# k-means, hierarchical clustering

*Philip oh*

```
library(ggplot2)
```

## k-means clustering

`kmeans(x, centers)` - x에는 데이터를 넣는다. - `centers` : 클러스터의 개수 or initial center를 정해줄 수 있다. 첫 번째 센터를 어떻게 정하느냐에 따라 결과는 달라질 수 있다. - `nstart` : 몇번 새로 시작할지 정해준다. 만약 10으로 하면, initial center를 10번 정해서 각각 kmeans를 돌려본다. - `algorithm` : 각 알고리즘은 추정을 얼마나 정확하게 하느냐, 빠르게 하느냐가 다르다.

```
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
```

```
a = kmeans(iris[,3:4], 3, nstart=20)
a
```

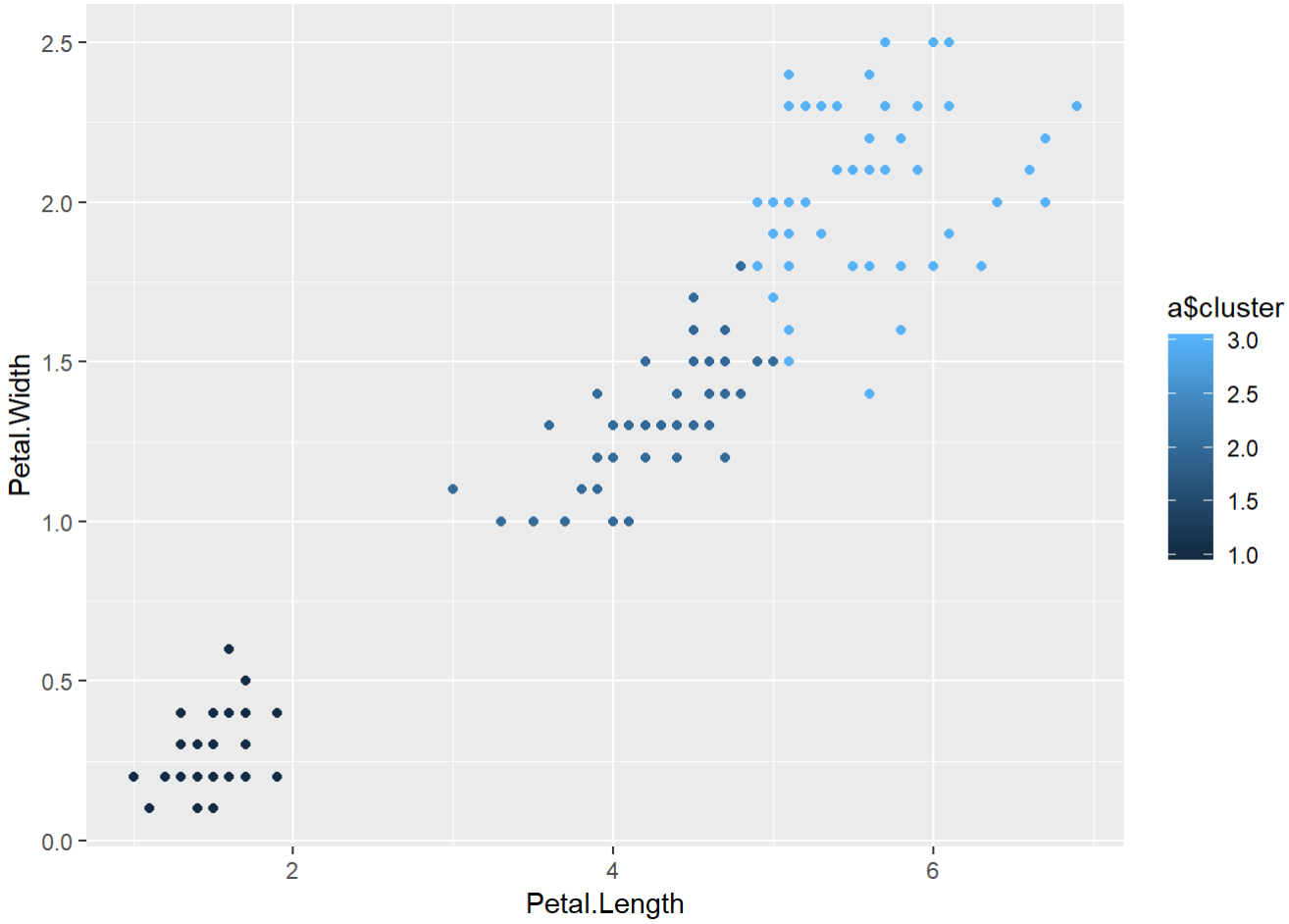
```
## K-means clustering with 3 clusters of sizes 50, 52, 48
##
## Cluster means:
##   Petal.Length Petal.Width
## 1     1.462000    0.246000
## 2     4.269231    1.342308
## 3     5.595833    2.037500
##
## Clustering vector:
##   [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
##  [36] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 2 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 2 2 2 3 3 3 3 3 3 3 3 3
## [106] 3 2 3 3 3 3 3 3 3 3 3 3 3 2 3 3 3 3 3 2 3 3 3 3 3 3 3 3 3 2 3 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"
```

- 우리는 이 데이터의 군집이 3개라는 걸 알고 있으니까 두번째 argument를 3으로 했지만, 보통은 2로 시작한다.

결과 해석

- K-means clustering with 3 clusters of sizes 52, 50, 48
- => 맨끝의 숫자(50, 48, 42)는 각각의 클러스터에 몇개의 데이터가 들어갔는지 보여준다.
- Clustering means(중요)
- => 각 군집의 센터를 의미한다.
- Clustering vector(중요)
- => 각 데이터가 어느 클러스터에 속하는지 알려준다.
- within cluster는 군집 내 거리를 의미한다. 작을 수록 군집끼리 더 모여 있다는 것을 의미한다. 즉, 작을 수록 좋다.
- between cluster는 군집 간 거리를 의미한다. 클 수록 군집간 거리가 멀다는 것을 의미한다.
- 군집 간 거리가 멀고 군집 내 거리는 가까운 것이 좋다는 것을 유추해볼 수 있다.
- 각 군집내 센터와 각 데이터들의 거리의 제곱합
- total_ss = within cluster_ss + between cluster_ss
- => between cluster / total이 클수록 total에서 between cluster 값이 크다는 것을 의미하고, 군집 간 거리가 멀다는 것을 의미한다.

```
ggplot(iris, aes(Petal.Length, Petal.Width, color = a$cluster)) +
  geom_point()
```

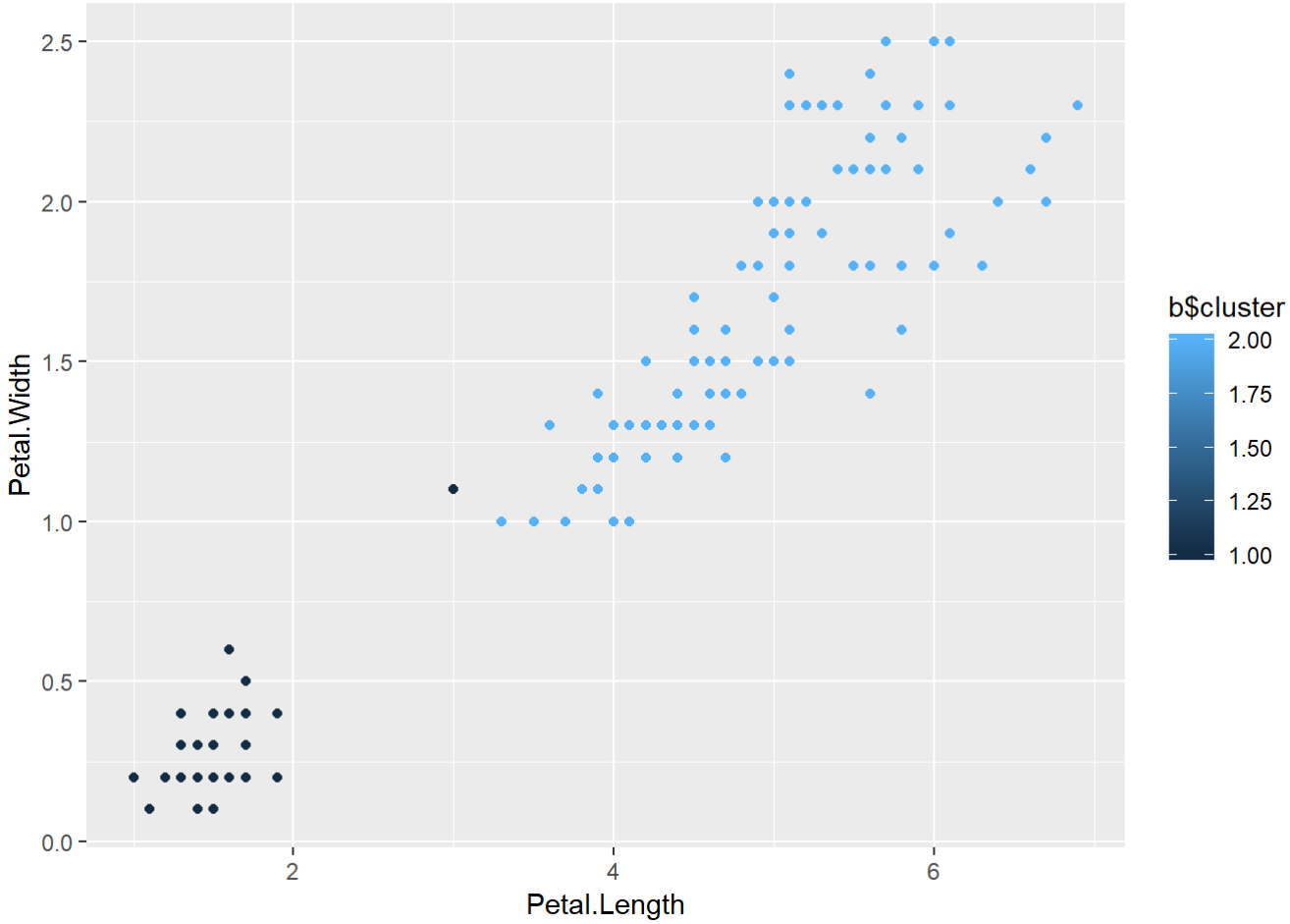



- 만약 시각화를 하고 싶다면, color에 clustering vector를 넣어주면 된다.

centers를 2개로 바꿔보았다.

- 실습에선 다루지 않은 내용이지만, 호기심이 생겨 centers를 2로 바꿔보았다. 데이터에 대해 모르는 상태에서 산점도만 보면, 누가 봐도 2개의 군집으로 나눌 수 있을 것 같은데, k-means는 과연 어떻게 군집화할까?

```
b = kmeans(iris[,3:4], 2, nstart=20)
ggplot(iris, aes(Petal.Length, Petal.Width, color = b$cluster)) +
  geom_point()
```




- centers를 2로 바꾸었더니 예상과는 다르게 하나의 관측치가 잘못 분류된 것처럼 보인다. 왜 이런 결과가 나왔을까?
- 시각적으로 보면, 잘못 분류된 관측치는 분명 오른쪽 군집에 속해야 하는 게 맞다. 하지만, k-means는 각 군집의 중심을 기준으로 데이터들을 군집화한다. 저 관측치는 오른쪽 군집의 중심점보다 왼쪽의 중심점에 더 가까우므로 왼쪽에 분류되었을 것이라 추측해볼 수 있다.
- 만약 이 관측치를 오른쪽 군집에 포함시키려면 어떻게 해야 할까? 아쉽게도 kmeans 함수 자체에서 조정할 수 있는 옵션은 없다.
- 아마 몇 가지 방법이 있을 것이다. 내가 생각한 방법은, 애초에 3개의 군집으로 시작해서 이후에 조작을 통해서 1번 군집과 2,3번 군집으로 조정하는 것이다. clustering vector에는 1~3까지 있을 것이고 임의의 조작을 통해서 2, 3을 하나의 군집으로 모으는 것이다.

## hierarchical clustering

`hclust(d,)` - d는 distacne(거리)를 의미한다. 매트릭스가 아니라 dist라는 형식을 사용한다는 것에 주의한다. - 만약 x라는 매트릭스가 있으면 그냥 dist(x)라고 하면, 알아서 바꿔준다. - method는 교재에 나와있는 complete, single 등의 방식으로 정해줄 수 있다.

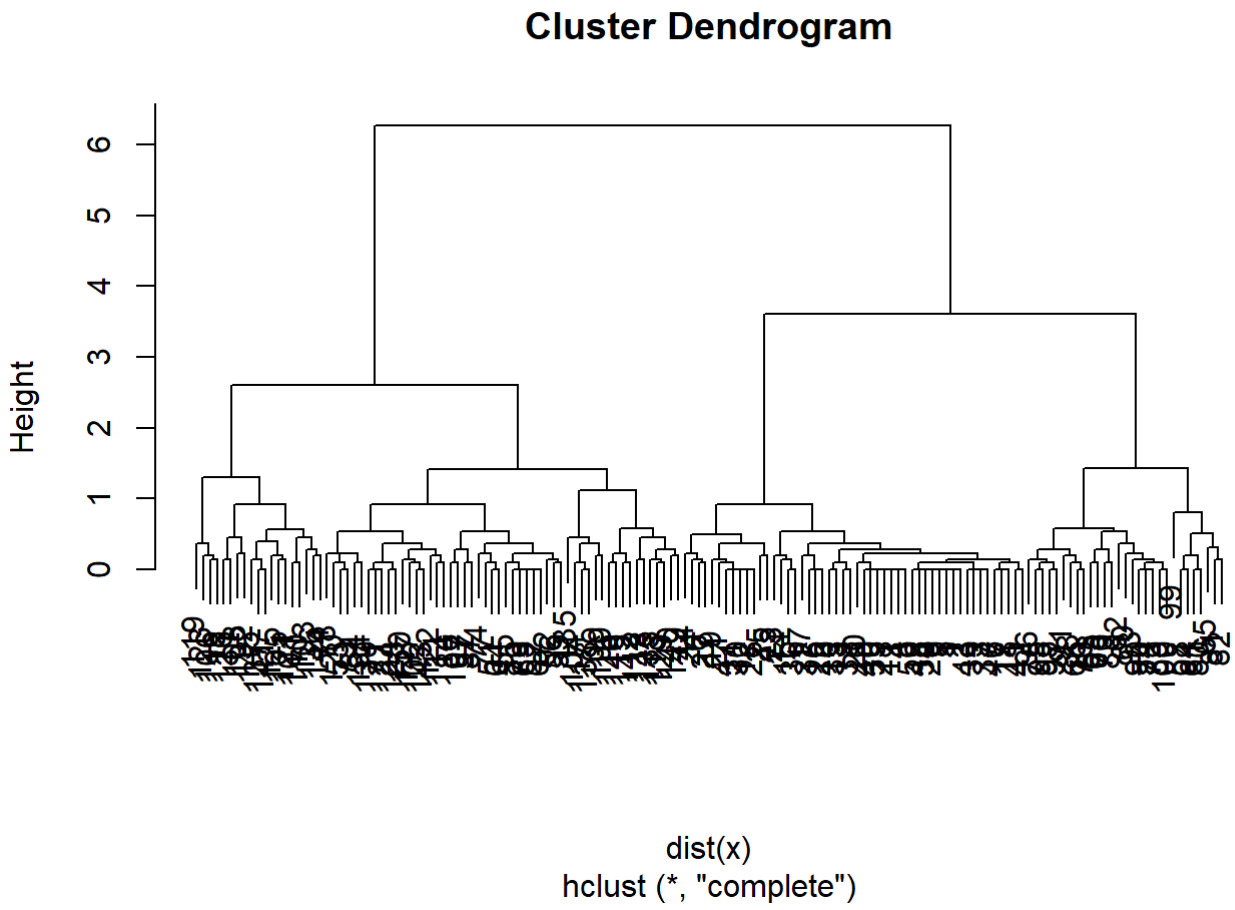
```
x = iris[, 3:4]
dist(x)
```

- 결과값이 너무 길어서 실행만 하고 출력은 하지 않았다.

```
hc.complete=hclust(dist(x), method="complete")
str(hc.complete)
```

```
## List of 7
##  $ merge      : int [1:149, 1:2] -1 -5 -9 -29 -34 -48 -50 -3 -39 -43 ...
##  $ height     : num [1:149] 0 0 0 0 0 0 0 0 0 0 ...
##  $ order      : int [1:150] 119 123 106 118 126 131 108 132 145 137 ...
##  $ labels     : NULL
##  $ method     : chr "complete"
##  $ call       : language hclust(d = dist(x), method = "complete")
##  $ dist.method: chr "euclidean"
##  - attr(*, "class")= chr "hclust"
```

```
plot(hc.complete)
```




- plot을 통해 바로 덴도그램으로 시각화 할 수 있다.
- 만약 2개의 클러스터로 자른다고 한다면, 아래와 같이 하면 된다.

```
h.clust = cutree(hc.complete, 2) # 2는 클러스터의 개수를 의미한다.
table(iris$Species, h.clust) # 이렇게 하면, 원래의 데이터와 군집화 후의 데이터를 비교할 수 있다.
```

```
##             h.clust
##                1  2
##   setosa      50  0
##   versicolor 29 21
##   virginica   0 50
```

- hclust는 계산도 오래 걸리고 linkage에 따라 결과값이 많이 달라지는 등 단점이 많아서 일반적으론 kmeans가 더 선호된다.

- 하지만 예를 들어 남녀를 구분하고 그 안에서 백인, 황인, 흑인으로 구분해야 하는 경우 hclust가 굉장히 유용하다.