CPSC 4300/6300: Applied Data Science

2024

Unsupervised Learning - Clustering

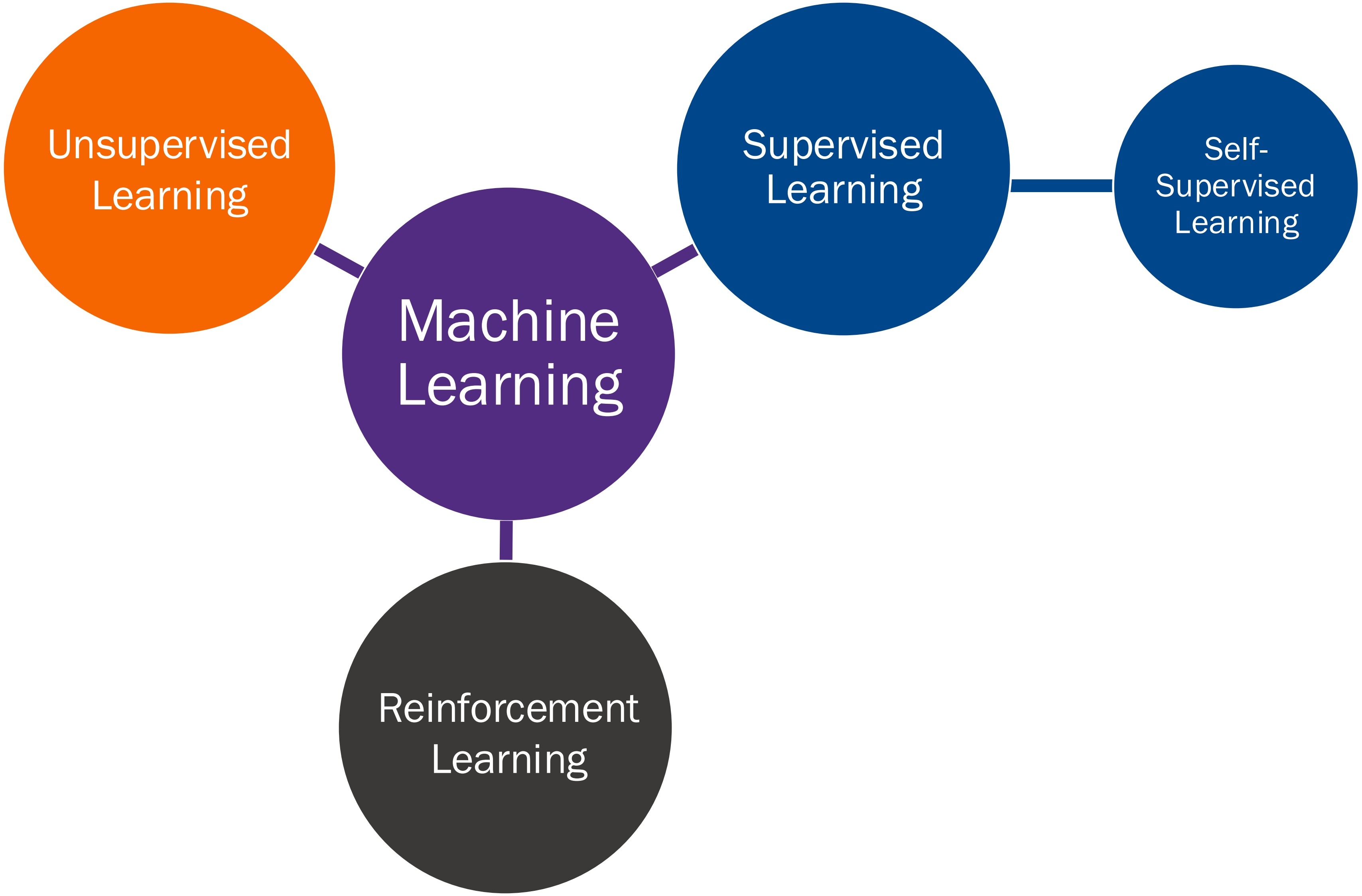
Dr. Aaron J Masino

Associate Professor, School of Computing

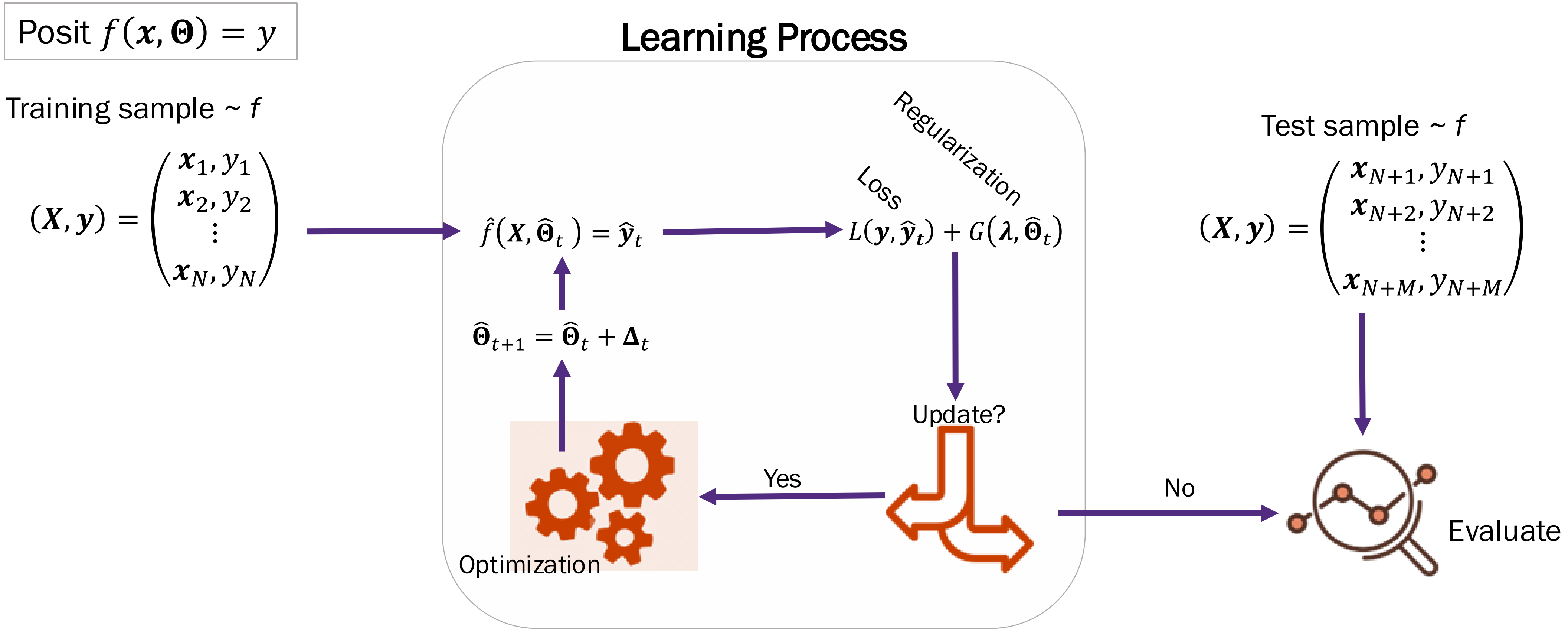
College of
ENGINEERING, COMPUTING
AND APPLIED SCIENCES

Review machine learning paradigms

Learning Paradigms



Supervised Learning



Posit $f(\boldsymbol{x}, \boldsymbol{\Theta}) = y$

Training sample ~ *f*

$$(\boldsymbol{X}, \boldsymbol{y}) = \begin{pmatrix} \boldsymbol{x}_1, y_1 \\ \boldsymbol{x}_2, y_2 \\ \vdots \\ \boldsymbol{x}_N, y_N \end{pmatrix}$$

Learning Process

$$\hat{f}(\boldsymbol{X}, \widehat{\boldsymbol{\Theta}}_t) = \widehat{\boldsymbol{y}}_t$$

Loss

Regularization

$$L(\boldsymbol{y}, \widehat{\boldsymbol{y}}_t) + G(\boldsymbol{\lambda}, \widehat{\boldsymbol{\Theta}}_t)$$

Update?

Yes

$$\widehat{\boldsymbol{\Theta}}_{t+1} = \widehat{\boldsymbol{\Theta}}_t + \boldsymbol{\Delta}_t$$

Optimization

No

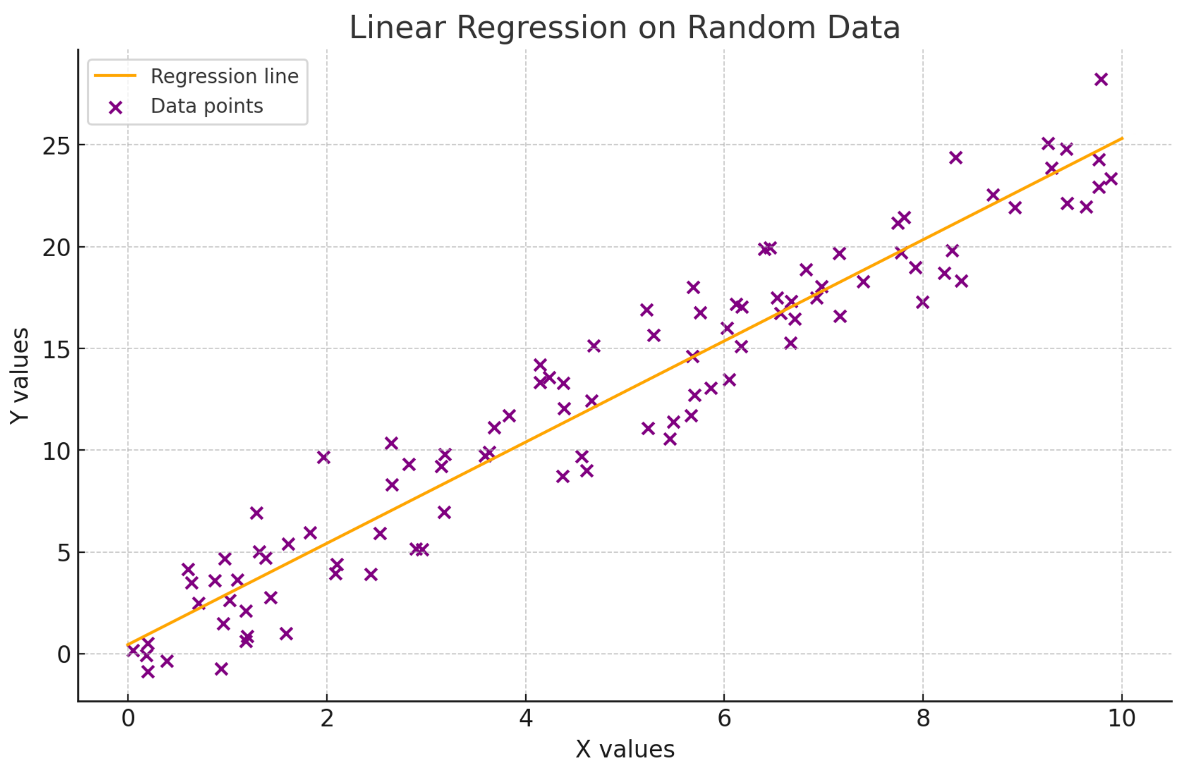
Test sample ~ *f*

$$(\boldsymbol{X}, \boldsymbol{y}) = \begin{pmatrix} \boldsymbol{x}_{N+1}, y_{N+1} \\ \boldsymbol{x}_{N+2}, y_{N+2} \\ \vdots \\ \boldsymbol{x}_{N+M}, y_{N+M} \end{pmatrix}$$

Evaluate

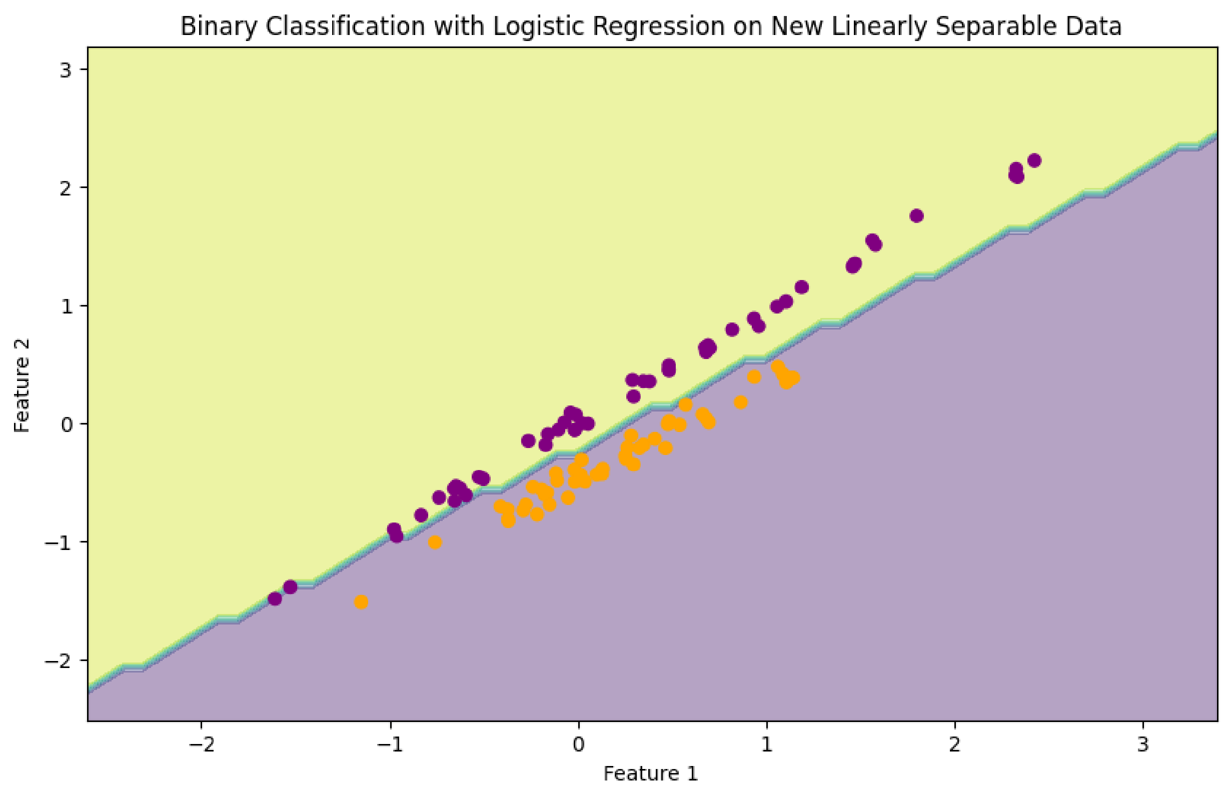
Regression

- Construct a function, f, to estimate the quantitative value, Y, from input X
- Methods we've covered
  - KNN regression
  - Linear regression
- There are others
  - Tree-based regression
  - Survival analysis
  - LSTMs



Classification

- Construct a function, f, to estimate the qualitative (class membership) value, Y, from input X
- We'll cover this the next few weeks

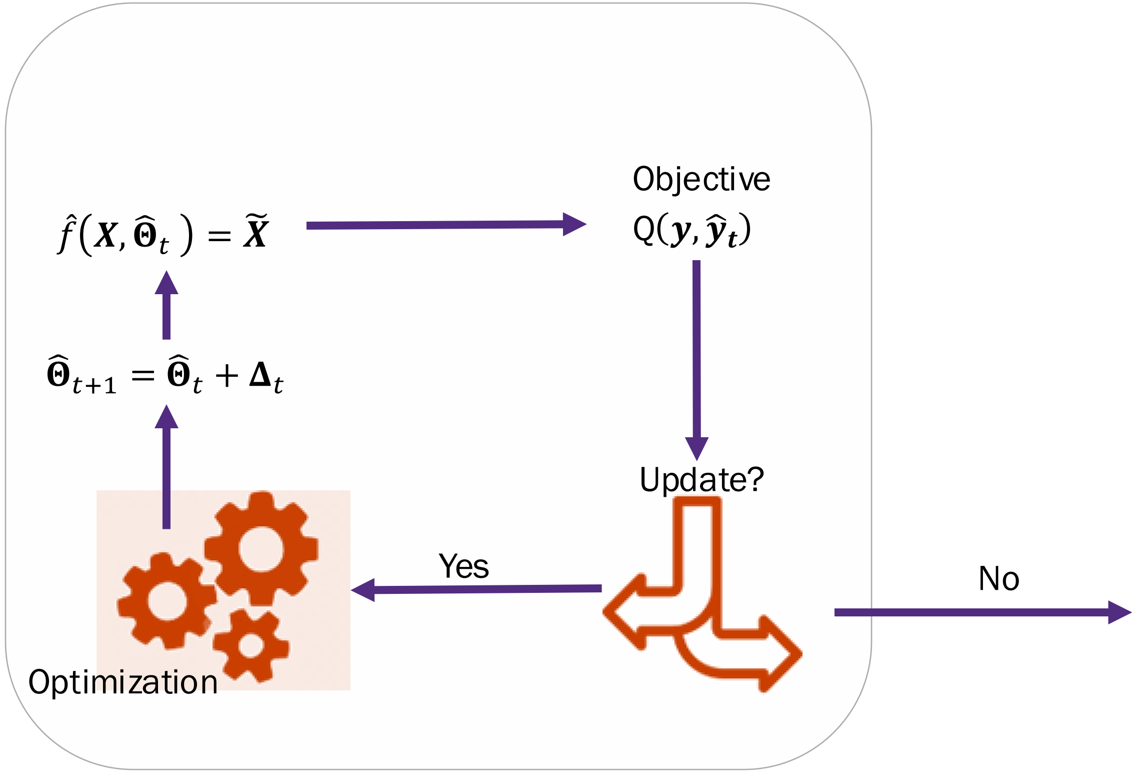

Binary Classification with Logistic Regression on New Linearly Separable Data

Unsupervised Learning

Posit $f(\boldsymbol{X}, \boldsymbol{\Theta}) = \widetilde{\boldsymbol{X}}$
subject to constraints $\boldsymbol{C}$

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_N \end{bmatrix}$$

Learning Process



$\hat{f}(\boldsymbol{X}, \widehat{\boldsymbol{\Theta}}_t) = \widetilde{\boldsymbol{X}}$

Objective
$Q(\boldsymbol{y}, \widehat{\boldsymbol{y}}_t)$

Update?

Yes

No

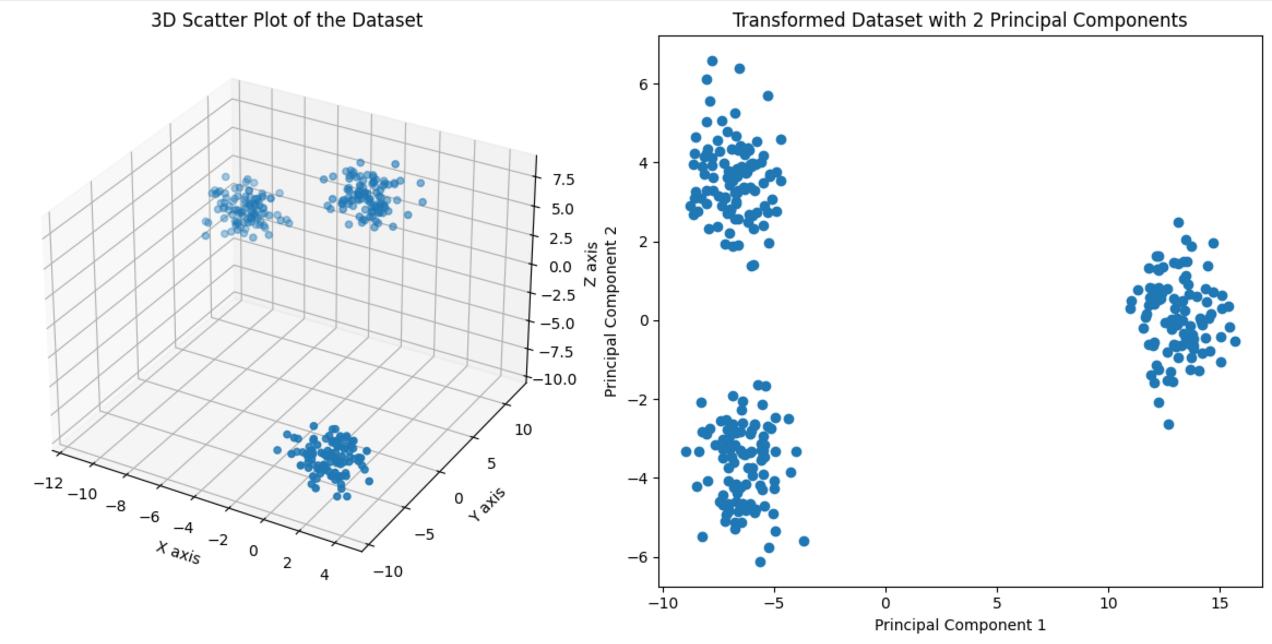
Optimization

$\widehat{\boldsymbol{\Theta}}_{t+1} = \widehat{\boldsymbol{\Theta}}_t + \boldsymbol{\Delta}_t$

$$\widetilde{\boldsymbol{X}} = \left[f \begin{pmatrix} \boldsymbol{x}_1 \\ \boldsymbol{x}_2 \\ \vdots \\ \boldsymbol{x}_N \end{pmatrix} \right]$$

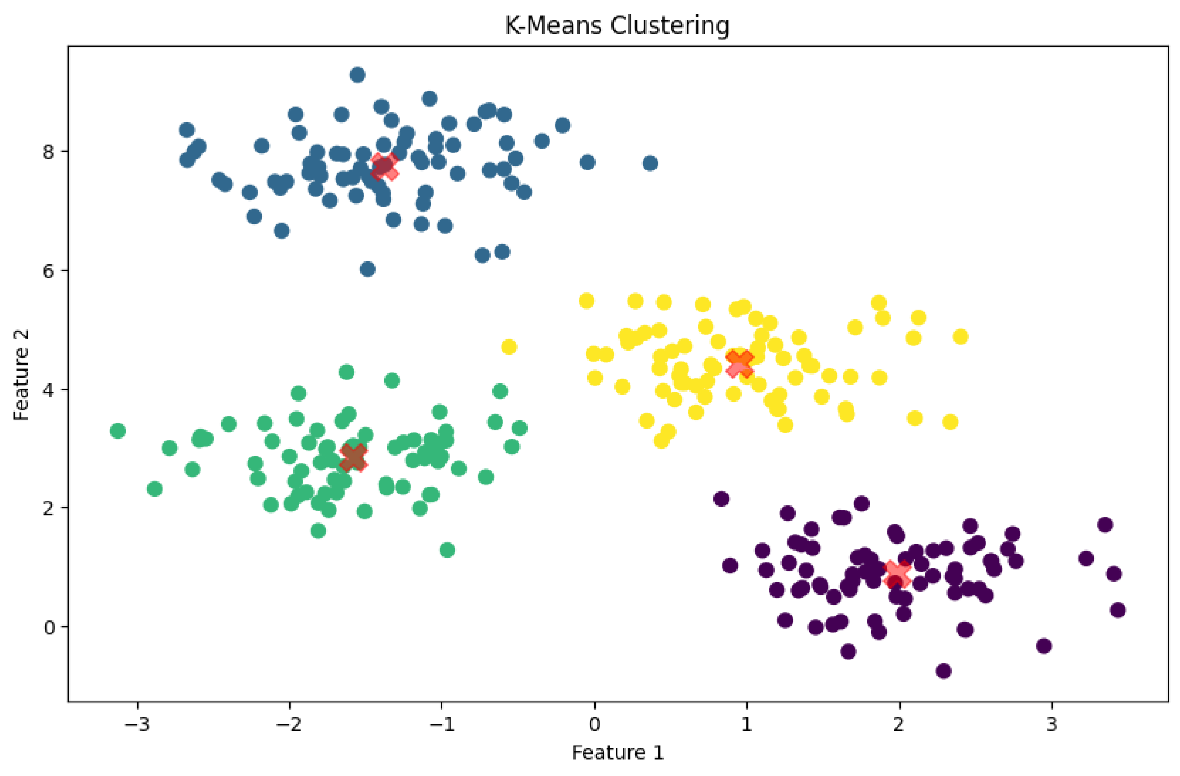
Dimensionality reduction

- Construct a function, f, to transform input X to a lower dimensional representations
- We’ll cover this later in the course

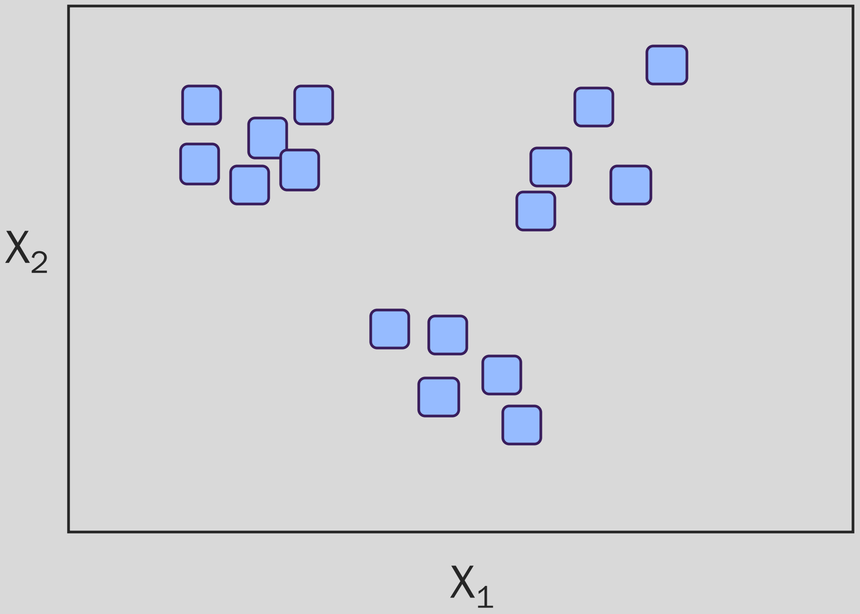


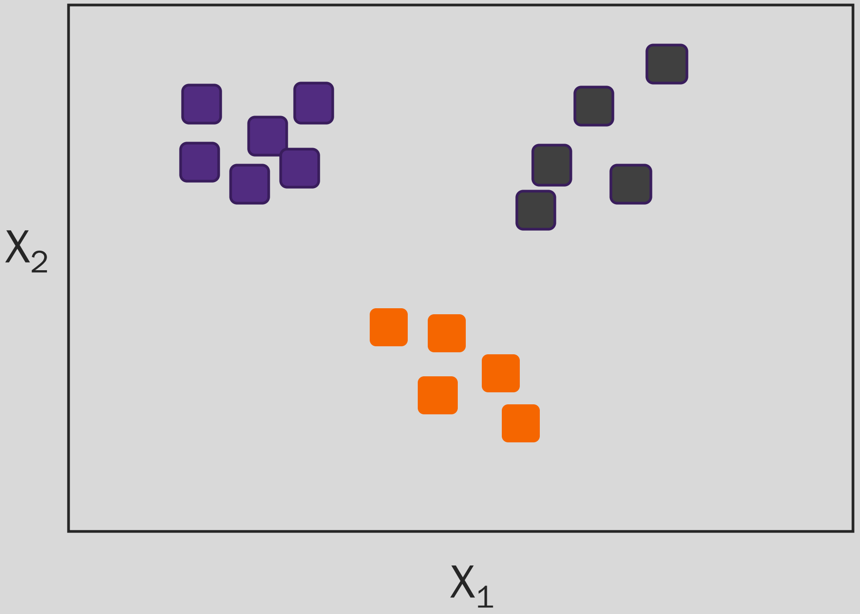
Clustering

- Construct a function, f, to identify structure (e.g., clusters) among samples with input X
- We'll discus this today



Clustering



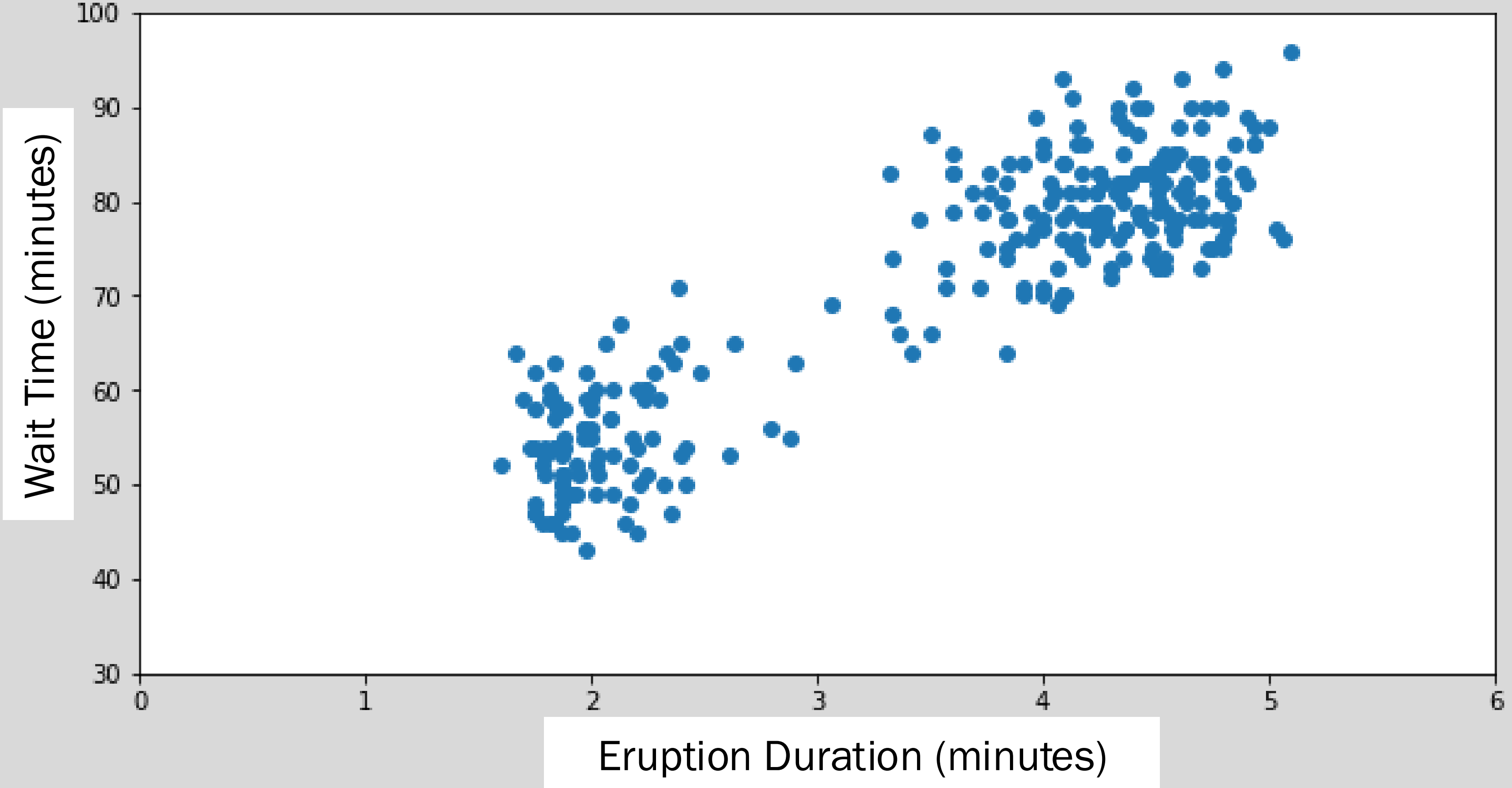


What is clustering?

- Goal: Find distinct groups in the data where the *similarity* between individuals in the same group is much higher than the similarity between individuals in different groups
- Why?
  - Identify meaningful sub-groups with distinctive characteristics
  - Can be used as a precursor to classifier model development
- Numerous methods (see scikit-learn)– we'll discuss three

Simple Example: Eruptions of Old Faithful

Data were collected on **time between eruptions and the duration of eruptions** for the Old Faithful geyser at Yellowstone National Park



Visually, it seems there may be two distinct clusters. How do we identify these analytically?

Outline

- Inter-observational Distances
- Partition-based Clustering
- Hierarchical Clustering
- Diagnostics and Optimizing Number of Clusters
- Density-based Clustering

Inter-observational Distances

- Need to assign a distance metric to assess distance between samples
- Pairwise distance calculations can influence the shape of your clusters
- Observations to be clustered can be non-standard (e.g., images, audio signals, etc.), so the process of computing distances first followed by clustering is a typical approach.

Inter-observational Distances

- Two common distance metrics

$$d_{Euc}(x_i, x_k) = \sqrt{\sum_{j=1}^{p} \left(X_{ij} - X_{kj}\right)^2}$$ Eucidean Distance (L_2)

$$d_{Man}(x_i, x_k) = \sum_{j=1}^{p} |x_{ij} - x_{kj}|$$ Manhattan Distance (L_1)

Recall:
n indicates the number of samples
p indicates the number of samples
i index refers to sample where $i \ni [1, n]$ (in Python $i \ni [0, n-1]$
j index refers to feature where $j \ni [1, p]$ (in Python $i \ni [0, p-1]$

Inter-observational Distances

- Other (correlation-based) distance metrics

$$d_{Pearson}(x_i, x_k) = 1 - \frac{\sum_{j=1}^{p}(X_{ij} - \overline{x_i})(X_{kj} - \bar{x}_k)}{\sqrt{\sum_{j=1}^{p}\left(X_{ij} - \bar{x}_i\right)^2 \sum_{j=1}^{p}\left(X_{kj} - \bar{x}_k\right)^2}}$$

$$d_{Spearman}(x_i, x_k) = 1 - \frac{\sum_{j=1}^{p}(W_{ij} - \overline{w_i})(W_{kj} - \bar{w}_k)}{\sqrt{\sum_{j=1}^{p}\left(W_{ij} - \bar{w}_i\right)^2 \sum_{j=1}^{p}\left(W_{kj} - \bar{w}_k\right)^2}}$$

where W_{ij} are the ranks of X_{ij} for feature j, and $\bar{w}_i$ is the average of the ranks for observation i. Spearman correlation is used when outliers might be a concern

Recall:
n indicates the number of samples
p indicates the number of samples
i index refers to sample where $i \ni [1, n]$ (in Python $i \ni [0, n-1]$
j index refers to feature where $j \ni [1, p]$ (in Python $i \ni [0, p-1]$

Distances and Scaling

Consider a data set that looks like the following:

X1	X2	X3	X4
498625	0.53	0.73	1.2
88635	0.12	0.13	2.3
623617	0.43	0.47	2.7
…	…	…	…

The distance between any two observations is basically determined by feature X1. This is an undesired consequence of variables having different scales.

Usually, we want to ensure each variable has equal contribution to the clustering algorithm, and therefore the distance computation.

Distances and Scaling

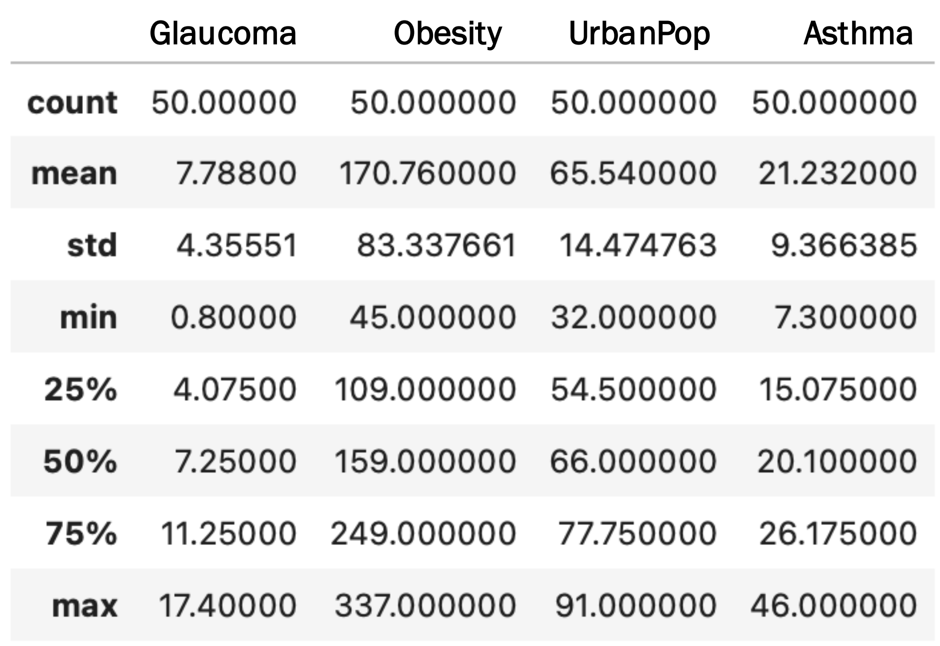
To address this problem, standardize the variables prior to computing distances

$$\frac{X_{ij} - center(x_j)}{scale(x_j)}$$

where $center(x_j)$can be the sample mean or median of the j^{th} variable, and $scale(x_j)$ can be the sample standard deviation or mean absolute deviation of the j^{th} variable.

Example: Diagnoses by State

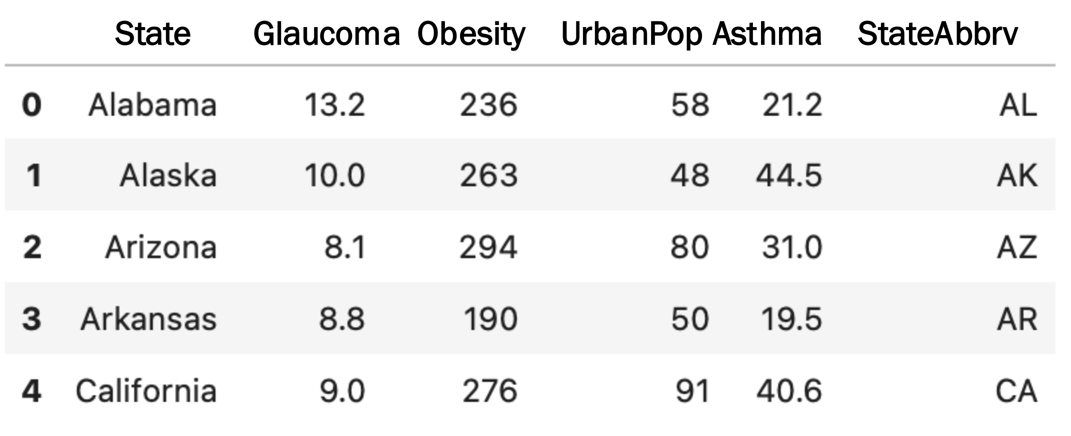
This fictious data set indicates diagnosis counts per 100,000 residents for asthma, obesity, glaucoma in each of the 50 US states in one year. Also given is the percentage of population living in urban areas.



	Glaucoma	Obesity	UrbanPop	Asthma
count	50.00000	50.000000	50.000000	50.000000
mean	7.78800	170.760000	65.540000	21.232000
std	4.35551	83.337661	14.474763	9.366385
min	0.80000	45.000000	32.000000	7.300000
25%	4.07500	109.000000	54.500000	15.075000
50%	7.25000	159.000000	66.000000	20.100000
75%	11.25000	249.000000	77.750000	26.175000
max	17.40000	337.000000	91.000000	46.000000

Example: Diagnoses by State

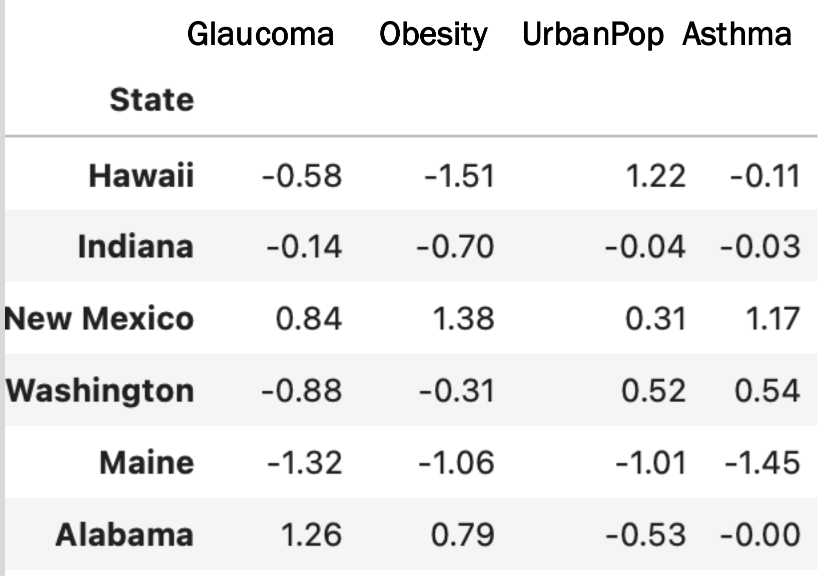
First few randomly chosen observations (NO SCALING)



	State	Glaucoma	Obesity	UrbanPop	Asthma	StateAbbrv
0	Alabama	13.2	236	58	21.2	AL
1	Alaska	10.0	263	48	44.5	AK
2	Arizona	8.1	294	80	31.0	AZ
3	Arkansas	8.8	190	50	19.5	AR
4	California	9.0	276	91	40.6	CA

Example: Diagnoses by State

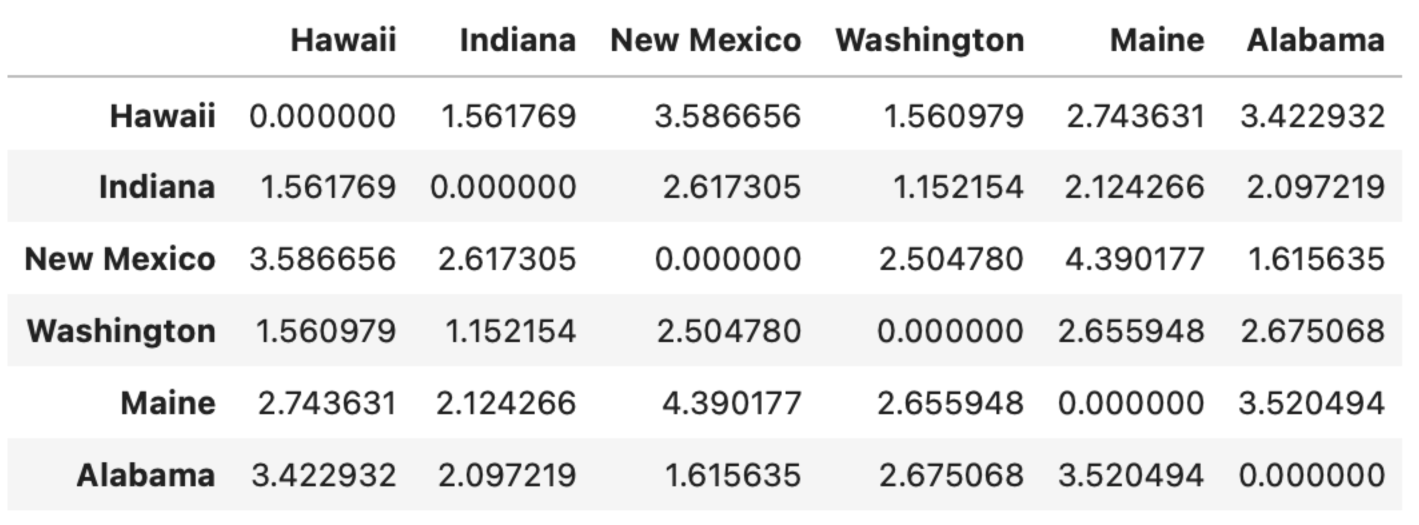
First few observations **rescaled** (subtract mean, divide by standard deviation)



	Glaucoma	Obesity	UrbanPop	Asthma
State				
Hawaii	-0.58	-1.51	1.22	-0.11
Indiana	-0.14	-0.70	-0.04	-0.03
New Mexico	0.84	1.38	0.31	1.17
Washington	-0.88	-0.31	0.52	0.54
Maine	-1.32	-1.06	-1.01	-1.45
Alabama	1.26	0.79	-0.53	-0.00

Example: Diagnoses by State

Euclidean distances calculated on scaled observations



	Hawaii	Indiana	New Mexico	Washington	Maine	Alabama
Hawaii	0.000000	1.561769	3.586656	1.560979	2.743631	3.422932
Indiana	1.561769	0.000000	2.617305	1.152154	2.124266	2.097219
New Mexico	3.586656	2.617305	0.000000	2.504780	4.390177	1.615635
Washington	1.560979	1.152154	2.504780	0.000000	2.655948	2.675068
Maine	2.743631	2.124266	4.390177	2.655948	0.000000	3.520494
Alabama	3.422932	2.097219	1.615635	2.675068	3.520494	0.000000

Outline

- Inter-observational Distances
- Partition-based Clustering
- Hierarchical Clustering
- Diagnostics and Optimizing Number of Clusters
- Density-based Clustering

Partition-based Clustering

Basic Idea: Specify the number of clusters into which the data will be partitioned, and then perform computation to group data so that

1. observations within clusters are similar (low distances), and
2. observations in different clusters are dissimilar (high distances).

We will address the optimal number of clusters separately.

K-means clustering

- Given n samples partition the samples into K distinct groups, $C_1, \cdots, C_k$
- Each sample is assigned to exactly one group (cluster)
- Conceptual approach is to assign cluster membership that minimizes

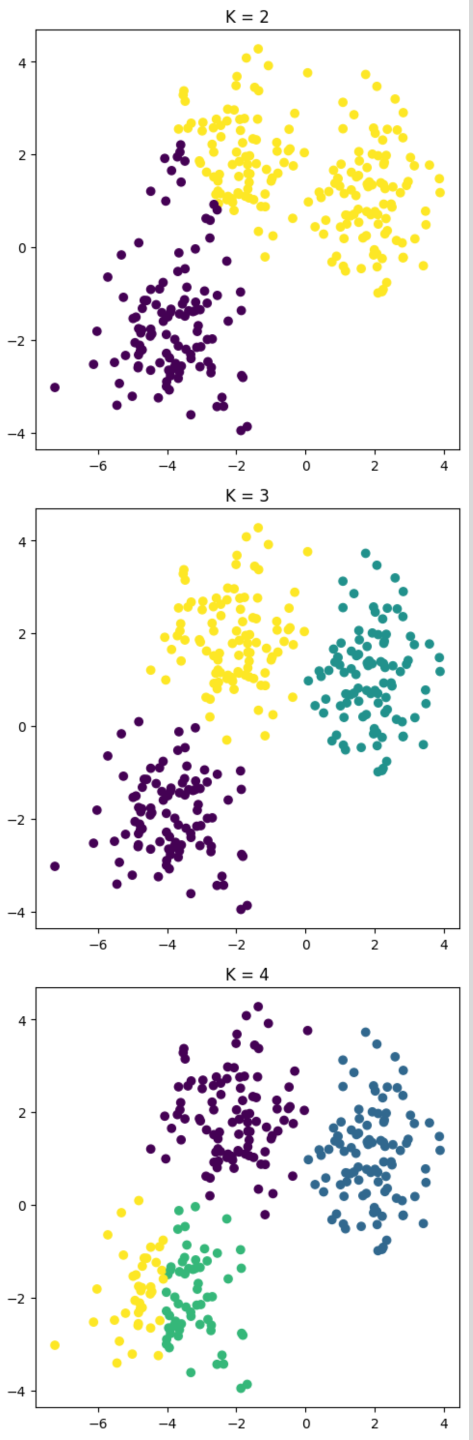
$$\sum_{k=1}^{K} W(C_k)$$

where $W(C_k)$ is a measure of *intra-cluster variation*

- Letting $W(C_k)$ be the *squared Euclidean distance,* we seek $C_1, \cdots, C_k$ that minimize

$$\sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} \left(x_{i,j} - x_{i',j}\right)^2$$

- Other metrics: Hamming (binary vectors), Manhattan (integer vectors), Gower’s (combined binary, numerical, categorical)



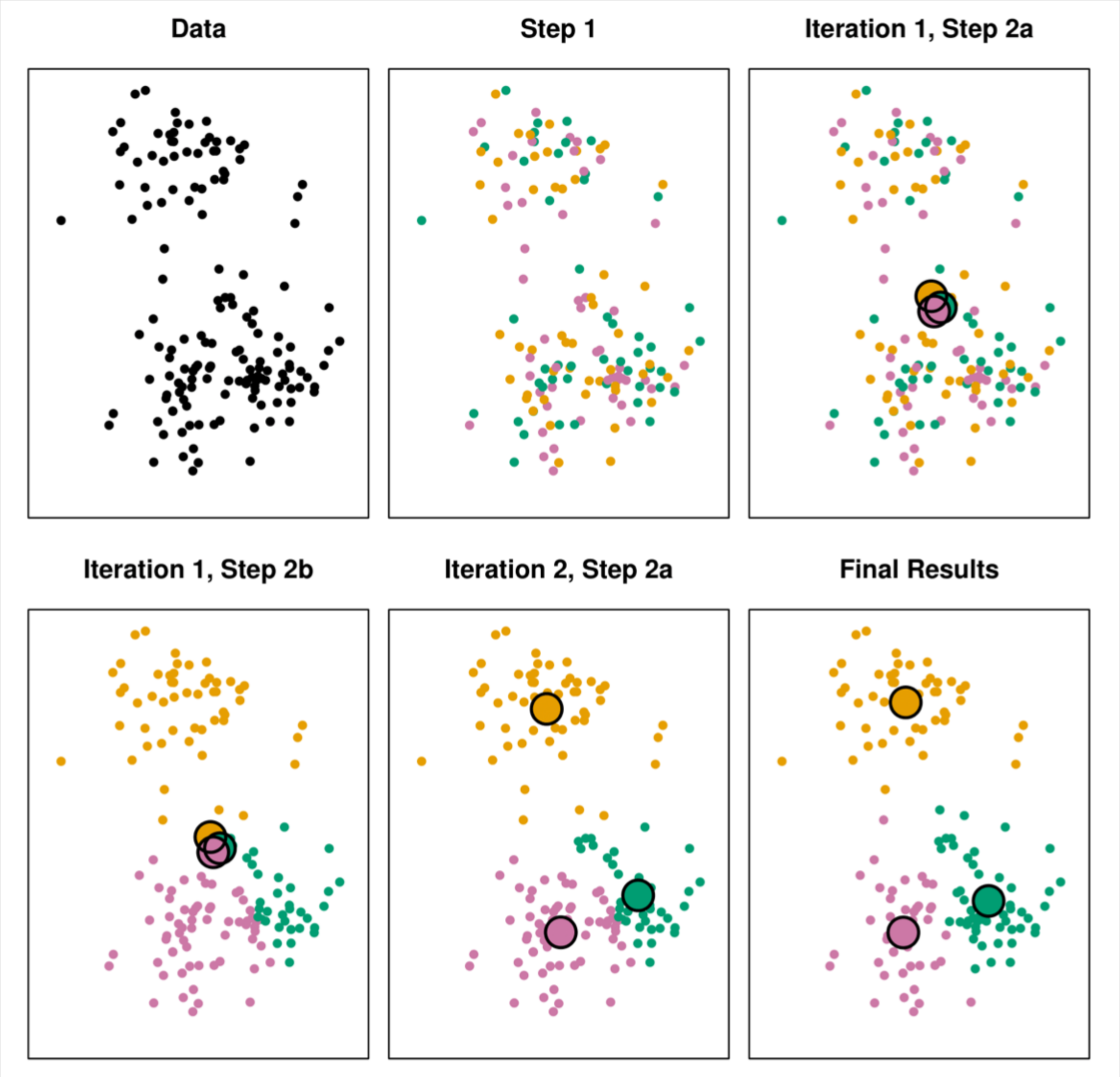
K-means clustering

- There are K^n possible partitions of n samples into K clusters
- Instead, randomly select clusters and iteratively update to find a local solution
- Result depends strongly on initial random cluster selection
- Apply multiple repetitions to evaluate robustness of solution

K-means Algorithm

1. Randomly assign a number in $[1, \cdots, K]$ to each sample. These represent the *initial cluster assignments*.
2. Do while the cluster assignments continue to change:
   a) For each cluster, compute the cluster *centroid* (vector of means of each feature for samples in the cluster)
   b) Reassign each sample to the cluster with the smallest distance (e.g., Euclidean) between the cluster centroid and the sample

K-Means Clustering Algorithm



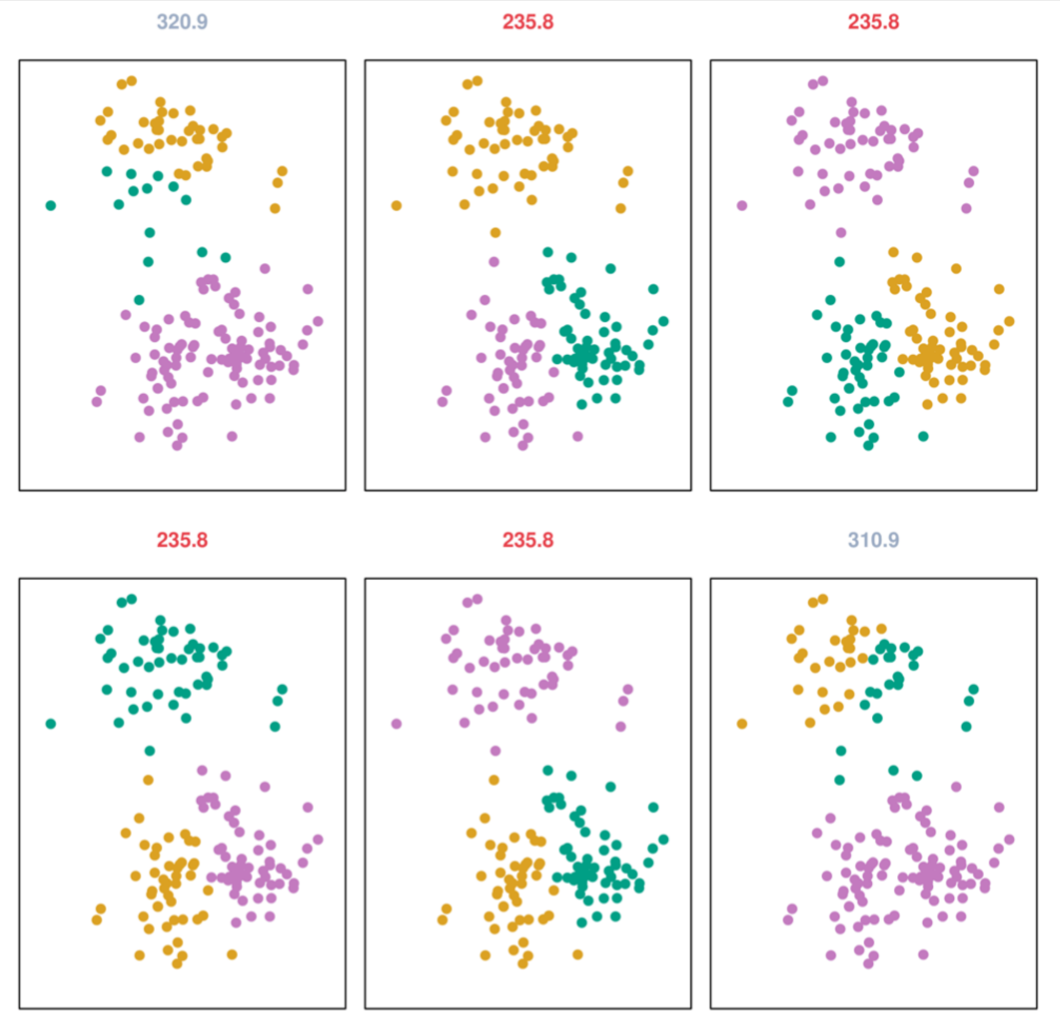
K-Means Clustering Algorithm

- Requires you to select K in advance.
- The algorithm is *locally optimal*, not globally optimal.
  - i.e., you can get different cluster results depending on the initial cluster assignments.

- Potential Solutions
  - Try different values of K and compare the results.
  - Try different initial cluster assignments in parallel and chose the one with the best within-cluster sum of squared deviations.

K-Means Clustering Algorithm

- Six results from different initializations



Clustering Diagnostics: Silhouette Plot

Once a clustering has been determined, let

- a_i = average dissimilarity between observation i and the other points in the cluster to which i belongs
- b_i = average dissimilarity between observation i and the other points in the *next closest* cluster to observation i.
- Dissimilarity is often taken as the inverse of distance

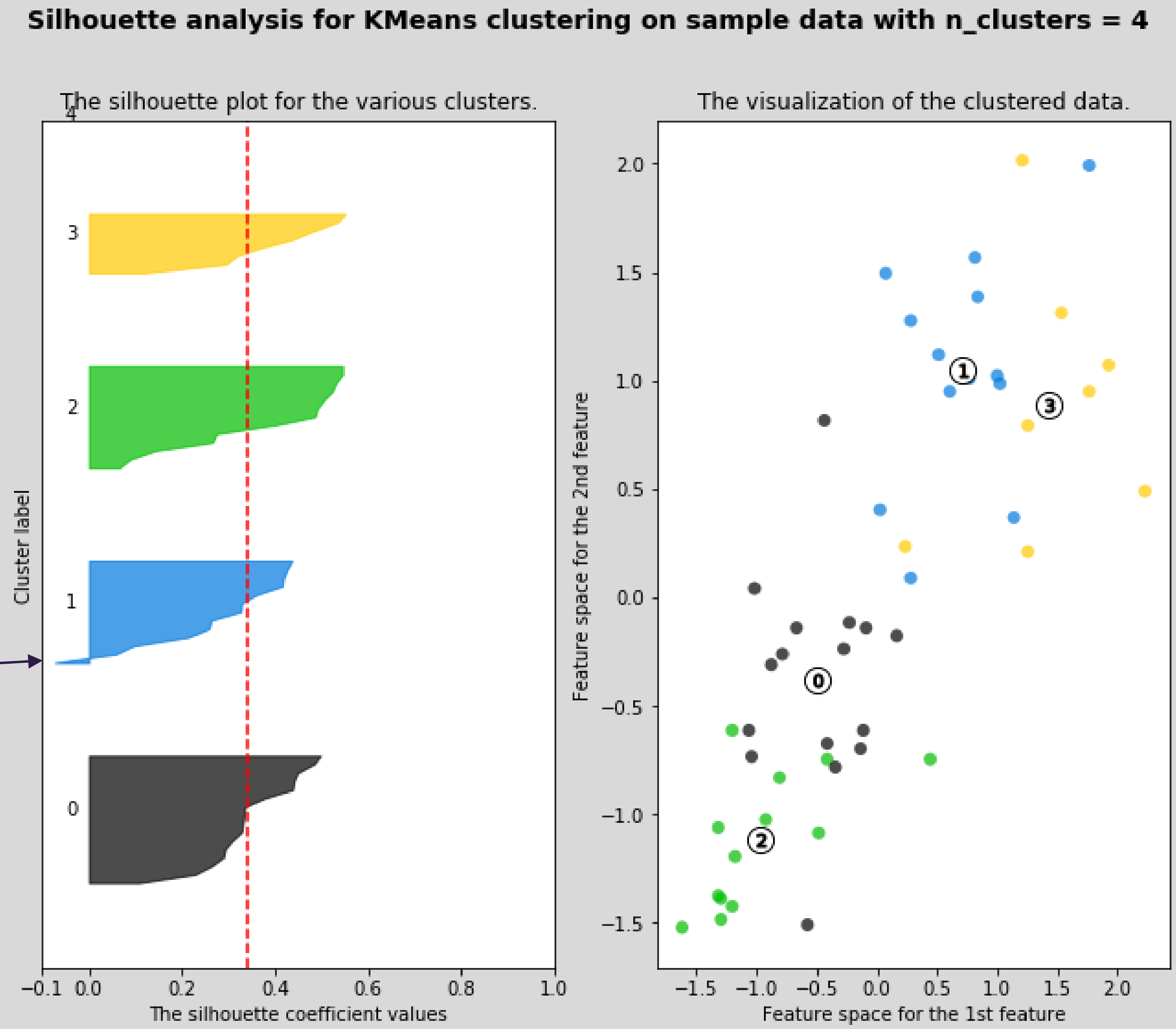
We define $s_i = \frac{b_i - a_i}{\max(a_i, b_i)}$ to be the silhouette score for observation i.

Clustering Diagnostics: Silhouette Plot

$$s_i = \frac{b_i - a_i}{\max(a_i, b_i)}$$

- Interpretation
  - Observations with $s_i \approx 1$ are well-clustered
  - Observations with $s_i \approx 0$ lie between two clusters
  - Observations with $s_i < 0$ are probably in the wrong cluster

Example: Diagnoses by State



Something may be in the wrong cluster

Outline

- Inter-observational Distances
- Partition-based Clustering
- Hierarchical Clustering
- Diagnostics and Optimizing Number of Clusters
- Density-based Clustering

Hierarchical Clustering

- K-Means clustering requires pre-specifying the number of clusters
- Hierarchical clustering does not require committing to a particular number of clusters
- Two types of hierarchical clustering
  - Agglomerative clustering (bottom-up approach)
  - Divisive clustering (top-down approach)

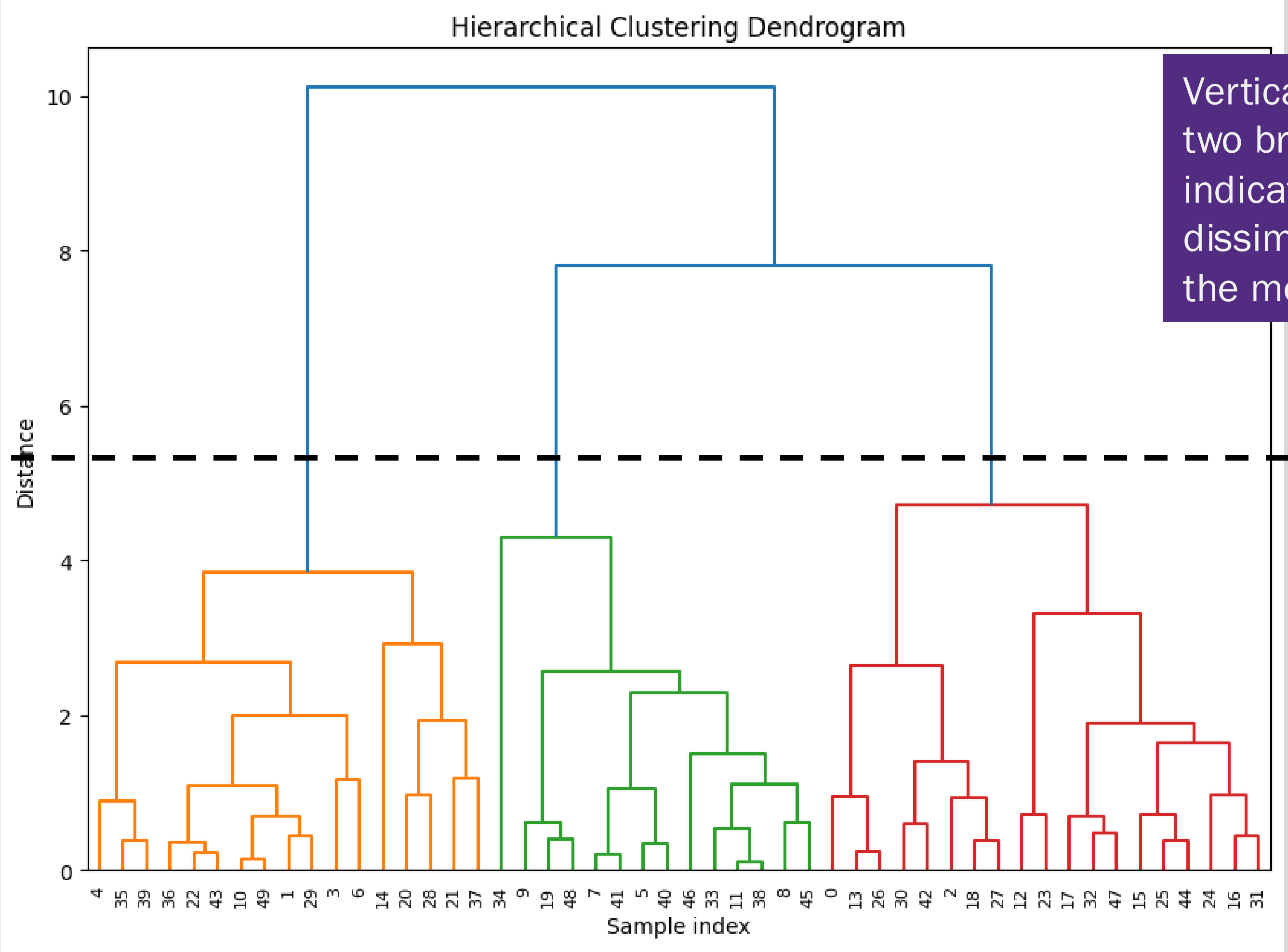
Agglomerative Clustering

The basic algorithm:

- Each observation starts as its own cluster
- At each step of the algorithm, two **clusters** that are most “similar” are combined into a new larger cluster
- This process of combining clusters is repeated until all observations are members of one single large cluster

Agglomerative clustering is particularly well-suited for identifying small clusters or when you believe there is a hierarchy among clusters.

Hierarchical clustering – interpreting the *dendrogram*



Vertical axis value at which two branches are merged indicates the overall dissimilarity of the items in the merged branches.

By drawing a horizontal *cut* at given height in the dendrogram, clusters are formed where samples are clustered into the branch that is cut by the line

Each leaf is a single sample

Agglomerative Clustering – Determining Cluster Similarity

Calculating the similarity of two clusters each with ≥ 1 sample requires a *linkage function* that outputs a scalar measure of the similarity of the two clusters

A few common approaches:

- *Complete (or maximum) Linkage Clustering:* For two clusters, determine the maximum dissimilarity between any observation in the first cluster and any observation in the second cluster.
- *Single Linkage Clustering:* For two clusters, determine the minimum dissimilarity between any observation in the first cluster and any observation in the second cluster.
- *Average Linkage Clustering:* Compute all pairwise dissimilarities between observations in the first and second cluster and calculate the average.

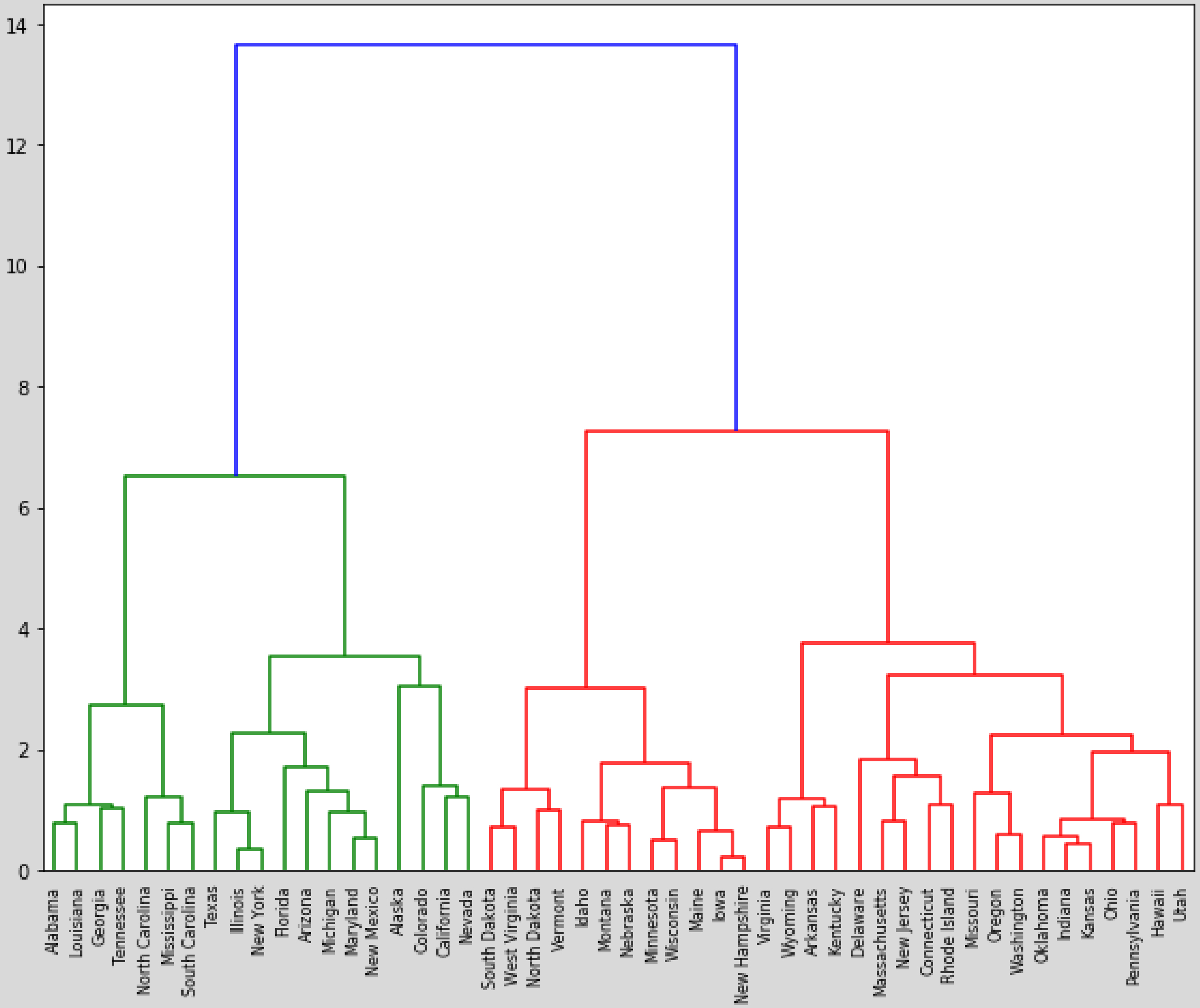
Other aggregation approaches are also possible.

Agglomerative Clustering

A popular approach is known as *Ward's Method:*

- Instead of joining two clusters based on their distances, use information about the variance of observations within clusters.
- Specifically, at each step, join two clusters whose merged cluster has the smallest within-cluster sum of squared distances.

Agglomerative Clustering – Ward's Method - Diagnoses



Outline

- Inter-observational Distances
- Partition-based Clustering
- Hierarchical Clustering
- Diagnostics and Optimizing Number of Clusters
- Density-based Clustering

Optimal Cluster Counts

But… how do we choose the optimal number of clusters?

There is no single principled way to choose clusters. We will explore two types of heuristic methods:

- Direct Methods that involve optimizing a particular criterion
  - Silhouette method
  - Elbow method

- Testing Methods that evaluate evidence against a null hypothesis
  - Gap Statistic

Elbow Method

As previously, let $W(C_k)$ be the within-cluster sum of squared distances between all pairs for cluster k for a particular clustering, and let

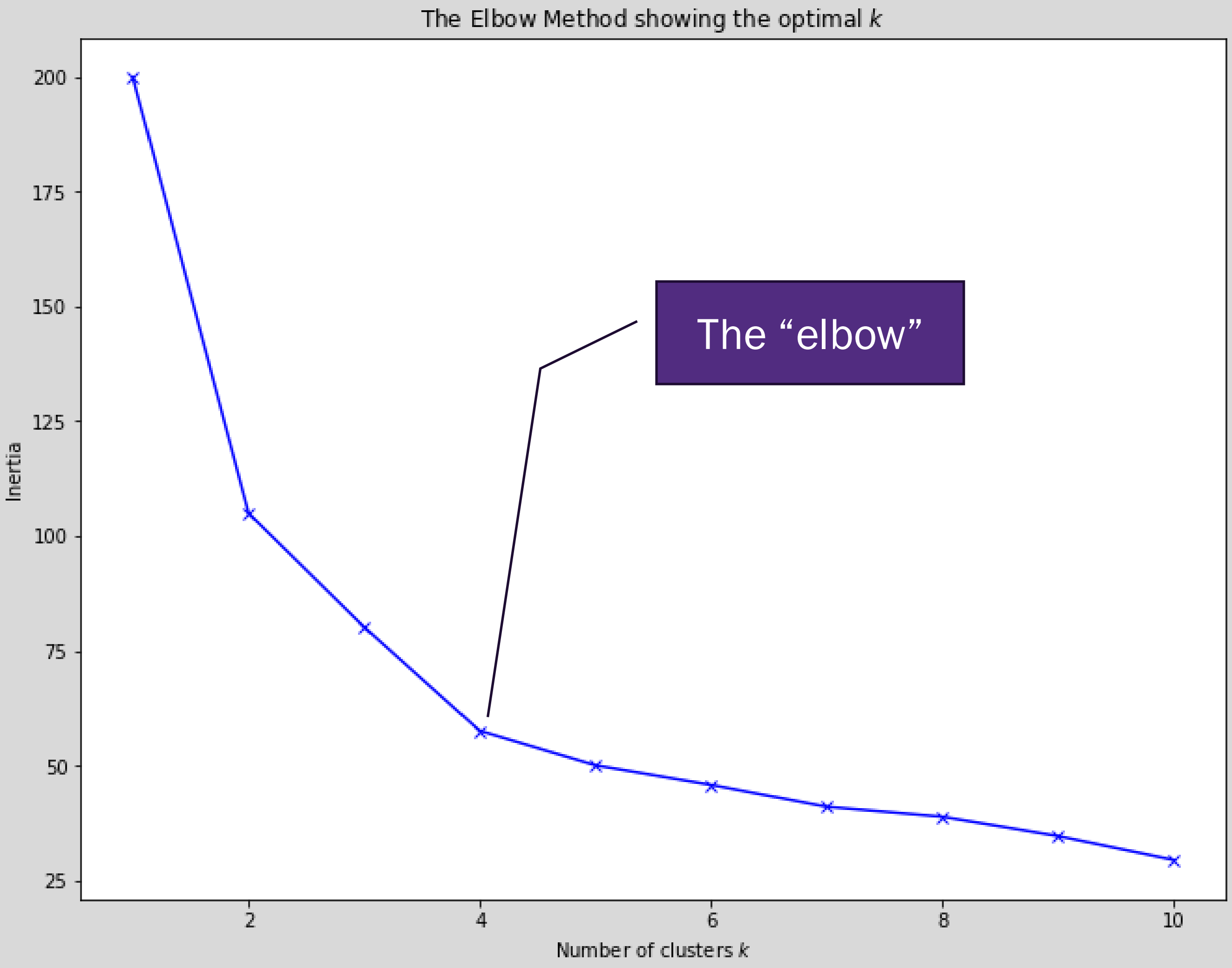
$$T_K = \sum_{k=1}^{K} W(C_k)$$

be the total within-cluster variation for the clustering of K clusters.

1. For a particular clustering method, let K vary over a range of values (e.g., 1 to 10)
2. Compute T_K for each K
3. Plot T_K against K and look for a clear bend in the graph

The value where the bend occurs is considered the correct number of clusters

Elbow Method



The “elbow”

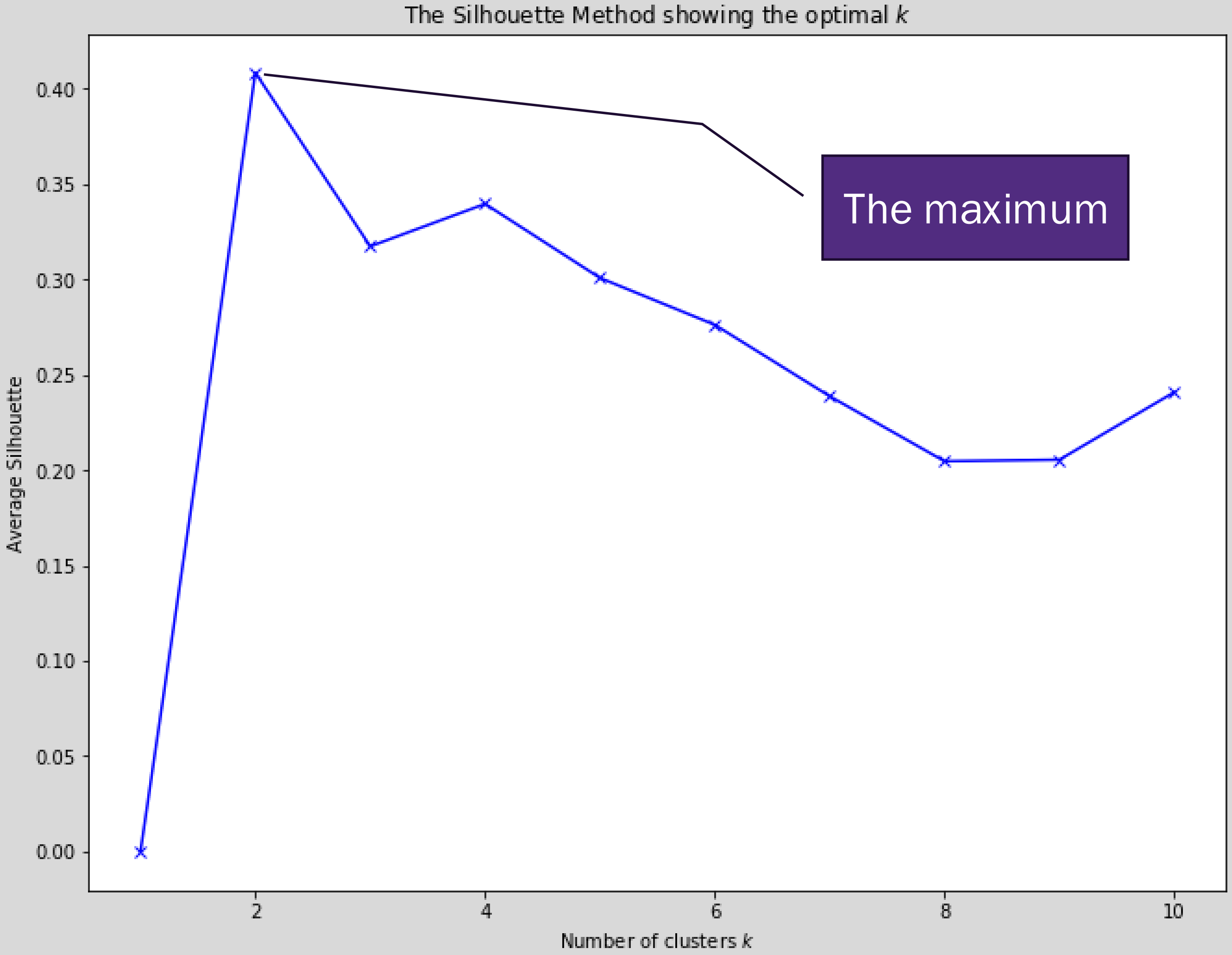
Average Silhouette Method

Similar to the elbow method:

1. For a particular clustering method, let K vary over a range of values (e.g., 1 to 10)
2. For each K, calculate the average silhouette across all observations
$$S_K = \frac{1}{n}\sum_{i=1}^{n} s_i$$
3. Plot S_K against K

The value of K where S_K is maximized is considered the appropriate number of clusters.

Average Silhouette Method



The maximum

Comments

- The elbow method suggests four (4) clusters for K-Means
- Average silhouette method suggests two (2)
- Computation for each method are different - inconsistent results are not uncommon
- Both approaches measure global clustering characteristics only, and are heuristic approaches

Gap Statistic

- **Idea:** For a particular choice of K clusters, compare the total within-cluster variation to the **expected within-cluster under the assumption that the data have no obvious clustering** (i.e., randomly distributed)

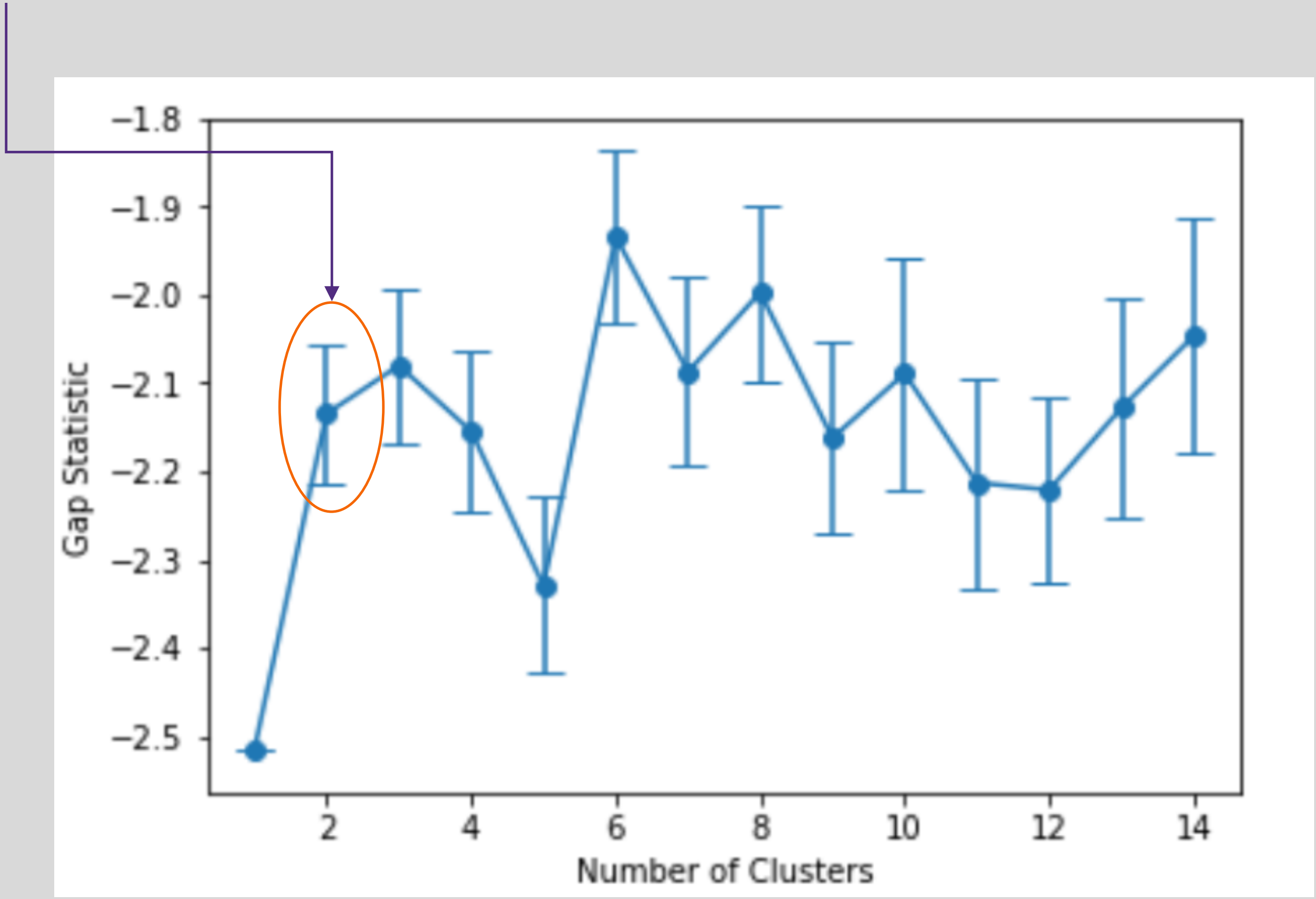
- The Gap Statistic in essence detects whether the observed data clustered into K groups has a more extreme metric value than random data

Gap Statistic Algorithm

1. Cluster the data at varying number of total clusters K. Let T_K be the total within-cluster sum of squared distances
2. Generate B reference data sets of size n, with the simulated values of variable j uniformly generated over the range of the observed variable x_j. Typically, $B = 500$.
3. For each generated data set $b = 1, \dots, B$, perform the clustering for each K. Compute the total within-cluster sum of squared distances, $T_k^{(b)}$. (What does this imply about computational requirements?)
4. Compute the Gap Statistic: $Gap(K) = \left[\frac{1}{B}\sum_{b=1}^{B} \log\left(T_K^{(b)}\right)\right] - \log(\mathrm{T_K})$
5. Let $\overline{w} = \frac{1}{B}\sum_{b=1}^{B} \log\left(T_K^{(b)}\right)$. Compute the standard deviation $sd(K) = \sqrt{\frac{1}{B}\sum_{b=1}^{B}\left(\log\left(T_K^{(b)}\right) - \overline{w}\right)^2}$. Define $s_K = sd(K)\sqrt{1 + {}^{1}/_{B}}$.
6. Finally, choose the number of clusters as the smallest K such that $Gap(K) \geq Gap(K+1) - s_{k+1}$

Gap Statistic Algorithm

$Gap(K) \geq Gap(K+1) - s_{k+1}$



Gap Statistic Algorithm

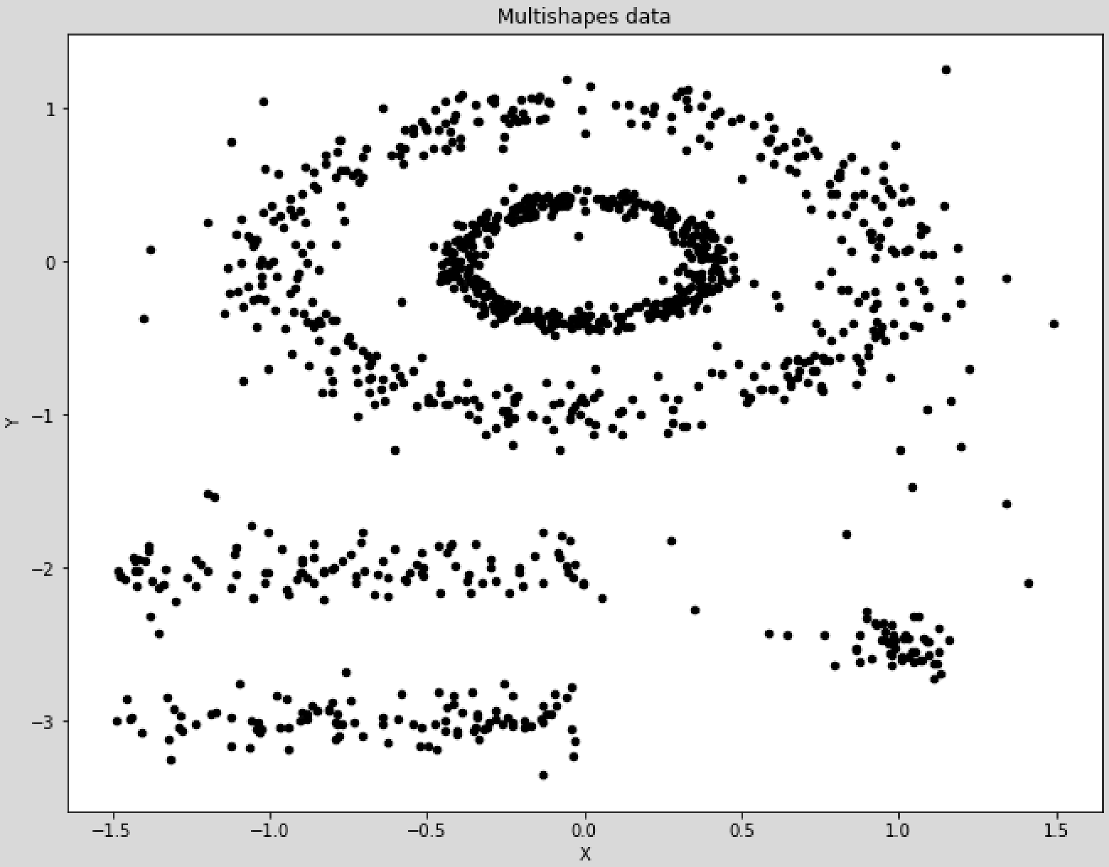
- K-Means clustering is optimized for $K = 2$ based on the Gap Statistic
- This is a principled approach to choosing cluster sizes, though clearly different conclusions can be reached based on different clustering approaches
- The Gap Statistic is generally understood to be conservative—tends to err on the side of picking a smaller number of clusters
- An alternative form of the gap statistic which replaces the $\log(T_K)$ and $\log(T_K^{(b)})$ terms in the formula with

$$Gap^*(K) = \left[\frac{1}{B}\sum_{b=1}^{B} T_K^{(b)}\right] - \mathrm{T_K}$$

This turns out to be less conservative.

More Clustering

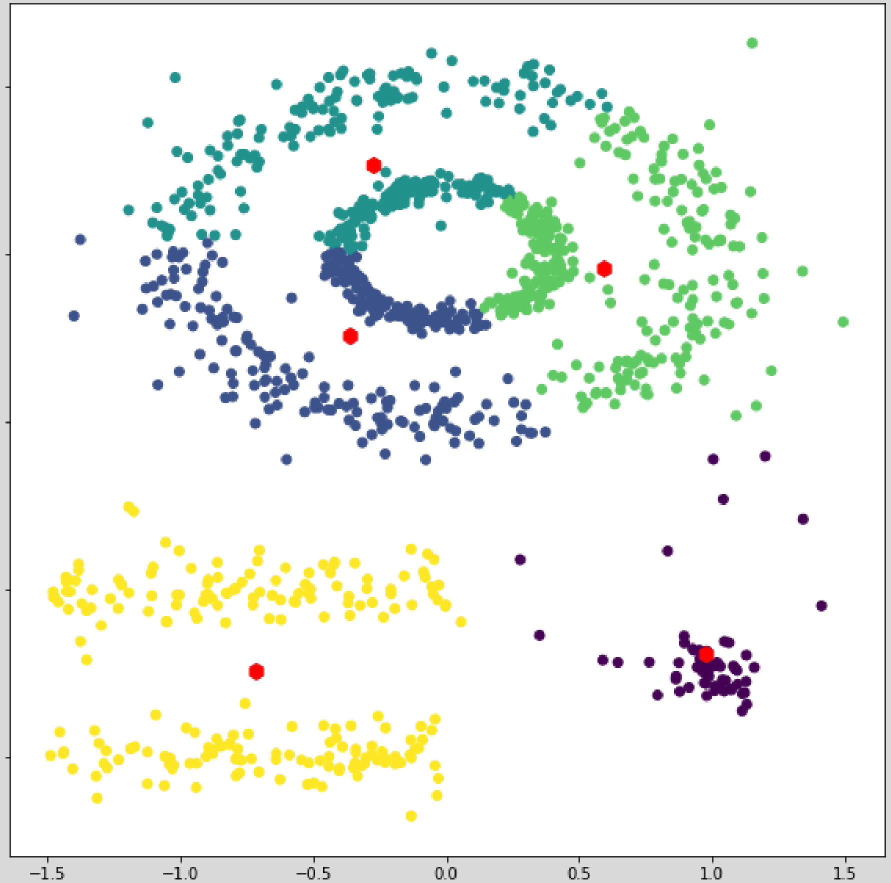
- Imagine trying to cluster something like this



What happens when we try K-means?

More Clustering

- K-Means struggles with this data set.



Outline

- Inter-observational Distances
- Partition-based Clustering
- Hierarchical Clustering
- Diagnostics and Optimizing Number of Clusters
- Density-based Clustering

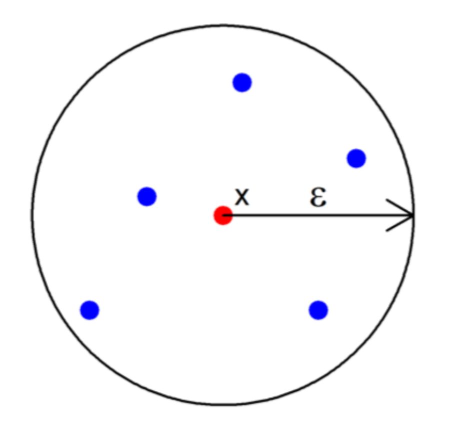
DBSCAN: Density-based Clustering Algorithm

- Unlike previous clustering algorithms, DBSCAN
  - Can find any shape of clusters
  - Identifies observations that do not belong to clusters as outliers
  - Does not require specifying the number of clusters
  - Can be used for predicting cluster membership for new data

DBSCAN: Density-based Clustering Algorithm

The DBSCAN algorithm identifies dense regions of observations. However, we need to specify two parameters for the algorithm:

1. ϵ: the radius of a neighborhood around an observation
2. $MinPts$: the minimum number of points within an ϵ radius of an observation to be considered a “core” point.



Example of core observation with $MinPts = 6$

DBSCAN: Density-based Clustering Algorithm

There are three types of points:

- **Core points:** observations with $MinPts$ total observations within an ϵ radius
- **Border points:** observations that are not core points, but are within ϵ of a core point
- **Noise points:** everything else

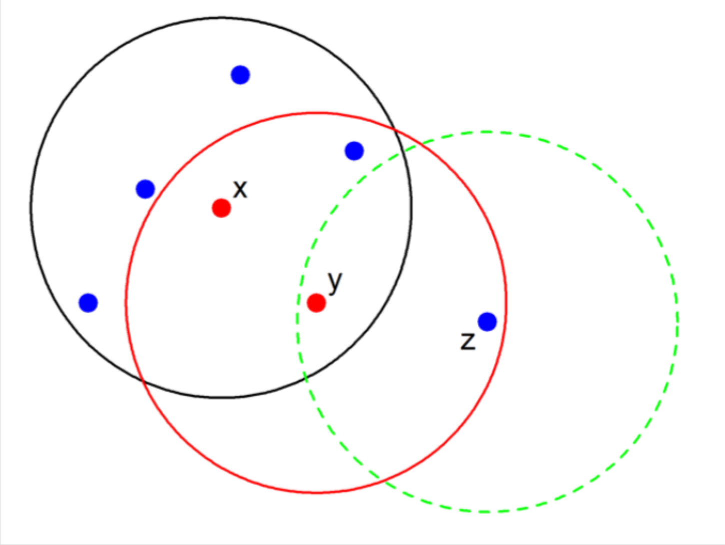
In this example:

x is a core point

y is a border point (so are the other blue dots

in the ϵ radius of x (black circle)

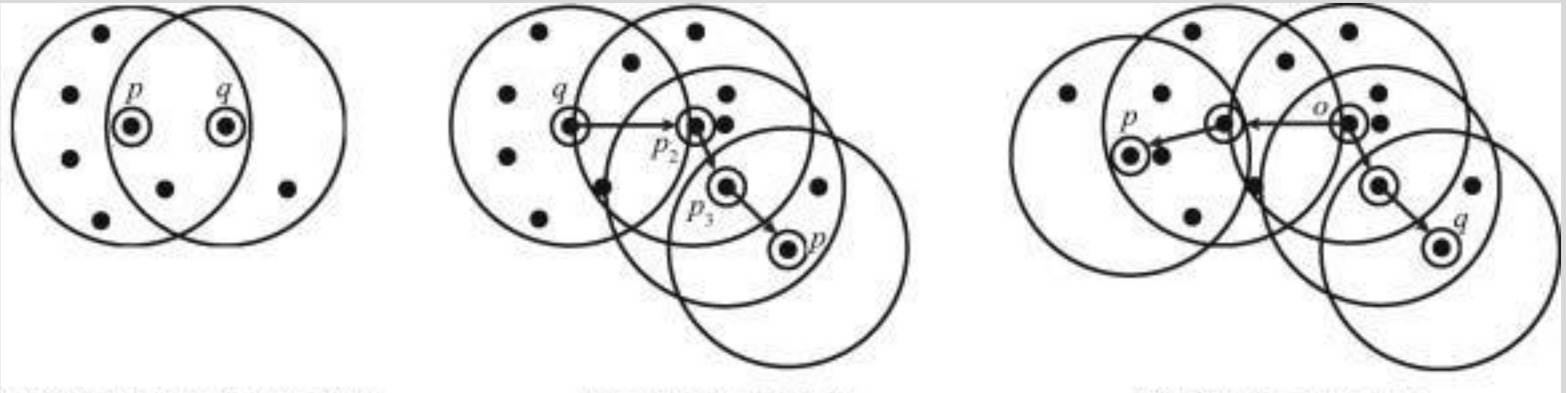
z is a noise point



DBSCAN: Density-based Clustering Algorithm

Two terms:

- **Density-reachable:** Point A is density-reachable from point B if there is a set of core points leading from B to A.
- **Density-connected:** Two points A and B are density-connected if there is a core point C such that both A and B are density-reachable from C.
- A density-based cluster is defined as a group of density-connected points.



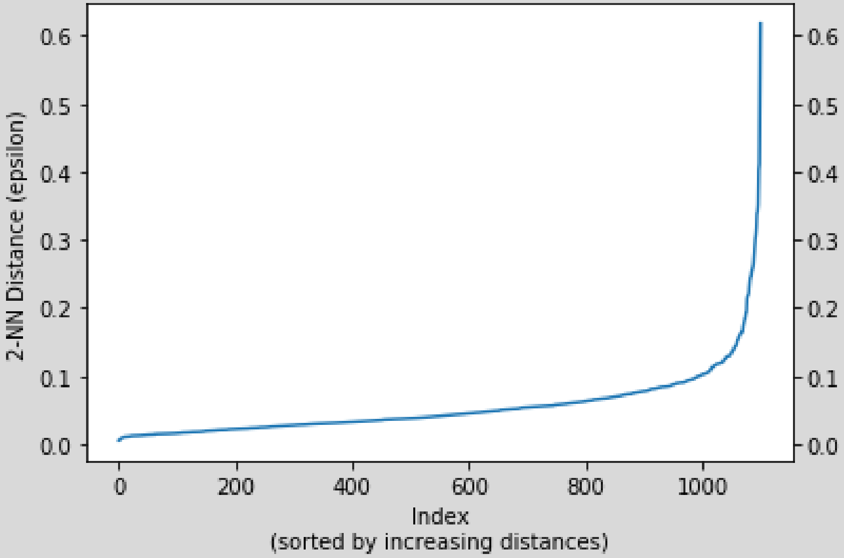
(a) Directly density-reachable (b) Density-reachable (c) Density-connected

Figures: https://levelup.gitconnected.com/dbscan-a-density-based-clustering-algorithm-110b726fd6fe

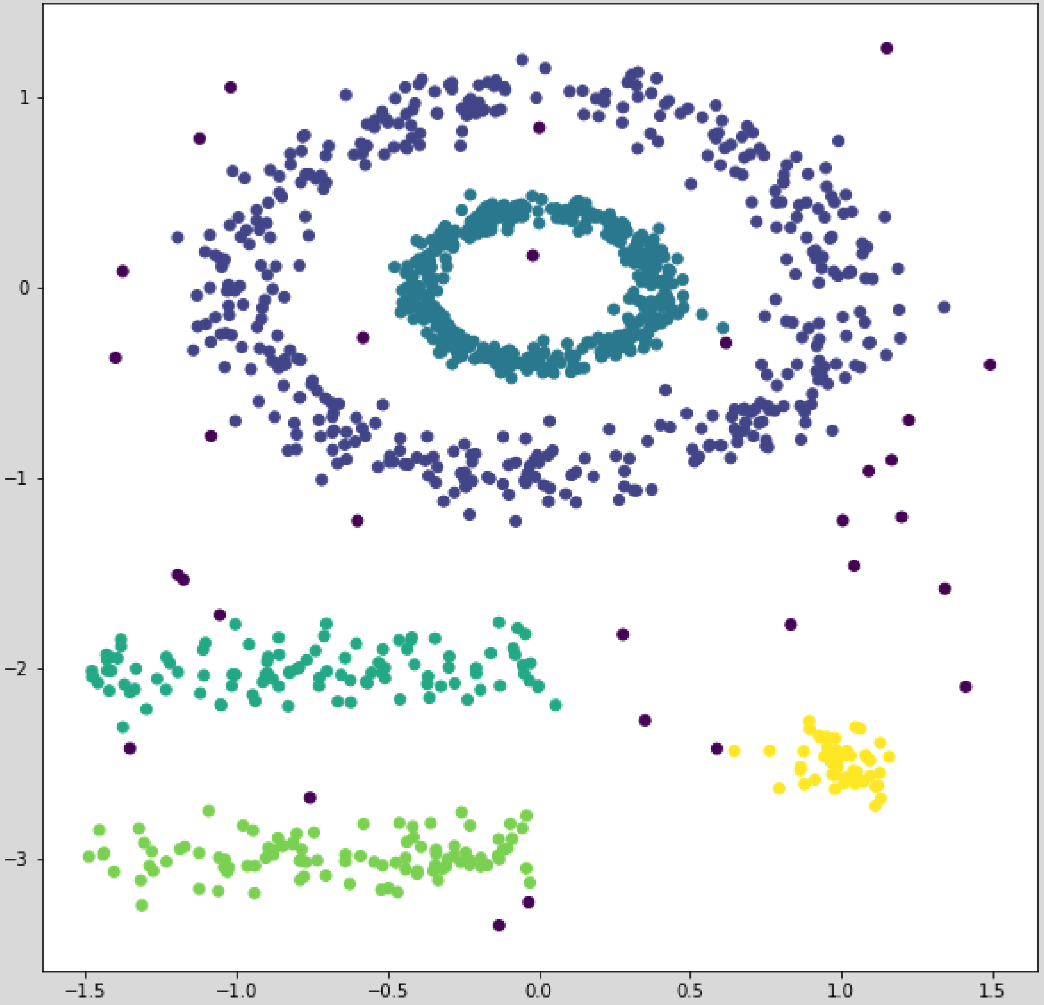
DBSCAN: Density-based Clustering Algorithm

Choosing ϵ (the radius of the neighborhood):

- Compute the k-th nearest neighbor distance for each point
- Plot the distances in sorted order
- Look for a bend (the “knee”) in the plot and use the distance at the knee as the choice of ϵ.



DBSCAN: Density-based Clustering Algorithm



Questions?