UNIVERSITY OF WISCONSIN
DATA SCIENCE

DS 740

Data Mining

Clustering Methods

How to Group Unsupervised

Important note: Transcripts are **not** substitutes for textbook assignments.

Learning Objectives

By the end of this lesson, you will be able to:

- Explain why clustering methods are unsupervised methods.
- Identify situations in which clustering is appropriate (versus classification or predictive methods).
- Define hierarchical and non-hierarchical modeling methods.
- Distinguish between hierarchical and non-hierarchical modeling methods and compare in context.
- Apply hierarchical clustering with the `hclust` function
- Apply one method of non-hierarchical clustering with the `kmeans` function

Iris Example

Using iris data set, already loaded in R, with variables.

- Four size measurements (all in cm) of iris plant: *Sepal.Length*, *Sepal.Width*, *Petal.Length*, *Petal.Width*
- *Species*, one of “setosa”, “versicolor”, and “virginica“

1. review classification with response.
2. identify when appropriate for clustering..
3. visualize clustering with plots.
4. overview possible “good” clustering methods.

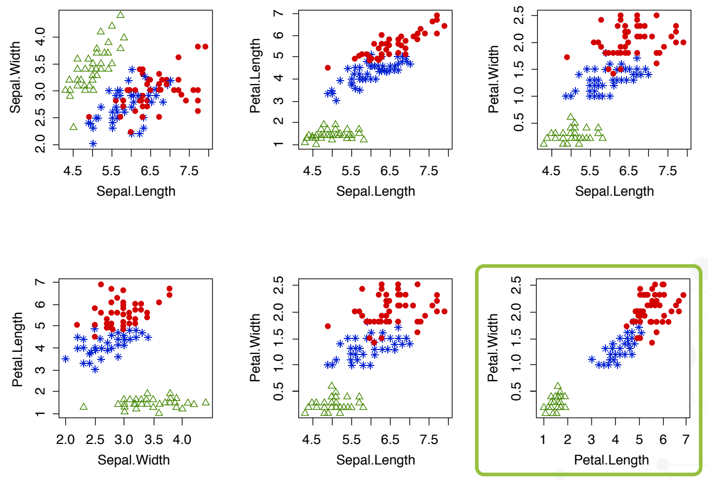
Throughout this lecture, we will be reusing the hopefully now familiar Iris dataset with the intention of presenting an example with readily visible clusters. The goal is to reorient our thinking about the purpose of the analysis towards clustering the data into similar groups along with discussion of visuals, methods of assessing similarity, and selection of numbers of clusters.

Notes:
Link: iris data set

Iris: With Known Species

setosa: △
versicolor: ✳
virginica: ●

Clearly separable groups, but both **Petal.Width** and **Petal.Length** distinguish well individually.



When we initially introduced the Iris data, the goal was to use various size measurements to best predict an observed classification into one of three possible species-- settosa, versicolor, and virginica. Knowing the end result-- the species class, which is visualized as different colors and symbols on the plots, allowed us to visualize the separation into groups across the various variables.

We have no way of visualizing the grouping across all four predictors at the same time. This is due to the so-called curse of dimensionality, which limits us to visual perception of two or, at most, three dimensions. This also plays into our later discussion of distance, or similarity, measurements.

However we can visualize the groupings across pairs of predictors, and this does allow us to select some predictors that are better discriminators.

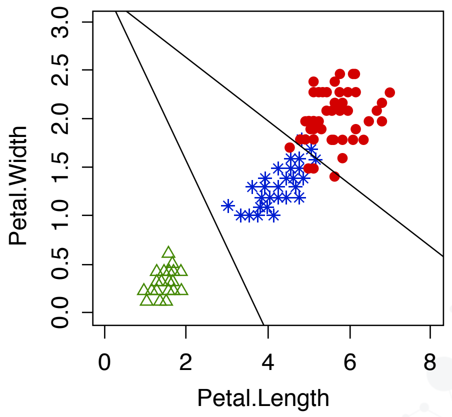
Iris: LDA Using Known Species

With known *Species* as response guidance, find boundaries to classify into groups (LDA solution with p=2).

Other possible approaches:

LDA or QDA with p=4;

k-Nearest-Neighbors; classification trees; multiple-class logistic regression.



Focusing on two predictors-- pedal length and pedal with-- each of which appears to have values that provide separation into the three species, we visualize their values jointly in a scatter plot. Noting that the three groups seem almost separable when we know what the groups are, we further use a Linear Discriminant Analysis, or LDA, to find lines in the dimensions of the two predictors that best separate the plants into species.

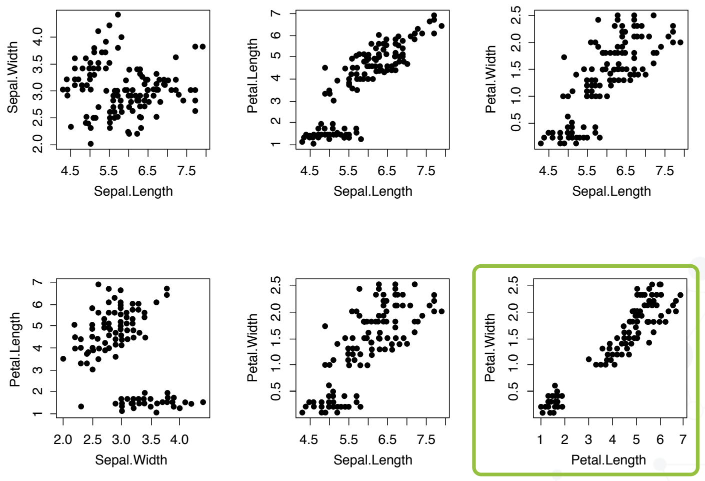
Note that this is just one of many possible approaches to the classification question, several methods or approaches of which have been covered in this class. But all these methods rely on having a known result-- a known classification response variable.

Iris: With Known Species

Species = ?? ● ~~("setosa", "versicolor", "virginica")~~

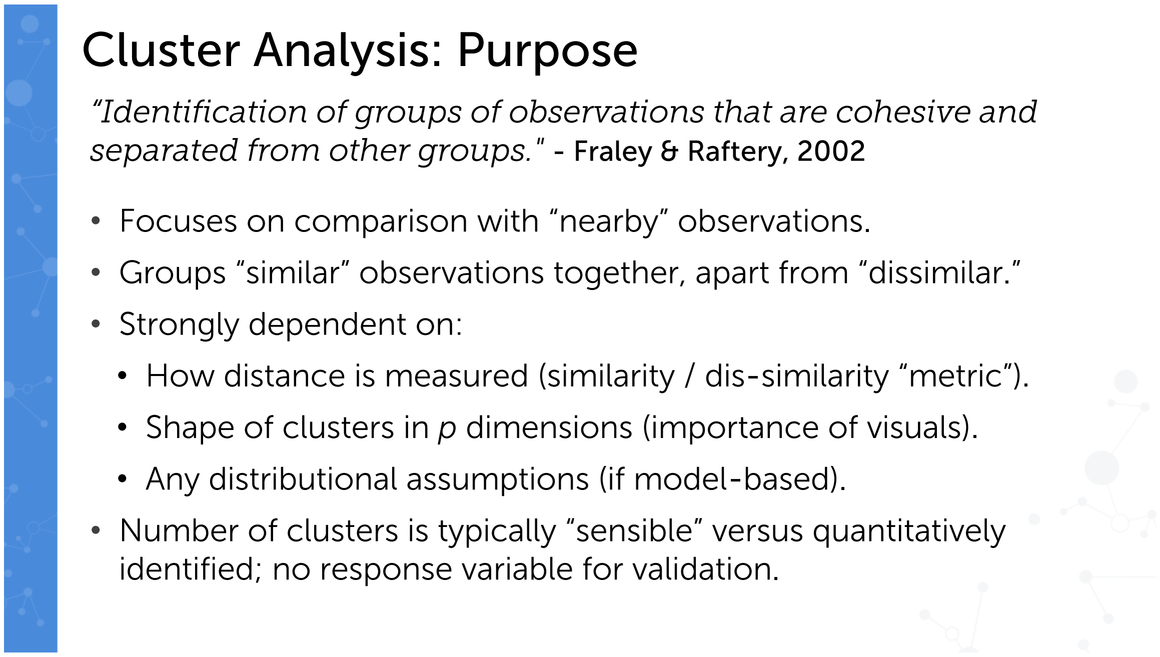
Visually clear groups:

1. Only two obvious.
2. No response classification to "check" → **unsupervised**.



We now reframe this example in a context without a known classification response variable. That is, we look at the grouping or orientation of the data across the pairs of size measurement variables, but we no longer mark according to species.

That is, we act as though we don't know the classification. Visually, we do still see clustering, or grouping of the data, but now only two groups stand out visually and are readily apparent in any of the scatter plots. And without a response, we have no way to check or to validate our answer. In the next few slides, we'll explore how we might approach the situation.



Cluster Analysis: Purpose

"Identification of groups of observations that are cohesive and separated from other groups." - Fraley & Raftery, 2002

- Focuses on comparison with "nearby" observations.
- Groups "similar" observations together, apart from "dissimilar."
- Strongly dependent on:
  - How distance is measured (similarity / dis-similarity "metric").
  - Shape of clusters in *p* dimensions (importance of visuals).
  - Any distributional assumptions (if model-based).
- Number of clusters is typically "sensible" versus quantitatively identified; no response variable for validation.

A foundational paper on clustering from the Journal of the American Statistical Association defines clustering as, "identification of groups of observations that are cohesive and separated from other groups." This means that we want to group, or cluster, observations with those which they are most similar, and to keep them apart from dissimilar observations.

The following slides will illustrate how this process can depend on, and even potentially be benefited by, logical choices for how the distance is measured-- that's our similarity or dissimilarity measurement-- the shape of clusters in the P predictors dimensions, and any distributional assumptions.

The number of clusters is an unknown. And understanding this allows us the flexibility to choose a number of clusters that is sensible to the purpose of the analysis.

Notes:
Fraley, C., and Raftery, A. Model-Based Clustering, Discriminant Analysis, and Density Estimation. Journal of the American Statistical Association, 97, p. 611-631, June 2002.

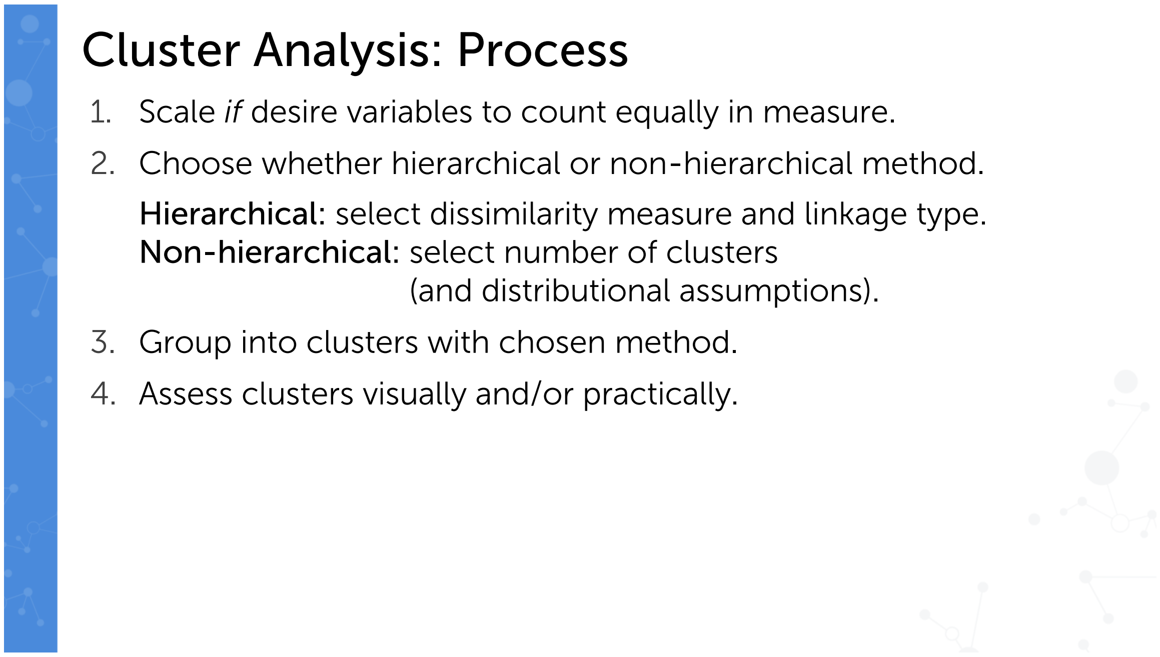
Some Possible Uses

Finding existence of clustering must precede studying possible causes.

- **Customers / purchasing audiences**: groups that can be targeted similarly by marketing.
- **Web use / preference**: individuals who might be interested in certain pages and/or ads.
- **Gene expression data**: genes that work together.
- **Image analysis**: identify similarly-colored regions of an image (restoration or compression).

Purposes for clustering vary widely. And this is also illustrated in the fact that clustering methods have arisen in a wide variety of areas and applications. There are a correspondingly wide variety of methods, but they can be loosely categorized. We will examine several of the most commonly applied and empirically supported methods in the next few slots.

Why might we want to group even if there is no response? We often cluster to find existence of similar groups towards the eventual aim of studying possible causes or approaches for those groups toward some purpose. Some examples in which clustering has commonly been applied is within purchasing audiences for marketing, web use or preference for marketing, and/or references, in gene expression data, for identifying genes that are connected, and in image analysis to work with regions that should be colored in the same way.



Cluster Analysis: Process

1. Scale *if* desire variables to count equally in measure.
2. Choose whether hierarchical or non-hierarchical method.

   Hierarchical: select dissimilarity measure and linkage type.
   Non-hierarchical: select number of clusters
   (and distributional assumptions).
3. Group into clusters with chosen method.
4. Assess clusters visually and/or practically.

When working through a cluster analysis, one of the first considerations is whether we scale the variables. Scaling does have an impact-- and often a dramatic impact-- on distance measurements, as we saw with, for example, K nearest neighbors.

However, there are situations in which scaling is not necessary, and we will see an example of such an instance in our Iris dataset as we progress through that. We will also see instances where it does make sense to scale, particularly if we have measurement units that can be changed and thus change the value that we have reported for the variable.

The next step is to choose whether we're using a hierarchical or non-hierarchical method. Each of these two methods has particular information associated with its fitting that needs to be specified as well, and we will take a look at these pieces of information in the slides ahead.

For hierarchical, we need a dissimilarity measurement and a linkage type. For non-hierarchical we need to select the number of clusters and potentially distributional assumptions if those are involved in the method.

We then take the chosen method and group the data into clusters, which we assess visually and or pragmatically to align with our desired purpose for the clustering.

Hierarchical and Non-Hierarchical

Goal of clustering: *Partition* the data “wisely”, split data into *K* groups such that:

- Each observation is in a group; and
- No observation is in more than one group.

Hierarchical clusters: larger clusters (that is, when there are few of them) are formed from smaller clusters.

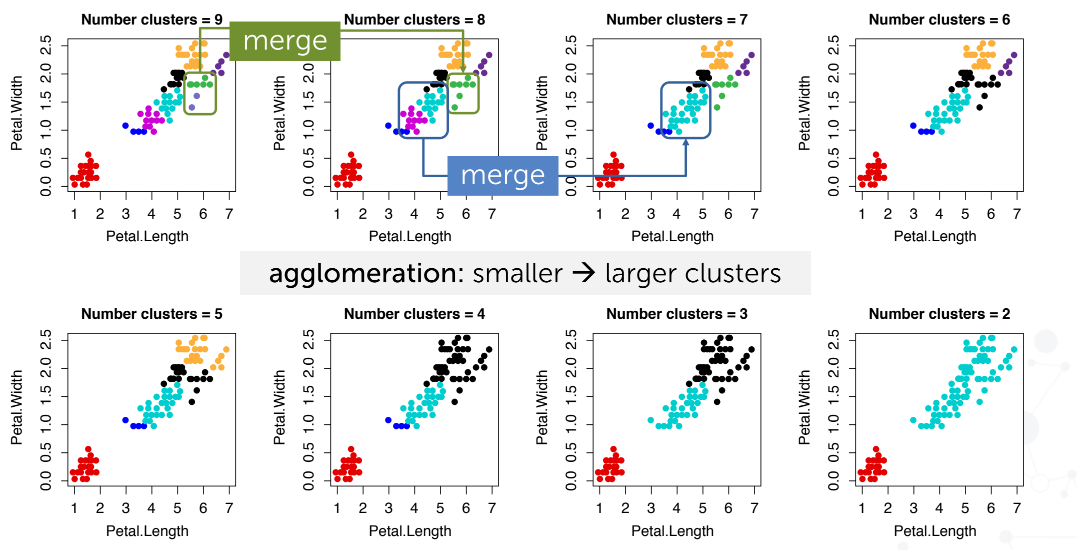
Non-hierarchical clusters: clusters are oriented towards a particular goal, for a particular number of clusters.

When we cluster, our goal is to partition the data, and to do so in a wise way. Partitioning means that we split the data into a certain number of groups-- we'll call that number of groups K-- in a way such that it is comprehensive, meaning that each observation is in a group, and unique, such that no observation is in more than one group.

The two basic ways of doing this are known as hierarchical clusters in which we can think of larger clusters as being formed or built up from smaller clusters. So when we have more-- or a larger number of clusters-- they must contain fewer observations, and then we group those smaller clusters together to come up with the larger clusters.

In that way, clusters can be built from smaller grouped into larger and larger. Non-hierarchical clusters, on the other hand, have a particular goal mathematically in mind that is used to determine the population of the different clusters.

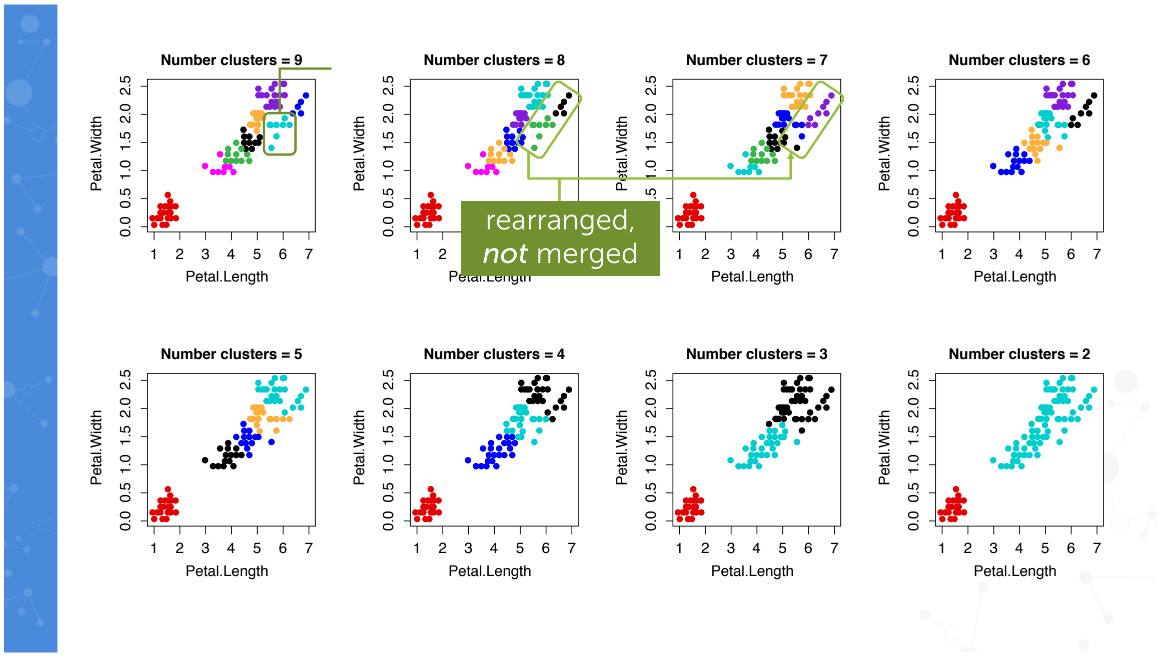
This goal is part of the specification of the method and is used with a particular number of clusters in mind. And so that number of clusters must be specified ahead of time.



A visualization of hierarchical clustering is shown here where the different colors correspond to different clusters. When we start with nine clusters, for example-- when we move to the plot just to the right of this, we note that the the two gray circles disappeared and became green. So the gray and green observations were in clusters that were merged to produce one cluster.

So moving from nine to eight merged two of the clusters into one. Correspondingly, as pictured here, moving from eight to seven clusters merged two slightly larger clusters-- the pink and the light blue clusters-- were merged into one.

As we progress through from eight to seven to six to five to four to three to two clusters, every single time we move two of the clusters are merging to produce one. And this is what is known as an agglomeration of clusters. Again, moving from smaller clusters when we have more of them to larger clusters when we have fewer of them.



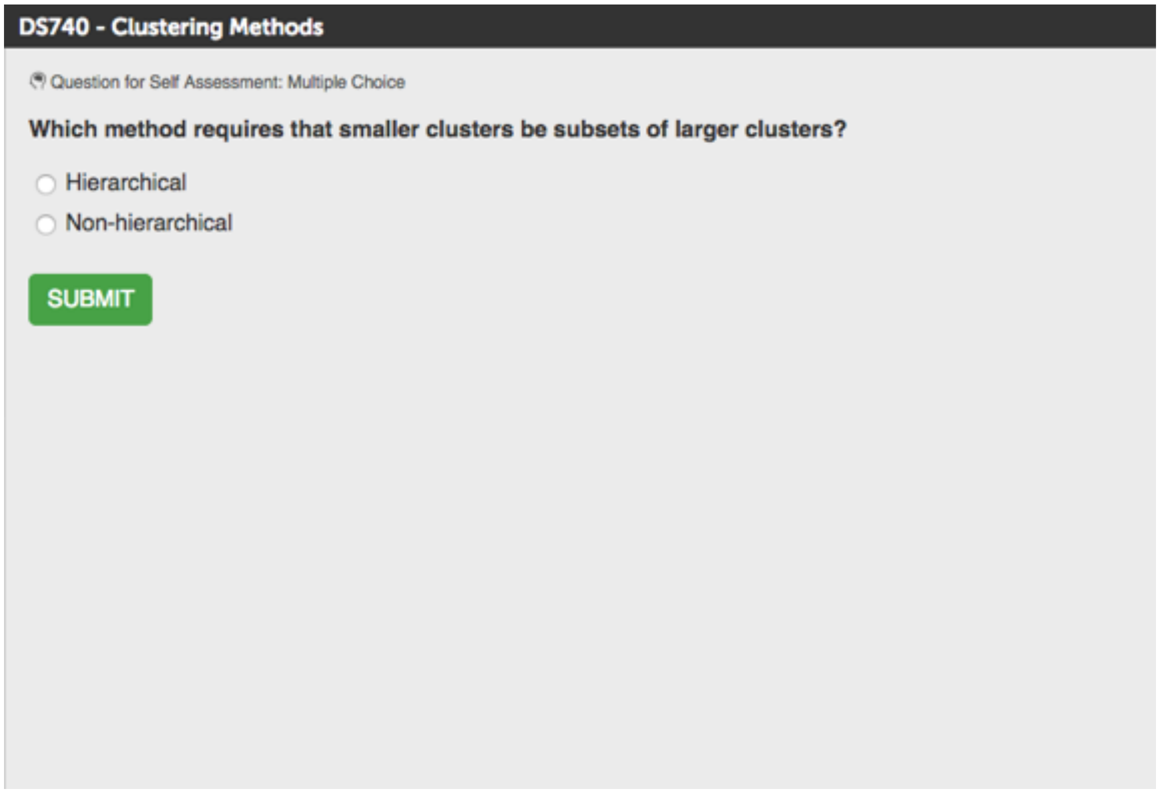
In contrast to this, we have nonhierarchical clustering. Here we see that while the basic groupings appear to be somewhat similar as you move from nine to eight to seven and so on and so forth through the number of clusters or groups, they don't always maintain the same sets.

The red set in the lower left corner is pretty much set as specified apart from the rest of the groups. However, when in the picture with the arrows going from eight to seven, we note that there was a green and a black group, which didn't actually merge, but rather were split into among several different groups.

So all the black observations went along and emerged with a few of the green observations into the purple group, but then a few of the green observations in the eight clusters graph became blue observations and moved into a different group. And a couple of the green observations moved into yet another group.

So that means the purpose for nonhierarchical clustering is to simply put the best-- assessed by some sort of measure-- possible groupings together regardless of what happened with a larger number of clusters, or a smaller number of clusters. Pick the number of clusters and group the data the best possible way with that particular number of clusters.

Question 1


DS740 - Clustering Methods

Question for Self Assessment: Multiple Choice

Which method requires that smaller clusters be subsets of larger clusters?

- ◯ Hierarchical
- ◯ Non-hierarchical

Answer is at the end of this transcript

Hierarchical Clustering: Description

Hierarchical clusters can be merged or split, but not re-formed.

- Typically "bottom-up" (agglomerative) approach: initialize each observation within own cluster.
- Choose a distance measure to compare elements.
- Decisions to merge based on dissimilarity, or "linkage", measure.
- Clusters can merge; once two elements are in same cluster, they will appear in all larger clusters "above" that.

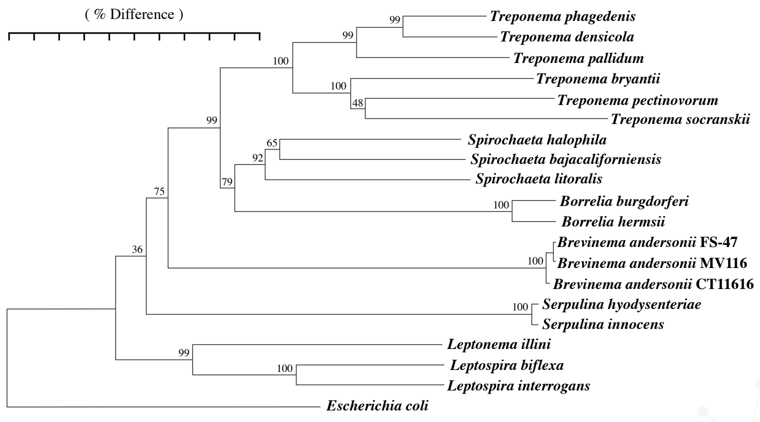
As visualized previously, the hierarchical clustering methods all share the same pattern regarding how clusters are formed. Specifically, smaller clusters starting, in fact, with each element in its own cluster, are consecutively merged to get larger and larger clusters.

Which clusters merge depends on some sort of way of measuring distance, as well as on a way of defining closeness or dissimilarities via a linkage measure. The linkage measure is typically defined so that clusters that are more dissimilar or further apart have higher values.

Hierarchical Clustering: Visualization

Entire process is visualized via a **dendrogram**, a tree-type plot of the steps of the clustering process.

- Cluster selection viewed as cross-section of dendrogram.



This process of grouping the clusters into larger and larger clusters, or building them up, can be visualized as a dendrogram, or a tree type of plot. Cluster selection is then visualized as a cross-section of the dendrogram.

Notes:
Image by Rgarre4 - Own work, CC BY-SA 4.0,
https://commons.wikimedia.org/w/index.php?curid=45099695

Hierarchical Clustering: Distance

Euclidean distance: square root of the sum of the squared distances between variable values of two data points.

Manhattan distance: sum of the absolute distances between variable values of two data points.

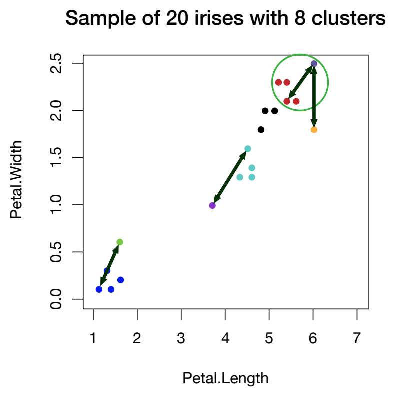
Compute distances between all pairs of data points.

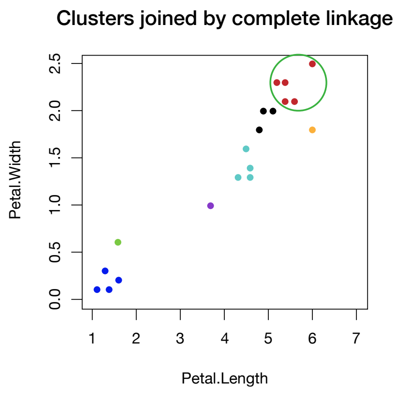
We can consider a variety of types of distances, but the two primary ones that will work with our Euclidean distance and Manhattan distance. Euclidean distance has been previously introduced as the square root of the sum of squared distances between the different variable values for the two data points.

Manhattan distance, is, in contrast, the sum of the absolute distances between the variable values of the two data points. In order to compare potential clusters, we must have distances between all possible pairs of data points, or all elements. So this involves a computation of essentially a matrix of distances for which R has a useful function called dist.

Hierarchical Clustering: Complete Linkage

Complete linkage: measure dissimilarity between clusters 1 and 2 as the **maximum** distance between all pairs of observations between the two clusters.



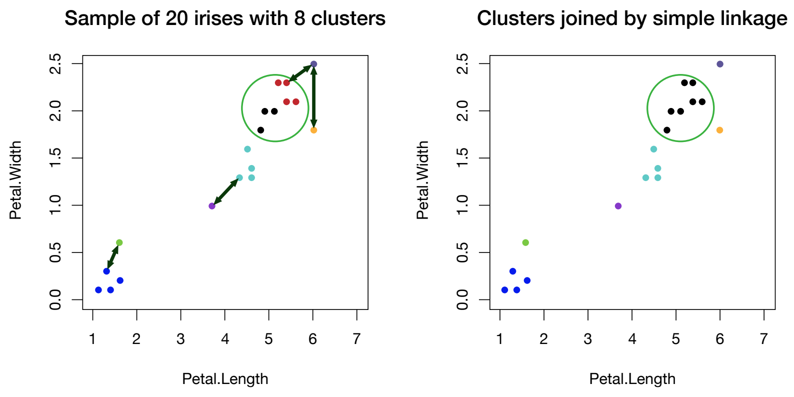


These distances are examined for all pairs of observations between two clusters, and some sort of linkage measure is used to select the closest clusters to merge. The first kind of linkage is known as complete linkage, and it measures the maximum distance between any of the observations between two clusters.

Thus, this method defines dissimilarity by the furthest observations between two clusters. And we again refer to dissimilarity, since larger values of this linkage measure mean more dissimilar values.

Hierarchical Clustering: Single Linkage

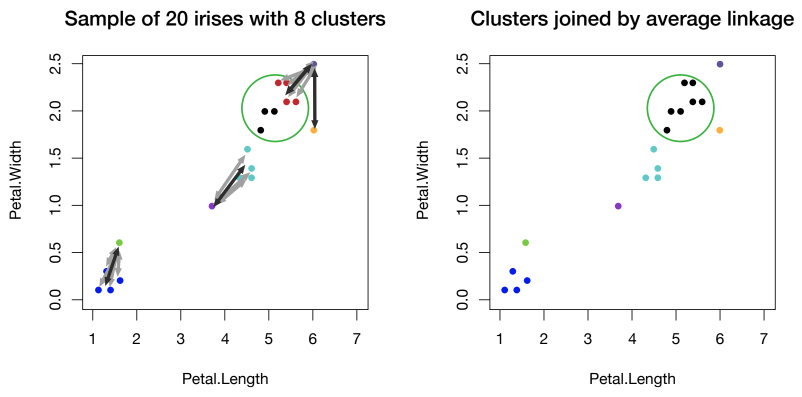
Single linkage: measure dissimilarity between clusters 1 and 2 as the **minimum** distance between all pairs of observations between the two clusters.



The next method-- single linkage-- identifies dissimilarity by the closest observations between two clusters. That is, we look at the minimum distances between any pair of observations from two clusters. A consequence of this type of linkage is that the clusters often tend to get strung along in a thin, long cluster in dimensional viewing. We will see this in our Iris example later. Note that that does not work well for cloud type clusters.

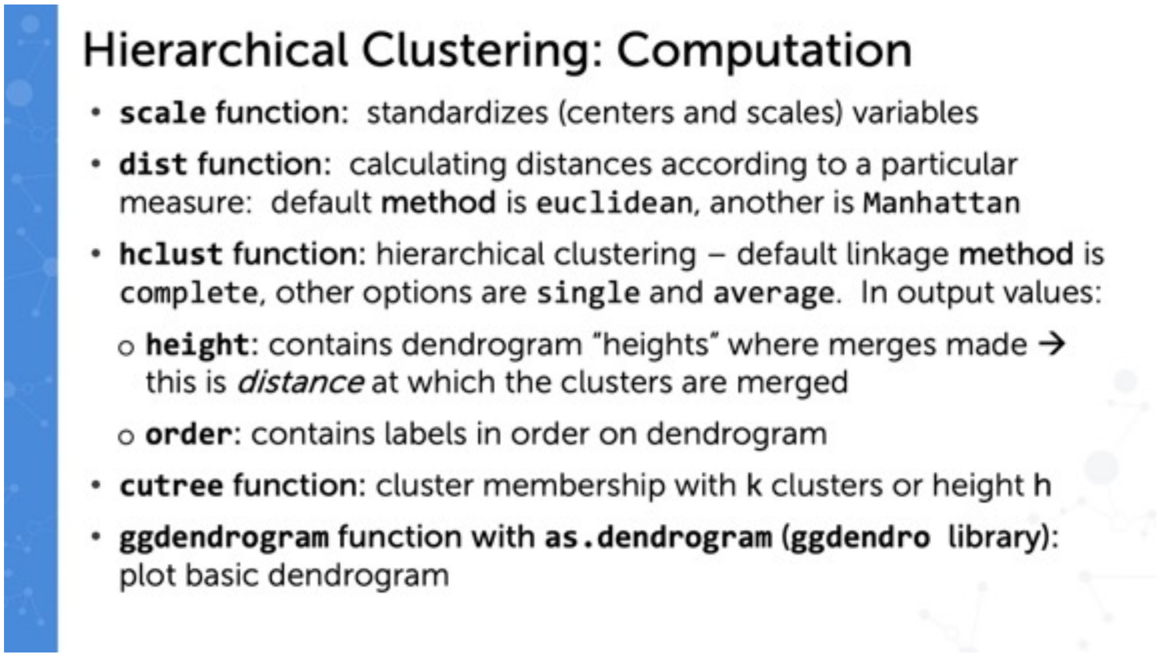
Hierarchical Clustering: Average Linkage

Average linkage: measure dissimilarity between clusters 1 and 2 as the **average** distance between all pairs of observations between the two clusters.



The final linkage method discussed identifies dissimilarity of clusters by the average of the distances among all pairs of observations between any two clusters. This merges clusters that have the lowest average distances between their elements. It tends to be good for cloud type data.

Notes:
Average linkage is different from Centroid linkage, which connects the centers and can have inversions where the order in which clusters link happens before the clusters “enter” consideration.



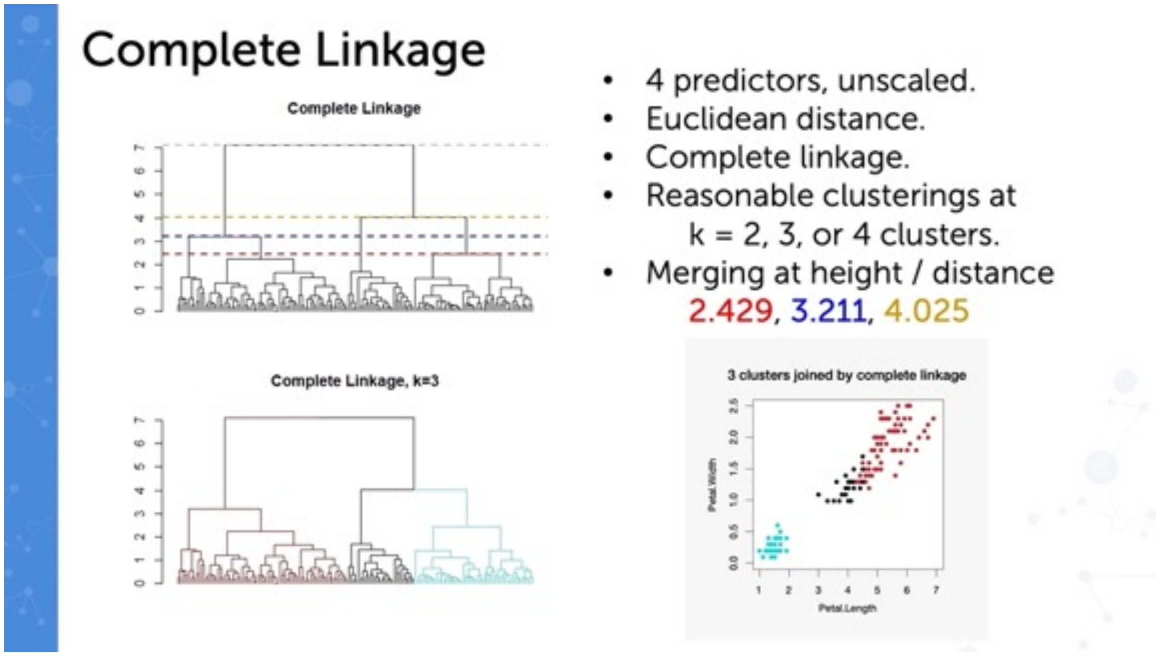
Hierarchical Clustering: Computation

- **scale function**: standardizes (centers and scales) variables
- **dist function**: calculating distances according to a particular measure: default **method** is **euclidean**, another is **Manhattan**
- **hclust function**: hierarchical clustering – default linkage **method** is **complete**, other options are **single** and **average**. In output values:
    - **height**: contains dendrogram "heights" where merges made → this is *distance* at which the clusters are merged
    - **order**: contains labels in order on dendrogram
- **cutree function**: cluster membership with **k** clusters or height **h**
- **ggdendrogram function with as.dendrogram (ggdendro library)**: plot basic dendrogram

For hierarchical costing, after using the scale function to standardize our variables, we will need commands for both the distance computation as well as to fit the hierarchical clustering for different linkage functions. This hclust function consecutively merges clusters and builds them into larger and larger clusters in a hierarchical process. The distance function allows for both Euclidean distance and Manhattan distance. And it is a function that is very well-connected and made use of in that hierarchical clustering function.

The three possible types of linkage are complete, single, and average as possible values for the method argument. As part of the output from the hclust function, we can identify the height at which the splits are made or really thinking about this as the distance at which the clusters building up from the bottom are merged.

And because sometimes it's difficult to visualize the labels in our dendrogram, which can be made by the ggdendrogram function, we can also pull the order from the output of the hclust function. One final important function is the cutree function, which identifies cluster membership at key clusters or height h.



Complete Linkage

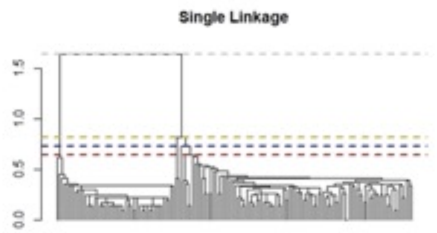
- 4 predictors, unscaled.
- Euclidean distance.
- Complete linkage.
- Reasonable clusterings at k = 2, 3, or 4 clusters.
- Merging at height / distance 2.429, 3.211, 4.025

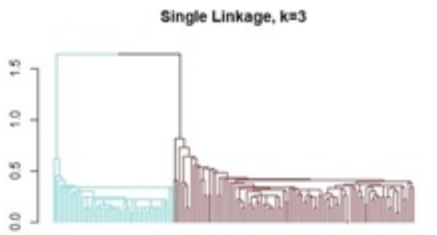
Visually, cluster selection can be considered as a visual cross-section of a dendrogram. Here we have the visualization of hierarchical clustering using complete linkage along with Euclidean distance measurement to define the similarity between clusters. In this dendrogram, we think about building from the bottom up as we see initialized at the bottom each observation within its own cluster. For visual clarity, we don't include the cluster labels here.

Those clusters are progressively built into larger and larger clusters, meaning there are fewer and fewer of them. The vertical axis lists the value of the linkage measure for which the merge occurs. And we tend to look at the points that are vertically distant from the prior merge point. So that is, we look for clusterings that took place at a larger distance from the next cluster.

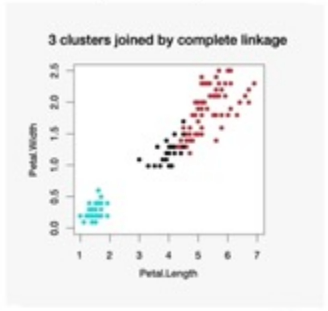
In this case, both two, three, and four clusterings are separated reasonably well from the next closest clustering in terms of a vertical distance. So those would be reasonable numbers of clusters to select. At the bottom of the slide, we visualize one such clustering, specifically the clustering into three clusters with complete linkage. This is shown both on the dendrogram and then visualized in a scatter plot of two of the potential variables used in the dissimilarity measure.

Single Linkage





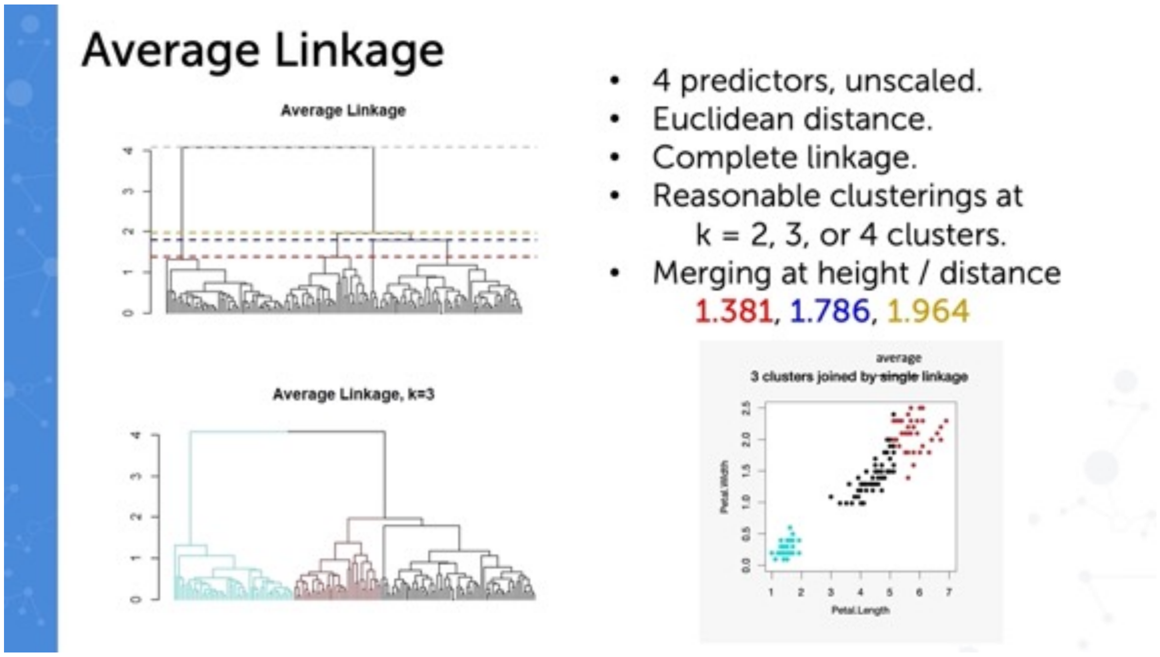
- 4 predictors, unscaled.
- Euclidean distance.
- Single linkage.
- Reasonable at k = 2 clusters
  (less clearly 3 or 4)
- Merging at height / distance
  0.648, 0.735, 0.819



We look at corresponding visualizations for the single-linkage method of combining clusters. Recall that single linkage tends to string along elements. This is visually apparent by the continuing addition of small clusters onto larger clusters in fact often adding just one observation at a time rather than merging of similarly sized clusters.

Here the vertical spacings do not show as clear of evidence for three or four clusters. But it is reasonable that we would use two clusters because of the large vertical distance between where the clusters are merged into two and where they are ultimately merged into one. We thus consider k equals 2 with single linkage to be the most evident type of clustering.

In fact, if we were to look at the clustering where there are k equal to 3 clusters, it shows that one of those clusters tends to-- only includes a couple points. And we are not able to distinguish differing groups, differing clusters, in the upper area of points as seen on the scatterplot. This again has to do with the fact that smaller clusters are tacked onto larger clusters. The single linkage does however seem to pick off the obviously visually separated cluster as shown in the light blue.



Average Linkage

- 4 predictors, unscaled.
- Euclidean distance.
- Complete linkage.
- Reasonable clusterings at
  k = 2, 3, or 4 clusters.
- Merging at height / distance
  1.381, 1.786, 1.964

Finally, looking at average linkage as a way of merging clusters, cross sections are reasonably designated for the merge points of three or two clusters noting that the cross section for designating two clusters has a large vertical distance from the merging into just one cluster again.

However, when we take a look at a visual of the three clusters, we both see a separation between the two main groups, but it also appears to distinguish two groupings in the upper right corner. The furthest of these in the reddish points shows a wider variability than the black points, which is perhaps how the clustering, which is a cloud-type clustering, differentiated these points from each other into those two groups.

Hierarchical Clustering: Linkage Comparison

Complete linkage
Tends to yield more balanced clusters, joined more regularly-spaced across distance measure.

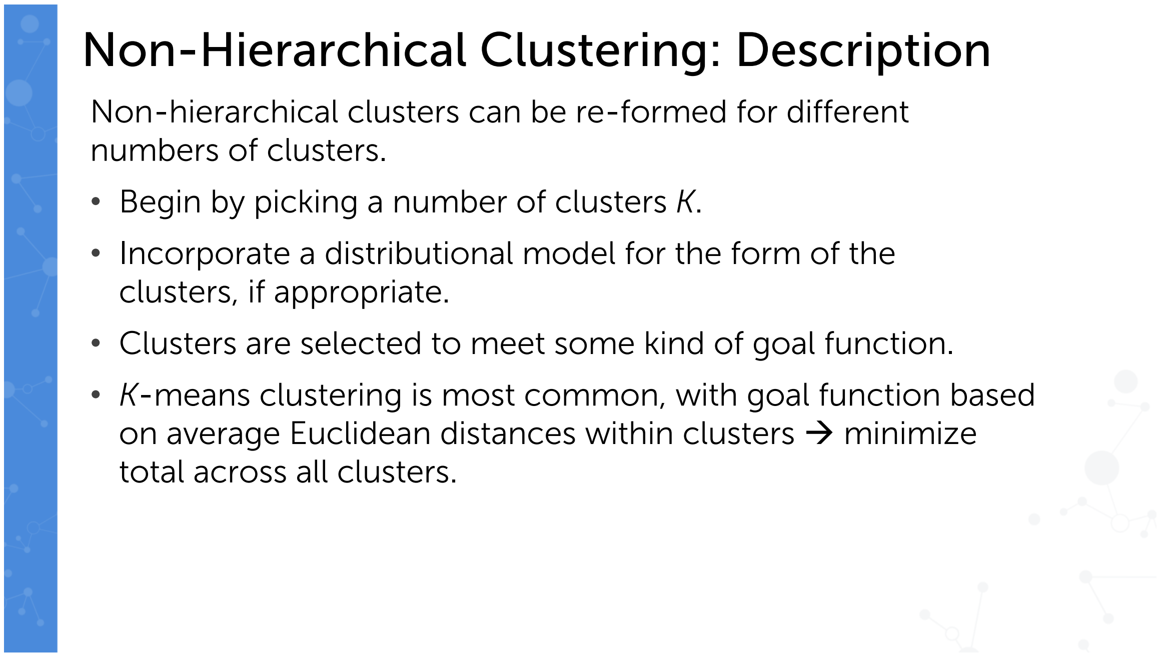
Single linkage
Spatially (across *p* predictors) tends to connect data in strings.

Average linkage
Spatially (across *p* predictors) tends to connect data in ovoid shapes.

Choose based on presumed or visualized behavior of clusters in the space of possible variable values.

The definition of each linkage method helps us identify the spatial behavior of each method. While we can only easily visualize in two or three dimensions, conceptually this extends to higher dimensions for more than three variables. The anticipated spatial behavior of clusters in a particular situation can be used to select a linkage, as well as can visualization, even if done after the different clustering methods are applied.

We tend to view clustering as a back and forth updating process between the method application and what appears to be working best to cluster the data.



Non-Hierarchical Clustering: Description

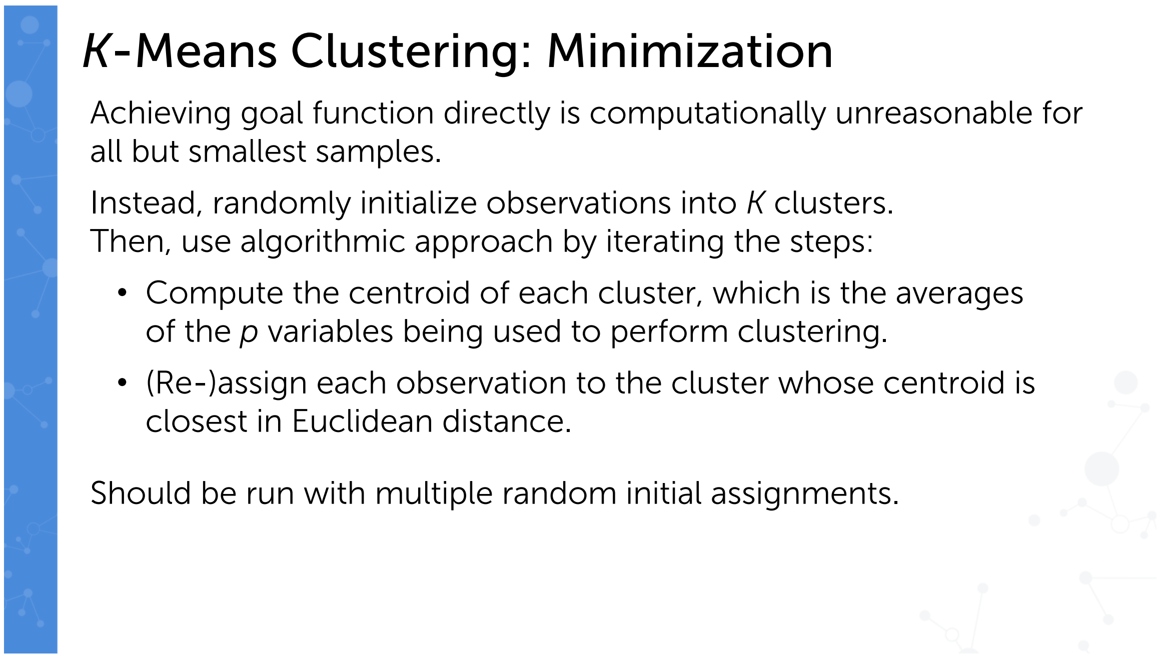
Non-hierarchical clusters can be re-formed for different numbers of clusters.

- Begin by picking a number of clusters *K*.
- Incorporate a distributional model for the form of the clusters, if appropriate.
- Clusters are selected to meet some kind of goal function.
- *K*-means clustering is most common, with goal function based on average Euclidean distances within clusters → minimize total across all clusters.

Our second major type of clustering is nonhierarchical clustering, but at the face of it means that clusters are not hierarchical, meaning smaller clusters do not necessarily get combined to get larger clusters. That is, for different numbers of clusters, we can reform which observations are in which cluster.

The process begins by picking a number of clusters K. If there are any particular model assumptions, such as normal distribution for the form of the clusters-- so a multivariate or bi-variate or normal distribution, that would be designated, as well. And clusters are then defined by how the ones that best meet some kind of goal function.

We'll examine the most common type of nonhierarchical clustering, which is called K-means clustering. And the reason for the K-means portion in the title has to do with the fact that the goal function is based on some sort of average distance within clusters. We're going to take a look at average Euclidean distance within clusters. And we want to minimize the total of those average Euclidean distances across all the clusters.



K-Means Clustering: Minimization

Achieving goal function directly is computationally unreasonable for all but smallest samples.

Instead, randomly initialize observations into *K* clusters.
Then, use algorithmic approach by iterating the steps:

- Compute the centroid of each cluster, which is the averages of the *p* variables being used to perform clustering.
- (Re-)assign each observation to the cluster whose centroid is closest in Euclidean distance.

Should be run with multiple random initial assignments.

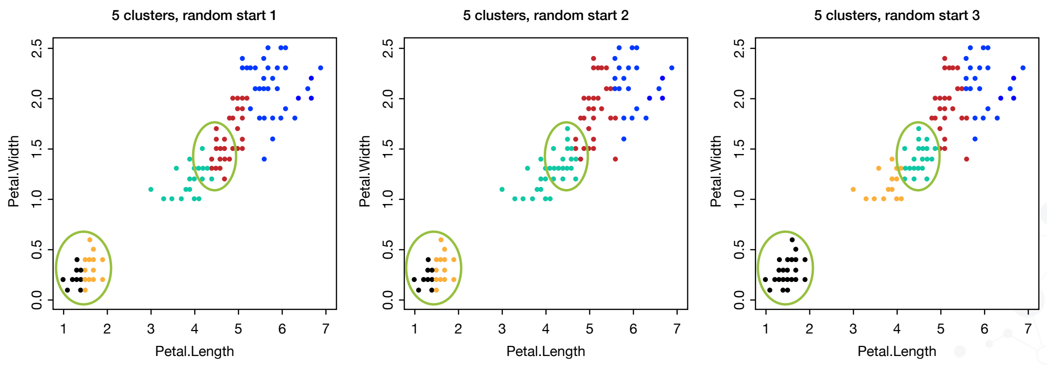
Due to the fact that a true minimisation of the K-means clustering goal function would involve looking at every possible set of clusters-- of K clusters-- this direct achievement of the goal function is computationally impossible with the computer power that we have, for all but the smallest, trivial example.

Thus, an algorithmic approach must be used iterating between a centroid computation for the clusters, and a assignment of observations to the closest centroid. This iteration proceeds until no further reassignments of observations can be done.

But because it is dependent on the starting place, and because it is algorithmic, not the true minimisation, it should be run with multiple random initial assignments. You will see this in our example.

Algorithmic Results, 5 Clusters

Full iris data, using p=2 predictors, into K = 5 clusters.



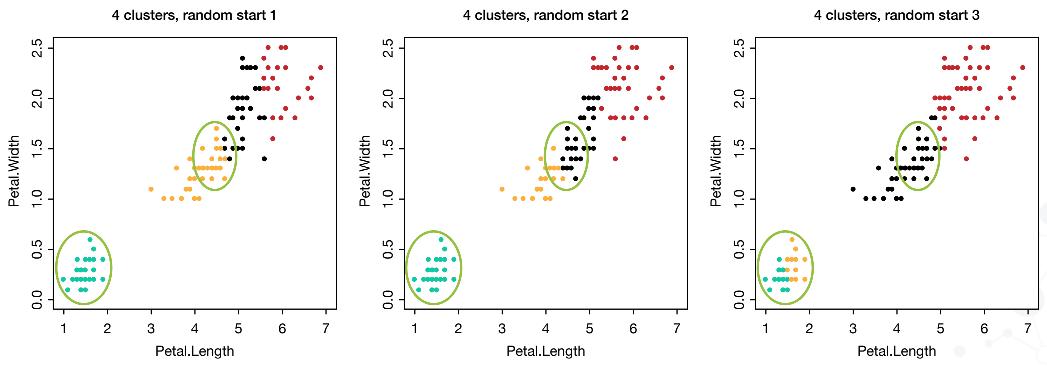
Begin by using the algorithmic approach to group our Iris data on just two of the predictors-- pedal length and pedal width-- into k equals 5 clusters. Remembering that we should initialize at multiple times, we have three random starts, which can be done using the R function. What we would like to focus on in this visualization, is how the clustering that are reached are not consistent.

Specifically, even in the far lower left separated cluster, two of the iterations result in splitting that into two groups, while one of them leaves that as a sole cluster. The other circled area in these plots shows that there is also an inconsistency in how the clusters are split in about the middle of the plot, and shows that there's a shift between the cutoff for the clusters.

In other words, the random starting points don't seem to be getting two consistent clusters. This suggests considering a different number of clusters.

Algorithmic Results, 4 Clusters

Full iris data, using p=2 predictors, into K = 4 clusters.

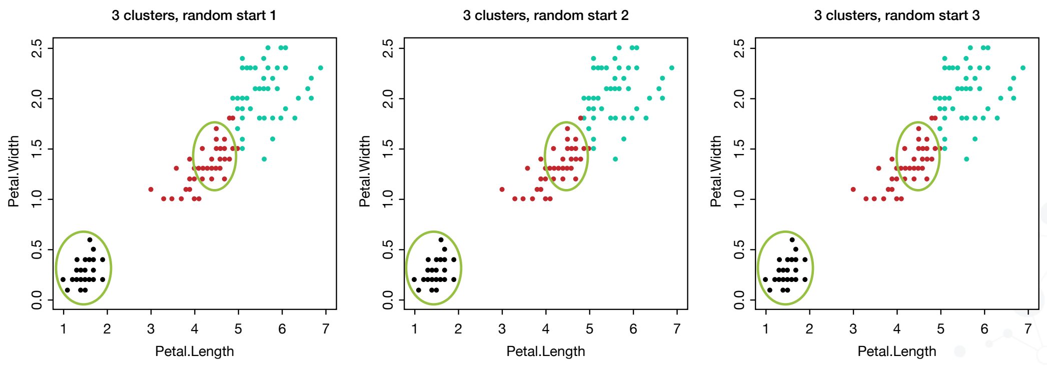


We apply the algorithm now to K equals 4 clusters. Again, starting from three random starts, although certainly in practice it would be wise to start off with more random starts. In this case, we visualize three of them where, again, the lower left cloud of points is sometimes split into two parts, sometimes not, and the mid part of the data cloud also circled is not consistently split.

The upper right quadrant with the red points also has inconsistencies between the three random starts. Thus, we'll consider moving into fewer yet number of clusters--- three clusters-- and see if we get some consistency there.

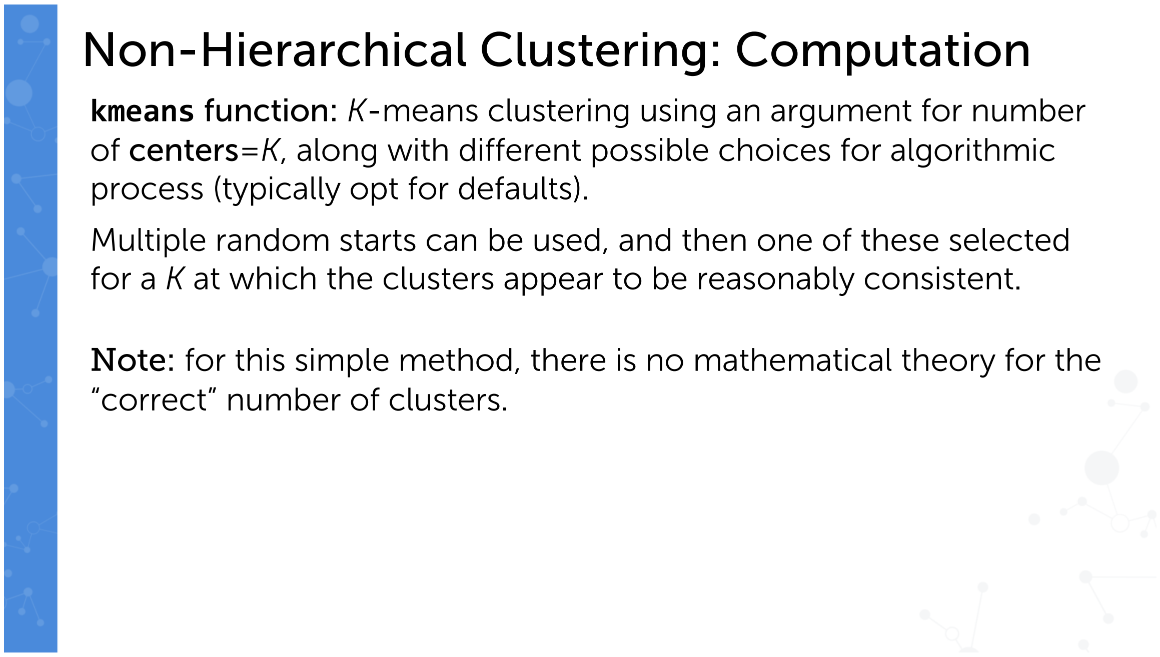
Algorithmic Results, 3 Clusters

Full iris data, using p=2 predictors, into K = 3 clusters.



Here we see that in all three random starts, the algorithm resulted in the sole cluster in the lower left corner-- the strongly separated grouping-- while the previously circled section remains consistently red. That is, there is no major difference in the splitting of the red and blue points. We do notice a point here or there that seems to move back and forth, but the end resulting clusters appear to be much more consistent across the random starts here, and so we would reasonably conclude that three clusters seems to work well for a K-means algorithm.

Now obviously knowing the answer, we understand why this is that those three clusters designate the different species, but in practice if we don't have an end response, we have to let the data guide us.



Non-Hierarchical Clustering: Computation

kmeans function: *K*-means clustering using an argument for number of **centers**=*K*, along with different possible choices for algorithmic process (typically opt for defaults).

Multiple random starts can be used, and then one of these selected for a *K* at which the clusters appear to be reasonably consistent.

Note: for this simple method, there is no mathematical theory for the "correct" number of clusters.

The K-means function is the obvious name for the function that's used to algorithmically fit K-means clustering and requires a selection of the number of clusters, which is denoted as centers equal k-- the number of clusters. And although there are different choices for the algorithmic process, we will typically opt for the default option.

A reminder that multiple random starts can and should be used, and based off the visualizations coming from those clusterings, a particular choice for k, at which the clusters seem to calm down or be rather consistent across the random starts can be selected.

This doesn't always happen. That is, there may not be a obvious K, at which this occurs and even more broadly, there's no mathematical theory for the correct number of clusters in this simple application. There are more complex applications that provide some theoretical foundation for a selection of number of clusters, but they are beyond the scope of this course.

Hierarchical Versus Non-Hierarchical

Hierarchical Clustering

+ Pros: Don't need to pre-choose number of clusters; all possible clusters (across all K) are generated at once.

– Cons: Large impact of initial choices for distance and linkage; can be slow to fit.

Non-hierarchical Clustering

+ Pros: Fast (due to algorithmic process).

– Cons: Need to pick a number of clusters prior to fitting; clusters change depending on random starting point.

So when do we choose hierarchical versus nonhierarchical clustering? This may depend on the amount of time that it takes to run the computation based on the different sizes of the datasets we have available. Nonhierarchical clustering tends to be faster because we aren't looking at all possible clusterings, whereas hierarchical clustering looks at all possible clusters generated at once.

The positive of that is we don't need to pre-select a number of clusters because all will be available to us through the hierarchical visual of the dendrogram. The downside of hierarchical clustering is there can be large impacts of the choices for distance in linkage, which if we have some information about those choices, can be reasonably done.

The cons for nonhierarchical clustering include that we need a number of clusters in order to fit the method, and the very readily apparent result of the algorithmic process and random start that the particular clustering-- final clusters-- may change depending on the random starting point.

Clustering: More Flexible But More Unknown

What is best number of classes?

What is best clustering method? How to define dissimilarity?

Trade-off between number of clusters and complexity of the clustering model.

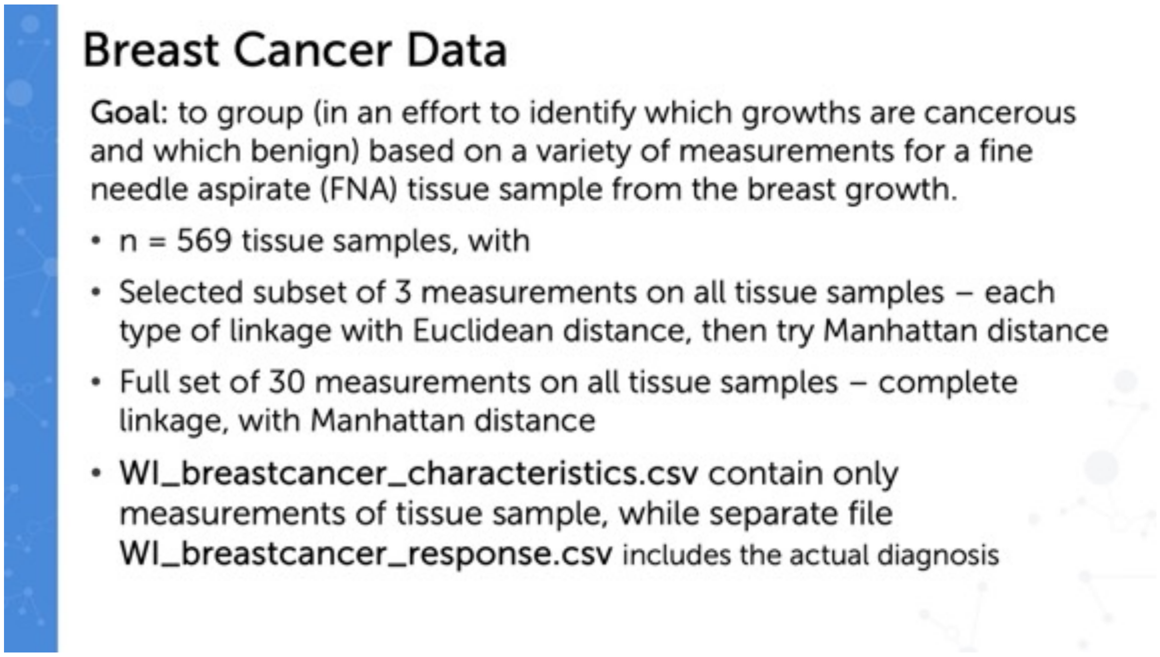
Visuals typically two-dimensional, but variable space may be many more dimensions.

Recommendation: consider clustering as a process iterating between method application and what appears reasonable.

While clustering is a highly flexible and widely applied method for grouping observations into similar spaces, it leaves us with a lot of unknowns, such as the best number of groups or clusters, which is the best clustering method to apply, and what are the parameters and inputs-- the distances and possibly the dissimilarity measures-- that we'll need to use to apply that clustering.

In addition, there can be a trade off if we were to get into more complex models. There is a trade off between number of clusters and the complexity of the model. Visuals can be problematic, as well, since the variable space can be in many more dimensions than we can actually take a look at.

And so sometimes there may be topological or spatial relationships going on that cannot be summarized in a two-dimensional visual. So we typically work with clustering as a process that moves back and forth between applying a method that appears appropriate and seeing how this shows up in the data. So does it appear reasonable? And then moving back to applying the method and working with that method and seeing if that appears reasonable. So it is an interracial, and it is a process, but it does allow us a flexible pathway to a final selection.



Breast Cancer Data

Goal: to group (in an effort to identify which growths are cancerous and which benign) based on a variety of measurements for a fine needle aspirate (FNA) tissue sample from the breast growth.

- n = 569 tissue samples, with
- Selected subset of 3 measurements on all tissue samples – each type of linkage with Euclidean distance, then try Manhattan distance
- Full set of 30 measurements on all tissue samples – complete linkage, with Manhattan distance
- WI_breastcancer_characteristics.csv contain only measurements of tissue sample, while separate file WI_breastcancer_response.csv includes the actual diagnosis

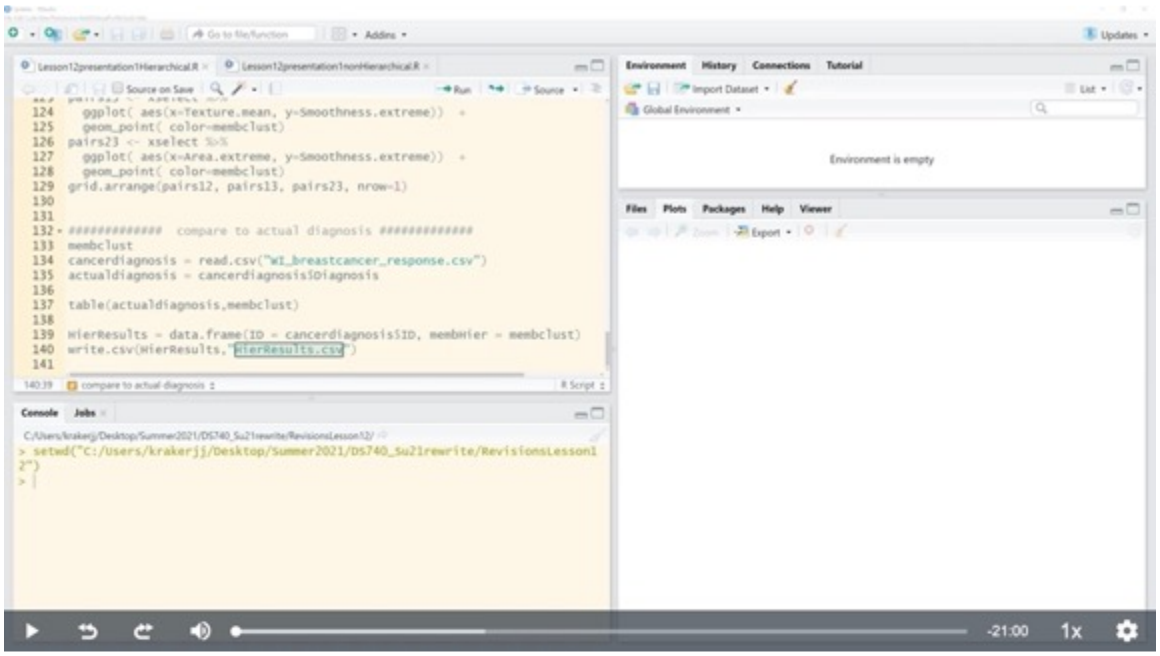
On this slide, we introduce the breast cancer data, which discusses a study of 569 tissue samples. We are going to try to group the tissue samples into alike clusters based on a variety of approximately 30 measurements or characteristics from a fine-needle aspirate tissue sample taken from the breast growth. There is a summary file attached if you wish to take a look at some of the definitions of these measurements.

There are a full set of 30 measurements, which we will ultimately consider. But we're also going to take a look at a selected subset of three measurements to start with, because this will help us get a little bit better visual of some of the clustering by taking a look at scatter plots made between two selections of those measurements taken at a time, and take a look at how the clusters are spread across the values of those different measurements.

Notes:
Public-source data: wdbc.data
Public-source data description: breast-cancer-wisconsin.names

This commonly-analyzed data set appears in a number of publications, including:

- Fraley & Raftery, 2002.
- W.N. Street, W.H. Wolberg and O.L. Mangasarian. Nuclear feature extraction for breast tumor diagnosis. *IS&T/SPIE 1993 International Symposium on Electronic Imaging: Science and Technology*, **1905**, p. 861-870, San Jose, CA, 1993.



```
  ggplot( aes(x=Texture.mean, y=Smoothness.extreme)) +
  geom_point( color=membclust)
pairs23 <- xselect %>%
  ggplot( aes(x=Area.extreme, y=Smoothness.extreme)) +
  geom_point( color=membclust)
grid.arrange(pairs12, pairs13, pairs23, nrow=1)


##############  compare to actual diagnosis ##############
membclust
cancerdiagnosis = read.csv("WI_breastcancer_response.csv")
actualdiagnosis = cancerdiagnosis$Diagnosis

table(actualdiagnosis,membclust)

HierResults = data.frame(ID = cancerdiagnosis$ID, membHier = membclust)
write.csv(HierResults,"HierResults.csv")
```

```
C:/Users/krakerjj/Desktop/Summer2021/DS740_Su21rewrite/RevisionsLesson12/
> setwd("C:/Users/krakerjj/Desktop/Summer2021/DS740_Su21rewrite/RevisionsLesson1
2")
> 
```

As described on the previous slide, we are going to be taking a look at the Wisconsin breast cancer characteristics. There are 30 columns in that data set that we'll be working with and we'll need to do some organization as well as set up some visualizations, unless these are the four required libraries, or packages, that we'll be loading.

After reading in the data, we can take a look at the names of the various variables, noting that ID is simply a label for an observation. So we'll want to take that off the front before we start to do our clustering. And we can see that we-- at least, in our original data set-- have 569 observations with 31 including ID characteristics. So we'll just use dplyr commands to pull out of cancer. We'll pipe that through, selecting the removal of the ID.

It's also going to be useful to take a look at just a select few of those measurements. And the three we pick are based on prior studies that observed these to be useful measurements to help distinguish groupings among the observations. So we'll define xselect to be just those three columns. We'll define some dimensions, n, p, and pselect. And we can take a look and make sure that matches with what we intend.

Now, the selection of only three variables here comes from a prior statistical study which shows that these particular measurements are good indicators of the diagnosis of cancerous versus benign in these tissue samples. Alternate origination of such a selection might come from a cancer specialist, who has seen, in their practice, that these particular measurements are most indicative of whether a growth is benign or malignant.

In other words, suppose we have some informed reason to narrow it down to these three predictors. And we'll take a quick look at the scales of these predictors, piping the selected the selected data frame into ggplot, picking out each of the variables as the x in turn for each of our separate plot designations, and picking out some different colors to help visualize them.

So I've defined the three plots, and I'm going to put them in a grid, where that grid.arrange is a command coming from the grid extra package. And that just helps us align these plots. So when we take a look at these, well, they certainly do appear to be slightly different shapes, but that's not what is really impactful.

What is important to observe here is the scale of measurement. So texture means are on a scale in the tens. Area extreme is on a scale in the thousands, smoothness extreme on the scale of ones. And in order to allow so that one of these-- particularly, area extreme, doesn't overwhelm the others, we are going to scale our measurements.

So we'll scale both the full set of measurements as well as our selected three measurements. Our next step is to fit our hierarchical models. And we'll consider both Euclidean distance, which is defined by the distance of our measurements. In this case, we've got scaled measurements. And in this case, we actually have two sets of measurements, just the 3 scaled measurements and all 30 scaled measurements.

So we're going to define our usual Euclidean distance as well as a separate distance measurement, which I'm going to denote by adding on a .man for Manhattan. So that looks at, effectively, an absolute value rather than a square root of the sum of squares. Manhattan distance is not as strongly affected by big differences. And that'll become important to note as we take a look at some of the scatter plots showing relationships between these measurements.

So let's start by taking a look at complete linkage using just our selected three variables scaled. And so we have the distances computed for those scaled three measurements. On that distance matrix, we're going to apply the hclust function using complete linkage. So that's the entirety of the application of each clust.

I'm also going to define a label here just for use in plotting. It's a text string called "Complete Linkage". And what this will allow me to do is later on when I am using different methods, it will allow me to change the designation but still use the same basic plotting commands.

So hc.fit includes my clusterings. I can take a look at the entirety of my hc.fit height, which includes all the different heights at which clusters are merged.

And these other designations I think will be visually helpful to see when we observe our actual dendrogram.

But the hc.4321 takes the heights and peels off the last four of them. And because we can only merge 568 times, we are going to take a look just up to the last value of these, which is actually n minus 1, not n. We're going to take a look from n minus 4 up to n minus n, which are the designations of the heights at which the clusters are merged into four or three or two or one. So that's the name of it.

I also take a look at the average heights between those. And again, this is going to be visualized. This is just for visual effects, so I'm not going to go into any detail about the hc.average. We could also, after defining these, take a look at just those last four heights.

And before we take a look at the meaning of the cluster labels, I will define those and then make the dendrogram and talk about the values with respect to that. We'll back to the obtaining cluster labels once we've seen our dendrogram. So visualizing in a dendrogram, we have to use as.dendrogram to make the fitted clustering into a form appropriate for use in gg.dendrogram.

So as, force it to be a dendrogram type. And we then make a dendrogram. Rotate equals false, labels equals false. The important one of these is labels equal false so that we don't have a very messy labeling at the bottom of our dendrogram. We have too many observations to have them all labeled.

We have our title as previously defined. And then I'm going to add some lines in here. And these lines are going to designate the heights-- or equivalently, the distances-- at which complete linkage merges into four and three and two and one clusters, respectively. And so we run the dend.merge and we get our dendrogram here.

Visualizing the height as the location at which we merge into one, two, three, four-- so this red line, this designates the distance for complete linkage at which we merge from 1, 2, 3, 4, 5 clusters into four, because these two clusters are merged into one. And so anywhere between this red and blue line we have, any distances between those red and blue lines, we are going to maintain at four clusters. At any distance between the blue and yellowish or goldish lines, we're going to designate three clusters, 1, 2, and then these two got merged into one.

So it can be useful to consider identifying the clusterings rings at a particular height. That height will occur anywhere between the points at which the merges happen. So between red and blue heights, we could pick a distance of around 6.2. And at that height, at that distance between clusters, we have 1, 2, 3, 4.

At a height of what looks to be around 7, this light green, we have 1, 2, 3. And at the height of this kind of grayish-gold line, we've got 1, 2. And you'll notice I made these lines thicker, because there's actually a range of values at which we can designate the clusters.

So how do we pick off the cluster labels? Well, obtaining cluster labels, again, just taking a look back at these heights with a little bit more detail, I could pick out the idea that I want to designate two clusters, the reason for that being if I only wanted to differentiate between benign and malignant, and I knew that ahead of time.

If that was the case, I could either define my membclust, which is simply a list of 1's and 2's, cluster's 1 and 2 designations, and simply designate the number of clusters. Or I could use cutree at an appropriate height. In this case, that height would have to fall somewhere between 7.05 and 8.058, approximately.

That is at some distance measurement between those two values, at any distance we're not going to be doing any further merges and we're going to be maintaining exactly two clusters. So I could, alternatively, cutree from my hierarchical clustering fit at a height of 7.56, which is about right there. That's that grayish-gold line.

All right, so that would be complete linkage. I could also consider different numbers of clustering. Well, let's take a look first at two clusterings. And how I'm going to do that is I'm going to make scatterplot pairs of the three selected variables. I'm going to run my three variables into ggplot, selecting one of each to be on the horizontal and vertical axis. You could switch those around if you would like to.

But what I'm trying to do here is get every pair of measurements together and then color them according to the member cluster. And I'm going to put those together in a grid so that I can see these side by side. It does appear that there are visual groupings, as clustering must do across the dimensions, these plots, aligning two variables in any scatter plot organizing these two variable values.

So nothing too surprising pulled out here. Although we might, for example, note that the one grouping in red appears to take pretty across-the-board values of area extreme, but seems to focus on lower values of smoothness extreme. So just something to keep in mind in terms of identifying values of our different characteristics or different measurements.

We can also define clusterings, for example, instead of k equals 2 clusterings, I could take a look at three clusterings. Now, obviously, this wouldn't correspond to the benign versus malignant designation. But it could group us into those that are most concerning, somewhat concerning, and not concerning, if we are trying to do a visual inspection.

And so after defining those members, I could again visualize this, the scatter plots, into these three groupings. And we see similar sort of clustering, say, over here, except that these lower values are actually now split-- these lower values in terms of smoothness extreme, which were previously in red, are split into two groups, a green and a red group.

We could do something similar where our member clusters are, say, four, most concerning, not as concerning, less concerning, not at all concerning. So same sort of idea here. And again, run our visuals, and we're going to get our set of scatter plots taking a look at those different groupings.

So some of how we identify the concern for these groupings would need to be done with expert information. So for example, if it is smoothness.extreme, if it is the smoothness measurement is lower, is that concerning or not concerning? And that would be something where we want to have the expert opinion of a doctor to talk about that. Our only purpose here is to group and to try and find some organization into different subsets or into different groupings, into different clusters of our data.

So going through all of that was only accomplished for our complete linkage using three variables and Euclidean distance. We could do something very similar with Manhattan distance. So for example, if I started out this way and ran all my commands and then visualized my dendrogram, when I run this with Manhattan distance, I see again a grouping, a dendrogram that allows me to pick any number of clusters I want to as it progressively builds up merges, as we go up in our complete linkage, distance compared via complete linkage.

However, it looks here like we are getting slightly fewer observations if we put this into two rather than a higher number of groupings. We could again visualize how this looks. Now, important to note here, the color designation has changed. And the reason for that is there is no specific way in which one and two are designated.

That is, on one clustering application, some of the first set of values that we considered in a previous clustering application might have been labeled as ones in that first clustering application and now are labeled as twos, simply because of how the clusterings are built. And so that's why you see this, what looks almost to be like a reversal of the colors. It's just that the cluster number designation is rather arbitrary.

So we've done that for Manhattan distance now. And one very clear observation over here is now, smoothness extreme actually does not seem to make much of a distinction in terms of our two clusters, but very extreme does seem to pull out to-- seems to be the most important effect on our clustering. So that's something, again, that we'd want to consider an expert's opinion, on what is the more meaningful one and that would help us inform what might be the better clustering technique here.

We could do a similar sort of thing with single linkage. So if you open this up, run a very similar set of commands for single linkage, and that single linkage using Euclidean distance, visualize a dendrogram. And after the dendrogram, which really does not look useful at all in terms of, particularly, getting two potential clusters-- or really, clusters of any meaningful size but fewer numbers of clusters.

So single linkage does not seem to be useful, as is the case if we're not just stringing observations along into smaller clusters. Average linkage, again, you could run this. But average linkage doesn't seem to be any more-- any better in terms of clear groupings than complete linkage.

So it seemed like complete linkage with Manhattan distance-- and if I go back to a prior visual-- because of some of the more extreme outlying values of some of our measurements, it's probably preferred to use Manhattan distance. And so ultimately, I'm going to do a complete linkage with p equals 30 and a Manhattan distance, so running this set of commands to get my hc.fit and my membclust, I can form my dendrograms or dendrogram with lines drawn across it.

Now, taking a look at two groupings, I get still one larger group and one smaller group, but it's a lot smaller. And I'd like to have some idea of the percentages of benign versus malignant if such information is available to try and decide if this is approximately the right thing to do. And I'm going to show you that here at the end of this.

So with all 30 measurements-- just visualized across these three measurements, because we can't visualize all 30 measurements, that would get quite messy-- we can see pretty clear clusterings, even among these three measurements. So the last thing I want to talk about is, how would we know where to go with these clusterings?

Well, as I said, one piece of information that could help us make that distinction is if we knew the approximate breakdown into groupings, like approximately what percent of such-- what percentage would be benign versus malignant? Of course, if we had the actual responses-- that is, the actual diagnosis of benign versus malignant-- we could compare our clusterings to that actual diagnosis. And that is contained in our response CSV file, under the diagnosis variable.

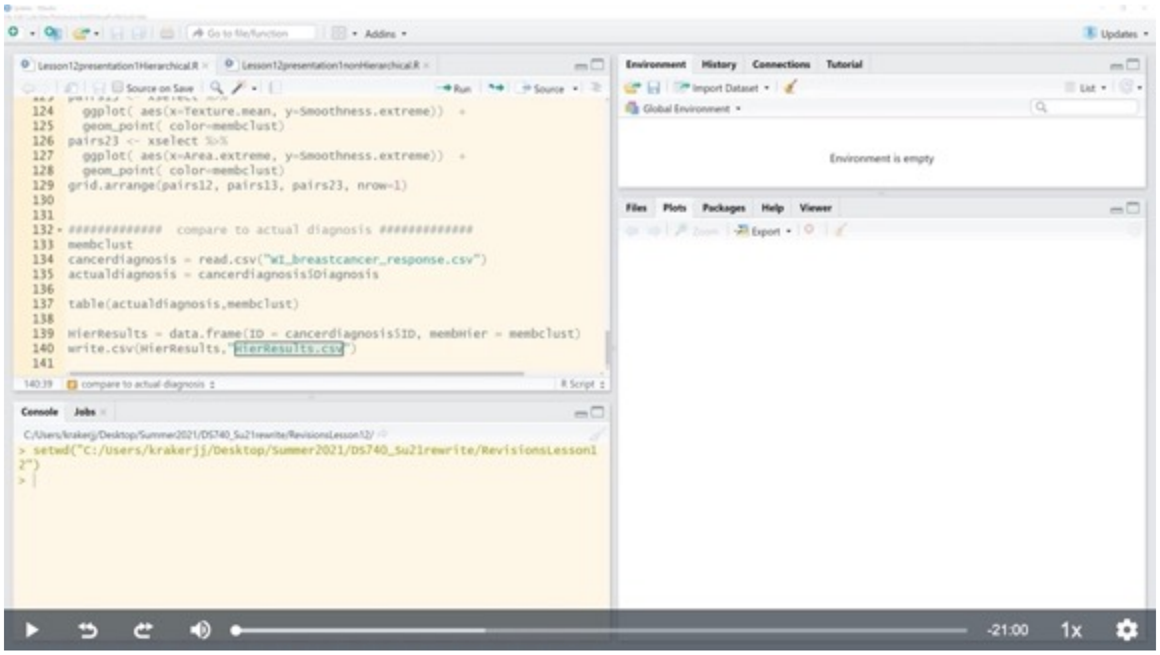
So if I make a table of the actual diagnosis versus member clustering, I can see that benign are almost all grouped into what is one of the clusters, happens to be labeled as 2-- again, this labeling of the two cluster numbers is pretty arbitrary. And a good majority of the actually malignant are grouped into the first cluster.

Now, concerningly, there are quite a few that are not that are put into the second group. So we might want to take a closer look, perhaps, at additional measurements, any other input that could be helpful in designating this, because we don't want to misdiagnose that many malignant, just from this visual scan.

But what this could be helpful with identifying is saying if we are grouped into those among the second cluster, nearly all of those-- a good chunk of those in the second cluster were benign. And in our first cluster, nearly all of those were indeed malignant. So if they're in the first cluster, you really want to have them go on to further testing. If they're in the second cluster, still do further testing, but it encouraged that in the second cluster we see much more benign.

We will examine non-hierarchical clustering in the next recording.

Notes:
See the online course for a downloadable R file containing the set of commands used in this demonstration.



```
  ggplot( aes(x=Texture.mean, y=Smoothness.extreme)) +
  geom_point( color=membclust)
pairs23 <- xselect %>%
  ggplot( aes(x=Area.extreme, y=Smoothness.extreme)) +
  geom_point( color=membclust)
grid.arrange(pairs12, pairs13, pairs23, nrow=1)


##############  compare to actual diagnosis ##############
membclust
cancerdiagnosis = read.csv("WI_breastcancer_response.csv")
actualdiagnosis = cancerdiagnosis$Diagnosis

table(actualdiagnosis,membclust)

HierResults = data.frame(ID = cancerdiagnosis$ID, membHier = membclust)
write.csv(HierResults,"HierResults.csv")
```

```
C:/Users/krakerjj/Desktop/Summer2021/DS740_Su21rewrite/RevisionsLesson12/
> setwd("C:/Users/krakerjj/Desktop/Summer2021/DS740_Su21rewrite/RevisionsLesson1
2")
> 
```

In this recording, we continue to examine the Wisconsin breast cancer data as introduced on the previous slide. If you did not write out the results from the hierarchical clustering into a CSV file, please feel welcome to use the code provided to do so. We're going to be comparing these hierarchical results to our analysis today, which is going to use non-hierarchical clustering methods, specifically the one introduced in this presentation, the k-means method of clustering.

We start as we did for our non-hierarchical clustering. We start as we did previously, using the same libraries, reading in our data, making sure that we're set to the correct directory. And with that, we'll be able to read in the data on the 30 dimensions, the 30 measurements of interest.

And we'll remove the ID as well as select just three of the measurements to be included in a smaller data frame. We'll define our dimensions. We'll scale our x and-- our full x and our selected variables. And then we'll go on to k-means clustering.

Because we are doing non-hierarchical clustering, by definition, there is not going to be a hierarchical building, a dendrogram, of the results. And so there's no mingling of the clusters into higher level clusters. Thus, the visuals are going to be limited to the individual clusterings, which can be displayed in something like tables or scatter plots between two variables at a. Time Hence, to start with, it is useful just to take a look at these three selected variables to get an idea of how the clusters are, in fact, operating.

Additionally, we need to start with a random seed when we do our clustering. And we're going to try clusterings of size 2 and 3 and 4. In fact, we'll start with four clusterings. Because the k-means function involves a random component, we need to-- to get the same sort of results-- start at the same place. And so before we run that k-means command, we're going to set the seed. I picked 12 for lesson 12.

So k-means is applied to the scaled three measurements using the number of clusters that I specified-- like I said, I'm going to start with four-- and then pulling off the cluster name. So if I take a look at what's in membclust where I stored the results of clustering A, it's a list of labels one through four.

As noted, we want to aim to visualize this. So I'll use the three scatter plots on our pairs of our three selected variables as we defined previously and put them side by side and hopefully open up a little bit more space, which I'll be right back.

After having rerun my commands, I should have come up with the same plot that you now have on your screen. We can see that area extreme in particular tends to be useful in pulling off this group that in this clustering is colored as black. Now, it's important to understand that a new clustering might have a different order of labels and might have slightly different assignment of labels to observations.

And what we're looking for is a relatively consistent labeling, or grouping, I should say. The labels might change up the numbers, and the colors may change up. But, overall, the same observation should generally be put into the same cluster or grouping.

So one way we can take a look at this is to rerun the k-means command, restore the results of that clustering in membclust, and then rerun our visuals. So rerunning the assignment with color equals membcluster should, in theory, pull all three of these up. And I'm going to try that one more time here.

So you'll notice that, in this clustering, this top clustering is now in blue. It's still effectively the same observations, or similar observations, but it's a different color because it was a different label because of slightly different random start. With this many observations, it's going to be tough to see visually whether we're getting consistent clusterings, especially with these four groupings.

While it might be easy, for example, to see this clustering in black here in this middle plot, it seems to be pretty well delineated on the dimensions of Texture.mean and Smoothness.extreme. The others are a little bit more intermingled. And so one way of handling this is by doing two clusterings. And I could rerun the visual commands if I wanted to.

But what I'm going to get-- and I'm just going to do a quick visual side by side. So I am going to put clustA and clustB side by side. We notice that, while the labels are different, the ones that were labeled as twos in our first clustering tend to be labeled in threes in our second clustering-- 2, 3; 2, 3; 2, 3.

So again, it's just a matter of a difference in labeling. And this seems to be pretty consistent. If it was a three in clustA, it was a four in clustB for the most part. If it was one in clustA, two in clustB.

Well, again, this is not the most efficient way to take a look at this. So we're going to take a look at a tabulation of how much these clusterings match. And so it's relatively consistent. If it was a four in clustA, it's mostly a one in clustB, but not perfectly.

So we're going to take a look at how often the groupings seem to match up. And so I could take the most common matching in this column-- 92-- plus 208 in the second column plus-- matching in A seems to match most directly with cluster three in our B clustering. And so there are 93 in that matched in that way and 95 here out of our total of 569 observations-- so about 86% matching between these two clustering, which is OK but not great.

To try and get this a little bit more efficient, we can take effectively the max from each column and add those together, dividing by n. And if we do this, we can just take, then, match total divided by n. So this gives me that 86% that I observed visually.

Now, if we do this a few more times-- rerun. And again, every time we run these k-means functions, we're going to be getting a new clustering because we're starting from a new randomized location in our k-means algorithm. So we're going to get different values in clustA and different values in clustB.

We're going to put those-- or tabulate those-- cross-tabulate those. And we get 88% matching, which is, again, pretty good but not fantastic. Oh, here we get perfect matching. So that will sometimes happen. Very good, very good. Not great-- or not bad but not great. Not bad but not great. Same. A little bit better. Really good. Not bad. Really good. So we're getting good matchings but not great matchings.

Well, let's try the same process with three clusterings. And if we proceed through-- again, we could visualize this. And in fact, maybe let's go ahead and do that, initializing our set seed. So you should get the exact same image I get here. We see our three clusterings across our dimensions.

Now, if I make two distinct such clusterings, each into three clusters-- 1, 2, and 3 as the labels-- I get very good matching on this first attempt. Perfect matching. Perfect. Oh, not good. Not good. So I'm not getting extremely consistent clusterings, and that's really what we're looking for here.

So four and three aren't getting exceptionally consistent clusterings. If I go to two-- which, of course, we know that's kind of our number of clusterings that we're most interested in based on thinking of benign versus malignant, and I take a look after having set the seed-- and again, because we set the seed, you should get the exact same image here-- we can see that very extreme isn't quite as meaningful as it seemed to be previously.

It does look like the values that are lower-- or, I'm sorry-- one of the clusterings takes only area extremes that are lower, but there are also some in the other clustering that are intermingled with that-- or in the other cluster that are intermingled with that.

So is this going to be something in which we get consistent results when we do multiple applications of this algorithm? So let's take a look at this-- clustA, clustB. Run this through a table match. This actually looks really quite good. In fact, if we take a look at the percentage match, 98%-- well, that looks good, but we really only tried that once. Well, this time, we got perfect match between the two clusterings. Perfect, perfect. Very close to perfect.

What we're seeing here, as we continue to run this, is when we do clusterings starting at different random centers, we are actually winding up between our two different clusterings with extremely consistent results. So this suggests that two clusters is a pretty good one to settle on. We're getting consistency when grouping into those two clusters.

All right. So we decide on two clusters, set the seed, decide on our membership as k-means application after setting that seed to our full-scale data. So we're going to go back to our full 30 measurements. And you could go through this and verify that. Indeed, we're still getting consistency when we base this on our full 30 measurements.

And then save those or rename those as membclustNonH. The reason I'm doing that is I additionally want to compare these results to the clusterings we got with hierarchical. So reading that in after I find my directory-- so reading in the CSV file and assigning the cluster membership to be called membclusterHier.

As always, it might be useful to take a look at what is in each of these. So it's a list of one and twos in my non-hierarchical cluster membership. And so I tried to make a name that identifies what I am placing in here. membclust from hierarchical clustering, as we did in the previous slide, is also a list of ones and twos.

And if we make a cross-tabulation of those-- now, in this case, it just so happens that the ones and twos match up, but we're really looking for matching clusterings. And so what we see here is quite a bit of consistency in terms of our clustering. But there are these 33 observations that were put into the second group for hierarchical clustering but were placed in the first group for non-hierarchical.

Well-- so that doesn't really give us a great comparison between these two methods, again, unless we had a prior idea of the sizes of clusters that we would anticipate or if we had the true diagnoses, which, in this case, we do kind of as an aside. We have those separate cancer diagnosis.

And so what I'd like to do is tabulate sort of the answers, the true diagnosis, against what we got from hierarchical clustering. We did this on the previous slide. And we said, hey, that looks pretty good. We seem to have our cluster two containing mostly benign and cluster one containing nearly all malignant. Or if you want to take a look at this from the terms of an actual diagnosis, a good majority of the malignant are picked up in cluster one, while nearly all the benign are picked up in cluster two.

However, if we take a look at non-hierarchical clustering, the results, compared to the truth, are better, I would suggest, because we're getting a much higher percentage of the malignant being clustered into one while nearly the same number of benign being in cluster two.

So in this case-- again, if we had a prior idea of the grouping numbers or if we were actually able to compare this to a diagnosis of known cases, we would decide to use non-hierarchical over hierarchical. In many cases, you will not have an answer, so to speak.

And so taking a look for consistency in non-hierarchical clustering as well as in our hierarchical clustering taking a look for spacing that suggests that a vertical spacing-- that is, spacing and distance that suggests that a particular clustering might make more sense-- are ways of picking out a correct number of clusterings. And having some idea of the proportional distribution would help, as well, in thinking of a comparison between the two types of clustering. Thanks for your attention, and have fun clustering.

Notes:
See the online course for a downloadable R file containing the set of commands used in this demonstration.

Summary One

Clustering is a set of approaches in data exploration that aim to identify groups that are internally similar to each other, but are also dissimilar from those in other groups.

- These methods are unsupervised, so there is typically no validation of the end result.
- Methods can be dependent on distance measurements, measures of dissimilarity, spatial orientation of clusters, number of clusters used, and scaling of variables.
- Existence of clustering can inform further study.

Summary Two

Two main types of clustering methods:

Hierarchical: smaller clusters are merged into larger clusters.

- Requires selection of distance measure and linkage type.
- All possible numbers of clusters are generated, and selection of number of clusters is visualized via cross-section of dendrogram.

Non-hierarchical: clusters may be re-formed for different *K*.

- Discussed *K*-means method, where the fit is achieved through an algorithmic process on a random initialization of clusters.
- Often faster, but no clear rules for number of clusters to use.