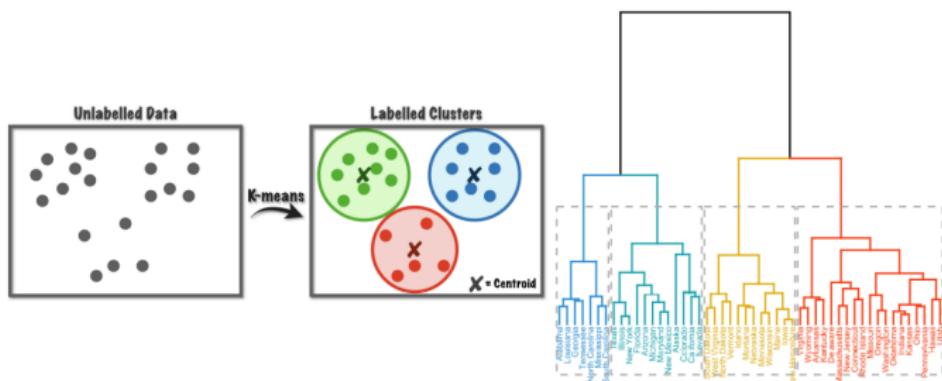


Clustering

PSY-GS 8875 Behavioral Data Science



Overview

Overview: Week 7

Readings

- ESL Chapters: 14.3
- HML Chapters: 20 and 21

Optional

- [Steinley - 2006](#)
- [Forbes et al. - 2023](#)

- Distances
- k -Means/Medoids Clustering
- Cluster Selection
- Cluster Comparison
- Hierarchical Clustering
- Activity: Schools that pay

Motivating Data

Motivating Data

Motivating Data

Wall Street Journal in 2008 put out data on median salary for different universities across the United States

Across 320 universities sampled, we want to know:

- Do universities tend to organize by region?
- Does Vanderbilt cluster with universities it likes to compare itself to (e.g., Harvard, Yale)?

Motivating Data

Set up data

```
# Load packages
library(ggplot2); library(factoextra); library(cluster)
library(NbClust); library(igraph); library(EGAnet)

# Set seed
set.seed(42)

# Load data
salary <- read.csv("../data/university_salary/schools_that_pay.csv")

# Select complete data (no missing values)
salary_voi <- salary[,-c(5, 8)]

# Regions
unique(salary_voi$region)

[1] "California"    "Western"        "Midwestern"      "Southern"       "Northeastern"
```

Motivating Data

Set up data

```
# Extract numeric values  
salary_numeric <- salary_voi[,-c(1:2)]  
  
# Scale numeric values  
salary_numeric <- scale(salary_numeric)  
# Not necessary with Gower's Distance *BUT*  
# necessary for many distances used in clustering
```

Clustering

Clustering

Clustering

clustering: identifying sets of **observations** or **variables** that are “like” each other based on their **variables** or **observations**

- *Most often*, clustering refers to identifying sets of **observations** that are similar in their values across one or more **variables**
- *Sometimes*, clustering refers to identifying sets of **variables** that are similar in their values across many **observations**
- *Always*, clustering refers to identifying sets of data that are alike

Purpose

- For our purposes, we will refer **clustering** as the identification of sets of **observations** that are similar in their values across one or more **variables**
- Goal: find sub-groups of observations that have *meaningful* differences

Distances

- Most clustering algorithms work on some kind of “distance” between observations
- Distances can be computed in many different ways including . . .
 - Actual distance measures: Euclidean, Manhattan, etc.
 - Correlations
 - Information theory: entropy, divergences (e.g., Kullback-Leibler)
- Distances are how *dissimilar* two observations are (across variables)
- 1 minus a distance is how *similar* two observations are (across variables)

Euclidean Distance

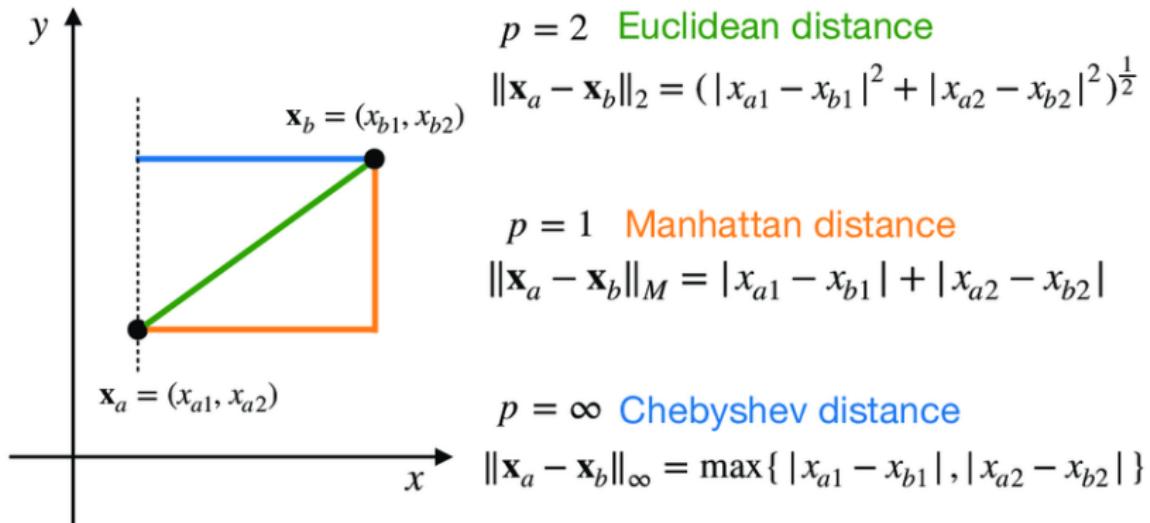
$$d(i, j) = \sqrt{\sum_{k=1}^n (x_{i,k} - x_{j,k})^2},$$

Euclidean Distance

$$d(i, j) = \sqrt{\sum_{k=1}^n (x_{i,k} - x_{j,k})^2},$$

Does this equation look familiar at all?

Common Distances (2-dimensions or variables)



Euclidean distance is, by far, the most commonly used

One problem though... Euclidean distance is intended for continuous variables

- Continuous data aren't common in the social sciences
- It's not uncommon to have mixed data types (continuous, ordinal, and binary)

Gower's Distance (Gower, 1971)

$$d(i, j) = 1 - \left(\frac{\sum_{k=1}^n w_k \cdot s_k(i, j)}{\sum_{k=1}^n w_k} \right),$$

where

- $s_k(i, j)$ = similarity scores between i and j for variable k
- w_k = weight assigned to k (usually equal to 1)

General form for the average of partial similarities

Continuous Variables

$$s_k(i, j) = 1 - \frac{|x_{i,k} - x_{j,k}|}{R_k},$$

where

- R_k = range of variable k across all values (normalizing term)

Ordinal Variables

$$s_k(i, j) = 1 - \frac{r_{ik} - r_{jk}}{n_k - 1},$$

where

- r = ranks of variable k for observations
- n_k = number of distinct ranks of variable k

Gower's Distance

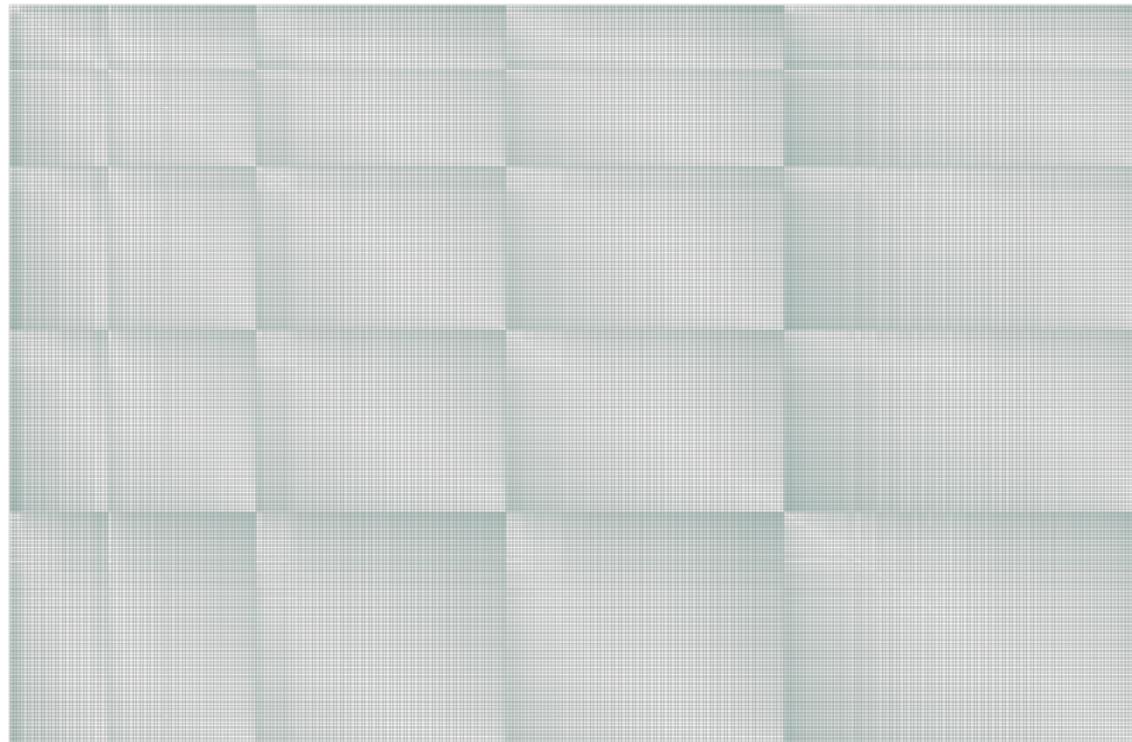
- Useful, general-purpose distance for mixed data types
- Roughly consistent with other distance measures with continuous data:
 - Manhattan: equivalent when performing $(1 - s_k) \times R_k$
 - Euclidean: scales similarly (though no direct translation in math)
- Can be made to **metric** by taking the square root
 - non-negativity: always positive values
 - symmetric: $d(i, j) = d(j, i)$
 - triangle inequality: $d(i, j) \leq d(i, k) + d(k, j)$
 - zero when values are identical

Compute Gower's Distance

```
# Convert to distance (from {cluster} package)
salary_distance <- daisy(salary_numeric, metric = "gower")

# Visualize
EGAnet:::ggssymmetric(salary_distance) +
  scale_fill_gradient2(
    name = "Gower's Distance", limits = c(0, 1),
    low = "lightgrey", high = "#A3C4BC"
  ) + theme(
    axis.text = element_blank(), axis.title = element_blank(),
    axis.ticks = element_blank(), legend.position = "bottom",
    legend.key.width = unit(2, "cm"),
    legend.key.height = unit(0.5, "cm")
  )
```

Clustering | Distances



Gower's Distance

0.00

0.25

0.50

0.75

1.00

k -means/ k -mediods

Goal: minimize the *mean/median* distance between observations using k clusters

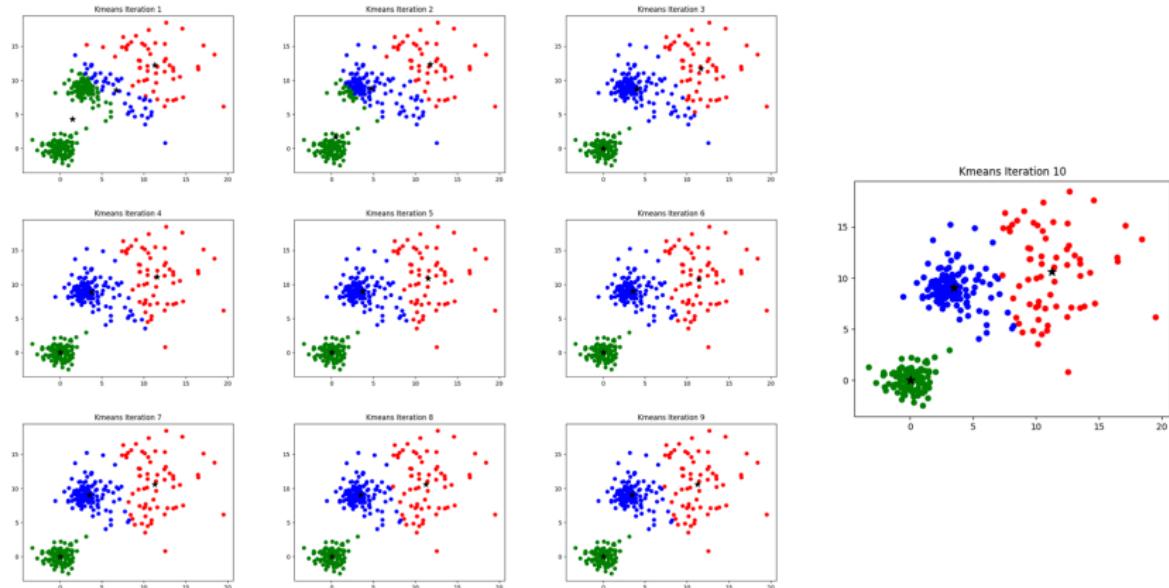
- k -means/ k -medioids minimizes this objective in an *unsupervised* manner
- **unsupervised learning:** using statistical patterns, without known labels, to learn the data-driven labels
- We'll focus on k -means but we can swap out "median" with "mean" at any point for k -medioids
 - Often, k -medioids is preferred because the median tends to be more robust to outliers

How does k -means “learn” the labels or sub-groups of the data?

Algorithm

- Step 1: Decide on the number of clusters (k)
- Step 2: Initialize centroids (for k -means) or medoids (for k -medoids) such as selecting random observations as the k centroids
- Step 3: Compute a “centroid” for each cluster representing p variable means (k -means) and the medoid for each cluster (k -medioids)
Step 4: Assign each observation to the cluster whose centroid (or medoid) is closest in terms of Euclidean distance.
Step 5: Repeat Steps 3 and 4 until there are no changes in centroid positions or cluster assignments.

Clustering | k -means/ k -medioids



Step 1: Decide on number of clusters (k)

- Are there any expected sub-groups in the data?
 - Pre-defined categories: regions of the country
 - High/low: personality trait profiles
 - Different samples: control/experimental
- If no expectation, then the groups can be “learned” from the data by some objective function (e.g., mean square error)

Step 2: Randomly assign $1-k$ values to observations

- This step is done within the algorithm
- If there is some theory, then starting values can be provided
- Otherwise, one or more sets of random values will be used and return the clusters that best minimize the objective function

Step 3: Compute a "centroid" for each cluster representing p variable means

- The initial centroids are the (random) starting values
- With initial centroids placed, the goal is to minimize the within-cluster sum of squared distances

$$\arg \min_{W(C_k)} \sum_{k=1}^K W(C_k) = \sum_{i \in C_k} d(x_i, \mu_k)^2$$

- C_k = set of all observations in the k^{th} cluster
- $d(x_i, \mu_k)$ = distance between observation i and the centroid μ_k of cluster C_k

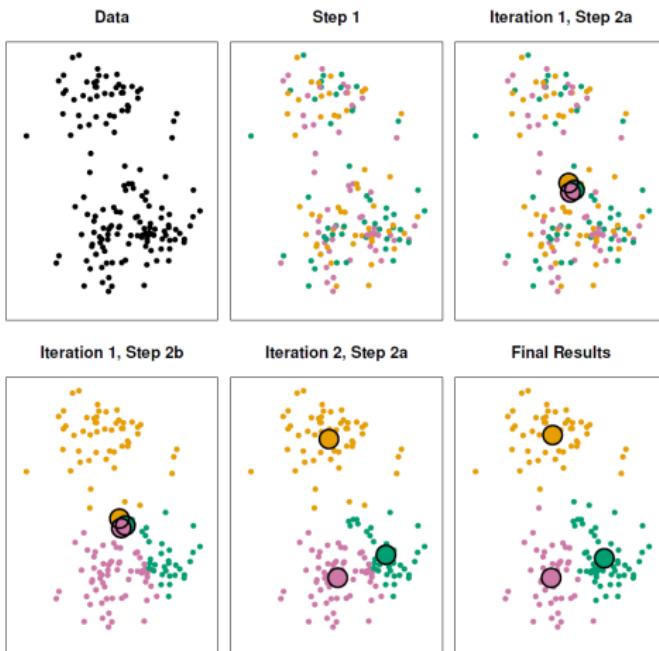
This equation represents the total within-cluster sum of squared distances

Step 4: Assign each observation to the cluster whose centroid is closest

- The centroid will move after Step 3 based on how best the total within-cluster sum of squared distances can be minimized

Step 5: Repeat Step 3 (computing distances) and Step 4 (moving centroids) until there are no changes

Simple Steps



Different Starting Values



Applying k-medoids

```
# Perform k-medoids based on number regions
theoretical_run <- pam(
  x = salary_distance, # supply distance
  k = 5, # number of clusters
  nstart = 25 # number of random starting values
)

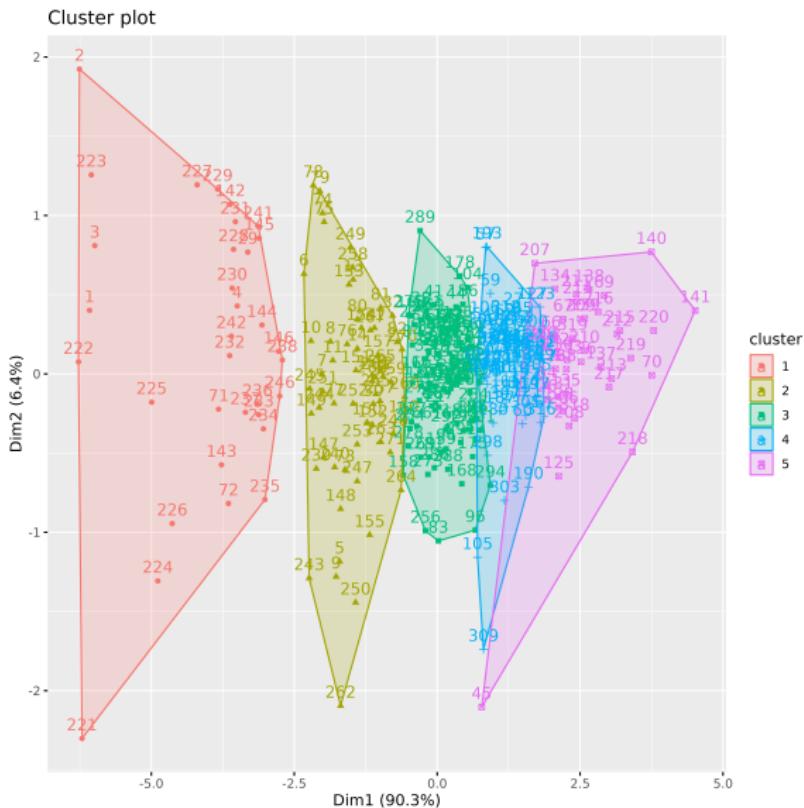
# Median observations
salary_voi[theoretical_run$medoids,]

# Need to supply data back to object
theoretical_run$data <- salary_numeric

# Let's visualize
fviz_cluster(theoretical_run)
```

		school_name	region
232		Cornell University	Northeastern
156		University of Texas (UT) - Austin	Southern
93		Ohio State University (OSU)	Midwestern
191	University of North Carolina at Charlotte (UNCC)		Southern
136	Kent State University		Midwestern
	starting_median	mid_career_median	mid_career_25
232	60300	110000	79800
156	49700	93900	67400
93	44900	83700	60700
191	43100	74000	53200
136	38700	62600	45800
		mid_career_75	
232		160000	
156		129000	
93		116000	
191		99500	
136		87000	

Clustering | k -means/ k -medioids



Cluster Comparison

How do these clusters compare with the five regions these universities were drawn from?

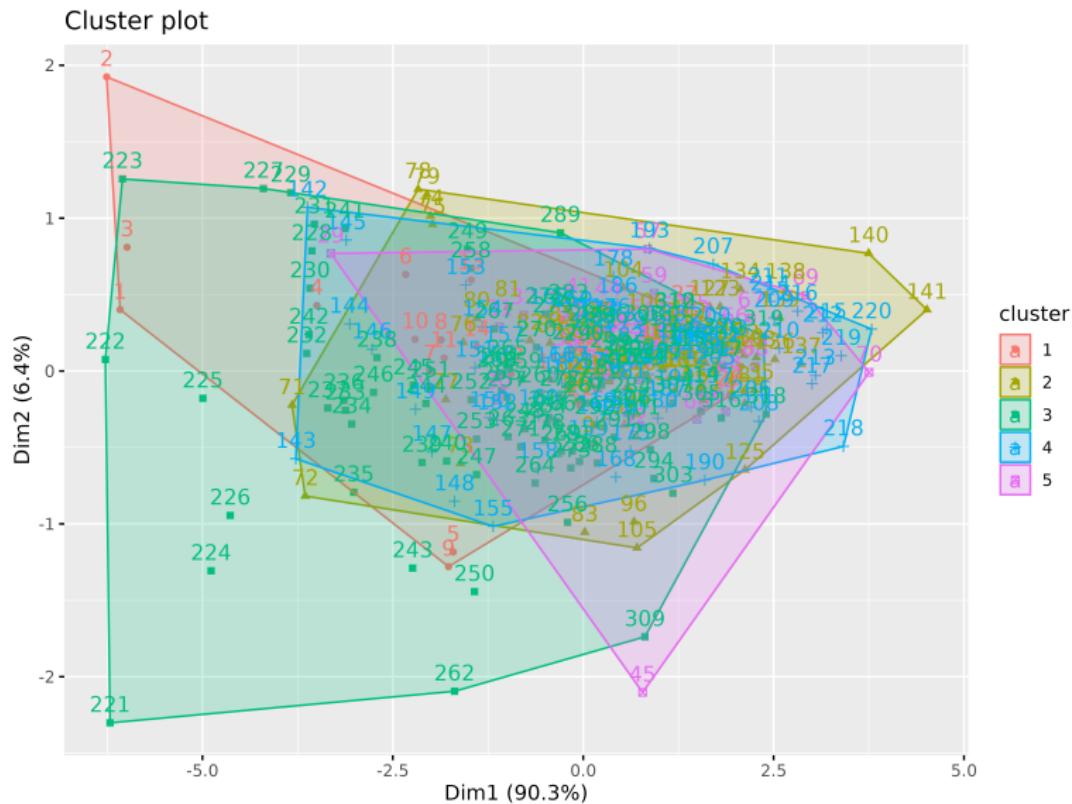
Visualize using “theoretical” clusters

```
# Make copy of theoretical
actual <- theoretical_run

# Replace clusters
actual$clustering <- as.numeric(factor(salary_voi$region))

# What does the region clusters look like?
fviz_cluster(object = actual, data = salary_numeric)
```

Clustering | Cluster Comparison



How do these clusters compare with the five regions these universities were drawn from?

- Adjusted Rand Index (ARI): best for larger and balanced clusters
 - $-1 < \text{ARI} < 0$: negative relationship between clustering
 - $\text{ARI} = 0$: clustering is no different than random
 - $0 < \text{ARI} < 1$: positive similarity between clustering

$$ARI = \frac{\sum_{ij} \binom{n_{ij}}{2} - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2} \right] / \binom{n}{2}}{\frac{1}{2} \left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2} \right] - \left[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2} \right] / \binom{n}{2}}$$

where

- n_{ij} = elements in both cluster i and cluster j
- a_i = elements in cluster i
- b_j = elements in cluster j
- n = total elements
- $\binom{n}{2} = \frac{n(n-1)}{2}$

How do these clusters compare with the five regions these universities were drawn from?

- Normalized Mutual Information (NMI): best for smaller and unbalanced clusters
 - $\text{NMI} = 0$: no relationship between clusters
 - $0 < \text{NMI} < 1$: positive similarity between clusters
 - $\text{NMI} = 1$: identical clusters

$$NMI(U, V) = \frac{2 \cdot I(U; V)}{H(U) + H(V)}$$

where

- H = entropy

$$\sum_{i=1}^K p_k \log(p_k)$$

- $I(U; V)$ = mutual information

$$\sum_{i=1}^{K_U} \sum_{j=1}^{K_V} p(i, j) \log \left(\frac{p(i, j)}{p(i)p(j)} \right)$$

Compare Empirical with Regions

```
## Adjusted Rand Index
compare(
  actual$clustering, theoretical_run$clustering,
  method = "adjusted.rand"
)
```

```
[1] 0.01825695
```

```
## Normalized Mutual Information
compare(
  actual$clustering, theoretical_run$clustering,
  method = "nmi"
)
```

```
[1] 0.06156435
```

Selecting Clusters

- Selecting the number of clusters is a key part of clustering methods
- Although we can set the number of clusters ourselves, there are methods that can help us select the number
- **Grid search** or fitting over a range of possible cluster values (e.g., 2-8 clusters) is usually employed

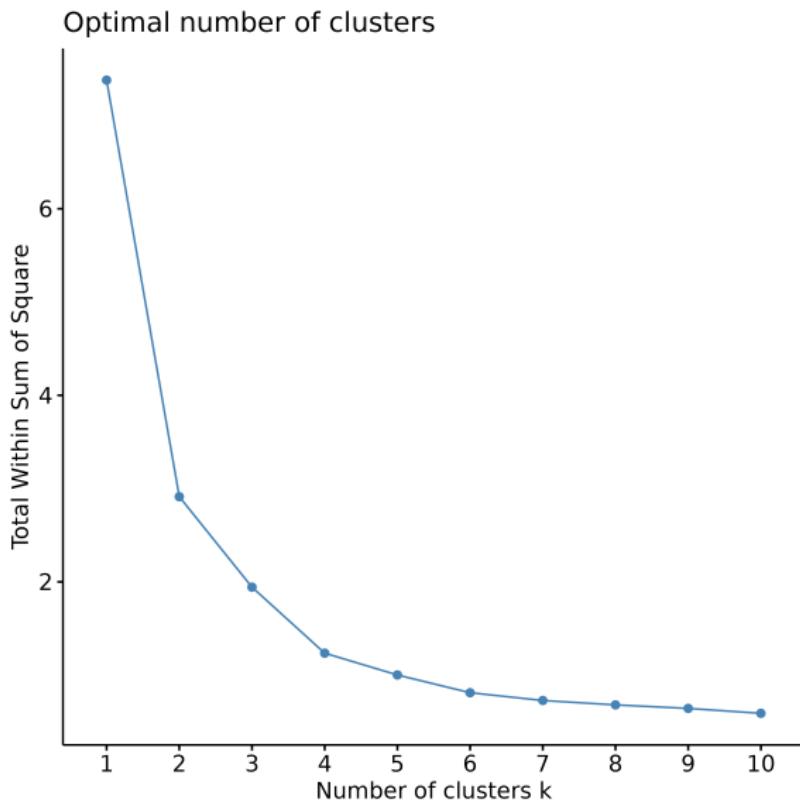
What do you expect to happen to the within-cluster variance (distance) as the number of clusters increases?

"Elbow" Approach

- Goal: identify a “drop off” and “leveling out” of within-cluster variance across the range of clusters

```
# Plot "elbow" method
fviz_nbclust(
  x = salary_numeric, # supply data
  FUNcluster = pam, # cluster function
  diss = salary_distance, # supply distance
  method = "wss", # within-cluster sum of squares
  k = 10, # maximum number of clusters
  nstart = 25 # same as our k-medoids setup
)
```

Clustering | Selecting Clusters



Statistical Approaches

- There is a vast number of statistical approaches
- `{NbClust}` package boasts 30 different methods
- Common methods include:
 - Silhouette method
 - Gap statistic

Silhouette Method

$$a(i) = \frac{1}{|C_I| - 1} \sum_{j \in C_I, i \neq j} d(i, j)$$

- C_I = number of points in cluster i
- $d(i, j)$ = distance (e.g., Euclidean) between points i and j

Silhouette Method

$$b(i) = \min_{J \neq I} \frac{1}{|C_J|} \sum_{j \in C_J} d(i, j)$$

- C_J = number of points in cluster j
- $d(i, j)$ = distance (e.g., Euclidean) between points i and j

Silhouette Method

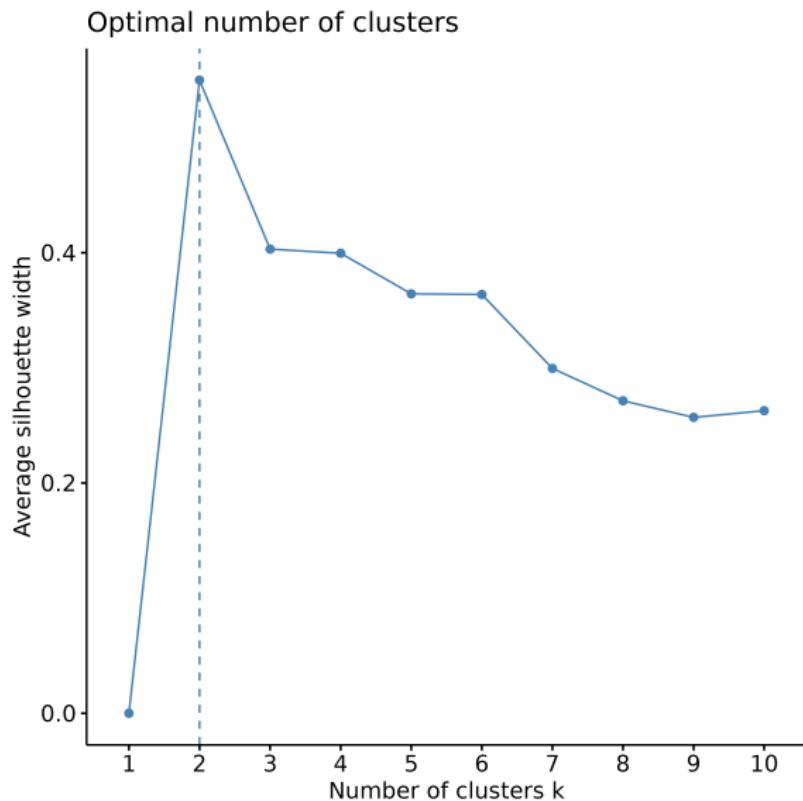
$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

- Average silhouette is used for each number of clusters
- Select clusters that **maximize** average silhouette

Silhouette Method

```
# Plot "elbow" method
fviz_nbclust(
  x = salary_numeric, # supply data
  FUNcluster = pam, # cluster function
  diss = salary_distance, # supply distance
  method = "wss", # within-cluster sum of squares
  k = 10, # maximum number of clusters
  nstart = 25 # same as our k-medoids setup
)
```

Clustering | Selecting Clusters



Majority Method

```
# {NbClust} has over 30 different metrics to evaluate
# the number of clusters -- majority approach:
majority <- NbClust(
  data = salary_numeric, # supply data
  diss = salary_distance, # supply distance
  distance = NULL, # using our own distance
  max.nc = 10, # maximum number of clusters
  method = "median", # perhaps more consistent with mediods
  index = "all" # all metrics
)
```

- * Among all indices:
 - * 9 proposed 2 as the best number of clusters
 - * 3 proposed 3 as the best number of clusters
 - * 1 proposed 4 as the best number of clusters
 - * 2 proposed 5 as the best number of clusters
 - * 4 proposed 6 as the best number of clusters
 - * 3 proposed 8 as the best number of clusters
 - * 2 proposed 10 as the best number of clusters
- * According to the majority rule, the best number of clusters is 2

Final Solution

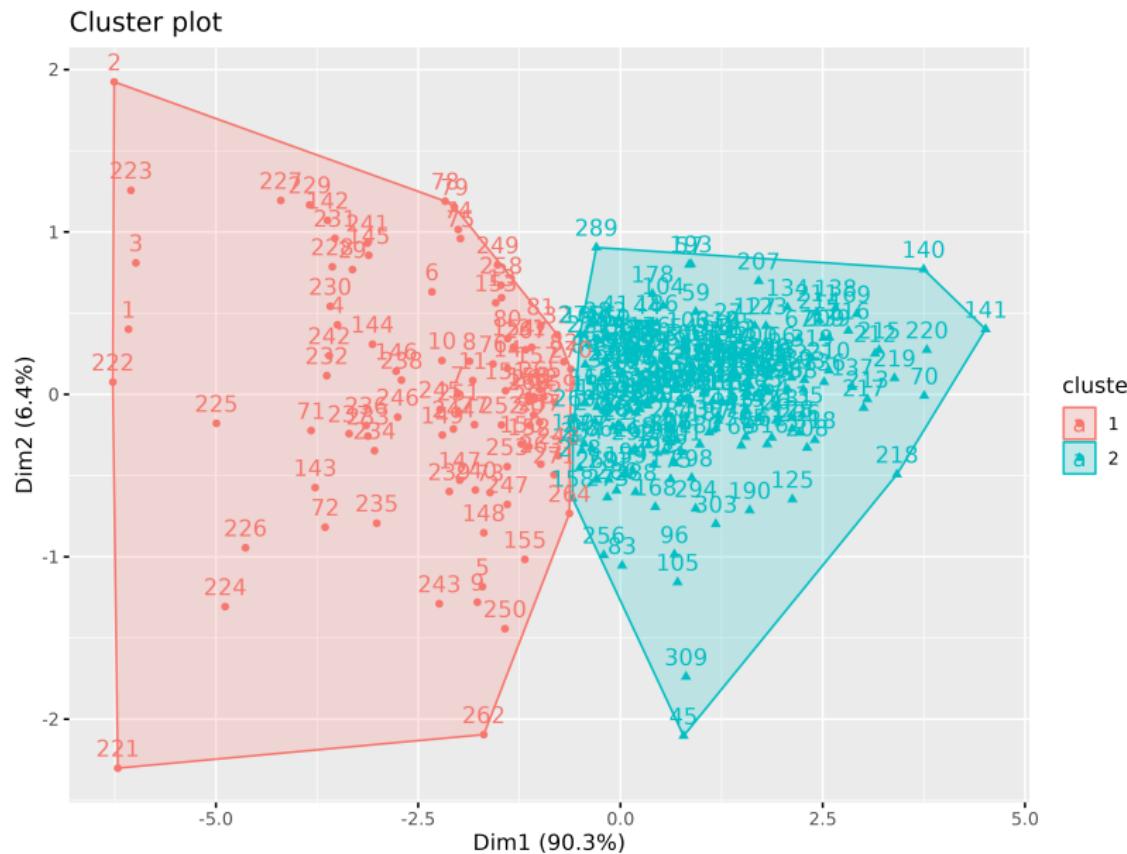
Clustering | Final Solution

```
# Perform k-medoids with silhouette
silhouette_run <- pam(
  x = salary_distance, # supply distance
  k = 2, # number of clusters
  nstart = 25 # number of random starting values
)

# Need to supply data back to object
silhouette_run$data <- salary_numeric

# Plot
fviz_cluster(silhouette_run, salary_numeric)
```

Clustering | Final Solution



Medoids

Median observations

```
salary_voi[silhouette_run$medoids,]
```

		school_name	region	starting_median
7	University of California at Los Angeles (UCLA)	California		52600
185	West Virginia University (WVU)	Southern		43100
	mid_career_median mid_career_25 mid_career_75			
7	101000	72500	139000	
185	78100	55700	106000	

Higher versus lower outcome in terms of median salary

Where did Vanderbilt fall?

Where does Vanderbilt come out?

```
silhouette_run$cluster[grep("Vanderbilt", salary$school_name)]
```

```
[1] 1
```

In-class Activity

- ① Perform hierarchical clustering on the salary_data using salary_distance (Gower's distance)
- ② Apply the Silhouette Method to determine the number of clusters
- ③ Extract clusters using cutree
- ④ Visualize clusters

Hierarchical

Goal: bring together/split “like” observations based on distance, one observation/cluster at a time

Like k -means/medioids, hierarchical “learns” in an *unsupervised* manner

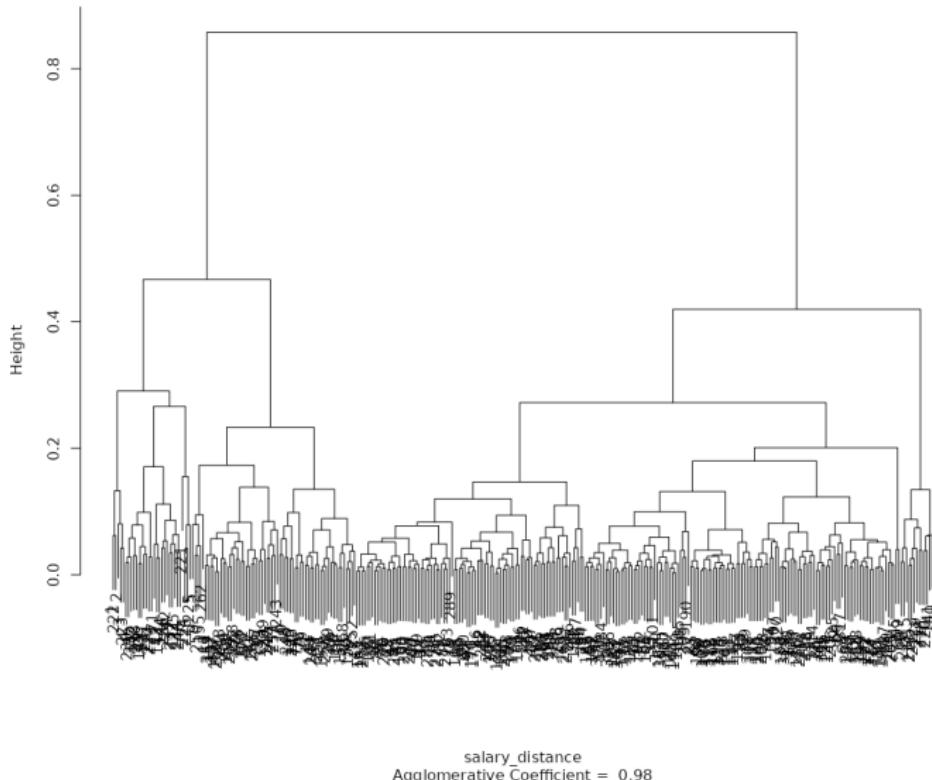
How does hierarchical clustering “learn” the labels or sub-groups of the data?

Algorithm

- Step 1: Decide on either **agglomerative** (“bottom-up”) or **divisive** (“top-down”) method
- Step 2: Compute (dis)similarity or *distance* measure between observations
- Step 3: Use a *linkage* function to **join**/**split** observations
- Step 4: Join/split **observations**/**clusters** that are most (dis)similar of all possible distance values
- Step 5: Repeat Steps 4 until there is **one cluster** or **individual observations**

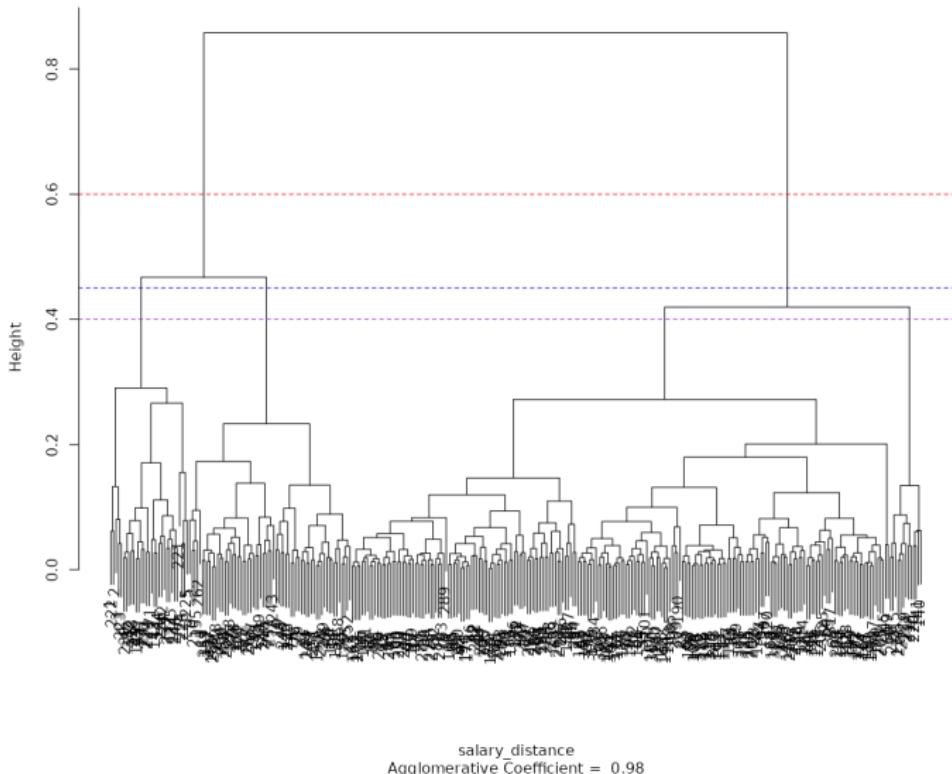
Clustering | Hierarchical

Dendrogram of `agnes(x = salary_distance, method = "complete")`



Clustering | Hierarchical

Dendrogram of `agnes(x = salary_distance, method = "complete")`



Step 1: Decide on **agglomerative** or **divisive** method

- **agglomerative**: starts with individual observations and joins them, one-by-one, until all observations form a single cluster (“bottom-up”)
- **divisive**: starts with all observations in a single cluster and splits them, one-by-one, until all observations are individual observations (“top-down”)

Step 2: Compute **(dis)similarity** or **distance measure** between observations

Step 3: Use a *linkage* function

Method	Equation	Description
Single	$\min_{i,j} d(\mathbf{X}_i, \mathbf{Y}_j)$	minimize distance between clusters
Complete	$\max_{i,j} d(\mathbf{X}_i, \mathbf{Y}_j)$	maximize distance between clusters
Average	$\frac{\sum_{i=1}^k \sum_j d(\mathbf{X}_i, \mathbf{Y}_j)}{kl}$	average distance between clusters
Centroid	$d(\bar{x}, \bar{y})$	distance between average location of cluster points
Ward's D	too complex for this table	minimizes sum of squares (like K-means)

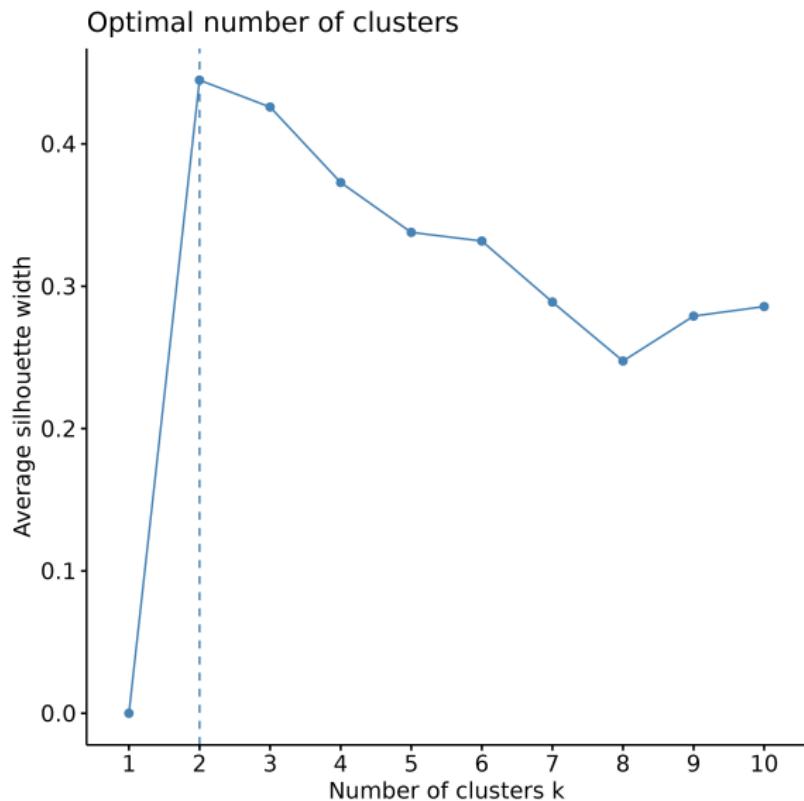
Step 4: Join/split **observations**/**clusters** that are most
(dis)similar of all possible distance values

Step 5: Repeat Steps 4 until there is **one cluster** or **individual observations**

Selecting Clusters: same as k -means/medioids

```
# Plot silhouette method
fviz_nbclust(
  x = salary_numeric, # supply data
  FUNcluster = hcut, # cluster function
  hc_func = "agnes", # ensures same approach
  hc_method = "complete", # ensures same approach
  diss = salary_distance, # supply distance
  method = "silhouette" # within-cluster sum of squares
)
```

Clustering | Hierarchical



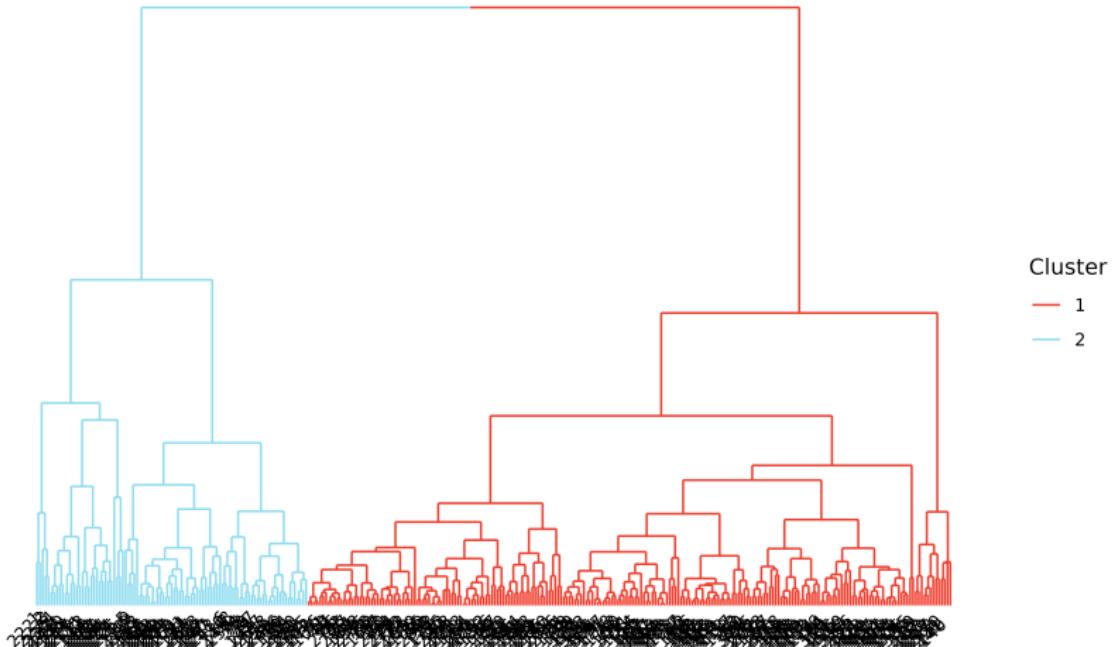
Selecting Clusters: make the “cut”

```
# Get agglomerative clusters based on silhouette
silhouette_clusters <- cutree(hierarchical, k = 2)

# Plot with clusters
pseudo_info <- list(
  clusters = silhouette_clusters,
  clusterTree = hierarchical,
  JSD = as.matrix(salary_distance)
)
# Set class
class(pseudo_info) <- "infoCluster"

# Plot
plot(pseudo_info)
```

Clustering | Hierarchical



At Home Activity

Bipolar/Depression Dataset [[source](#)]

- Actual outcome: Expert.Diagnose ("Normal", "Depression", "Bipolar Type-1", "Bipolar Type-2")
- Variables of interest:
 - binary (yes/no): Mood.Swing, Suicidal.thoughts, Anoxia, Authority.Respect, Try.Explanation, Aggressive.Response, Ignore...Move.On, Nervous.Break.down, Admit.Mistakes, Overthinking
 - ordinal (1-4): Sadness, Euphoric, Exhausted, Sleep.dissorder
 - continuous (1-10): Sexual.Activity, Concentration, Optimism

Goal: use the variables of interest to derive clusters and compare them to the Expert.Diagnose

At Home Activity

- Using the `bipolar_depression_clean.csv`,
 - Use columns 2-18 as your features (see previous slide)
- Apply *either* k -mediods or hierarchical clustering
 - Compute Gower's distance
 - Perform silhouette or majority (use "median") method
 - Visualize
- Compare clusters derived from your analysis with `Expert.Diagnose`
 - Compute Adjusted Rand Index
 - Compute Normalized Mutual Information

Readings for Next Week

Readings

- ESL Chapters: 17.1-17.3
- Epskamp and Fried - 2018
- Golino and Epskamp - 2017

Optional

- Pons and Latapy -2006
- Christensen - 2024
- Schmittmann et al. - 2013
- Blondel et al. - 2008
- Christensen et al. - 2023 - community
- Golino et al. - 2020