



UNSUPERVISED LEARNING IN R

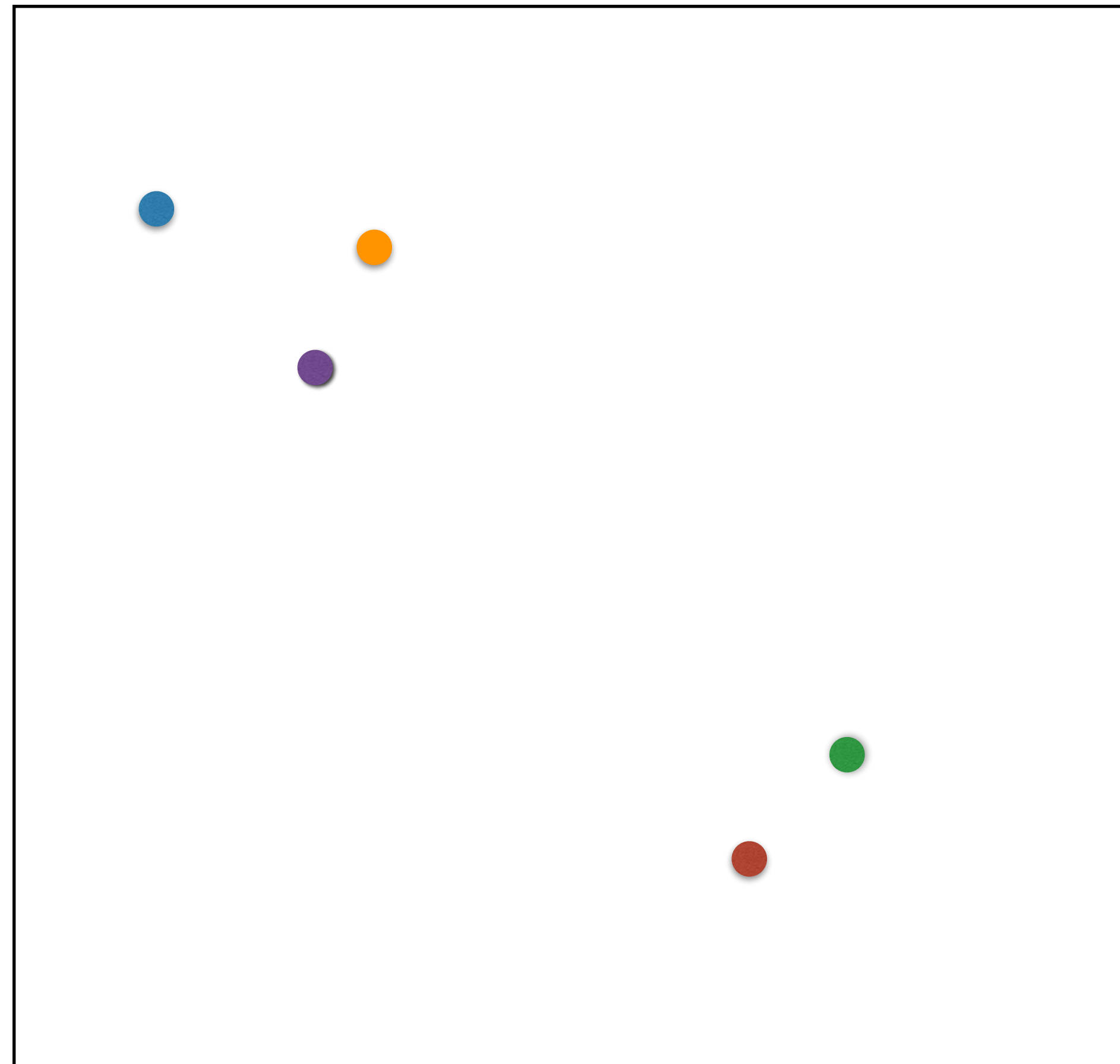
# **Introduction to hierarchical clustering**

# Hierarchical clustering

- Number of clusters is not known ahead of time
- Two kinds: bottom-up and top-down, this course bottom-up

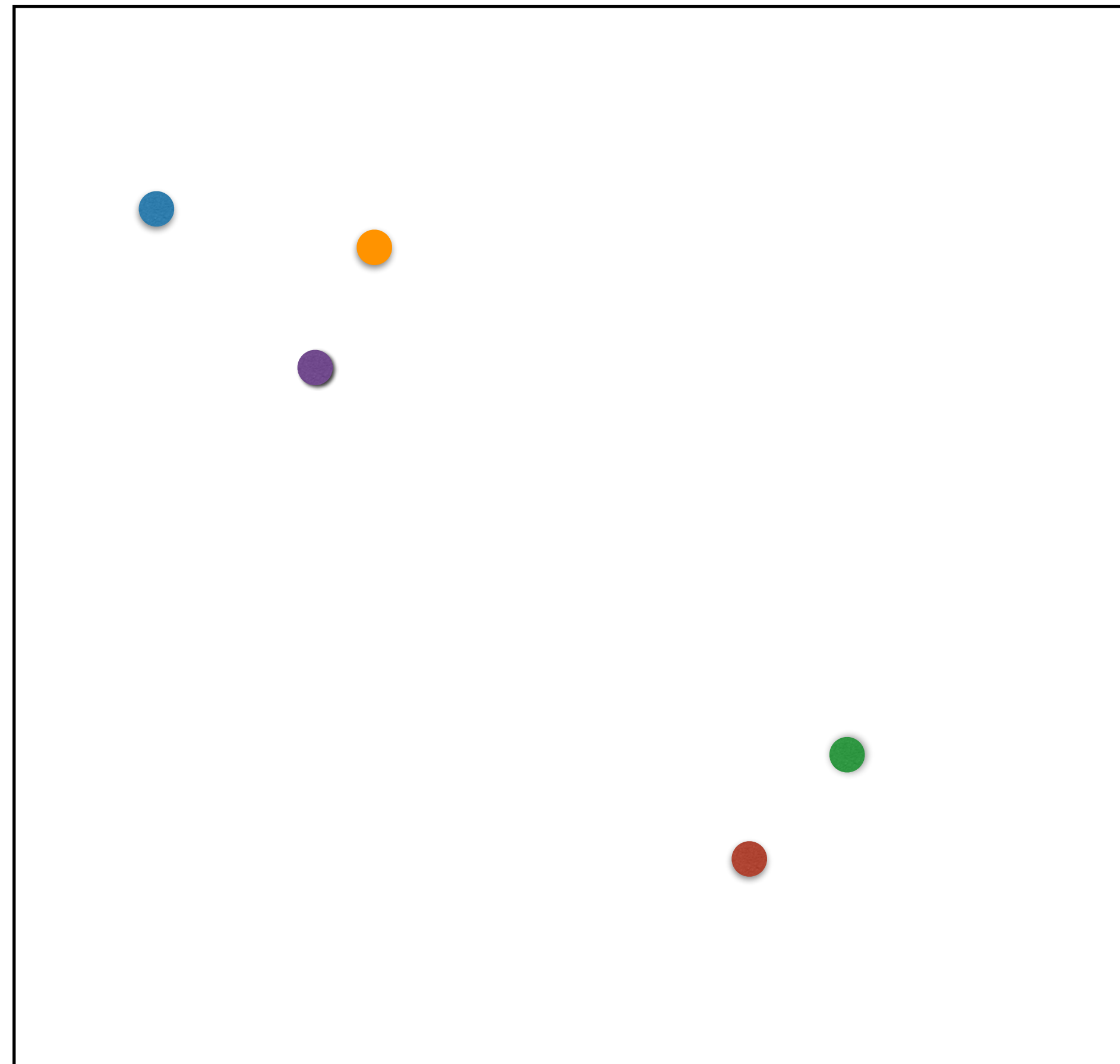
# Hierarchical clustering

Simple Example



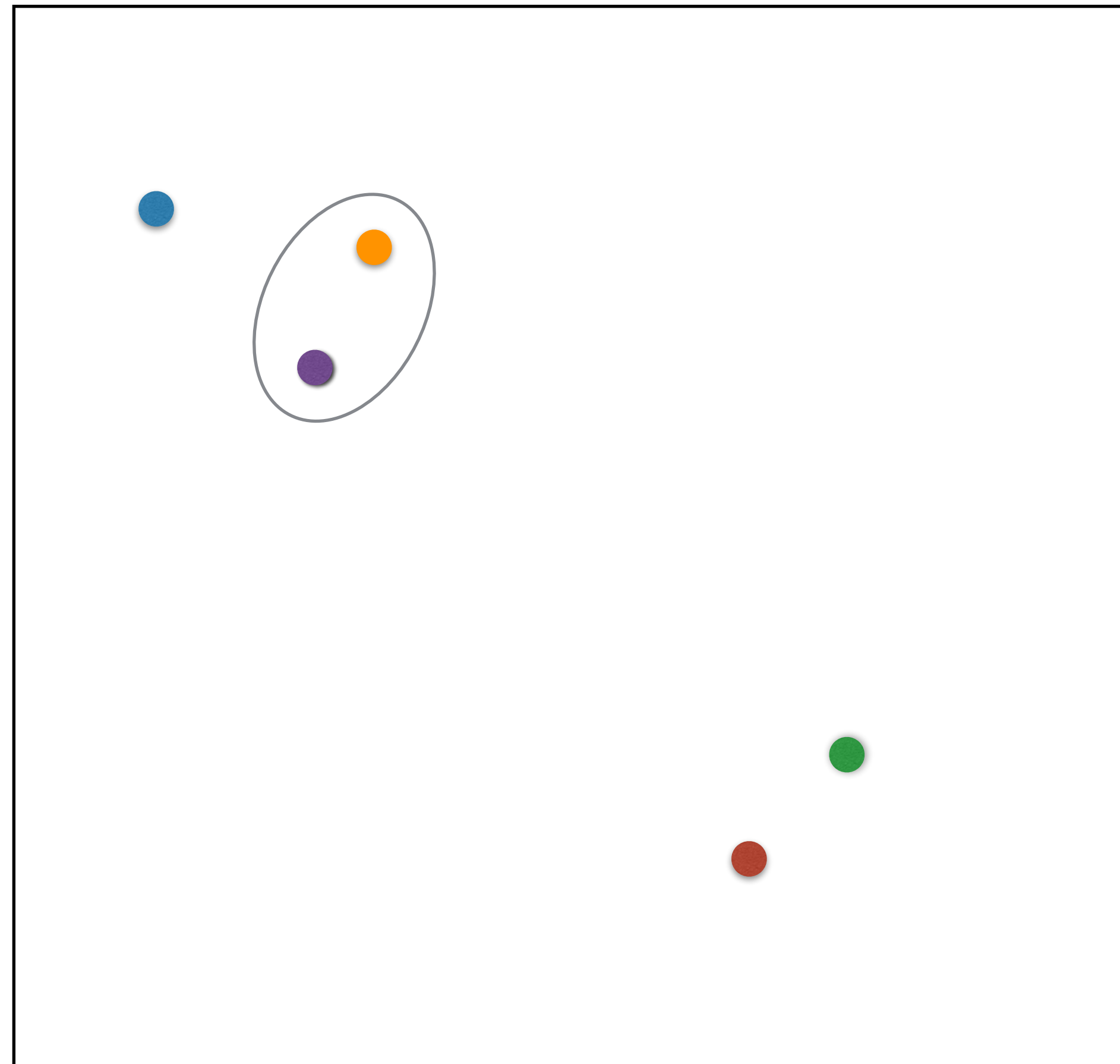
# Hierarchical clustering

5 Clusters  
Each point a cluster



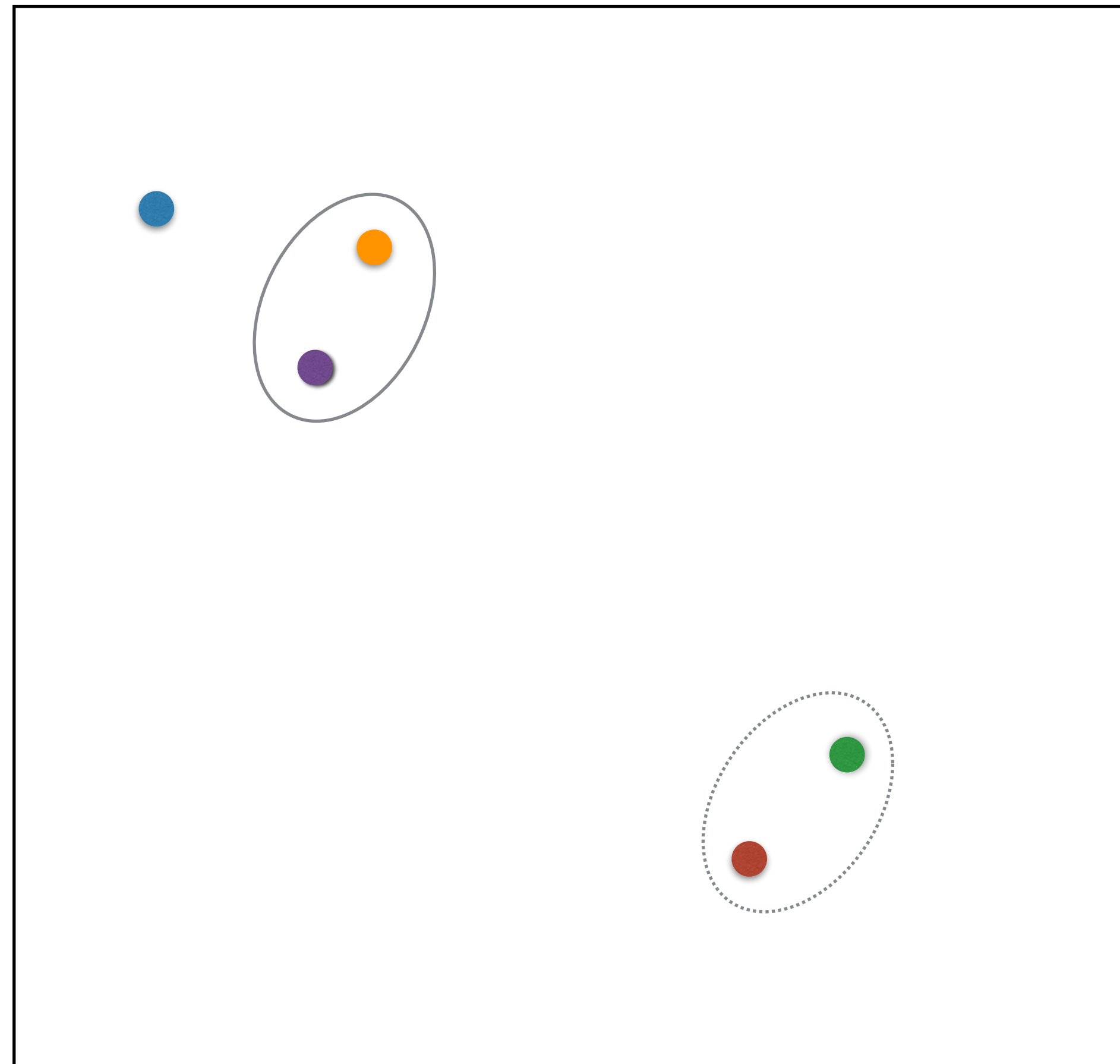
# Hierarchical clustering

4 Clusters



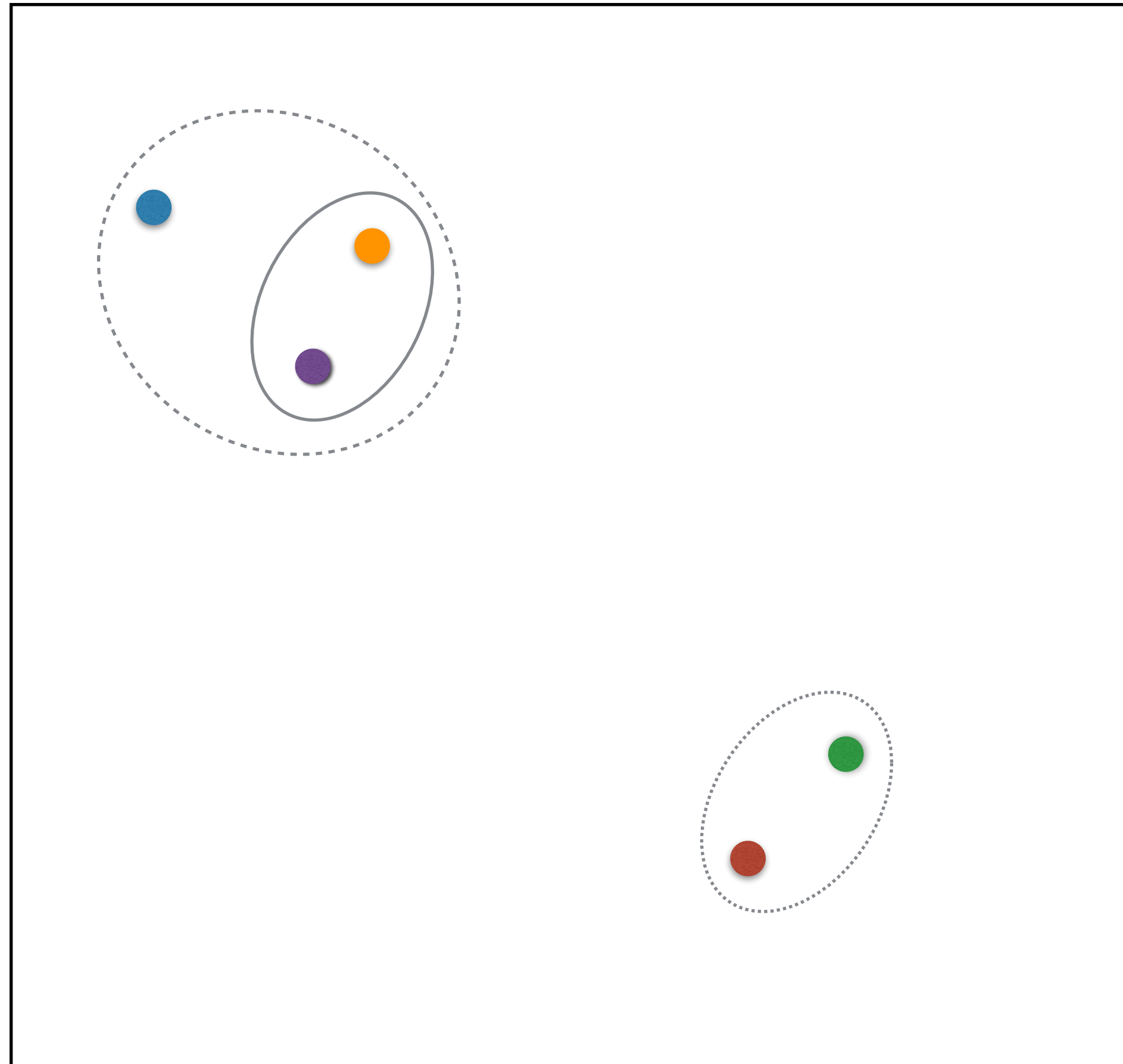
# Hierarchical clustering

3 Clusters



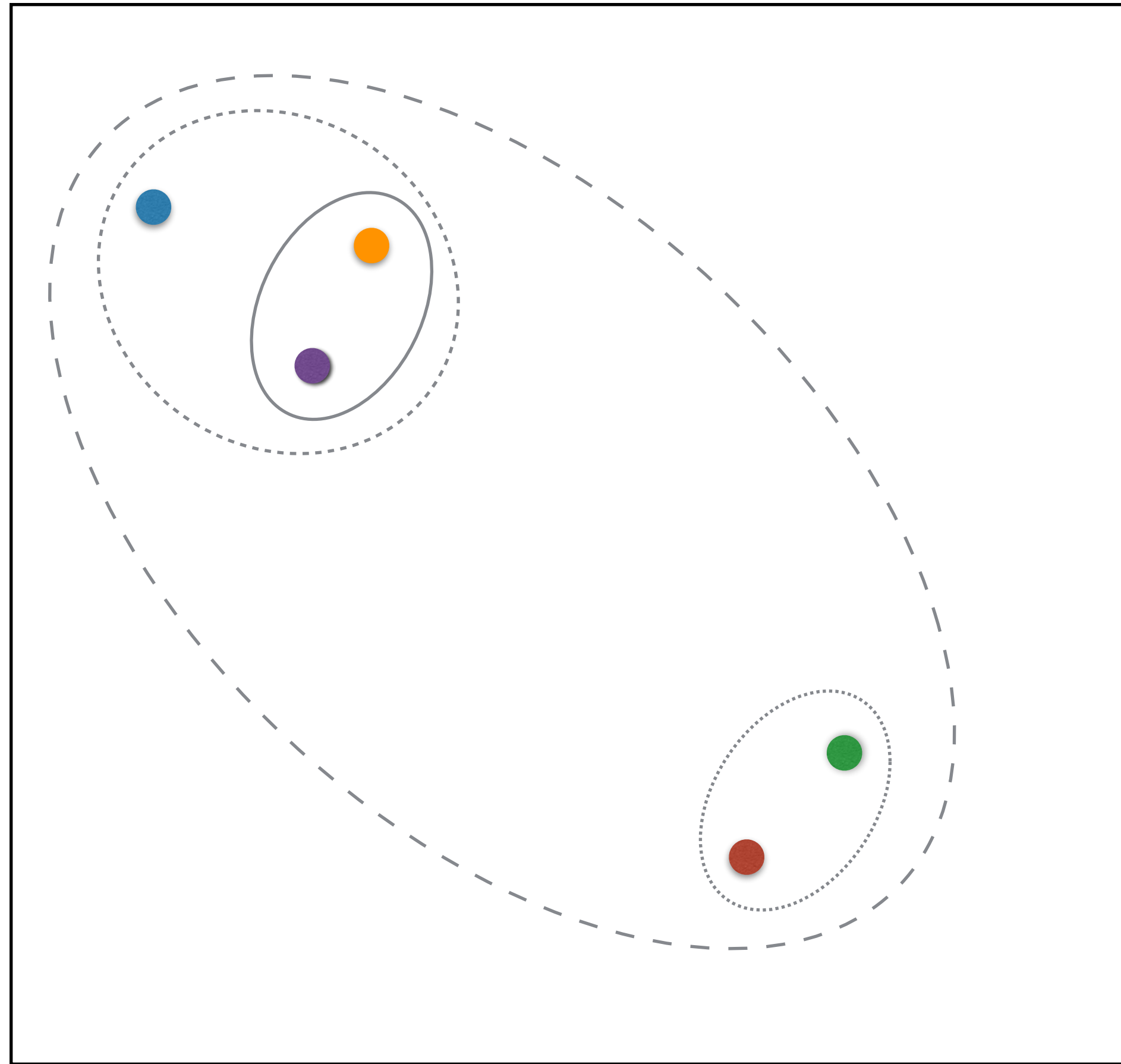
# Hierarchical clustering

2 Clusters



# Hierarchical clustering

1 Cluster





# Hierarchical clustering in R

```
> # Calculates similarity as Euclidean distance between observations
> dist_matrix <- dist(x) x is a data matrix

> # Returns hierarchical clustering model
> hclust(d = dist_matrix)
```

Call:

```
hclust(d = s)
```

```
Cluster method      : complete
```

```
Distance           : euclidean
```

```
Number of objects: 50
```



UNSUPERVISED LEARNING IN R

**Let's practice!**



UNSUPERVISED LEARNING IN R

# Selecting number of clusters

# Interpreting results

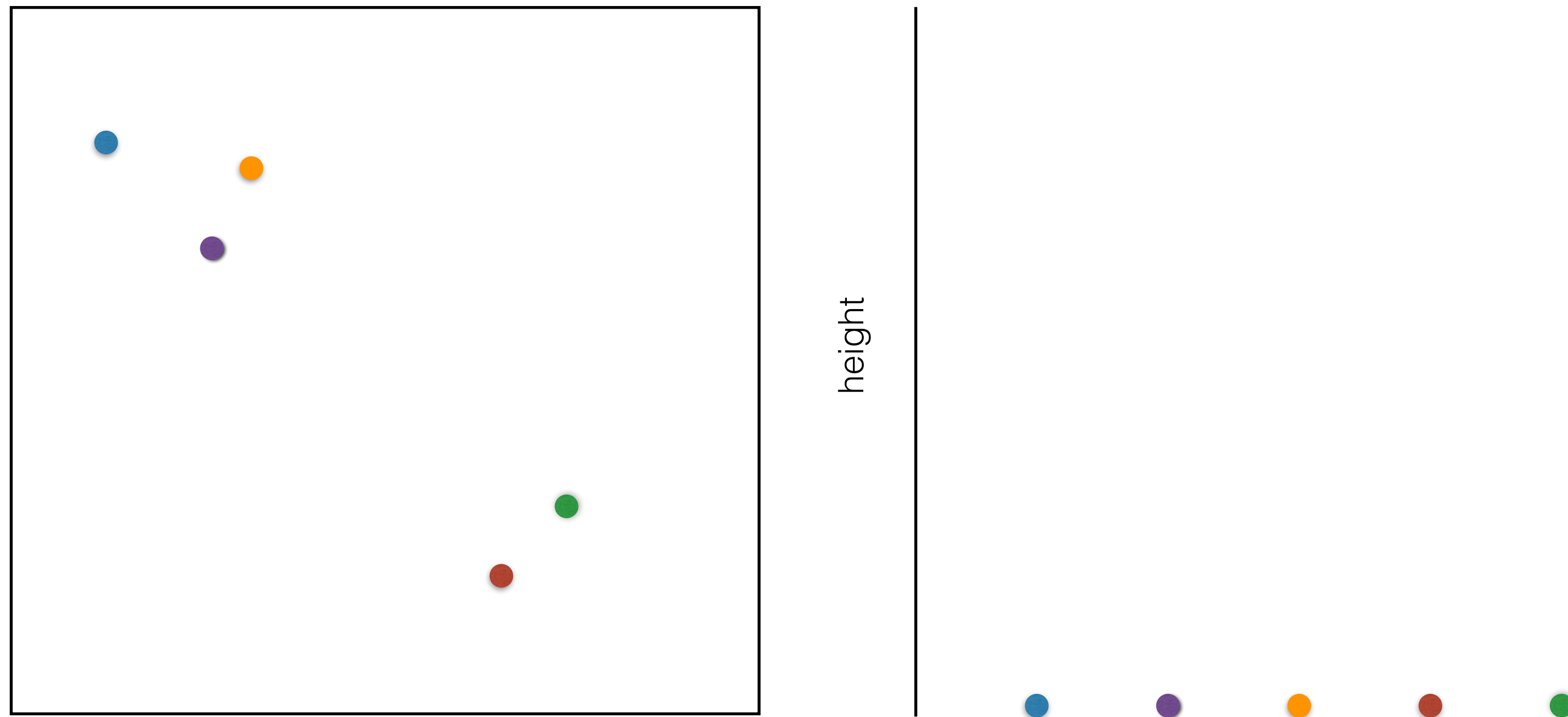
```
> # Create hierarchical cluster model: hclust.out  
> hclust.out <- hclust(dist(x))  
  
> # Inspect the result  
> summary(hclust.out)
```

	Length	Class	Mode
merge	98	-none-	numeric
height	49	-none-	numeric
order	50	-none-	numeric
labels	0	-none-	NULL
method	1	-none-	character
call	2	-none-	call
dist.method	1	-none-	character

**Information isn't particularly useful**

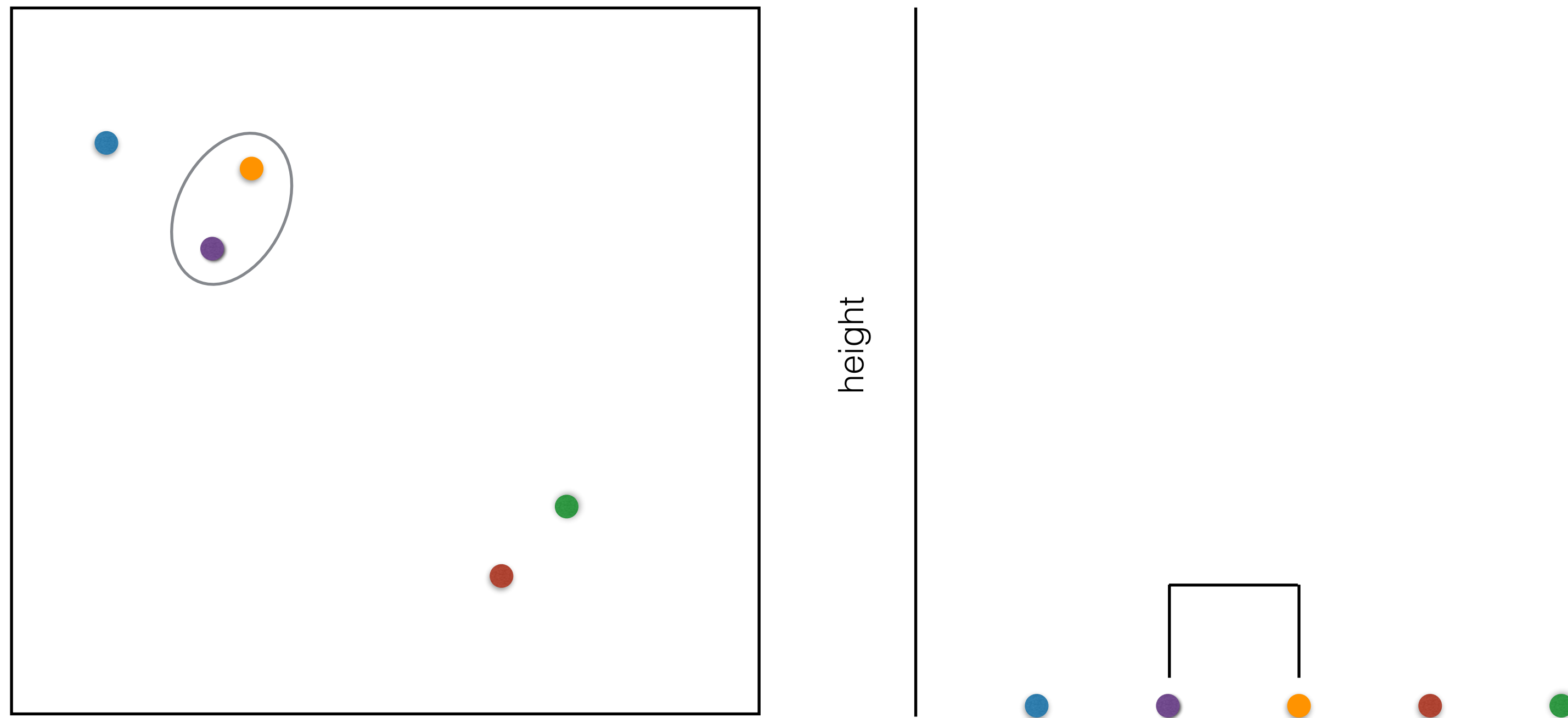
# Dendrogram

- Tree shaped structure used to interpret hierarchical clustering models



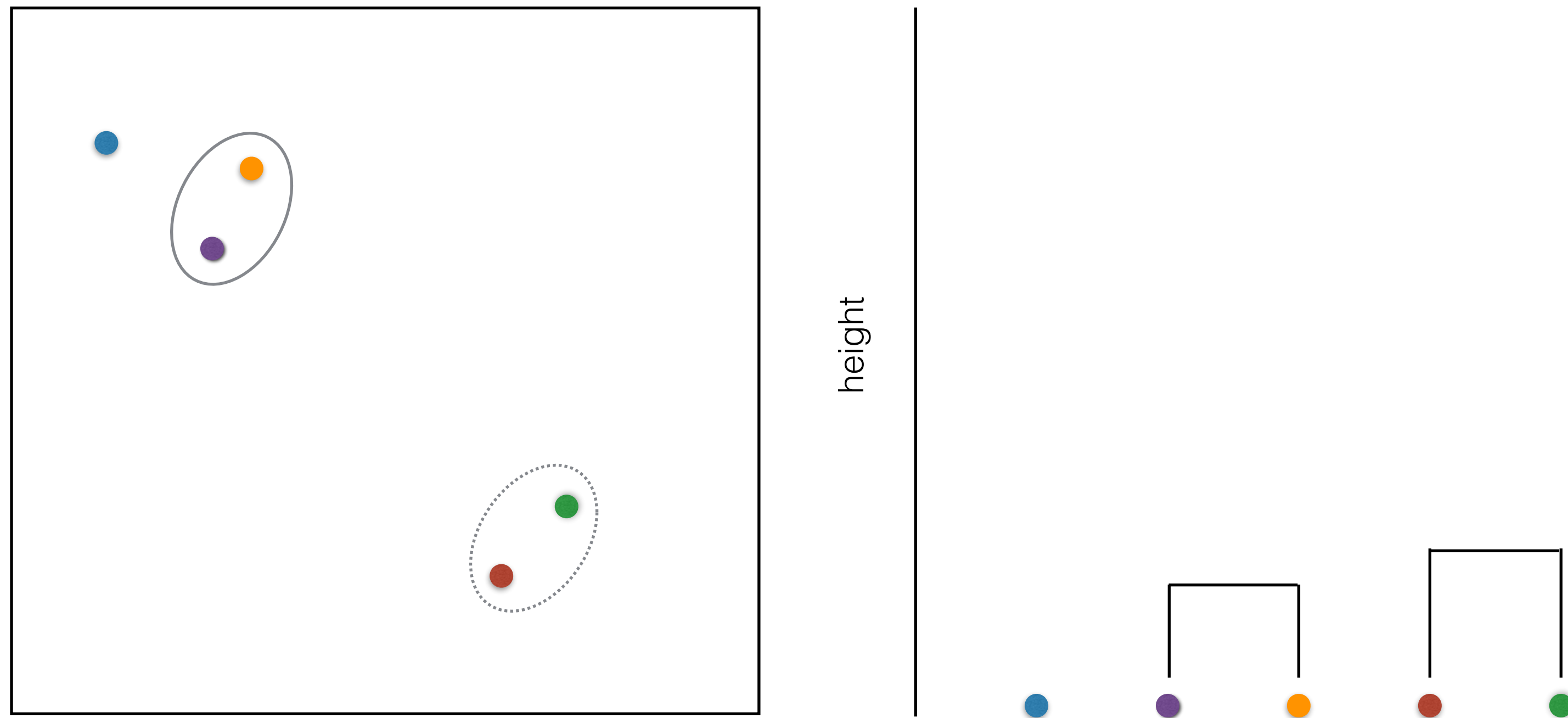
# Dendrogram

- Tree shaped structure used to interpret hierarchical clustering models



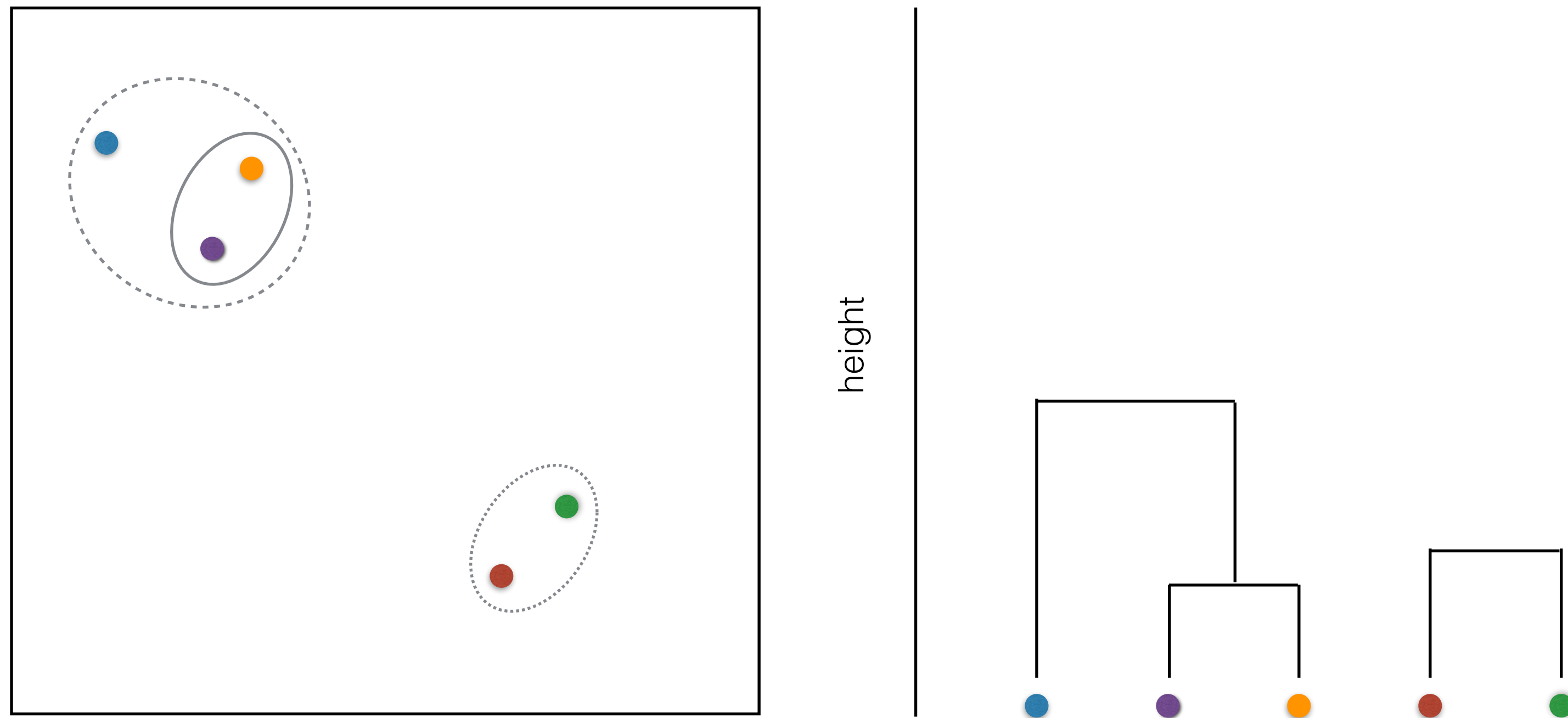
# Dendrogram

- Tree shaped structure used to interpret hierarchical clustering models



# Dendrogram

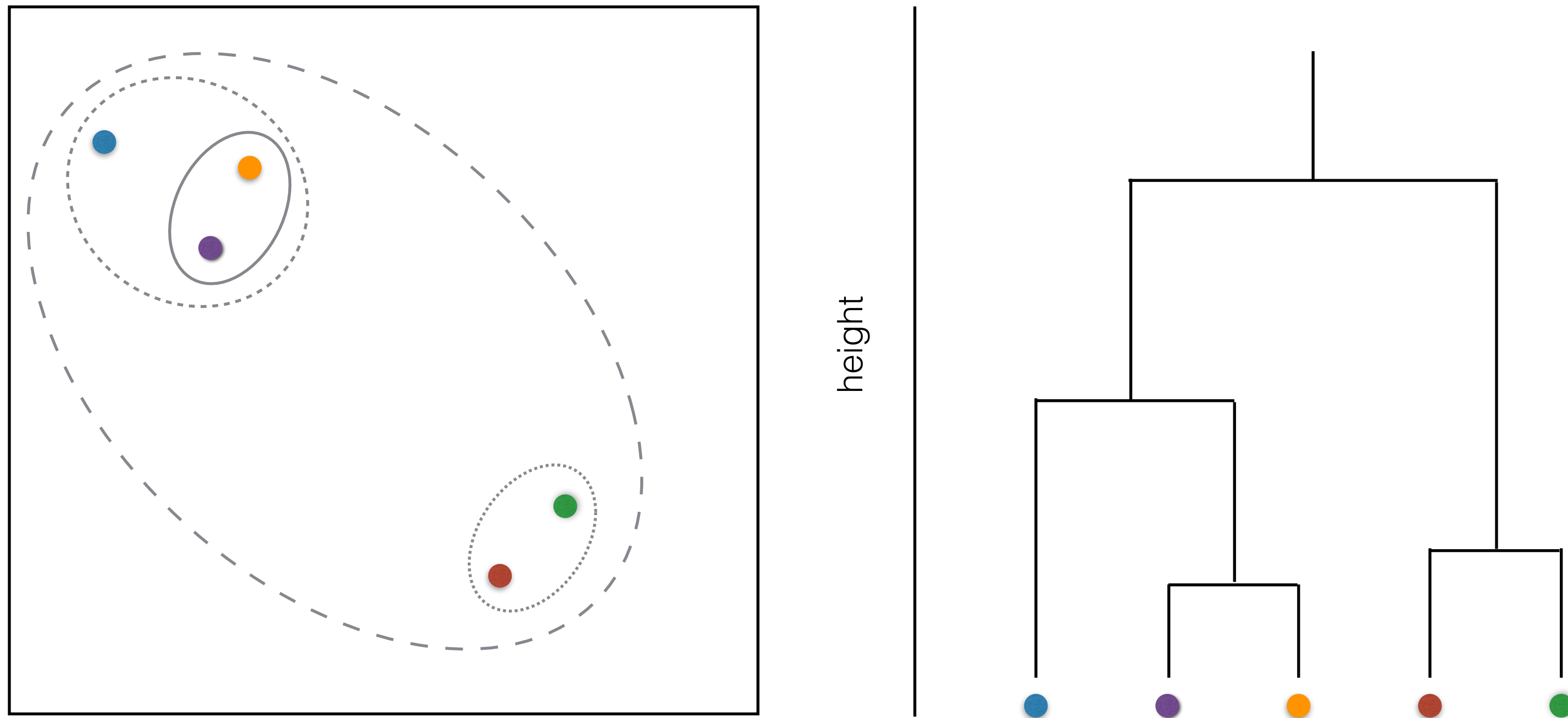
- Tree shaped structure used to interpret hierarchical clustering models





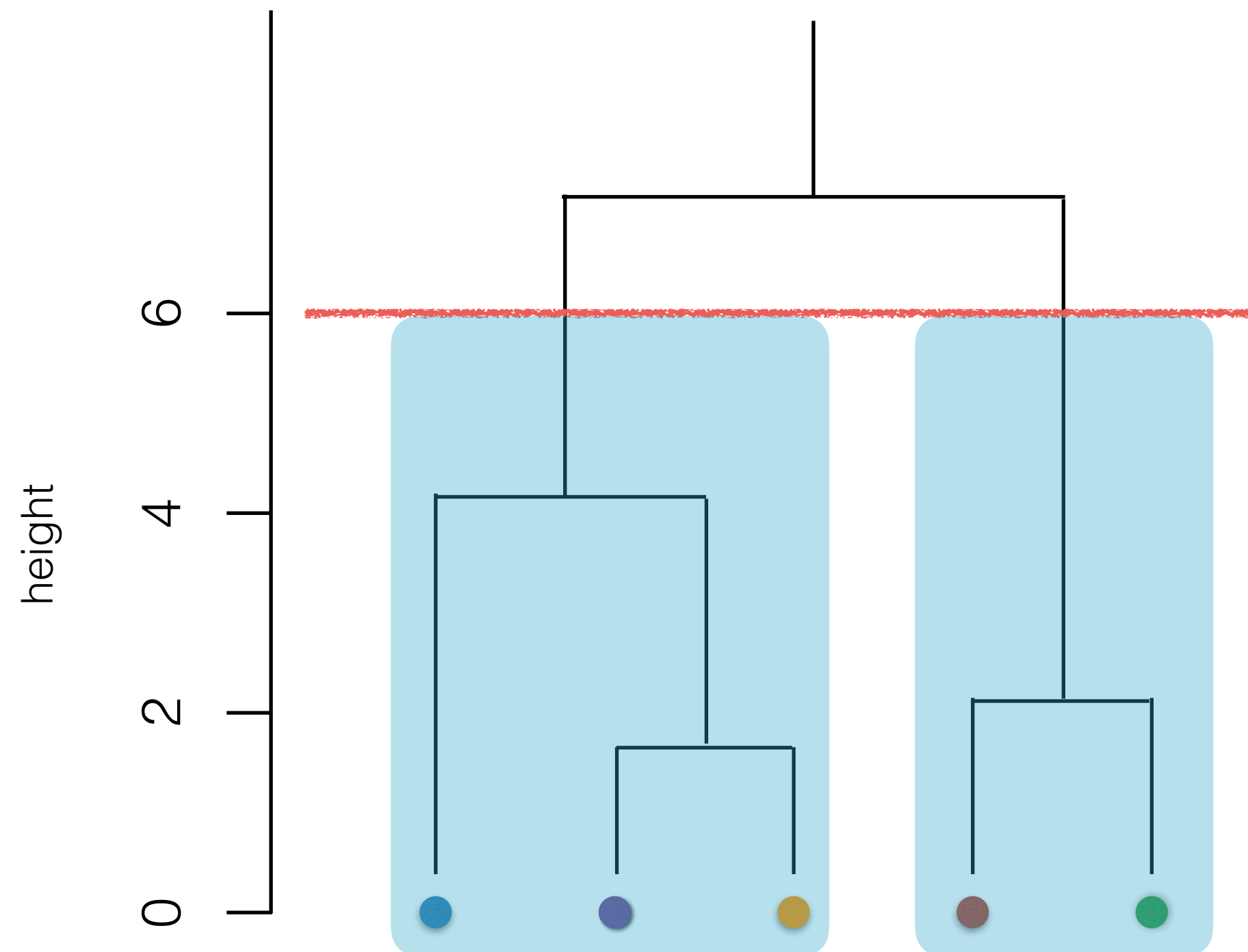
# Dendrogram

- Tree shaped structure used to interpret hierarchical clustering models



# Dendrogram plotting in R

```
> # Draws a dendrogram  
> plot(hclust.out)  
> abline(h = 6, col = "red")
```



# Tree 'cutting' in R

- Need to cut the tree to get cluster assignments

```
> # Cut by height h
> cutree(hclust.out, h = 6)
[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 3 3 3 3 3 3
[32] 3 3 3 3 4 4 4 4 4 4 4 4 4 4 2 4 2 4 4

> # Cut by number of clusters k
> cutree(hclust.out, k = 2)
[1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2
[32] 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```



UNSUPERVISED LEARNING IN R

**Let's practice!**



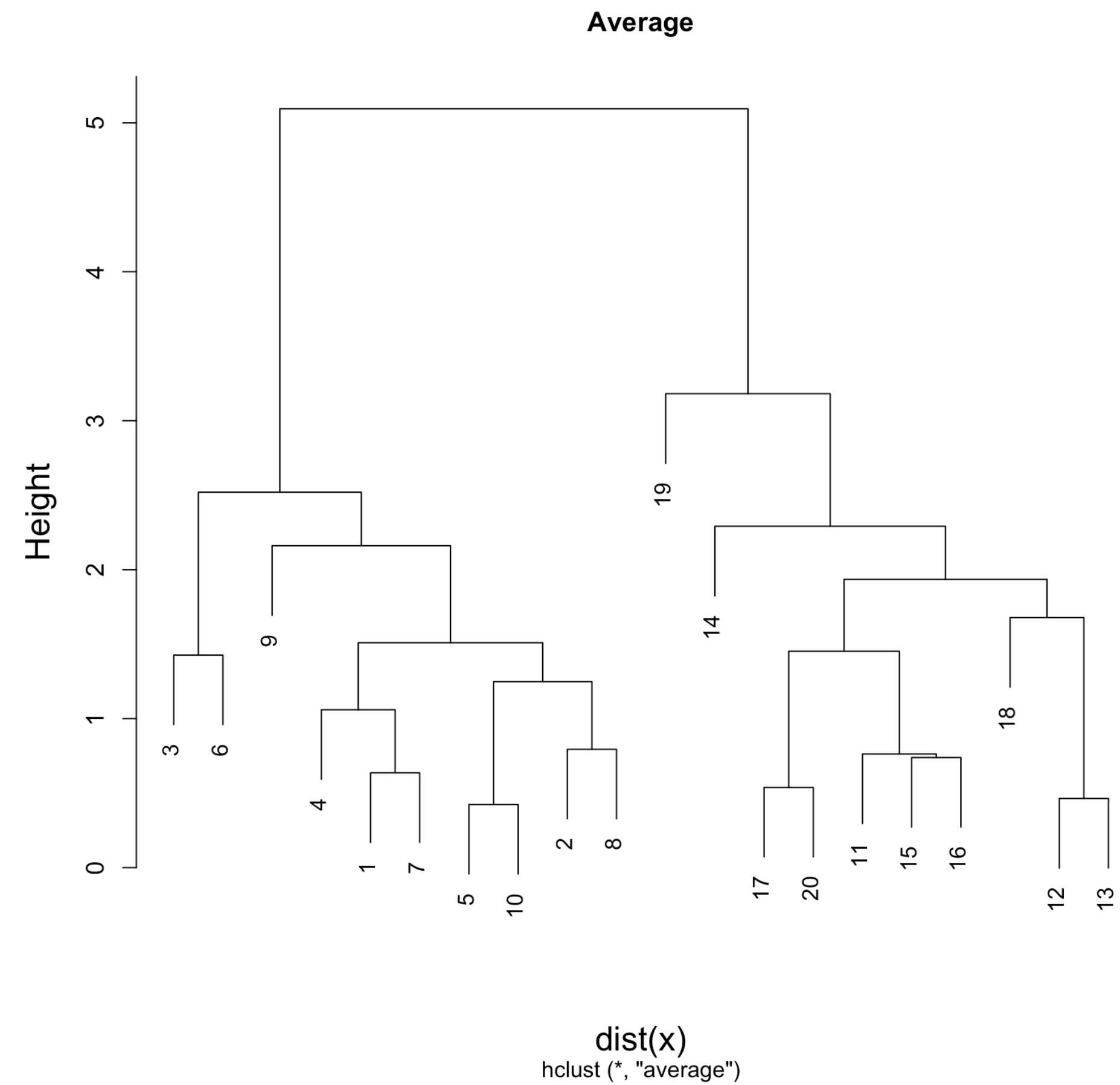
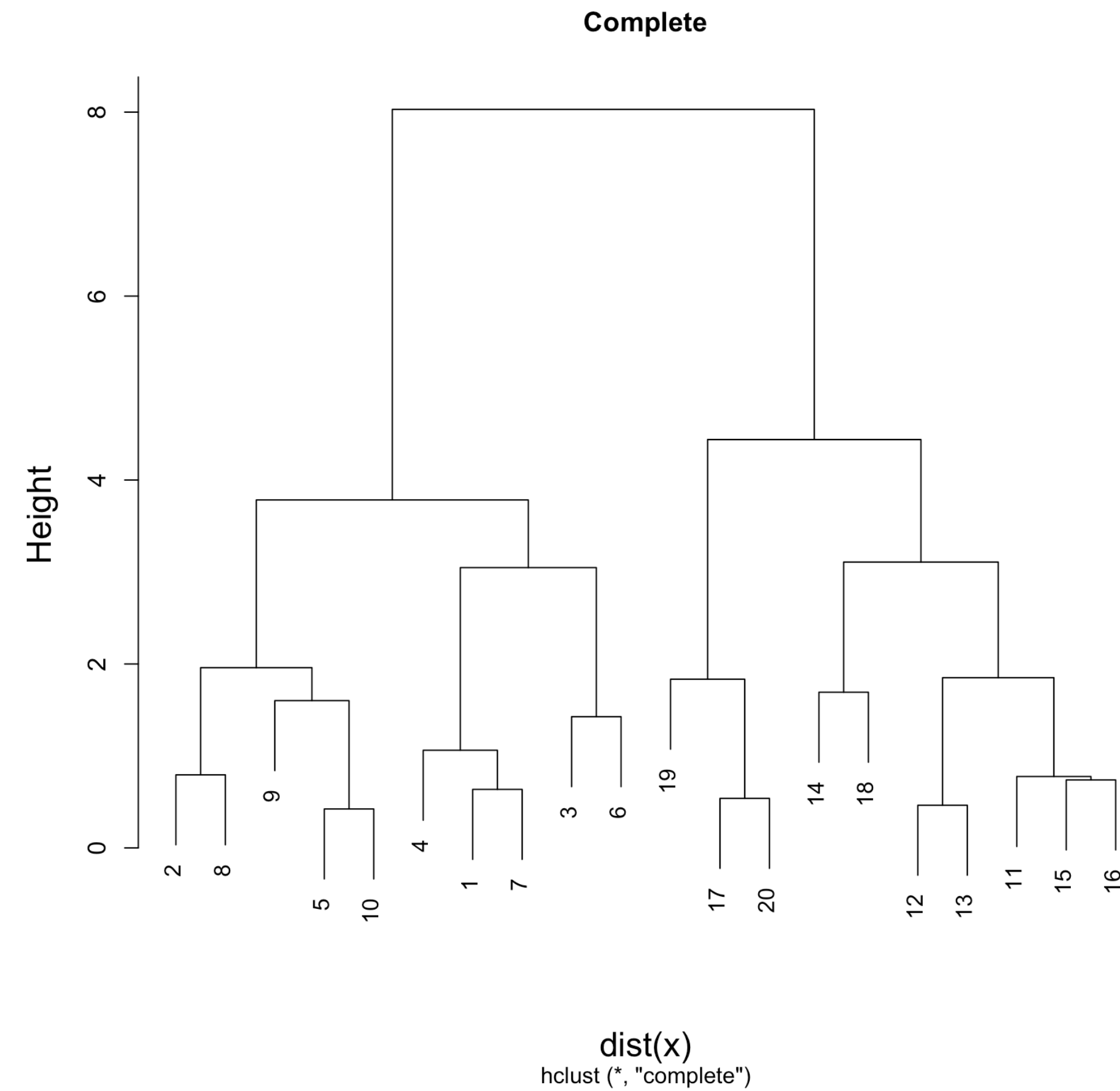
UNSUPERVISED LEARNING IN R

# **Clustering linkage and practical matters**

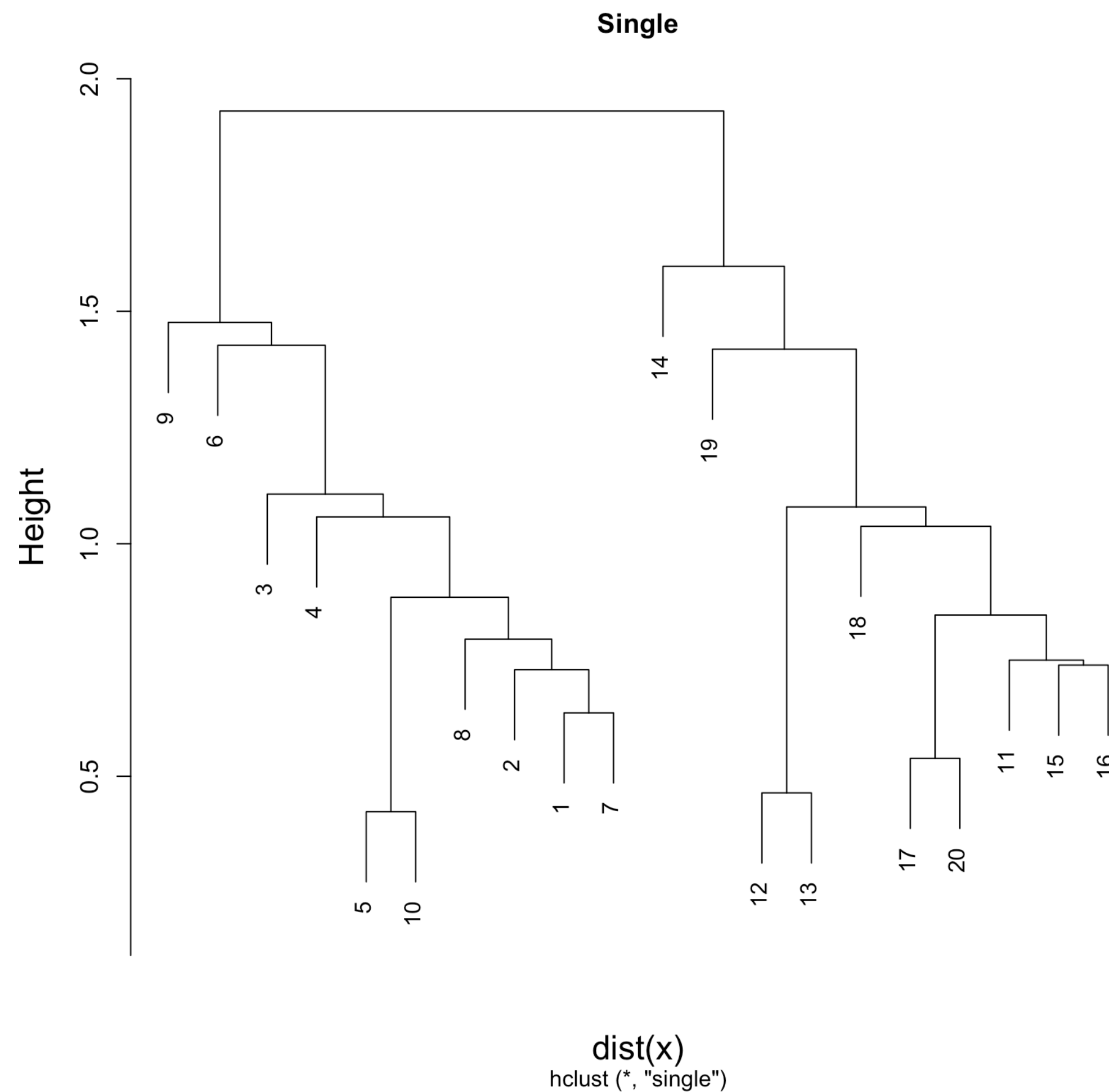
# Linking clusters in hierarchical clustering

- How is distance between clusters determined? Rules?
- Four methods to determine which cluster should be linked
  - *Complete*: pairwise similarity between **all observations** in cluster 1 and cluster 2, and uses **largest of similarities**
  - *Single*: same as above but uses **smallest of similarities**
  - *Average*: same as above but uses **average of similarities**
  - *Centroid*: finds **centroid** of cluster 1 and **centroid** of cluster 2, and uses **similarity between two centroids**

# Linking methods: complete and average

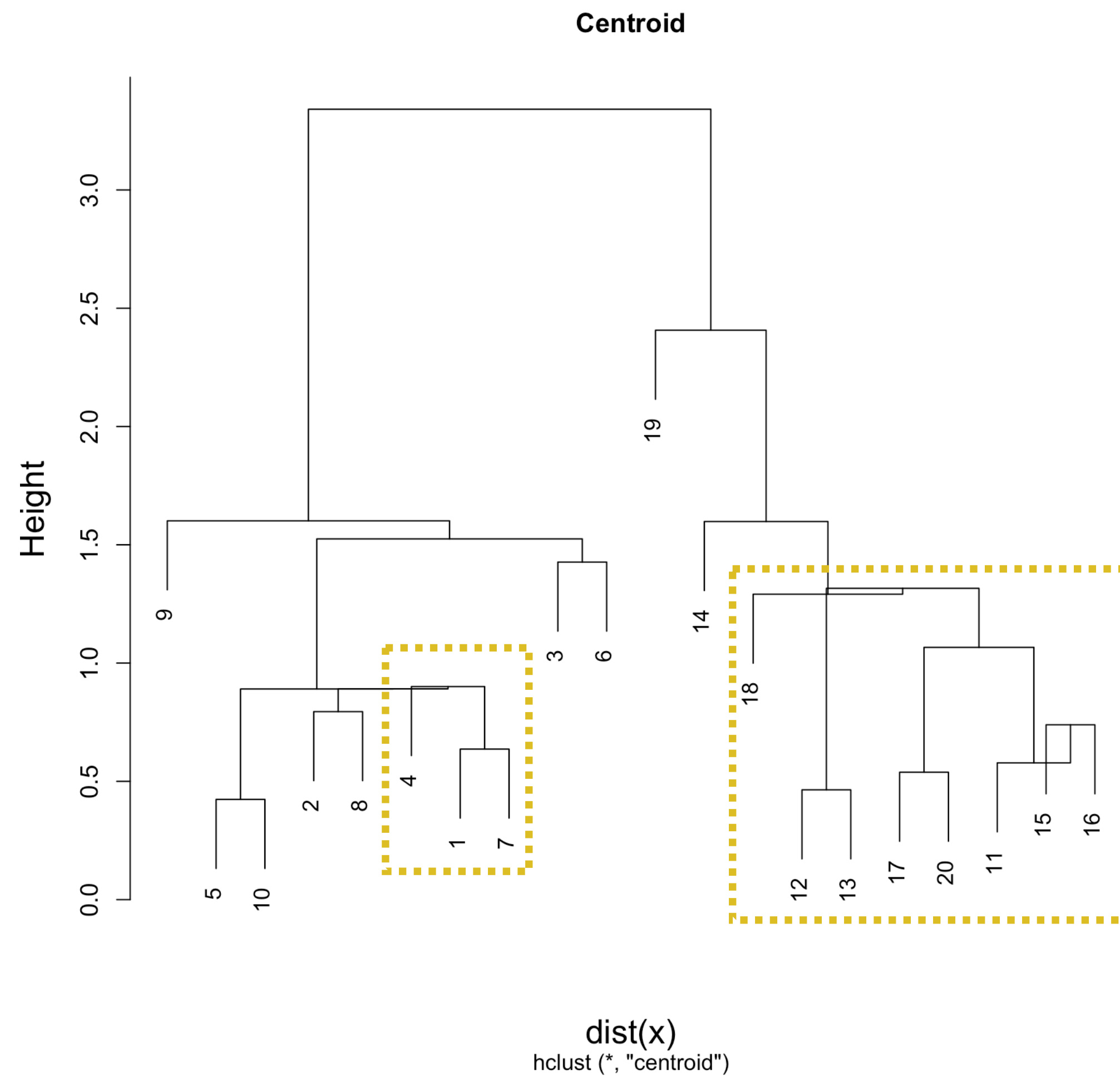


# Linking method: single





# Linking method: centroid



# Linkage in R

```
> # Fitting hierarchical clustering models using different methods
> hclust.complete <- hclust(d, method = "complete")
> hclust.average <- hclust(d, method = "average")
> hclust.single <- hclust(d, method = "single")
```

# Practical matters

- Data on different scales can cause undesirable results in clustering methods
- Solution is to scale data so that features have same mean and standard deviation
  - Subtract mean of a feature from all observations
  - Divide each feature by the standard deviation of the feature
  - Normalized features have a mean of zero and a standard deviation of one

# Practical matters

```
> # Check if scaling is necessary
> colMeans(x) x is a data matrix
[1] -0.1337828  0.0594019

> apply(x, 2, sd)
[1] 1.974376 2.112357

> # Produce new matrix with columns of mean of 0 and sd of 1
> scaled_x <- scale(x)

> colMeans(scaled_x)
[1] 2.775558e-17 3.330669e-17

> apply(scaled_x, 2, sd)
[1] 1 1
```

**Normalized features have mean of 0  
and standard deviation of 1**



UNSUPERVISED LEARNING IN R

**Let's practice!**



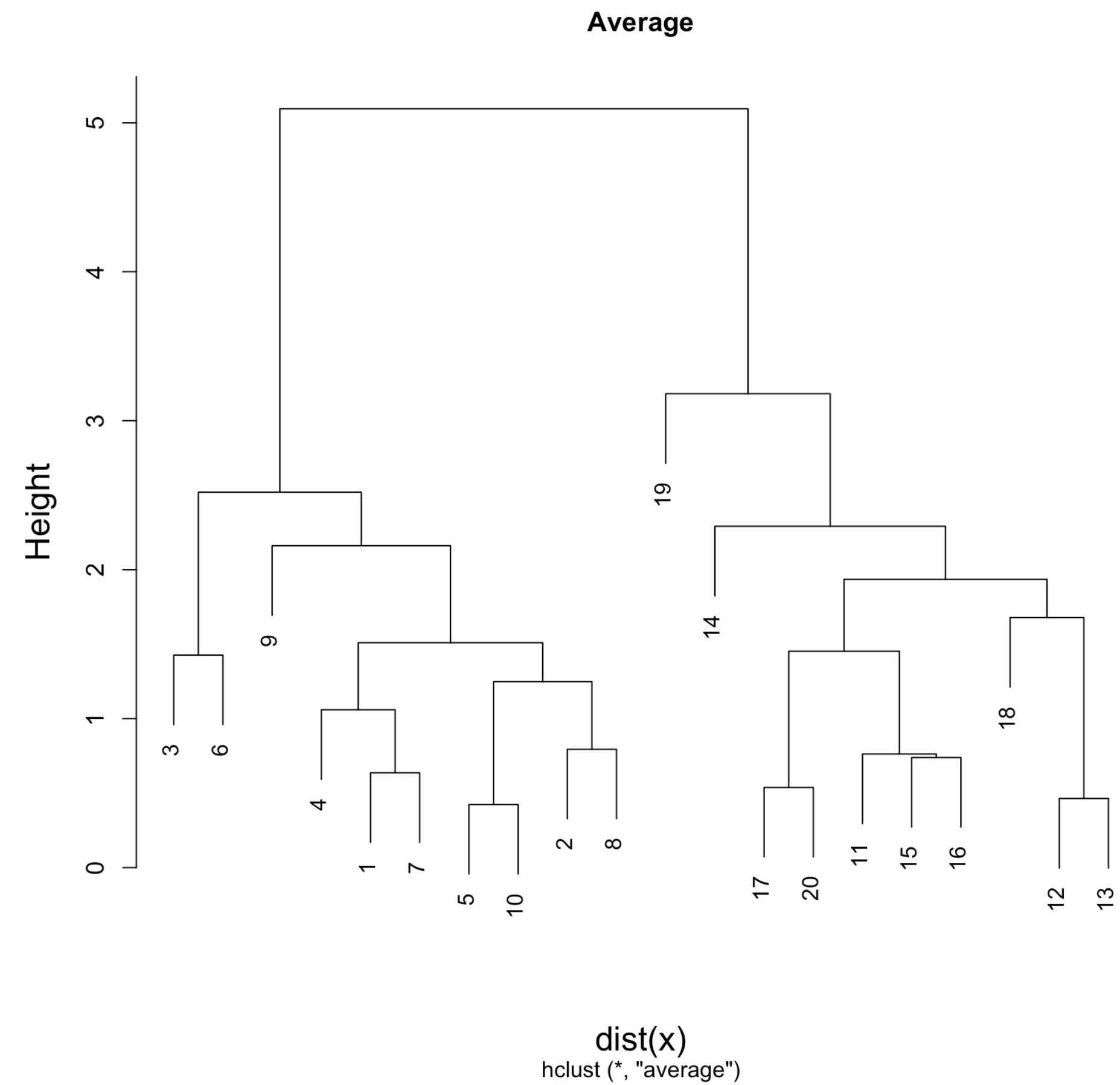
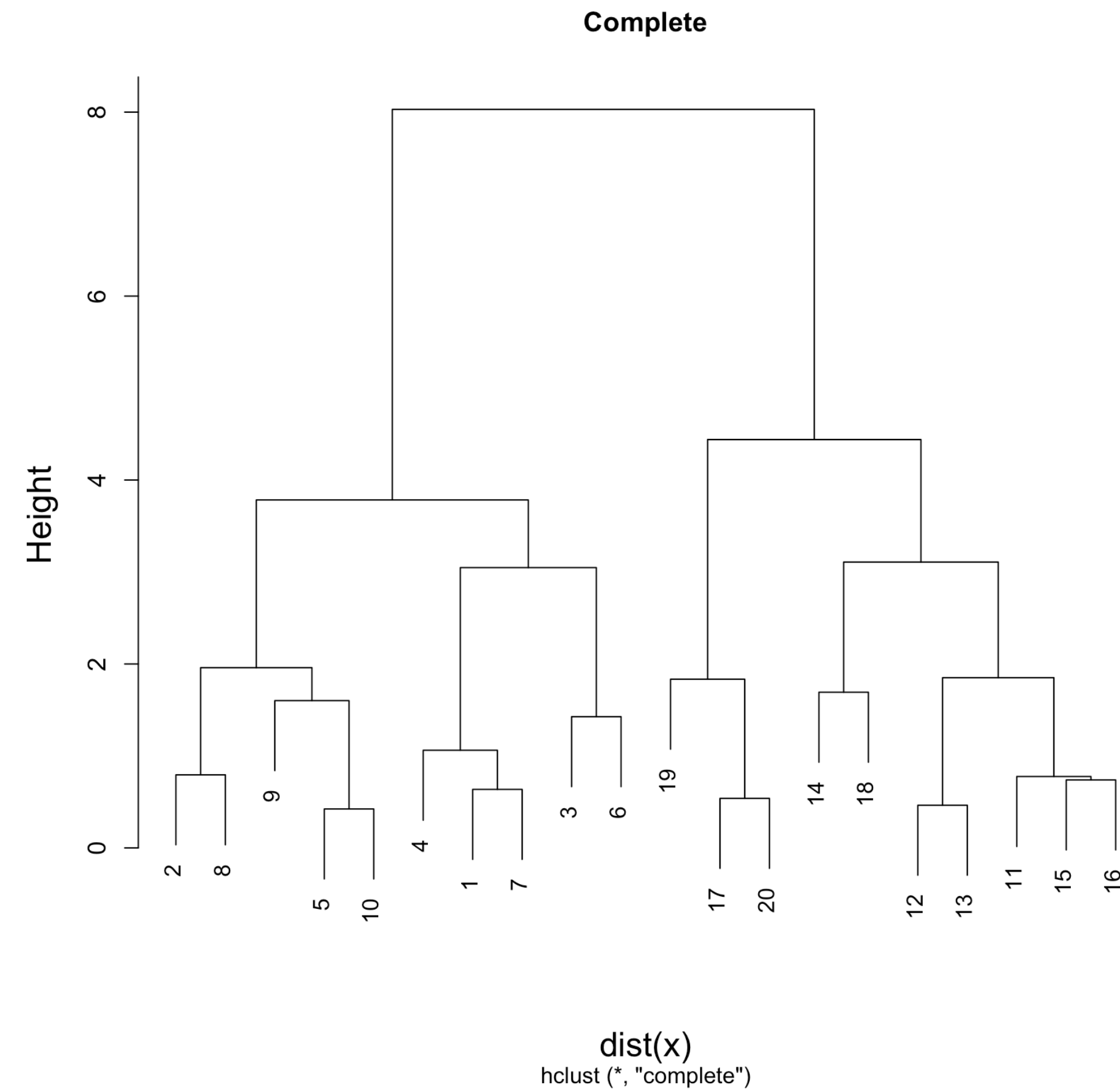
UNSUPERVISED LEARNING IN R

# **Review of hierarchical clustering**

# Hierarchical clustering review

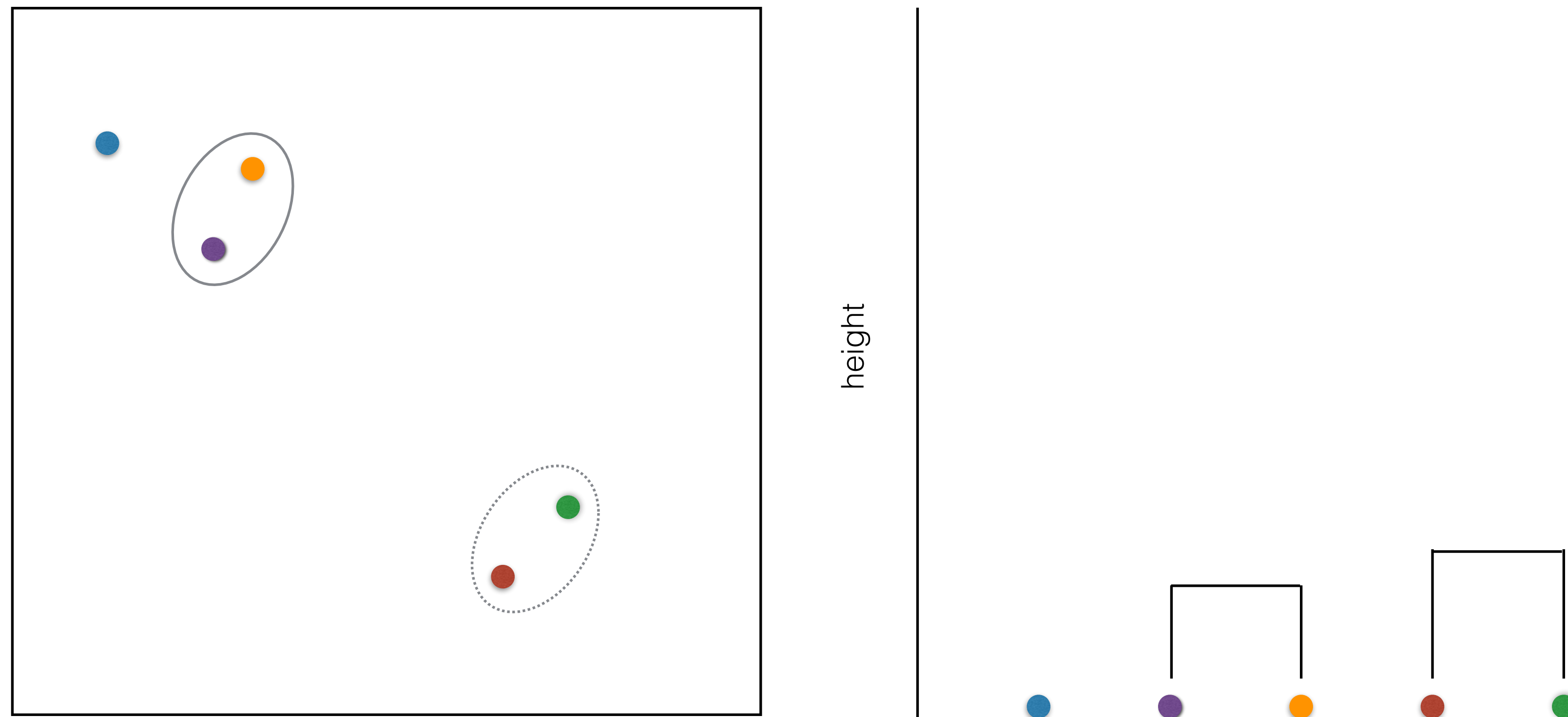
```
> # Fitting various hierarchical clustering models
> hclust.complete <- hclust(d, method = "complete")
> hclust.average <- hclust(d, method = "average")
> hclust.single <- hclust(d, method = "single")
```

# Linking methods: complete and average

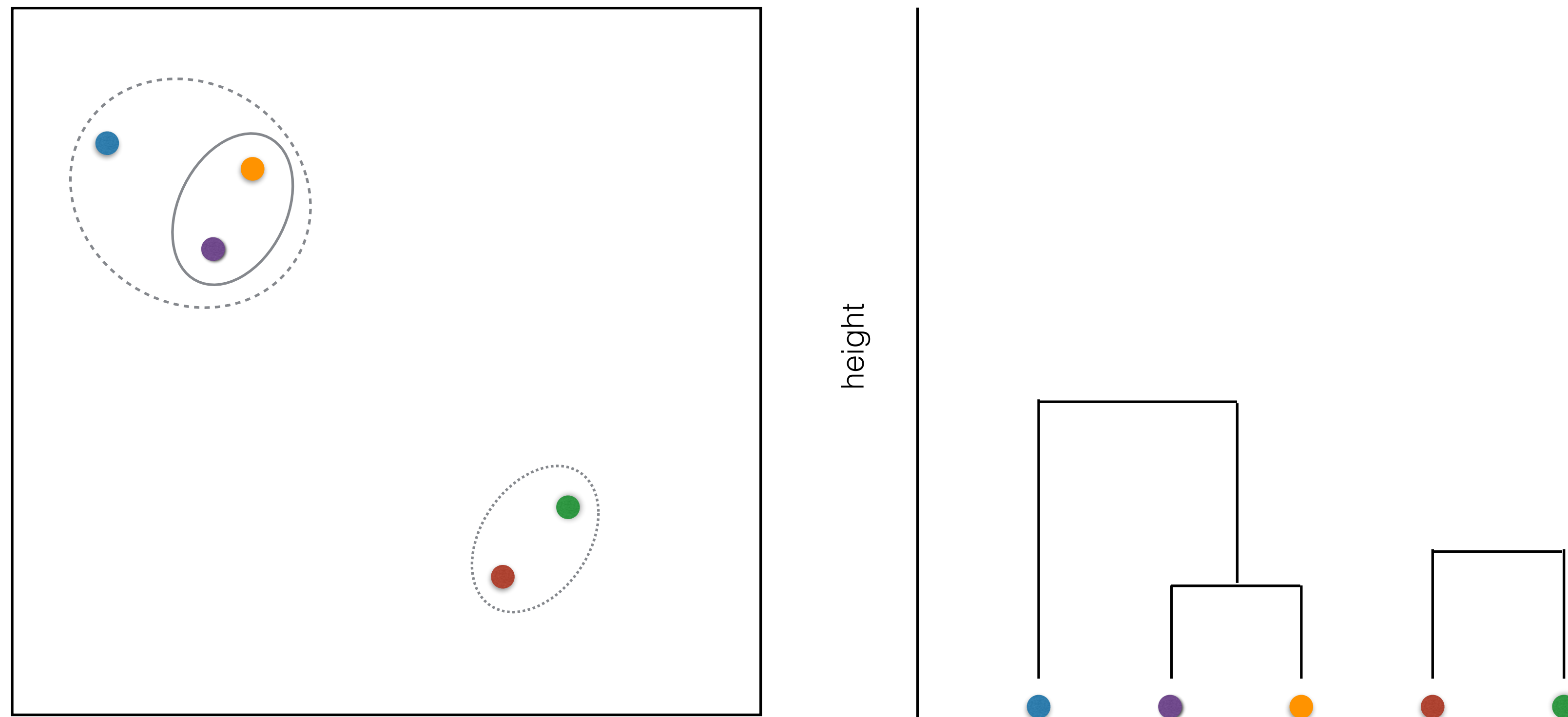




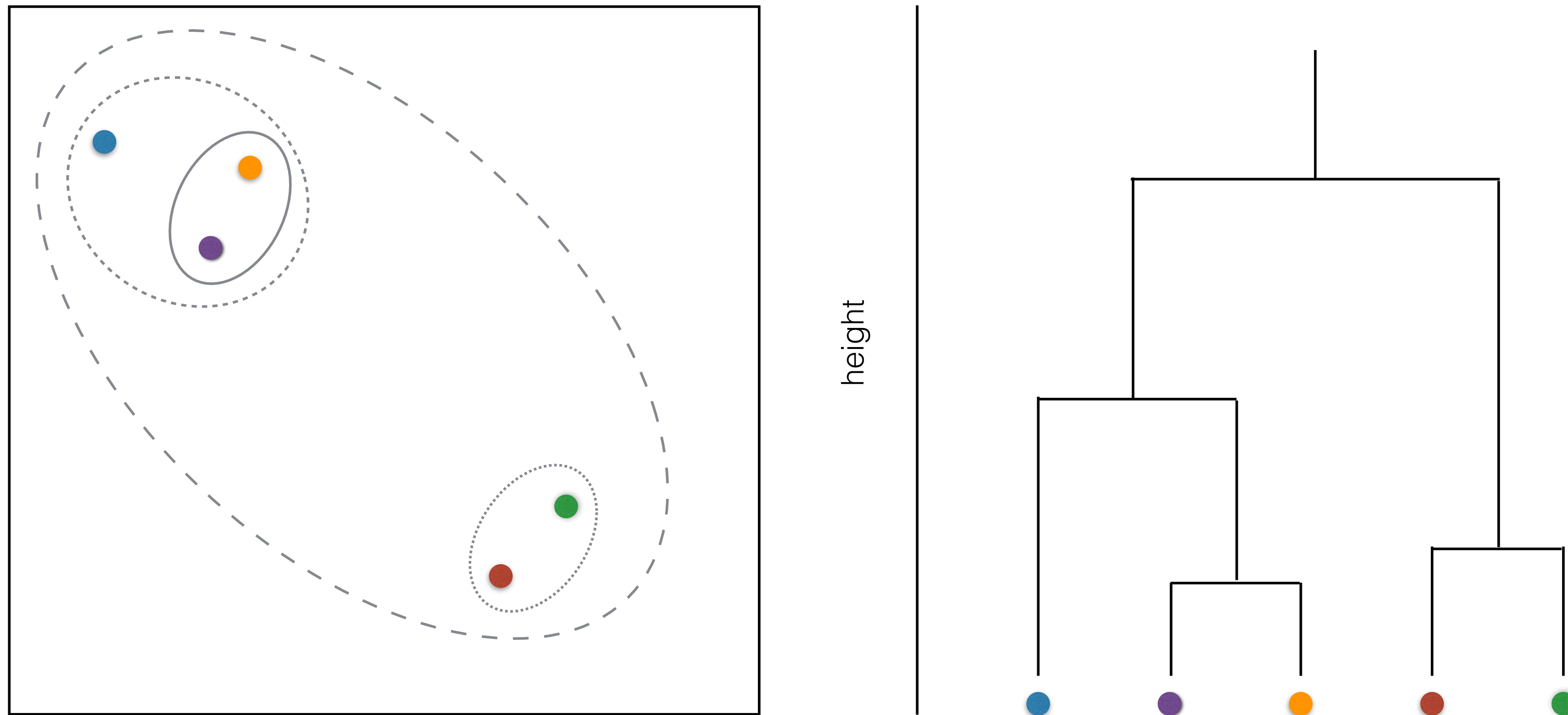
# Hierarchical clustering review



# Hierarchical clustering review

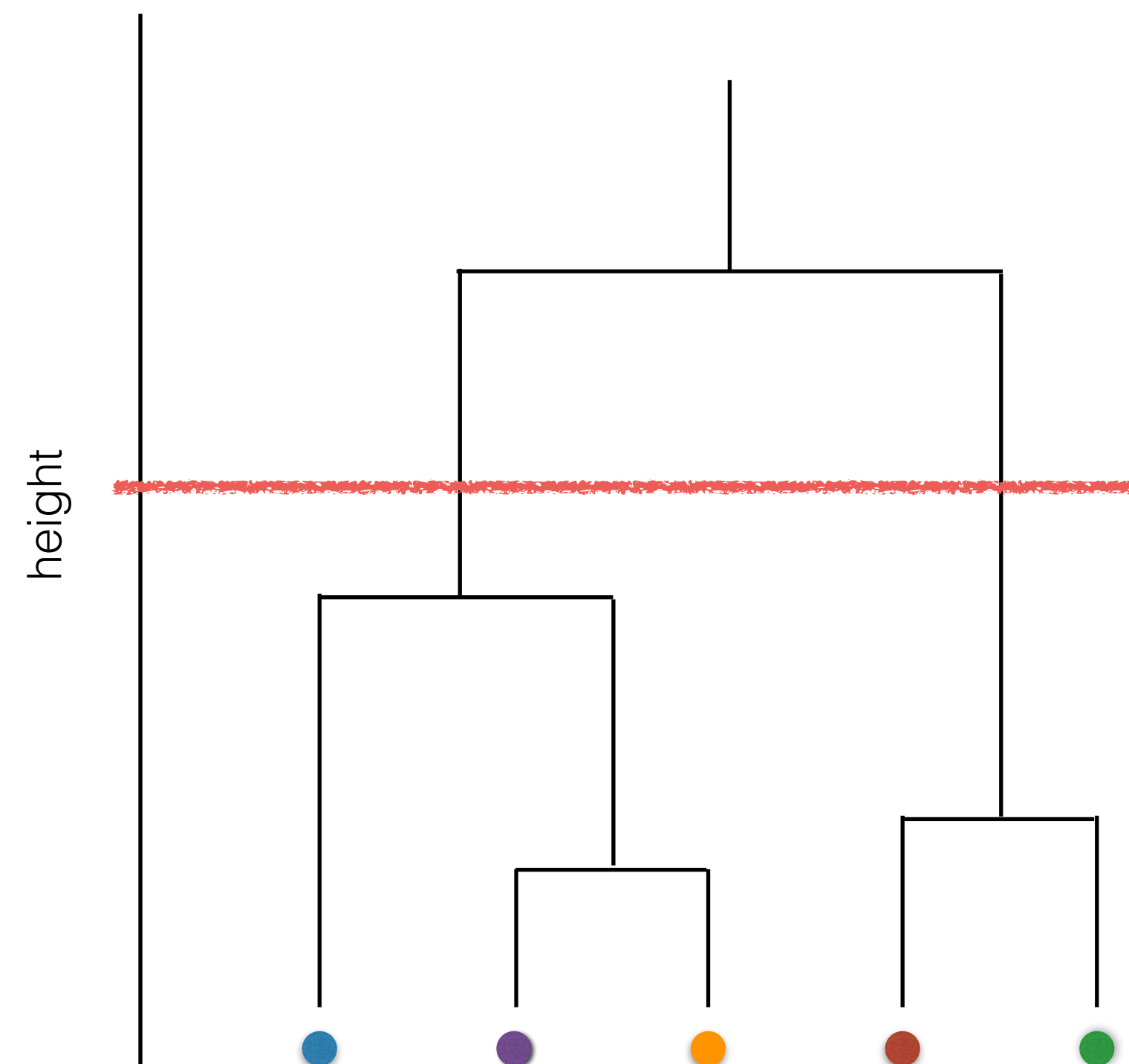
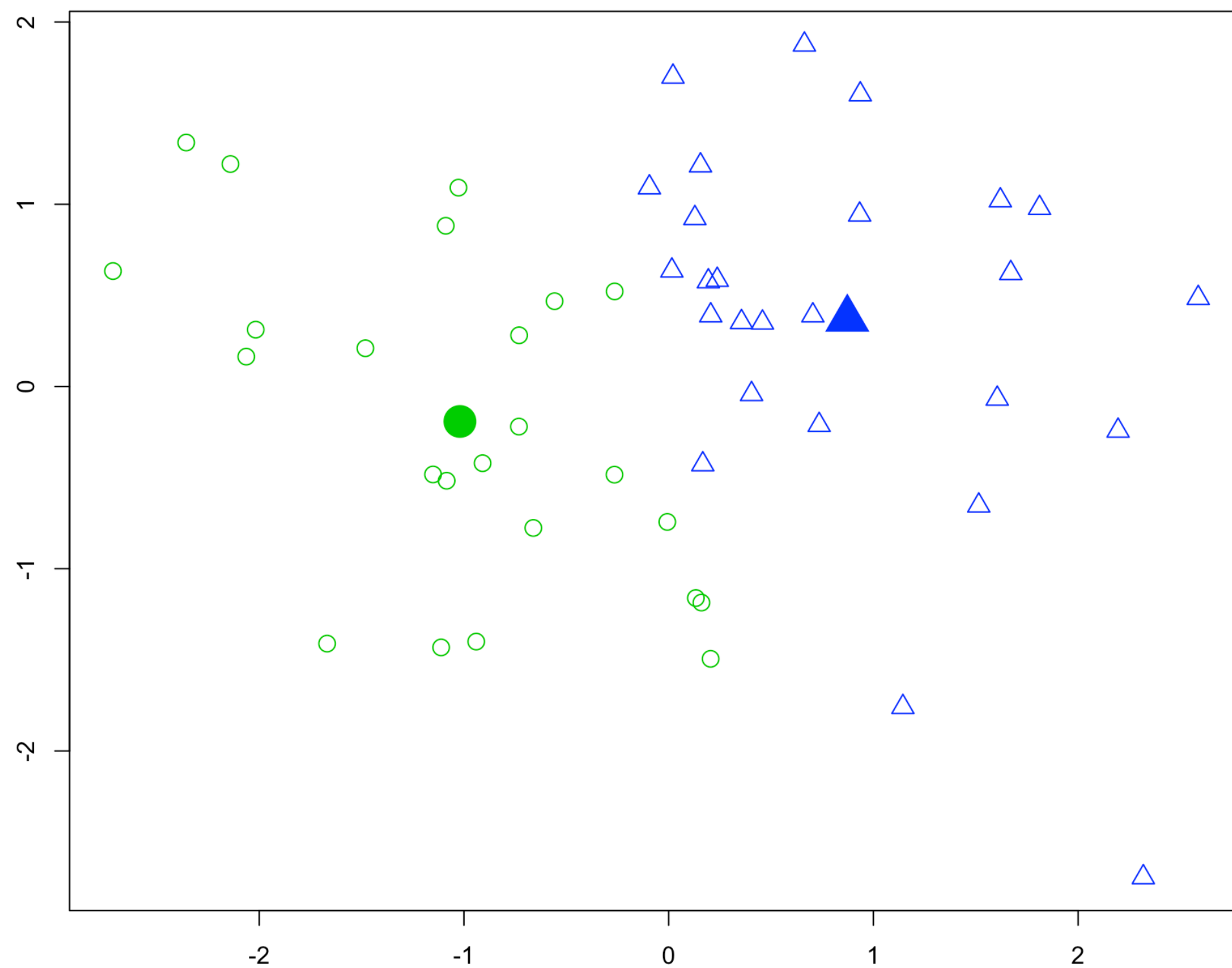


# Hierarchical clustering review



# Hierarchical clustering review

Iteration 5



# Hierarchical clustering review

```
> # Scale the data
> pokemon.scaled <- scale(pokemon)

> # Create hierarchical and k-means clustering models
> hclust.pokemon <- hclust(dist(pokemon.scaled), method = "complete")
> km.pokemon <- kmeans(pokemon.scaled, centers = 3,
                        nstart = 20, iter.max = 50)

> # Compare results of the models
> cut.pokemon <- cutree(hclust.pokemon, k = 3)
> table(km.pokemon$cluster, cut.pokemon)
  cut.pokemon
      1      2      3
1 242      1      0
2 342      1      0
3 204      9      1
```



UNSUPERVISED LEARNING IN R

**Let's practice!**