Lecture 1:
Clustering and Segmentation
(It's all about distance)

Dr Matt Wheeler

What makes things different?

The genus, thus, signifies indeterminately everything that is in the species; it does not signify the matter alone. Similarly, the difference, too, signifies everything in the species, and not the form alone; the definition, too, signifies the whole, and so does the species, but in diverse ways.
-St. Thomas Aquinas, *On Being and Essence*

Things are exist, and have essential differences; we can group them! Let's Learn!

Things are pointless, and all groupings are arbitrary. Class is over! Go to Starbucks and have pseudointellectual conversations!

The diversity of things, their individuality, were only an appearance, a veneer. This veneer had melted, leaving soft, monstrous masses, all in disorder—naked, in a frightful, obscene nakedness. I kept myself from making the slightest movement, but I didn't need to move in order to. - John Paul Sartre ,*Nausea*

- All clustering is based upon the fundamental assumption that there are difference (qualitative and quantitative) between things.

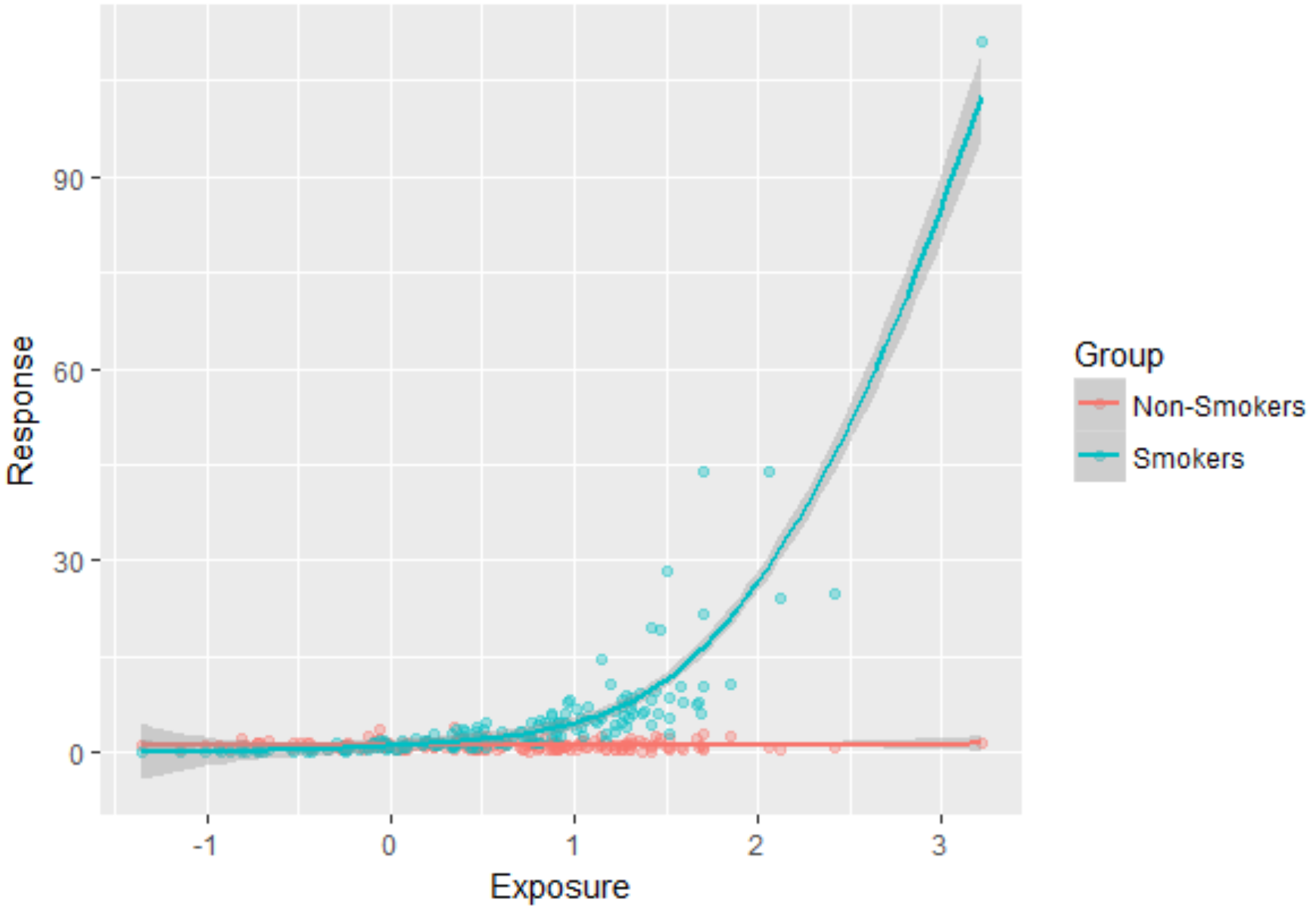
- **Our goal:** Develop methods of comparing things and then finding out what is “similar” and what is different.

- Comparison requires us to define “**distance.**”

Distance:

Simple Thought Experiment: Is 1 close to 2?

It depends!


Response
90
60
30
0
-1
0
1
2
3
Exposure
Group
Non-Smokers
Smokers

Types of “measurements”

- Initially we are going to start off with continuous numbers, so we can build our concepts.
- We will go into text and “function” data. Dichotomous and Categorical data will be used too.
- Exercise caution when combining data types!!!

- Differences are determinate on a large number of factors, some measured, some qualitative, but the key is we want to figure out what is different, and group them.

If I ask my teenage daughters which Chris is better, am I going to get a different response?

Comparisons come from numerical distances!

My daughters can't answer the previous question unless they have some "hypothetical Chris ordering."

All orderings or distances $d(x, y)$ have the following properties (from a mathematical perspective):

1. $d(x, y) > 0$	**Non-negativity**
2. $d(x, y) = 0 \leftrightarrow x = y$	**Identity**
3. $d(x, y) = d(y, x)$	**Symmetry**
4. $d(x, z) \leq d(x, y) + d(y, z)$	**Triangle Inequality**

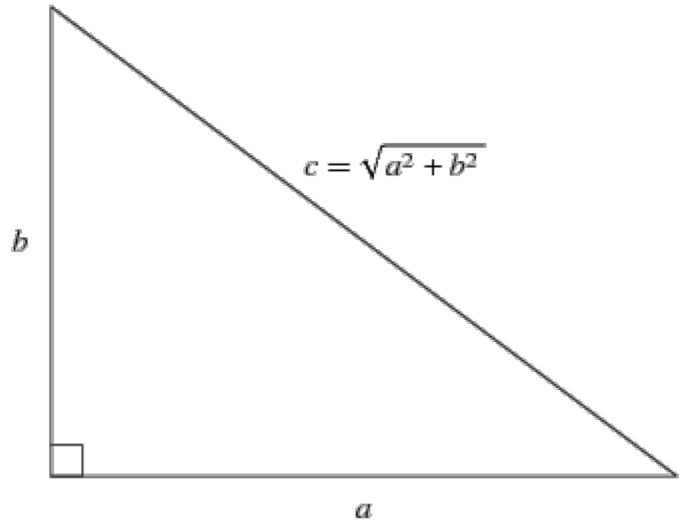
Common Distances:

Euclidean:

$\boldsymbol{X} \subset \mathbb{R}^n$ We have a real vector of n elements, then for $x, y \in \boldsymbol{X}$ we have:

$$d(x, y) = \sqrt{(x_1 - y_1)^2 + \cdots + (x_n - y_n)^2}$$

Note: Sometimes called the L2 Distance



Hamming Distance

- $\boldsymbol{X} \subset \mathbb{W}^n$ We have a vector of whole numbers of n elements, then for $x, y \in \boldsymbol{X}$ one has

$$\mathrm{d}(\mathrm{x}, \mathrm{y}) = \sum_i x_i = y_i$$

> Examples:

1. $\mathrm{x} =$ '101110' $\mathrm{y} =$ '111001' $\mathrm{d}(\mathrm{x}, \mathrm{y}) = 4$
2. $\mathrm{x} =$ 'Sticky' $\mathrm{y} =$ '_picky' $\mathrm{d}(\mathrm{x}, \mathrm{y}) = 2$
3. $\mathrm{x} =$ '<3,102,5>' $\mathrm{y} =$ '<3,120,6> $\mathrm{d}(\mathrm{x}, \mathrm{y}) = 2$

Manhattan Distance:

$\boldsymbol{X} \subset \mathbb{R}^n$ We have a real vector of n elements, then for $x, y \in \boldsymbol{X}$ we have:

Sometimes called L1 Distance

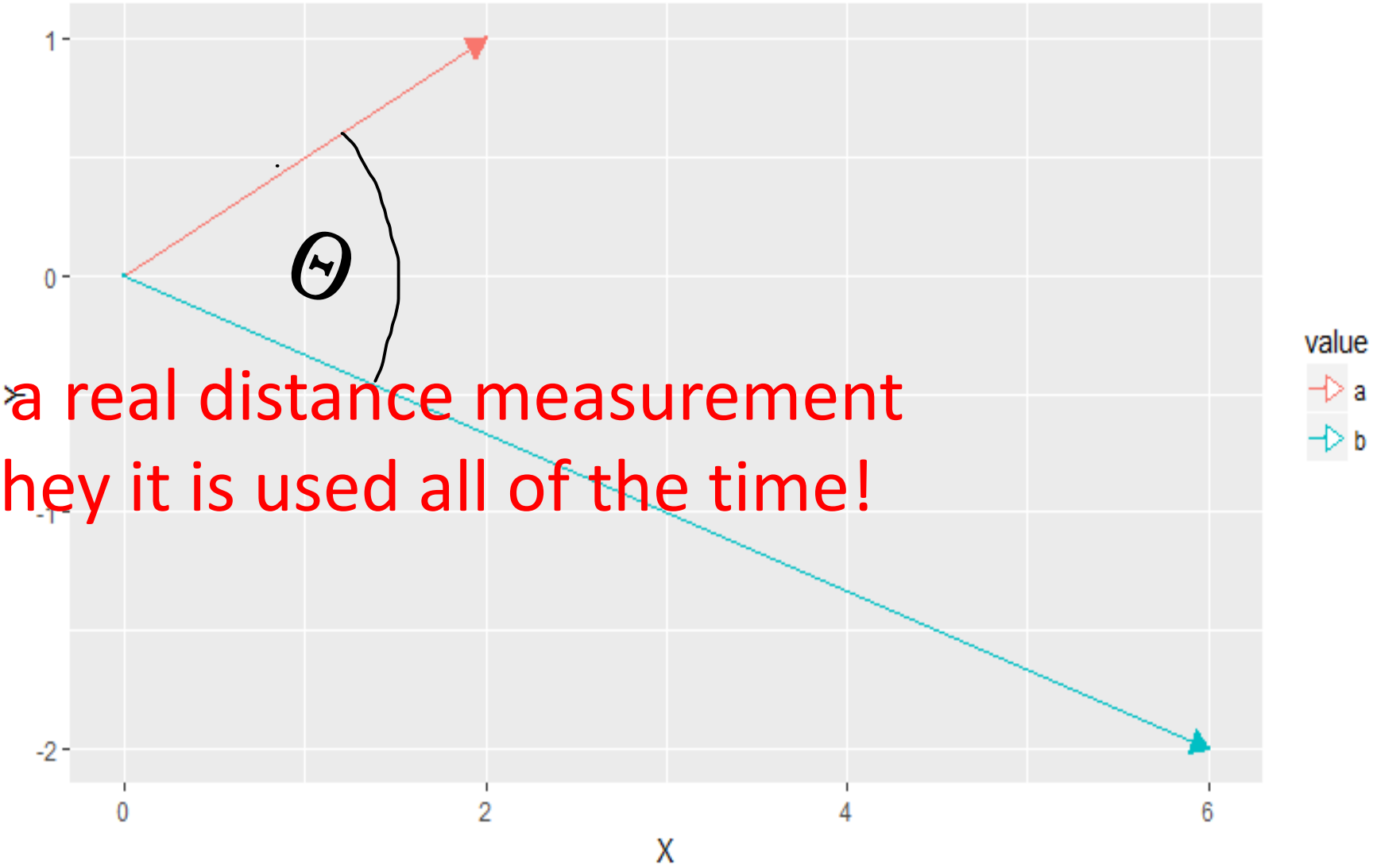
Cosine distance (used clustering text data)

$\boldsymbol{X} \subset \mathbb{R}^n$ We have a real vector of n elements, then for $x, y \in \boldsymbol{X}$ we have:

$$d(x, y) = \cos^{-1} \frac{\sum_i x_i y_i}{\sum_i x_i^2 \sum y_i^2}$$

Note: Measures the angular distance between two vectors. Think linear algebra and orthogonality. The numerator is the dot product. The denominator is the product of the squared Euclidean distance.

Cosine distance (cont.)



Not a real distance measurement
but hey it is used all of the time!

And many more…

The key is we need to think about what it means to be different in a particular domain. This will lead us to the distance metric, and that will then lead us to our ‘solution.’

Note: None of these metrics mentioned account for correlation between variables. We will get to that later, but the correlation can help tease out things!

How are we going to use distances to “group” objects?

Intuitively: The name of the game is simple: Things that are “close” are similar. Things that are far away are different. Group things that are close into the same bin. Measurement becomes important.

Hierarchical Clustering (Agglomerative Clustering)

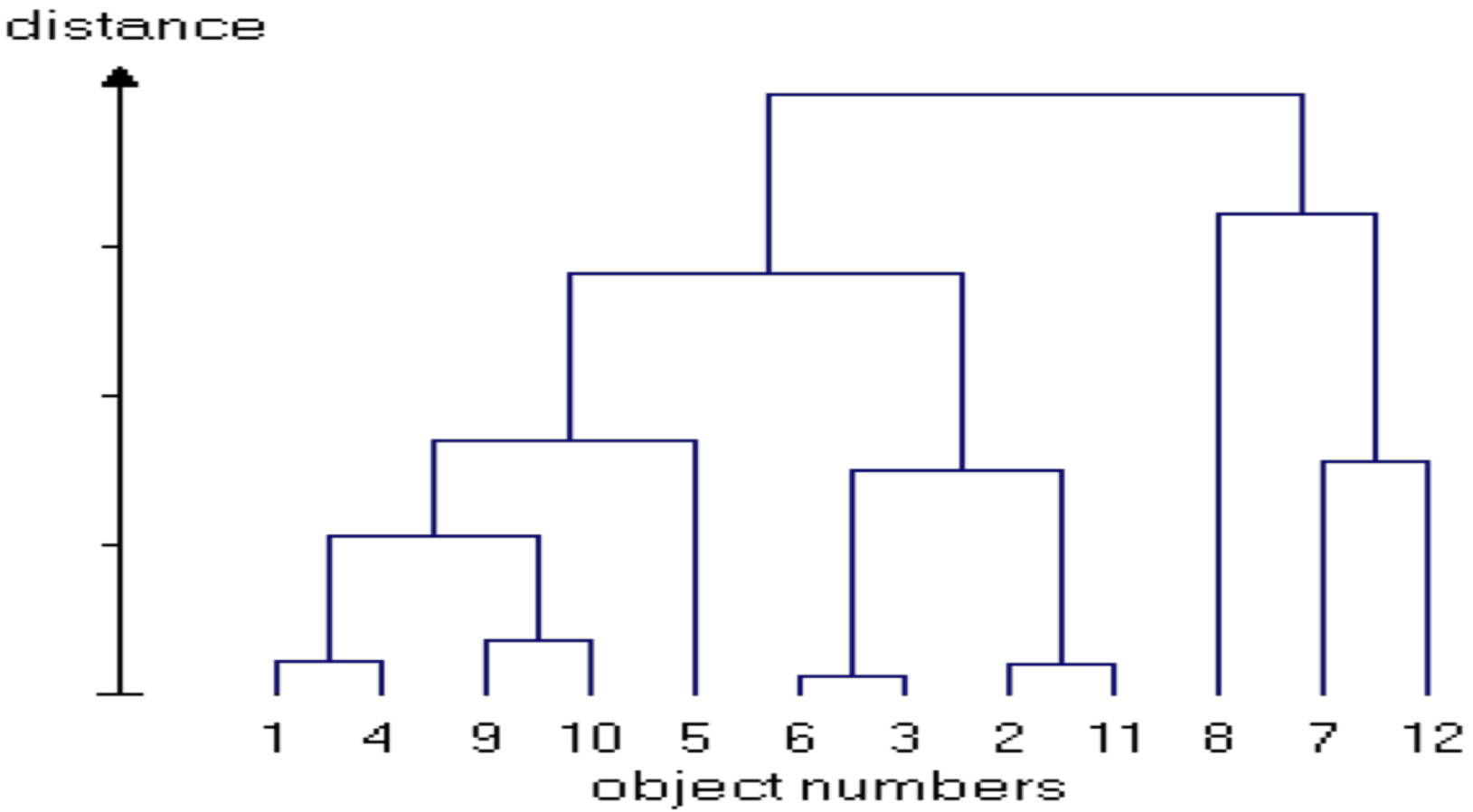
General Idea:

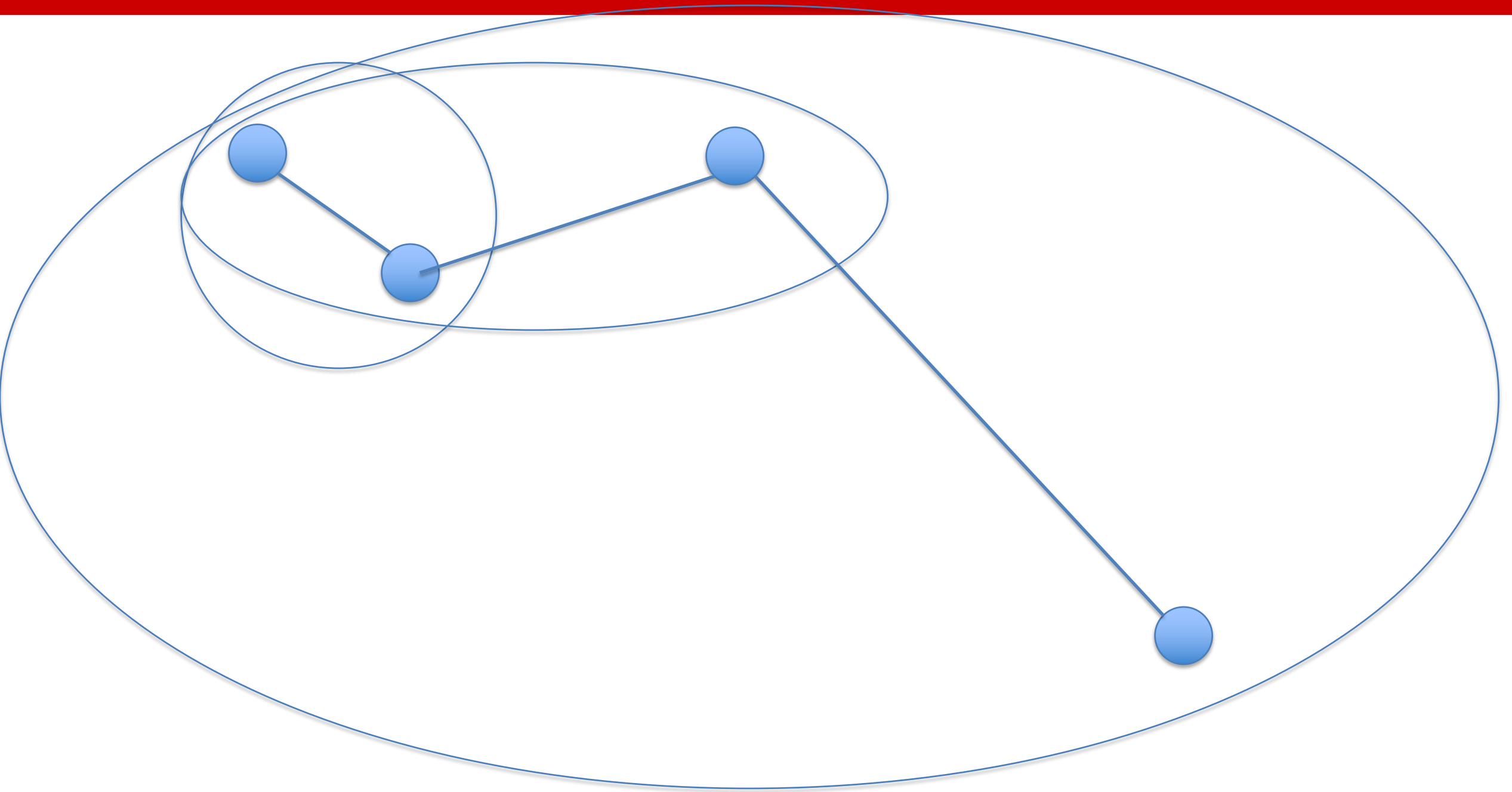
Everyone starts in their own cluster. Build groups based upon those objects that are more similar. Start small build up.

Algorithm:

```
Do Until No more clusters in S are available
        Find 2 most similar clusters (x,y) <- min_{x,y∈S} d(x,y)
        Create New Cluster  z = (x U y)
        Remove  x and y from S
        Insert z in S
Repeat
```

Dendrogram (One way to think of it)





When there are multiple “objects” in a cluster we have to define the function

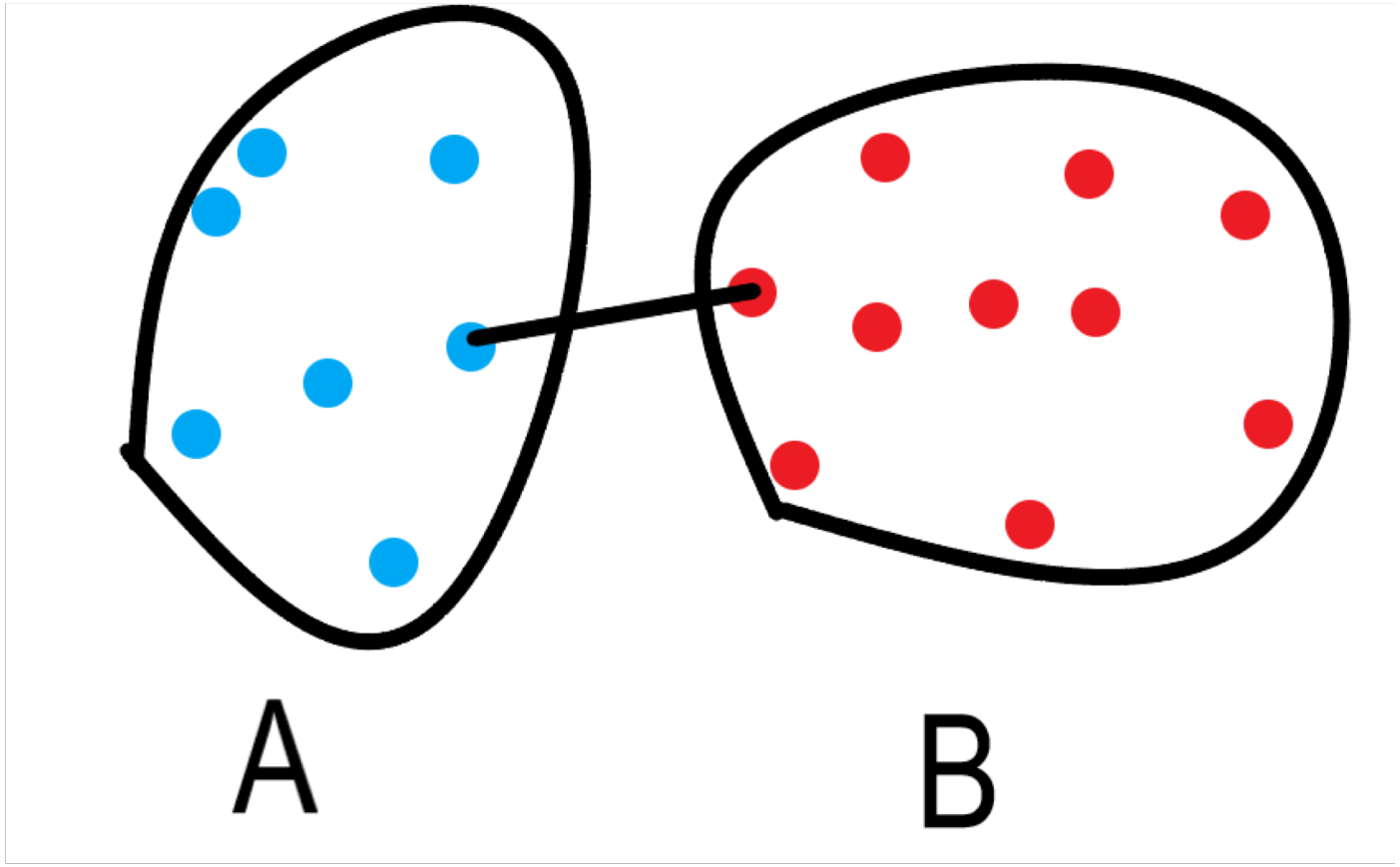
$$\min_{x,y \in S} d(x, y)$$

That is:

We have defined the distance between objects, but how do we define the distance between groups of objects to link groups?

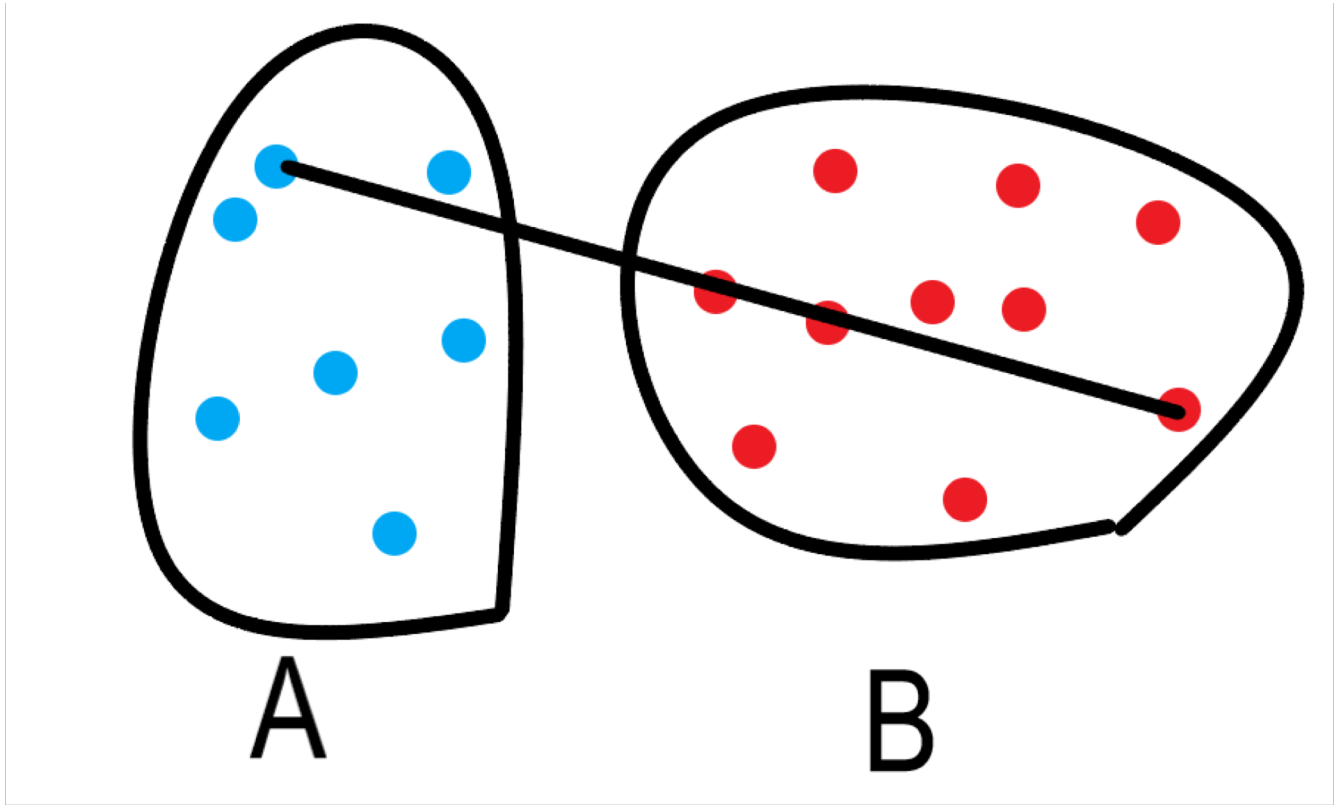
Single Linkage:

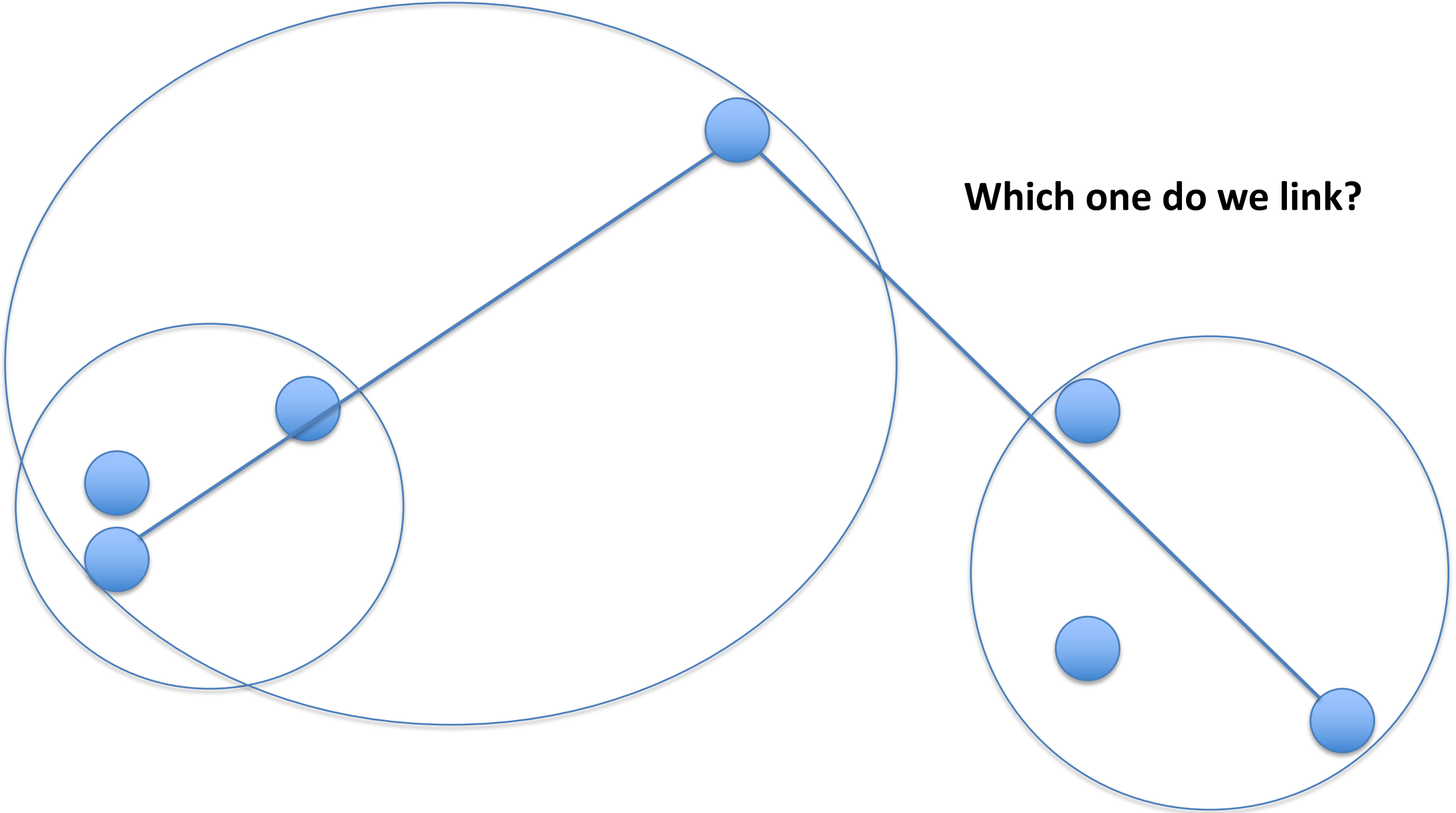
For all objects in group **A** and all objects in group **B** the distance between the minimum distance between any two objects in **A** and **B**.



Complete Linkage:

For all objects in group **A** and all objects in group **B** the distance between the maximum distance between any two objects in **A** and **B**.





Which one do we link?

Linkage:

Single and complete linkage are not the only definitions. There are many more. For more examples see the documentation of ?hclust in R.

The choice of linkage does impact your result!

Hierarchical Clustering in R

Let's start with a TOY problem that gives us very nice answers. Real life isn't so nice.

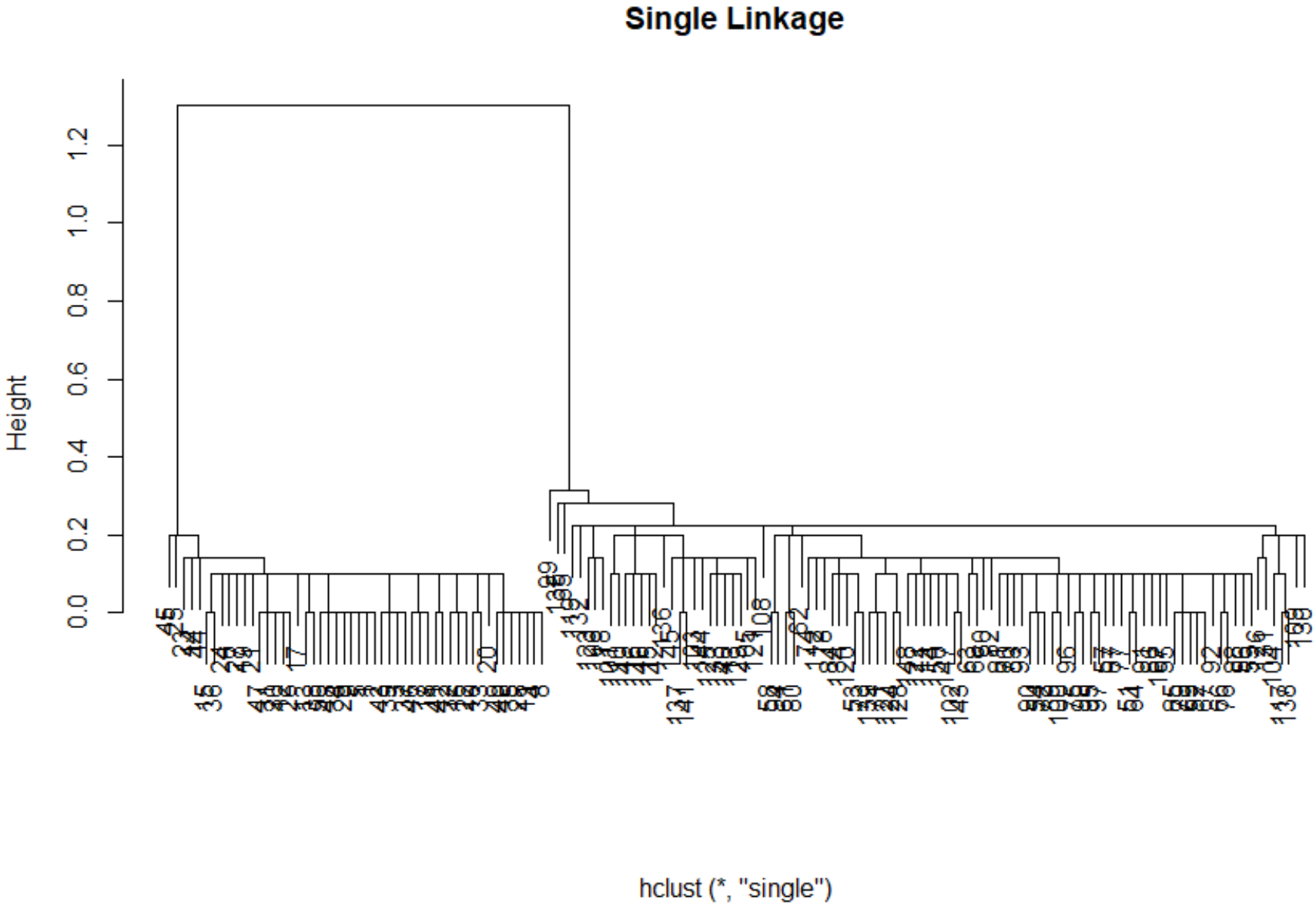
It is a classic dataset that measures features of different flowers, and is freely available in R

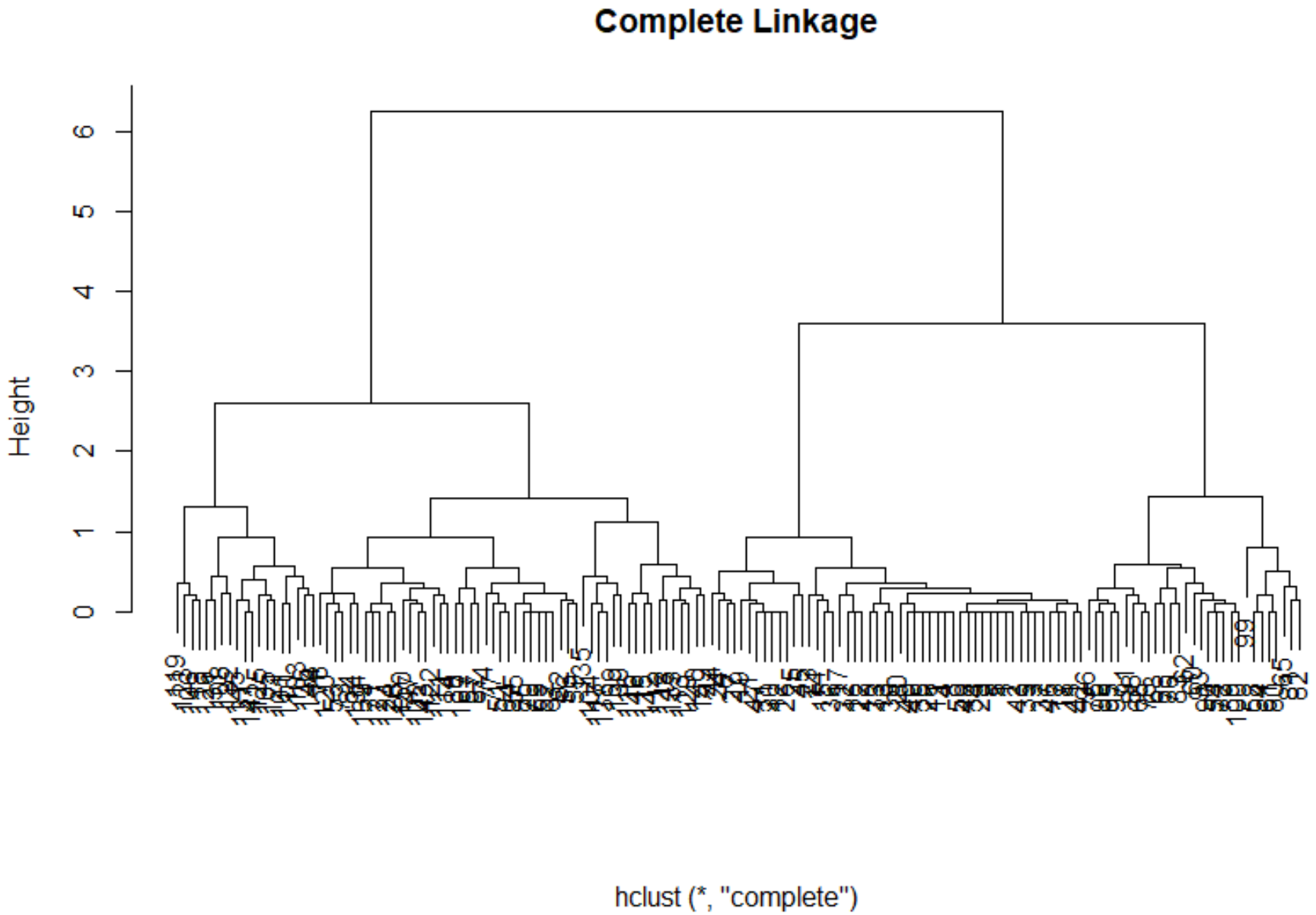
Sepal Length	Sepal Width	Petal Length	Petal Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa

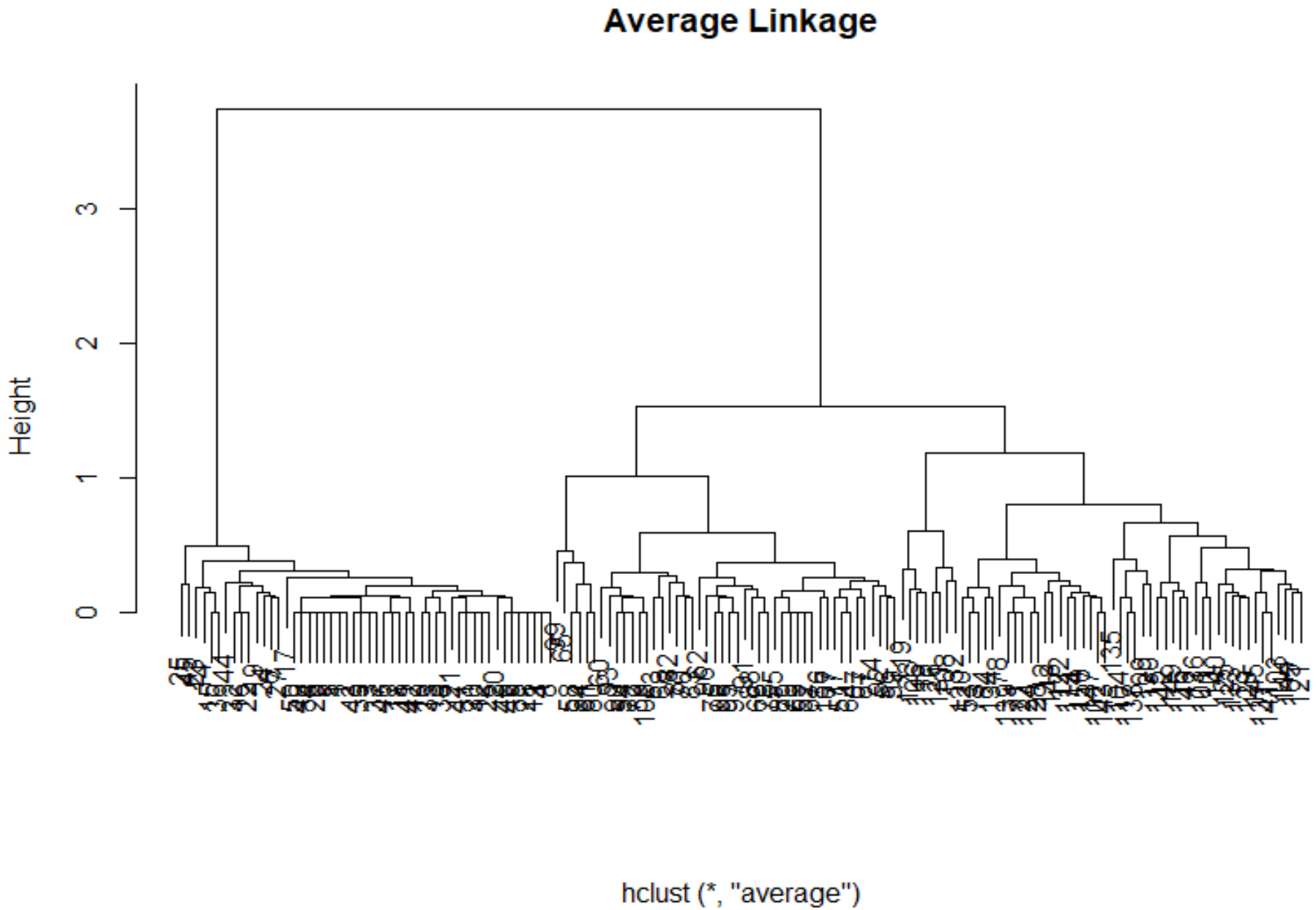
The type of linkage you use really matters!!!

```
library(datasets)  #This is where the dataset is found
library(ggplots2)

#Build 3 different hierarchical cluster trees
iris.clusters.single <- hclust(dist(iris[, 3:4]),method="single")
iris.clusters.complete <- hclust(dist(iris[, 3:4]),method="complete")
iris.clusters.centroid <- hclust(dist(iris[, 3:4]),method="average")
#Look at their dendrograms
plot(iris.clusters.single,main="Single Linkage",xlab="")
plot(iris.clusters.complete,main="Complete Linkage",xlab="")
plot(iris.clusters.centroid,main="Average Linkage",xlab="")
```


Single Linkage
Height
0.0
0.2
0.4
0.6
0.8
1.0
1.2
hclust (*, "single")


Complete Linkage
Height
0
1
2
3
4
5
6
hclust (*, "complete")


Average Linkage
Height
0
1
2
3
hclust (*, "average")

Hierarchical clustering doesn't actually cluster!!! You actually must choose the number of clusters!!

```
clusters_3 <- cutree(iris.clusters.complete , 3) # partition tree into 3 clusters
clusters_4 <- cutree(iris.clusters.complete , 4) # partition tree into 4 clusters

tail(cbind(clusters_3,clusters_4),n=10)          #compare two clusters last 10
                                                 #elements
```

Our clusters are really based upon how one chooses a cut point and linkage method.

But it also greatly depends upon what are inputs are.

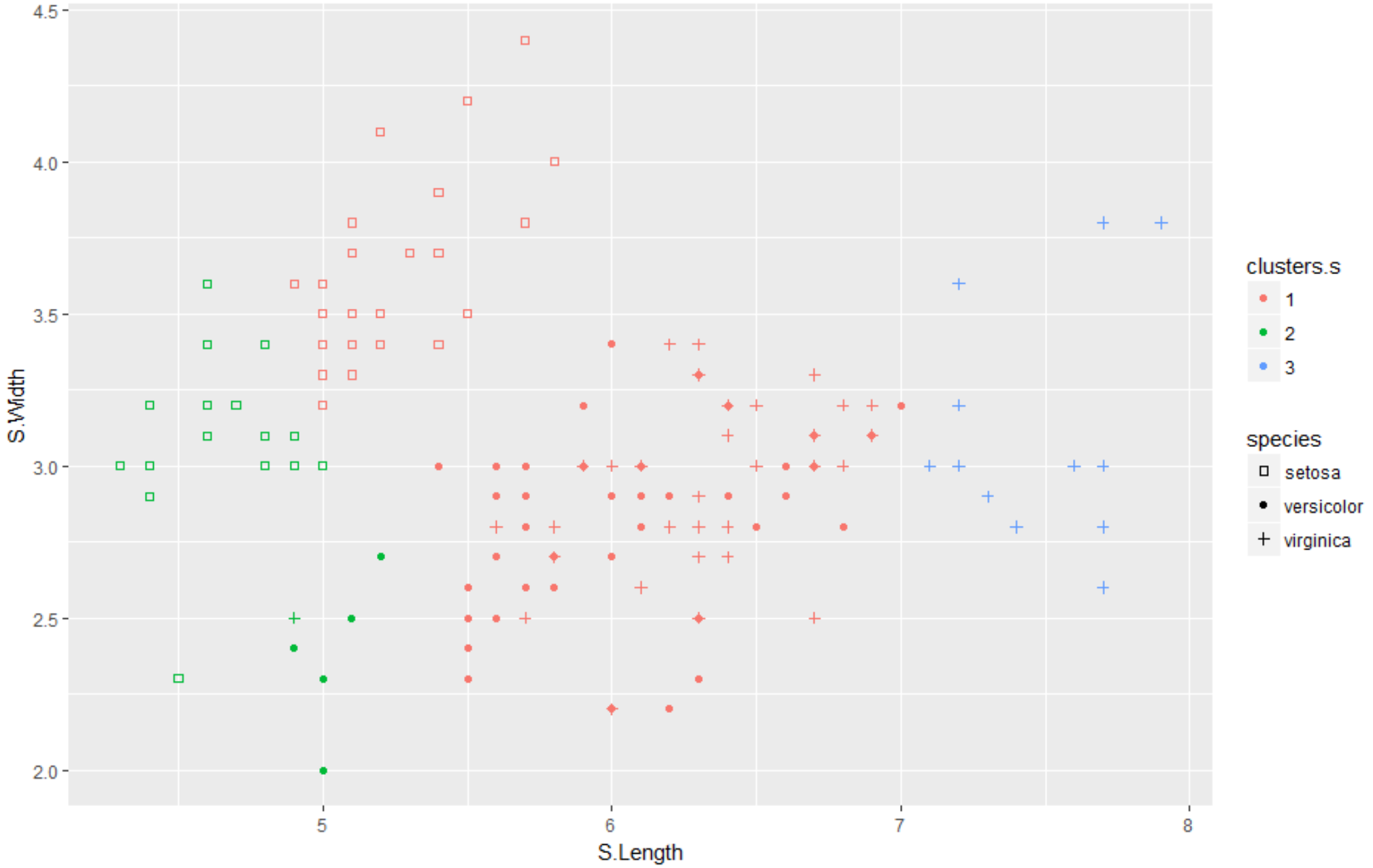
Maybe Sartre was right all along…

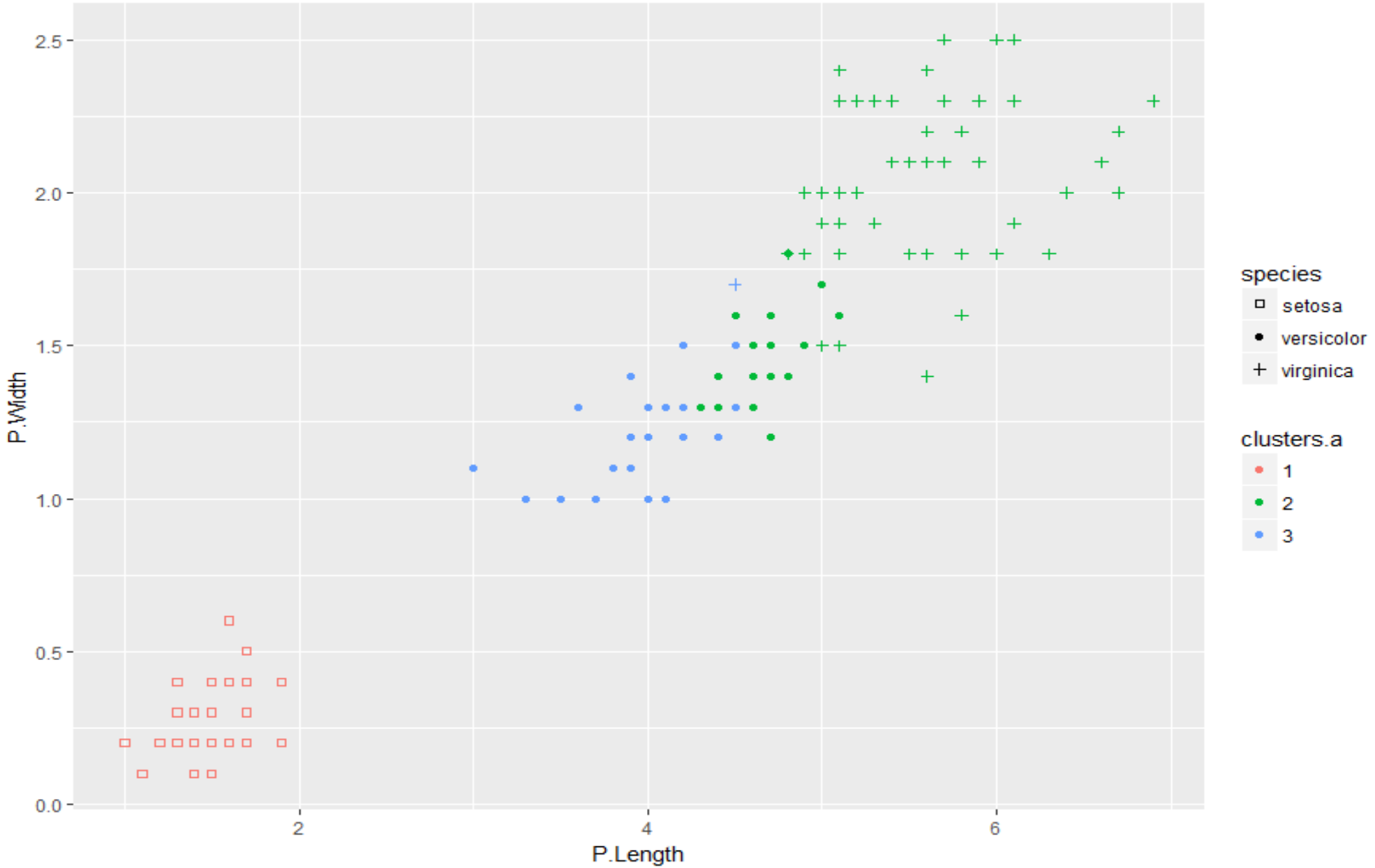
3 different inputs 3 different outputs!
Default : Complete linkage

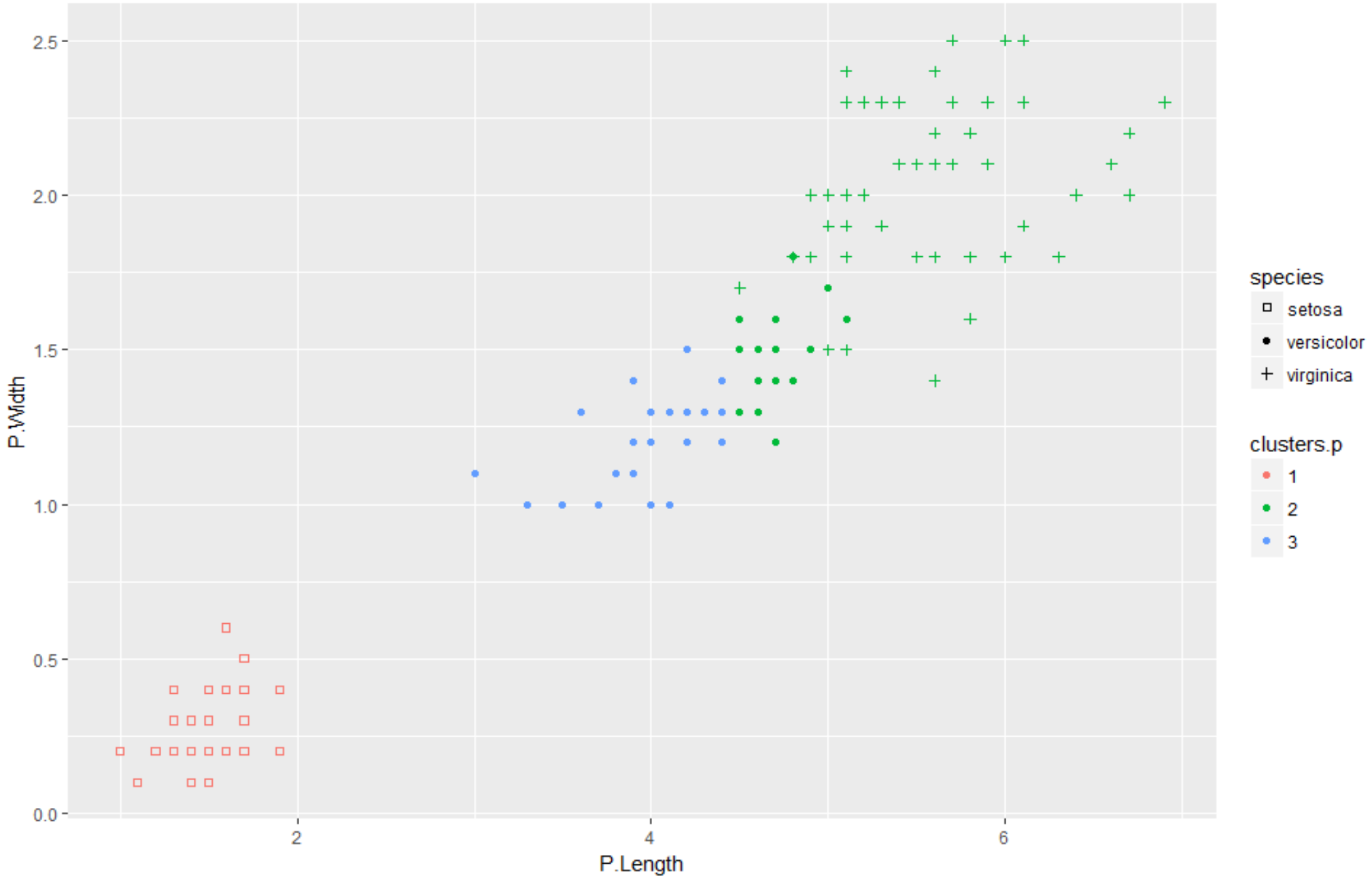
```
iris.clusters <- hclust(dist(iris[, 1:2])) #First two columns
clusters.s <-    as.factor(cutree(iris.clusters,3))

iris.clusters <-  hclust(dist(iris[, 3:4])) #Second two columns
clusters.p    <-  as.factor(cutree(iris.clusters,3))

iris.clusters <-  hclust(dist(iris[,1:4])) #All Data
clusters.a    <-  as.factor(cutree(iris.clusters,3))
```

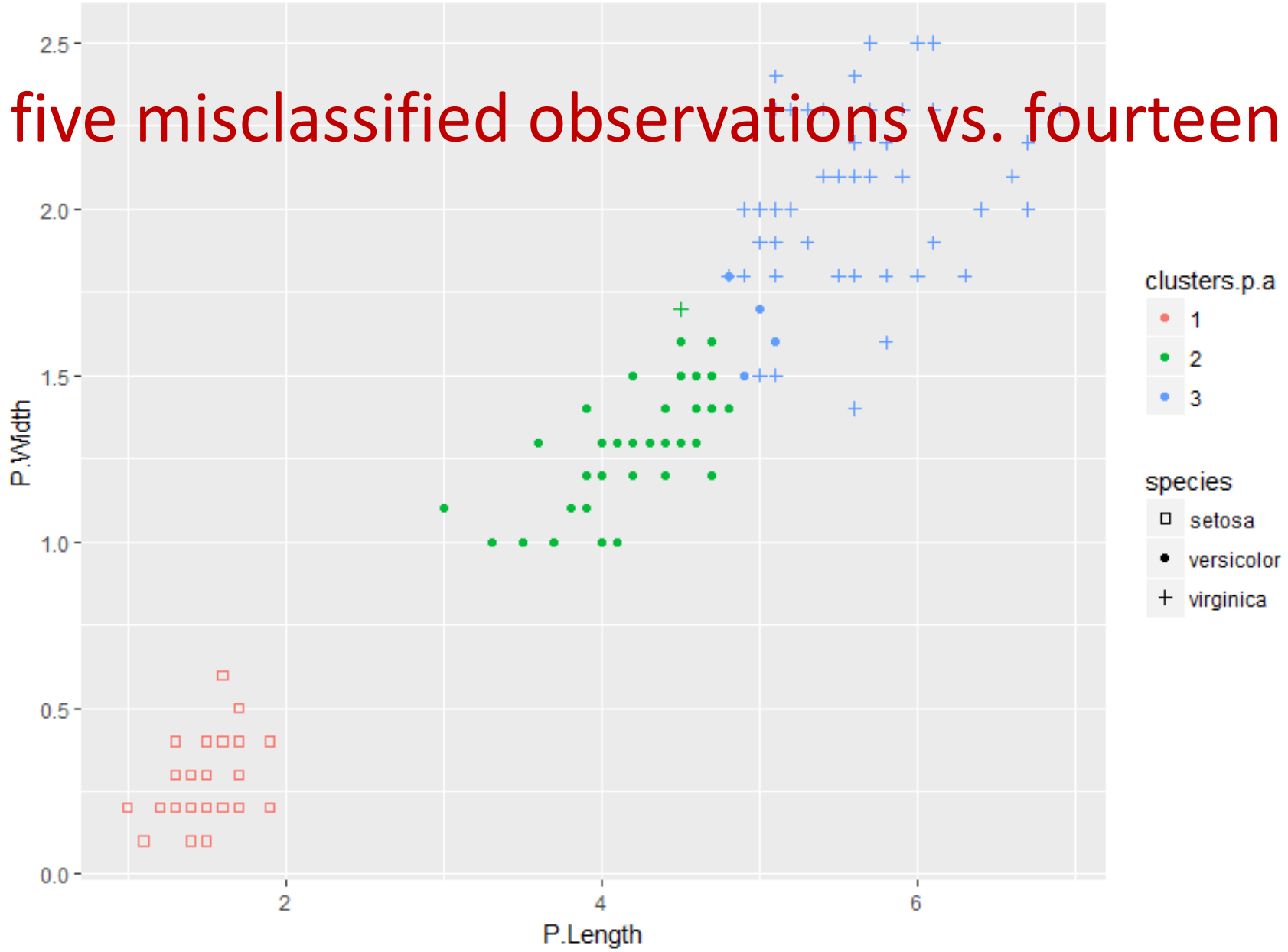

4.5
4.0
3.5
3.0
2.5
2.0
S.Width
5
6
7
8
S.Length
clusters.s
1
2
3
species
setosa
versicolor
virginica


2.5
2.0
1.5
1.0
0.5
0.0
P.Width
2
4
6
P.Length
species
setosa
versicolor
virginica
clusters.a
1
2
3


2.5
2.0
1.5
1.0
0.5
0.0
P.Width
2
4
6
P.Length
species
setosa
versicolor
virginica
clusters.p
1
2
3

What about the last slide using average linkage?

Only five misclassified observations vs. fourteen



A few observations

- Clustering on all of the data is worse than clustering on just petal characteristics.
- Clustering on sepal characteristics is worse than petal characteristics.
- We knew there were 3 clusters. In real life we will have no idea how many clusters there really are.
- The method of choosing between inter cluster distance changes the result.

A few observations (continued)

- Though we proceed as if Thomas Aquinas is right, maybe Sartre had a little bit to him
  - In general, there is no universally best way to “cluster things.
    - GIGO (Garbage in garbage out).
    - We must focus on the task and purpose of our clustering/segmentation.
    - This doesn’t imply some Hegelian “egg” in the process of becoming a “chicken,” i.e., we do have real differences
    - We must be cautious reporting our findings. There is grey area.
  - AFTER ALL: We are searching for qualitative groupings that are usefull! This is not finding the cure for cancer! Even though they do this stuff looking for new drugs.

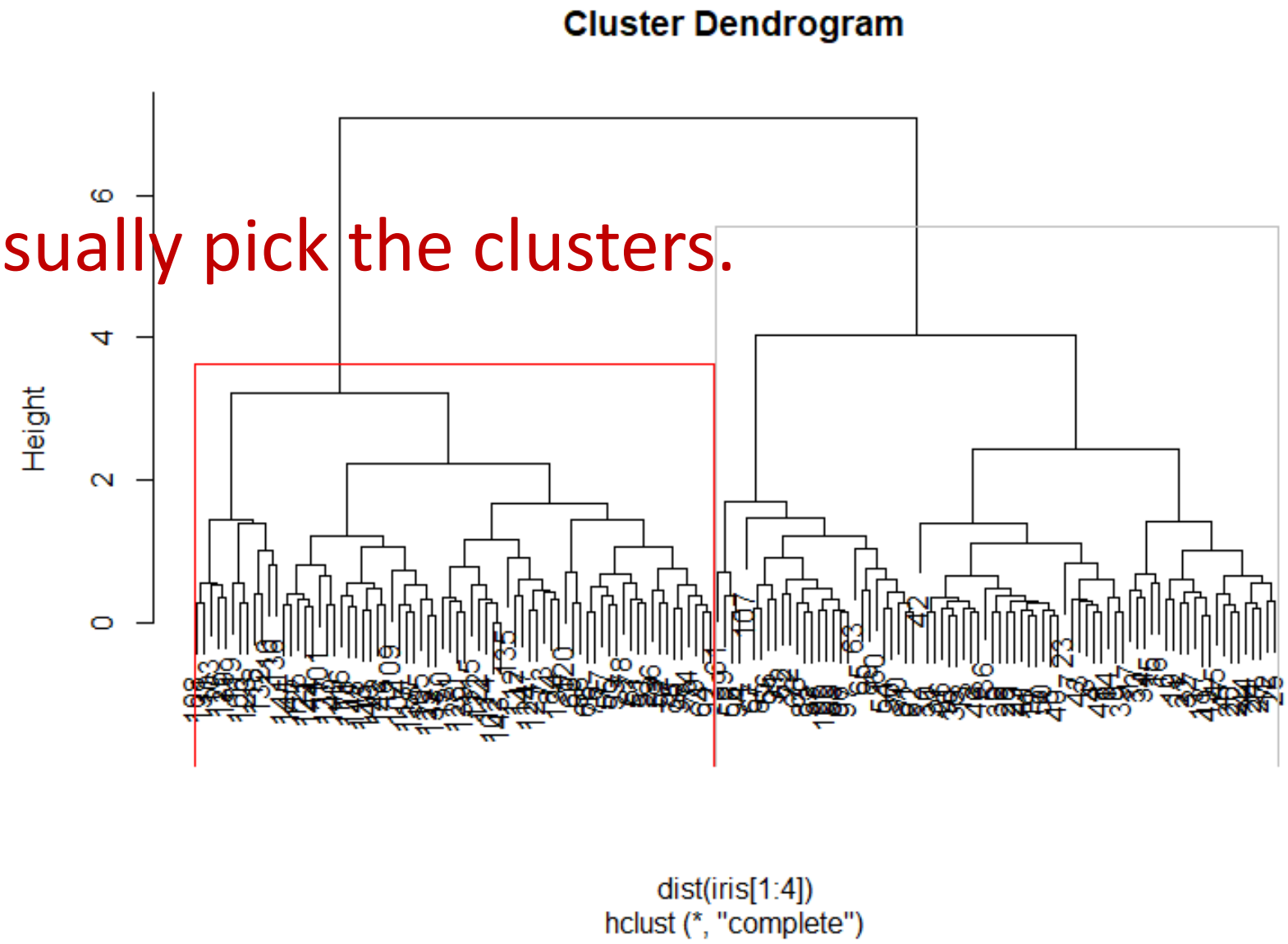
For Qualitative groupings ideally we want:

- High within class similarity.
- Low between class similarity.

This is just a fancy way of us saying we are not making a distinction out of thin air and reporting it to our boss.

```
plot(iris.clusters)
ans <-identify(iris.clusters) #choose clusters
                               #ans will give observations
                               # in each cluster
```

Here I visually pick the clusters.



A more quantitative way to choose the optimal number of clusters?

Kind of…

Let's think of between and within cluster similarity and between cluster similarity and variance.

Idea: The "correct" number of clusters is found by minimizing the within cluster sum of squares.

$$\sum_j \sum_i d\left(x_{ji}, \bar{x}_j\right)^2$$

Total distance from the centroid mean

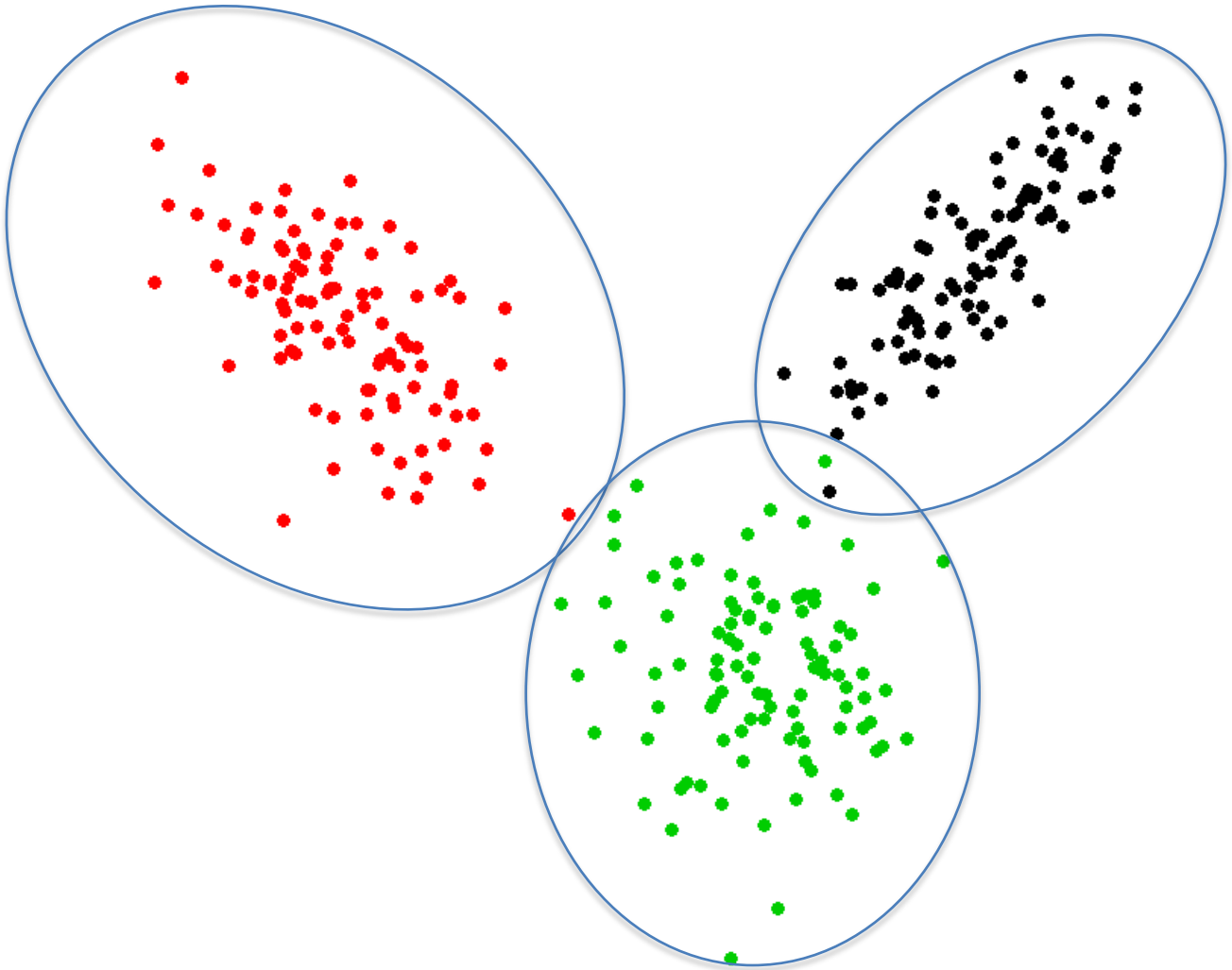
Elbow Method: WSS

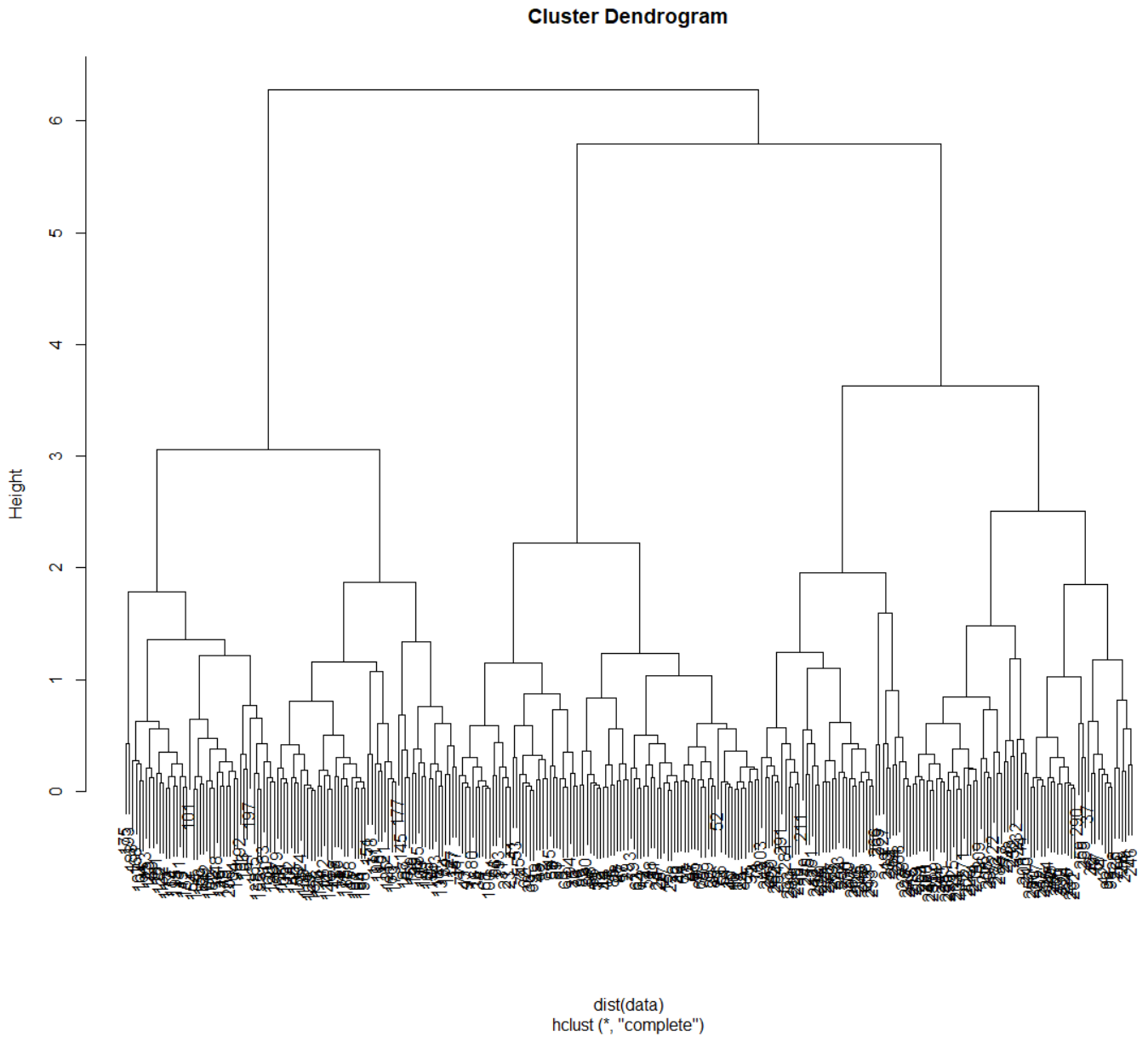
Let's think of between and within cluster similarity and between cluster similarity and variance.

Idea: The "correct" umber of clusters is found by finding the "sweet" spot between the number of clusters and the within cluster sum of squares.

If I have say 3 clusters, does the total variance of each cluster decrease a lot if I go to 4 clusters etc?

Note: The total variance goes to zero as the number of clusters increases.

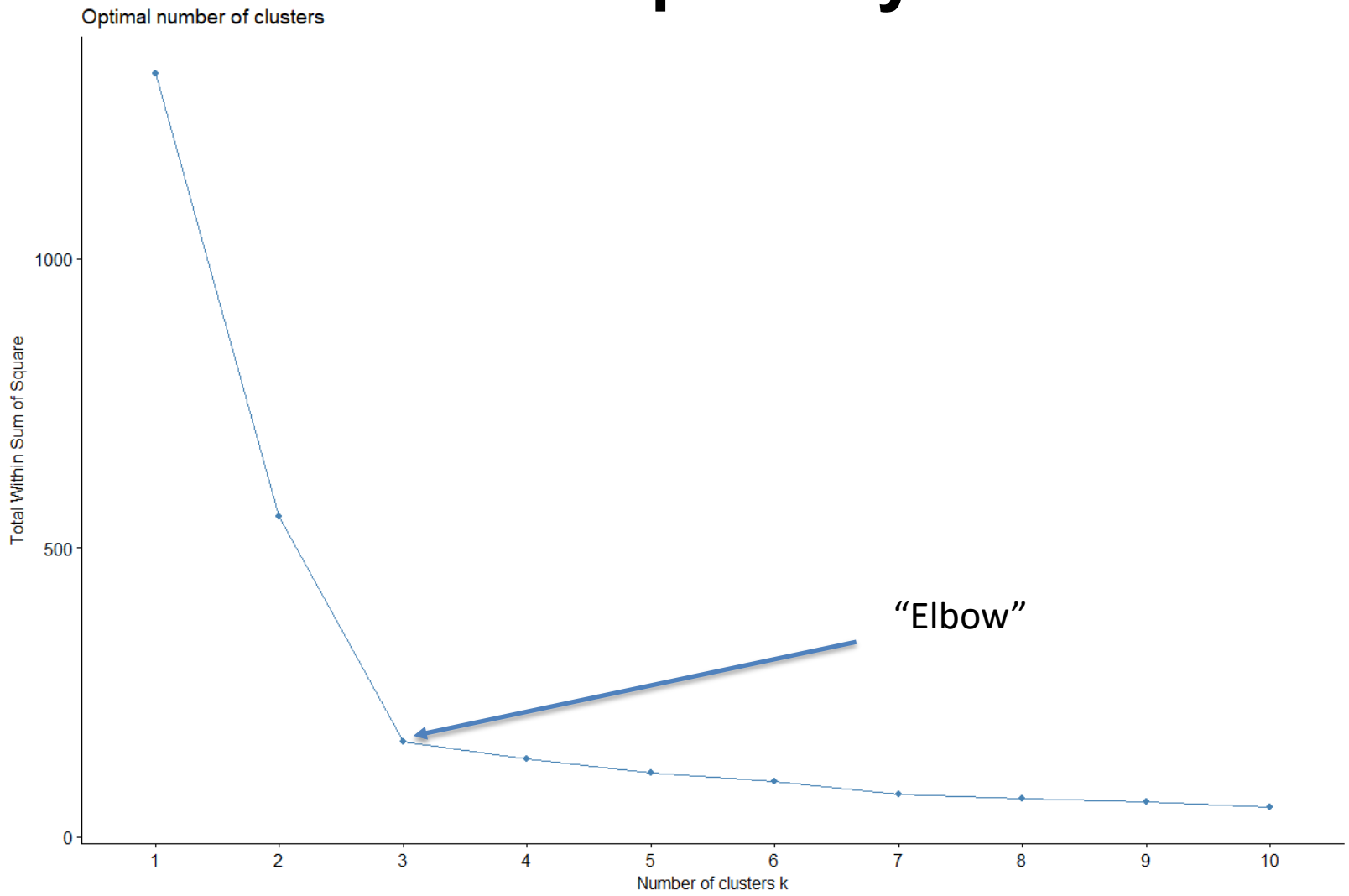



Cluster Dendrogram
Height
0
1
2
3
4
5
6
dist(data)
hclust (*, "complete")

As you add clusters the WSS starts to change less and less.

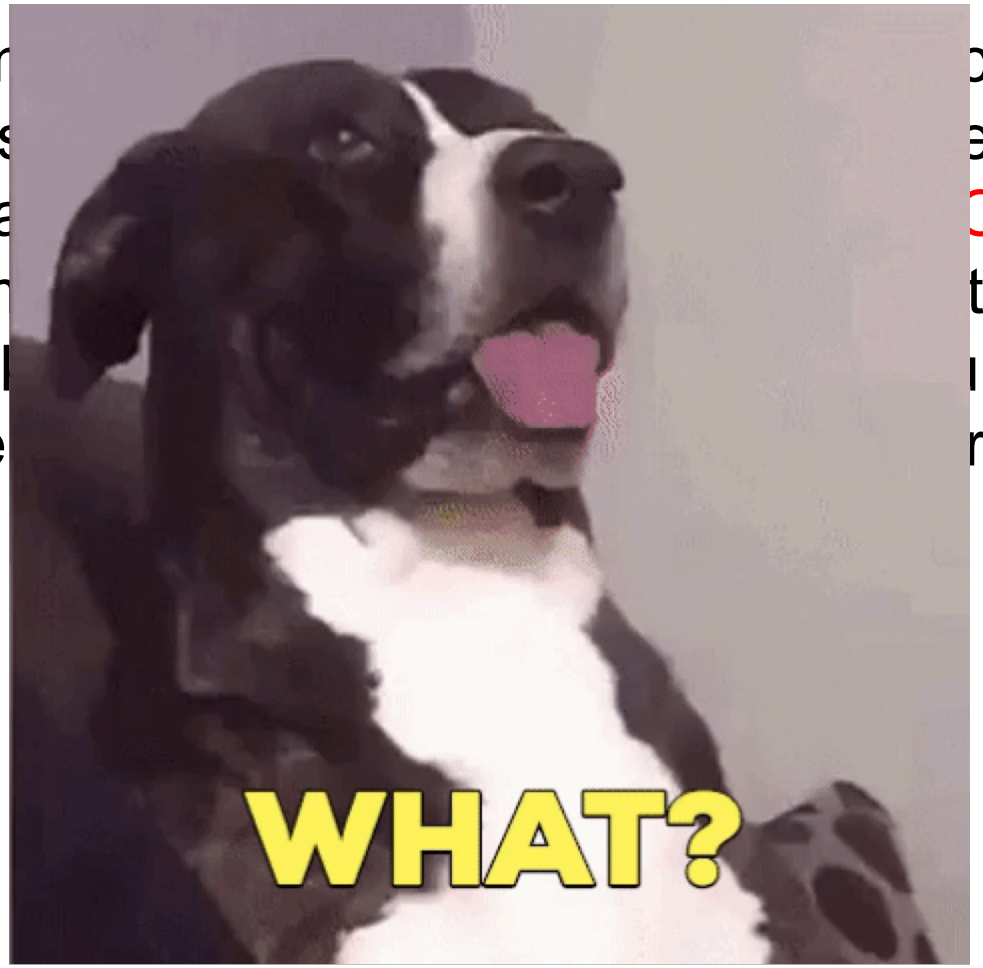
Clusters	WSS
1	1320.2
2	554.0
3	163.5
4	110.2
5	95.4
6	73.3
7	66.8

Graphically



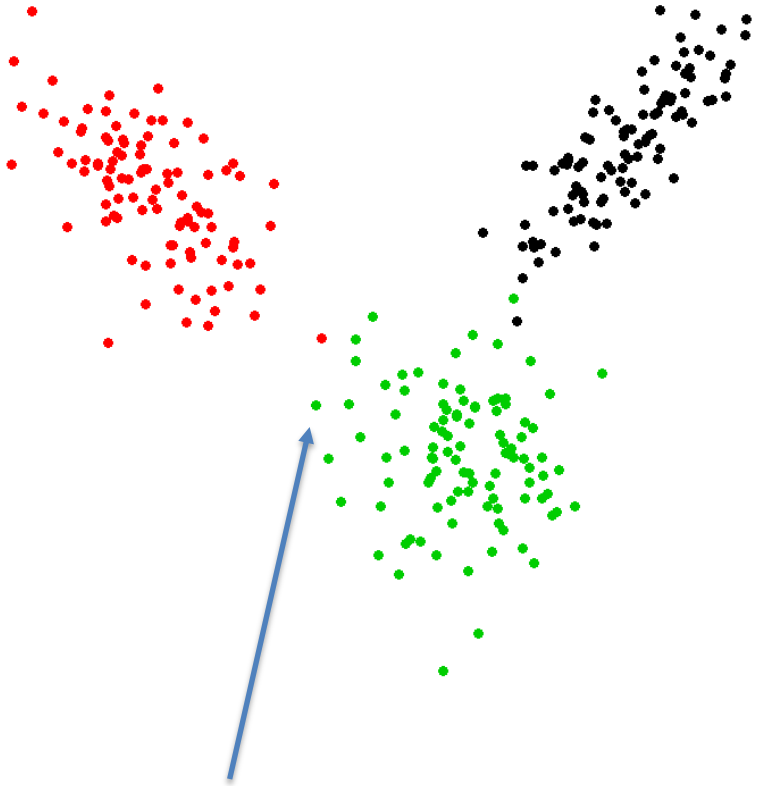
Gap Statistic

Idea: When we are com cluster there is the possibility we have chos er that doesn't really exist. We sample from a O clusters and then tell the algorithm "give n this distance to the data set where we find l mber of clusters that maximizes the distance ribution and our distance.



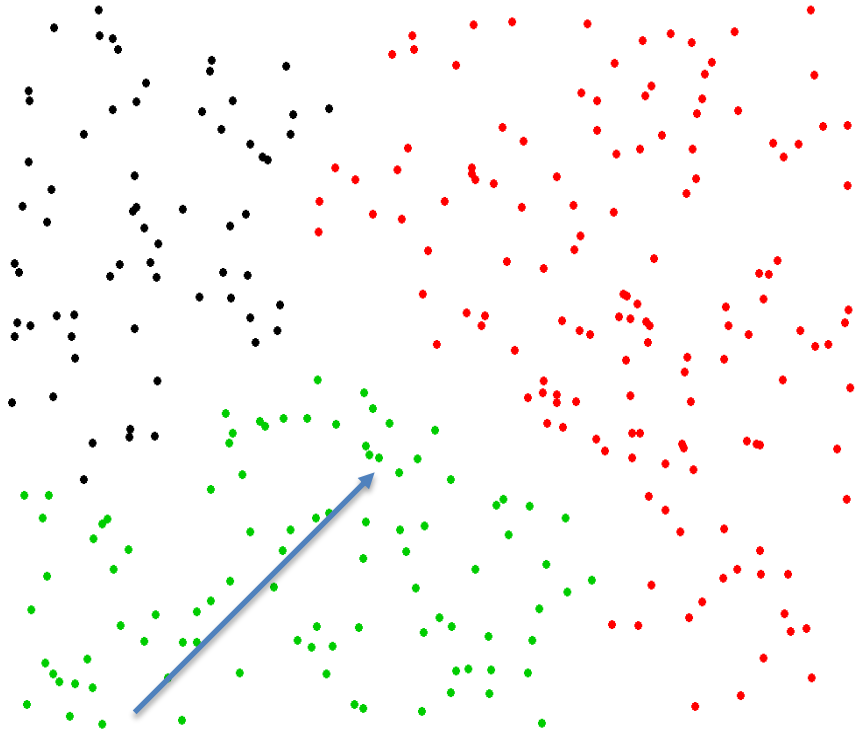
Gap Statistic

3 Real Clusters



This is going to have a smaller distance

3 Fake Clusters



On Average this is going to have an 'typical' or expected 'fake cluster distance'

We want to maximize the distance between the “random noise” cluster distances and the “non-random” cluster distances.

Total Within Cluster C_k distance:

$$D_k = \sum_{i,i' \in C_k} d(x_i, x_{i'})$$

Total Pairwise Distances

$$W_k = \sum_{r=1} \frac{1}{2n_r} D_k$$

Gap Statistic

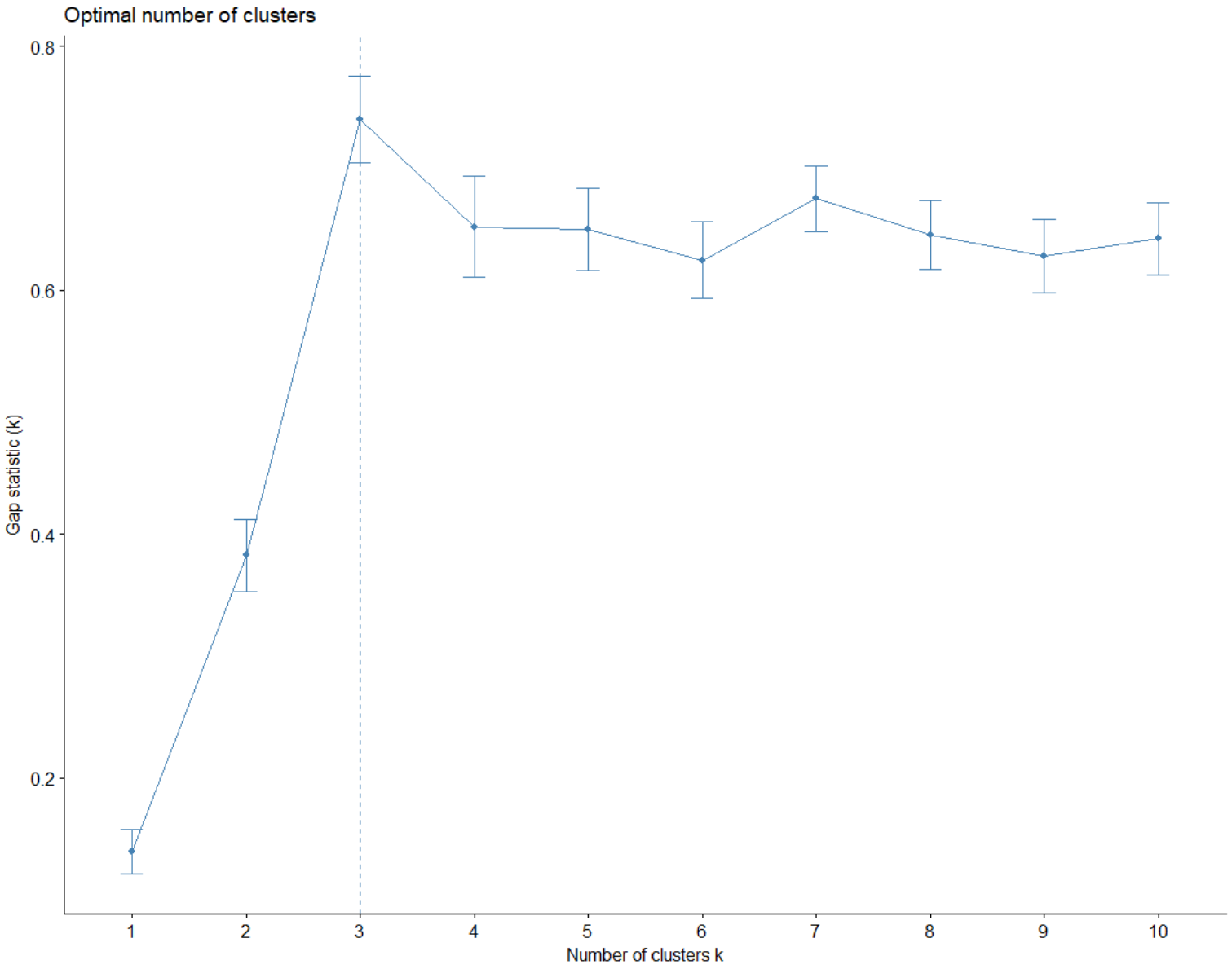
$$Gap_n(k) = E_n\left[\log\left(W_k^{noise}\right)\right] - \log(W_k)$$

Expected “noise” pairwise distances with k clusters

Data pairwise distance With k clusters

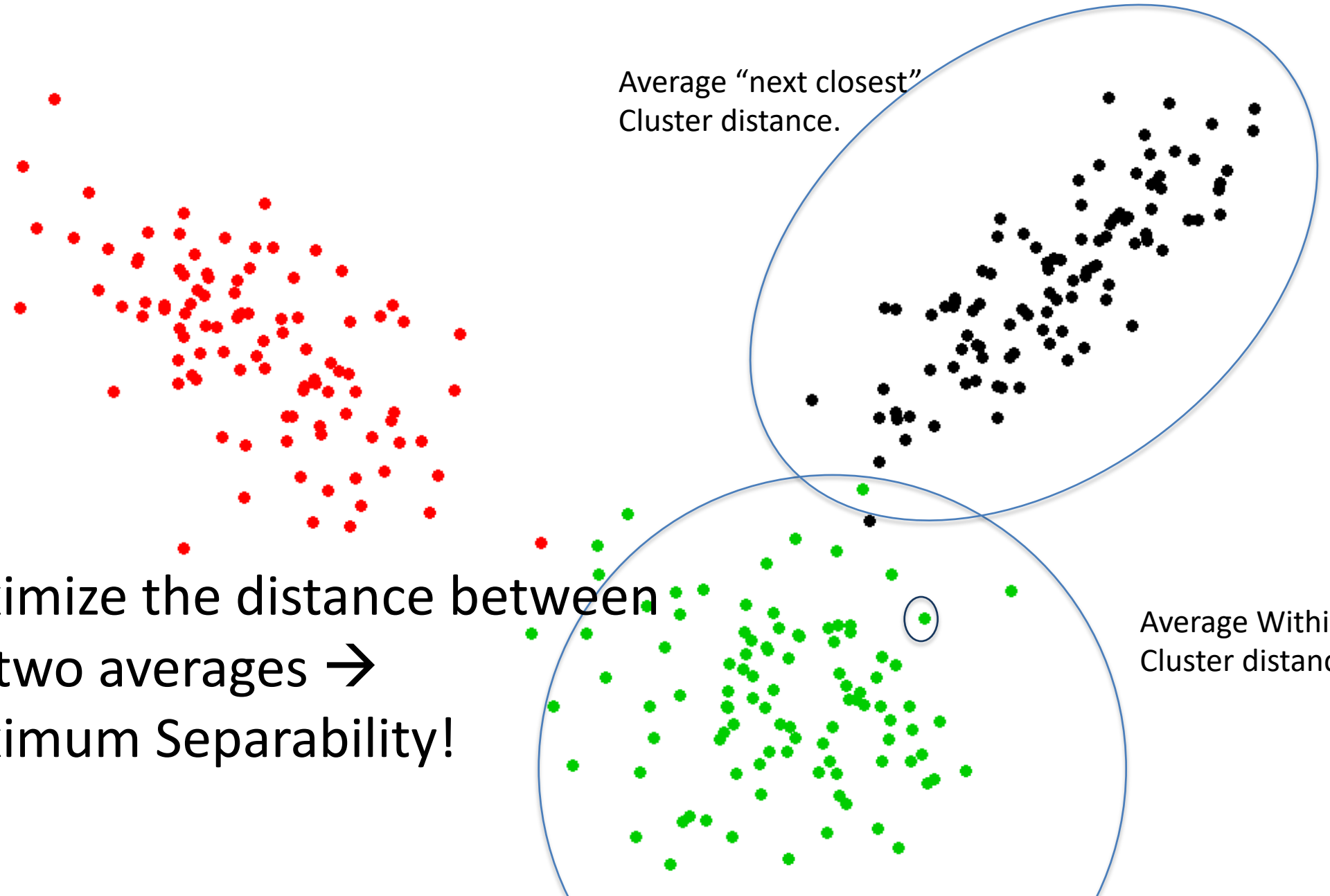
Idea Revisited: Initially there is going to be an increase in the gap statistic because I am increasing the number of clusters, BUT at a certain point I should be adding fake clusters thus noise and the gap statistic should go down. I stop at the first down.

Gap Statistic



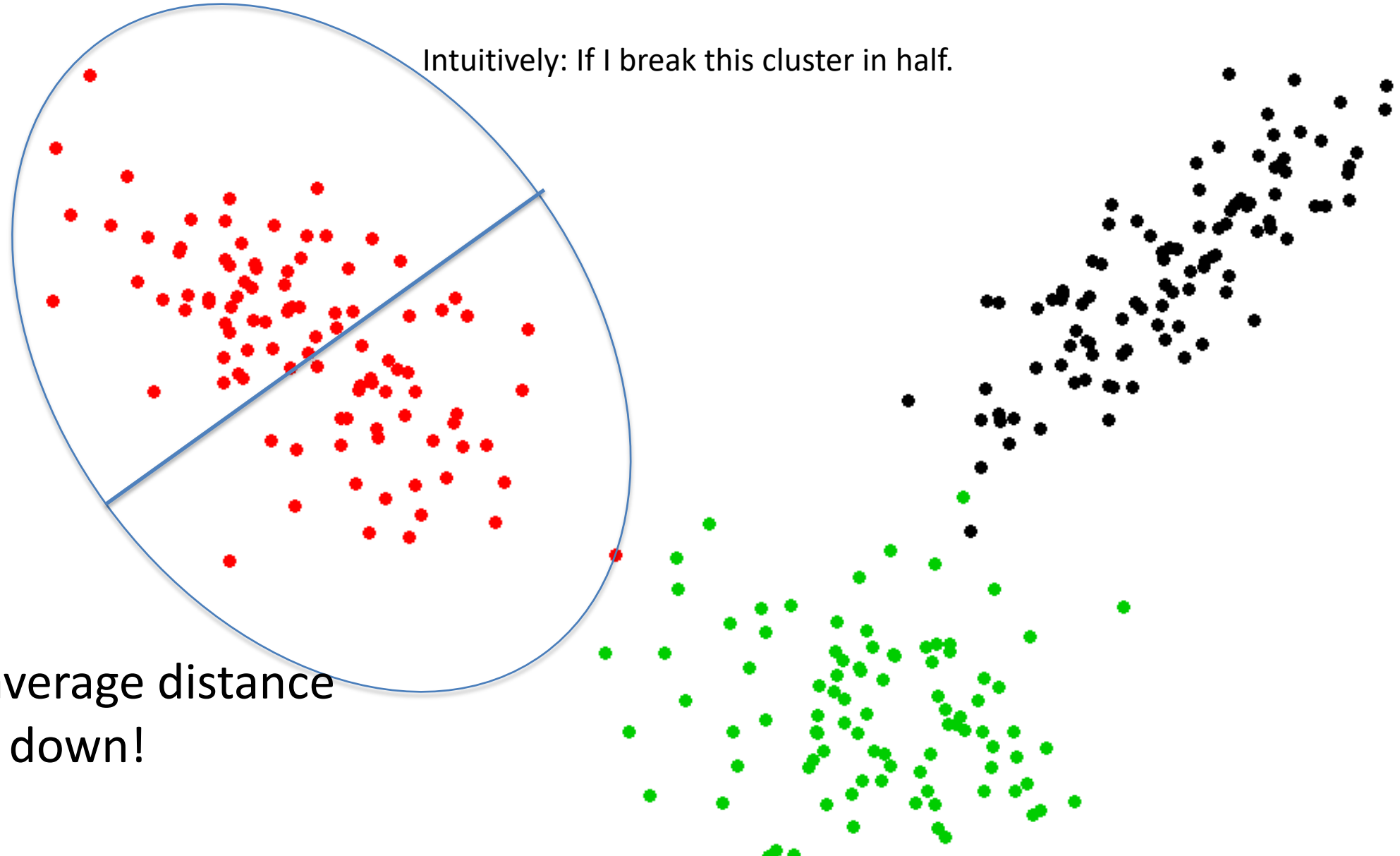
Silhouette Method

Idea: In every cluster, every observation has a within cluster distance, and a between cluster distance. For each observation, look at the average within cluster distance, and the average distance to each observation in the “next closest” cluster. Subtract the average between cluster distance by the within cluster distance and normalize this by the maximum distance between observations in two groups. We want to maximize this value across all observations.

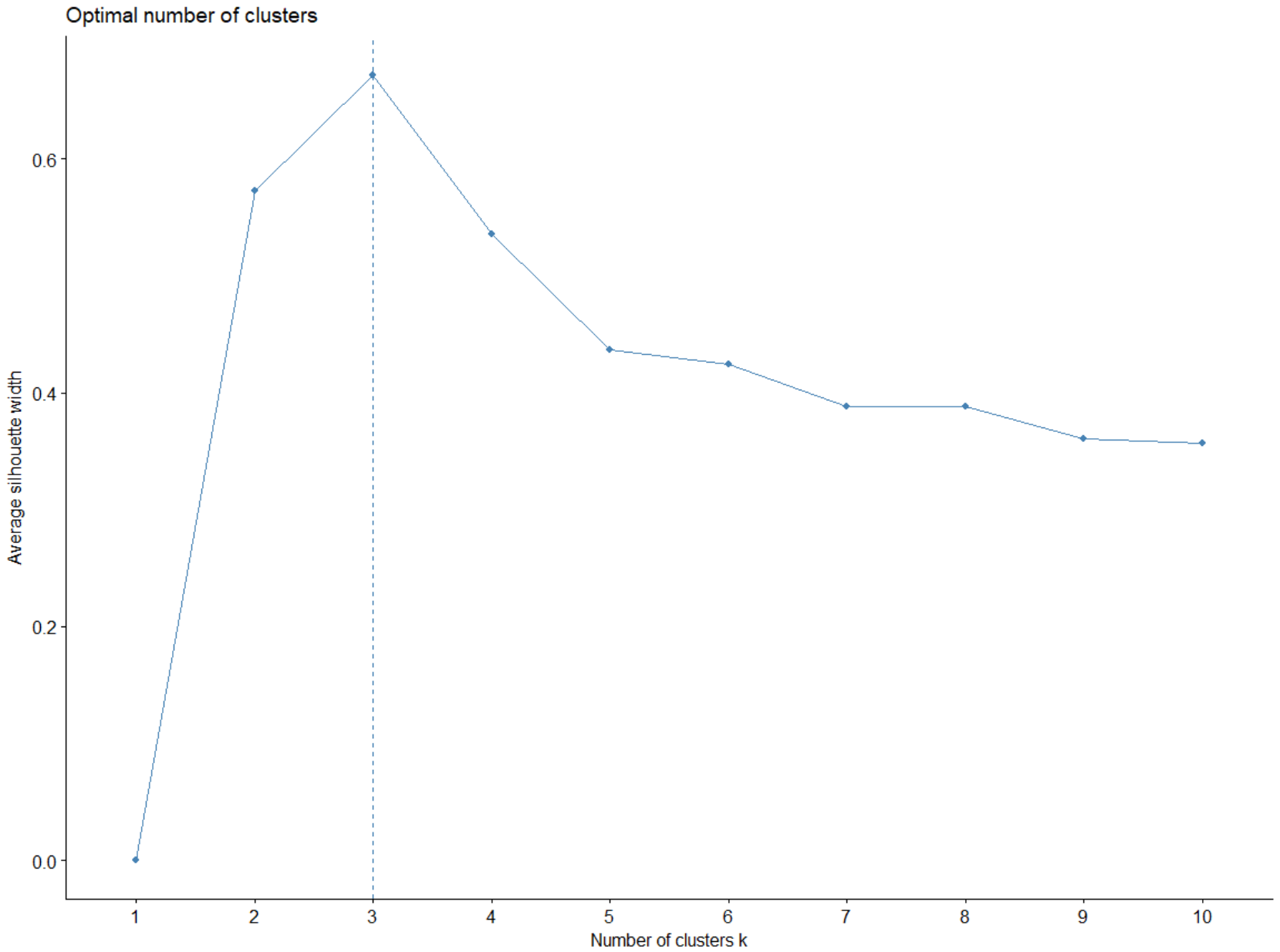


Maximize the distance between the two averages →
Maximum Separability!

Intuitively: If I break this cluster in half.

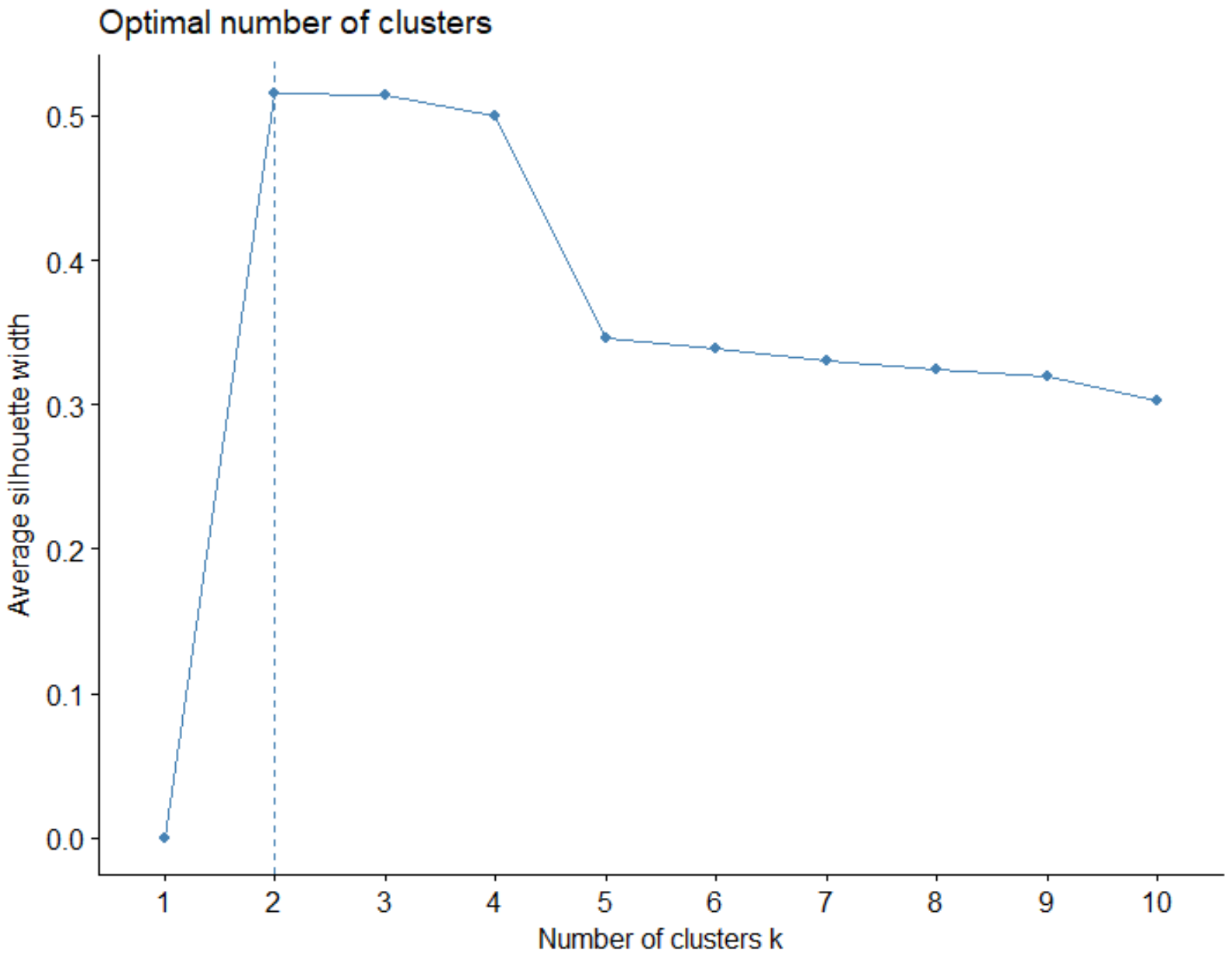


My average distance goes down!

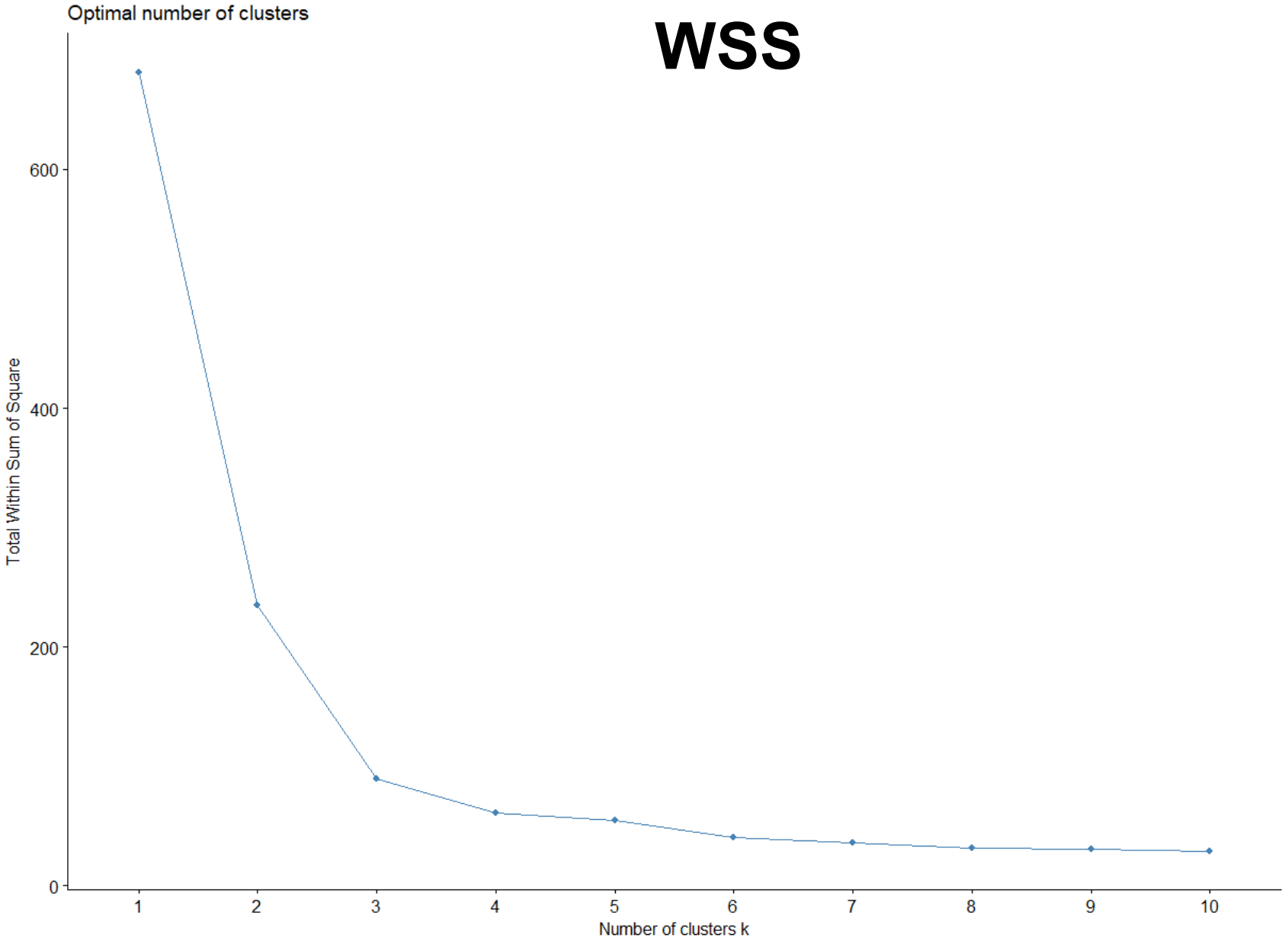

Optimal number of clusters
Average silhouette width
0.0
0.2
0.4
0.6
1
2
3
4
5
6
7
8
9
10
Number of clusters k

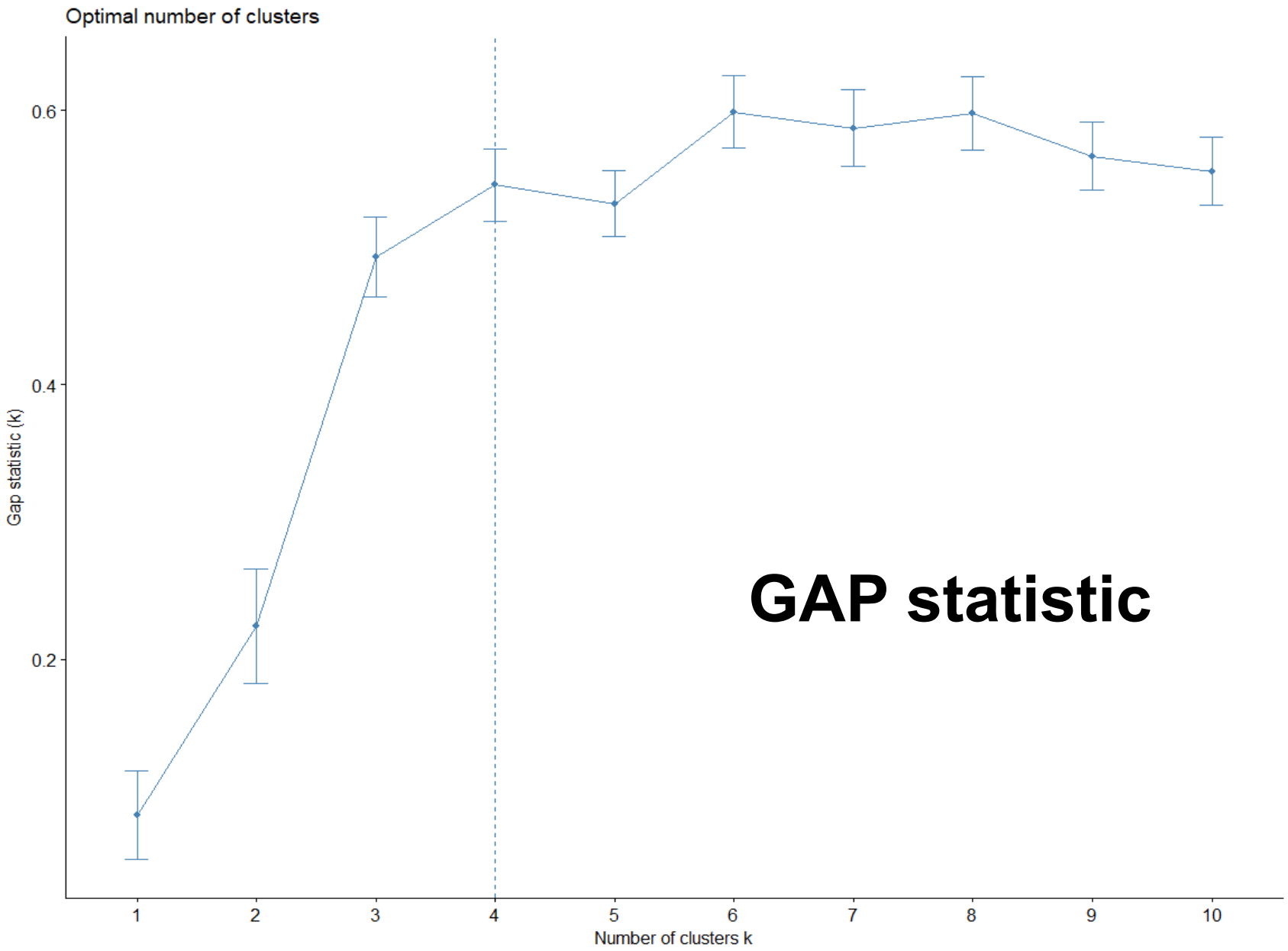
For our “super toy problem”

- Every method produces the exact same answer 3 clusters.
- Let’s try the IRIS data set. Still a toy problem.

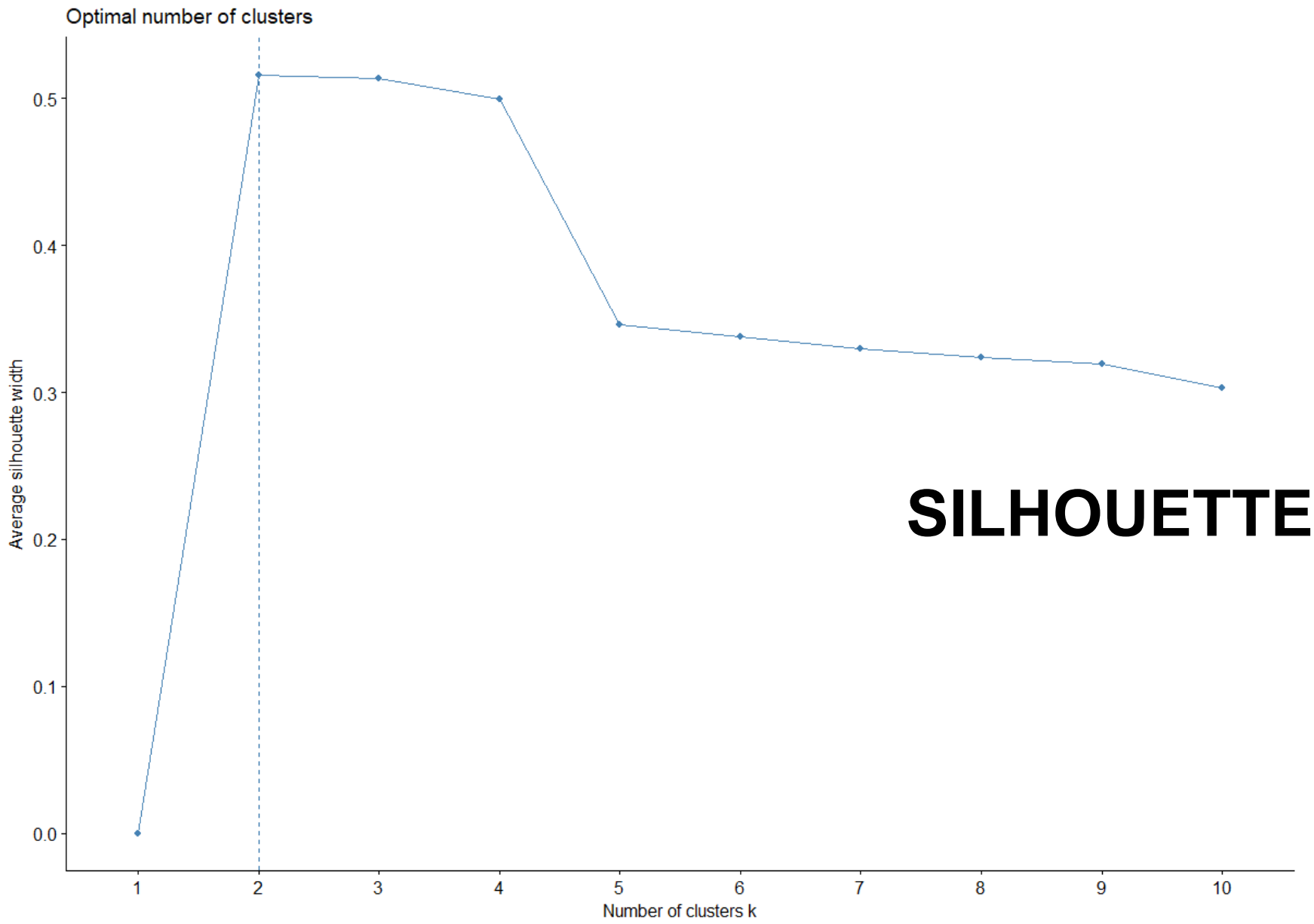

Optimal number of clusters
Average silhouette width
0.0
0.1
0.2
0.3
0.4
0.5
1
2
3
4
5
6
7
8
9
10
Number of clusters k

WSS


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



GAP statistic


Optimal number of clusters
Average silhouette width
0.0
0.1
0.2
0.3
0.4
0.5
1
2
3
4
5
6
7
8
9
10
Number of clusters k

SILHOUETTE

Three Methods: Three different results

- Each method produces a different “optimal number of clusters”
  - Which one should you use?
    - It depends based upon the question you are trying to answer.
      - Are you trying to maximize the differences between groups: That is you want your groups to be different? Maybe the smallest number of clusters is best!
      - Is this purely a qualitative exercise for management? Do they want clusters/segments that describe the little differences? Maybe the largest answer is the best.
      - Etc. etc. etc.

Questions for future lectures:

- How can we visualize clusters: Especially if we have a high number of dimensions?
- Hierarchical clustering is SLOW I have to find the distance between ALL POINTS: Are there better approximate clustering algorithms?
- This only applies to data where I observe continuous numbers: What do I do with text? What do I do with functions?