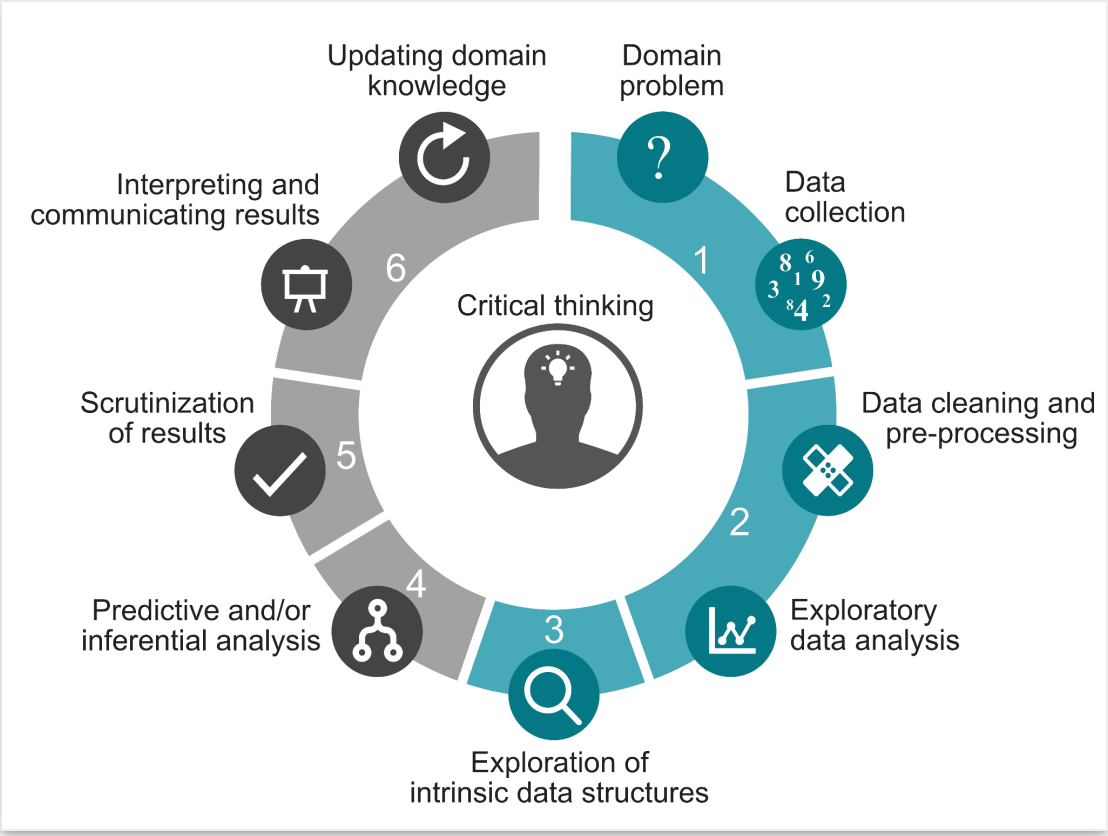
Introduction to Unsupervised Learning

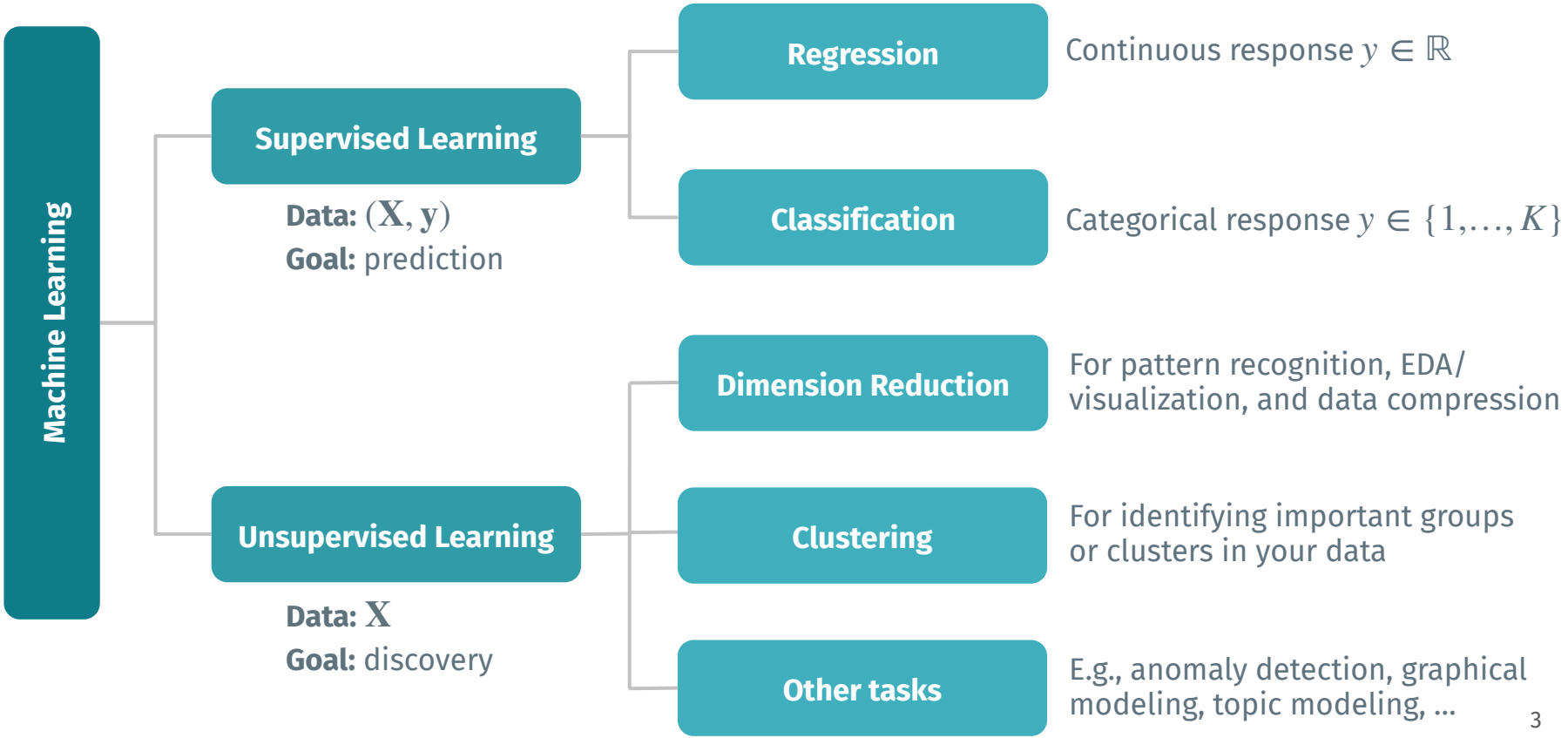
February 10, 2026

The Big Picture: Data Science Life Cycle

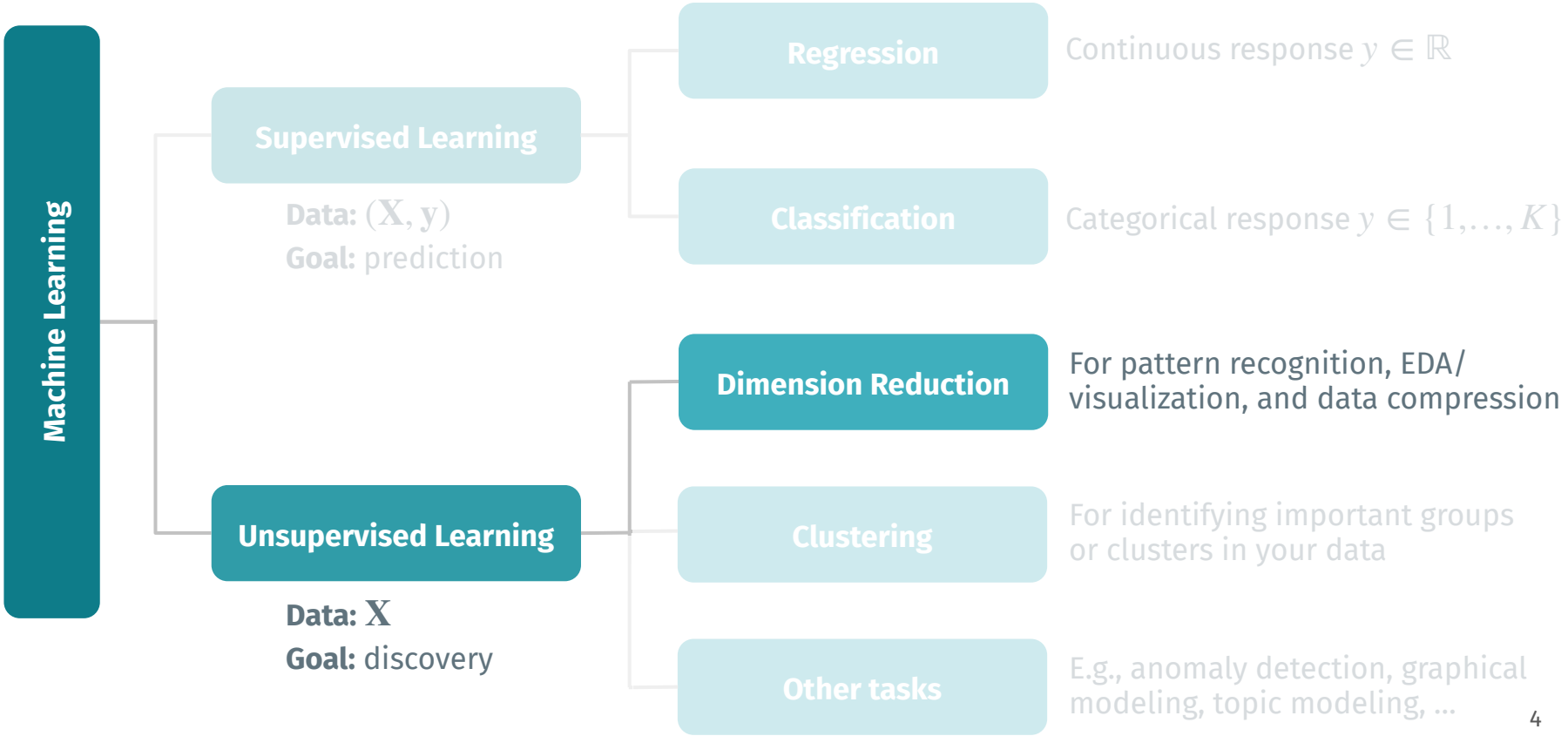


Credits: Veridical Data Science (VDS) Textbook

Last Time


Machine Learning
Supervised Learning
Data: $(\mathbf{X}, \mathbf{y})$
Goal: prediction
Regression
Continuous response $y \in \mathbb{R}$
Classification
Categorical response $y \in \{1, \ldots, K\}$
Unsupervised Learning
Data: $\mathbf{X}$
Goal: discovery
Dimension Reduction
For pattern recognition, EDA/ visualization, and data compression
Clustering
For identifying important groups or clusters in your data
Other tasks
E.g., anomaly detection, graphical modeling, topic modeling, ...

Last Time


Machine Learning
Supervised Learning
Data: $(\mathbf{X}, \mathbf{y})$
Goal: prediction
Regression
Continuous response $y \in \mathbb{R}$
Classification
Categorical response $y \in \{1, \ldots, K\}$
Unsupervised Learning
Data: $\mathbf{X}$
Goal: discovery
Dimension Reduction
For pattern recognition, EDA/ visualization, and data compression
Clustering
For identifying important groups or clusters in your data
Other tasks
E.g., anomaly detection, graphical modeling, topic modeling, ...

Recap: Dimension Reduction Methods

	PCA	t-SNE	UMAP	Autoencoders
Feature Interpretability	Yes	No	No	No
Linear/nonlinear	Linear	Nonlinear	Nonlinear	Nonlinear
Number of components	Orthogonal, nested; Can compute all *p* components at once	Non-nested; need to re-run for each choice of rank (typically, rank = 2 or 3)	Non-nested; need to re-run for each choice of rank (typically, rank = 2 or 3)	Non-nested; need to re-run for each choice of rank
Computation	Fast	Slower	Slower; faster than t-SNE	Expensive (best on GPU)
Unique, global solution	Yes	Local solution (results can change with different seed)	Local solution (results can change with different seed)	Local solution (results can change with different seed)
Other considerations?	No hyperparameters	Results can change drastically depending on hyperparameters; Not good at preserving global structure; Typically do PCA before inputting into tSNE	Results can change drastically depending on hyperparameters; Better at preserving global structure than tSNE; Typically do PCA before inputting into tSNE	Results can change drastically depending on architecture

Today's Plan

1. Supervised vs **Unsupervised** Learning
2. Overview of Popular **Dimension Reduction** Methods
3. Overview of Popular **Clustering** Methods
4. **Model Selection** and **Evaluation**
5. **In-Class Lab:** Linguistics Data

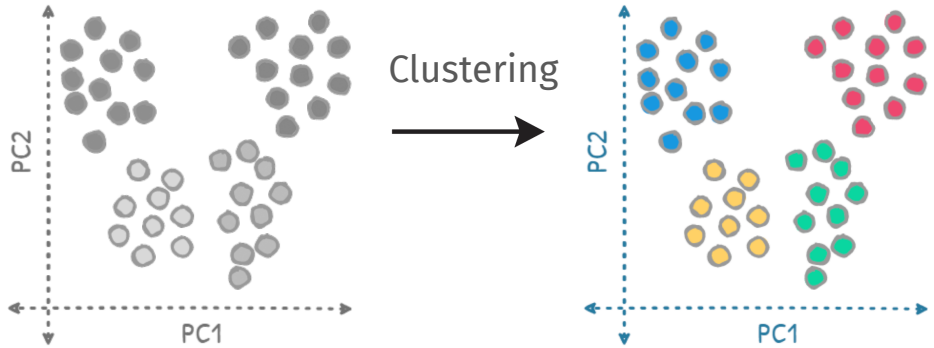
Clustering

Clustering

Clustering: aims to identify groups/clusters of observations that are "similar"

Popular approaches:

- k-means clustering
- Hierarchical clustering
- Spectral clustering



k-means Clustering

For a pre-specified k:

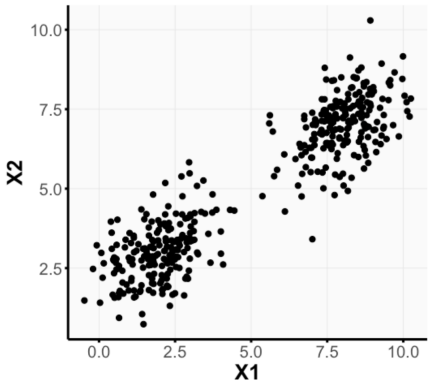
- **Idea:** find k clusters which result in the "tightest" groups (i.e., has the smallest within-cluster variance)

k-means Clustering

For a pre-specified k:

- **Idea:** find k clusters which result in the "tightest" groups (i.e., has the smallest within-cluster variance)

Original 2D Data

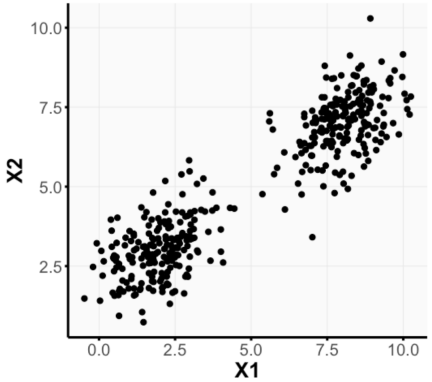


k-means Clustering

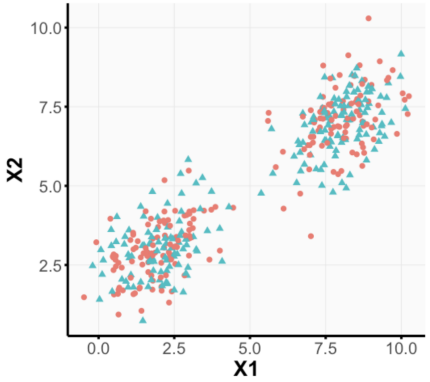
For a pre-specified k:

- **Idea:** find k clusters which result in the "tightest" groups (i.e., has the smallest within-cluster variance)

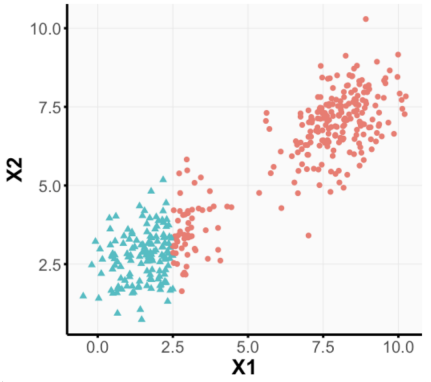
Original 2D Data



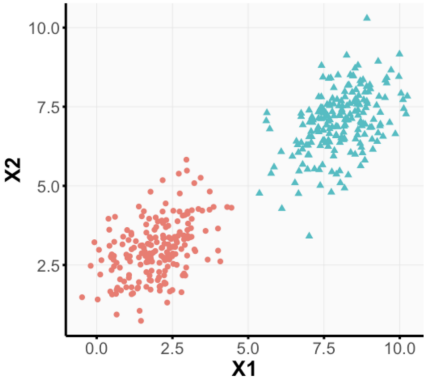
Candidate Cluster A



Candidate Cluster B



Candidate Cluster C

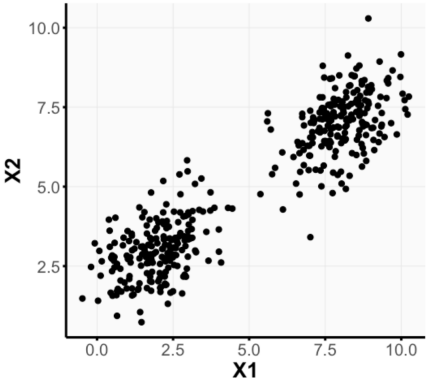


k-means Clustering

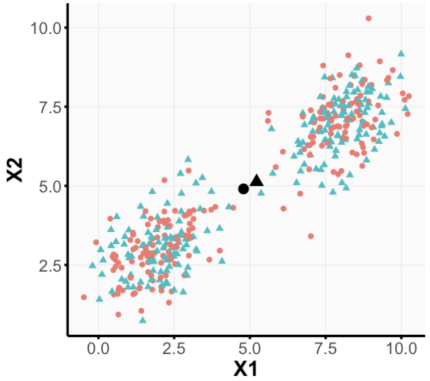
For a pre-specified k:

- **Idea:** find k clusters which result in the "tightest" groups (i.e., has the smallest within-cluster variance)

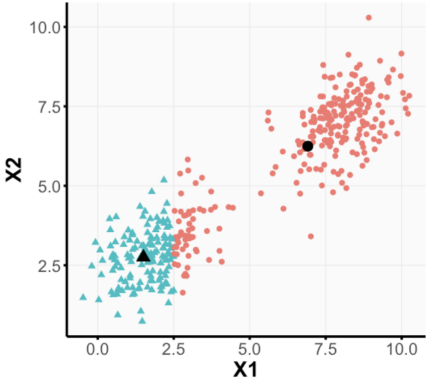
Original 2D Data



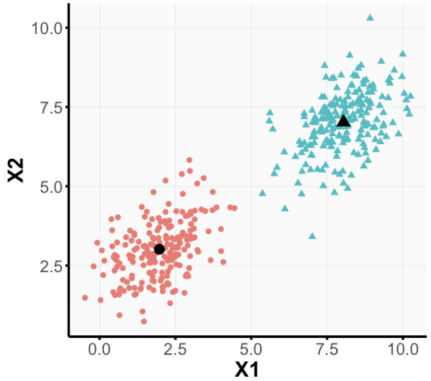
Candidate Cluster A



Candidate Cluster B



Candidate Cluster C

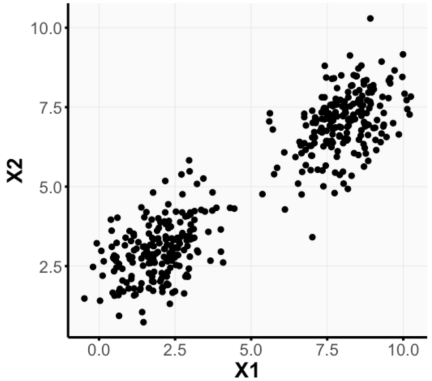


k-means Clustering

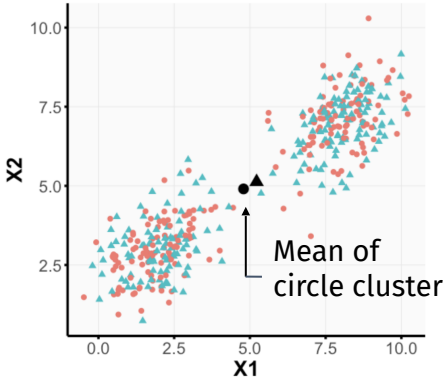
For a pre-specified k:

- **Idea:** find k clusters which result in the "tightest" groups (i.e., has the smallest within-cluster variance)

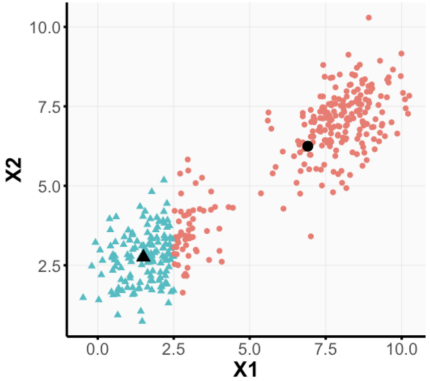
Original 2D Data



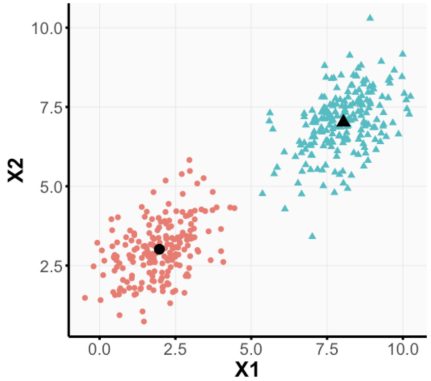
Candidate Cluster A



Candidate Cluster B



Candidate Cluster C

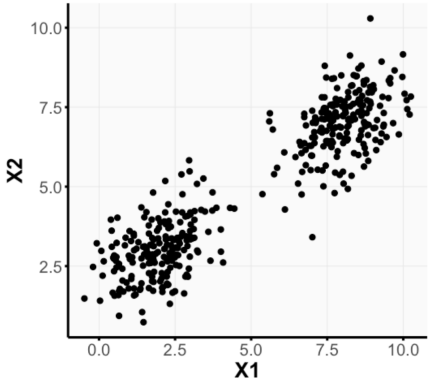


k-means Clustering

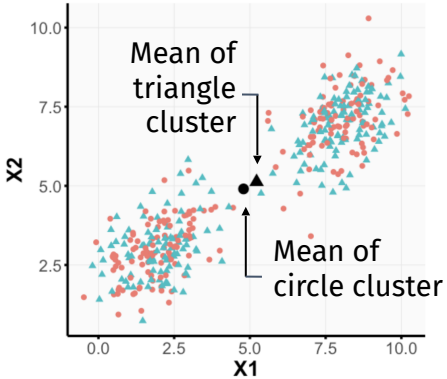
For a pre-specified k:

- **Idea:** find k clusters which result in the "tightest" groups (i.e., has the smallest within-cluster variance)

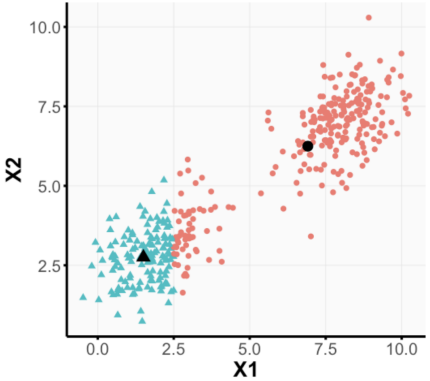
Original 2D Data



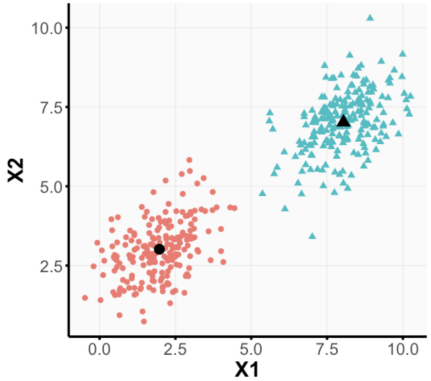
Candidate Cluster A



Candidate Cluster B



Candidate Cluster C

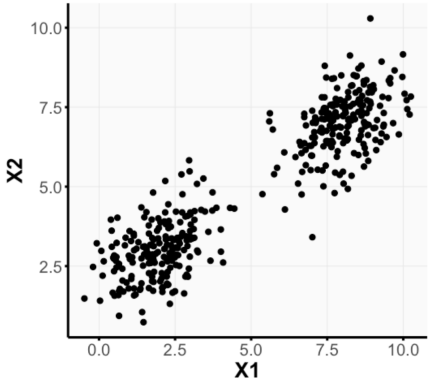


k-means Clustering

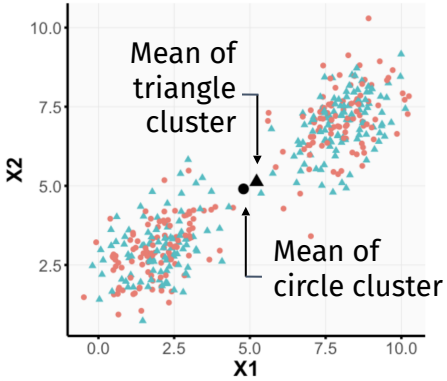
For a pre-specified k:

- **Idea:** find k clusters which result in the "tightest" groups (i.e., has the smallest within-cluster variance)

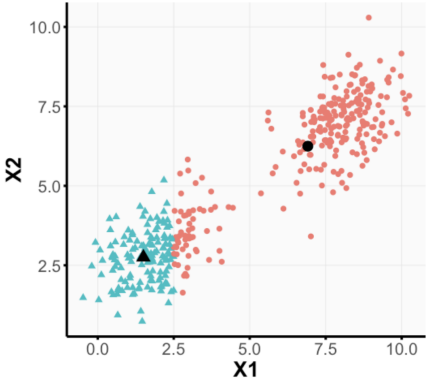
Original 2D Data



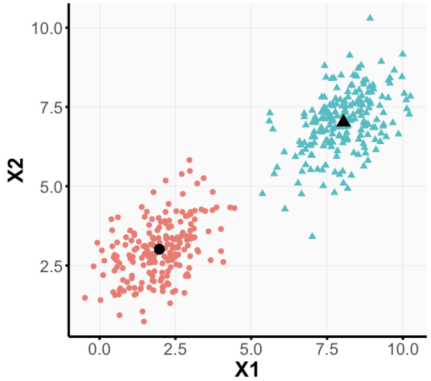
Candidate Cluster A



Candidate Cluster B



Candidate Cluster C



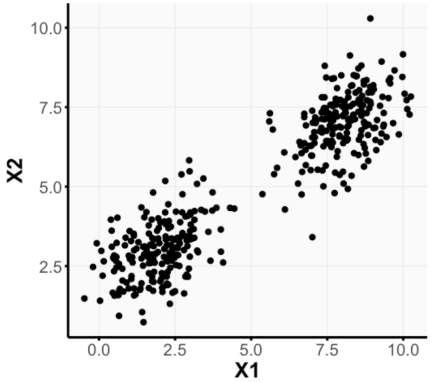
Blue: high variance
Red: high variance

k-means Clustering

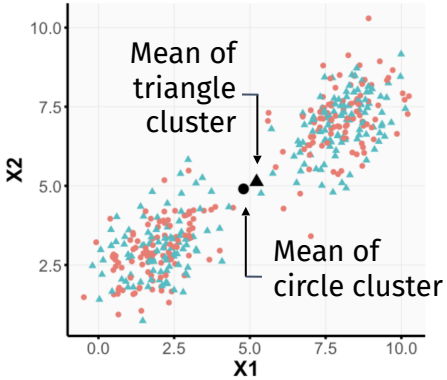
For a pre-specified k:

- **Idea:** find k clusters which result in the "tightest" groups (i.e., has the smallest within-cluster variance)

Original 2D Data

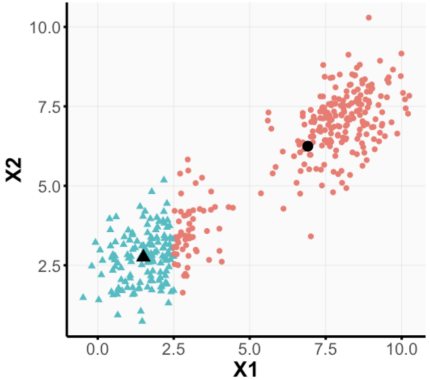


Candidate Cluster A



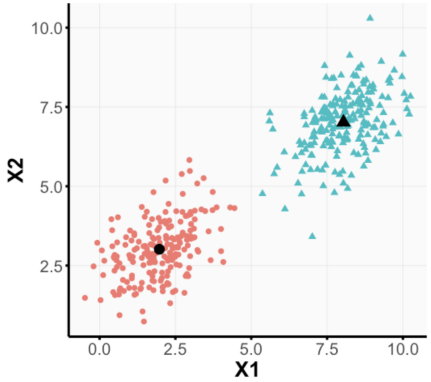
Blue: high variance
Red: high variance

Candidate Cluster B



Blue: low variance
Red: high variance

Candidate Cluster C

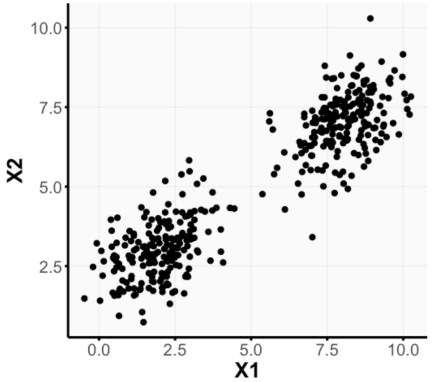


k-means Clustering

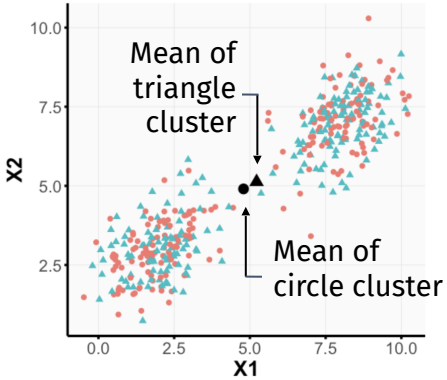
For a pre-specified k:

- **Idea:** find k clusters which result in the "tightest" groups (i.e., has the smallest within-cluster variance)

Original 2D Data

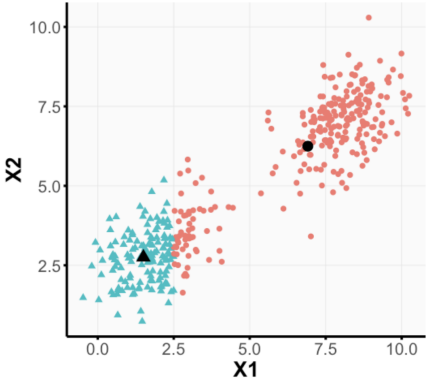


Candidate Cluster A



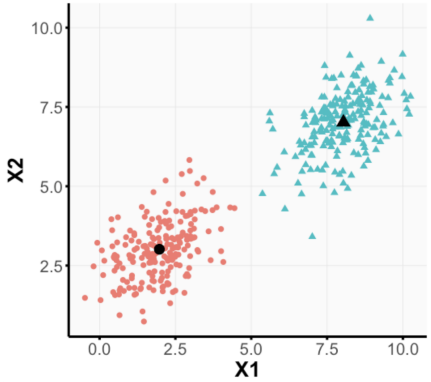
Blue: high variance
Red: high variance

Candidate Cluster B



Blue: low variance
Red: high variance

Candidate Cluster C



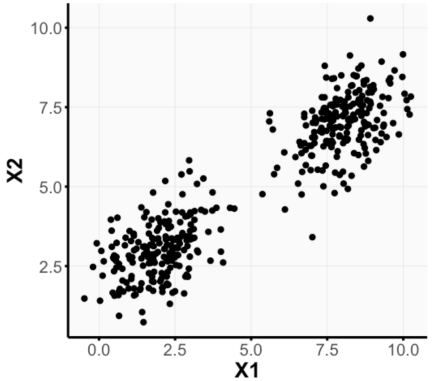
Blue: low variance
Red: low variance

k-means Clustering

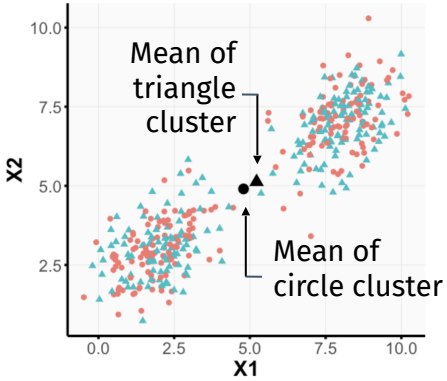
For a pre-specified k:

- **Idea:** find k clusters which result in the "tightest" groups (i.e., has the smallest within-cluster variance)

Original 2D Data

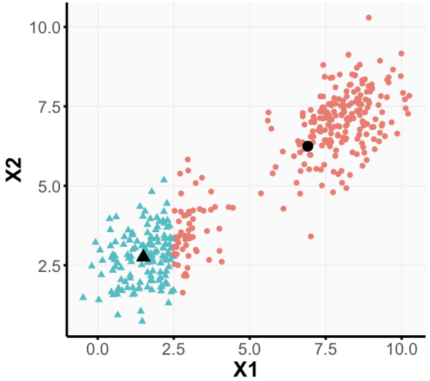


Candidate Cluster A



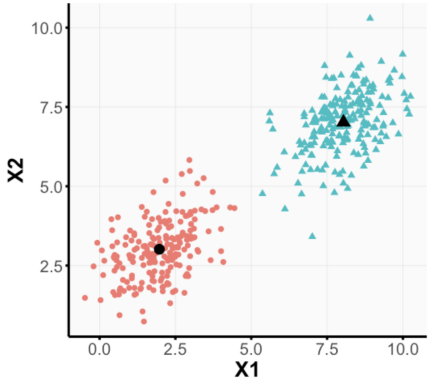
Blue: high variance
Red: high variance

Candidate Cluster B



Blue: low variance
Red: high variance

Candidate Cluster C

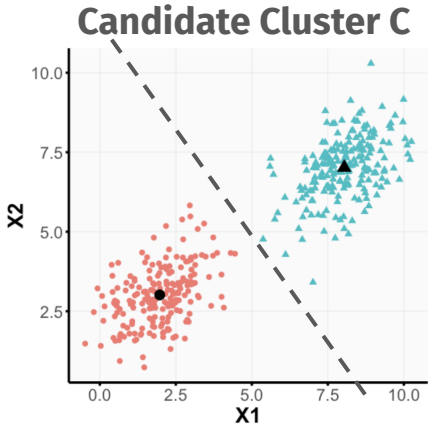


Blue: low variance
Red: low variance

k-means Clustering

For a pre-specified k:

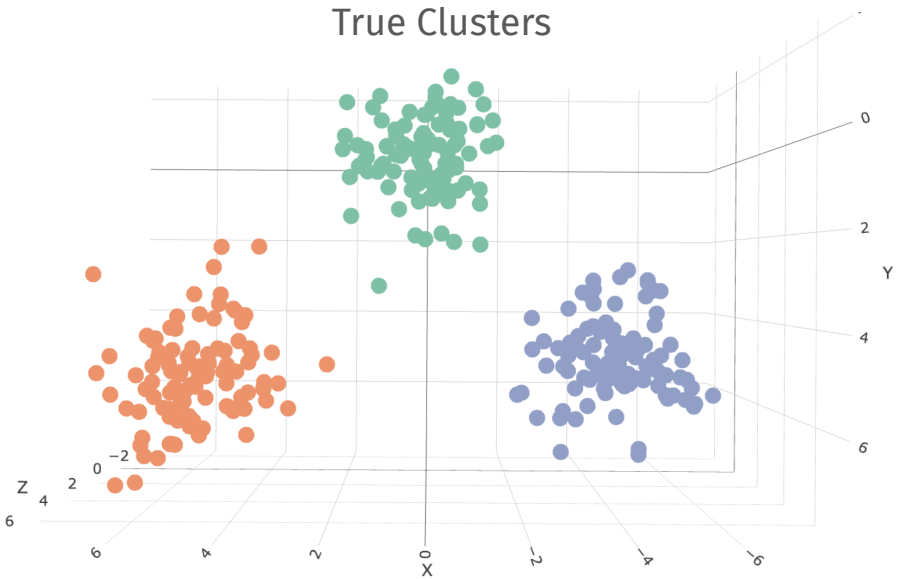
- **Idea:** find k clusters which result in the "tightest" groups (i.e., has the smallest within-cluster variance)



Points get clustered to the closest centroid

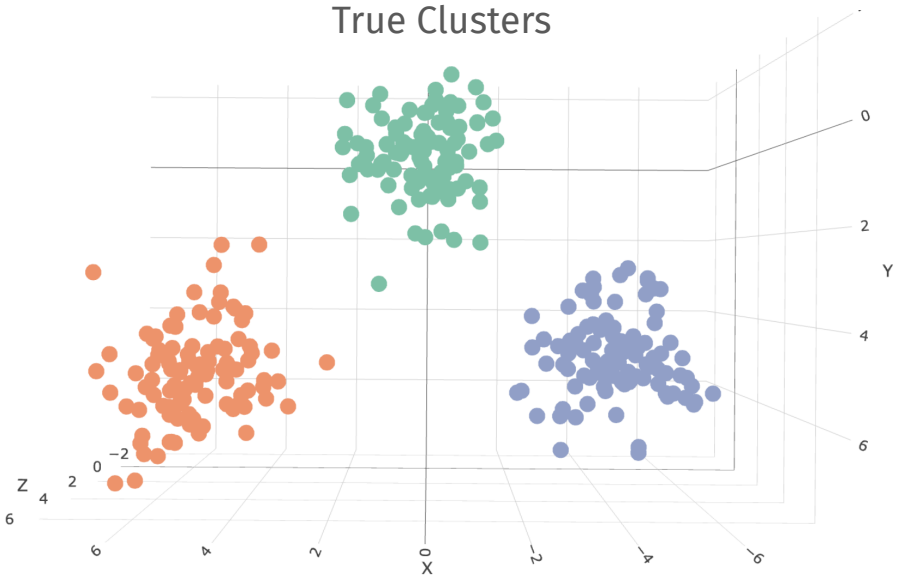
When does k-means "work" and when does k-means "not work"?

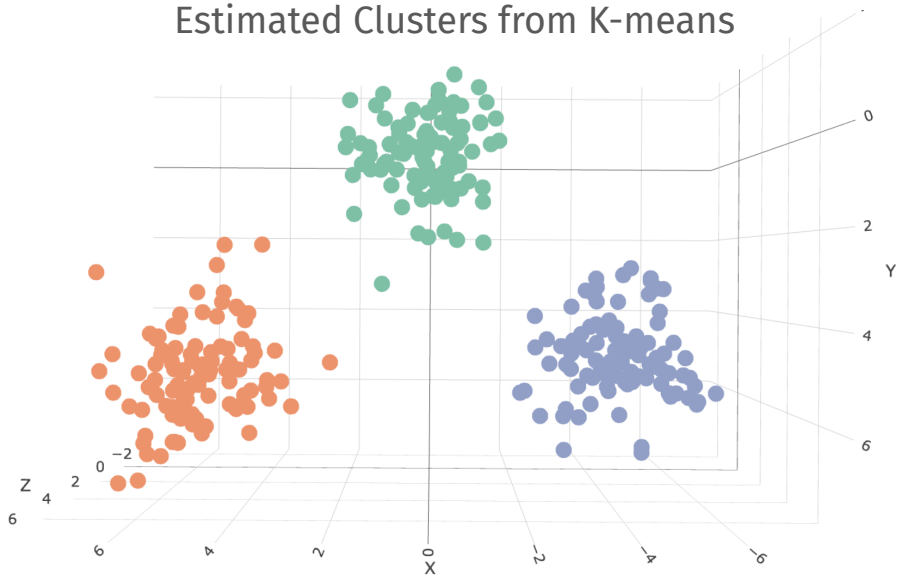
Scenario: Spherical, linearly-separable clusters



When does k-means "work" and when does k-means "not work"?

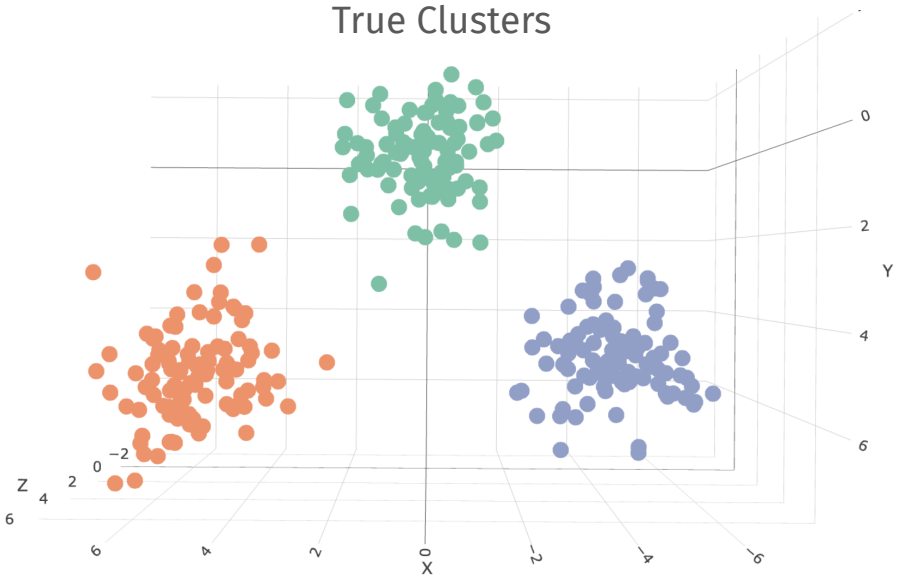
Scenario: Spherical, linearly-separable clusters

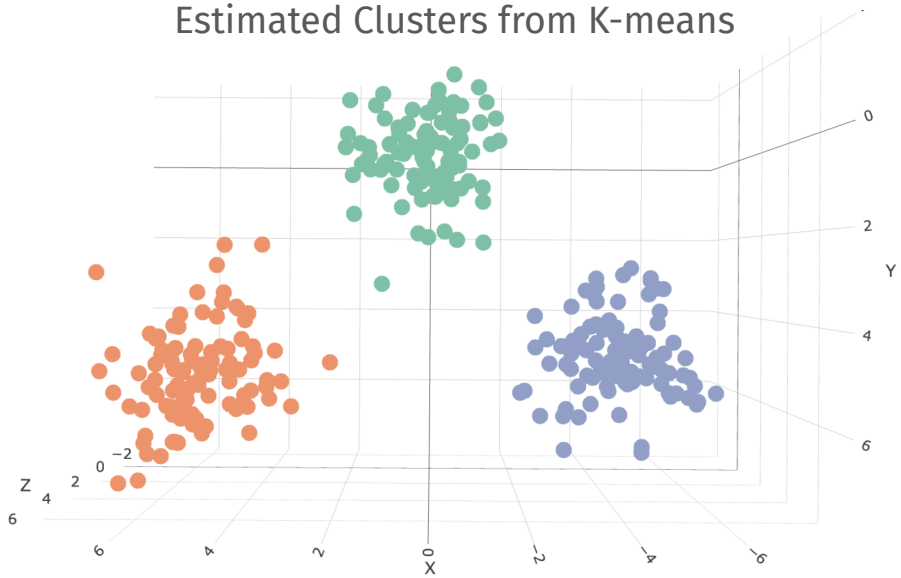




When does k-means "work" and when does k-means "not work"?

Scenario: Spherical, linearly-separable clusters





✔ This is the ideal scenario for K-means

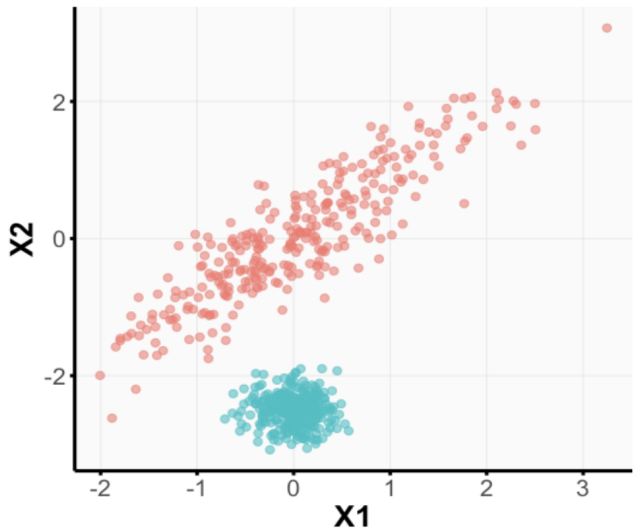
When does k-means "work" and when does k-means "not work"?

Scenario: Non-spherical clusters

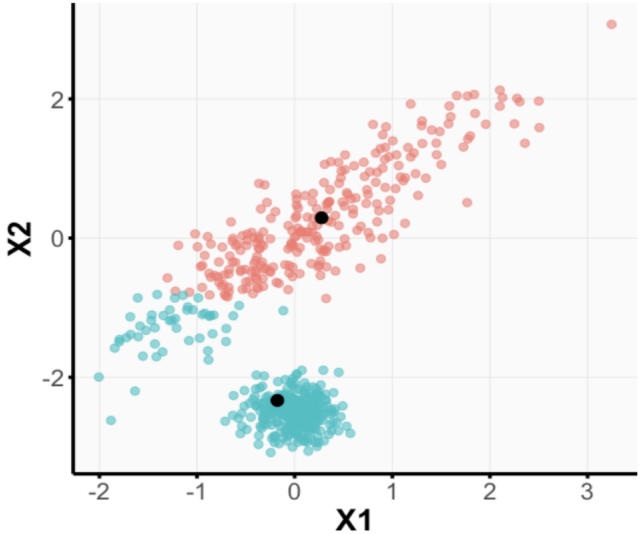
When does k-means "work" and when does k-means "not work"?

Scenario: Non-spherical clusters

True Clusters



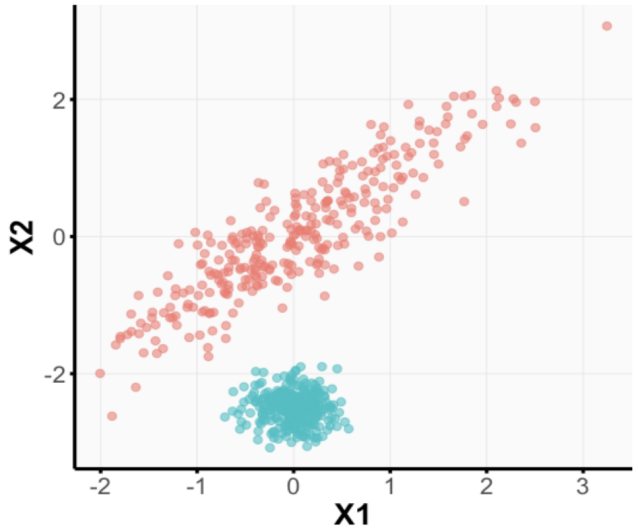
Estimated Clusters from K-means



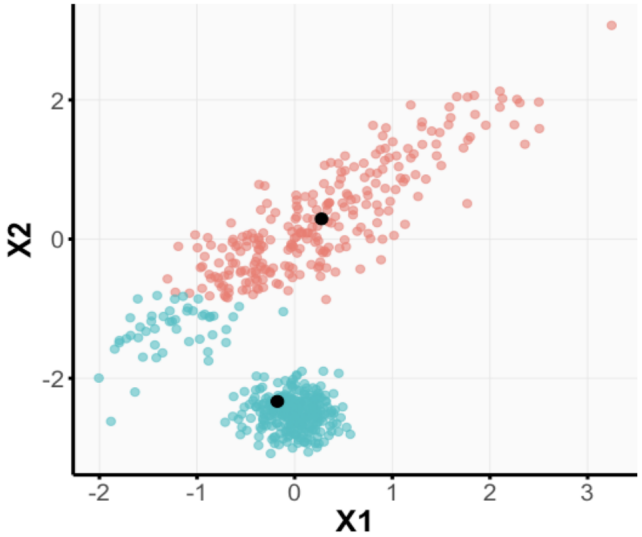
When does k-means "work" and when does k-means "not work"?

Scenario: Non-spherical clusters

True Clusters



Estimated Clusters from K-means



✘ Not great for non-spherical clusters

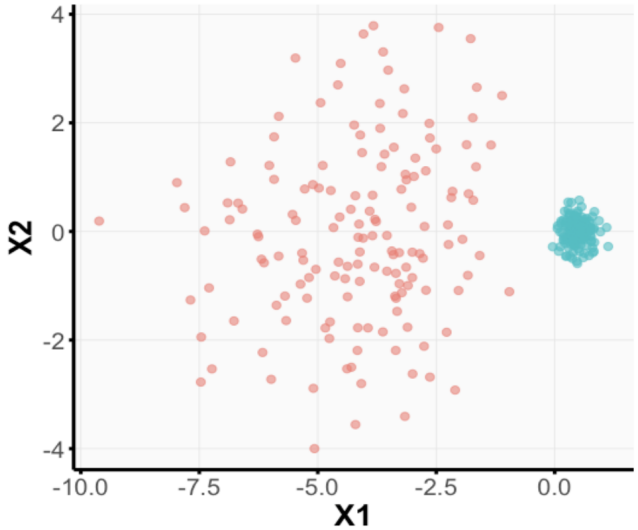
When does k-means "work" and when does k-means "not work"?

Scenario: Clusters with different variances

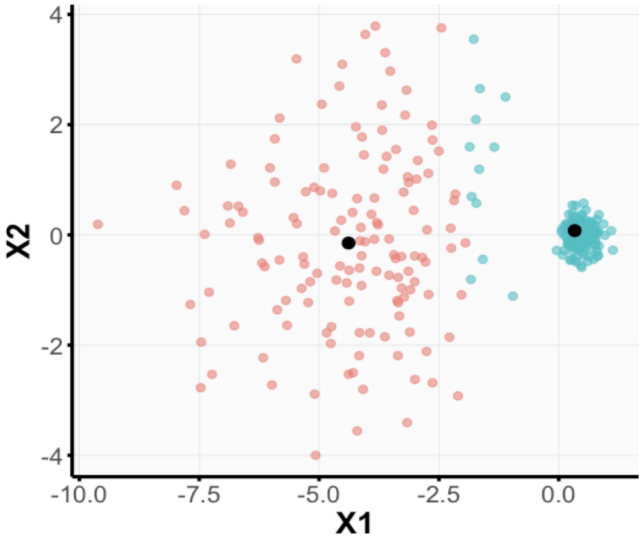
When does k-means "work" and when does k-means "not work"?

Scenario: Clusters with different variances

True Clusters



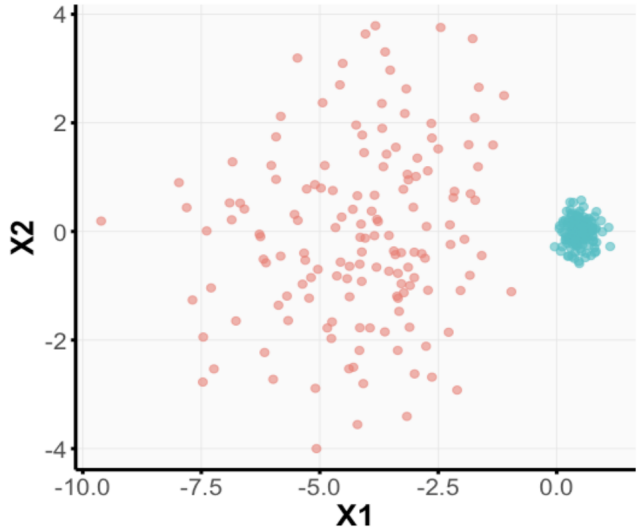
Estimated Clusters from K-means



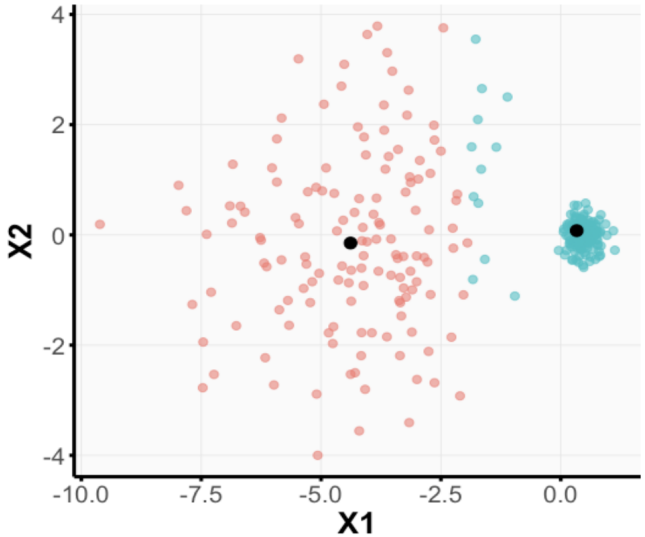
When does k-means "work" and when does k-means "not work"?

Scenario: Clusters with different variances

True Clusters



Estimated Clusters from K-means



✘ Not great for clusters with different variances

Practical considerations when running k-means

By far the most difficult part of getting k-means to work is **choosing the right number of clusters** k

- This k must be chosen **before** running k-means
- For every k under consideration, we need to re-run k-means

In contrast, hierarchical clustering allows us to run the clustering algorithm once and then choose *K* afterwards

Hierarchical Clustering

- A **greedy, agglomerative** algorithm
  - At the lowest level, each cluster contains a single observation
  - As we move up the tree, some leaves begin to fuse into branches – these are observations that are most **similar** to each other

Hierarchical Clustering

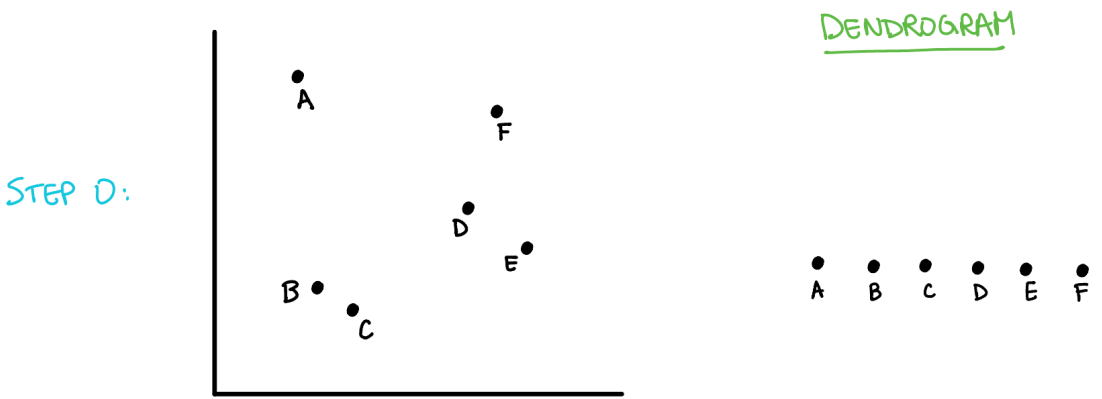
- A **greedy, agglomerative** algorithm
  - At the lowest level, each cluster contains a single observation
  - As we move up the tree, some leaves begin to fuse into branches – these are observations that are most **similar** to each other

Initialization (Step 0): Each point starts as its own singleton cluster

Hierarchical Clustering

- A **greedy, agglomerative** algorithm
  - At the lowest level, each cluster contains a single observation
  - As we move up the tree, some leaves begin to fuse into branches – these are observations that are most **similar** to each other

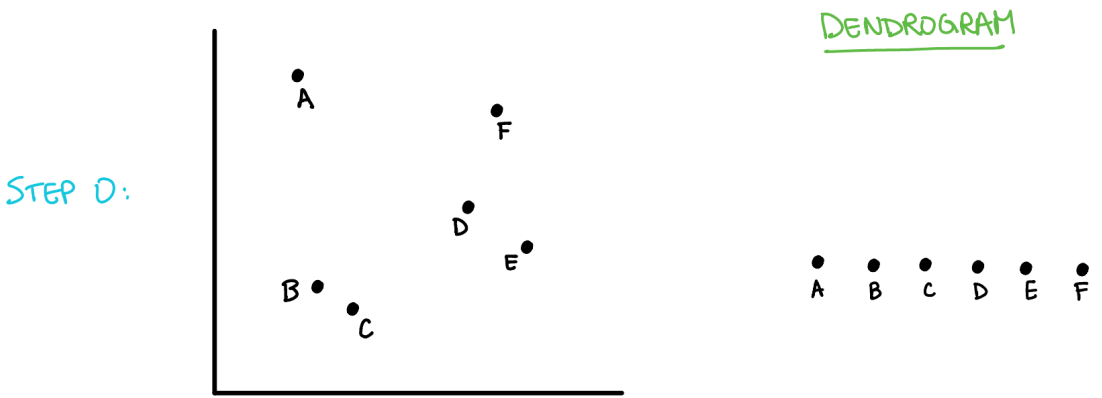
Initialization (Step 0): Each point starts as its own singleton cluster



Hierarchical Clustering

- A **greedy, agglomerative** algorithm
  - At the lowest level, each cluster contains a single observation
  - As we move up the tree, some leaves begin to fuse into branches – these are observations that are most **similar** to each other

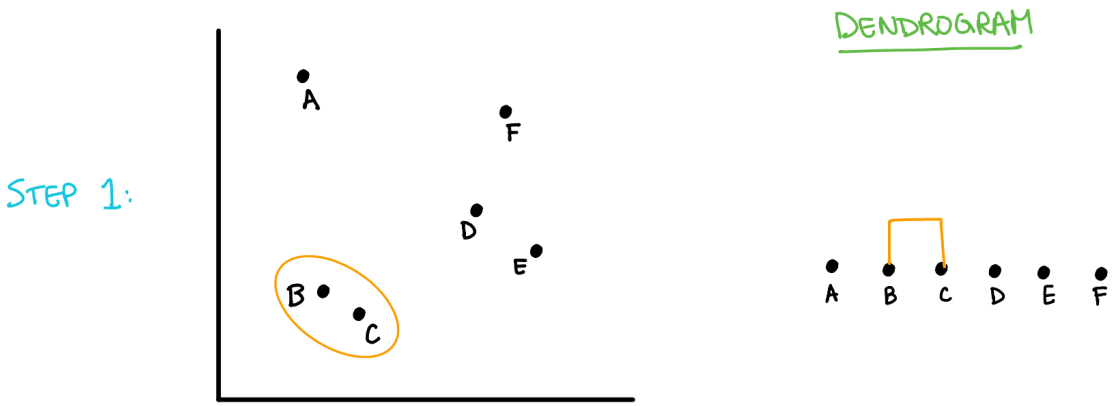
Next Step: Join the two points/clusters that are "closest" together



Hierarchical Clustering

- A **greedy, agglomerative** algorithm
  - At the lowest level, each cluster contains a single observation
  - As we move up the tree, some leaves begin to fuse into branches – these are observations that are most **similar** to each other

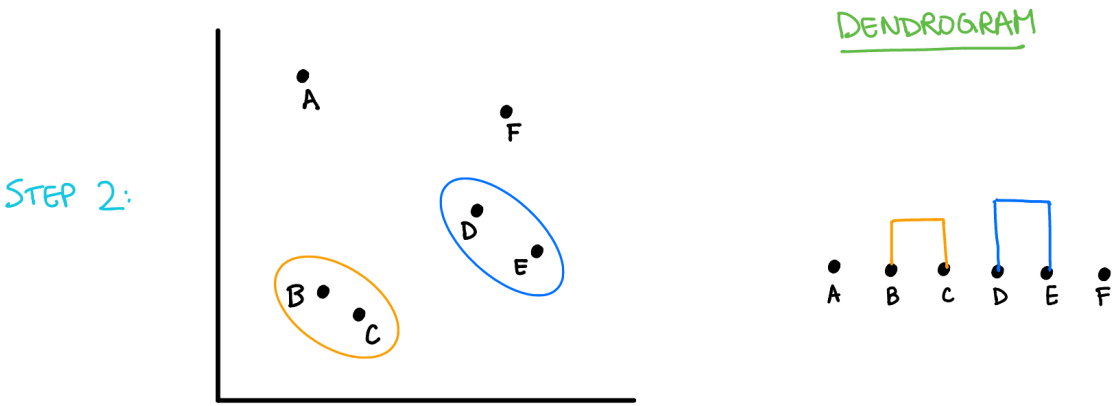
Next Step: Join the two points/clusters that are "closest" together



Hierarchical Clustering

- A **greedy, agglomerative** algorithm
  - At the lowest level, each cluster contains a single observation
  - As we move up the tree, some leaves begin to fuse into branches – these are observations that are most **similar** to each other

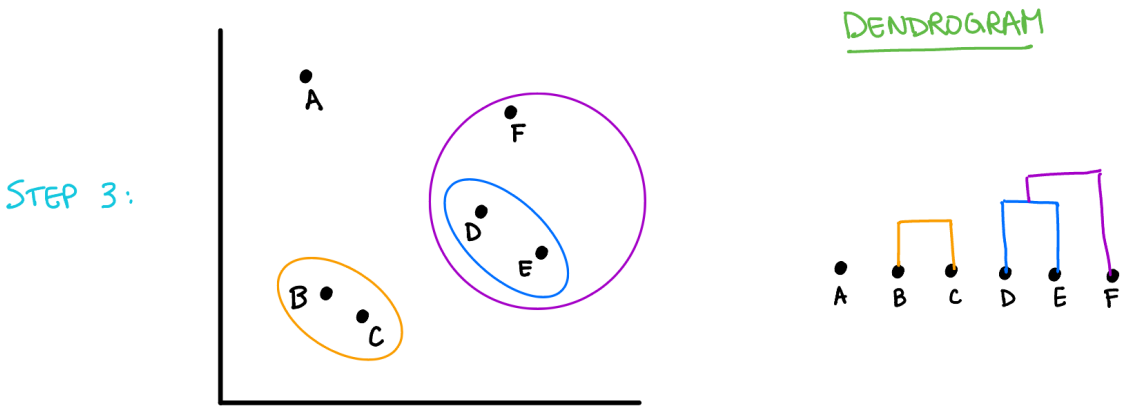
Next Step: Join the two points/clusters that are "closest" together



Hierarchical Clustering

- A **greedy, agglomerative** algorithm
  - At the lowest level, each cluster contains a single observation
  - As we move up the tree, some leaves begin to fuse into branches – these are observations that are most **similar** to each other

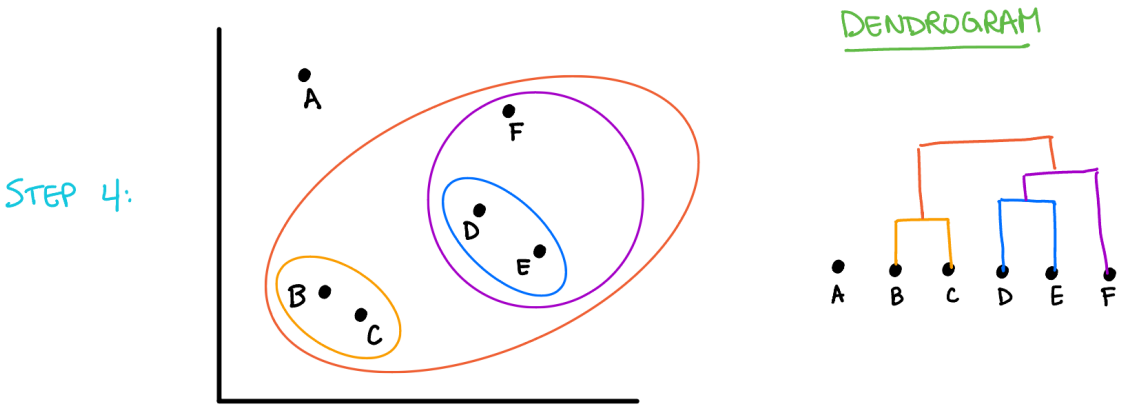
Next Step: Join the two points/clusters that are "closest" together



Hierarchical Clustering

- A **greedy, agglomerative** algorithm
  - At the lowest level, each cluster contains a single observation
  - As we move up the tree, some leaves begin to fuse into branches – these are observations that are most **similar** to each other

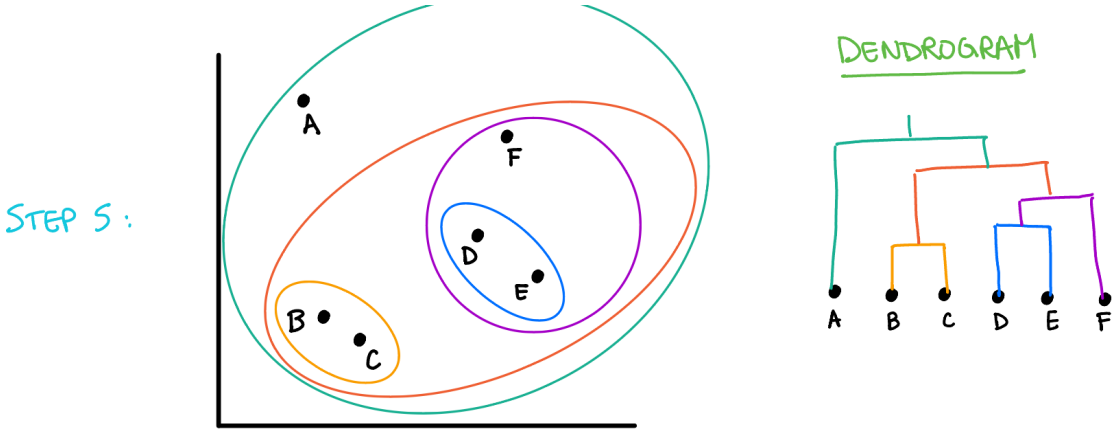
Next Step: Join the two points/clusters that are "closest" together



Hierarchical Clustering

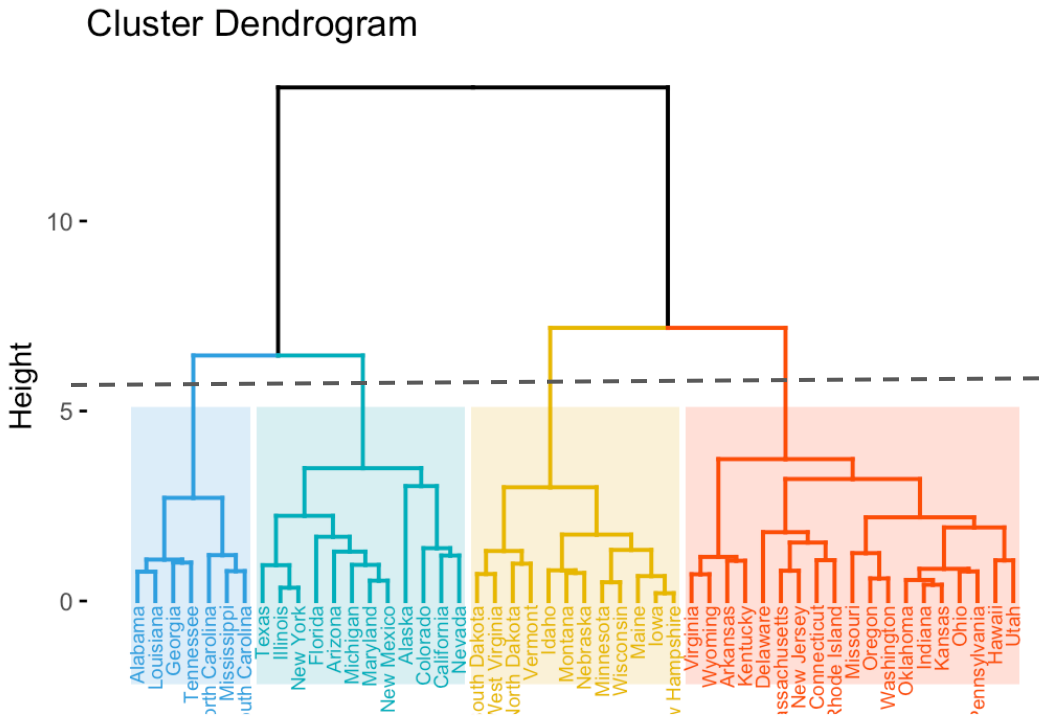
- A **greedy, agglomerative** algorithm
  - At the lowest level, each cluster contains a single observation
  - As we move up the tree, some leaves begin to fuse into branches – these are observations that are most **similar** to each other

Next Step: Join the two points/clusters that are "closest" together



Hierarchical Clustering

- Gives family of nested clusterings, presented as a tree



How to join clusters/observations

1. **Distance metric**: a measure of dissimilarity between **two data points**
    - Examples: l_2 (Euclidean), l_1 (absolute value), any norm, 1 – cor(x, y)

2. **Linkage metric**: a measure of dissimilarity between **two clusters**

How to join clusters/observations

1. **Distance metric**: a measure of dissimilarity between **two data points**
   - Examples: l_2 (Euclidean), l_1 (absolute value), any norm, 1 – cor(x, y)

2. **Linkage metric**: a measure of dissimilarity between **two clusters**

Single Linkage

(min)

Join clusters based on minimum pairwise distance

How to join clusters/observations

1. **Distance metric**: a measure of dissimilarity between **two data points**
   - Examples: l_2 (Euclidean), l_1 (absolute value), any norm, 1 – cor(x, y)

2. **Linkage metric**: a measure of dissimilarity between **two clusters**

Single Linkage
(min)

Join clusters based on minimum pairwise distance

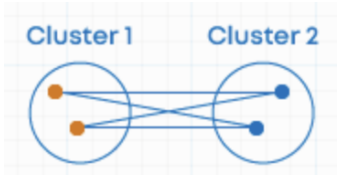
Complete Linkage
(max)

Join clusters based on maximum pairwise distance

How to join clusters/observations

1. **Distance metric**: a measure of dissimilarity between **two data points**
   - Examples: l_2 (Euclidean), l_1 (absolute value), any norm, 1 – cor(x, y)

2. **Linkage metric**: a measure of dissimilarity between **two clusters**

Single Linkage (min)	***Complete Linkage*** (max)	***Average Linkage*** (average)
		
Join clusters based on minimum pairwise distance	Join clusters based on maximum pairwise distance	Join clusters based on average pairwise distance

How to join clusters/observations

1. **Distance metric**: a measure of dissimilarity between **two data points**
   - Examples: l_2 (Euclidean), l_1 (absolute value), any norm, 1 – cor(x, y)

2. **Linkage metric**: a measure of dissimilarity between **two clusters**

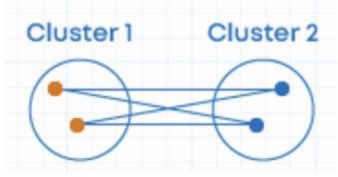
Single Linkage
(min)

Join clusters based on minimum pairwise distance

Complete Linkage
(max)

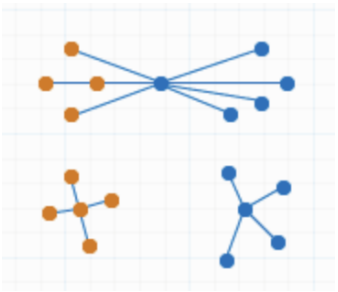
Join clusters based on maximum pairwise distance

Average Linkage
(average)



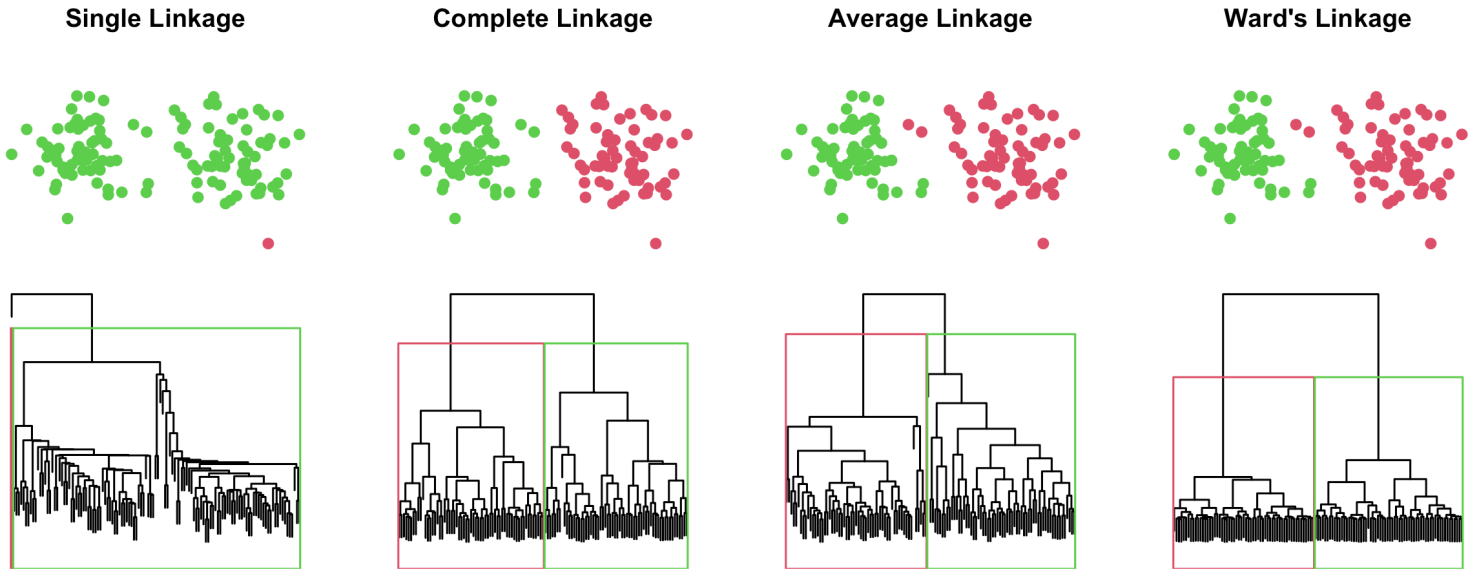
Join clusters based on average pairwise distance

Ward's Linkage
(min variance)

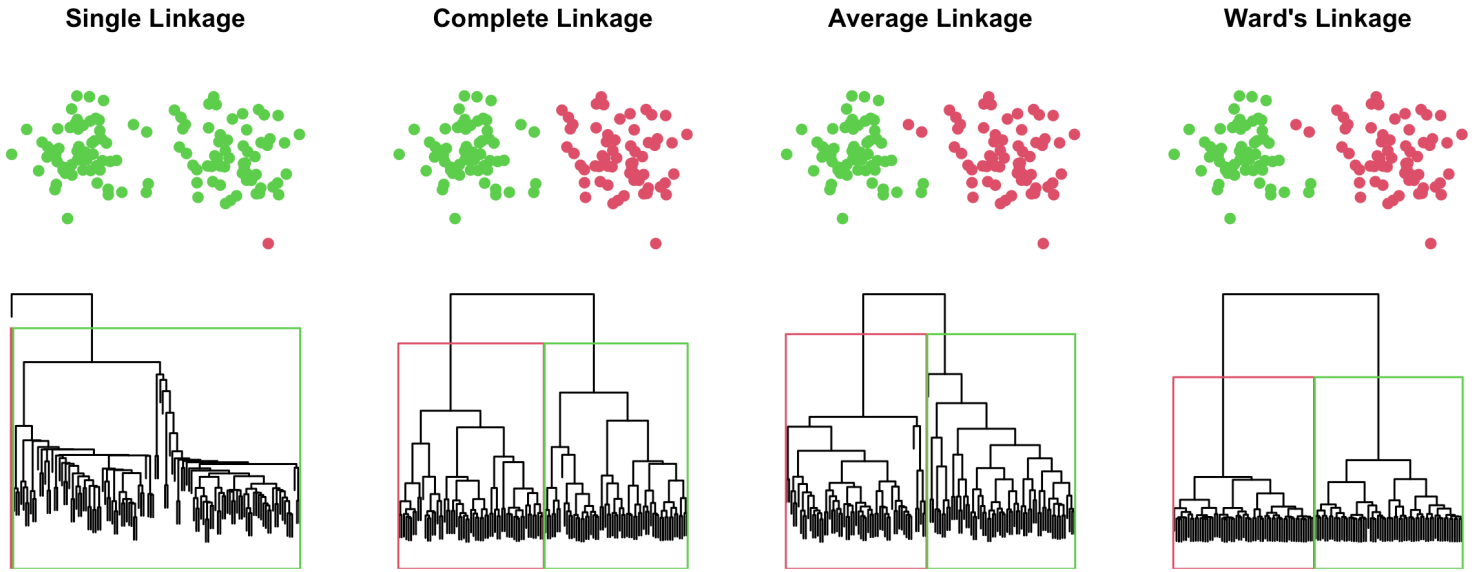


Joins clusters that results in the smallest increase in total within-cluster variance

Linkage Examples



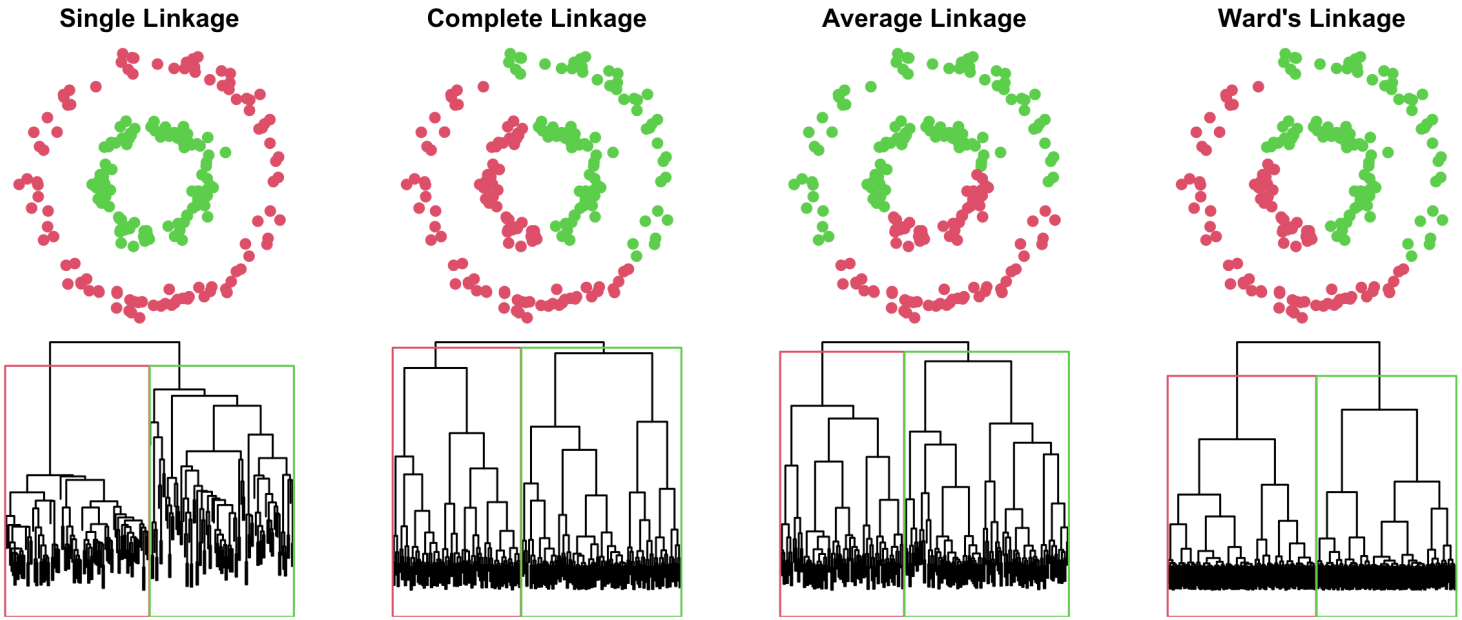
Linkage Examples



Single Linkage (min)

- Very sensitive to outliers or noise
- Extended, trailing clusters in which observations are fused one at a time – chaining

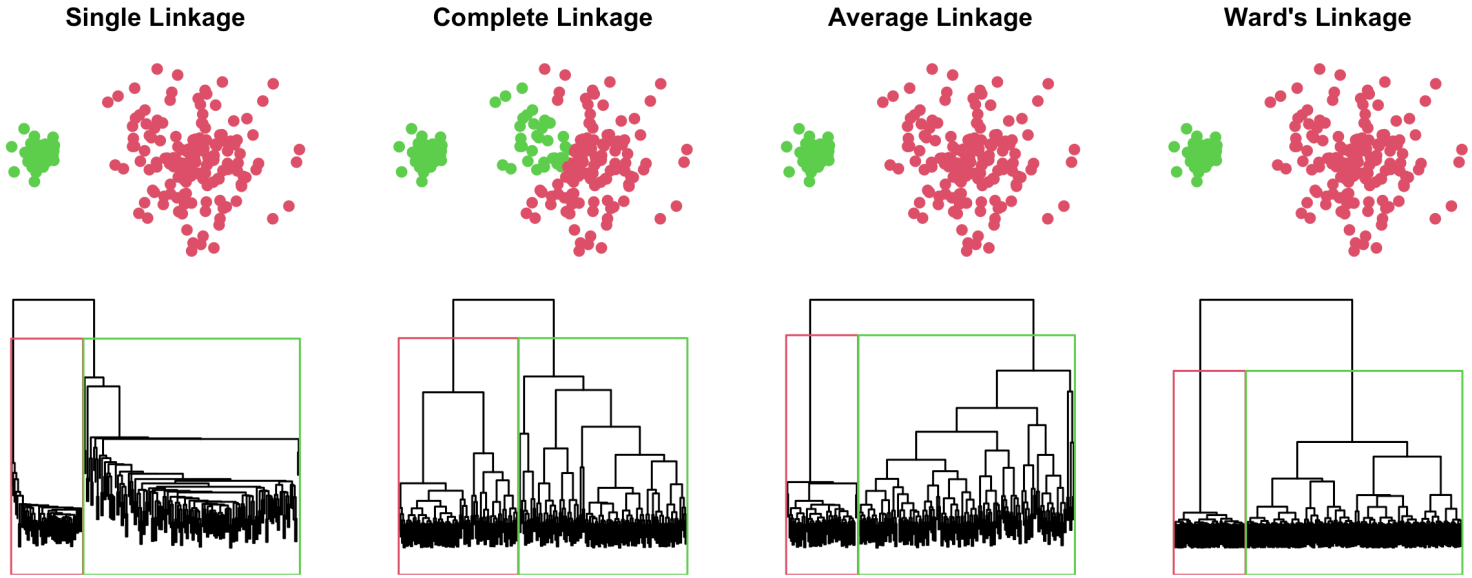
Linkage Examples



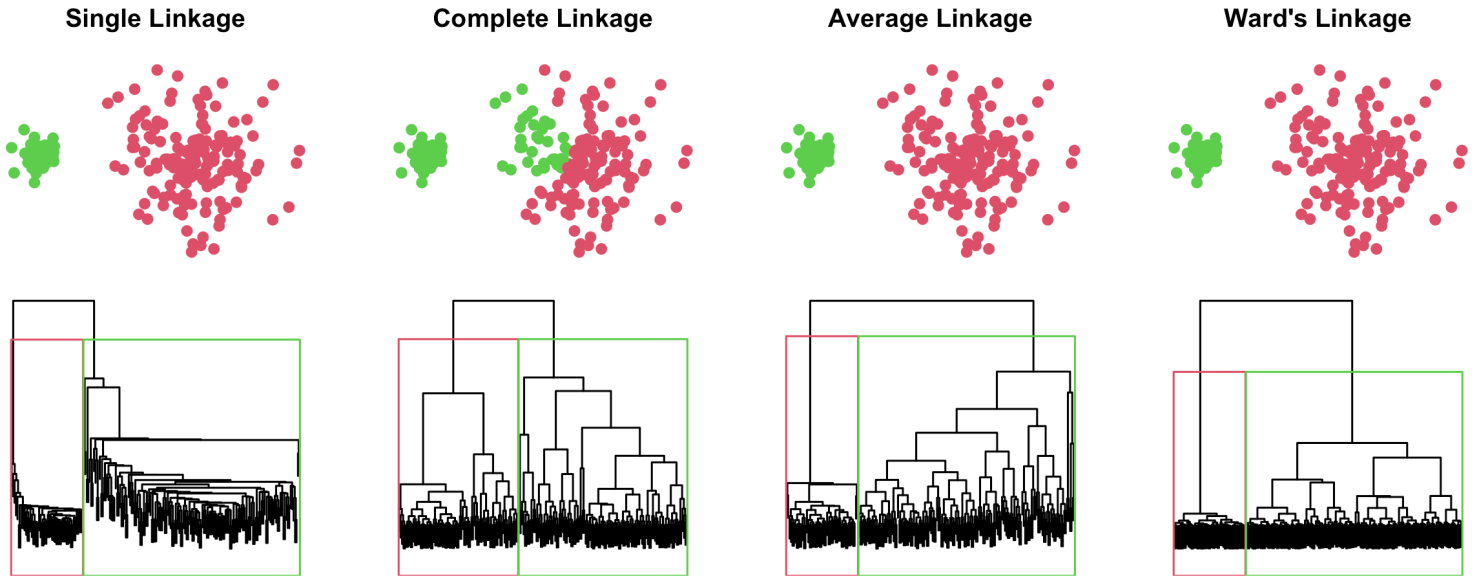
Single Linkage (min)

- Very sensitive to outliers or noise
- Extended, trailing clusters in which observations are fused one at a time – chaining
- Can handle diverse shapes

Linkage Examples



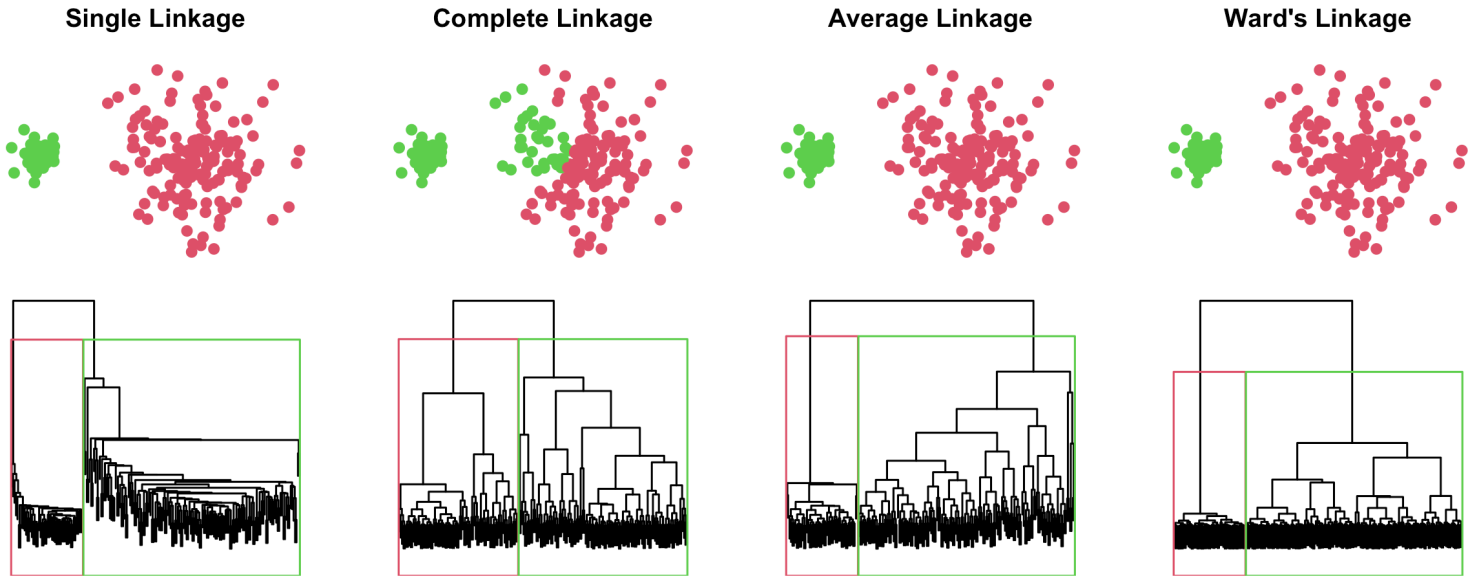
Linkage Examples



Complete Linkage (max)

+ Less sensitive to outliers
+ Often gives cluster with similar sizes

Linkage Examples



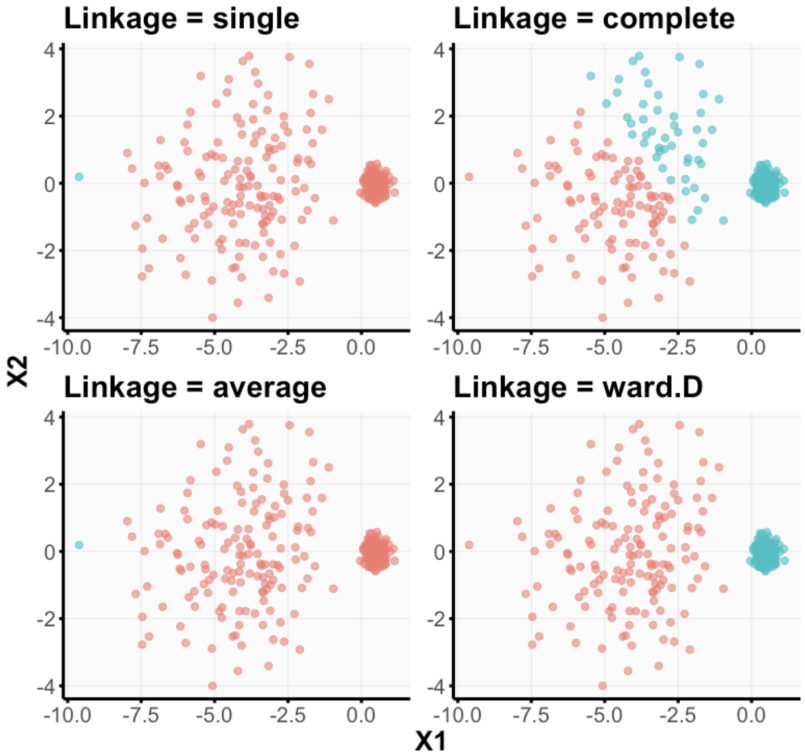
Complete Linkage (max)

- \+ Less sensitive to outliers
- \+ Often gives cluster with similar sizes

Average Linkage

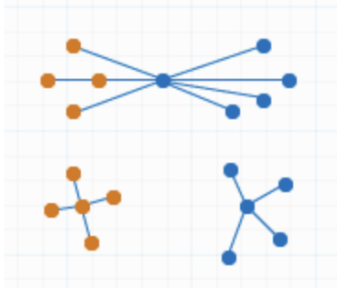
- \+ Compromise between single & complete
- \+ Less sensitive to outliers than single linkage, but not as robust as complete linkage

Linkage Examples



Ward's Linkage (min variance)

- Tends to form compact, spherical clusters
- Works well for normally distributed data
- Computationally expensive



Recap: Clustering Methods

	K-means Clustering	Hierarchical Clustering
Advantages	• Super fast and intuitive • Good when clusters are spherical and linearly separable	• Gives nested family of clusterings • Convenient visualizations with dendrograms
Disadvantages	• Bad when clusters are not spherical or have different variances • Must choose *k* a priori • Local solution; depends on initialization	• Depends *heavily* on linkage (single, complete, average, Ward's linkage) • Greedy search
Shared Disadvantages	• Irrelevant variables are treated as equals with relevant ones • Suffers from "Curse of Dimensionality": computing distances between two points in high dimensions is hard and inaccurate	

Recap: Clustering Methods

	K-means Clustering	Hierarchical Clustering
Advantages	• Super fast and intuitive • Good when clusters are spherical and linearly separable	• Gives nested family of clusterings • Convenient visualizations with dendrograms
Disadvantages	• Bad when clusters are not spherical or have different variances • Must choose *k* a priori • Local solution; depends on initialization	• Depends *heavily* on linkage (single, complete, average, Ward's linkage) • Greedy search
Shared Disadvantages	• Irrelevant variables are treated as equals with relevant ones • Suffers from "Curse of Dimensionality": computing distances between two points in high dimensions is hard and inaccurate	

→ Do dimension reduction first and then clustering using the dimension-reduced data

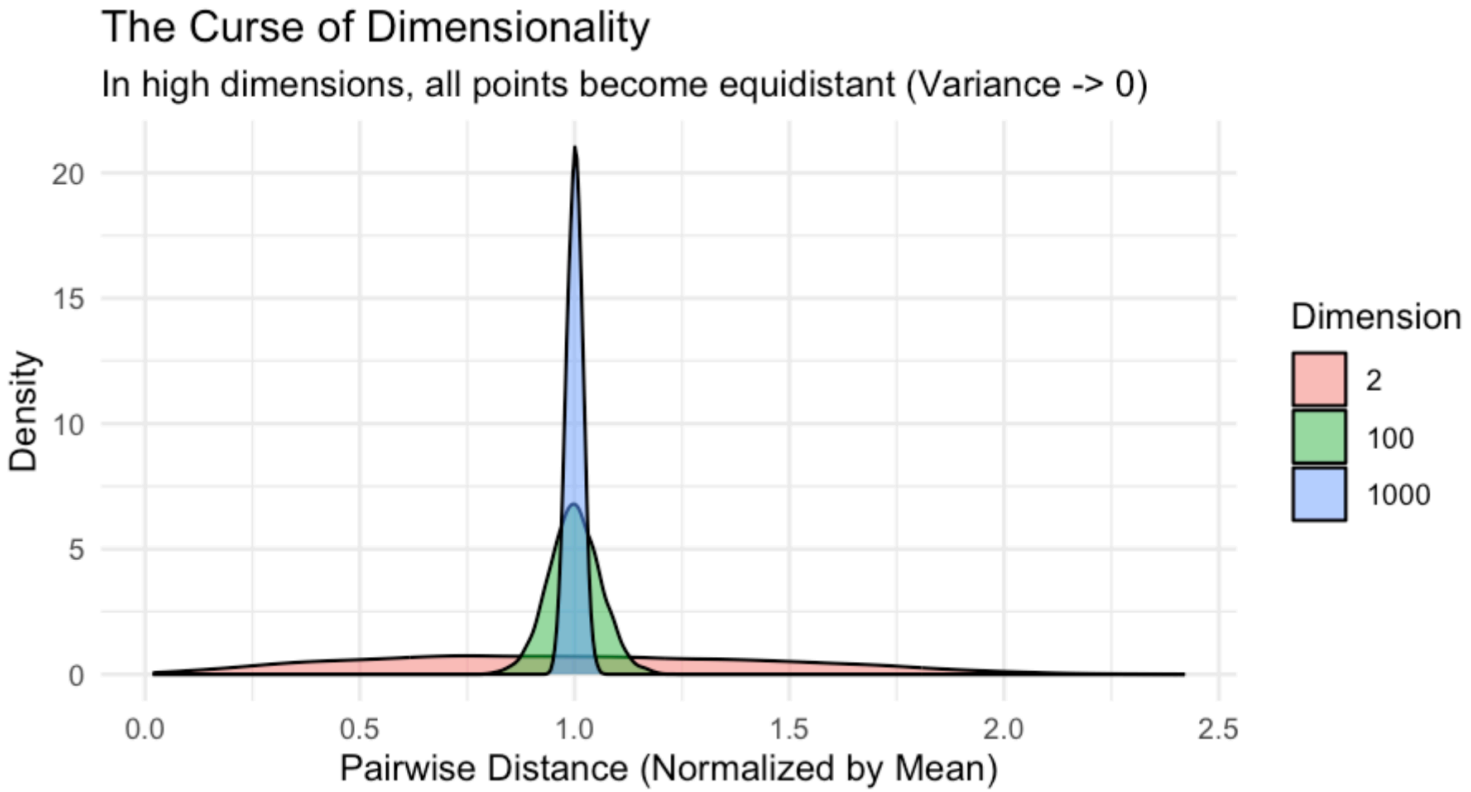
Recap: Clustering Methods

	K-means Clustering	Hierarchical Clustering
Advantages	• Super fast and intuitive • Good when clusters are spherical and linearly separable	• Gives nested family of clusterings • Convenient visualizations with dendrograms
Disadvantages	• Bad when clusters are not spherical or have different variances • Must choose *k* a priori • Local solution; depends on initialization	• Depends *heavily* on linkage (single, complete, average, Ward's linkage) • Greedy search
Shared Disadvantages	• Irrelevant variables are treated as equals with relevant ones • Suffers from "Curse of Dimensionality": computing distances between two points in high dimensions is hard and inaccurate	

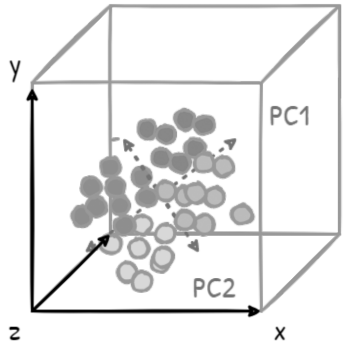
→ Do dimension reduction first and then clustering using the dimension-reduced data

Other clustering methods: spectral clustering, mixture models, DBSCAN, K-mediods

Curse of Dimensionality

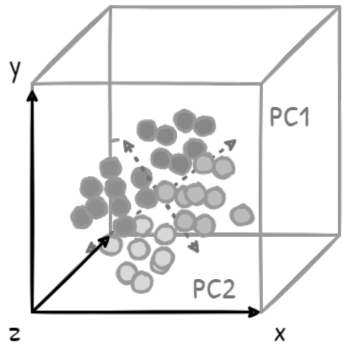


A Common Unsupervised Learning Pipeline



Original Data
(often “high-dimensional”)

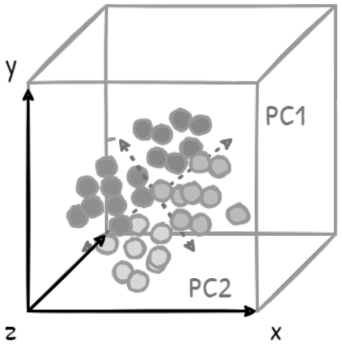
A Common Unsupervised Learning Pipeline



Original Data
(often “high-dimensional”)

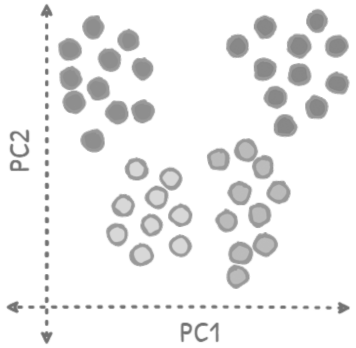
Step 1:
Dimension
reduction

A Common Unsupervised Learning Pipeline



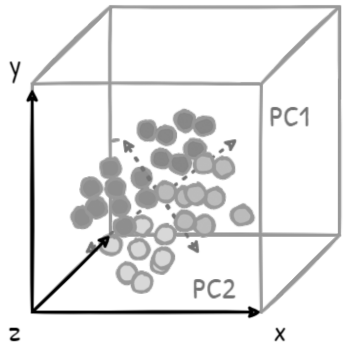
Original Data
(often “high-dimensional”)

Step 1:
Dimension
reduction



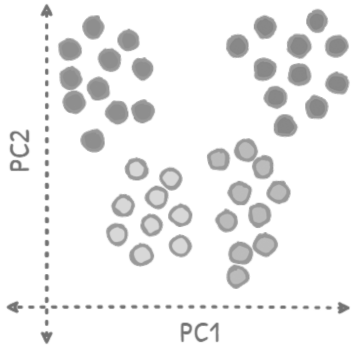
“Lower-dimensional”
representation

A Common Unsupervised Learning Pipeline



Original Data
(often “high-dimensional”)

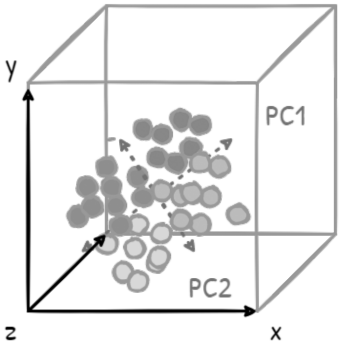
Step 1:
Dimension reduction



“Lower-dimensional” representation

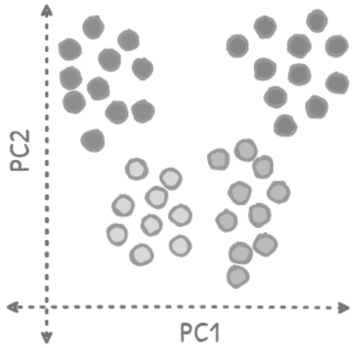
Step 2:
Clustering

A Common Unsupervised Learning Pipeline



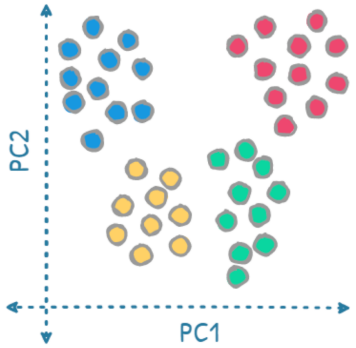
Original Data
(often “high-dimensional”)

Step 1:
Dimension
reduction

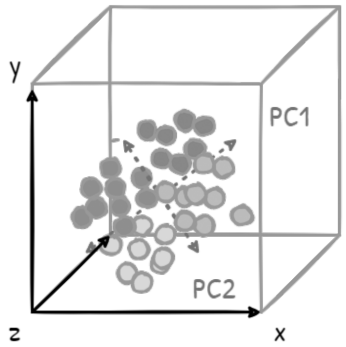


“Lower-dimensional”
representation

Step 2:
Clustering

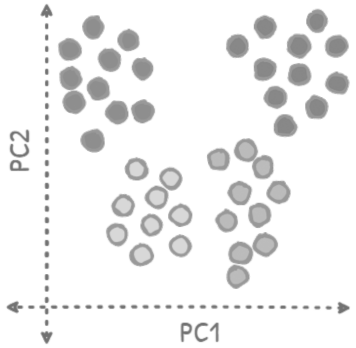


A Common Unsupervised Learning Pipeline



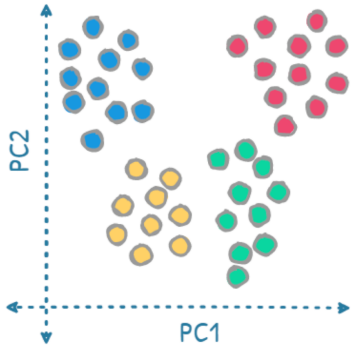
Original Data
(often “high-dimensional”)

Step 1:
Dimension
reduction



“Lower-dimensional”
representation

Step 2:
Clustering



* Focus of Lab 2

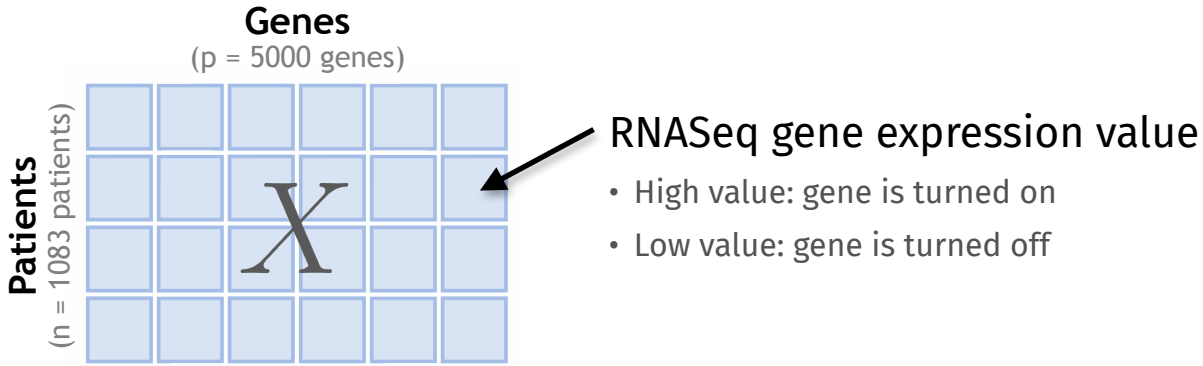
Lab 2 Introduction

Breast Cancer Subtyping

- Breast cancer is a very heterogeneous disease
  - Can be sub-divided into smaller subtypes (e.g., HER2+, HER2-, etc)
  - Each subtype has a different severity/prognosis as well as different recommended treatment plans
- How were these subtypes originally discovered?
  - Dimension reduction + clustering

Breast Cancer Subtyping

- **Data:**



- **Your goal:** conduct your own dimension reduction + clustering analysis to re-discover different breast cancer subtypes (i.e., cluster the patients into groups)
  - You will also see an example of how data cleaning can substantially impact your results
- **Output:** your two best guesses for the # of clusters and the cluster membership labels
  - I have the true subtype labels but will not release them until after the lab
  - Best guess will earn bonus points

Summary

Recap + Next Time

- **Dimension Reduction:** aims to find a lower-dimensional representation of the data which preserves as much of the original information as possible
  - Includes methods such as PCA, tSNE, UMAP, and autoencoders
- **Clustering:** aims to identify groups/clusters of observations that are "similar"
  - Includes methods such as K-means and hierarchical clustering
  - How do you choose the number of clusters *K*? Next week

Additional Resources:

- Course website under "5 Unsupervised Learning" (includes R/Python code for implementing methods)
- Elements of Statistical Learning Textbook Chapter 14

Next Time:

- Dimension reduction + clustering in practice