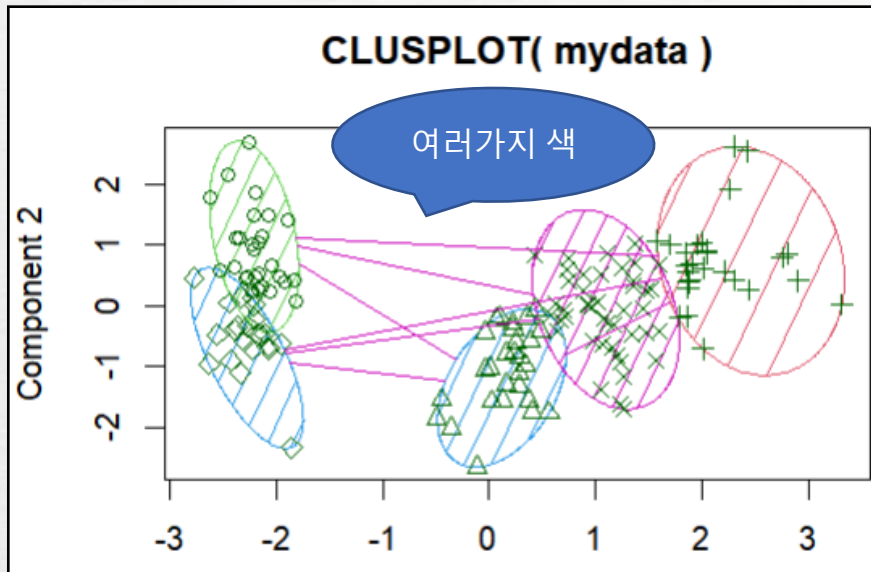
```
clusplot(mydata, fit$cluster, color=TRUE, shade=TRUE, labels=1, lines=0)
```

shade: FALSE: 무늬 없음, TRUE: 무늬 있음

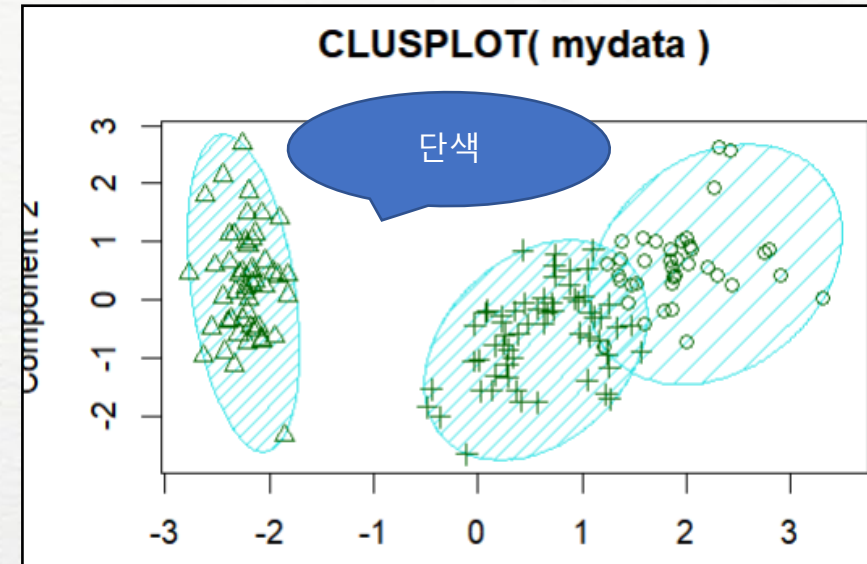
labels: 2=>레이블을 표시, 1=>레이블을 표시하지 않을

lines: 0=> 선이 그려지지 않음
2=>각 군집의 중심에서 군집의 경계까지 선이 그려짐
1=>각 군집의 중심에서 각 데이터 포인트까지 선이 그려짐

```
clusplot(mydata, fit$cluster, color=TRUE, shade=TRUE, labels=1, lines=0)
```



```
clusplot(mydata, fit$cluster, color=FALSE, shade=TRUE, labels=1, lines=0)
```



4. iris데이터 k-평균 군집화 코드 분석

4.2 그래프그리기

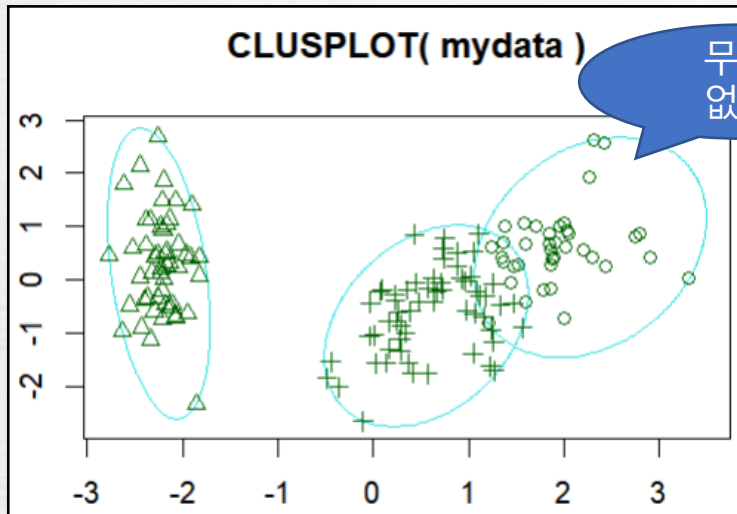
```
clusplot(mydata, fit$cluster, color=TRUE, shade=TRUE, labels=1, lines=0
```

shade: FALSE: 무늬 없음, TRUE: 무늬가 있음

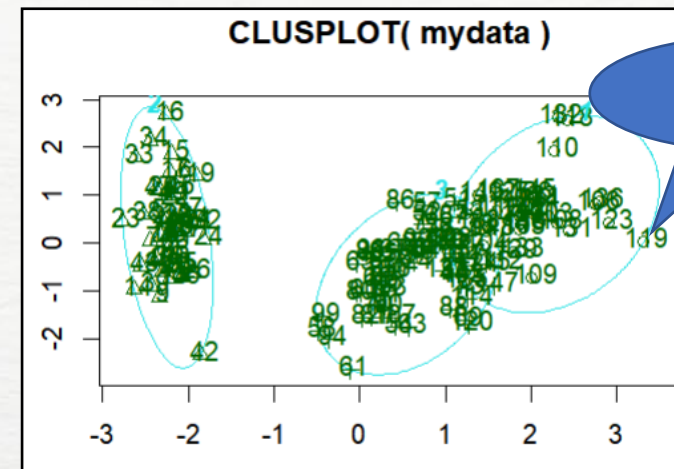
labels: 2=>레이블을 표시, 1=>레이블을 표시하지 않을

lines: 0=> 선이 그려지지 않음
2=>각 군집의 중심에서 군집의 경계까지 선이 그려짐
1=>각 군집의 중심에서 각 데이터 포인트까지 선이 그려짐

```
clusplot(mydata, fit$cluster, color=FALSE, shade=FALSE, labels=1, lines=0)
```



```
clusplot(mydata, fit$cluster, color=FALSE, shade=FALSE, labels=2, lines=0)
```



4. iris데이터 k-평균 군집화 코드 분석

4.2 그래프그리기

```
clusplot(mydata, fit$cluster, color=TRUE, shade=TRUE, labels=1, lines=0
```

shade: FALSE: 무늬 없음, TRUE: 무늬 있음

labels: 2=>레이블을 표시, 1=>레이블을 표시하지 않을

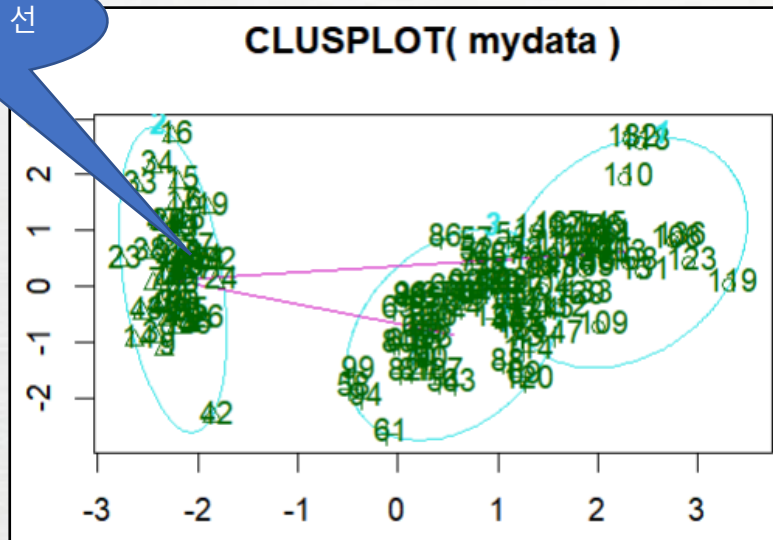
lines: 0=> 선이 그려지지 않음

1=>각 군집의 중심에서 각 데이터 포인트까지 선이 그려짐

2=>각 군집의 중심에서 군집의 경계까지 선이 그려짐

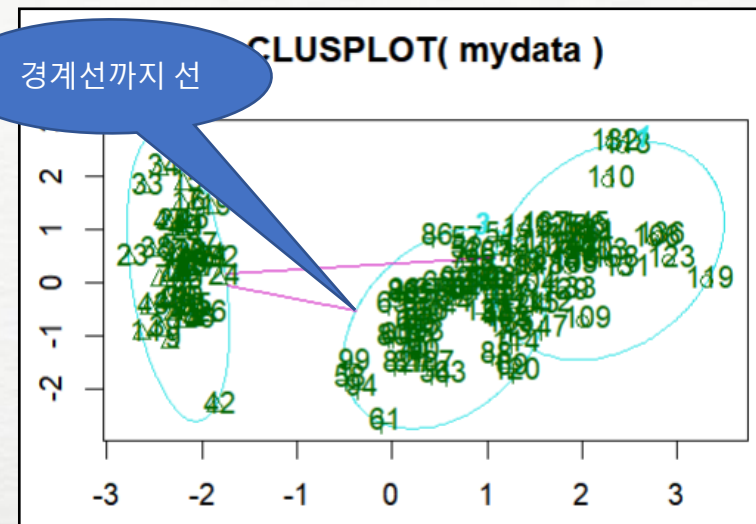
```
clusplot(mydata, fit$cluster, color=FALSE, shade=FALSE, labels=2, lines=1)
```

포인트까지 선



```
clusplot(mydata, fit$cluster, color=FALSE, shade=FALSE, labels=2, lines=2)
```

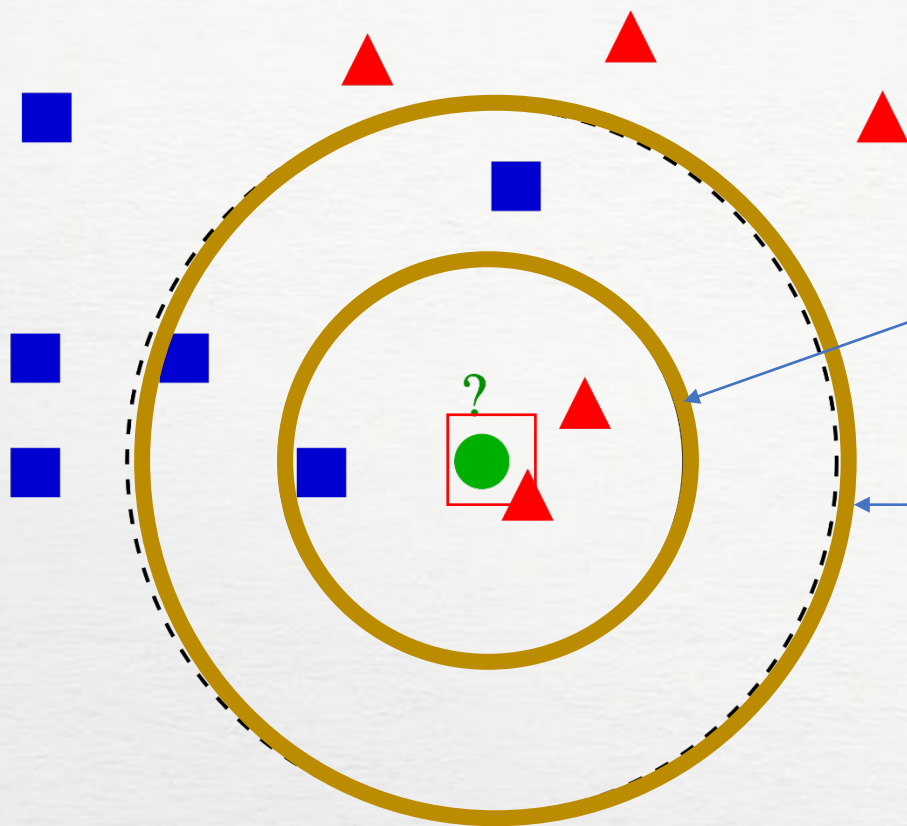
경계선까지 선



5. k-최근접 이웃 분류

5.1 정의

- K-최근접 이웃(k-NN, k-Nearest Neighbor)은 간단한 머신러닝 알고리즘으로 새로운 데이터에 대해 이와 가장 거리가 가까운 k개의 과거 데이터의 결과를 이용해 다수결로 분류하는 방법

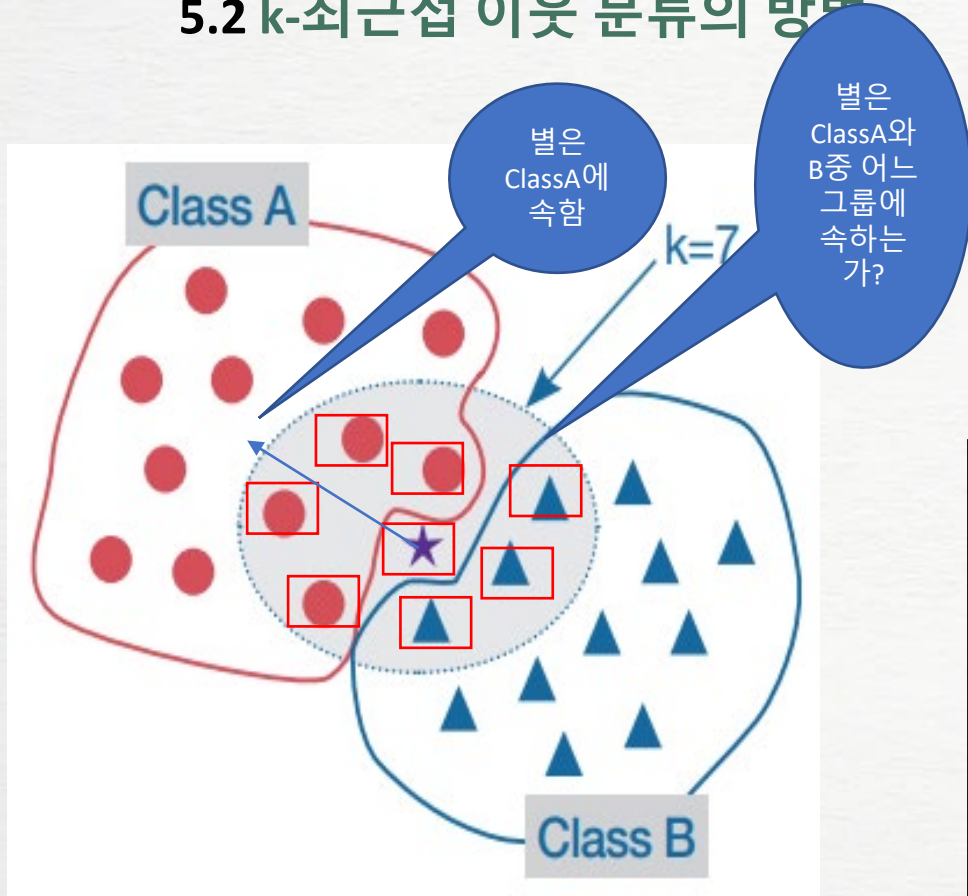


- 녹색 원은 새로운 데이터인데 과거 데이터를 이용해 파란색 네모 또는 빨간색 세모로 분류하고자 함
- 실선으로 된 원은 k가 3개인데 네모가 1개, 세모가 2개로 다수결에 의해 원은 세모로 분류 됨
- 점선으로 된 원은 k가 5개인데 네모가 3개, 세모가 2개로 다수결에 의해 원은 네모로 분류 됨

5. k-최근접 이웃 분류

1단계

5.2 k-최근접 이웃 분류의 방법



그룹을 모르는 데이터 p에 대해 이미 그룹이 알려진 데이터 중 p와 가장 가까이에 있는 k (7)개의 데이터를 수집

2단계

k개의 데이터가 가장 많이 속해 있는 군집을 p의 군집으로 정함

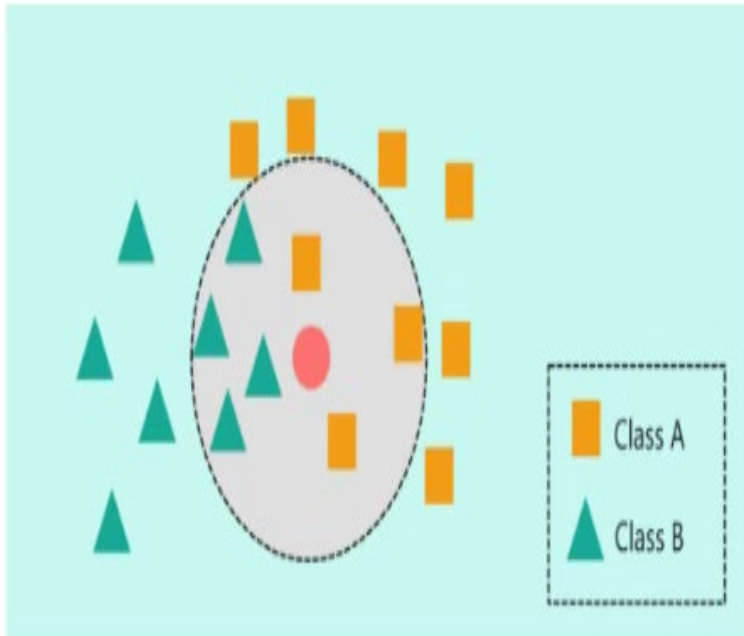
- k-최근접 이웃 알고리즘에서 k=7이라고 했을 때, 별점과 가장 가까이에 있는 7개의 점들을 찾아보면 회색 타원 안에 있는 점
- 이때 7개의 점 중에서 4개는 Class A에 속하고 3개는 Class B에 속하는데, 다수결에 의해 이웃의 점 4개가 속해 있는 Class A를 별점의 그룹이라고 예측

k-최근접 이웃 알고리즘 설명(k=7)

5. k-최근접 이웃 분류

5.2 k-최근접 이웃 분류의 방법

k개의 이웃이 속한 군집을 p의 군집으로 정함



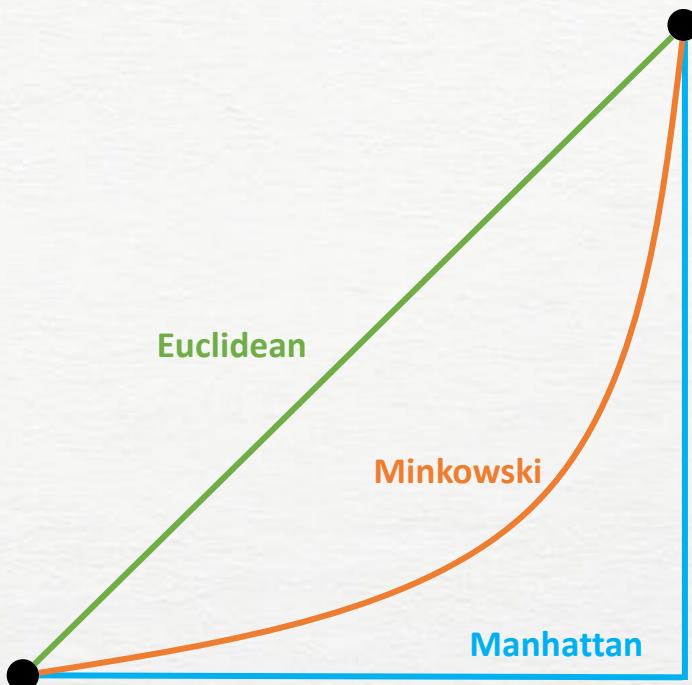
k-최근접 이웃 알고리즘 설명(k=7)

- k-최근접 이웃 알고리즘에서 k=7이라고 했을 때, 별점과 가장 가까이에 있는 7개의 점들을 찾아보면 회색 타원 안에 있는 점
- 이때 7개의 점 중에서 4개는 Class A에 속하고 3개는 Class B에 속하는데, 다수결에 의해 이웃의 점 4개가 속해 있는 Class A를 별점의 그룹이라고 예측

5. k-최근접 이웃 분류

5.3 k-최근접 이웃과 거리 계산

- 새 데이터에 더 가까운 이웃일수록 더 먼 이웃보다 평균에 더 많이 기여하도록 이웃의 기여에 가중치(weight)를 줄 수 있음
- 예를들어, 이웃까지의 거리가 d라면 해당 이웃들에게는 거리의 반비례인 1/d만큼의 가중치를 부여할 수 있으며 아래는 대표적인 거리 계산 방법

거리 종류	계산식	예시
Euclidean (유클리디안)	$\sqrt{\sum_{i=1}^k (x_i - y_i)^2}$	
Minkowski (민코프스키)	$\left(\sum_{i=1}^k (x_i - y_i)^p \right)^{1/p}$	
Manhattan (맨해튼)	$\sum_{i=1}^k x_i - y_i $	

6. k-최근접 이웃 분류를 이용한 꽃의 종 분류 코드

6.1 데이터 준비 코드

훈련용 데이터와 테스트용 데이터 준비

```
tr.idx <- c(1:4)
```

```
ds.tr <- iris[tr.idx, 1:2]
```

```
ds.ts <- iris[-tr.idx, 1:2]
```

```
cl.tr <- factor(iris[tr.idx, 5])
```

```
cl.ts <- factor(iris[-tr.idx, 5])
```

행의 수-훈련용 데이터의 인덱스(1번, 2번, 3번, 4번)

(tr.idx) 1행~4행: 훈련용 데이터셋(열은 Sepal.Length, Sepal.Width)

5행~6행: 테스트용 데이터셋(열은 Sepal.Length, Sepal.Width)

훈련용 데이터셋의 그룹(품종) 정보(5열은 품종정보를 나타냄)

테스트용 데이터셋의 그룹(품종) 정보

ds.tr: 1행~4행: 훈련용 데이터셋

cl.tr: 훈련용
데이터셋의
그룹(품종) 정보

	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	versicolor
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	virginica
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	virginica

ds.ts: 5행~6행: 테스트용
데이터셋

cl.ts: 테스트용 데이터셋의
그룹(품종) 정보

테스트데이터셋의 그룹 정보를
모른다고 가정하고 그룹정보를 예측

6. k-최근접 이웃 분류를 이용한 꽃의 종 분류 코드

6.2 k-최근접 이웃 분류

훈련용 데이터와 테스트용 데이터 준비

```
tr.idx <- c(1:4)
```

```
ds.tr <- iris[tr.idx, 1:2]
```

```
ds.ts <- iris[-tr.idx, 1:2]
```

```
cl.tr <- factor(iris[tr.idx, 5])
```

```
cl.ts <- factor(iris[-tr.idx, 5])
```

```
pred <- knn(ds.tr, ds.ts, cl.tr, k=3, prob=TRUE)
```

훈련용 데이터의 인덱스

1행~4행: 훈련용 데이터셋, 변수는 2개

5행~6행: 테스트용 데이터셋, 변수는 2개

훈련용 데이터셋의 그룹(품종) 정보(5열)

테스트용 데이터셋의 그룹(품종) 정보(5열)

	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	versicolor
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	virginica
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	virginica

pred

setosa
versicolor

- ds.tr

훈련용 데이터셋을 지정한다.

- ds.ts

테스트용 데이터셋을 지정한다.

- cl.tr

훈련용 데이터셋의 그룹 정보를 지정한다.

- k=3

최근접 이웃의 개수를 지정한다.

- prob=TRUE

예측된 그룹에 대한 지지 확률을 표시할지 여부를 결정한다. 위와 같이 k=3인 경우 최근접 이웃이 Class A가 2개, Class B가 1개라면 예측된 그룹은 Class A이고 Class A에 대한 지지 확률은 $2/3(=0.6666)$ 이다.

테스트데이터셋의 그룹 정보를
모른다고 가정하고 그룹정보를 예측

6. k-최근접 이웃 분류를 이용한 꽃의 종 분류 코드

6.3 정확도 분석

훈련용 데이터와 테스트용 데이터 준비

```
tr.idx <- c(1:4)
```

```
ds.tr <- iris[tr.idx, 1:2]
```

```
ds.ts <- iris[-tr.idx, 1:2]
```

```
cl.tr <- factor(iris[tr.idx, 5])
```

```
cl.ts <- factor(iris[-tr.idx, 5])
```

```
pred <- knn(ds.tr, ds.ts, cl.tr, k=3, prob=TRUE)
```

```
acc <- mean(pred==cl.ts)
```

훈련용 데이터의 인덱스

1행~4행:훈련용 데이터셋, 변수는2개

5행~6행:테스트용 데이터셋, 변수는2개

훈련용 데이터셋의 그룹(품종) 정보(5열_Species)

테스트용 데이터셋의 그룹(품종) 정보(5열 _Species)

예측 정확도

	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	versicolor
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	versicolor
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	virginica

cl.ts

==

pred

같음

다름

50% ← acc(예측 정확도)

7. k-최근접 이웃 분류를 이용한 병아리 품종 분류 코드

7.1 k-최근접 이웃 실시

k-최근접 이웃 실시 명령어

```
c_knn3 <- kNN( Y ~ x1+x2..., 훈련데이터셋, 테스트데이터셋, k = 이웃의수)
```

```
> c_train <- read.csv("a.csv", header = TRUE) # 훈련용 데이터셋 불러오기  
> c_test <- read.csv("b.csv", header = TRUE) # 테스트용 데이터셋 불러오기
```

```
# 병아리 품종을 종속변수로 두고 나머지 변수 전체를 독립변수 설정, k = 3으로 해봄
```

```
> c_knn3 <- kNN(breeds ~ wing_length+ tail_length+ comb_height, c_train, c_test, k = 3)
```

훈련

+

+

➡ breeds 학습

```
> c_train  
  wing_length tail_length comb_height breeds  
1         238         63         34      a  
2         236         67         30      a  
3         256         65         34      a  
4         240         63         35      a  
5         246         65         30      a
```

테스트

+

+

➡ breeds 예측

```
> c_test  
  wing_length tail_length comb_height breeds c_knn3  
1         258         67         32      ?      ?  
2         260         64         34      ?      ?  
3         251         63         31      ?      ?  
4         248         63         30      ?      ?  
5         254         62         32      ?      ?  
-         ...         ...         ...      ?      ?
```

7. k-최근접 이웃 분류를 이용한 병아리 품종 분류 코드

7.2 정확도 분석

```
> c_knn3 <- kNN(breeds ~ wing_length+ tail_length+ comb_height, c_train, c_test, k = 3)
> c_test$pred3 <- c_knn3
> acc <- mean(c_test$breeds == c_test$pred3 )
```

예측값을 c_test 데이터셋에 pred3열을 만들어서 입력
예측 정확도

테스트				+ + → breeds예측	
> c_test				c_knn3	
	wing_length	tail_length	comb_height	breeds	
1	258	67	32		? a
2	260	64	34		? a
3	251	63	31		? a
4	248	63	30		? a
5	254	62	32		? a
-	---	--	--		

acc->예측한 값이 실제 값과 같은가?

II. 코드분석문제

1. k-평균 군집(문제1)

* 유클리디안 거리

학생	키	시력
A	180	1.2
B	170	0.9
C	100	1.5

A, B, C 학생의 키와 시력

문제1_B와 C와의 거리를 계산하도록 ?,??를 채우시오.

$$\begin{aligned}d(A,B) &= \sqrt{(180 - 170)^2 + (1.2 - 0.9)^2} \\&= \sqrt{(10)^2 + (0.3)^2} \\&= \sqrt{100 + 0.09}\end{aligned}$$

A는 C보다
B가 더
가깝다.

A와 C간의 거리

$$\begin{aligned}d(A,C) &= \sqrt{(180 - 100)^2 + (1.2 - 1.5)^2} \\&= \sqrt{(80)^2 + (0.3)^2} \\&= \sqrt{1600 + 0.09}\end{aligned}$$

B와 C간의 거리

$$d(B,C) = \sqrt{(? - 100)^2 + (?? - 1.5)^2}$$

2. iris데이터 k-평균 군집화 코드 분석(

문제2_3개의 변수를 사용하여 2개의 꽃 군집을 만들도록 (가), (나)를 쓰시오.

* 군집화 코드

```
#Sepal.Length, Sepal.Width, Petal. Length
```

```
mydata <- iris[ (가) ] # 데이터 준비(변수3개)
```

```
# mydata를 2개의 군집으로 군집화
```

```
fit <- kmeans( (나) ) #군집 2개
```

```
fit$cluster # 각 데이터에 대한 군집 번호
```

```
fit$centers # 각 군집의 중심점 좌표
```

3개의 변수를 사용하여 비슷한
꽃끼리 군집을 만들

사용하지 않음

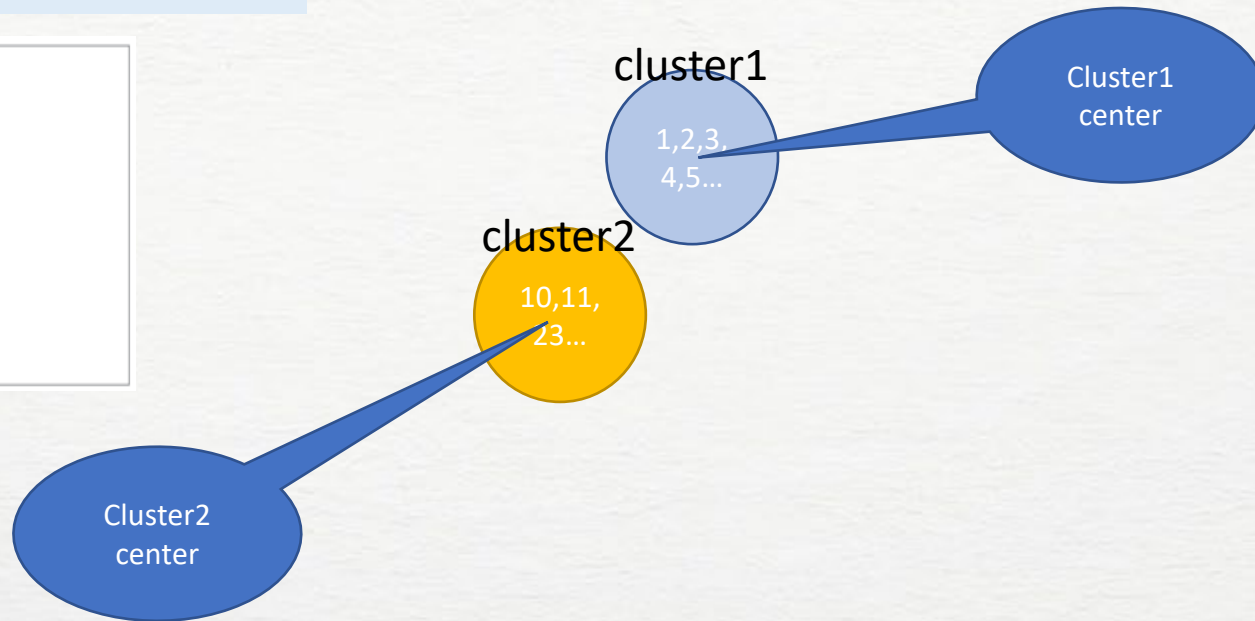
	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	
2	4.9	3.0	1.4	0.2	
3	4.7	3.2	1.3	0.2	
4	4.6	3.1	1.5	0.2	
5	5.0	3.6	1.4	0.2	
6	5.4	3.9	1.7	0.4	

- `x=mydata`

매개변수 x에는 군집화의 대상이 되는 데이터셋을 지정한다.

- `centers=3`

매개변수 centers에는 군집의 개수를 입력한다.



3. iris데이터 k-평균 군집화 코드

문제3. 아래 그림과 같이 군집이 보이도록 (가), (나)를 쓰시오.

* 그래프그리기

```
clusplot(mydata, fit$cluster, color=TRUE, shade=(가), labels=1, lines=(나))
```

shade: FALSE-무늬없음, TRUE-무늬있음

labels: 2=>레이블을 표시, 1=>레이블을 표시하지 않을

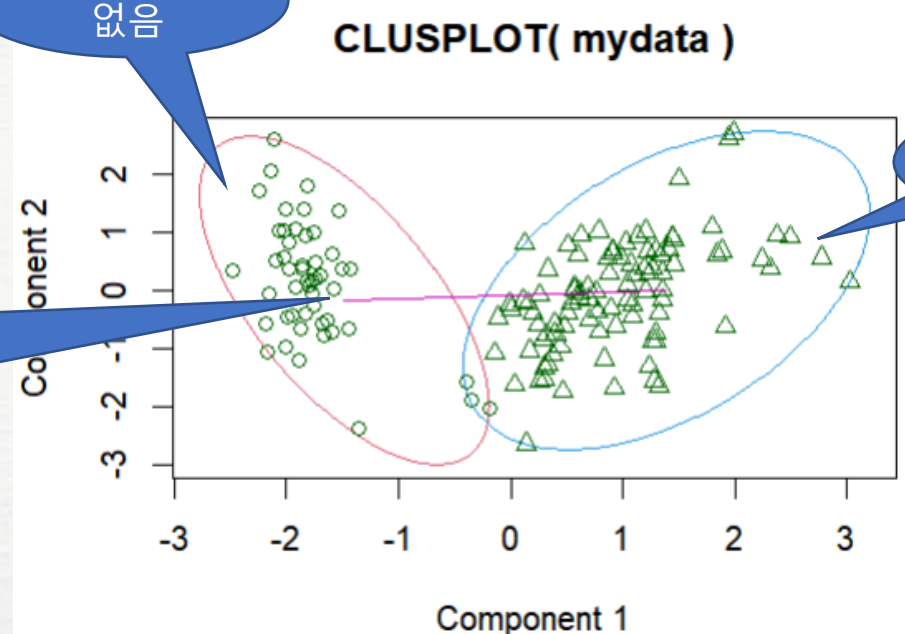
lines: 0=> 선이 그려지지 않음

2=>각 군집의 중심에서 군집의 경계까지 선이 그려짐

1=>각 군집의 중심에서 각 데이터 포인트까지 선이 그려짐

무늬
없음

군집의
중심에서
각 데이터
포인트까
지 선이
그려짐



레이블
안보임

4. k-최근접 이웃 분류를 이용한 꽃의 종 분류 코드(문제4)

* 데이터 준비 코드

훈련용 데이터와 테스트용 데이터 준비

```
tr.idx <- c(가)
```

```
ds.tr <- iris[tr.idx, 1:4]
```

```
ds.ts <- iris[-tr.idx, 1:4]
```

```
cl.tr <- factor(iris[tr.idx, 5])
```

```
cl.ts <- factor(iris[-tr.idx, 5])
```

문제4. 아래 그림과 같이 1~5행은 훈련데이터셋으로
6행은 테스트 데이터셋으로 구성하도록 (가)를 쓰시오.

훈련용 데이터의 인덱스(1행~5행)

1행~5행:훈련용 데이터셋(변수는 4개 모두)

6행:테스트용 데이터셋(6개 중 -(1:5)이므로 6행만 남음)

훈련용 데이터셋의 그룹(품종) 정보

테스트용 데이터셋의 그룹(품종) 정보

ds.tr: 1행~5행:훈련용 데이터셋

cl.tr: 훈련용
데이터셋의
그룹(품종) 정보

	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	versicolor
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	virginica
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	virginica

테스트데이터셋의 그룹 정보를
모른다고 가정하고 그룹정보를 예측

ds.ts: 6행:테스트용 데이터셋

cl.ts: 테스트용 데이터셋의
그룹(품종) 정보

5. k-최근접 이웃 분류를 이용한 꽃의

문제5. 네개의 이웃을 기준으로 분류하기 위해 (가)를 쓰시오.

* k-최근접 이웃 분류

훈련용 데이터와 테스트용 데이터 준비

```
tr.idx <- c( 문제4 )
```

```
ds.tr <- iris[tr.idx, 1:4]
```

```
ds.ts <- iris[-tr.idx, 1:4]
```

```
cl.tr <- factor(iris[tr.idx, 5])
```

```
cl.ts <- factor(iris[-tr.idx, 5])
```

훈련용 데이터의 인덱스

1행~5행:훈련용 데이터셋

6행:테스트용 데이터셋

훈련용 데이터셋의 그룹(품종) 정보

테스트용 데이터셋의 그룹(품종) 정보

```
predict1 <- knn(ds.tr, ds.ts, cl.tr, (가) ) #4개의 이웃
```

	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	versicolor
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	virginica
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	virginica

predict1

versicolor

- ds.tr

훈련용 데이터셋을 지정한다.

- ds.ts

테스트용 데이터셋을 지정한다.

- cl.tr

훈련용 데이터셋의 그룹 정보를 지정한다.

- k=3

최근접 이웃의 개수를 지정한다.

- prob=TRUE

예측된 그룹에 대한 지지 확률을 표시할지 여부를 결정한다. 위와 같이 k=3인 경우 최근접 이웃이 Class A가 2개, Class B가 1개라면 예측된 그룹은 Class A이고 Class A에 대한 지지 확률은 $2/3(=0.6666)$ 이다.

테스트데이터셋의 그룹 정보를 모른다고 가정하고 그룹정보를 예측

6. k-최근접 이웃 분류를 이용한 꽃의 종 분류 코드(문제6)

*정확도 분석

문제6. 예측 정확도를 구하기 위하여 (가)를 쓰시오.

훈련용 데이터와 테스트용 데이터 준비

```
tr.idx <- c(1:5)
```

```
ds.tr <- iris[tr.idx, 1:4]
```

```
ds.ts <- iris[-tr.idx, 1:4]
```

```
cl.tr <- factor(iris[tr.idx, 5])
```

```
cl.ts <- factor(iris[-tr.idx, 5])
```

```
predict1 <- knn(ds.tr, ds.ts, cl.tr, k=4, prob=TRUE)
```

```
acc <- mean( (가) )
```

훈련용 데이터의 인덱스

1행~5행:훈련용 데이터셋

6행:테스트용 데이터셋

훈련용 데이터셋의 그룹(품종) 정보

테스트용 데이터셋의 그룹(품종) 정보

예측 정확도

	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	versicolor	
3	4.7	3.2	1.3	0.2	setosa	
4	4.6	3.1	1.5	0.2	versicolor	
5	5.0	3.6	1.4	0.2	setosa	
6	5.4	3.9	1.7	0.4	virginica	

cl.ts

==

predict1

versicolor

다름

0% ← acc(예측 정확도)

7. k-최근접 이웃 분류를 이용한 병아리 품종 분류 코드(문제7)

* k-최근접 이웃 실시

문제7. wing_length와 tail_length를 사용하여 breeds를 예측하도록 (가)를 쓰시오.

k-최근접 이웃 실시 명령어

```
c_knn2 <- kNN( Y ~ x1+x2..., 훈련데이터셋, 테스트데이터셋, k = 이웃의수)
```

```
> c_train <- read.csv("a.csv", header = TRUE) # 훈련용 데이터셋 불러오기
> c_test <- read.csv("b.csv", header = TRUE) # 테스트용 데이터셋 불러오기

# 병아리 품종을 종속변수로 두고 나머지 변수 전체를 독립변수 설정, k = 2으로 해봄
> c_knn2 <- kNN(breeds ~ (가), c_train, c_test, k = 2)
```

훈련

+

➡ breeds학습

```
> c_train
  wing_length tail_length comb_height breeds
1         238         63         34      a
2         236         67         30      a
3         256         65         34      a
4         240         63         35      a
5         246         65         30      a
```

테스트

+

➡ breeds예측

```
> c_test
  wing_length tail_length comb_height breeds c_knn2
1         258         67         32      ?      ?
2         260         64         34      ?      ?
3         251         63         31      ?      ?
4         248         63         30      ?      ?
5         254         62         32      ?      ?
-         ---         --         --      -      -
```

8. k-최근접 이웃 분류를 이용한 병아리 품종 분류 코드(문제8)

* 정확도 분석

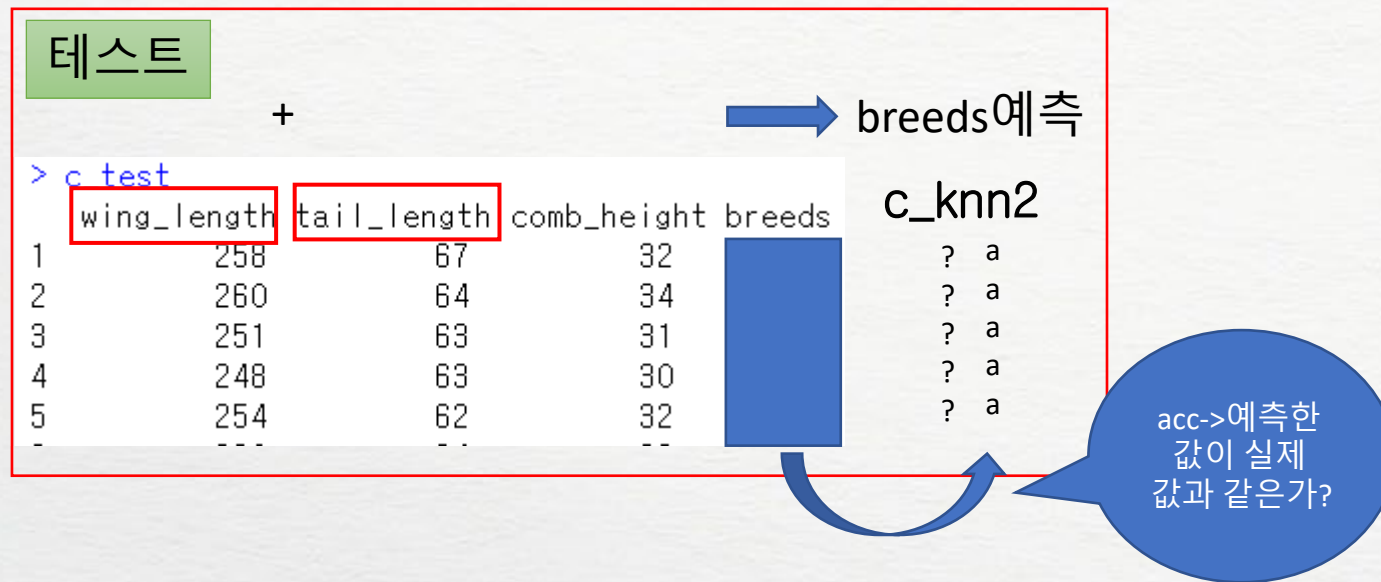
문제8. wing_length와 tail_length를 사용하여 예측된 breeds의 정확도를 평가하도록 (가)를 쓰시오.

```
> c_knn2 <- kNN(breeds ~ wing_length+ tail_length, c_train, c_test, k = 2)
```

```
> c_test$pred2 <- c_knn2
```

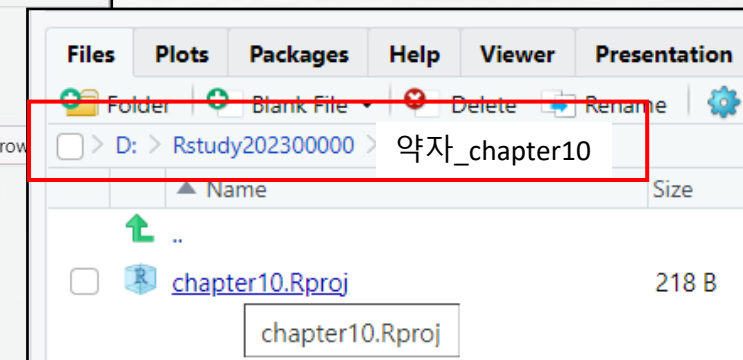
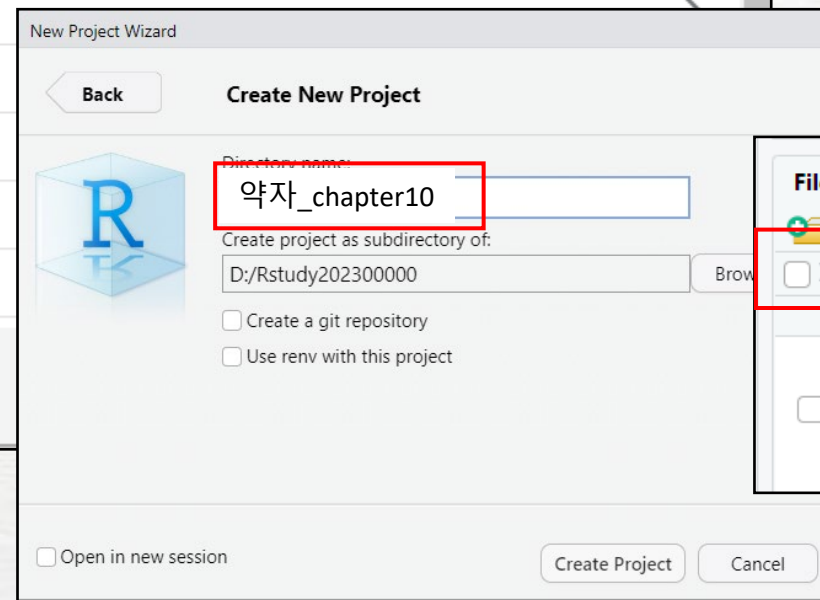
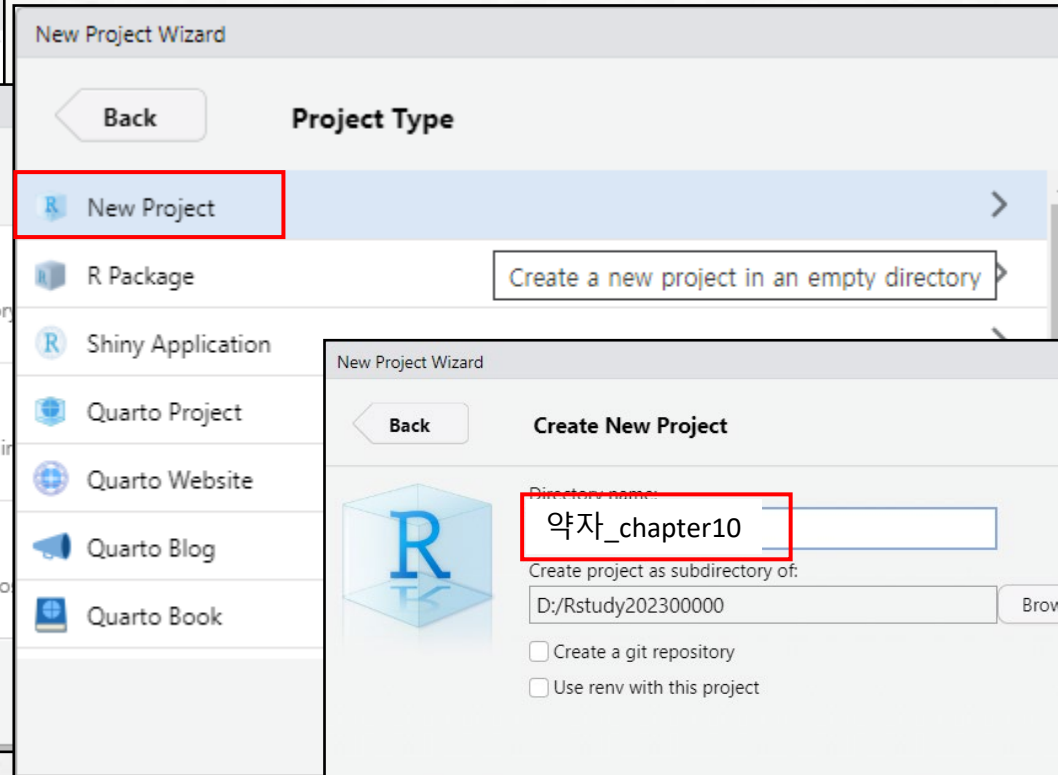
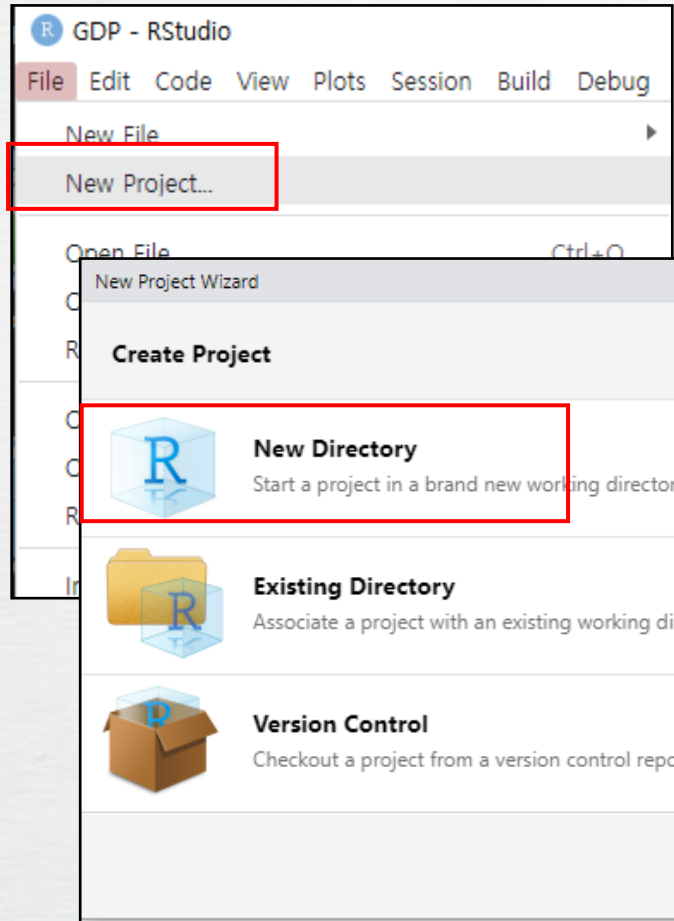
```
> acc <- mean(c_test$breeds == (가) )
```

예측값을 c_test 데이터셋에 pred2열을 만들어서 입력
예측 정확도



III. k-평균 군집

* 프로젝트 시작



1. iris데이터 k-평균 군집화

1.1 데이터 불러온 후 군집화

코드

```
mydata <- iris[,1:4]          # 데이터 준비

fit <- kmeans(x=mydata, centers=3)
fit
fit$cluster                   # 각 데이터에 대한 군집 번호
fit$centers                   # 각 군집의 중심점 좌표
```

- **x=mydata**
매개변수 x에는 군집화의 대상이 되는 데이터셋을 지정한다.
- **centers=3**
매개변수 centers에는 군집의 개수를 입력한다.

1. iris데이터 k-평균 군집화

1.1 데이터 불러온 후 군집화

```
> mydata <- iris[,1:4] # 데이터 준비  
> fit <- kmeans(x=mydata, centers=3)
```

- **x=mydata**
매개변수 x에는 군집화의 대상이 되는 데이터셋을 지정한다.
- **centers=3**
매개변수 centers에는 군집의 개수를 입력한다.

```
> fit  
K-means clustering with 3 clusters of sizes 62, 38, 50
```

Cluster means:

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
1	5.901613	2.748387	4.393548	1.433871
2	6.850000	3.073684	5.742105	2.071053
3	5.006000	3.428000	1.462000	0.246000

1. iris데이터 k-평균 군집화

1.1 데이터 불러온 후 군집화

Clustering vector:

```
[1] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3  
[34] 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 1 1 2 1 1 1 1 1 1 1 1 1  
[67] 1 1 1 1 1 1 1 1 1 1 1 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
[100] 1 2 1 2 2 2 2 1 2 2 2 2 2 2 1 1 2 2 2 2 1 2 1 2 2 1 1 2 2 2  
[133] 2 1 2 2 2 2 1 2 2 2 1 2 2 2 1 2 2 1
```

Within cluster sum of squares by cluster:

```
[1] 39.82097 23.87947 15.15100
(between_SS / total_SS = 88.4 %)
```

Available components:

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

1. iris데이터 k-평균 군집화

1.1 데이터 불러온 후 군집화

[illegible]

그림 12-5 군집화 결과에 대한 주요 정보

[illegible]

1. iris데이터 k-평균 군집화

1.2 차원 축소 후 군집 시각화

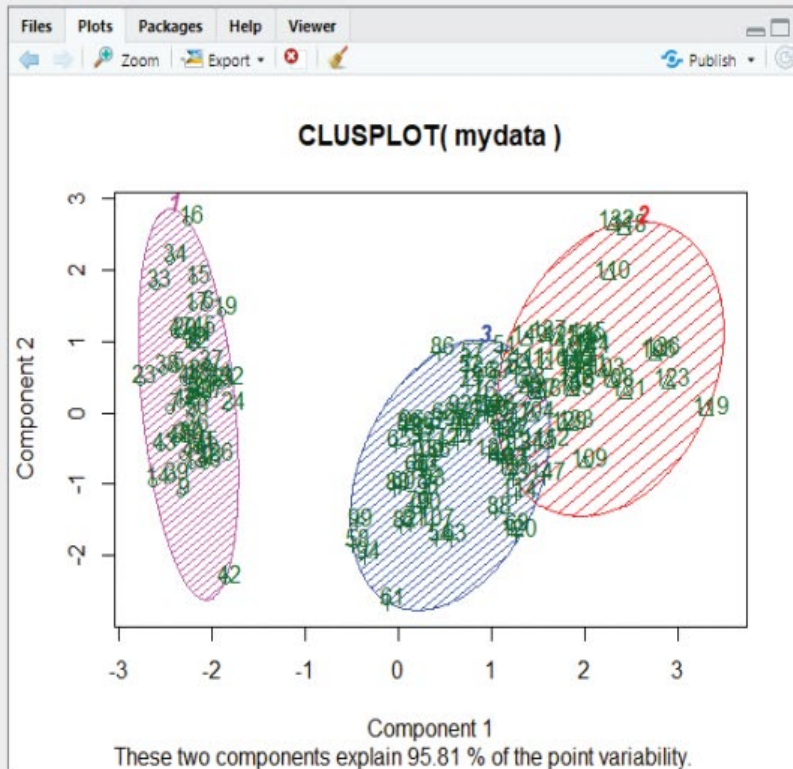
코드

```
# 차원 축소 후 군집 시각화
library(cluster)
clusplot(mydata, fit$cluster, color=TRUE, shade=TRUE, labels=2, lines=0)

# 데이터에서 두 번째 군집의 데이터만 추출
subset(mydata, fit$cluster==2)
```

1. iris데이터 k-평균 군집화

```
> # 차원 축소 후 군집 시각화
> library(cluster)
> clusplot(mydata, fit$cluster, color=TRUE, shade=TRUE,
+         labels=2, lines=0)
>
```



- **mydata**

군집화 대상 데이터셋을 지정한다.

- **fit\$cluster**

군집화 결과에서 관측값별 군집 번호이다.

- **color=TRUE**

군집을 표시하는 원의 색깔을 군집별로 다르게 할지 여부를 결정한다.

- **shade=TRUE**

군집을 표시하는 원 안에 빗금을 표시할지 여부를 결정한다.

- **labels=2**

군집화 대상 데이터셋에서 개별 관측값을 그래프상에 어떻게 나타낼지를 지정한다.

- 1: 관측값을 ○, △, +와 같은 기호로 표시

- 2: 관측값을 숫자 번호로 표시

- **lines=0**

군집의 중심점과 중심점을 연결하는 선을 표시할지 여부를 결정한다.

- 0: 표시하지 않음

- 1: 표시함

1. iris데이터 k-평균 군집화

1.2 차원 축소 후 군집 시각화

```
> # 데이터에서 두 번째 군집의 데이터만 추출
> subset(mydata, fit$cluster==2)
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
51	7.0	3.2	4.7	1.4
52	6.4	3.2	4.5	1.5
54	5.5	2.3	4.0	1.3
55	6.5	2.8	4.6	1.5
56	5.7	2.8	4.5	1.3
57	6.3	3.3	4.7	1.6

...(이하 생략)

1. iris데이터 k-평균 군집화

1.3 대상 데이터 표준화 후 군집화

- 변수 A의 값들을 0~1 사이로 표준화하는 공식

$$(x - \min(A)) / (\max(A) - \min(A))$$

=>x는 변수 A의 임의의 관측값으로, max(A), min(A)는 변수 A의 관측값 중 최댓값과 최솟값

코드

```
std <- function(X) {                # 표준화 함수
  return((X-min(X)) / (max(X)-min(X)))
}

mydata <- apply(iris[,1:4], 2, std) # 표준화된 데이터 준비

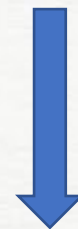
fit <- kmeans(x=mydata, centers=3)
fit
```

1. iris데이터 k-평균 군집화

1.3 대상 데이터 표준화 후 군집화

코드

```
fit$cluster          # 각 데이터에 대한 군집 번호
fit$centers           # 각 군집의 중심점 좌표
# 차원 축소 후 군집 시각화
library(cluster)
clusplot(mydata, fit$cluster, color=TRUE, shade=TRUE, labels=2, lines=0)
```



```
> fit$centers           # 각 군집의 중심점 좌표
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1    5.901613    2.748387    4.393548    1.433871
2    6.850000    3.073684    5.742105    2.071053
3    5.006000    3.428000    1.462000    0.246000
```

```
> fit$centers
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1    0.7072650    0.4508547    0.79704476    0.82478632
2    0.4412568    0.3073770    0.57571548    0.54918033
3    0.1961111    0.5950000    0.07830508    0.06083333
```

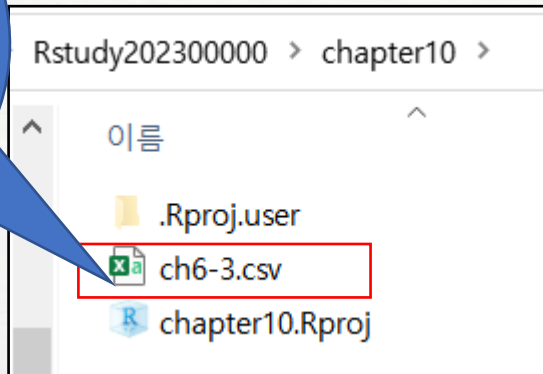
2. 효과적인 사육을 위한 사육환경 분리 기준 탐색을 위한 k-평균 군집

2.1 데이터 불러와서 확인하기

- 분석 목적 : 효과적인 사육을 위한 사육환경 분리 기준 탐색
- ch6-3.csv 데이터 셋의 경우 품종(breeds)이 동일한(a) 닭 100마리의 몸무게(weight)와 하루 평균 사료 섭취량(food) 데이터

LMS에서 다운 받은
[ch6-3.csv]파일을
[약자_chaoter10]폴
더로 이동

코드



ch6-3.csv 데이터 셋

A	B	C
breeds	weight	food
a	2765	217
a	2843	235
a	2678	207
a	2595	204
a	2734	226
a	2616	197
a	2605	216
a	2838	219
a	2900	237

2. 효과적인 사육을 위한 사육환경 분리 기준 탐색을 위한 k-평균 군집

2.1 데이터 불러와서 확인하기

- 분석 목적 : 효과적인 사육을 위한 사육환경 분리 기준 탐색
- ch6-3.csv 데이터 셋의 경우 품종(breeds)이 동일한(a) 닭 100마리의 몸무게(weight)와 하루 평균 사료 섭취량(food) 데이터
- summary() 함수와 산점도를 통해 데이터 분포 확인

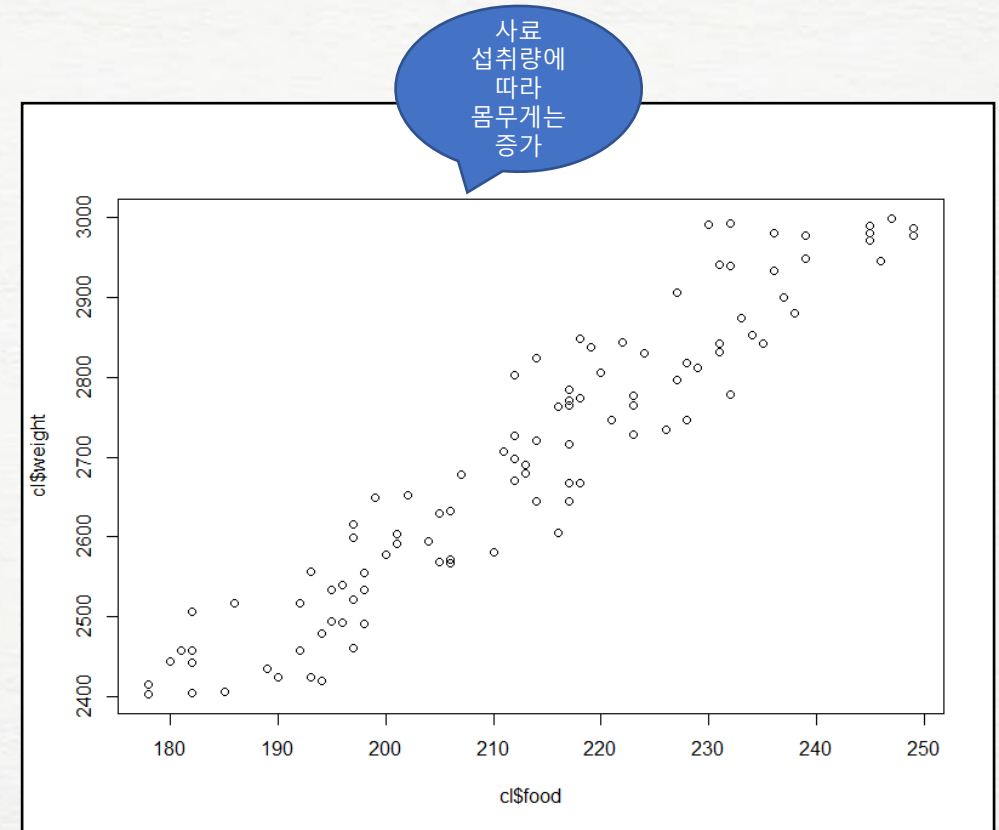
코드

```
> cl <- read.csv("ch6-3.csv", header = TRUE) # 데이터 셋 불러오기
> str(cl) # 데이터 셋 형태 확인

'data.frame': 100 obs. of 3 variables:
 $ breeds: Factor w/ 1 level "a": 1 1 1 1 1 1 1 1 1 1 ...
 $ weight: int 2765 2843 2678 2595 2734 2616 2605 2838 2900 2415 ...
 $ food : int 217 235 207 204 226 197 216 219 237 178 ...

> summary(cl)
breeds weight food
a:100 Min. :2403 Min. :178
      1st Qu.:2551 1st Qu.:197
      Median :2694 Median :214
      Mean :2696 Mean :213
      3rd Qu.:2834 3rd Qu.:228
      Max. :2999 Max. :249

> plot(cl$food, cl$weight) # 산점도, x축 food, y축 weight
```



2. 효과적인 사육을 위한 사육환경 분리 기준 탐색을 위한 k-평균 군집

2.2 k-평균 군집 실시

- kmeans() 함수의 경우 별도 패키지 설치없이 사용 가능하며 kmeans(독립변수, k개수)로 사용

코드

```
> cl_kmc <- kmeans(cl[,2:3], 3) # k-means 군집 실시, k=3  
> cl_kmc # 군집분석 결과 확인
```

K-means clustering with 3 clusters of sizes 37, 29, 34

Cluster means:

```
  weight  food  
1 2503.973 193.7568  
2 2913.414 234.2069  
3 2718.765 215.7353
```

Clustering vector:

```
[1] 3 2 3 1 3 3 1 2 2 1 1 2 1 3 1 3 1 2 1 2 2 3 2 3 2 3 1 3 2 1 2 1 1 3  
[35] 2 2 1 3 2 1 2 2 1 1 1 1 3 3 2 2 1 1 3 3 3 3 3 3 2 3 1 1 3 2 2 1 2 2  
[69] 2 1 1 3 3 1 2 3 1 2 3 1 1 2 1 1 1 1 3 3 3 3 3 1 2 3 3 3 1 1 1 2
```

Within cluster sum of squares by cluster:

```
[1] 160475.8 119763.8 115200.7  
(between_SS / total_SS = 87.5 %)
```

Available components:

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

A	B	C
breeds	weight	food
a	2765	217
a	2843	235
a	2678	207
a	2595	204
a	2734	226
a	2616	197
a	2605	216
a	2838	219
a	2900	237

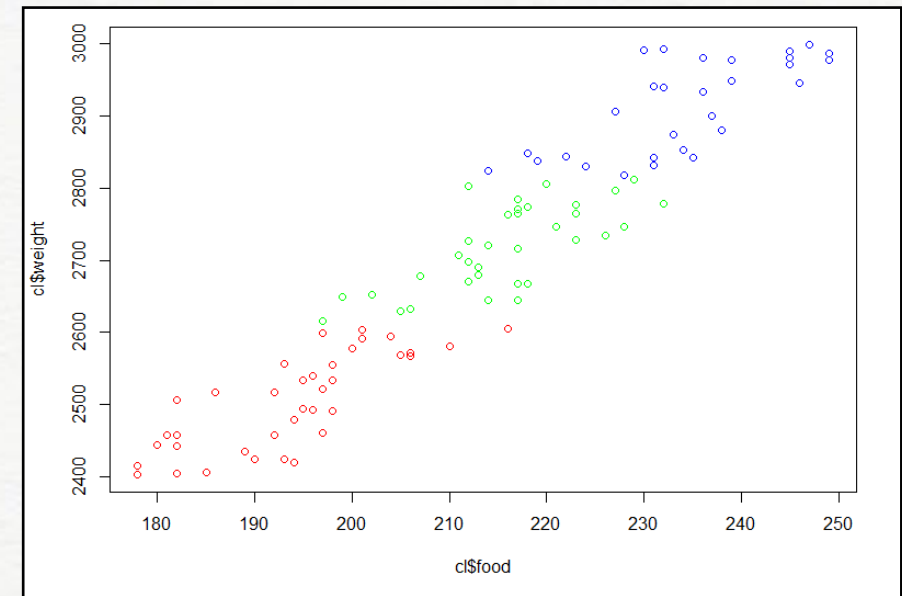
2. 효과적인 사육을 위한 사육환경 분리 기준 탐색을 위한 k-평균 군집

2.3 k-평균 군집 결과 확인

- cluster 객체에 군집 결과가 있으며 이를 기존 데이터 셋에 별도 열을 만들어서 추가 후 산점도로 표시

코드

```
> cl$cluster <- cl_kmc$cluster # 군집결과 기존 데이터 셋에  
입력  
> head(cl) # 데이터 확인  
  breeds weight food cluster  
1     a   2765  217      3  
2     a   2843  235      2  
3     a   2678  207      3  
4     a   2595  204      1  
5     a   2734  226      3  
6     a   2616  197      3  
  
# 산점도를 이용해 군집결과 확인, cluster에 따라 3가지 색상  
부여  
> plot(cl$food, cl$weight, col = c("red", "blue",  
"green")[cl$cluster])
```



∴ 몸무게(weight)와 하루 평균 사료 섭취량(food)에 따라
3개의 군집으로 잘 나뉘었음
(몸무게 2600, 2800g 기준으로 3개의 사육장으로 분리)

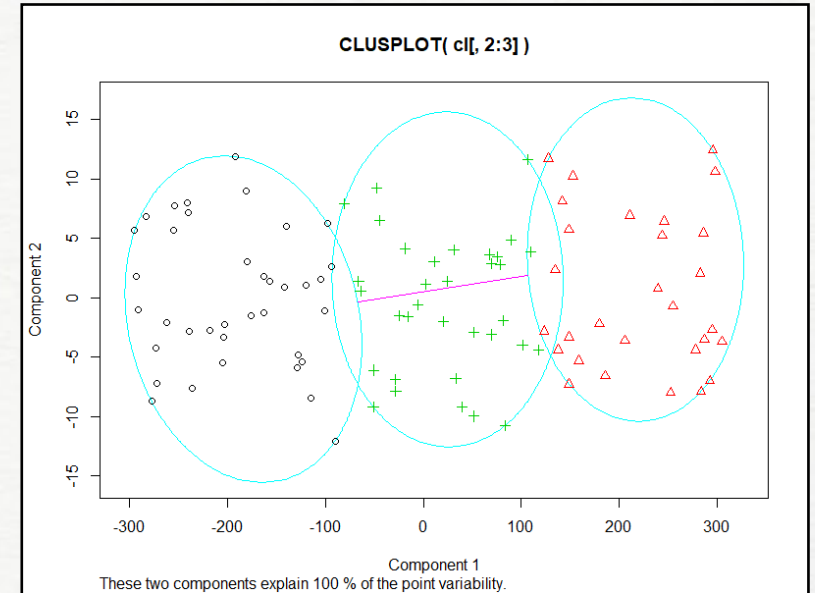
2. 효과적인 사육을 위한 사육환경 분리 기준 탐색을 위한 k-평균 군집

2.3 k-평균 군집 결과 확인

- cluster 패키지의 clusterplot() 함수를 이용할 경우 군집 표시에 더 효과적임

코드

```
> library(cluster)
# clusplot 함수를 이용해 더 보기 쉽게 군집 표현, col.p 옵션을
# 통해 군집에 따른 색상지정
> clusplot(cl[,2:3], cl$cluster, col.p = cl$cluster)
```



➔ 3개의 군집으로 더 보기 편해졌음

IV. k -최근접 이웃 분류

1. k-최근접 이웃 분류를 이용한 꽃의 종 분류

1.1 k-최근접 이웃 분류

코드

```
library(class)

# 훈련용 데이터와 테스트용 데이터 준비
tr.idx <- c(1:25, 51:75, 101:125)      # 훈련용 데이터의 인덱스
ds.tr <- iris[tr.idx, 1:4]             # 훈련용 데이터셋
ds.ts <- iris[-tr.idx, 1:4]            # 테스트용 데이터셋
cl.tr <- factor(iris[tr.idx, 5])        # 훈련용 데이터셋의 그룹(품종) 정보
cl.ts <- factor(iris[-tr.idx, 5])       # 테스트용 데이터셋의 그룹(품종) 정보

pred <- knn(ds.tr, ds.ts, cl.tr, k=3, prob=TRUE)
pred

acc <- mean(pred==cl.ts)               # 예측 정확도
acc

table(pred, cl.ts)                     # 예측값과 실제값 비교 통계
```

- **ds.tr**
훈련용 데이터셋을 지정한다.
- **ds.ts**
테스트용 데이터셋을 지정한다.
- **cl.tr**
훈련용 데이터셋의 그룹 정보를 지정한다.
- **k=3**
최근접 이웃의 개수를 지정한다.
- **prob=TRUE**
예측된 그룹에 대한 지지 확률을 표시할지 여부를 결정한다. 위와 같이 k=3인 경우 최근접 이웃이 Class A가 2개, Class B가 1개라면 예측된 그룹은 Class A이고 Class A에 대한 지지 확률은 $2/3(=0.6666)$ 이다.

1. k-최근접 이웃 분류를 이용한 꽃의 종 분류

1.1 k-최근접 이웃 분류

```
> library(class)
> # 훈련용 데이터와 테스트용 데이터 준비
> tr.idx <- c(1:25,51:75, 101:125)      # 훈련용 데이터의 인덱스
> ds.tr <- iris[tr.idx, 1:4]            # 훈련용 데이터셋
> ds.ts <- iris[-tr.idx, 1:4]           # 테스트용 데이터셋
> cl.tr <- factor(iris[tr.idx, 5])       # 훈련용 데이터셋의 그룹(품종) 정보
> cl.ts <- factor(iris[-tr.idx, 5])     # 테스트용 데이터셋의 그룹(품종) 정보
> pred <- knn(ds.tr, ds.ts, cl.tr, k=3, prob=TRUE)
```

1. k-최근접 이웃 분류를 이용한 꽃의 종 분류

1.1 k-최근접 이웃 분류

```
> pred
```

```
[1] setosa      setosa      setosa      setosa      setosa      setosa
```

```
[7] setosa      setosa      setosa      setosa      setosa      setosa
```

```
[13] setosa      setosa      setosa      setosa      setosa      setosa
```

```
... (중간 생략)
```

```
[61] virginica   virginica   virginica   versicolor virginica   virginica
```

```
[67] virginica   virginica   virginica   virginica   virginica   virginica
```

```
[73] virginica   virginica   virginica
```

```
attr(,"prob")
```

```
[1] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
```

```
[7] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
```

```
[13] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
```

```
... (중간 생략)
```

```
[67] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 0.6666667
```

```
[73] 1.0000000 1.0000000 0.6666667
```

```
Levels: setosa versicolor virginica
```


1. k-최근접 이웃 분류를 이용한 꽃의 종 분류

1.2 k-최근접 이웃 분류 성능 평가

코드

```
acc <- mean(pred==cl.ts)      # 예측 정확도
acc

table(pred,cl.ts)             # 예측값과 실제값 비교 통계
```

```
> acc <- mean(pred==cl.ts)    # 예측 정확도
> acc
[1] 0.9333333
> table(pred,cl.ts)           # 예측값과 실제값 비교 통계
      cl.ts
pred    setosa versicolor virginica
setosa     25         0         0
versicolor  0         23         3
virginica   0          2        22
```

1. k-최근접 이웃 분류를 이용한 꽃의 종 분류

1.3 k-최근접 이웃 분류 궁금한 점

(1) k 값은 어떻게 정하는가?

데이터셋의 관측값의 개수(행 의 수)가 100이라면 k는 10보다 작은 것이 좋음

보통은 1~7의 값을 차례로 실험해보고, 예측모델의 정확도 가 가장 높게 나오는 k를 선택

(2) 그룹을 예측할 때 다수결에서 동수가 나오면 어떻게 하는가?

액티언`knn()` 함수에서는 둘 중 하나를 임의로 선택

(3) k-최근접 이웃 알고리즘의 단점은 무엇인가?

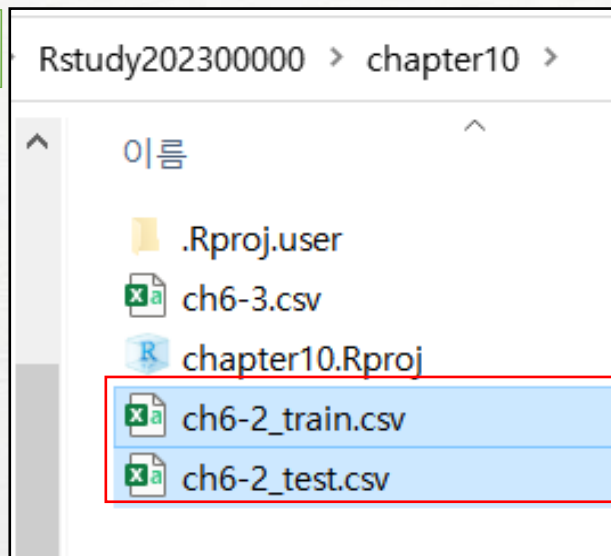
k개의 최근접 이웃을 알아내기 위해서는 그룹을 모르는 데이터 P와 그룹 정보가 알려진
훈련용 데이터셋의 모든 데이터들과 거리 계산해야 함

2. k-최근접 이웃 분류를 이용한 병아리 품종 분류

2.1 데이터 준비

- [ch6-2_train.csv], [ch6-2_test.csv]의 파일을 [약자_chapter10]의 폴더로 이동

코드



- ch6-2_test.csv, ch6-2_train.csv 데이터 셋
- 날개길이(wing_length), 꼬리길이(tail_length), 볏의 길이(comb_height), 품종(breeds)

ch6-2_train.csv				
	A			D
1	wing_length	tail_length	comb_height	breeds
2	238	63	34	a
3	236	67	30	a
4	256	65	34	a
5	240	63	35	a
6	246	65	30	a
7	233	65	30	a
8	235	66	30	a
9	241	66	35	a

ch6-2_test.csv				
	A			D
1	wing_length	tail_length	comb_height	breeds
2	238	63	34	a
3	236	67	30	a
4	256	65	34	a
5	240	63	35	a
6	246	65	30	a
7	233	65	30	a
8	235	66	30	a
9	241	66	35	a

2. k-최근접 이웃 분류를 이용한 병아리 품종 분류

2.2 k-최근접 이웃 실시(k=3)

- DMwR2 패키지의 kNN() 함수 사용

코드

- K-NN의 경우 훈련과 테스트가 한번에 이뤄짐(게으른 학습)

```
> install.packages("DMwR2") # k-NN 수행을 위한 패키지 설치
> library(DMwR2)

> c_train <- read.csv("ch6-2_train.csv", header = TRUE) # 훈련용 데이터셋 불러오기
> c_test <- read.csv("ch6-2_test.csv", header = TRUE) # 테스트용 데이터셋 불러오기

# 병아리 품종을 종속변수로 두고 나머지 변수 전체를 독립변수 설정, k = 3으로 해봄
> c_knn3 <- kNN(breeds ~ wing_length+ tail_length+ comb_height, c_train, c_test, k = 3)
> c_test$pred3 <- c_knn3 # 예측값을 c_test 데이터셋에 pred3열을 만들어서 입력
> head(c_test) # 데이터 확인
  wing_length tail_length comb_height breeds pred3
1         258          67          32      a     a
2         260          64          34      a     a
3         251          63          31      a     a
4         248          63          30      a     a
5         254          62          32      a     a
6         230          64          33      a     a
```


2. k-최근접 이웃 분류를 이용한 병아리 품종 분류

2.3 k-최근접 이웃 분류 성능 평가(k=3)

코드

```
acc<- mean(c_test$breeds== c_test$pred3)  
acc
```

예측 정확도

```
table(c_test$breeds,c_test$pred3)
```

예측값과 실제값 비교 통계

```
> acc<- mean(c_test$breeds==c_test$pred3)  
> acc  
[1] 0.9333333  
> table(c_test$breeds,c_test$pred3)
```

	a	b	c
a	19	1	0
b	1	18	1
c	0	1	19

2. k-최근접 이웃 분류를 이용한 병아리 품종 분류

2.4 k-최근접 이웃 실시(k=5)

k=3과 k=5가 어떤 차이가 있는가를 실습

코드

```
> c_train <- read.csv("ch6-2_train.csv", header = TRUE) # 훈련용 데이터셋 불러오기
> c_test <- read.csv("ch6-2_test.csv", header = TRUE) # 테스트용 데이터셋 불러오기

# 병아리 품종을 종속변수로 두고 나머지 변수 전체를 독립변수 설정, k = 5으로 해봄
> c_knn5 <- kNN(breeds ~ wing_length+ tail_length+ comb_height, c_train, c_test, k = 5)
> c_test$pred5 <- c_knn5 # 예측값을 c_test 데이터셋에 pred3열을 만들어서 입력
> head(c_test) # 데이터 확인
```

	wing_length	tail_length	comb_height	breeds	pred5
1	258	67	32	a	a
2	260	64	34	a	a
3	251	63	31	a	a
4	248	63	30	a	a
5	254	62	32	a	a
6	230	64	33	a	a

2. k-최근접 이웃 분류를 이용한 병아리 품종 분류

2.4 k-최근접 이웃 분류 성능 평가(k=5)

코드 `acc<- mean(c_test$breeds== c_test$pred5)`
`acc`

예측 정확도

`table(c_test$breeds,c_test$pred5)`

예측값과 실제값 비교 통계

k=5인 경우
분류
정확도가
k=3인
경우보다
높음

```
> acc<- mean(c_test$breeds==c_test$pred3)
> acc
```

```
[1] 0.9333333
```

```
> table(c_test$breeds,c_test$pred3)
```

	a	b	c
a	19	1	0
b	1	18	1
c	0	1	19

```
> acc<- mean(c_test$breeds== c_test$pred5)
> acc
```

```
[1] 0.95
```

```
>
> table(c_test$breeds,c_test$pred5)
```

	a	b	c
a	20	0	0
b	1	18	1
c	0	1	19

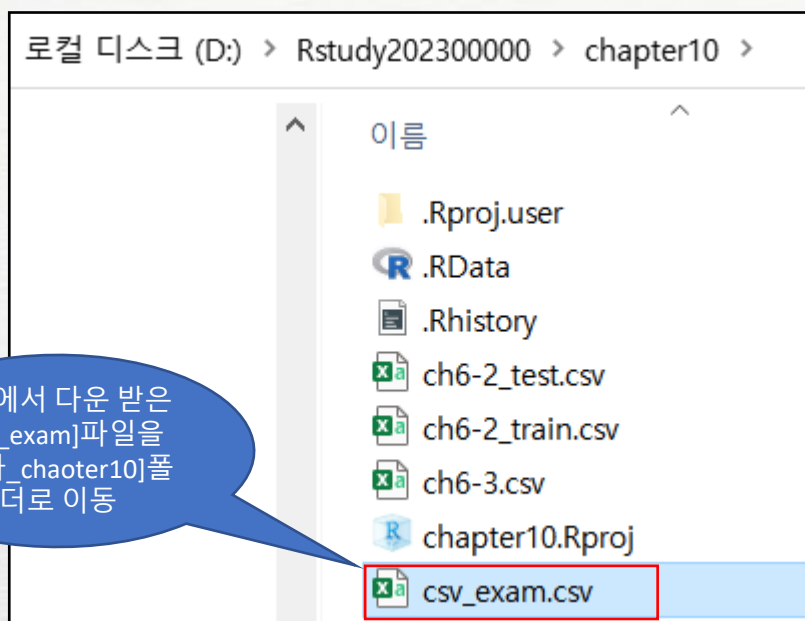
V. 응용1

1. 유사한 점수분포의 학생들을 군집하기 위한 k-평균 군집

1.1 데이터 불러와서 확인하기

- 분석 목적 : 과목들의 점수를 이용하여 유사한 사용자를 군집
- [csv_exam.csv] 데이터 셋의 [math, English, science] 변수 사용

[csv_exam.csv] 데이터 셋



	A	B	C	D	E
	id	class	math	english	science
	1	1	50	98	50
	2	1	60	97	60
	3	1	45	86	78
	4	1	30	98	58
	5	2	25	80	65
	6	2	50	89	98
	7	2	80	90	45
	8	2	90	78	25

1. 유사한 점수분포의 학생들을 군집하기 위한 k-평균 군집

1.2 데이터 호출&k-평균 군집 실시

응용1_(가),(나)를 채우시오.

코드 > cl약자 <- read.csv("csv_exam.csv", header = TRUE) # 데이터 셋 불러오기

코드 > 약자K <- kmeans((가) ,3) #3개의 과목에 대해 k-means 군집 실시, k=3, 해당열을 [3:5]열임을 주의
> 약자K
> 약자K\$cluster # 각 학생의 군집번호
[1] 3 3 1 3 3 1 2 2 3 3 1 3 3 3 1 3 1 1 1 2
> (나) # 약자 K의 중심점

	math	english	science
1	67.00000	72.85714	84.85714
2	82.66667	83.66667	42.66667
3	43.20000	93.70000	46.70000

1. 유사한 점수분포의 학생들을 군집하기 위한 k-평균 군집

1.3 k-평균 군집 결과 확인

응용2_(가)를 채우시오.

코드

```
> cl약자$cluster <- (가)  
> head(cl약자)
```

```
# (약자K)의 군집번호를 기존 데이터 셋에 추가  
# 데이터 확인
```

```
> head(cl)  
  id class math english science cluster  
1  1     1  50     98     50         3  
2  2     1  60     97     60         3  
3  3     1  45     86     78         1  
4  4     1  30     98     58         3  
5  5     2  25     80     65         3  
6  6     2  50     89     98         1
```

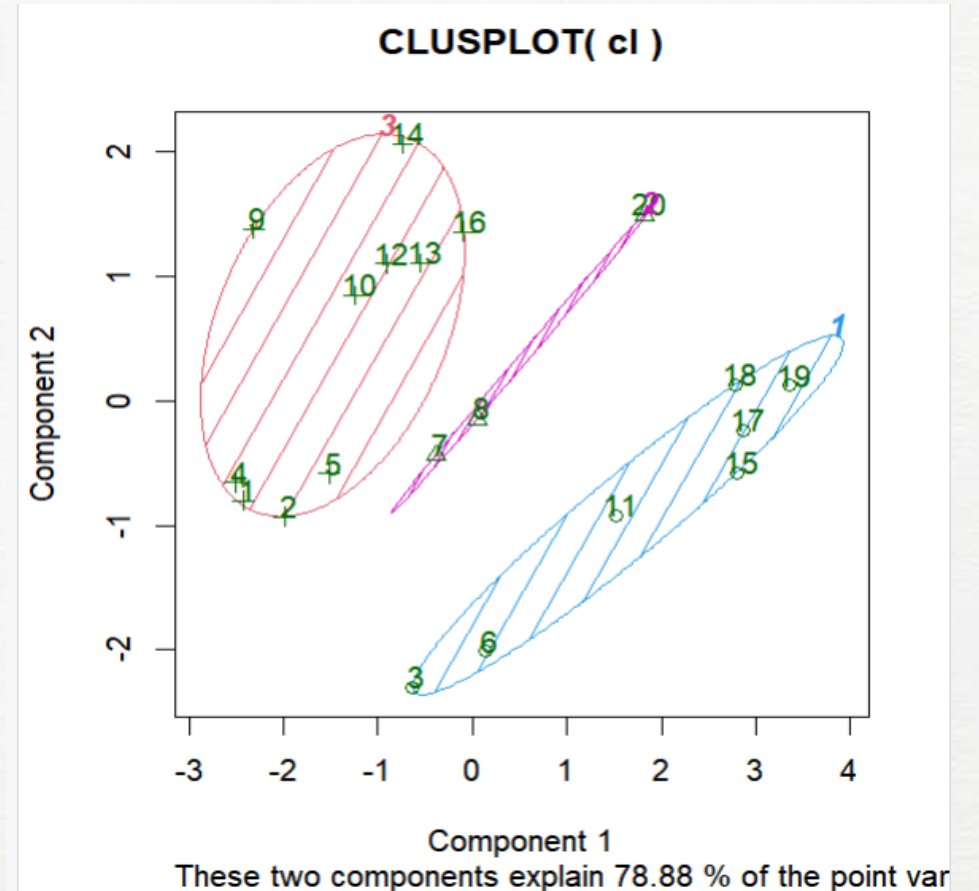
1. 유사한 점수분포의 학생들을 군집하기 위한 k-평균 군집

1.3 k-평균 군집 결과 확인

응용3_(가) 를 채우시오. 그래프도 올리시오.

코드

```
> library(cluster)
# clusplot 함수를 이용해 더 보기 쉽게 군집 표현(shade), 무늬가 있음을 나타냄
> clusplot(ci약자, ci약자$cluster, color=TRUE, (가), labels=2, lines=0)
```



➔ 3개의 군집으로 더 보기 편해졌음

2. k-최근접 이웃 분류를 응용4_(가),(나)를 채우시오. 그리고 예측정확도를 캡처해서 올리시오.

코드 * k-최근접 이웃 분류

```
library(class)
# [cl약자]훈련용 데이터와 테스트용 데이터 준비
tr.idx <- c(가) ) # 1:15행까지 훈련용 데이터의 인덱스
ds.tr <- cl약자[tr.idx, 3:5] # 훈련용 데이터셋(math, English, science)
ds.ts <- (나) # - 사용테스트용 데이터셋 (math, English, science)
cl.tr <- factor(cl약자[tr.idx, 6]) # 훈련용 데이터셋의 그룹 정보
cl.ts <- cl약자[-tr.idx, 6] # 테스트용 데이터셋의 그룹 정보
```

```
pred약자 <- knn(ds.tr, ds.ts, cl.tr, k=3, prob=TRUE) #k-최근접 이웃 분류
pred약자 #16번부터 20번까지의 군집 번호
```

```
[1] 3 1 1 1 1
Acc <- mean(pred약자==cl.ts) # 예측 정확도를 구함
```

```
Acc # 예측 정확도는 0.8
```

```
[1] 0.8
```

```
table(pred약자,cl.ts) # 예측값과 실제값 비교 통계
```

```
cl.ts
Pred 1 2 3
1 3 1 0
2 0 0 0
3 0 0 1
```

[cl약자]=>학생들이 속한 군집 번호 추가됨

	id	class	math	english	science	cluster
1	1	1	50	98	50	3
2	2	1	60	97	60	3
3	3	1	45	86	78	1
4	4	1	30	98	58	3
5	5	2	25	80	65	3
6	6	2	50	89	98	1
7	7	2	80	90	45	2
8	8	2	90	78	25	2
9	9	3	20	98	15	3
10	10	3	50	98	45	3
11	11	3	65	65	65	1
12	12	3	45	85	32	3
13	13	4	46	98	65	3
14	14	4	48	87	12	3
15	15	4	75	56	78	1
16	16	4	58	98	65	3
17	17	5	65	68	98	1
18	18	5	80	78	90	1
19	19	5	89	68	87	1
20	20	5	78	83	58	2

VI. 응용2

1. 친구를 찾기 위한 k-평균 군집

1.1 데이터 불러와서 확인하기

- 분석 목적 : 학생들의 취향을 기준으로 유사한 학생들을 군집
- [2023_favorite.csv] 데이터 셋의 [Hamburger Pizza Cat Dog Summer] 변수 사용

코드

« 로컬 디스크 (D:) > Rstudy202300000 > chapter10

이름

.Rproj.user

.RData

.Rhistory

2023_favorite.csv

ch6-2_test.csv

ch6-2_train.csv

ch6-3.csv

chapter10.Rproj

csv_exam.csv

완성된
[2023_favorite.
csv]파일을
[약자_chaoter1
0]폴더로 이동

[2023_favorite.csv] 데이터 셋

문항번호	문항					
1	햄버거를 얼마나 좋아하나요?(0~10)					
2	피자를 얼마나 좋아하나요?(0~10)					
3	고양이를 얼마나 좋아하나요?(0~10)					
4	강아지를 얼마나 좋아하나요?(0~10)					
5	여름을 얼마나 좋아하나요?(0~10)					
	학번	문항1	문항2	문항3	문항4	문항5
No	ClassNum	Hamburger	Pizza	Cat	Dog	Summer
1	202210101	10	2	1	10	3
2	202010101	8	2	3	2	1
3	201939393	4	2	1	3	3
4	201839393	5	1	10	2	10
5	202130303	2	1	3	2	1
6	202010101	8	2	3	2	1
7						
8						

반전체
학생의
데이터를
입력

1. 친구를 찾기 위한 k-평균 군집

응용5_(가),(나)를 채우시오.

1.2 데이터 호출&k-평균 군집 실시

코드

```
> favoritecsv약자 <- read.csv(" 2023_favorite.csv", header = TRUE)
```

데이터 셋 불러오기

코드

```
> 약자F <- kmeans(favoritecsv약자[ , (가) ], 3 )
```

3열에서 7열, k=3, 콤마주의

```
> 약자F
```

```
> (나) #약자F의 cluster번호
```

```
[1] 3 3 1 3 3 1 2 2 3 3 1 3 3 3 1 3 1 1 1 2
```

```
> 약자F$center
```

#약자F의 중심점

	Hamburger	Pizza	Cat	Dog	Summer
1	8.705882	2.000000	2.294118	4.823529	1.705882
2	5.000000	1.000000	10.000000	2.000000	10.000000
3	2.833333	1.416667	2.166667	2.416667	1.833333

No	ClassNum	Hamburger	Pizza	Cat	Dog	Summer
1	202210101	10	2	1	10	3
2	202010101	8	2	3	2	1
3	201939393	4	2	1	3	3
4	201839393	5	1	10	2	10

데이터가
다르므로
다른
결과가
나올 수
있음

1. 친구를 찾기 위한 k-평균 군집

1.3 k-평균 군집 결과 확인

코드

> favoritecsv약자\$cluster<- (가) # 군집결과 약자F의 cluster번호를 기존 데이터 셋에 입력
> head(favoritecsv약자) # 데이터 확인

데이터가
다르므로
다른
결과가
나올 수
있음

```
> head(favoritecsv약자)
```

	No	ClassNum	Hamburger	Pizza	Cat	Dog	Summer	cluster
1	1	202210101	10	2	1	10	3	1
2	2	202010101	8	2	3	2	1	1
3	3	201939393	4	2	1	3	3	3
4	4	201839393	5	1	10	2	10	2
5	5	202130303	2	1	3	2	1	3
6	6	202010101	8	2	3	2	1	1

1. 친구를 찾기 위한 k-평균 군집

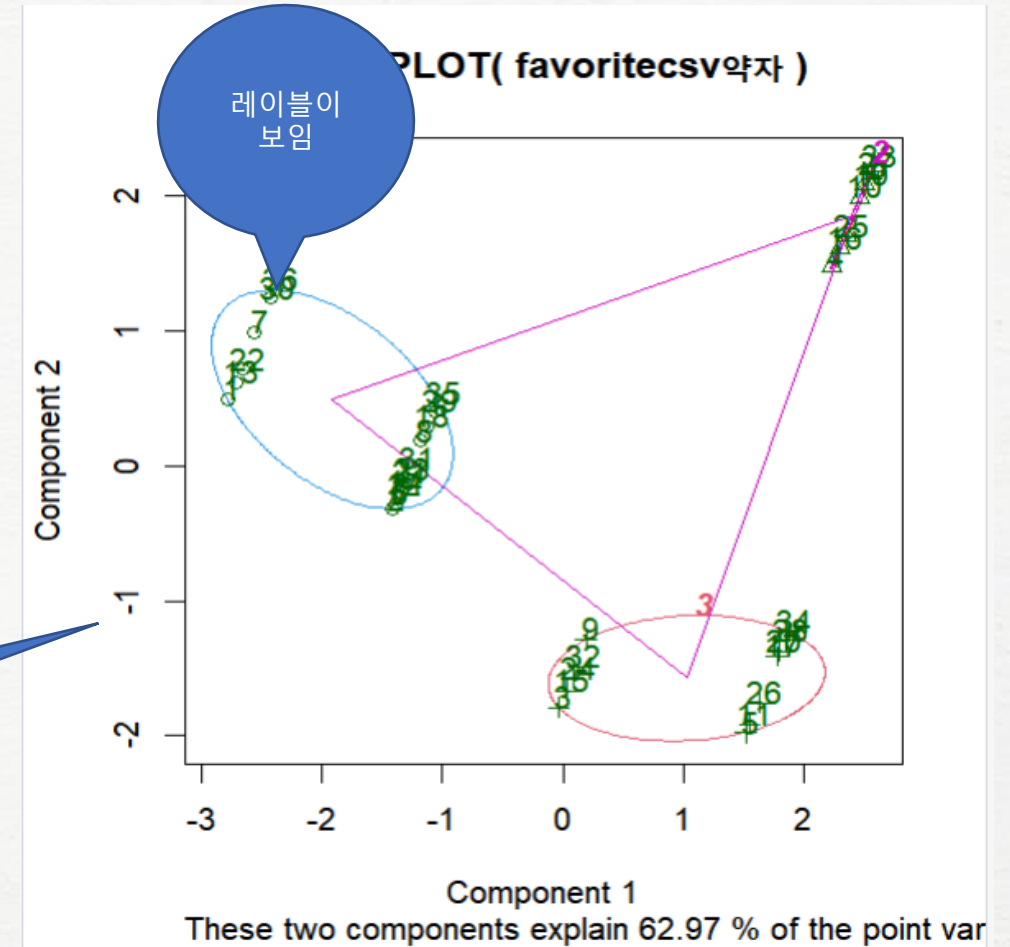
응용7_(가)를 채우시오.
그래프를 올리시오.

1.3 k-평균 군집 결과 확인

코드

```
> library(cluster)
#레이블이 보이도록 함
> clusplot(favoritecsv약자, favoritecsv약자$cluster, color=TRUE,
shade=FALSE, (가), lines=1 )
```

자신이 속한
군집의 번호
확인



오늘도 잘했어요 🍷