Hierarchical Clustering

Philip D. Waggoner

MACS 40800: Unsupervised Machine Learning

October 17, 2019

Lecture Outline

Lecture Outline

Review of Clustering Basics

- We are interested in reducing the dimensionality of the space to make the data more accessible and understandable

Review of Clustering Basics

- We are interested in reducing the dimensionality of the space to make the data more accessible and understandable
- This means, we are looking for underlying, *non-random* patterns in some feature space

Review of Clustering Basics

- We are interested in reducing the dimensionality of the space to make the data more accessible and understandable
- This means, we are looking for underlying, *non-random* patterns in some feature space
- Yet how we think about *grouping* is dependent on the distribution of the data, which has implications for algorithm selection and validation

Review of Clustering Basics

- We are interested in reducing the dimensionality of the space to make the data more accessible and understandable
- This means, we are looking for underlying, *non-random* patterns in some feature space
- Yet how we think about *grouping* is dependent on the distribution of the data, which has implications for algorithm selection and validation hierarchical and partitioning
- Key difference between clustering approaches: **subdividing the data**

Review of Clustering Basics

- We are interested in reducing the dimensionality of the space to make the data more accessible and understandable
- This means, we are looking for underlying, *non-random* patterns in some feature space
- Yet how we think about *grouping* is dependent on the distribution of the data, which has implications for algorithm selection and validation hierarchical and partitioning
- Key difference between clustering approaches: **subdividing the data**
  - Hierarchical $\rightsquigarrow$ No
  - Partitioning $\rightsquigarrow$ Yes

Review of Clustering Basics

- Steps to clustering (very general):

Review of Clustering Basics

- Steps to clustering (very general):
  - Diagnose clusterability

Review of Clustering Basics

- Steps to clustering (very general):
  - Diagnose clusterability
  - Standardize/scale the data

Review of Clustering Basics

- Steps to clustering (very general):
    - Diagnose clusterability
    - Standardize/scale the data
    - Select distance measure
        - Select the linkage methods (in hierarchical *only*)

Review of Clustering Basics

- Steps to clustering (very general):
  - Diagnose clusterability
  - Standardize/scale the data
  - Select distance measure
    - Select the linkage methods (in hierarchical *only*)
  - Run the algorithm

Review of Clustering Basics

- Steps to clustering (very general):
  - Diagnose clusterability
  - Standardize/scale the data
  - Select distance measure
    - Select the linkage methods (in hierarchical *only*)
  - Run the algorithm
  - Inspect patterns, usually visually

Review of Clustering Basics

- Steps to clustering (very general):
  - Diagnose clusterability
  - Standardize/scale the data
  - Select distance measure
    - Select the linkage methods (in hierarchical *only*)
  - Run the algorithm
  - Inspect patterns, usually visually
  - Check validation, especially if comparing across clustering algorithms (more next week)

Lecture Outline

1. Quick Review
2. Hierarchical Clustering
3. Linkage Methods
4. Dendrograms & Tree Cutting
5. Divisive Hierarchical Clustering
6. Demonstration in R
7. Due Dates

Hierarchical *Agglomerative* Clustering

- The most common form is agglomerative, or “bottom-up” clustering

Hierarchical *Agglomerative* Clustering

- The most common form is agglomerative, or "bottom-up" clustering
- Opposite partitioning, hierarchical clustering is concerned with creating clusters one observation at a time (or recursively for divisive)

Hierarchical *Agglomerative* Clustering

- The most common form is agglomerative, or "bottom-up" clustering
- Opposite partitioning, hierarchical clustering is concerned with creating clusters one observation at a time (or recursively for divisive)
- Each observation is treated as a cluster (a "singleton"), and is joined together with the next closest observation

Hierarchical *Agglomerative* Clustering

- The most common form is agglomerative, or "bottom-up" clustering
- Opposite partitioning, hierarchical clustering is concerned with creating clusters one observation at a time (or recursively for divisive)
- Each observation is treated as a cluster (a "singleton"), and is joined together with the next closest observation
- This cluster is joined with other similar observations in a pairwise fashion and converges when all observations belong to a single cluster, k

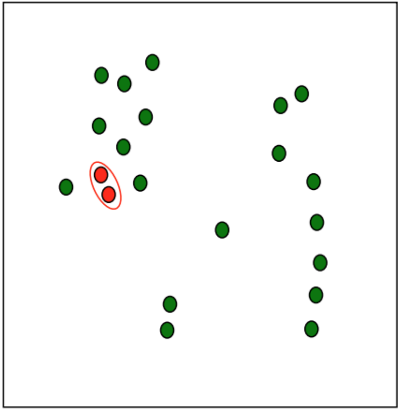
Hierarchical *Agglomerative* Clustering

- The most common form is agglomerative, or “bottom-up” clustering
- Opposite partitioning, hierarchical clustering is concerned with creating clusters one observation at a time (or recursively for divisive)
- Each observation is treated as a cluster (a “singleton”), and is joined together with the next closest observation
- This cluster is joined with other similar observations in a pairwise fashion and converges when all observations belong to a single cluster, k
- Key terms and necessary ingredients:

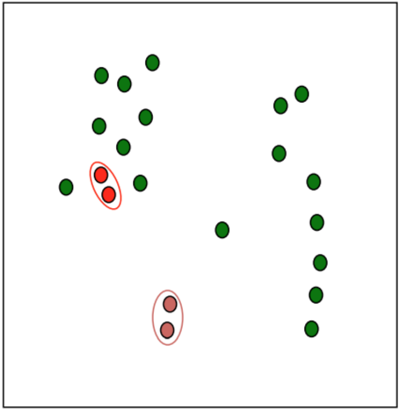
Hierarchical *Agglomerative* Clustering

- The most common form is agglomerative, or "bottom-up" clustering
- Opposite partitioning, hierarchical clustering is concerned with creating clusters one observation at a time (or recursively for divisive)
- Each observation is treated as a cluster (a "singleton"), and is joined together with the next closest observation
- This cluster is joined with other similar observations in a pairwise fashion and converges when all observations belong to a single cluster, k
- Key terms and necessary ingredients: distance measure, linkage methods, dendrogram, tree-cutting

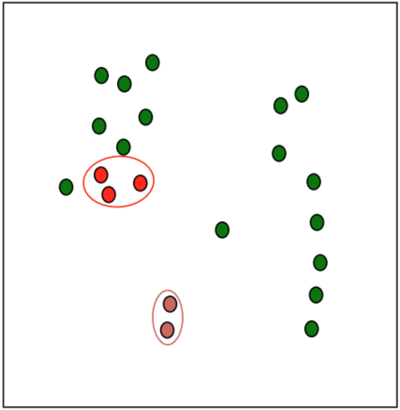
Hierarchical Clustering: The Idea



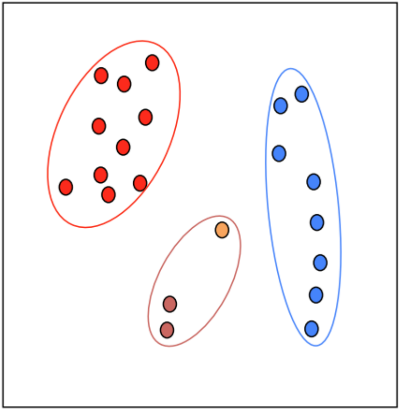
Hierarchical Clustering: The Idea



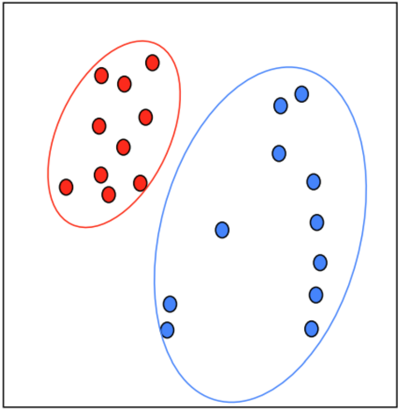
Hierarchical Clustering: The Idea



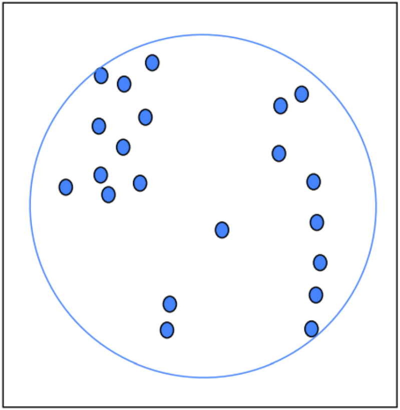
Hierarchical Clustering: The Idea



Hierarchical Clustering: The Idea



Hierarchical Clustering: The Idea



A Simple Hierarchical Agglomerative Clustering Algorithm

- Treat each observation as a singleton

A Simple Hierarchical Agglomerative Clustering Algorithm

- Treat each observation as a singleton
- Begin with n observations and some [distance] measure of all the $\binom{n}{2} = \frac{n(n-1)}{2}$ pairwise dissimilarities

A Simple Hierarchical Agglomerative Clustering Algorithm

- Treat each observation as a singleton
- Begin with *n* observations and some [distance] measure of all the $\binom{n}{2} = \frac{n(n-1)}{2}$ pairwise dissimilarities
- Examine all pairwise dissimilarities among the clusters and identify the pair of clusters that are *least* dissimilar

A Simple Hierarchical Agglomerative Clustering Algorithm

- Treat each observation as a singleton
- Begin with n observations and some [distance] measure of all the $\binom{n}{2} = \frac{n(n-1)}{2}$ pairwise dissimilarities
- Examine all pairwise dissimilarities among the clusters and identify the pair of clusters that are *least* dissimilar
- Join these clusters together based on the linkage method

A Simple Hierarchical Agglomerative Clustering Algorithm

- Treat each observation as a singleton
- Begin with n observations and some [distance] measure of all the $\binom{n}{2} = \frac{n(n-1)}{2}$ pairwise dissimilarities
- Examine all pairwise dissimilarities among the clusters and identify the pair of clusters that are *least* dissimilar
- Join these clusters together based on the linkage method
- Compute the new pairwise inter-cluster dissimilarities among the remaining clusters

A Simple Hierarchical Agglomerative Clustering Algorithm

- Treat each observation as a singleton
- Begin with n observations and some [distance] measure of all the $\binom{n}{2} = \frac{n(n-1)}{2}$ pairwise dissimilarities
- Examine all pairwise dissimilarities among the clusters and identify the pair of clusters that are *least* dissimilar
- Join these clusters together based on the linkage method
- Compute the new pairwise inter-cluster dissimilarities among the remaining clusters
- Stop when we reach k clusters ($k = 1$ in agglomerative; $k = n$ in divisive)

Inter-Cluster Dissimilarity

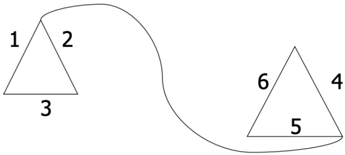
- You may have recognized that a key to the algorithm is the notion of inter-cluster dissimilarity

Inter-Cluster Dissimilarity

- You may have recognized that a key to the algorithm is the notion of inter-cluster dissimilarity
- For example, suppose we had 6 points in the following 2 clusters,

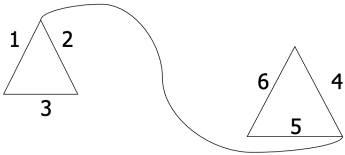
Inter-Cluster Dissimilarity

- You may have recognized that a key to the algorithm is the notion of inter-cluster dissimilarity
- For example, suppose we had 6 points in the following 2 clusters,



Inter-Cluster Dissimilarity

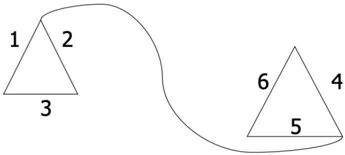
- You may have recognized that a key to the algorithm is the notion of inter-cluster dissimilarity
- For example, suppose we had 6 points in the following 2 clusters,



- We can think about dissimilarities between these clusters as distances between some point (whether real or calculated) in cluster 1 on the left, and some point in cluster 2 on the right

Inter-Cluster Dissimilarity

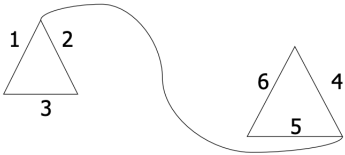
- You may have recognized that a key to the algorithm is the notion of inter-cluster dissimilarity
- For example, suppose we had 6 points in the following 2 clusters,



- We can think about dissimilarities between these clusters as distances between some point (whether real or calculated) in cluster 1 on the left, and some point in cluster 2 on the right
- In a sea of possible clusters, subject to some constraint, smaller clusters are fused to create a new, larger cluster, and the process iterates until $k = 1$

Inter-Cluster Dissimilarity

- You may have recognized that a key to the algorithm is the notion of inter-cluster dissimilarity
- For example, suppose we had 6 points in the following 2 clusters,



- We can think about dissimilarities between these clusters as distances between some point (whether real or calculated) in cluster 1 on the left, and some point in cluster 2 on the right
- In a sea of possible clusters, subject to some constraint, smaller clusters are fused to create a new, larger cluster, and the process iterates until $k = 1$
- So how do we determine what constrains cluster fusion...?

Lecture Outline

Linkage Methods

- Linkage defines the (dis)similarity between two groups of observations

Linkage Methods

- Linkage defines the (dis)similarity between two groups of observations
- There are five common types of linkage: complete, single, Ward's method, average, and centroid

Linkage Methods

- **Complete** linkage uses the *maximal* inter-cluster dissimilarity,

$$d_{complete}(C_x, C_y) = max\{C_x, C_y\}$$

Linkage Methods

- **Complete** linkage uses the *maximal* inter-cluster dissimilarity,

$$d_{complete}(C_x, C_y) = max\{C_x, C_y\}$$

- **Single** linkage uses the *minimal* inter-cluster dissimilarity,

$$d_{single}(C_x, C_y) = min\{C_x, C_y\}$$

Linkage Methods

- **Complete** linkage uses the *maximal* inter-cluster dissimilarity,

$$d_{complete}(C_x, C_y) = max\{C_x, C_y\}$$

- **Single** linkage uses the *minimal* inter-cluster dissimilarity,

$$d_{single}(C_x, C_y) = min\{C_x, C_y\}$$

- **Ward's** linkage method joins the two clusters whose fusion is constrained by the smallest increase in SSE calculated per cluster, C,

$$\sum_{i=1}^{n}(x_i - \bar{x})^2$$

Linkage Methods

- **Average** linkage uses the *mean* inter-cluster dissimilarity,

$$d_{average}(C_x, C_y) = \frac{\sum_i \sum_j d_{ij}}{N_{C_x} N_{C_y}}$$

where, d_{ij} is the pairwise distance between observations i and j, and N_* is the total number of observations in the computed cluster, C

Linkage Methods

- **Average** linkage uses the *mean* inter-cluster dissimilarity,

$$d_{average}(C_x, C_y) = \frac{\sum_i \sum_j d_{ij}}{N_{C_x} N_{C_y}}$$

where, d_{ij} is the pairwise distance between observations i and j, and N_* is the total number of observations in the computed cluster, C

- **Centroid** linkage computes the dissimilarity between the *centroid* for cluster, $\bar{x}_{C_*}$,

$$d_{centroid}(C_x, C_y) = d(\bar{x}_{C_x}, \bar{x}_{C_y})$$

Linkage Methods

- **Average** linkage uses the *mean* inter-cluster dissimilarity,

$$d_{average}(C_x, C_y) = \frac{\sum_i \sum_j d_{ij}}{N_{C_x} N_{C_y}}$$

where, d_{ij} is the pairwise distance between observations i and j, and N_* is the total number of observations in the computed cluster, C

- **Centroid** linkage computes the dissimilarity between the *centroid* for cluster, $\bar{x}_{C_*}$,

$$d_{centroid}(C_x, C_y) = d(\bar{x}_{C_x}, \bar{x}_{C_y})$$

- Difference between average and centroid:

Linkage Methods

- **Average** linkage uses the *mean* inter-cluster dissimilarity,

$$d_{average}(C_x, C_y) = \frac{\sum_i \sum_j d_{ij}}{N_{C_x} N_{C_y}}$$

where, d_{ij} is the pairwise distance between observations i and j, and N_* is the total number of observations in the computed cluster, C

- **Centroid** linkage computes the dissimilarity between the *centroid* for cluster, $\bar{x}_{C_*}$,

$$d_{centroid}(C_x, C_y) = d(\bar{x}_{C_x}, \bar{x}_{C_y})$$

- Difference between average and centroid: average $\rightsquigarrow$ average of all *pairwise* calculations;

Linkage Methods

- **Average** linkage uses the *mean* inter-cluster dissimilarity,

$$d_{average}(C_x, C_y) = \frac{\sum_i \sum_j d_{ij}}{N_{C_x} N_{C_y}}$$

where, d_{ij} is the pairwise distance between observations i and j, and N_* is the total number of observations in the computed cluster, C

- **Centroid** linkage computes the dissimilarity between the *centroid* for cluster, $\bar{x}_{C_*}$,

$$d_{centroid}(C_x, C_y) = d(\bar{x}_{C_x}, \bar{x}_{C_y})$$

- Difference between average and centroid: average $\rightsquigarrow$ average of all *pairwise* calculations; centroid $\rightsquigarrow$ fuses across centroids, which are intra-cluster averages

Which linkage method should we select?

Which linkage method should we select?

- There is no rule on determining which method is "best", as best is subjective, and typically project (or domain) dependent

Which linkage method should we select?

- There is no rule on determining which method is "best", as best is subjective, and typically project (or domain) dependent
  - **Average**, **complete** and **single** linkage are most popular among statisticians, while **centroid** is often used in genomics and **Ward's** is most common in text mining

Which linkage method should we select?

- There is no rule on determining which method is "best", as best is subjective, and typically project (or domain) dependent
  - **Average**, **complete** and **single** linkage are most popular among statisticians, while **centroid** is often used in genomics and **Ward's** is most common in text mining
  - **Average** and **complete** linkage tend to be preferred because they tend to yield more balanced dendrograms (more in a moment)

Which linkage method should we select?

- There is no rule on determining which method is "best", as best is subjective, and typically project (or domain) dependent
  - **Average**, **complete** and **single** linkage are most popular among statisticians, while **centroid** is often used in genomics and **Ward's** is most common in text mining
  - **Average** and **complete** linkage tend to be preferred because they tend to yield more balanced dendrograms (more in a moment)
  - **Single** linkage can result in elongated, stringy-type clusters where each observation fuses one-at-a-time (i.e., its not pretty...)

Which linkage method should we select?

- There is no rule on determining which method is "best", as best is subjective, and typically project (or domain) dependent
  - **Average**, **complete** and **single** linkage are most popular among statisticians, while **centroid** is often used in genomics and **Ward's** is most common in text mining
  - **Average** and **complete** linkage tend to be preferred because they tend to yield more balanced dendrograms (more in a moment)
  - **Single** linkage can result in elongated, stringy-type clusters where each observation fuses one-at-a-time (i.e., its not pretty...)
  - **Ward's** method is based on minimizing the "loss of information" from joining two groups

Reiterating Differences between Distance Measures

- It might be confusing that we are talking about two types of distance in hierarchical clustering: "distance measure" and "linkage method"

Reiterating Differences between Distance Measures

- It might be confusing that we are talking about two types of distance in hierarchical clustering: “distance measure” and “linkage method”

- Thus, its worth reiterating the difference here to avoid confusion

Reiterating Differences between Distance Measures

- It might be confusing that we are talking about two types of distance in hierarchical clustering: "distance measure" and "linkage method"
- Thus, its worth reiterating the difference here to avoid confusion
- **Distance**: the measure of (dis)similarity between *observations* (e.g., $d_{euclidean}(p, q)$)

Reiterating Differences between Distance Measures

- It might be confusing that we are talking about two types of distance in hierarchical clustering: "distance measure" and "linkage method"
- Thus, its worth reiterating the difference here to avoid confusion
- **Distance**: the measure of (dis)similarity between *observations* (e.g., $d_{euclidean}(p, q)$)
- **Linkage**: also distance, but the measure of (dis)similarity between *clusters* (e.g., $d_{single}(C_x, C_y)$)

Reiterating Differences between Distance Measures

- It might be confusing that we are talking about two types of distance in hierarchical clustering: "distance measure" and "linkage method"
- Thus, its worth reiterating the difference here to avoid confusion
- **Distance**: the measure of (dis)similarity between *observations* (e.g., $d_{euclidean}(p, q)$)
- **Linkage**: also distance, but the measure of (dis)similarity between *clusters* (e.g., $d_{single}(C_x, C_y)$)
- Therefore, the input for a hierarchical clustering algorithm is an $N \times N$ distance matrix, from which **inter-cluster distances** are calculated via the selected linkage method

Lecture Outline

Evaluating Hierarchical Clustering Output: Dendrograms

- So we fit a hierarchical clustering algorithm... now what?

Evaluating Hierarchical Clustering Output: Dendrograms

- So we fit a hierarchical clustering algorithm... now what?
- The most common, clear way to diagnose and explore output is a dendrogram

Evaluating Hierarchical Clustering Output: Dendrograms

- So we fit a hierarchical clustering algorithm... now what?
- The most common, clear way to diagnose and explore output is a dendrogram
- A tree-like structure with “leaves” and “branches”

Evaluating Hierarchical Clustering Output: Dendrograms

- So we fit a hierarchical clustering algorithm... now what?
- The most common, clear way to diagnose and explore output is a dendrogram
- A tree-like structure with "leaves" and "branches"
  - **Leaves**: observations

Evaluating Hierarchical Clustering Output: Dendrograms

- So we fit a hierarchical clustering algorithm... now what?
- The most common, clear way to diagnose and explore output is a dendrogram
- A tree-like structure with "leaves" and "branches"
  - **Leaves**: observations
  - **Branches**: inter-cluster connectors

Evaluating Hierarchical Clustering Output: Dendrograms

- So we fit a hierarchical clustering algorithm... now what?
- The most common, clear way to diagnose and explore output is a dendrogram
- A tree-like structure with "leaves" and "branches"
  - **Leaves**: observations
  - **Branches**: inter-cluster connectors
- As each **leaf** is fused with another, progressing from the bottom-up until all singletons are merged into a single **tree**

Evaluating Hierarchical Clustering Output: Dendrograms

- So we fit a hierarchical clustering algorithm... now what?
- The most common, clear way to diagnose and explore output is a dendrogram
- A tree-like structure with "leaves" and "branches"
  - **Leaves**: observations
  - **Branches**: inter-cluster connectors
- As each **leaf** is fused with another, progressing from the bottom-up until all singletons are merged into a single **tree**
- There is only one value-axis: the Y axis is the measured distance

Evaluating Hierarchical Clustering Output: Dendrograms

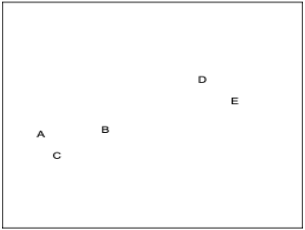
- So we fit a hierarchical clustering algorithm... now what?
- The most common, clear way to diagnose and explore output is a dendrogram
- A tree-like structure with "leaves" and "branches"
    - **Leaves**: observations
    - **Branches**: inter-cluster connectors
- As each **leaf** is fused with another, progressing from the bottom-up until all singletons are merged into a single **tree**
- There is only one value-axis: the Y axis is the measured distance
- So we can get clear clustering when branches along the Y axis are long (suggesting greater distance from other clusters), and less obvious clustering when the branches are shorter

Evaluating Hierarchical Clustering Output: Dendrograms

- Let's consider another simple toy example, and then move to a real case in `R`

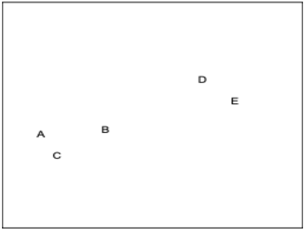
Evaluating Hierarchical Clustering Output: Dendrograms

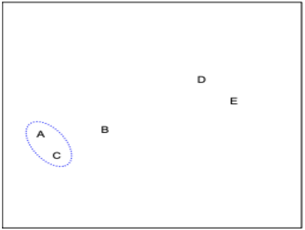
- Let's consider another simple toy example, and then move to a real case in R



Evaluating Hierarchical Clustering Output: Dendrograms

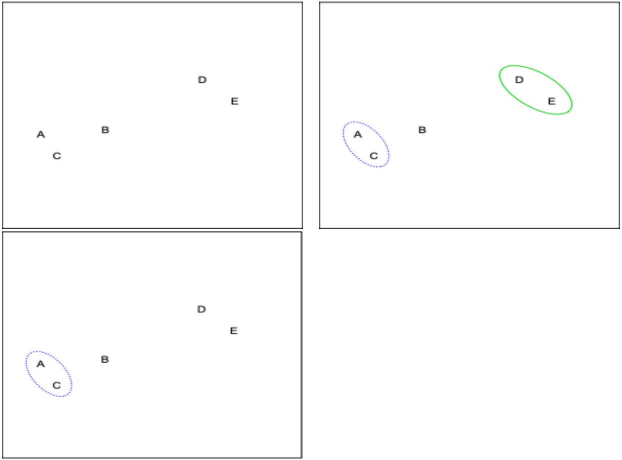
- Let's consider another simple toy example, and then move to a real case in `R`





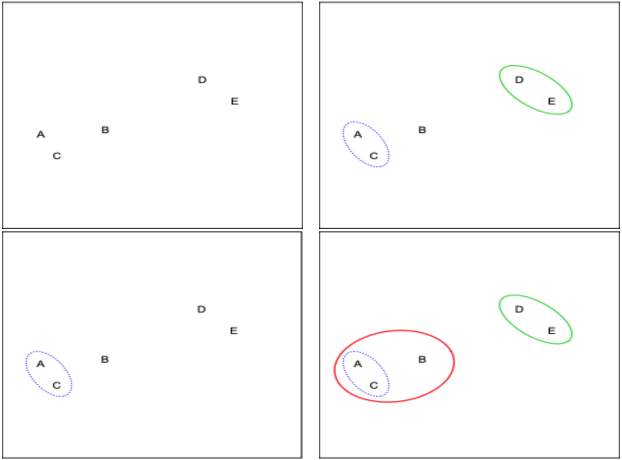
Evaluating Hierarchical Clustering Output: Dendrograms

- Let's consider another simple toy example, and then move to a real case in R



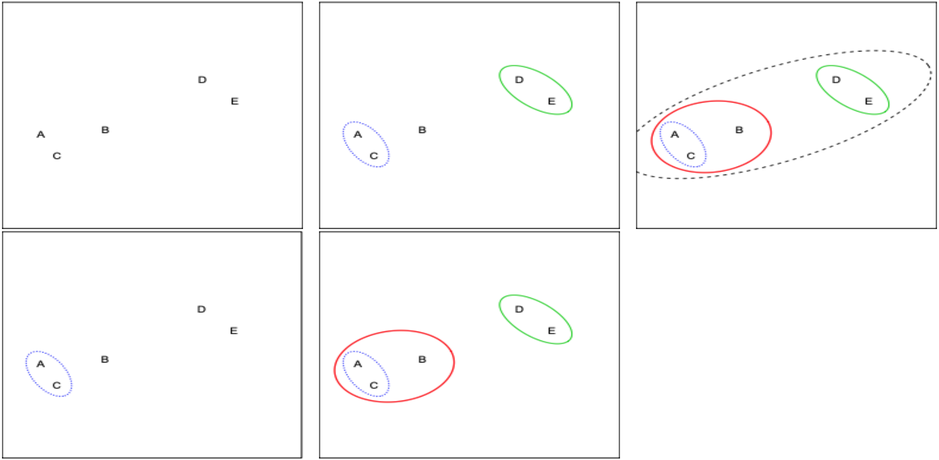
Evaluating Hierarchical Clustering Output: Dendrograms

- Let's consider another simple toy example, and then move to a real case in R



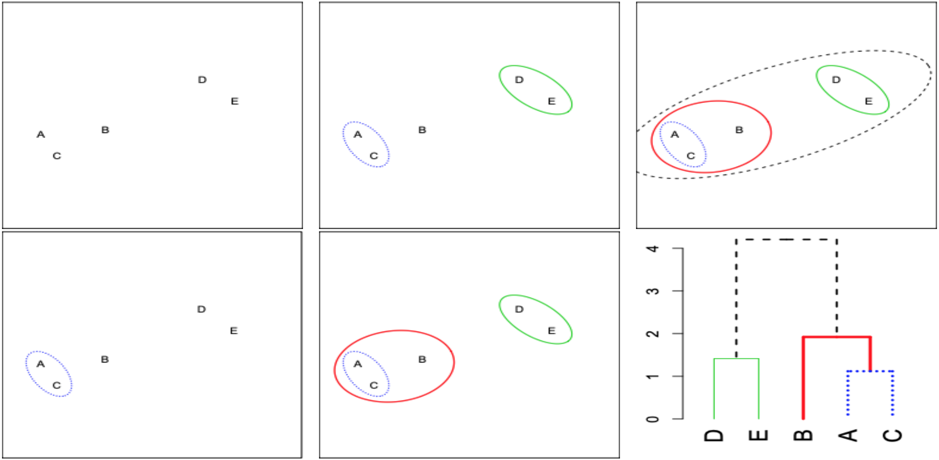
Evaluating Hierarchical Clustering Output: Dendrograms

- Let's consider another simple toy example, and then move to a real case in R



Evaluating Hierarchical Clustering Output: Dendrograms

- Let's consider another simple toy example, and then move to a real case in R



Final Considerations

- Though hierarchical clustering is lauded for being assumption free in that we don't select k *a priori* (as we will in k-means, e.g.), we actually do need to select k, but *post hoc*

Final Considerations

- Though hierarchical clustering is lauded for being assumption free in that we don't select k *a priori* (as we will in k-means, e.g.), we actually do need to select k, but *post hoc*
- In other words, we fit our algorithm, we plotted our dendrogram, and we now have to do something useful with the output

Final Considerations

- Though hierarchical clustering is lauded for being assumption free in that we don't select k *a priori* (as we will in k-means, e.g.), we actually do need to select k, but *post hoc*
- In other words, we fit our algorithm, we plotted our dendrogram, and we now have to do something useful with the output
- The "useful" thing is to cut the tree where likely clusters exist, which is typically where branches are longest on the Y axis

Final Considerations

- Though hierarchical clustering is lauded for being assumption free in that we don't select k *a priori* (as we will in k-means, e.g.), we actually do need to select k, but *post hoc*
- In other words, we fit our algorithm, we plotted our dendrogram, and we now have to do something useful with the output
- The "useful" thing is to cut the tree where likely clusters exist, which is typically where branches are longest on the Y axis
- So how many clusters *should* accurately characterize the data...?

Final Considerations

- Though hierarchical clustering is lauded for being assumption free in that we don't select k *a priori* (as we will in k-means, e.g.), we actually do need to select k, but *post hoc*
- In other words, we fit our algorithm, we plotted our dendrogram, and we now have to do something useful with the output
- The "useful" thing is to cut the tree where likely clusters exist, which is typically where branches are longest on the Y axis
- So how many clusters *should* accurately characterize the data...?
- That's inherently subjective and largely up to the researcher

Final Considerations

- Though hierarchical clustering is lauded for being assumption free in that we don't select k *a priori* (as we will in k-means, e.g.), we actually do need to select k, but *post hoc*
- In other words, we fit our algorithm, we plotted our dendrogram, and we now have to do something useful with the output
- The "useful" thing is to cut the tree where likely clusters exist, which is typically where branches are longest on the Y axis
- So how many clusters *should* accurately characterize the data...?
- That's inherently subjective and largely up to the researcher
- HC is usually most useful in comparison to other approaches allowing for comparison and validation (more next week)

Final Considerations

- Though hierarchical clustering is lauded for being assumption free in that we don't select k *a priori* (as we will in k-means, e.g.), we actually do need to select k, but *post hoc*
- In other words, we fit our algorithm, we plotted our dendrogram, and we now have to do something useful with the output
- The "useful" thing is to cut the tree where likely clusters exist, which is typically where branches are longest on the Y axis
- So how many clusters *should* accurately characterize the data...?
- That's inherently subjective and largely up to the researcher
- HC is usually most useful in comparison to other approaches allowing for comparison and validation (more next week)
- Good to note: HC is pretty computationally inefficient and expensive, especially on large datasets

Lecture Outline

Divisive Hierarchical Clustering

- Divisive clustering is often referred to “top down” clustering

Divisive Hierarchical Clustering

- Divisive clustering is often referred to “top down” clustering
- We begin by assigning all of observations to a single cluster

Divisive Hierarchical Clustering

- Divisive clustering is often referred to "top down" clustering
- We begin by assigning all of observations to a single cluster
- We then partition the cluster into two *least* similar clusters, of all possible split values

Divisive Hierarchical Clustering

- Divisive clustering is often referred to "top down" clustering
- We begin by assigning all of observations to a single cluster
- We then partition the cluster into two *least* similar clusters, of all possible split values
- We proceed recursively on each cluster until each cluster is a singleton

Divisive Hierarchical Clustering

- Divisive clustering is often referred to "top down" clustering
- We begin by assigning all of observations to a single cluster
- We then partition the cluster into two *least* similar clusters, of all possible split values
- We proceed recursively on each cluster until each cluster is a singleton
- This is significantly more expensive even than agglomerative, given the many split calculations required at each split (hence its less popular)

Lecture Outline

Demonstration in R

`HCA_demo.R`

Lecture Outline

Due Dates

- **Team project proposals due today, Thursday, 10/17, by 5 pm**
- **Problem set # 2 due Saturday, 10/19, by 12 pm**

Due Dates

- **Team project proposals due today, Thursday, 10/17, by 5 pm**
- **Problem set # 2 due Saturday, 10/19, by 12 pm**
- Take off a bit early and use the time while you're together to polish your proposals