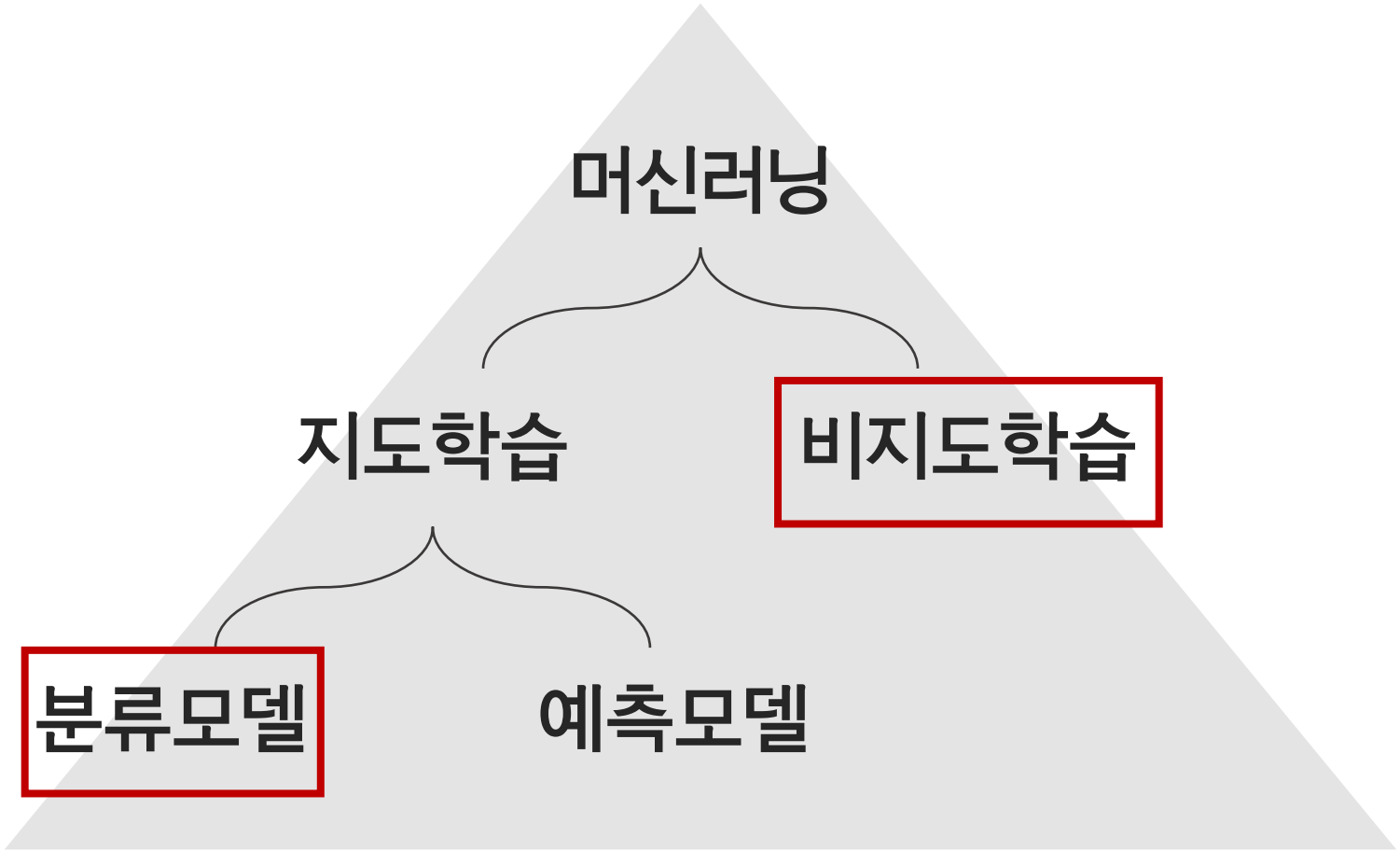
KNN / KMEANS

비어플 4기 이용헌

목차


머신러닝
지도학습
비지도학습
분류모델
예측모델

지도학습 – 분류모델, 예측모델

▶ 정답이 있는 데이터로 학습을 해서 정답이 없는 데이터의 정답을 분류, 예측하는 학습 과정

분류 : y가 명목형 / 예측 : y가 연속형

비지도학습

▶ 정답이 없는 데이터의 특징을 찾아 비슷한 특징을 가진 데이터끼리 묶는 학습 과정

Classification(분류)

데이터의 유사성이나 공통 기준에 따라 범주로 분류하는 방법이다.

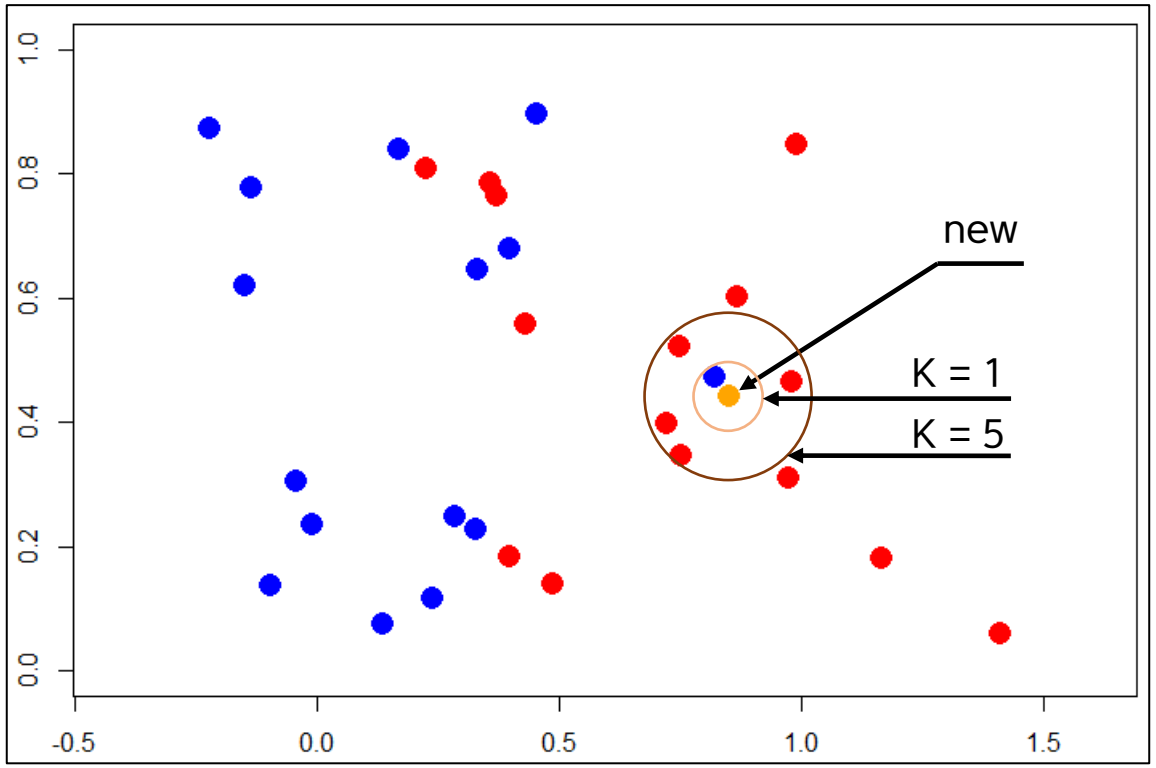
〈 지도학습 〉

K-Nearest Neighbor Algorithm

새로운 데이터가 주어졌을 때, 기존 데이터 가운데 가장 가까운 k개의 이웃 데이터 정보로 새로운 데이터를 예측하는 알고리즘

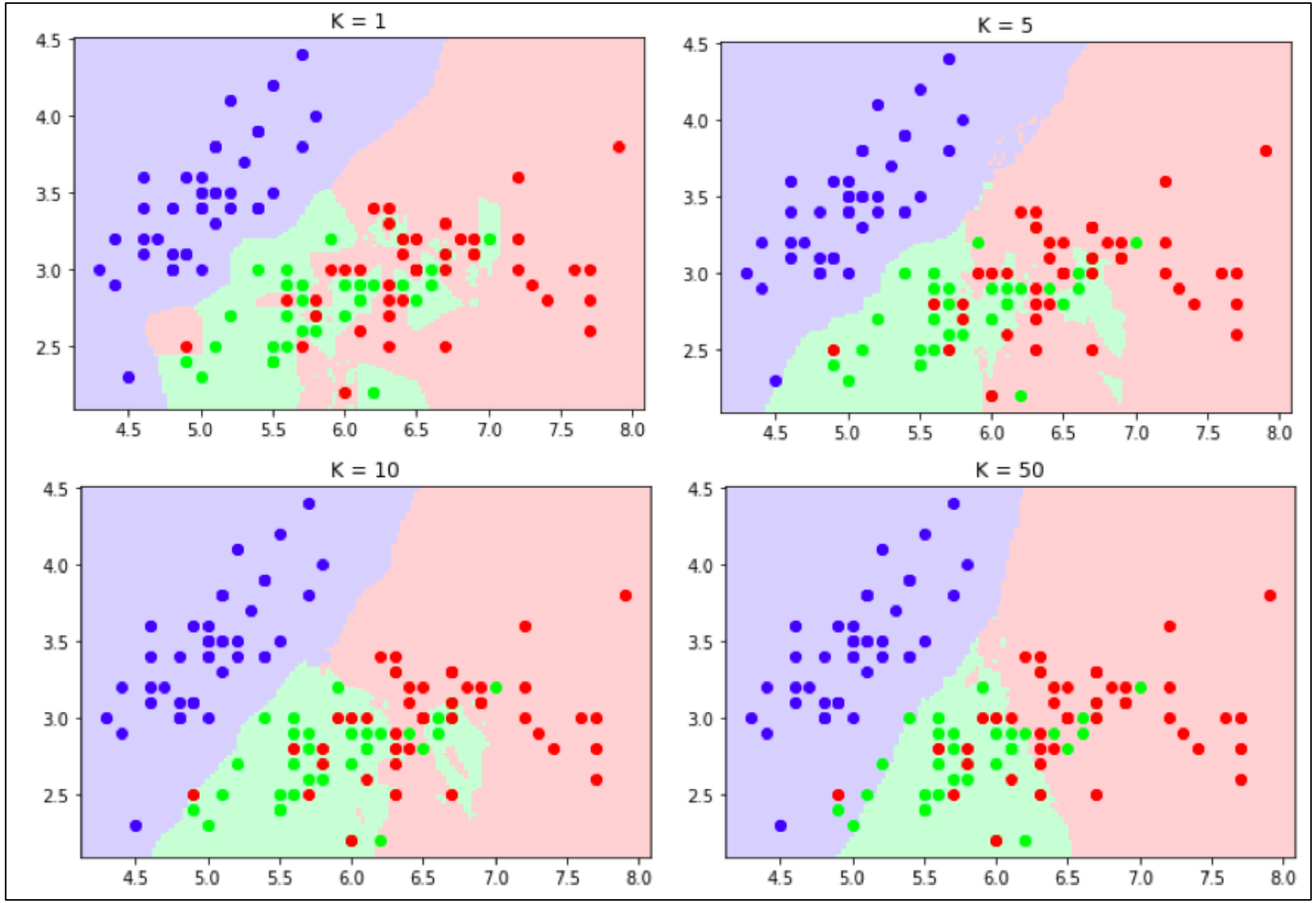
프로세스

1. 얼마나 많은 수의 주변 값을 포함할지 결정(K 결정)
2. 새로운 값인 테스트 셋과 기존 트레인 셋 간 각 obs 거리 계산
3. K 값을 기준으로 테스트 셋 예측 값 결정



K = 1 : 파란색으로 분류

K = 5 : 빨간색으로 분류



<출처> Medium.com : Day 3 — K-Nearest Neighbors and Bias–Variance Tradeoff

K가 지나치게 작아지면, overfitting

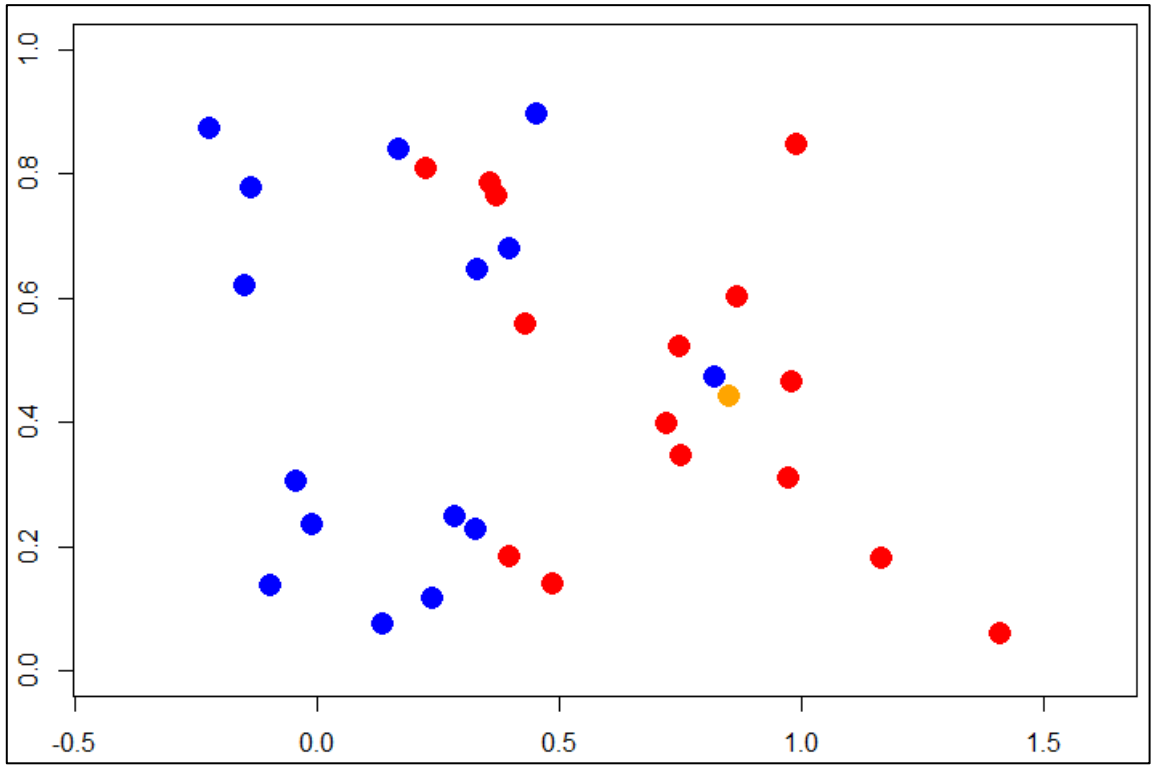
K가 지나치게 커지면, underfitting

Lazy Algorithm

▶ 우리가 흔히 아는 회귀 등의 모델은 모형의 weight parameters를 데이터를 통해 학습한다.

▶ 하지만 학습 단계에서 knn은 데이터를 그저 기억할 뿐이다. 덕분에 knn은 학습 단계에서 자원 소모를 적게 한다.

▶ 하지만 predict 단계에서 꽤나 비용이 많이 든다. 모든 train 데이터와 비교를 하고 있기 때문이다.



거리 지표

$X = (x_1, x_2, \dots, x_n)$

$Y = (y_1, y_2, \dots, y_n)$

1. Euclidean Distance

$$d(X,Y) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + \cdots + (x_n - y_n)^2}$$

2. Manhattan Distance

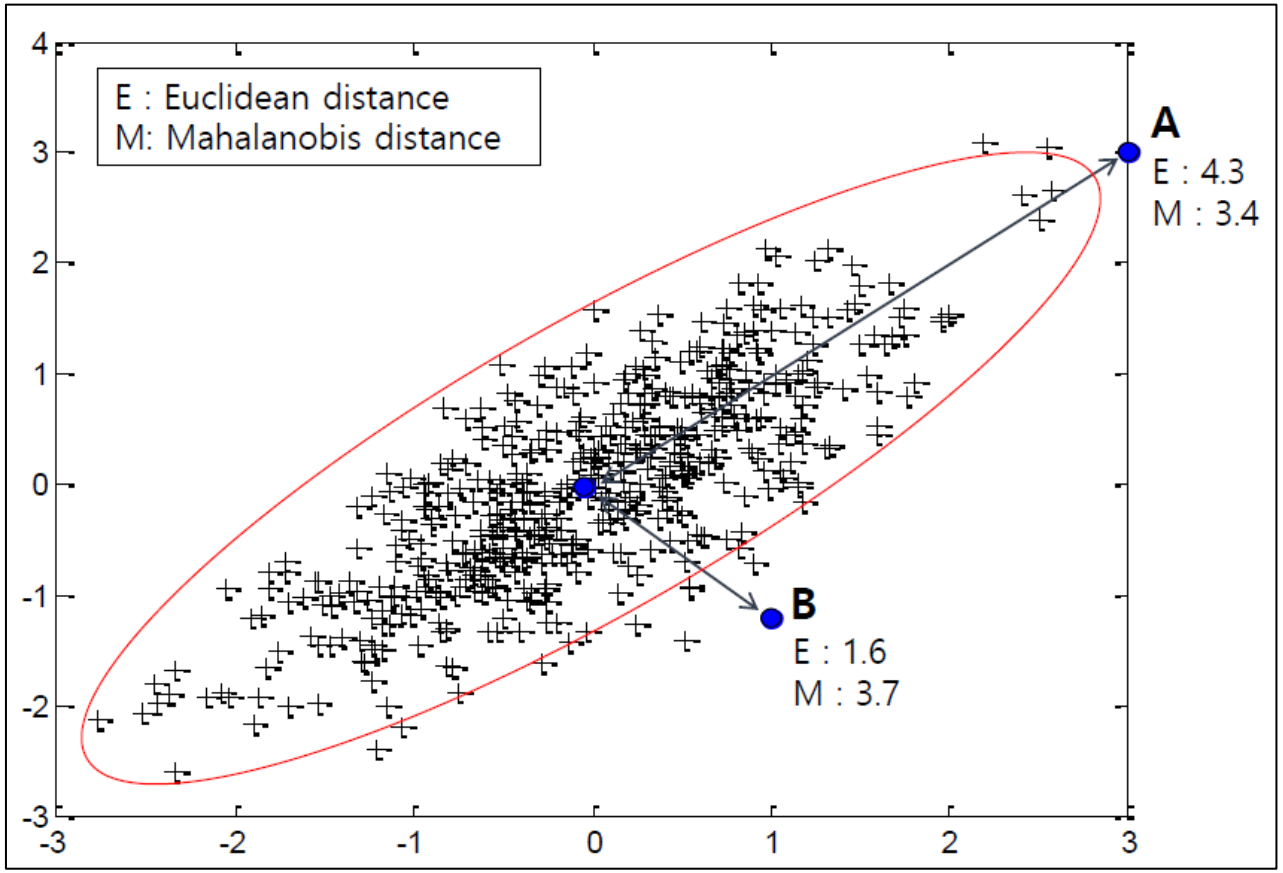
$$d_{\text{Manhattan}}(X,Y) = \sum_{i} |x_i - y_i|$$

<출처> Quora.com : What is the difference between Manhattan and Euclidean distance measures?

3. Mahalanobis Distance

$$d_{Mahalanobis}(X,Y) = \sqrt{(X-Y)^T \sum^{-1} (X-Y)}$$

$\sum^{-1}$ = *inverse of covariance matrix*



<출처>https://imgur.com/ernTml0

유의점

1. 정규화

도시	인구(명)	미세먼지농도 (㎍/㎥)
서울	1000만	200
시애틀	67만	40

유의점

2. 수치화

One hot Encoding

Obs	주량(병)	성별
1	2	남
2	2	여
3	1	남
4	1	여
5	0	여
6	1	여
7	1	남

Obs	주량(병)	남	여
1	2	1	0
2	2	0	1
3	1	1	0
4	1	0	1
5	0	0	1
6	1	0	1
7	1	1	0

Example code

```
############## KNN ##############
rm(list=ls())
library(class)
data(iris)

# train, test split
set.seed(2020)
allrows <- 1:nrow(iris)
trainrows <- sample(allrows, replace = F, size = 0.6 * length(allrows))
train_iris <- iris[trainrows, 1:4]
train_target <- iris[trainrows, 5]

test_iris <- iris[-trainrows, 1:4]
test_target <- iris[-trainrows, 5]
table(test_target)
`
# modeling
result_knn <- knn(train = train_iris, test = test_iris, cl = train_target, k = 3)

# result
table(test_target, result_knn)
```

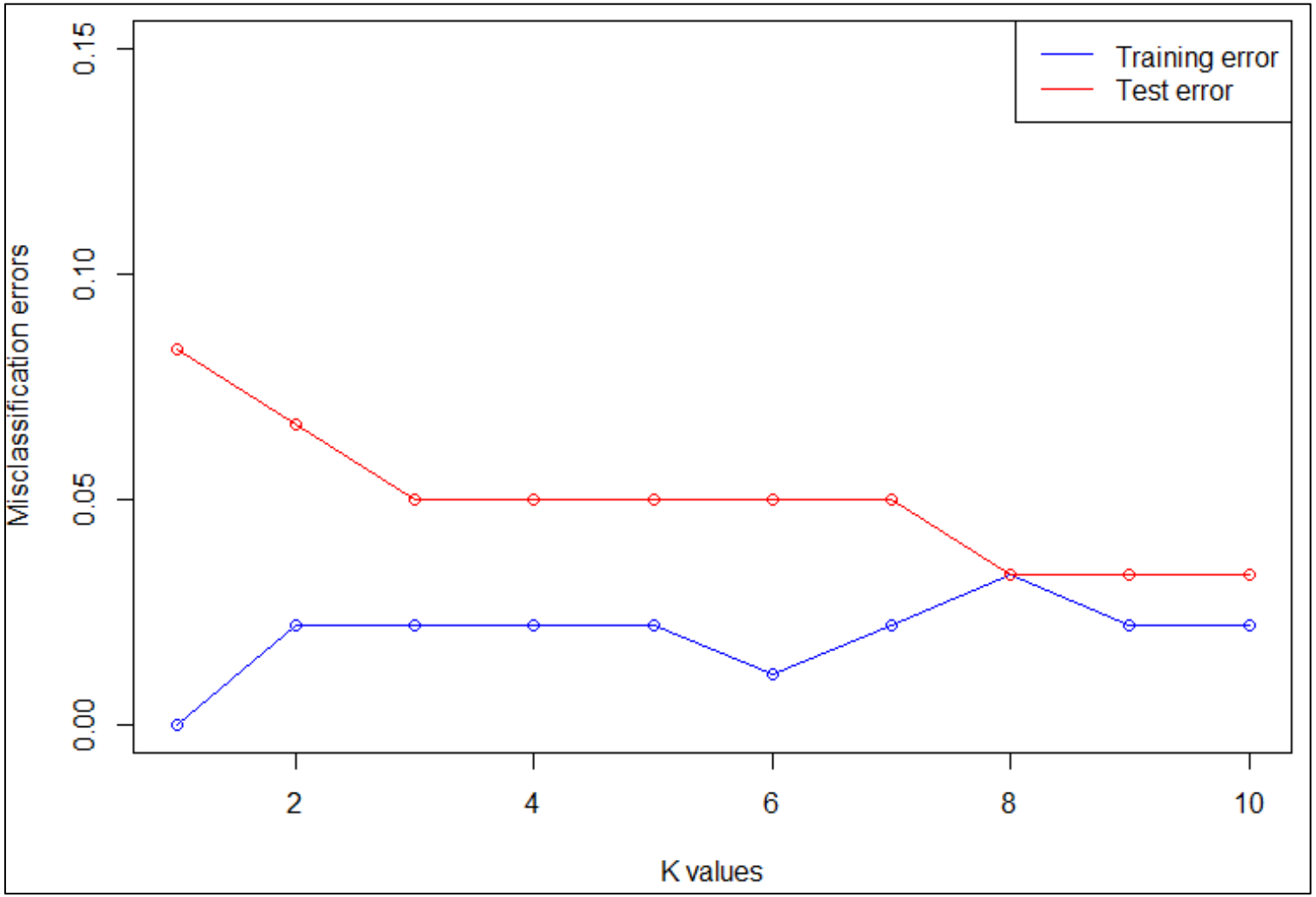
```
> table(test_target, result_knn)
             result_knn
test_target   setosa versicolor virginica
  setosa          18          0         0
  versicolor       0         22         2
  virginica        0          1        17
```

Best K 찾기 : 오분류가 가장 줄어드는 부분을 K로 결정한다.

Code

```
# find best k
# overfitting을 확인하기 위해 train set 예측력도 확인
error.train <- rep(0, 10)
error.test <- rep(0, 10)

for(k in 1:10) {
  pred_iris <- knn(train = train_iris, test = train_iris, cl = train_target, k)
  error.train[k] <- 1 - mean(pred_iris == train_target)
  pred_iris <- knn(train = train_iris, test = test_iris, cl = train_target, k)
  error.test[k] <- 1 - mean(pred_iris == test_target)
}

plot(error.train, type = "o", ylim = c(0,0.15), col = "blue", xlab = "K values", ylab = "Misclassification errors")
lines(error.test, type = "o", col="red")
legend("topright", legend = c("Training error","Test error"), col = c("blue","red"), lty = 1:1)
```

Best K 찾기 : 오분류가 가장 줄어드는 부분을 K로 결정한다.



Clustering(군집화)

동일한 그룹(cluster)의 개체가 다른 그룹의 개체보다 더 비슷하도록 개체 집합을 그룹화 하는 방법이다.

패턴 인식, 이미지 분석, 정보 검색, 데이터 압축 등 많은 분야에서 사용되는 통계 데이터 분석 기법이다.

〈 비지도학습 〉

K-Means

크리스토퍼 비숍의 패턴 인식과 머신 러닝을 일부 참조하였습니다.

데이터가 주어졌을 때, K개의 cluster로 비슷한 특성을 지닌 obs끼리 묶어주는 알고리즘

※ Knn과 아무 관련 없음.

프로세스

1. 몇 개의 cluster로 묶어줄지 결정(K 결정)
2. 초기 Centroid 선택
3. Centroid를 기준으로 거리가 가장 가까운 데이터끼리 clustering
4. 만들어진 cluster의 중심을 새로운 Centroid로 3번 계속 반복
5. Centroid가 완전히 수렴 또는 미리 설정한 오차 내로 움직이면 중지

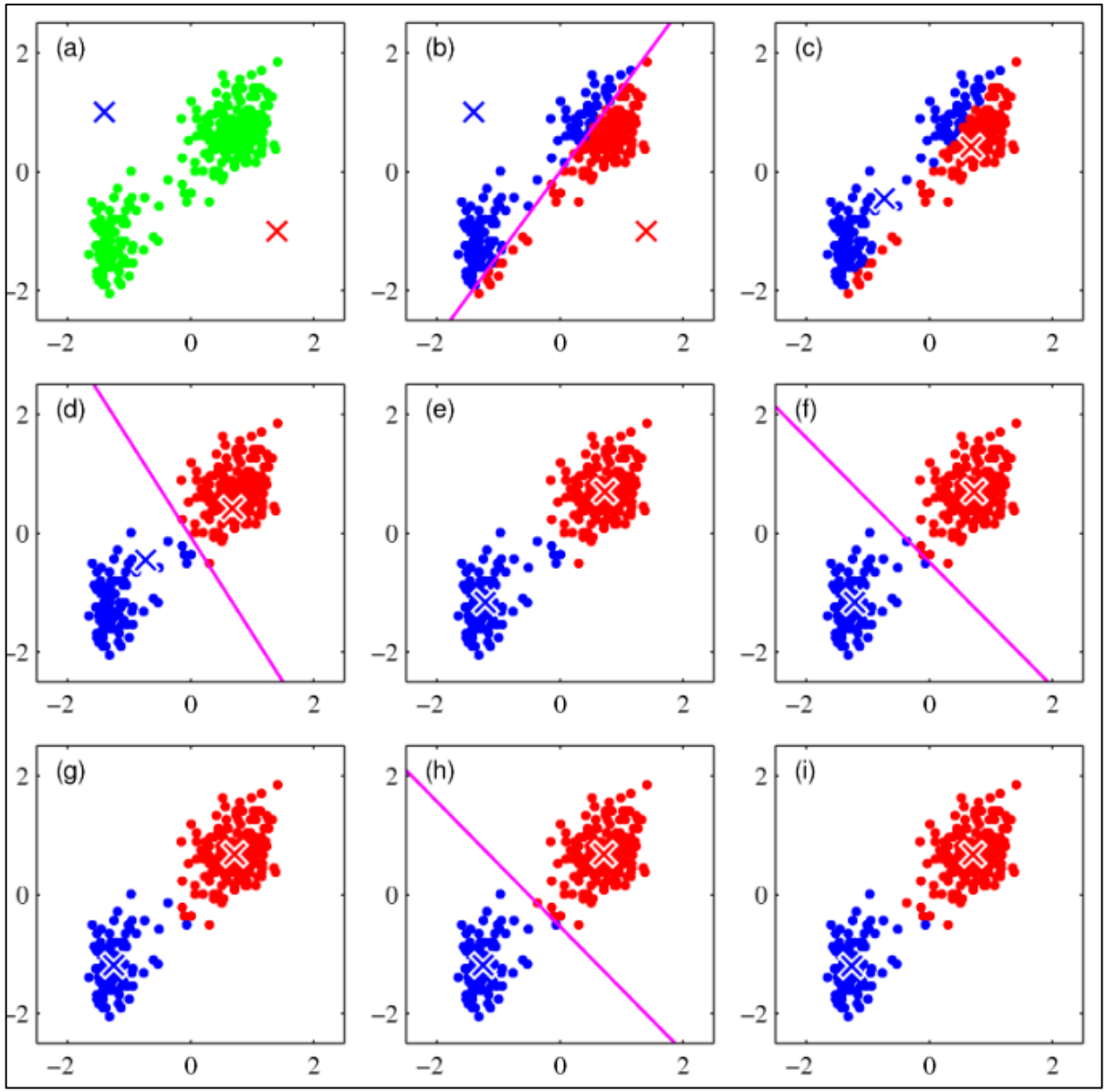
K-Means

▶ EM 알고리즘을 기반으로 작동

▶ E : Expectation / M : Maximization

▶ 각 군집 중심의 위치 / 각 객체가 어떤 군집에 속해야 하는지

▶ 위 두 가지를 계속 반복해야 하기 때문에 EM 알고리즘을 적용한다.



참조 : https://m.blog.naver.com/PostView.nhn?blogId=kangdonghyun&logNo=221017901062&proxyReferer=https:%2F%2Fwww.google.co.kr%2F

수리적 풀이

$$J = \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}\|x_n - \mu_k\|^2$$

$$r_{nk} = \begin{cases} 1, if\ k = argmin_j\|x_n - \mu_j\|^2 \\ 0, \qquad otherwise \end{cases}$$

$$2\sum_{n=1}^{N} r_{nk}(x_n - \mu_k) = 0$$

$$\mu_k = \frac{\sum_n r_{nk}x_n}{\sum_n r_{nk}}$$

x_n : 데이터 집합, K : 클러스터 수, N : 총 데이터 수,
μ_k : k번째 클러스터의 중심

▶ J를 왜곡 측정 함수라 한다.

▶ 같은 클러스터에 속하는 각각의 점들로부터 그 클러스터의 평균과의 거리의 합을 제곱한 함수다.

▶ $\|x_n - \mu_j\|^2$식이 최소값이 되는 j번째 클러스터에서만 r_{nk}의 값이 1이 되고 나머지는 0이다.

▶ k를 고정한 상태에서 왜곡 측정 함수 J는 μ_k에 대해 이차 형식으로 미분을 통해 최소가 되는 지점인 μ_k 를 찾을 수 있다.

유의점

1. 거리 계산을 할 수 있는 자료 타입

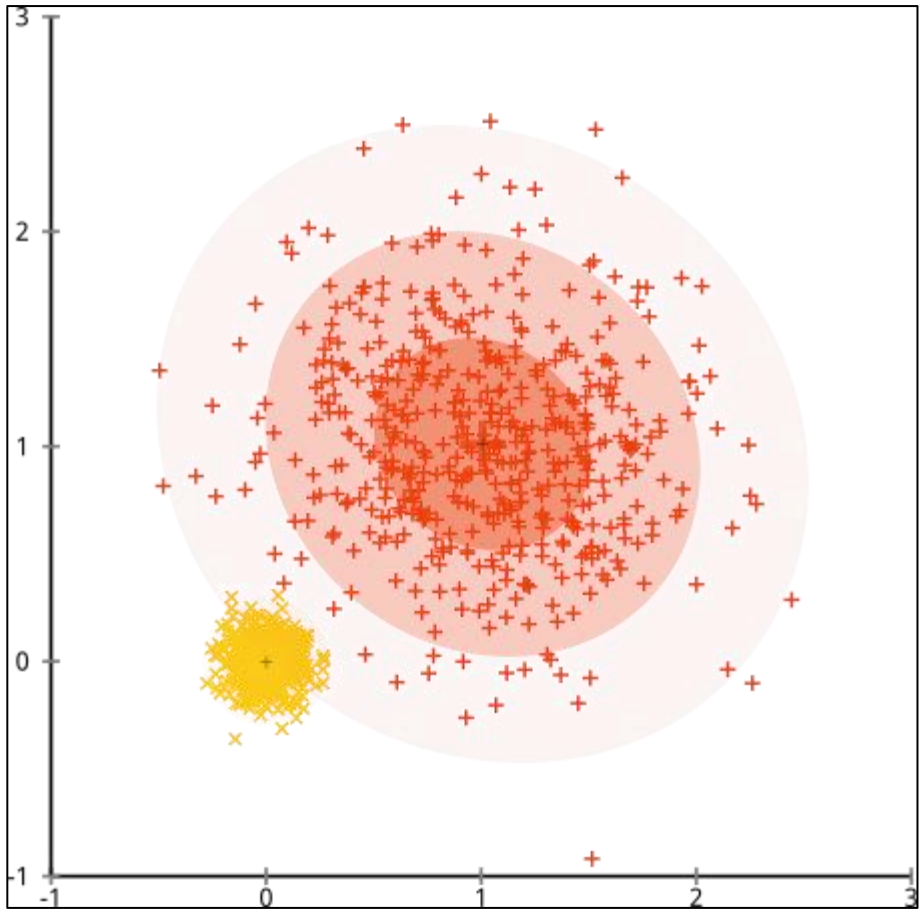
K-means 알고리즘은 샘플과 평균 간의 거리 측정 방식을 사용하기 때문에
수치형 자료 타입만을 적용시킬 수 있다.

2. 이상치에 영향을 많이 받는 알고리즘

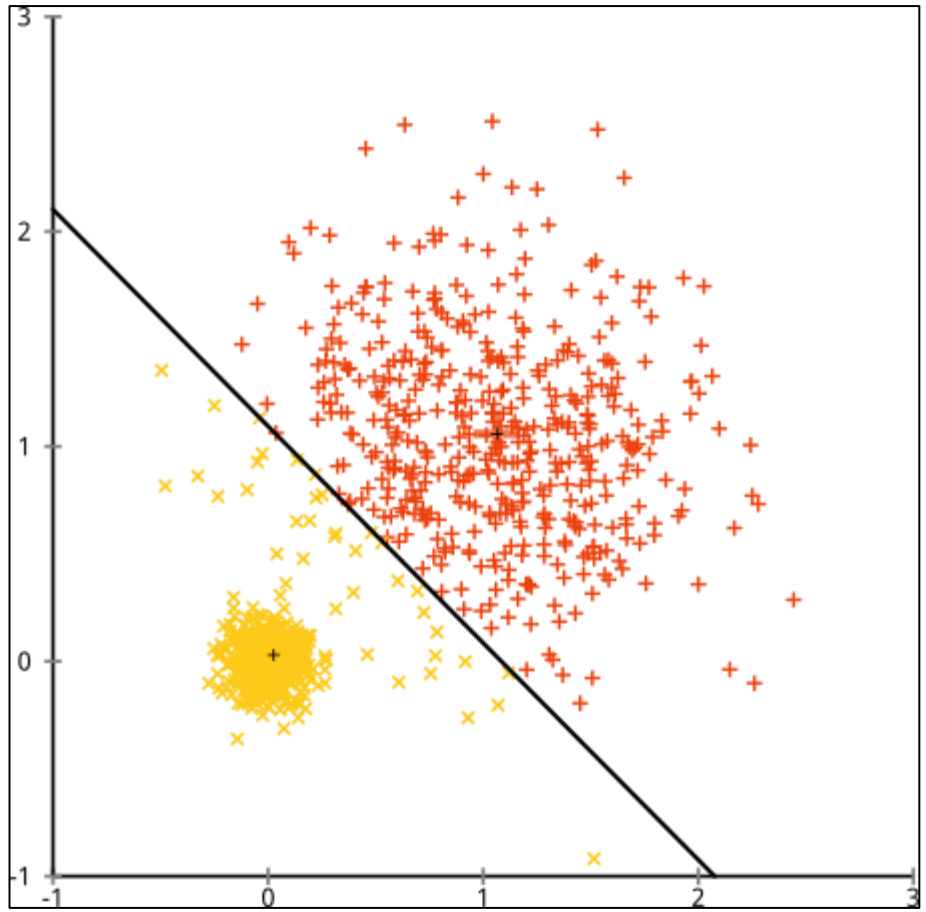
이상치에 민감한 평균을 사용한다.

3. Local Optimum

초기값 설정이 잘못될 경우 데이터의 특성을 반영하지 못하고 전혀 다른 곳에
고착화된 Local Optimum이 발생할 수 있다.

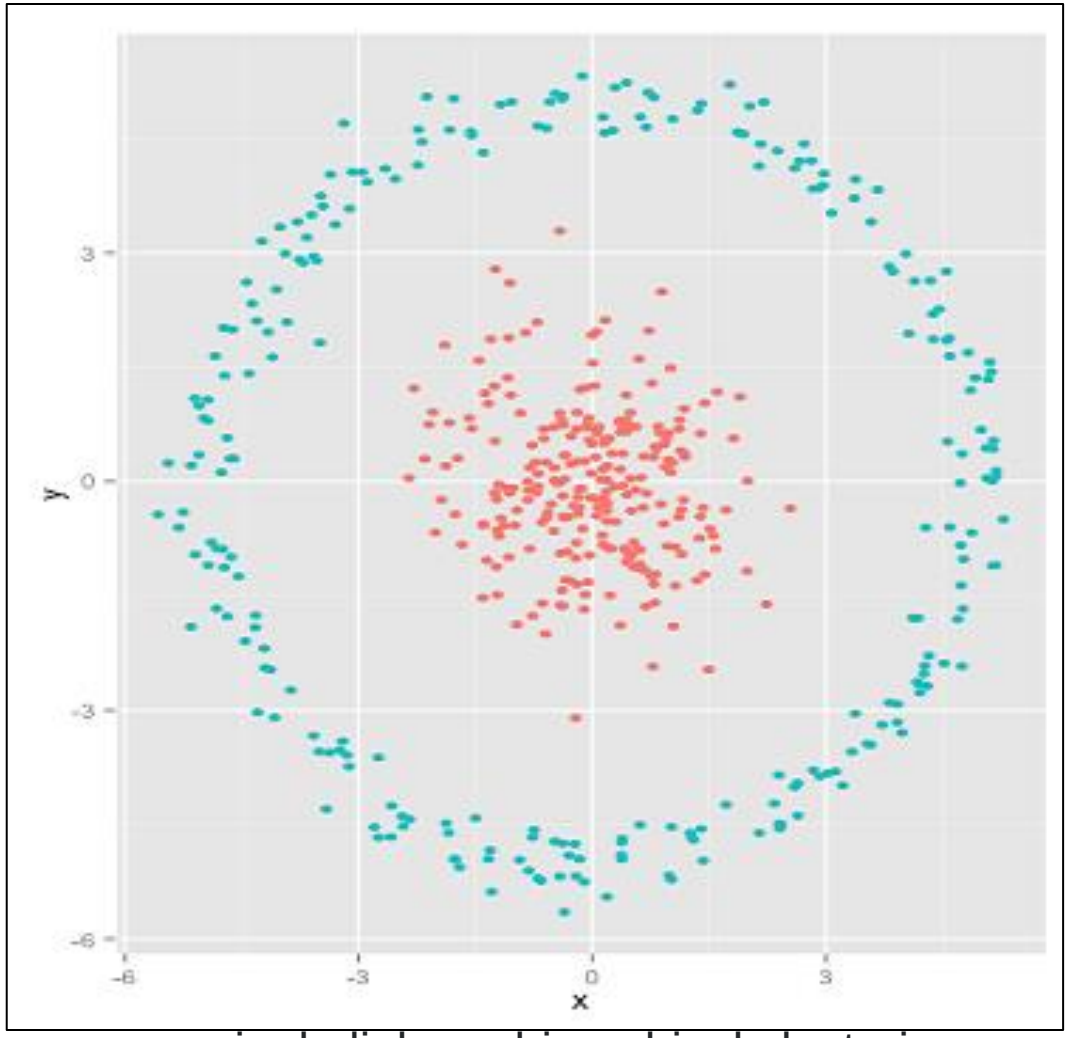


Gaussian Mixture Model



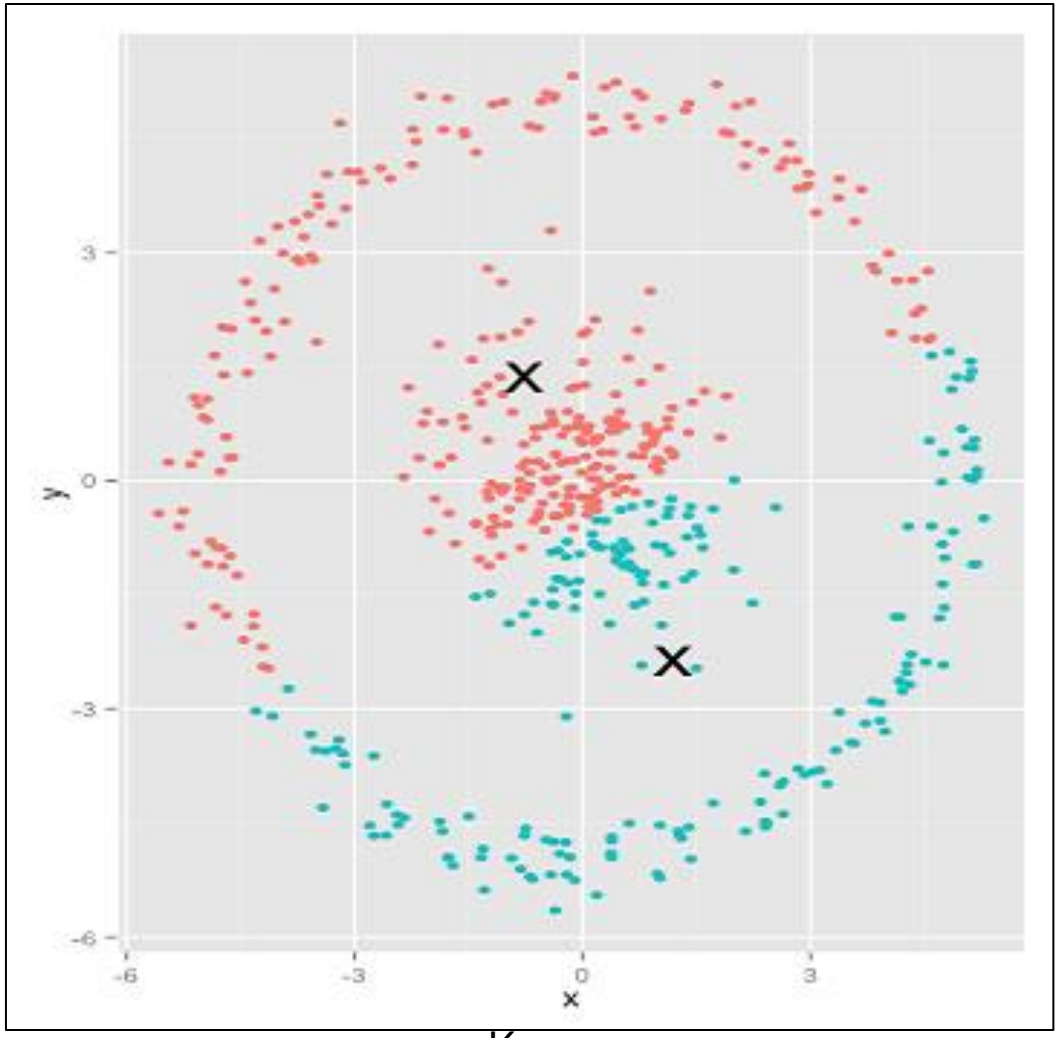
K-means

참조 : https://stats.stackexchange.com/questions/133656/how-to-understand-the-drawbacks-of-k-means



single linkage hierachical clustering

A.k.a hclust



K-means

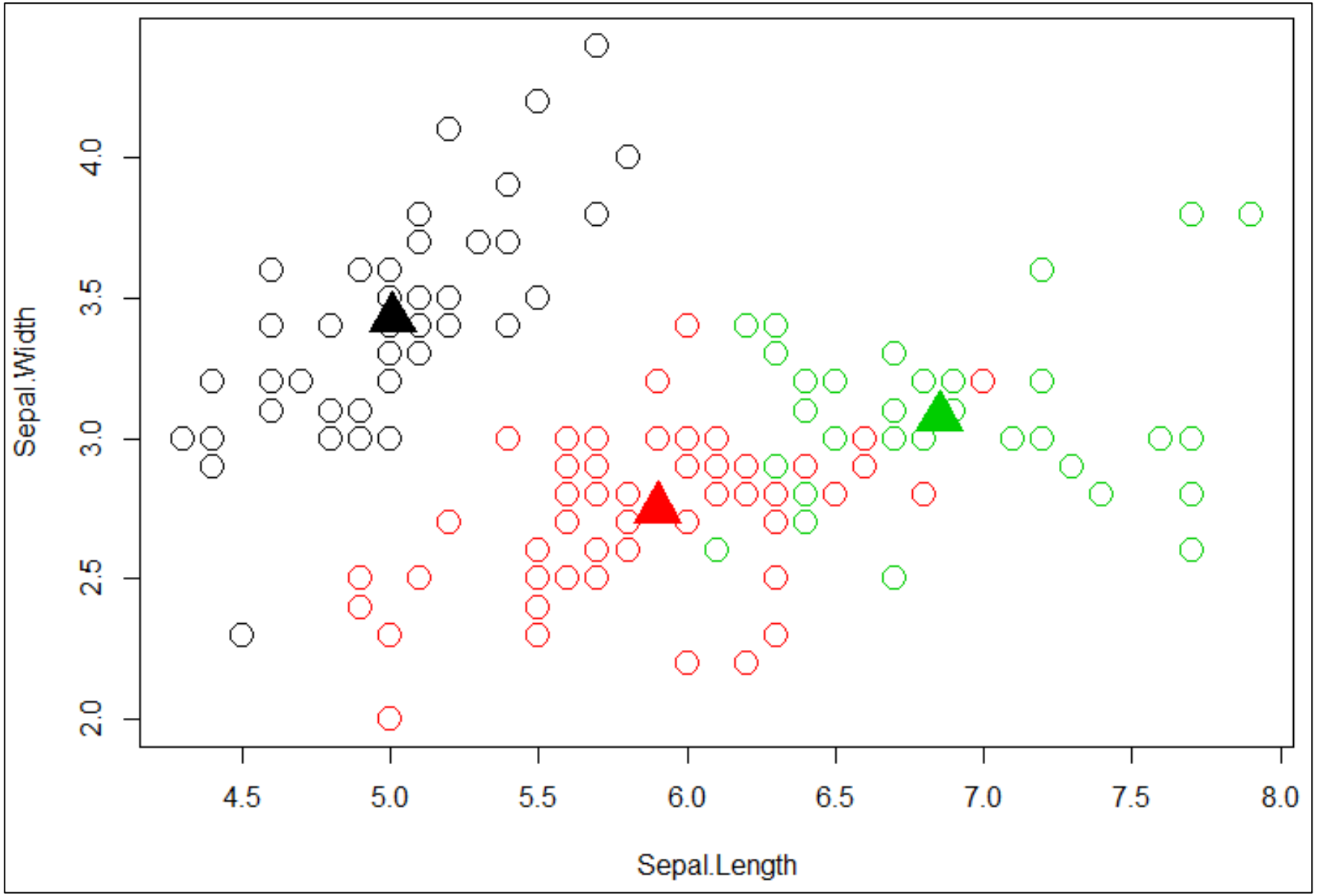
참조 : https://stats.stackexchange.com/questions/133656/how-to-understand-the-drawbacks-of-k-means

K-Means

Example code

```
############## Kmeans ##############
rm(list=ls())
# dataset 객체 할당 및 target 변수 분리
data(iris)
data_iris <- iris
data_iris <- data_iris[, -5]
target <- iris$Species

# n_cluster = 3으로 설정
set.seed(1234)
model_km <- kmeans(data_iris, 3, nstart = 5)
model_km

table(target, model_km$cluster)

plot(data_iris[, c("Sepal.Length", "Sepal.Width")], col = model_km$cluster, cex = 2, pch = 1)
points(model_km$centers[, c("Sepal.Length", "Sepal.Width")], col = 1:3, cex = 3, pch = 17)
```

```
> table(target, model_km$cluster)

target        1  2  3
  setosa     50  0  0
  versicolor  0 48  2
  virginica   0 14 36
```


Sepal.Width
4.0
3.5
3.0
2.5
2.0
4.5
5.0
5.5
6.0
6.5
7.0
7.5
8.0
Sepal.Length

Best K 찾기

1. elbow-method

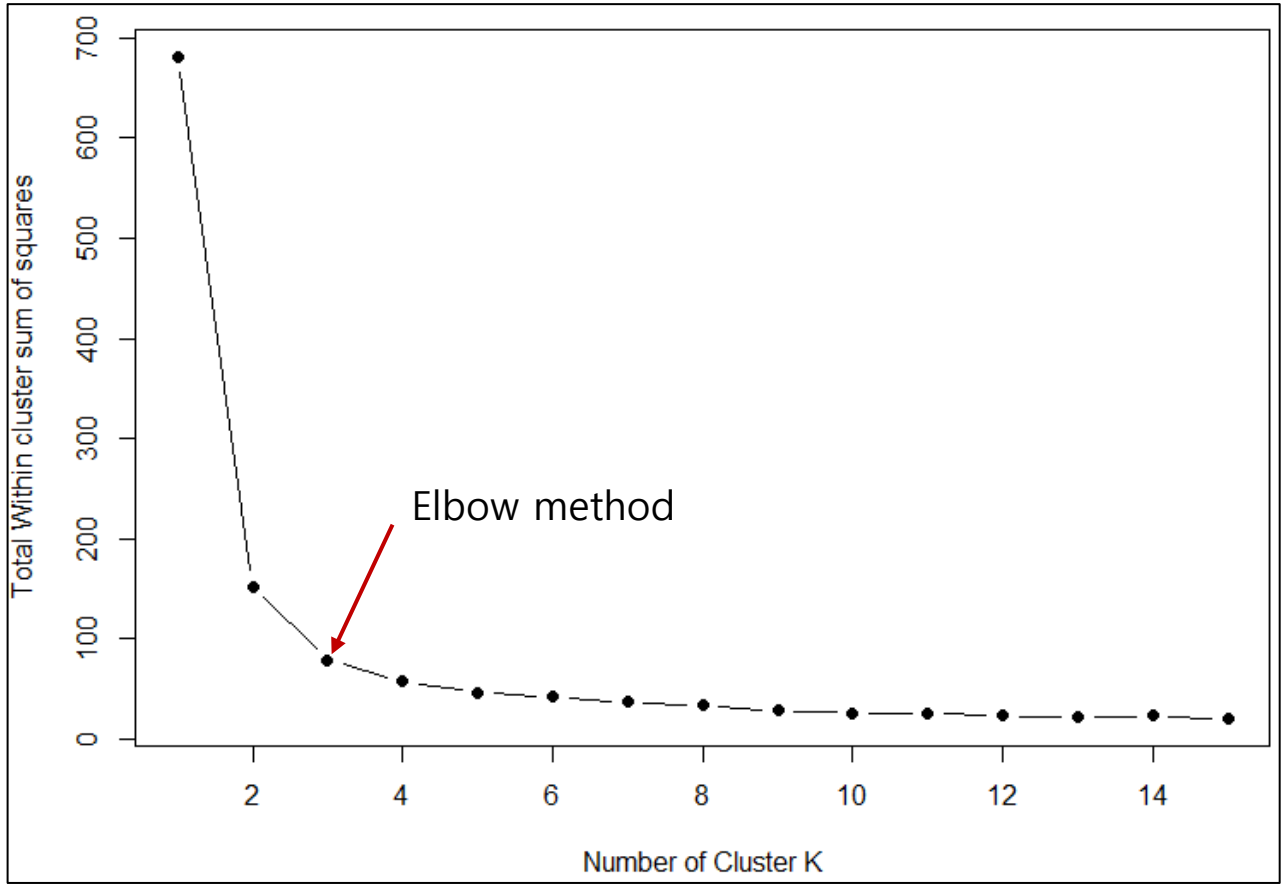
▶ elbow-method는 군집분석에서 클러스터의 수를 결정할 때 일반적으로 활용되는 방법이다.

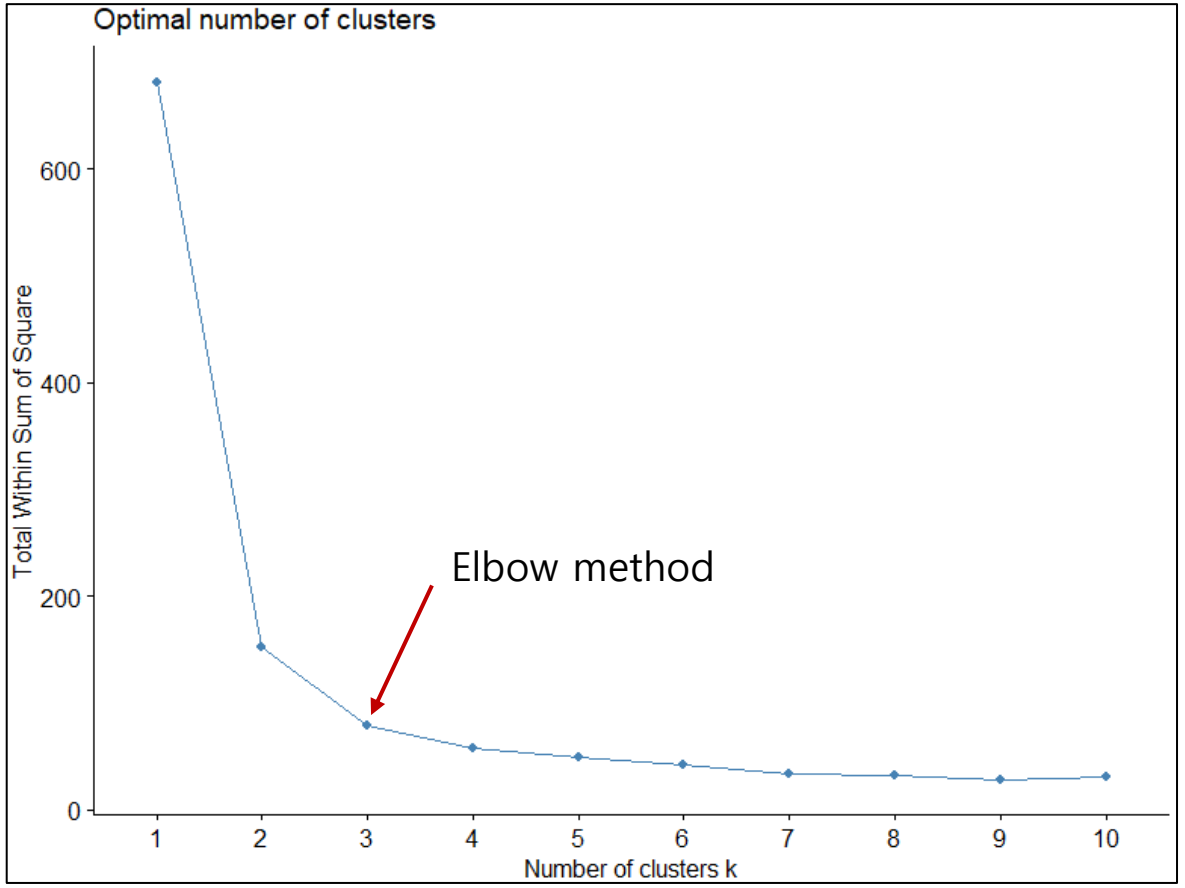
▶ X축 : 클러스터 수 / Y축 : Within Variance(SSE)으로 line을 그렸을 때, 줄어드는 양이 급격히 작게 나타나는 부분(즉, 꺾이는 부분)을 팔꿈치와 유사하다 하여 elbow-method라 부르고, 이때 팔꿈치에 해당하는 X축의 값을 클러스터의 수로 결정한다.

Best K 찾기

1. elbow-method code

```
# elbow - method 1
set.seed(1234)
func_kmeans <- function(data, k) {
  kmeans(data, k, nstart = 5)$tot.withinss
}

k.values <- 1:15
SSE <- rep(0, 15)
for(k in k.values) {
  SSE[k] <- func_kmeans(data_iris, k)
}

plot(x = k.values, y = SSE, type = "b", pch = 19,
     xlab = "Number of Cluster K", ylab = "Total Within cluster sum of squares")

# elbow - method 2
library(factoextra)
fviz_nbclust(data_iris, kmeans, method = "wss")
```


Total Within cluster sum of squares
0
100
200
300
400
500
600
700
2
4
6
8
10
12
14
Number of Cluster K
Elbow method


Optimal number of clusters
Total Within Sum of Square
0
200
400
600
1
2
3
4
5
6
7
8
9
10
Number of clusters k
Elbow method

Best K 찾기

2. Silhouette

▶ Silhouette은 클러스터 내의 일관성을 해석하고 검증하는 방법이다.

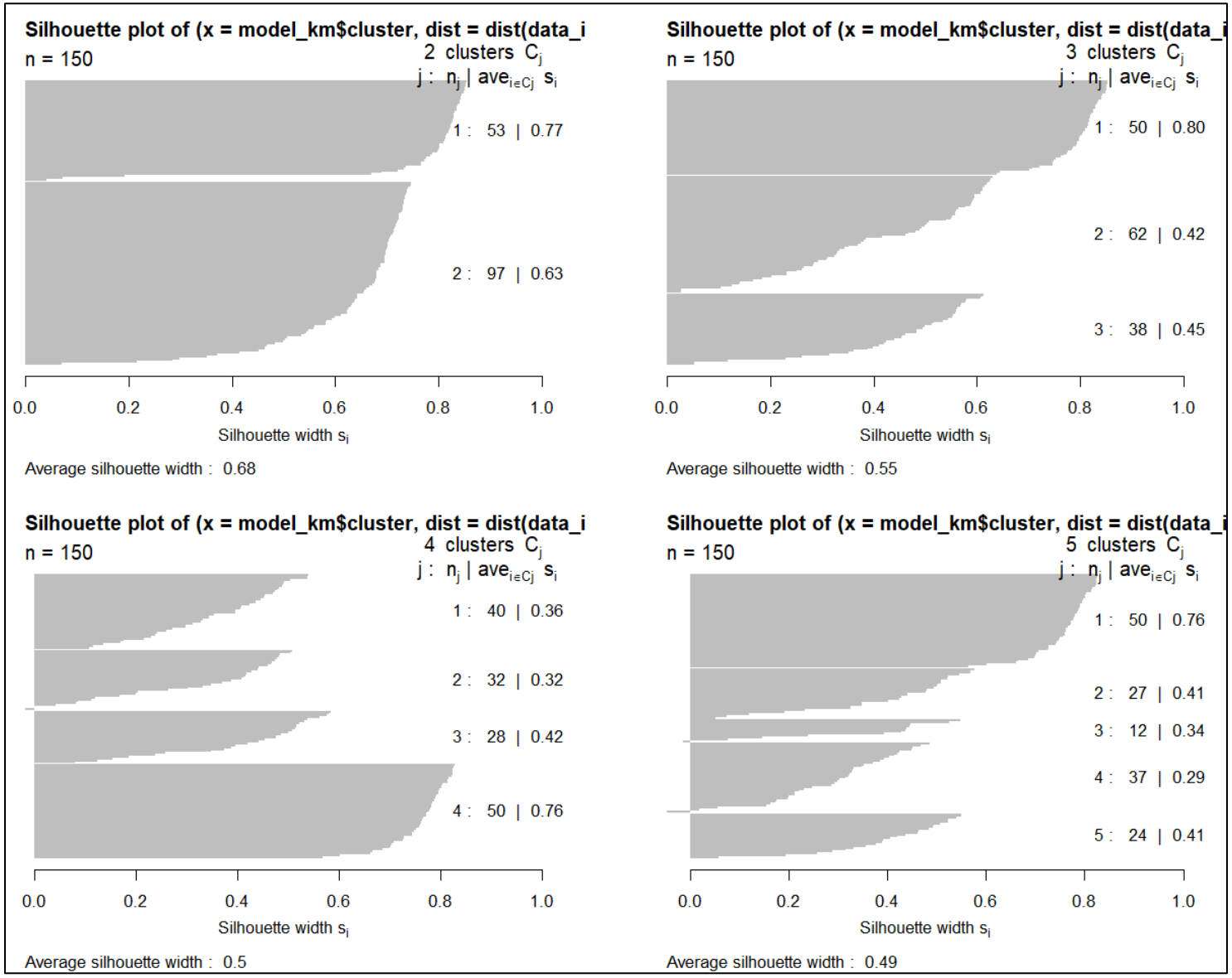
▶ Silhouette 값은 객체가 다른 군집에 비해 자신의 군집과 얼마나 유사한지를 나타내는 척도이다. Silhouette 의 범위는 -1에서 1까지며, 높은 값은 객체가 자신의 군집과 잘 맞고, 인접 군집과 일치하지 않음을 나타낸다.

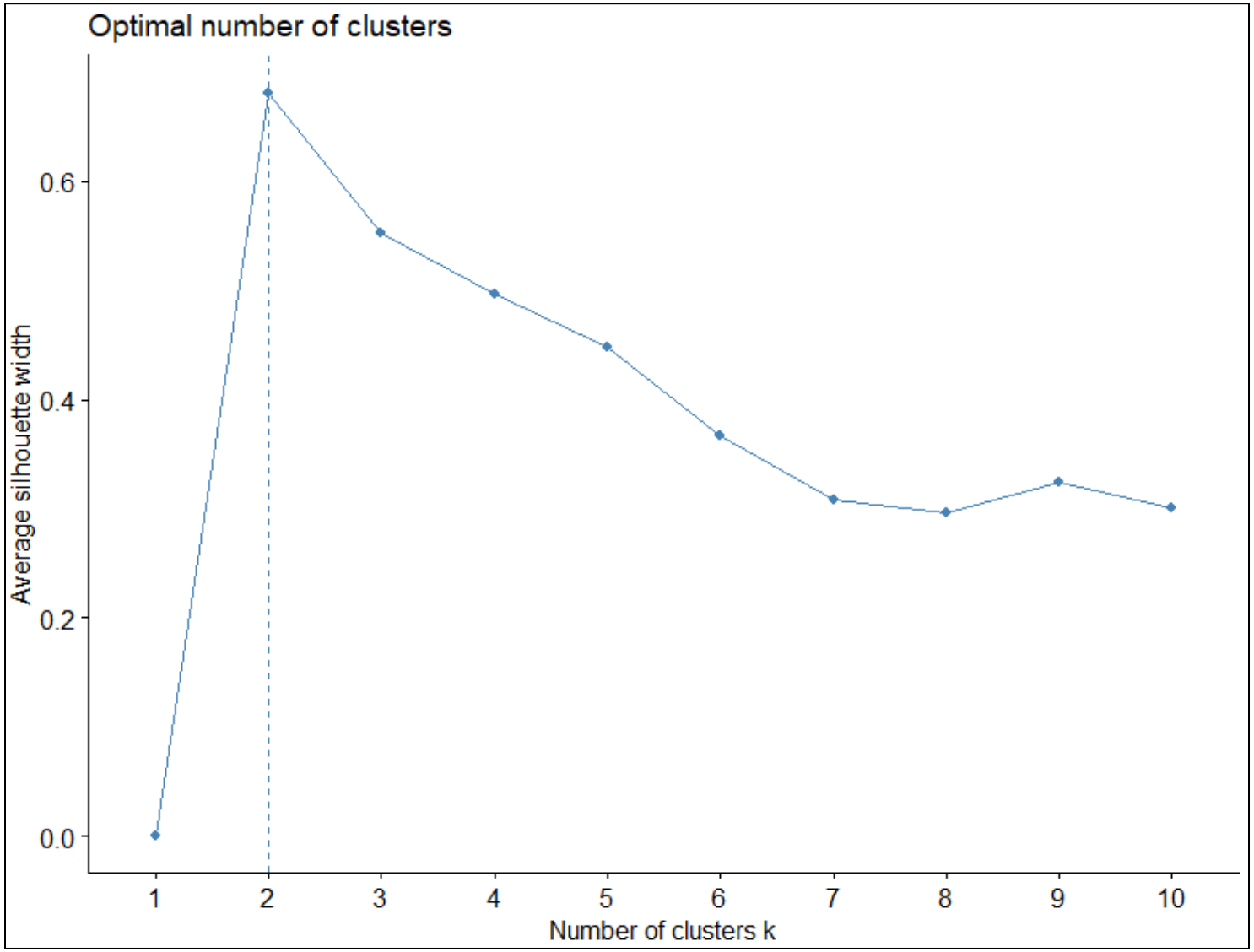
▶ 대부분의 객체가 높은 값을 갖는 경우 클러스터링 구성이 적절하다. 반대의 경우 적절하지 않다. Silhouette는 Euclidean distance, Manhattan distance 등을 통해 계산 가능하다.

Best K 찾기

2. Silhouette code

```
# silhouette
library(cluster)
windows()
par(mfrow = c(2, 2))
for(k in 2:5) {
  set.seed(1234)
  model_km <- kmeans(data_iris, k, nstart = 5)
  plot(silhouette(model_km$cluster, dist = dist(data_iris)))
}
dev.off()

# method 2
library(factoextra)
fviz_nbclust(data_iris, kmeans, method = "silhouette")
```


Silhouette plot of (x = model_km$cluster, dist = dist(data_i
n = 150
2 clusters C_j
j : n_j | $ave_{i∈Cj}$ s_i
1 : 53 | 0.77
2 : 97 | 0.63
0.0 0.2 0.4 0.6 0.8 1.0
Silhouette width s_i
Average silhouette width : 0.68
Silhouette plot of (x = model_km$cluster, dist = dist(data_i
n = 150
3 clusters C_j
j : n_j | $ave_{i∈Cj}$ s_i
1 : 50 | 0.80
2 : 62 | 0.42
3 : 38 | 0.45
0.0 0.2 0.4 0.6 0.8 1.0
Silhouette width s_i
Average silhouette width : 0.55
Silhouette plot of (x = model_km$cluster, dist = dist(data_i
n = 150
4 clusters C_j
j : n_j | $ave_{i∈Cj}$ s_i
1 : 40 | 0.36
2 : 32 | 0.32
3 : 28 | 0.42
4 : 50 | 0.76
0.0 0.2 0.4 0.6 0.8 1.0
Silhouette width s_i
Average silhouette width : 0.5
Silhouette plot of (x = model_km$cluster, dist = dist(data_i
n = 150
5 clusters C_j
j : n_j | $ave_{i∈Cj}$ s_i
1 : 50 | 0.76
2 : 27 | 0.41
3 : 12 | 0.34
4 : 37 | 0.29
5 : 24 | 0.41
0.0 0.2 0.4 0.6 0.8 1.0
Silhouette width s_i
Average silhouette width : 0.49


Optimal number of clusters
Average silhouette width
0.0
0.2
0.4
0.6
1
2
3
4
5
6
7
8
9
10
Number of clusters k

별첨. K-means Clustering on Image

참고 : https://www.r-bloggers.com/r-k-means-clustering-on-an-image/

이미지 분할

▶ 디지털 이미지는 pixel 들이 모여 보여진다.

▶ 각 pixel의 R, G, B 값으로 이미지를 array 형태로 만들 수 있다.

Ex) 해상도 500 x 300의 이미지 파일은 (500, 300, 3) 형태의 array와 같다.

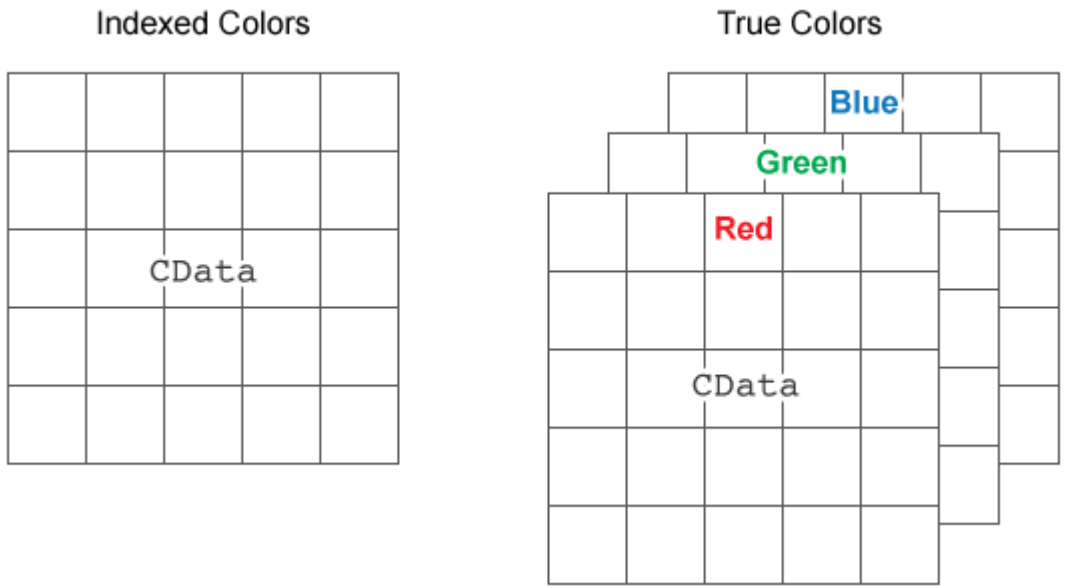
▶ 이때 각 pixel 단위로 분리된 데이터를 비슷한 특성을 보이는 obs끼리 clustering 해 주어진 이미지를 단순화하여 이미지를 분석하기 좀 더 편한 형태로 만드는 과정이다.

Code

```
############## kmeans를 활용한 image 분석 ##############
setwd("")  # image 파일이 포함된 working directory 지정할 것
# Load the package
library(jpeg)
library(ggplot2)

img <- readJPEG("ColorfulBird.jpg") # Read the image
str(img)

# Obtain the dimension
imgDm <- dim(img)

# Assign RGB channels to data frame
imgRGB <- data.frame(
  x = rep(1:imgDm[2], each = imgDm[1]),
  y = rep(imgDm[1]:1, imgDm[2]),
  R = as.vector(img[,,1]),
  G = as.vector(img[,,2]),
  B = as.vector(img[,,3])
)

# Plot the image
windows()
ggplot(data = imgRGB, aes(x = x, y = y)) +
  geom_point(colour = rgb(imgRGB[c("R", "G", "B")])) +
  labs(title = "Original Image: Colorful Bird") +
  xlab("x") +
  ylab("y")
# dev.off()
```

```
# clustering
windows()
kClusters <- 2
kMeans <- kmeans(imgRGB[, c("R", "G", "B")], centers = kClusters)
kColours <- rgb(kMeans$centers[kMeans$cluster,])

ggplot(data = imgRGB, aes(x = x, y = y)) +
  geom_point(colour = kColours) +
  labs(title = paste("k-Means Clustering of", kClusters, "Colours")) +
  xlab("x") +
  ylab("y")
# dev.off()
```

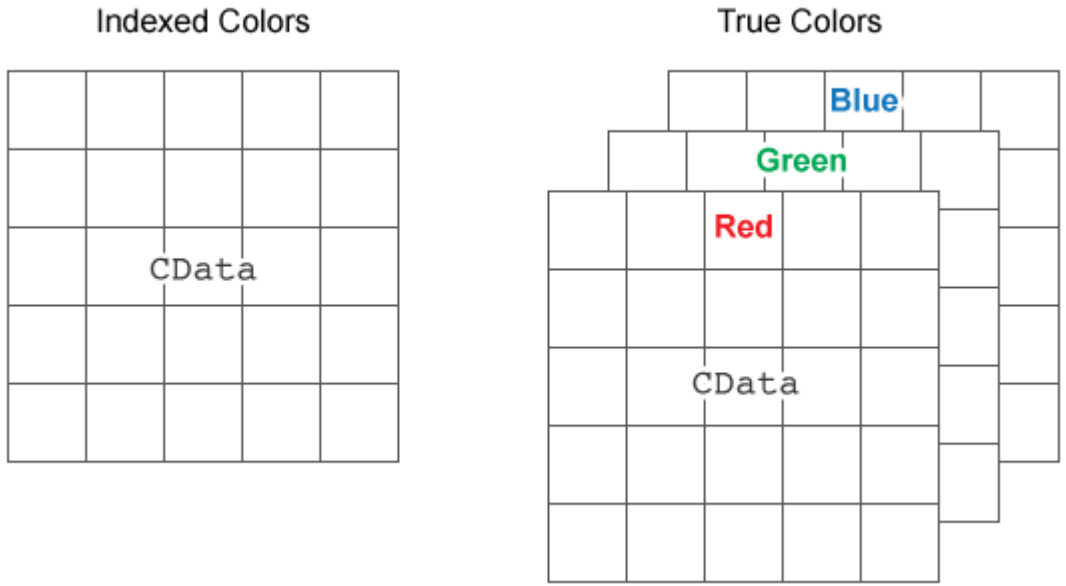
image array



0	1	2	3	4
5	6	7	8	9
10	11	12	13	14
15	16	17	18	19
20	21	22	23	24
25	26	27	28	29

0	1	2	3	4	5	6	7	8	...

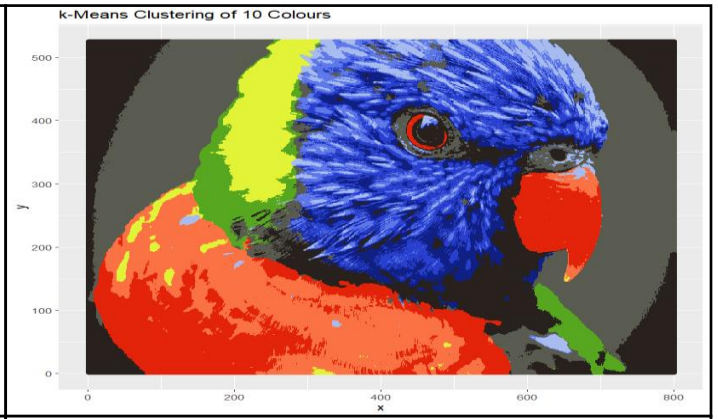
image array



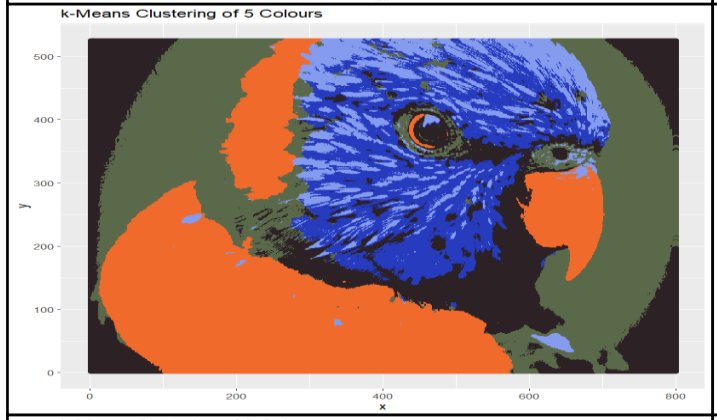
X	Y	R	G	B
1	5	0.1	0.2	0.1
1	4	0.2	0.3	0.4
1	3	0.1	0.2	0.4
1	2	0.4	0.2	0.8
1	1	0.7	0.2	0.4

original

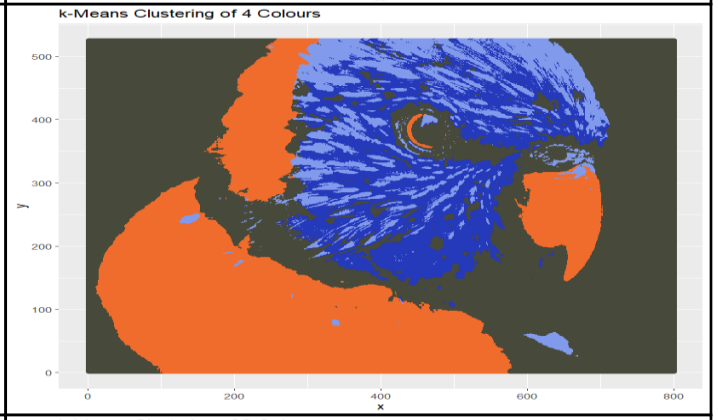
K = 10



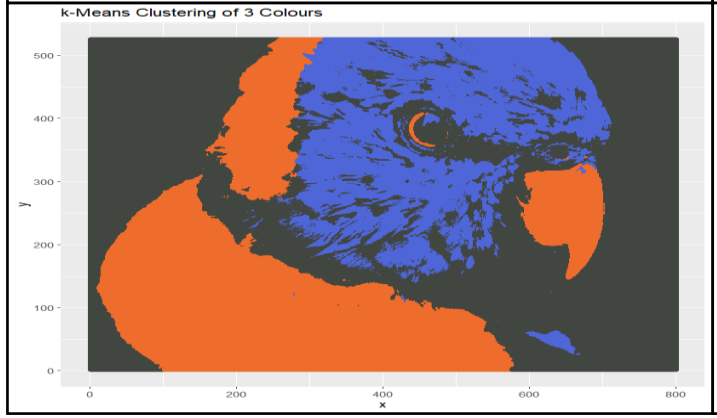
K = 5



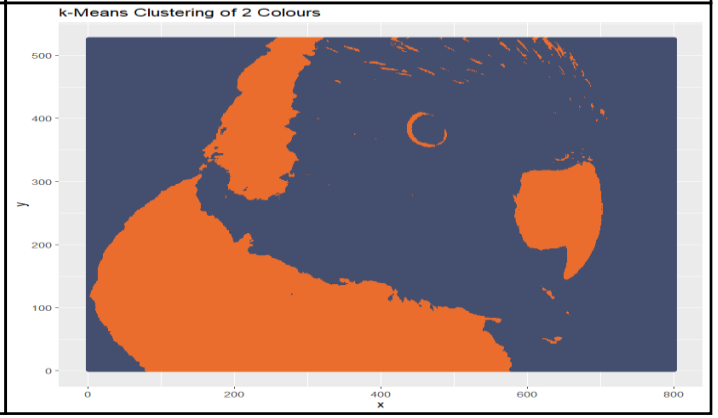
K = 4



K = 3



K = 2



KNN / KMEANS

감사합니다