Clustering
Motivation, Methods, and Examples

Natesh Pillai

Spring 2025

Thanks to Alex Young

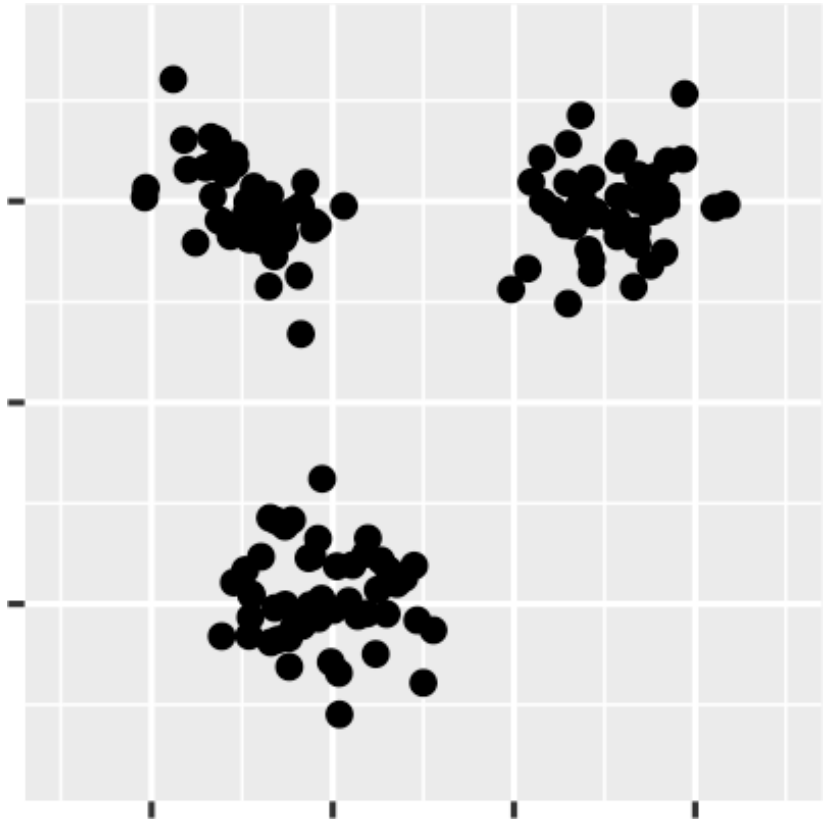
Learning Objectives

- Meaning and motivations for clustering
  - Examples and applications
- Highlight challenges and sensitivity to the shape of clusters
  - High-dimensional data is particularly difficult
- Present common notions of distance/dissimilarity
  - Highlight the associated strengths and weaknesses
  - Outline weaknesses and potential solutions
- Present methods for estimating the number (or existence) of clusters
- Discuss the previous two points through two popular methods
  - K-means
  - DBSCAN
  - t-SNE

Clustering: Motivations and Meaning

What does it mean for a data set to have clusters?

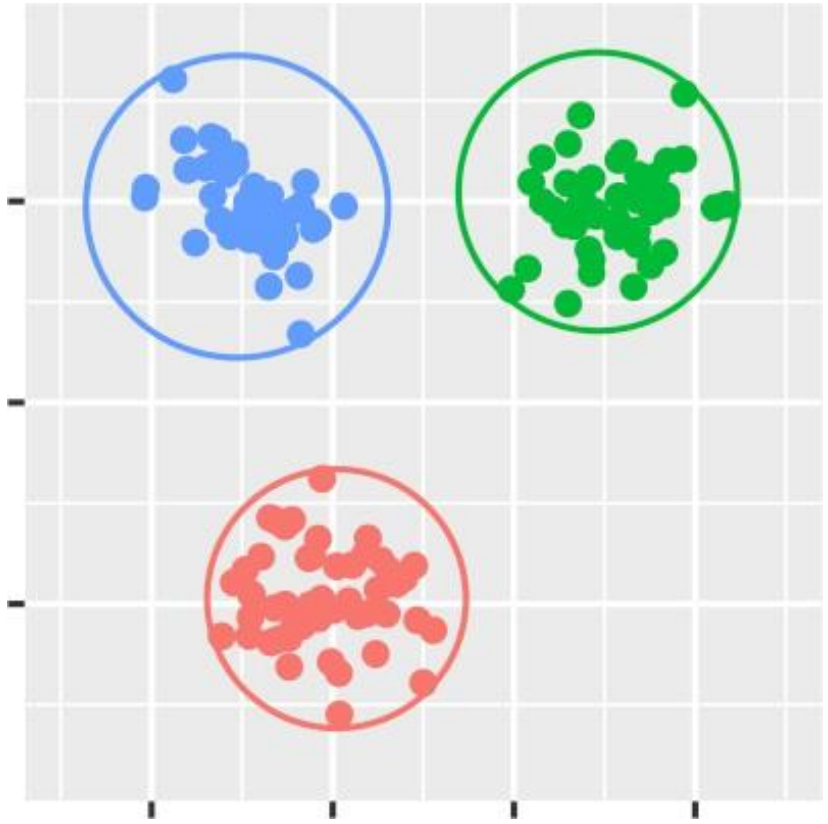
- Clusters are subsets of our data such that
  - Samples in the same cluster are (very) similar
  - Samples in different clusters are (very) dissimilar
- Goal of cluster analysis is to discover subgroups of similar observations or variables
  - Similarity is not related to a specific response (unsupervised learning)



Clustering: Motivations and Meaning

What does it mean for a data set to have clusters?

- Clusters are subsets of our data such that
  - Samples in the same cluster are (very) similar
  - Samples in different clusters are (very) dissimilar
- Goal of cluster analysis is to discover subgroups of similar observations or variables
  - Similarity is not related to a specific response (unsupervised learning)



Applications of clustering

- Numerous applications to both unsupervised and supervised learning
- Exploratory data analysis
  - e.g. Finding subgroups in data for separate or comparative analysis
- Data compression
  - Some algorithm provide representative point for each cluster

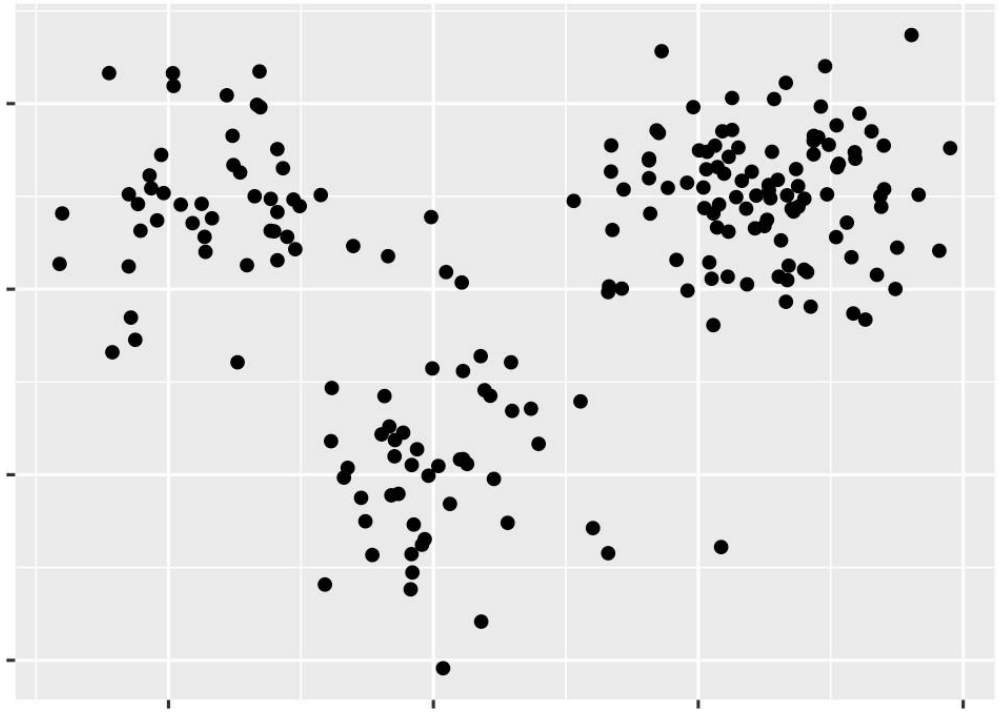
Examples

- Bioinformatics
  - Inferring population structure
- Imaging
  - Tissue differentiation
- Business
  - Market Segmentation
- Computer Science
  - Recommender systems
- And many more fields
  - Geology, Climatology, Robotics, etc.

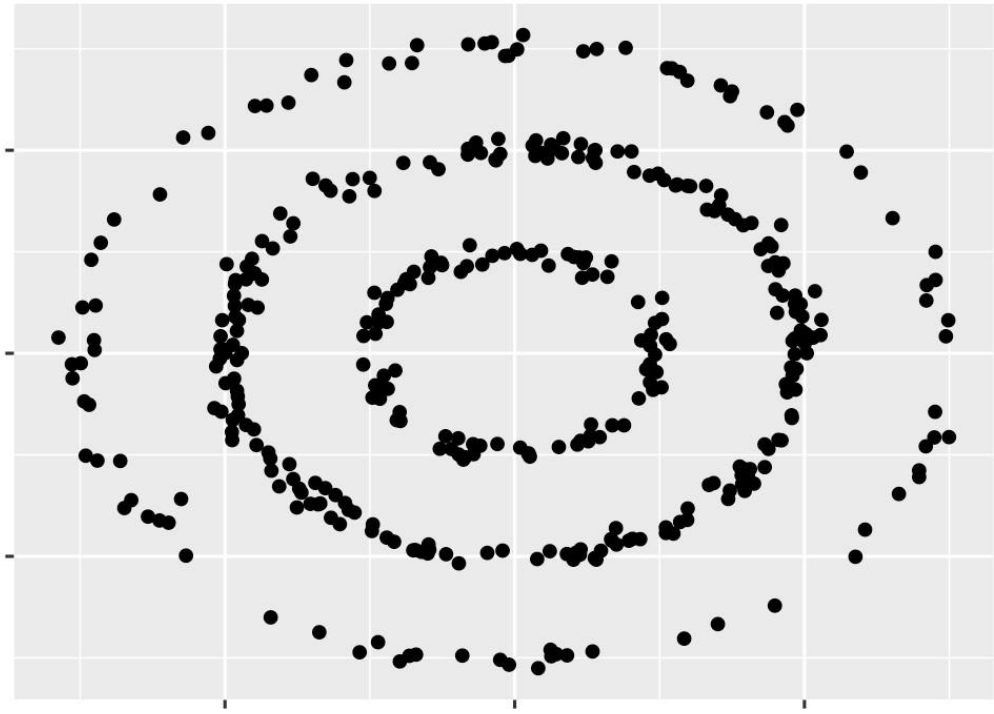
Challenges: (Dis)similarity is context dependent

In each image, how many clusters do you see? How would you arrange points into clusters?

Example 1



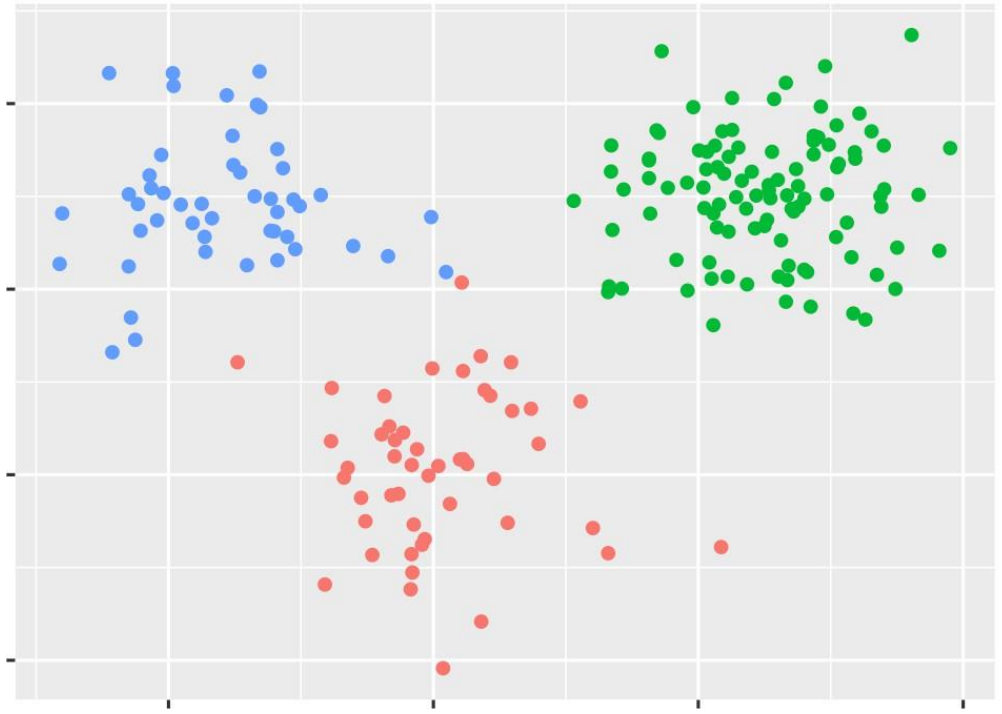
Example 2



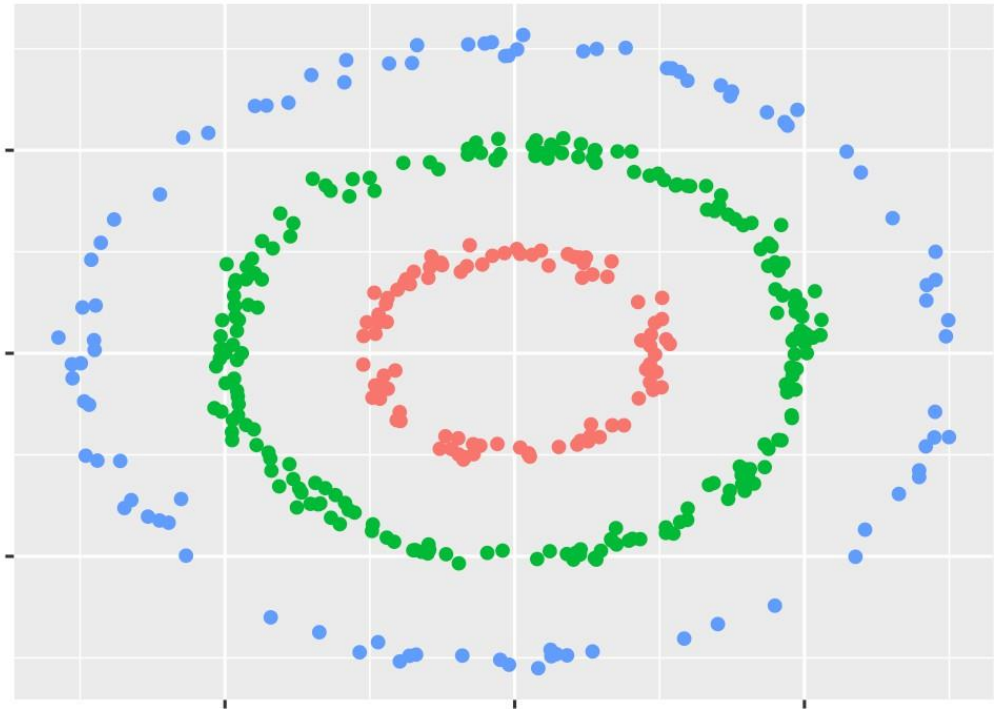
Challenges: (Dis)similarity is context dependent

In each image, how many clusters do you see? How would you arrange points into clusters?

Example 1



Example 2

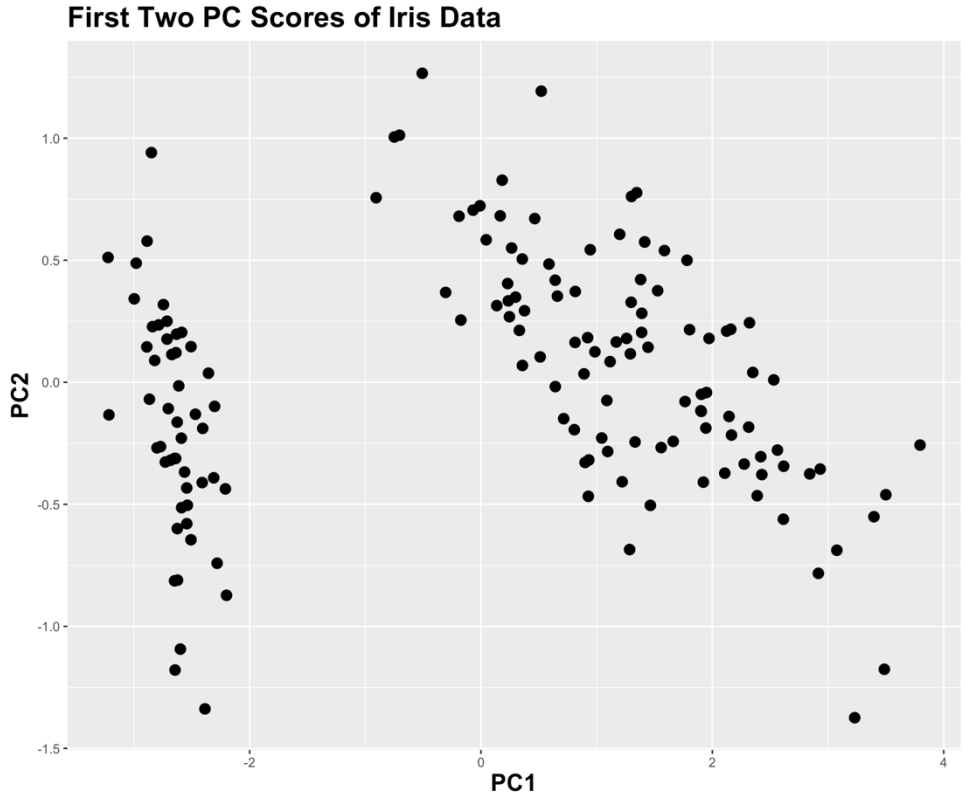


Challenges: Number (existence) of clusters is often unknown.

Consider the *iris* data set which has measurements of 150 different flowers.

Sepal.Length	Sepal.Width	Petal.Length	Pet
5.1	3.5	1.4	
4.9	3.0	1.4	
4.7	3.2	1.3	
4.6	3.1	1.5	
5.0	3.6	1.4	
5.4	3.9	1.7	

How many clusters are there?



Challenges and Implicit Assumptions

- Different notions of what makes points (dis)similar.
  - (Euclidean) distance, cosine similarity, and more
  - Different choices of (dis)similarity prioritize different types of cluster shapes
- The number (or existence) of clusters is often unknown.
  - Can we specify the number of cluster based on outside expert knowledge, or
  - Estimate directly from the data
- We are often dealing with high-dimensional data that we cannot easily visualize
  - Makes assessing the two issues above difficult in practice

Key takeaway

Preprocessing and choice of clustering algorithm greatly influence the results

- Some algorithms prioritize certain shapes of clusters and miss or overfit others.
- There are a lot of choices.

First, some notation

Data are stored in an $N \times d$ data matrix $\mathbf{X} \in \mathrm{R}^{N \times d}$ where the rows correspond to different samples and the columns correspond to some common variable measured across the samples

- $\mathbf{X}_{ij}$ is the measurement of variable j in the i observation.

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
5.1	3.5	1.4	0.2
4.9	3.0	1.4	0.2
4.7	3.2	1.3	0.2

- We'll use the vector notation $\vec{x}_i$ to refer to the ith observation correspondingly row i of $\mathbf{X}$. In the above example,

$$\vec{x}_1 = (5.1, 3.5, 1.4, 0.2)$$
$$\vec{x}_3 = (4.7, 3.2, 1.3, 0.2)$$

First, some notation

The data have an associated sample mean

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} \vec{x}_i$$

and sample covariance matrix

$$\hat{\Sigma} = \frac{1}{N-1}\sum_{i=1}^{N} (\vec{x}_i - \bar{x})^T (\vec{x}_i - \bar{x}) \in \mathbb{R}^{d \times d}$$

An aside on notation

For the iris data, the sample mean is

```
iris.mean()
```

```
## Sepal.Length    5.843333
## Sepal.Width     3.057333
## Petal.Length    3.758000
## Petal.Width     1.199333
## dtype: float64
```

and the sample covariance is

```
iris.cov()
```

```
##                Sepal.Length  Sepal.Width  Petal.Length  Petal.Width
## Sepal.Length       0.685694    -0.042434      1.274315     0.516271
## Sepal.Width       -0.042434     0.189979     -0.329656    -0.121639
## Petal.Length       1.274315    -0.329656      3.116278     1.295609
## Petal.Width        0.516271    -0.121639      1.295609     0.581006
```

Measuring dissimilarity

Choosing a measure of dissimilarity is often the first (implicit) step in clustering

Distance-based Measurement

For observations $\vec{x}_j, \vec{x}_k \in \mathbb{R}^d$

$$d_{Man}(\vec{x}_j, \vec{x}_k) = \sum_{l=1}^{d} \left| \boldsymbol{X}_{jl} - \boldsymbol{X}_{kl} \right| = \| \vec{x}_j - \vec{x}_k \|_1$$

$$d_{Euc}(\vec{x}_j, \vec{x}_k) = \left(\sum_{l=1}^{d} \left(\boldsymbol{X}_{jl} - \boldsymbol{X}_{kl} \right)^2 \right)^{1/2} = \| \vec{x}_j - \vec{x}_k \|_2$$

$$d_{Mahal}(\vec{x}_j, \vec{x}_k) = (\vec{x}_j - \vec{x}_k)\hat{\Sigma}^{-1}(\vec{x}_j - \vec{x}_k)^T$$

Measuring dissimilarity

Correlation-based Measurement: Cosine similarity

For observations $\vec{x}_j, \vec{x}_k \in \mathbb{R}^d$

$$d_{Cos}\left(\vec{x}_j, \vec{x}_k\right) = 1 - \frac{\vec{x}_j \cdot \vec{x}_k}{\| \vec{x}_j \|_2 \| \vec{x}_k \|_2} \in [0,2]$$

Importance of location and scale

- All of the preceding measures of dissimilarity depend on a coordinate- wise comparison or product of observations. Consider this modified subset of the *mtcars* data

	mpg	hp	wt
Mazda RX4	21.0	110	2620
Mazda RX4 Wag	21.0	110	2875
Datsun 710	22.8	93	2320
Hornet 4 Drive	21.4	110	3215

where *mpg* is miles per gallon, *hp* is horsepower, and *wt* is weight in pounds.

- Will each variable (column) have a similar impact on pairwise dissimilarities between points?

- For the Mazda RX4 and Datsun 710 we have the following dissimilarities.

Manhattan	318.80000
Euclidean	300.48667
Mahalonobis	0.09922
Cosine	0.00000

Importance of location and scale

- Let's rescale the *wt* variable to be in thousands of pounds

	mpg	hp	wt
Mazda RX4	21.0	110	2.620
Mazda RX4 Wag	21.0	110	2.875
Datsun 710	22.8	93	2.320
Hornet 4 Drive	21.4	110	3.215
Hornet Sportabout	18.7	175	3.440
Valiant	18.1	105	3.460

- What happens to the distances?

- For the Mazda RX4 and Datsun 710 the dissimilarities are now

Manhattan	19.10000
Euclidean	17.09766
Mahalonobis	0.09922
Cosine	0.00134

Impact of Preprocessing

Standardization

- The common approach to prevent one variable from having an outsized influence on dissimilarities is to center and standardize

$$\boldsymbol{X}_{ij} \rightarrow \frac{\boldsymbol{X}_{ij} - \bar{x}_j}{\left(\hat{\Sigma}_{jj}\right)^{1/2}}$$

Normalization

- Alternatively, one can rescale all columns so they fall in interval [0,1]

$$\boldsymbol{X}_{ij} \rightarrow \frac{\boldsymbol{X}_{ij} - min_i \boldsymbol{X}_{ij}}{max_i \boldsymbol{X}_{ij} - min_i \boldsymbol{X}_{ij}}$$

Nonlinear transformations

- log(·), log(1+·), arcsinh(·), etc.

Effects of Standardization

- Dissimilarities between Mazda RX4 and Datsun 710 **before** standardization

Manhattan	19.10000
Euclidean	17.09766
Mahalonobis	0.09922
Cosine	0.00134

- Dissimilarities between Mazda RX4 and Datsun 710 after standardization

Manhattan	0.85321
Euclidean	0.49465
Mahalonobis	0.09922
Cosine	0.01502

Effects of Transformations

Dissimilarity	Translation	Rescaling	Rotation	Reflection	Subset or New Data
Manhattan	Same	Different	Different	Same	Same
Euclidean	Same	Different	Same	Same	Same
Mahalonobis	Same	Same	Same	Same	Same
Cosine	Different	Different	Same	Same	Same

K-means clustering

- partition N data points into K disjoint subsets $C_1, C_2, \dots C_K$
- minimize the within-cluster distance

$$\text{argmin}_C \sum_{k=1}^{K} \sum_{i \in C_k} \| \vec{x}_i - \vec{m}_k \|_2^2$$

  - $\vec{m}_1, \dots, \vec{m}_K$ are cluster centers

- How do we find $C_1, C_2, \dots C_K$ minimizing the within-cluster distance?

Finding optimal partitioning

- Option 1: search over all partitions
  - Naively, we could consider all possible configurations of data into K cluster, $\approx K^N$ with overcounting
  - Known to be NP-hard for ≥ 2 clusters
  - If K and d are fixed, then optimal solution can be found over a number of computations $\propto N^{dK+1}$, which is intractable in most settings

- Option 2: Iterative algorithm converging to (locally) optimal solution
  - Standard approach is based on Lloyd's algorithm
  - Special initialization to avoid convergence to poor optima This is the default method in *sklearn.cluster.KMeans()*

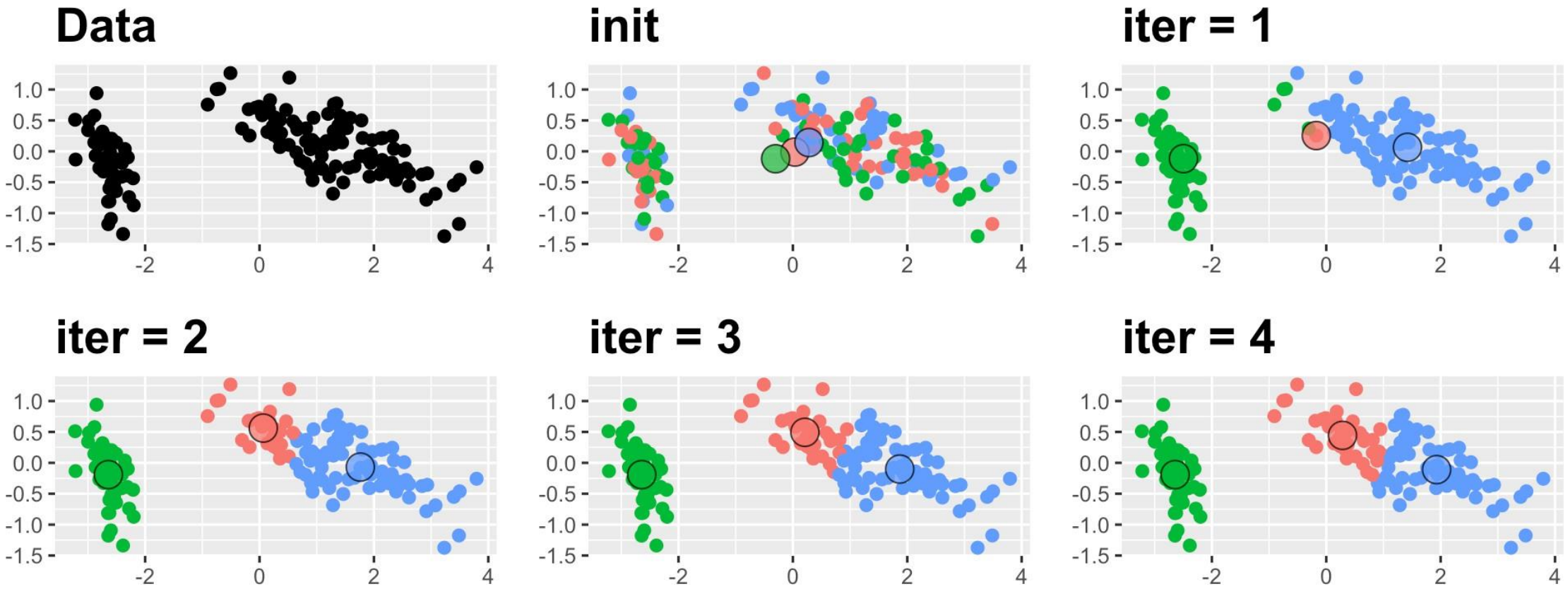
(Stuart) Lloyd's algorithm with random initializations

Algorithm:

1. Random initializations
   - Randomly assign partitions
2. Update cluster centers
   - Compute $\vec{m}_1, \ldots, \vec{m}_K$ by taking sample mean of points in each cluster
3. Reassign clusters
   - Assign i to cluster j if $\vec{x}_i$ is closest to $\vec{m}_j$
   - Repeat for all $i = 1, \ldots, N$
4. Iterate Steps 1 and 2 until convergence.

Iris demo: K = 3

Looking at first two PCs. Centers are outlined in black.



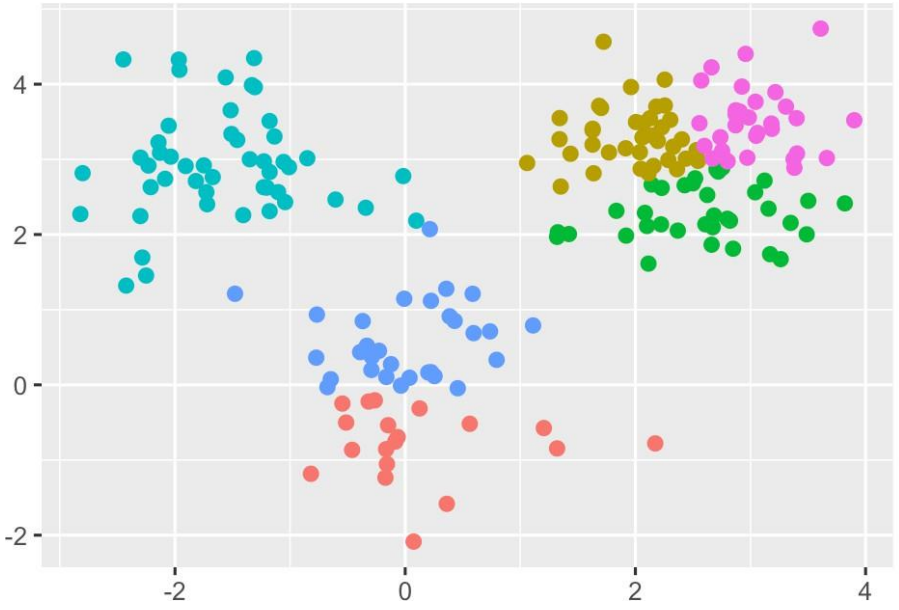
Strengths

- Typically converges very fast in practice
  - Worst case convergence takes $\propto N^{K+1}$iterations
  - In practice, typically linear in number of data points
  - Can set maximum number of iterations and converge to approximate suboptimal solutions
- Centers can be interpreted as *representative* prototypes of each cluster
  - New data can be merged into cluster with nearest center
- Clusters are based off interpretable notion of Euclidean distance
  - Points in a cluster are closer to one another than to points in other clusters (on average)
- Given these strengths K-means is often the default clustering algorithm used for many, but ...

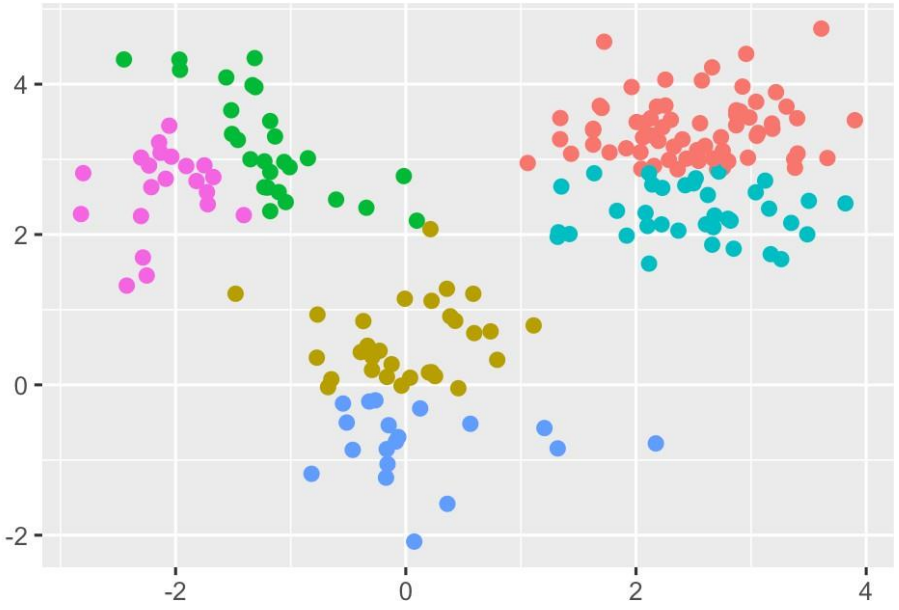
Limitations (computational)

- Requires choosing K in advance (more on this shortly)
  - No connection between K-means cluster solution and $(K+1)$ -means cluster solutions
  - Need to rerun for different values of K
- Convergence is dependent on initial condition

$T_K = 125$



$T_K = 112$



Addressing limitations (computational)

- Avoiding convergence to poor local optimal
  - Run many different ICs to convergence and pick best solution, and/or
  - Use better initialization, e.g. K-means++, which is default in *sklearn*

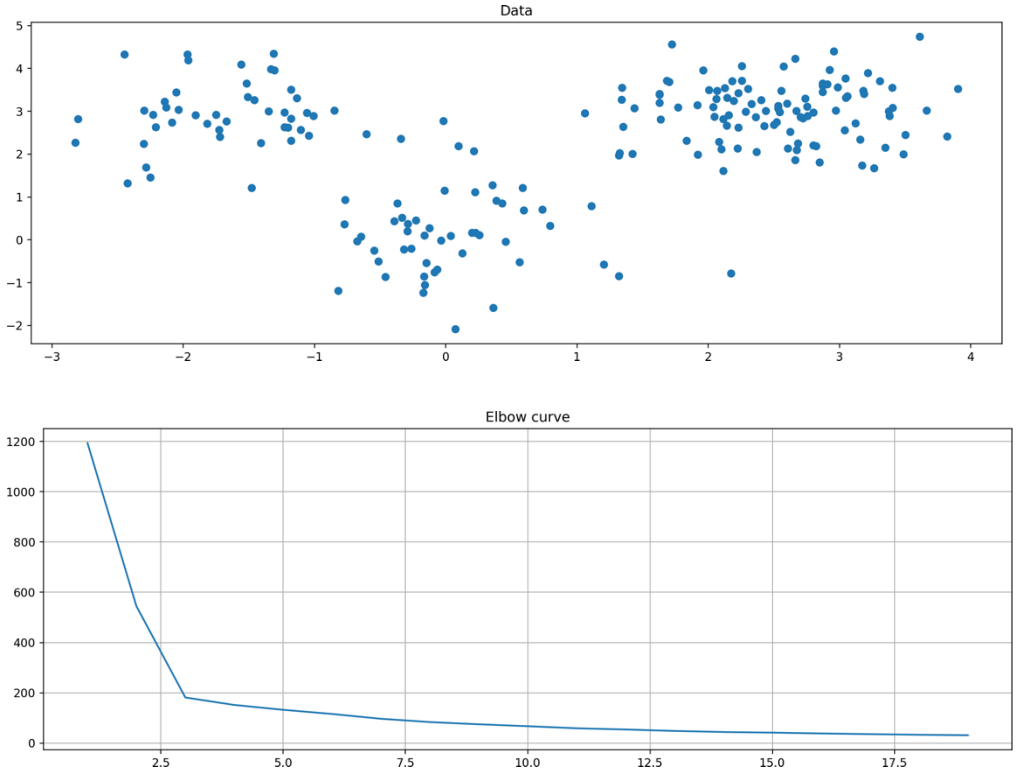
- Choosing K
  - Many different methods which tend to give (slightly) different results
  - Direct methods: optimize a target score
    - Elbow, sihouette diagnostics
  - Testing methods: comparison to null model
    - Gap Statistic

Elbow plot

- The inertia will decrease as the number of clusters increases.
- To choose K we find the inertia as a function of the number of clusters.

```
from sklearn.cluster import KMeans
dat = pd.read_csv("Data/three-clus.cs
inertias = []
for k in range(1, 20):
    kmeans = KMeans(n_clusters=k, n_i
    kmeans.fit(dat)
    inertias.append(kmeans.inertia_)
```

- Ideally (but rarely), we would see a clear delineation where the inertia saturates.



Gap Statistic

- Due to Tibshirani et al., (JRSS-B, 2001).

- Idea: For a particular choice of K clusters
  - Compare the total within cluster variation to the expected within-cluster variation under the assumption that the data have no obvious clustering (i.e., randomly distributed).
  - Can be used to select an optimal number of clusters or
  - As evidence that there is no clustering structure

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 uniformly generated over the range of the observed variable. Typically, $B = 500$.
3. For each generated data set $\mathrm{b} = 1, \ldots, B$ perform the clustering for each K. Compute the total within-cluster sum of squared distances $T_K^{(b)}$.
4. Compute the Gap statistic

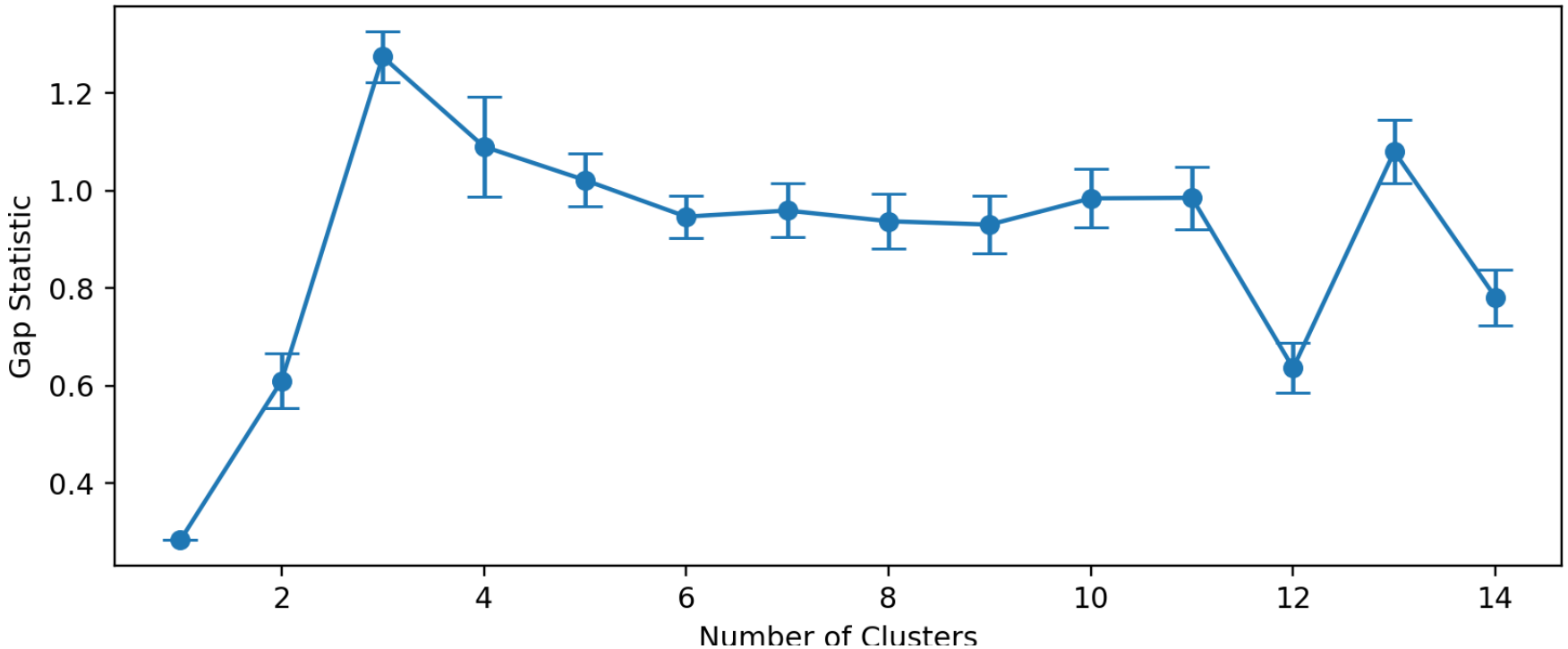
$$Gap(K) = \bar{w} - \log(T_K), \bar{w} = \frac{1}{B}\sum_{b=1}^{B} \log\left(T_K^{(b)}\right)$$

5. Compute the sample variance

$$var(K) = \frac{1}{B-1}\sum_{b=1}^{B}\left(\log\left(T_K^{(b)}\right) - \bar{w}\right)^2$$

Gap Statistic: Example

```
## Optimal number of clusters:  3
```



Gap Statistic: Observations

- This is a more principled approach to choosing cluster sizes, **but** clearly different conclusions can be reached based on different clustering approaches

- Generally understood to be conservative
  - Tends to err on the side of picking a smaller number of clusters

- Variants and modifications of this approach exist including
  - Gap* (Mohajer, M., Englmeier, K. H., & Schmid, V. J., 2011)
    - Tends to identify fewer clusters
  - Max Gap: selecting K from a range which maximizes $Gap(K)$
    - Tends to identify more clusters

Silhouette plots, coefficients

For a given clustering, we would like to determine how well each sample is clustered.

a_i = avg. dissimilarity of $\vec{x}_i$ with all other samples in same cluster

b_i = avg. dissimilarity of $\vec{x}_i$ with samples in the *closest* cluster

We then define

$$s_i = \frac{b_i - a_i}{\max(b_i, a_i)} \in (-1,1)$$

as the silhouette for $\vec{x}_i$.

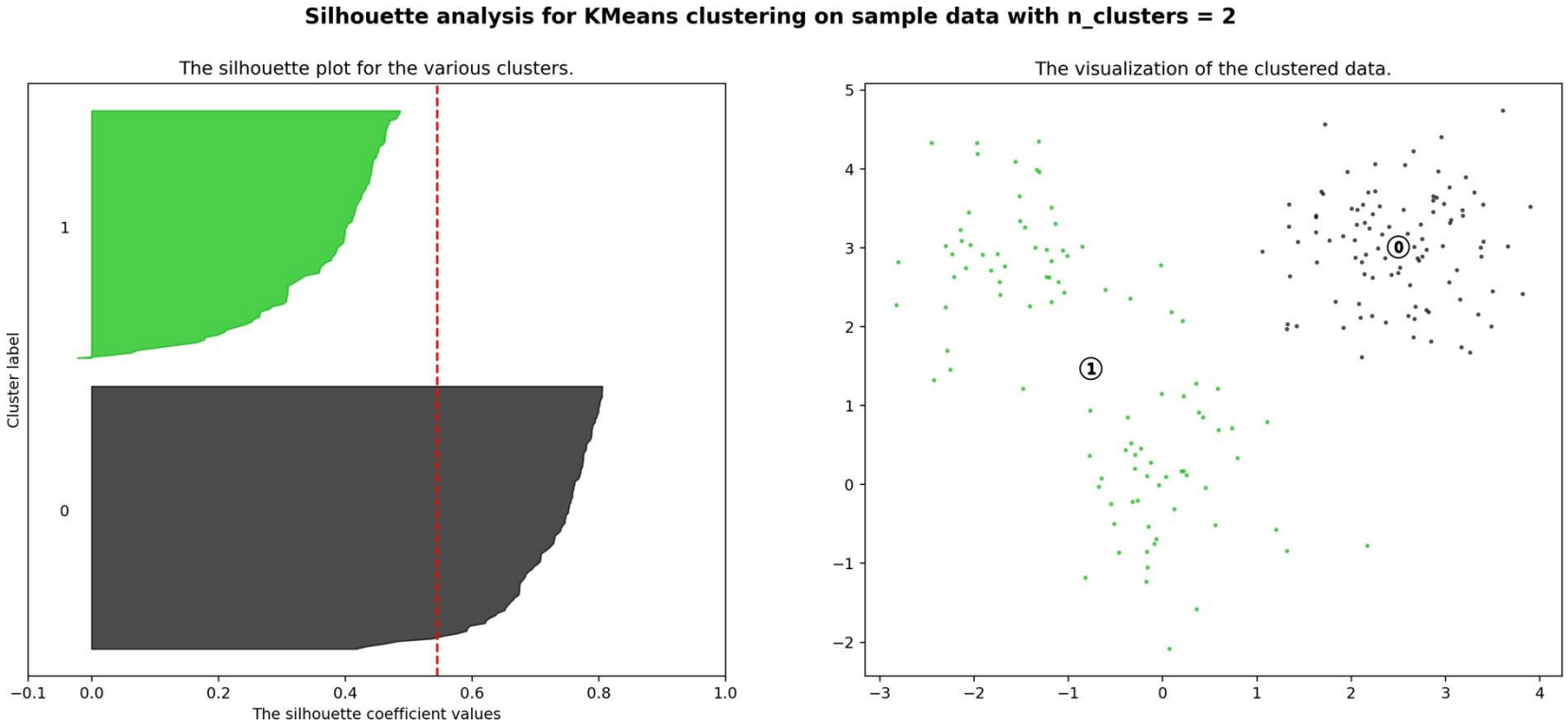
- Observations with $s_i \approx 1$ are well clustered
- Observations with $s_i \approx 0$ are between clusters
- Observations with $s_i < 0$ are probably in wrong cluster

We can use any dissimilarity!

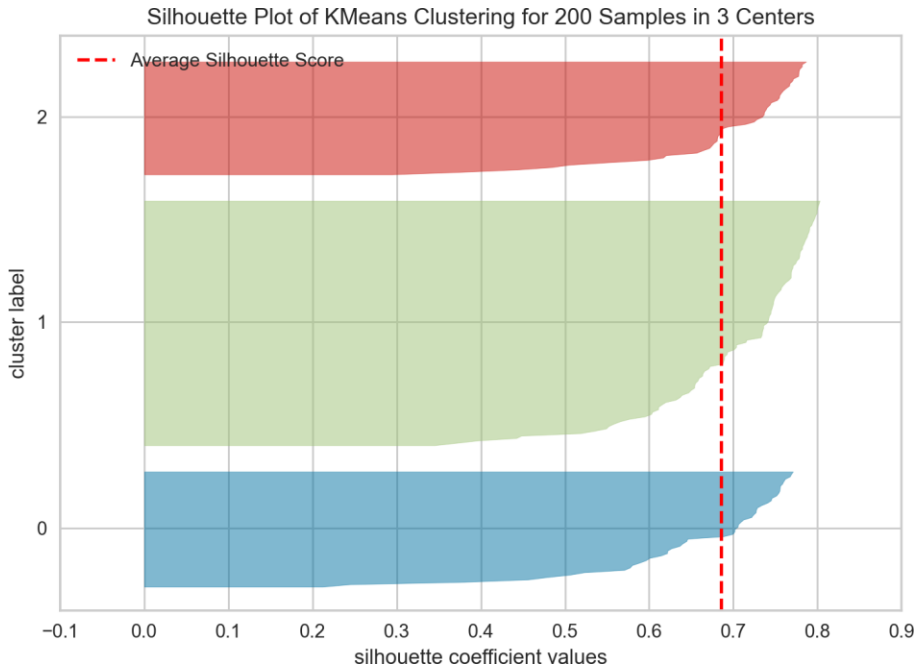
- Can use silhouettes as diagnostics of any method

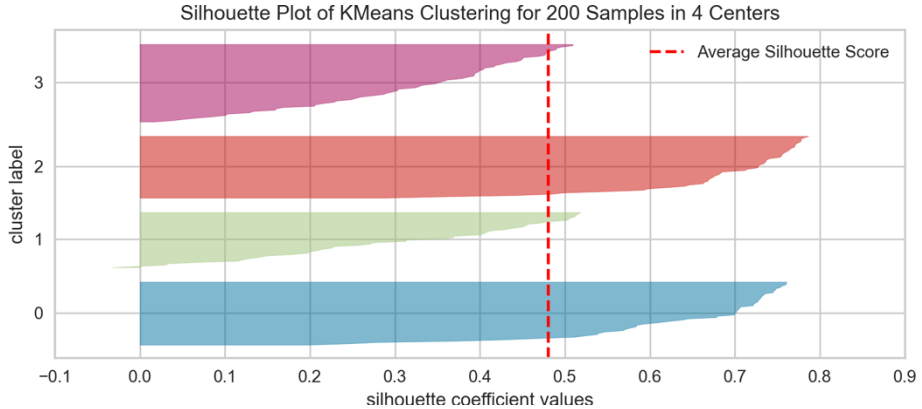
Comparing clusterings: K = 2

```
## []
```

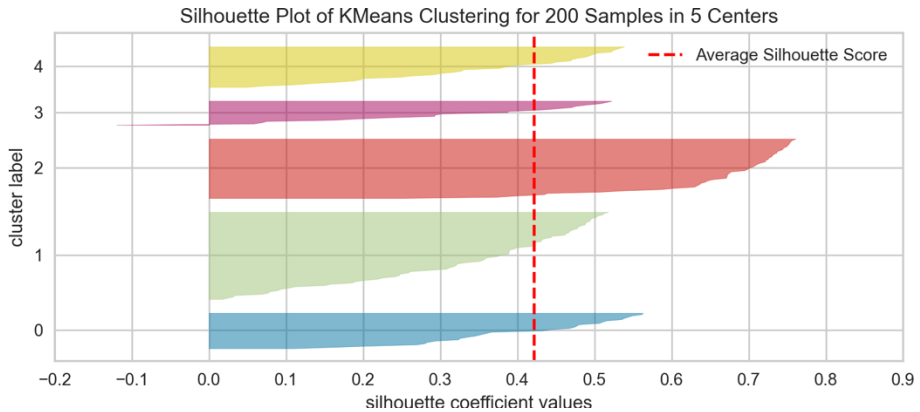


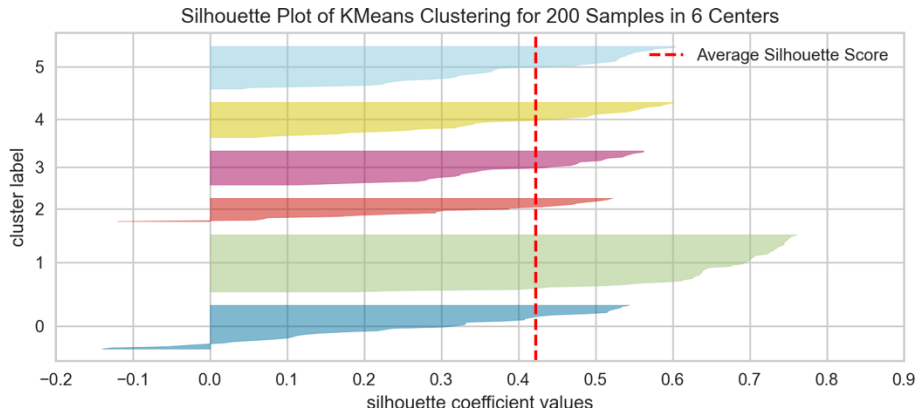
Comparing clusterings





Comparing clusterings





Average Silhouette

- A great clustering will have high silhouettes for all samples.
- To compare different values of K (and different methods), we can compute the average silhouette

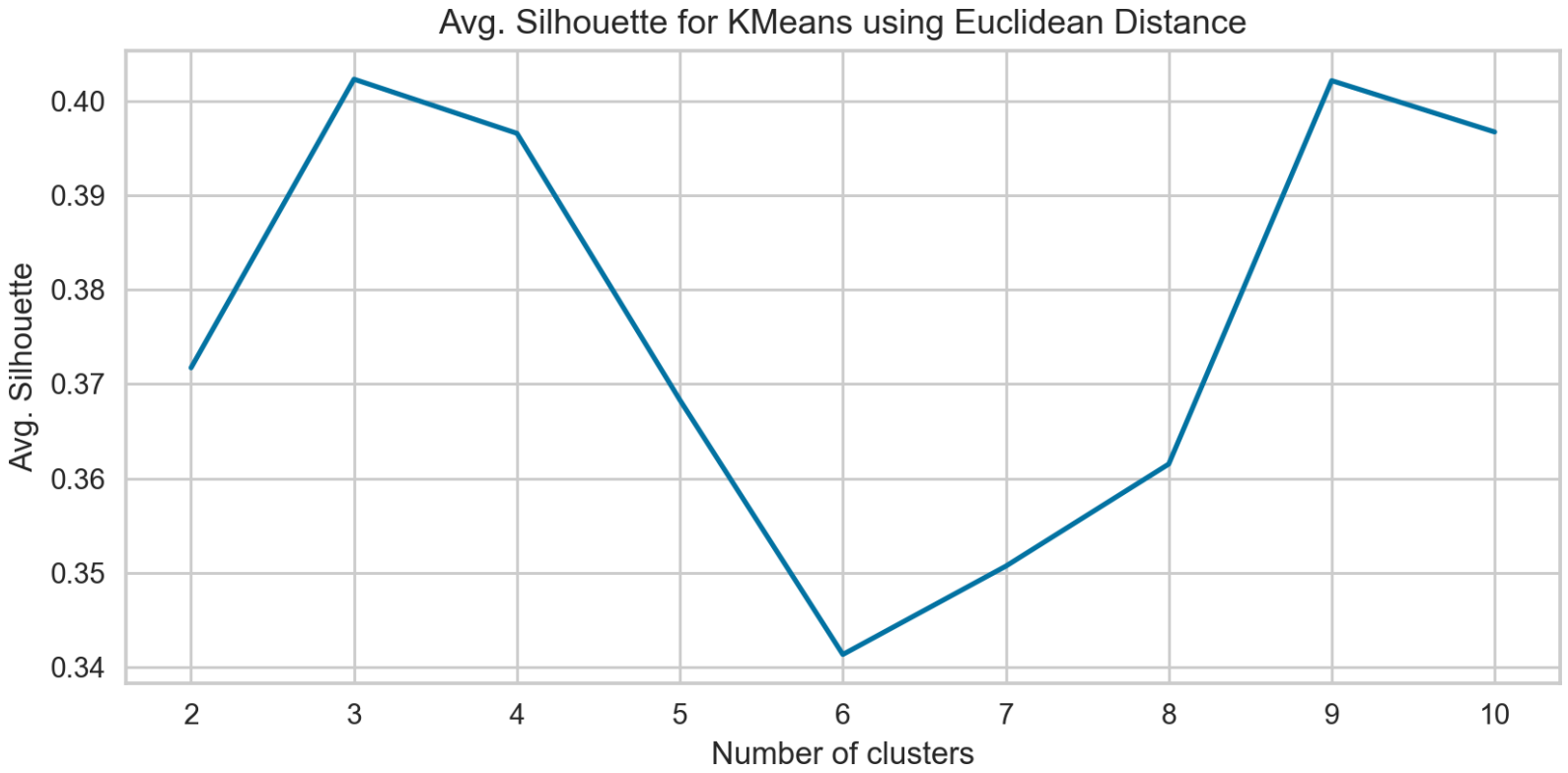
$$S_K = \frac{1}{N}\sum_{i=1}^{N} s_i$$

- over a range of values of K and choose the K which maximizes S_K .

- *sklearn* implements different measures of dissimilarity
  - Default is Euclidean distance

Average Silhouette

For the previous silhouette plots, we have the following average silhouettes for K-means after different values of K.



- Suggests $K = 3$ is the optimal number of clusters

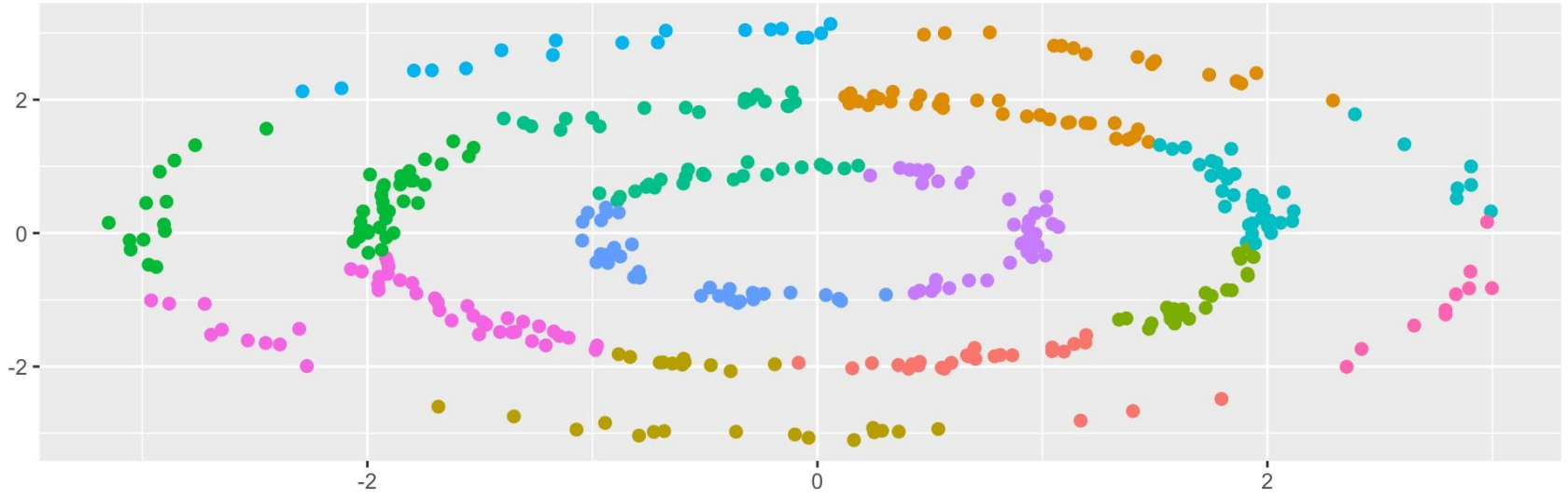
K-means sales pitch

- A (typically) fast and interpretable method for clustering with
  - clear notion of dissimilarity
  - simple implementation
  - cluster centers which can be interpreted as representative points of each cluster
  - methods for assessing the number (or existence) of clusters

- But there are other issues

Limitations (geometrical)

- Entirely dependent on squared Euclidean distance
  - Outliers and scale are problems
  - Standardization can address scale but not outliers
- Biased to clusters which are spherical in shape
  - Tends to overestimate number of clusters otherwise
- If (dis)similarity not best captured through squared Euclidean distance, poor clustering is inevitable

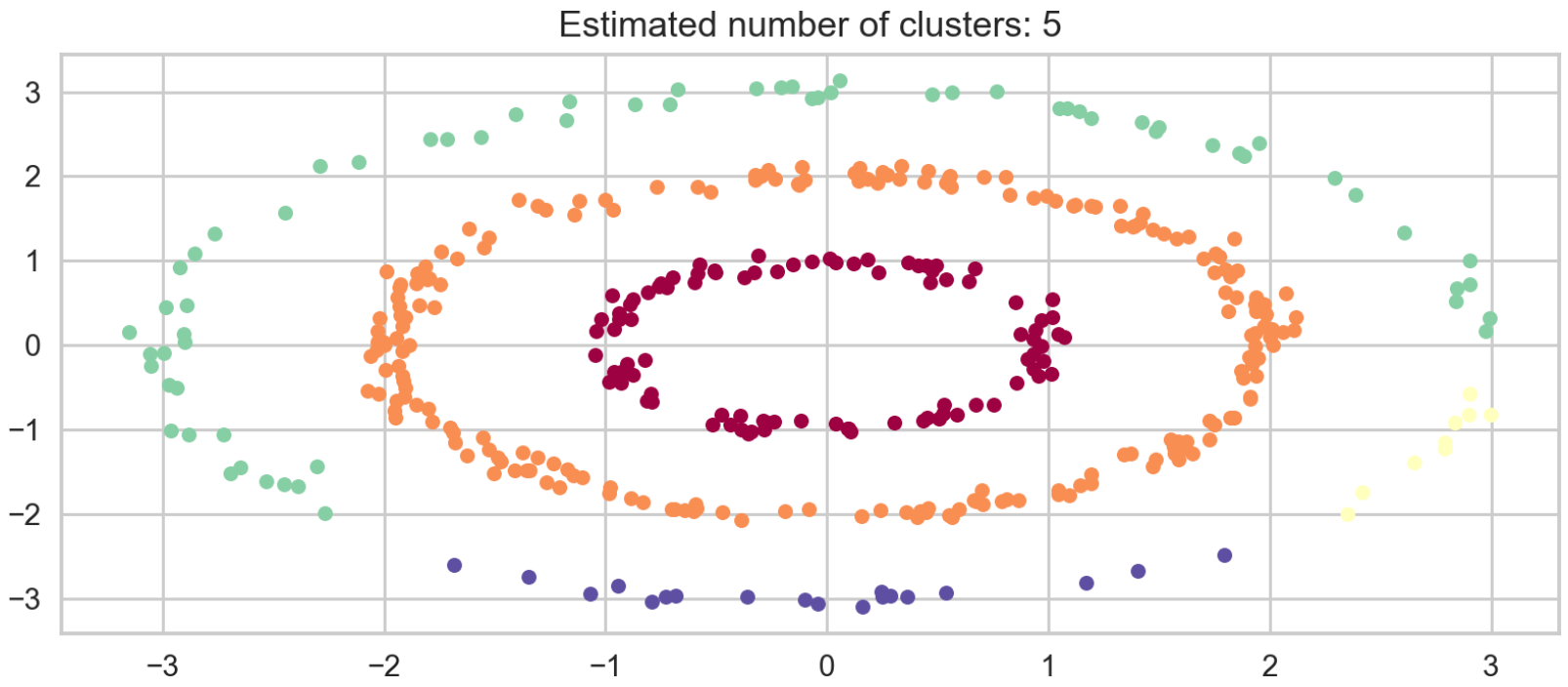


What are alternatives?

- If clusters are hyperelliptic in shape rather than spherical, we could consider Gaussian Mixture Models
- Employ a method which is agnostic to cluster shape
  - DBSCAN (and its extensions)
- Appeal to the Manifold Hypothesis
  - High dimensional data are concentrated on lower dimensional manifold(s)
  - Ideally: Use intrinsic distances on manifold(s) to compute dissimilarities
  - Practically: nonlinear dimension reduction followed by clustering
    - Ex: Spectral clustering (Laplacian Eigenmaps (almost) followed by k-means)

Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

A Density Based Algorithm for Discovering Clusters by Ester et al., 1996, KDD-96



Motivation

What is/are a cluster(s)?

- A cluster is a connected region with a high density of points
- Clusters are separated by regions with few or no points (low density regions)
- DBSCAN uses this idea to build clusters iteratively.

Strengths and weaknesses

Strengths

Unlike K-means, DBSCAN

- Can find any shape of clusters
  - Idea is to grow cluster by connecting points in high density regions then expanding outwards
- 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

Weaknesses

- Sensitive to its two tuning parameters
  - ε-maximal dissimilarity for two points to be considered neighbors
  - MinPts - minimum number neighbors (including itself) for a point to be considered "core"

Important definitions

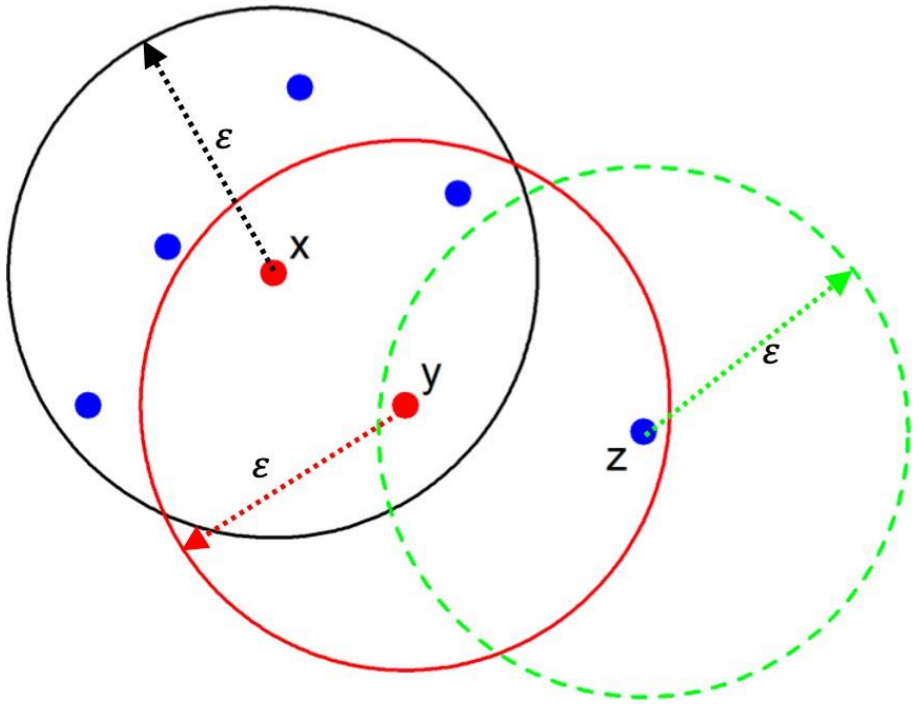
For a given choice of ϵ and MinPts

- Core points - observations with MinPts total observations within a distance/dissimilarity ε
- Border points - observations that are not core points, but are within ε of a core point
- Noise points - everything else

In the figure on the right

- x is a core point
- y is a border point
- z is a noise point

MinPts = 6



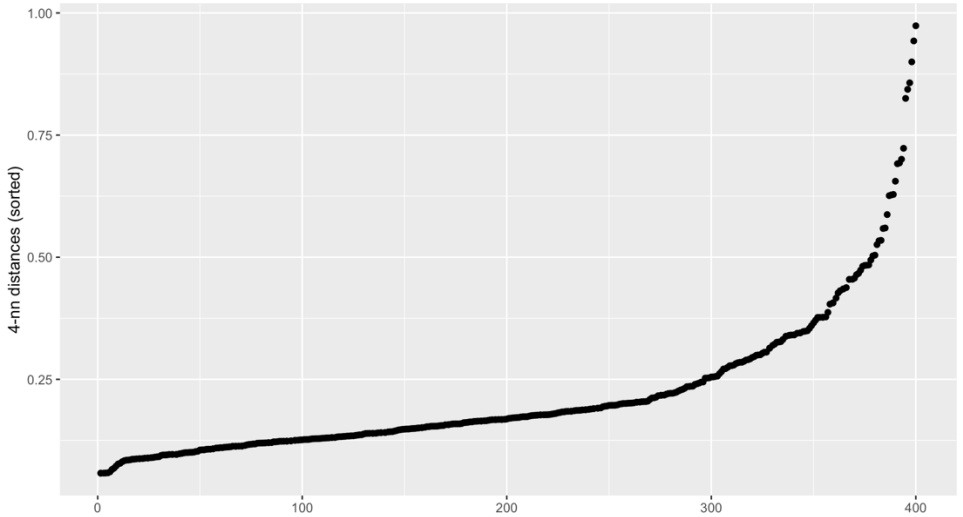
Crude (and slightly ill-posed) Algorithm

See original paper for more detailed and efficient implementation

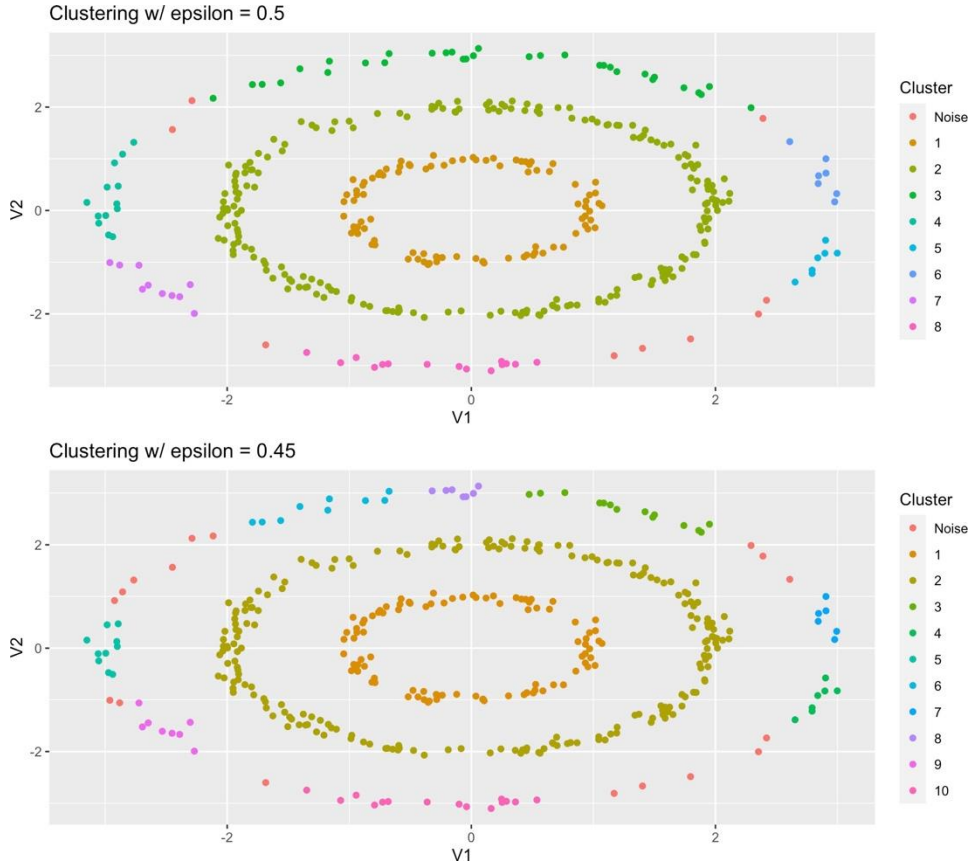
1. Build the neighborhood graph.
   - Compute all pairwise dissimilarities. Default is Euclidean distance, but free to specify any method you prefer.
   - Connect points with an edge if their dissimilarity is $\leq \epsilon$.
2. Identify core points: All points on graph with at least MinPts-1 edges.
3. Ignoring non-core points, find the connected components of neighborhood graph of core point.
4. Assign each non-core point to the cluster within distance ϵ.
   - For each such point, call it a border point
   - If no such cluster exists, call it a noise point

Example with Rings Data, MinPts = 5

Graph of nearest neighbor distance

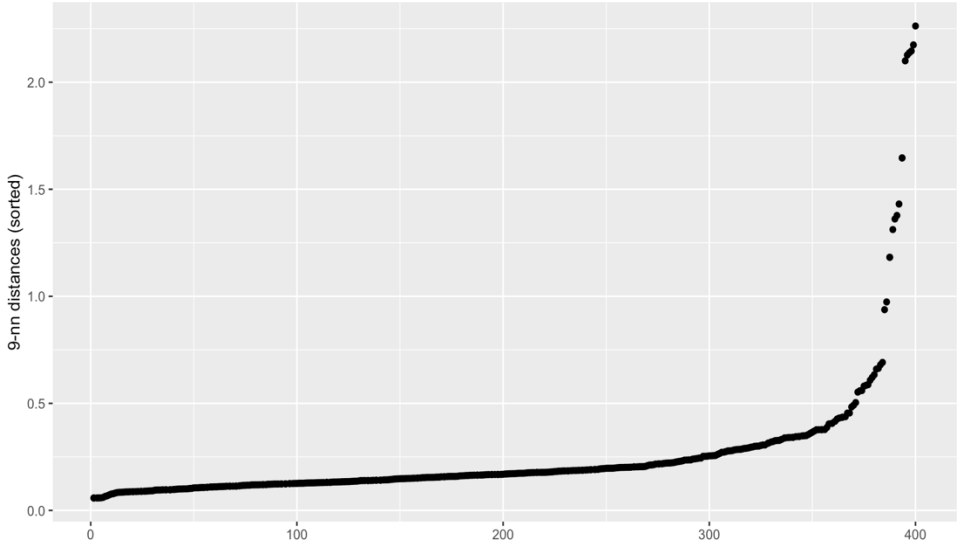


Sensitivity to ϵ

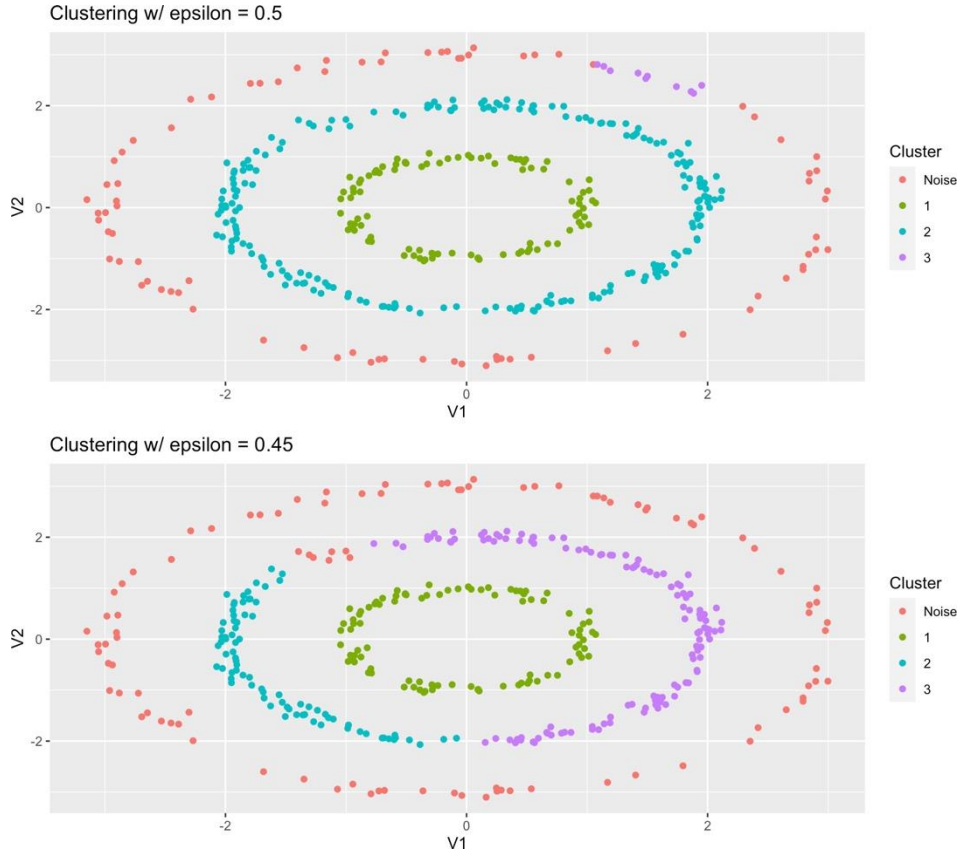


Example with Rings Data, MinPts = 10

Graph of nearest neighbor distance



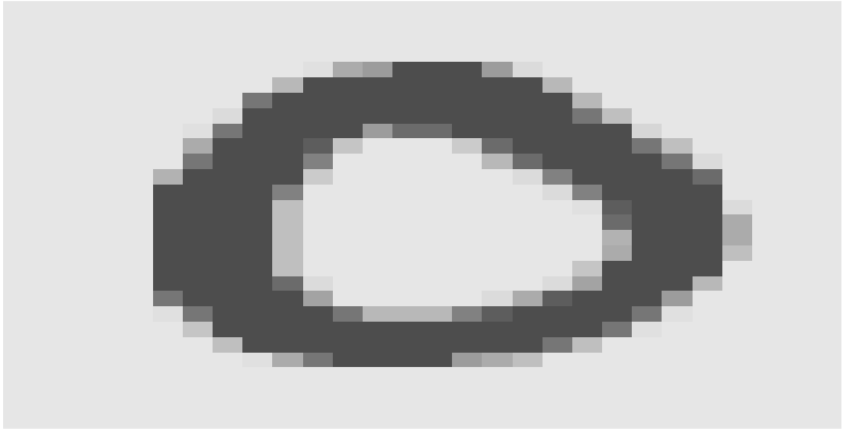
Sensitivity to ϵ



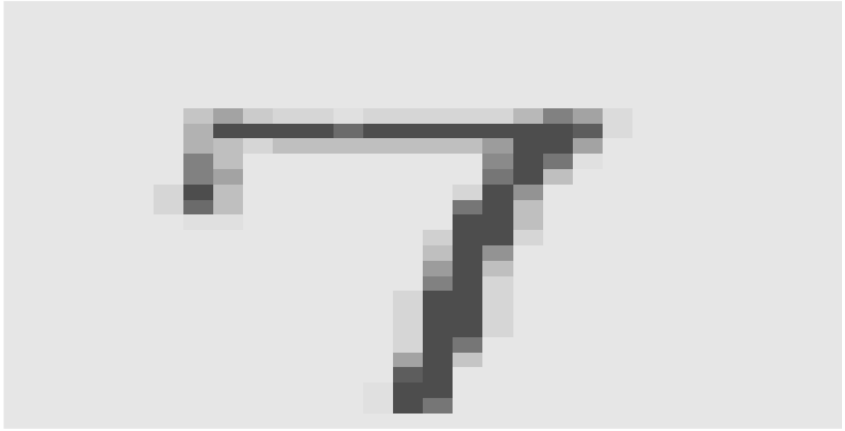
Example 1: MNIST

- The MNIST dataset contain 42000 grayscale images of handwritten digits (0-9) each at a 28×28 pixel resolution. We can view each image as a $28 \times 28 = 784$-dimensional vector.

0

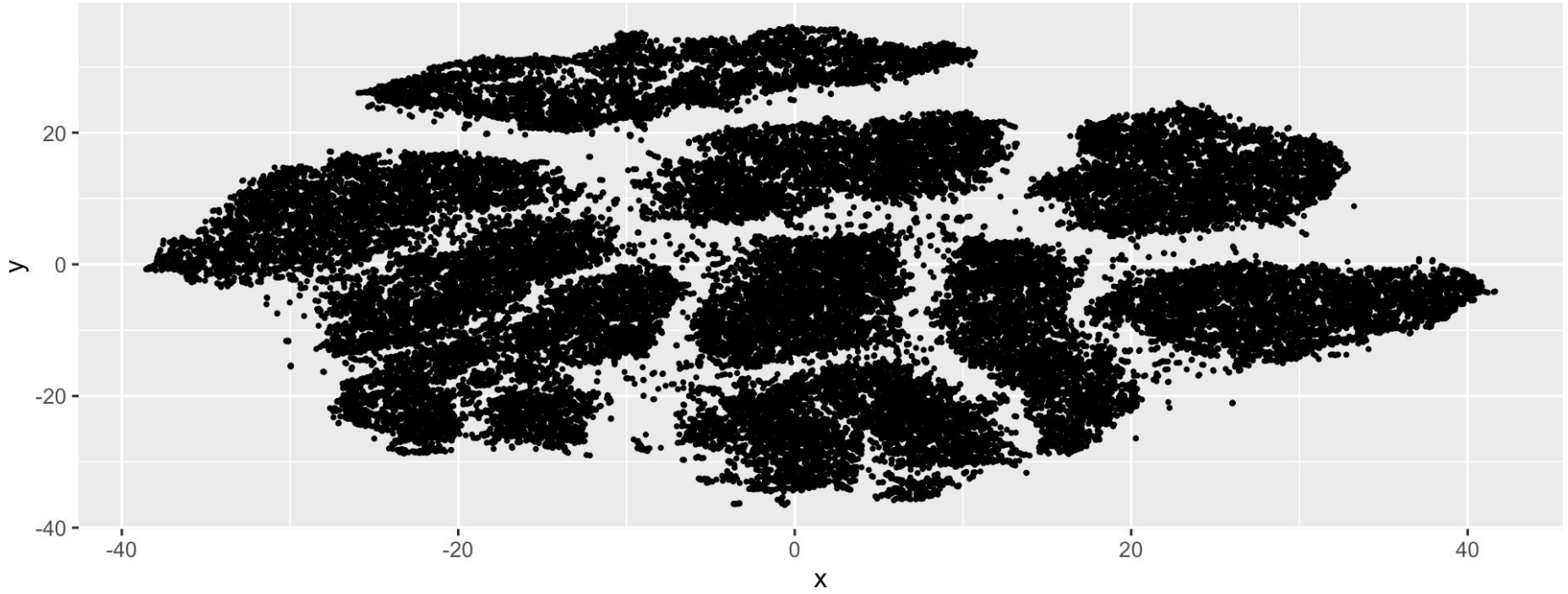


7



t-SNE: A Representation of Data

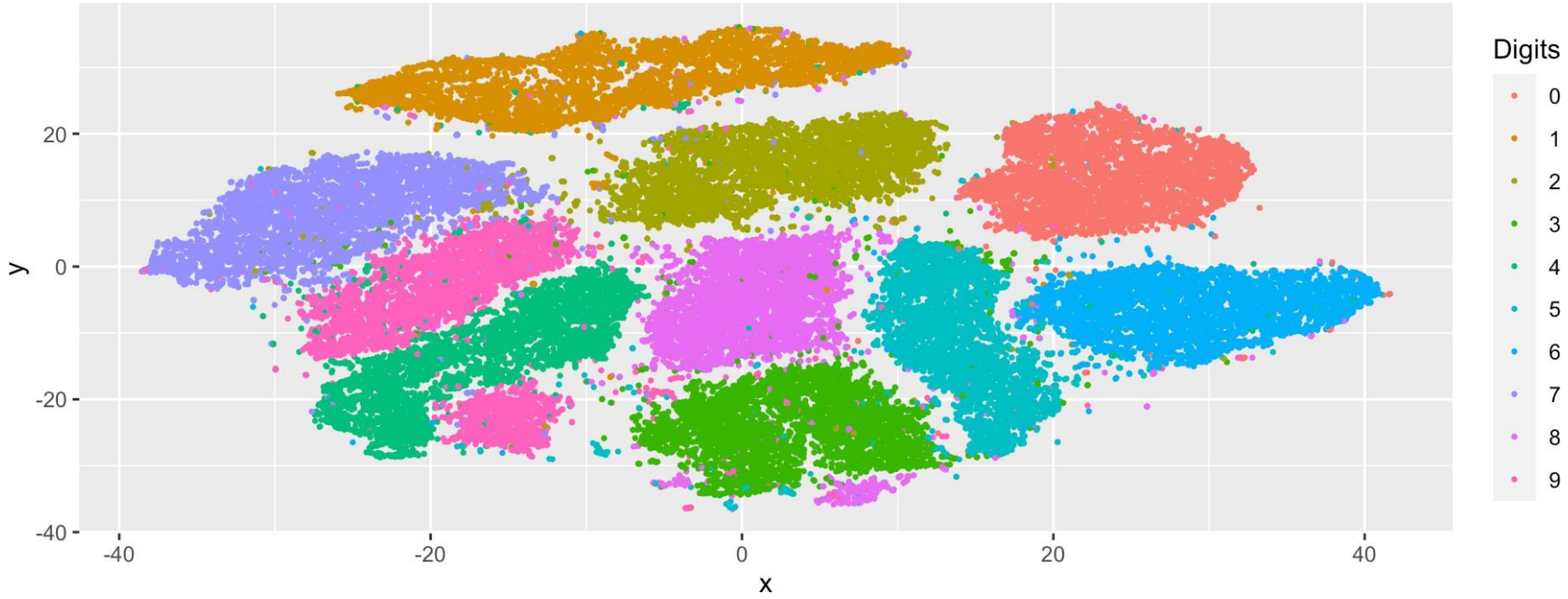
t-SNE is a method of nonlinear dimension reduction known for preserving clusters



t-SNE applied to subset of MNIST, perplexity = 50, iter = 1000

t-SNE: A Representation of Data

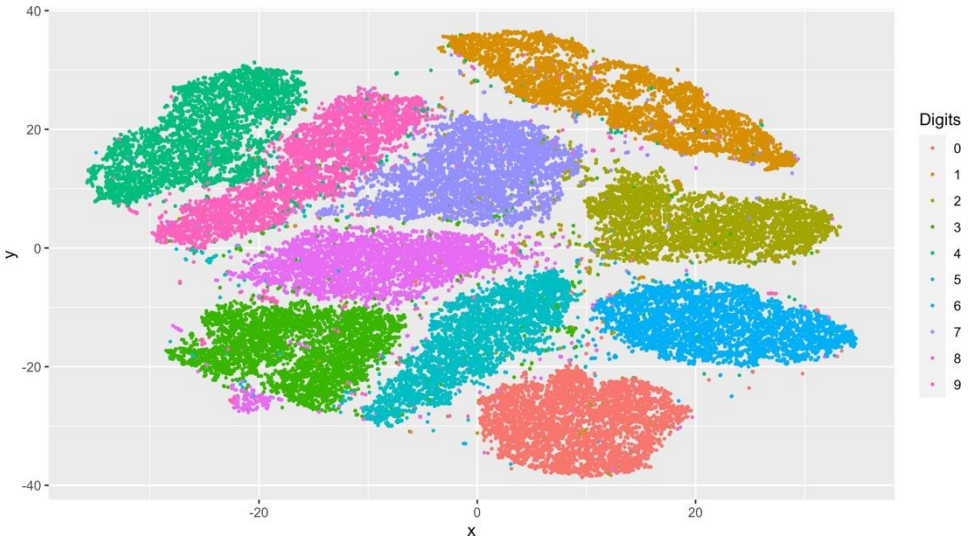
t-SNE is a method of nonlinear dimension reduction known for preserving clusters



t-SNE applied to subset of MNIST, colored by label

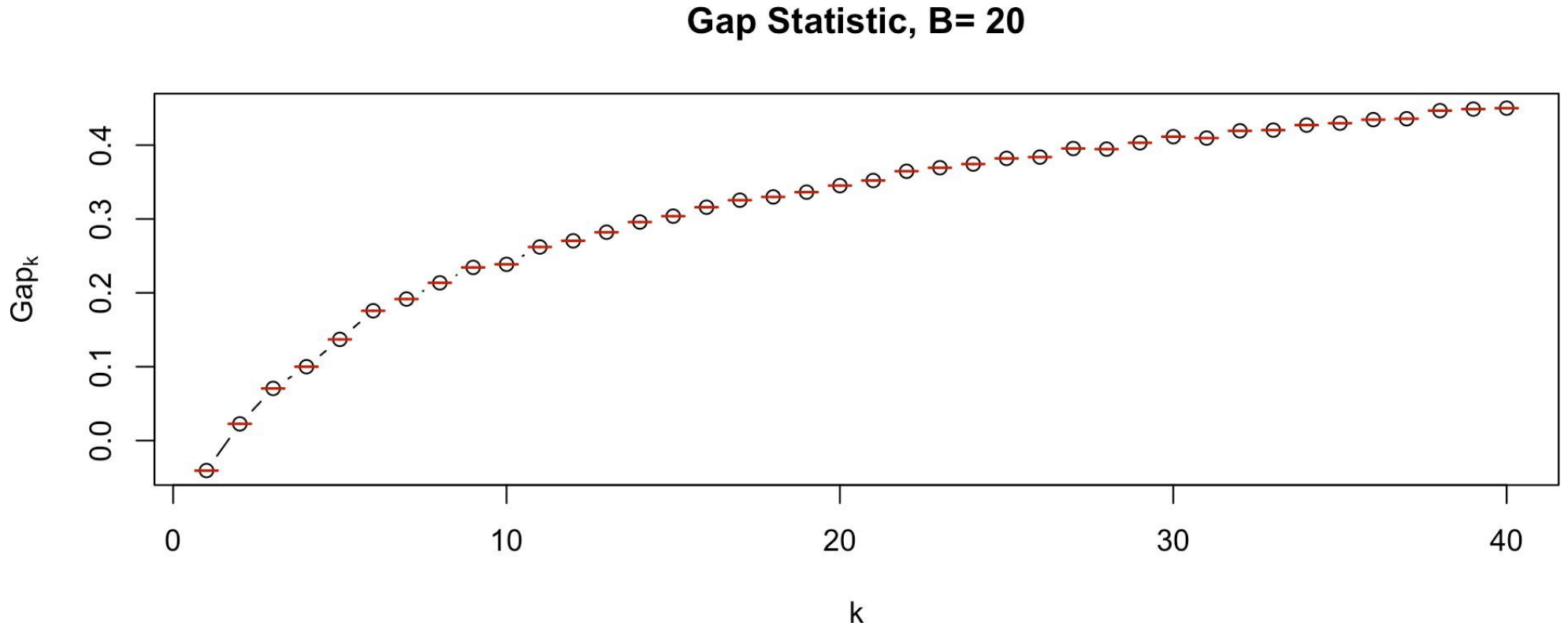
Preprocessing choices

- Grayscale colors of the pixels are coded as digits {0, … , 255}.
  - Pixels near the outer edges of the images are never written on, thus always showing white (0).
  - There are 76 such columns in the data matrix, which we drop leaving 708.
- We rescale the remaining columns by dividing by 255
  - Each variable in the data matrix now lies in the interval [0, 1].
- We'll apply clustering algorithms to the preprocessed data then color points in the t-SNE plot based on results.



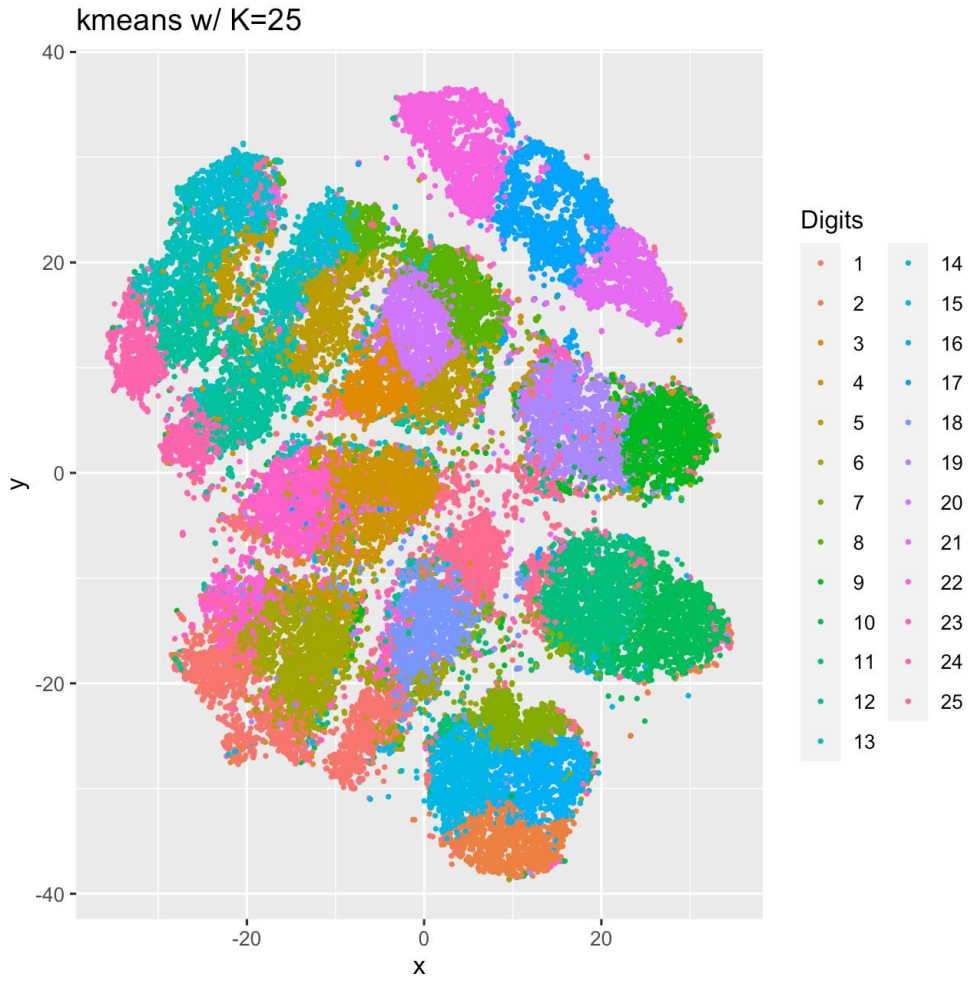
Updated t-SNE after preprocessing, perplexity = 50, iter = 1000

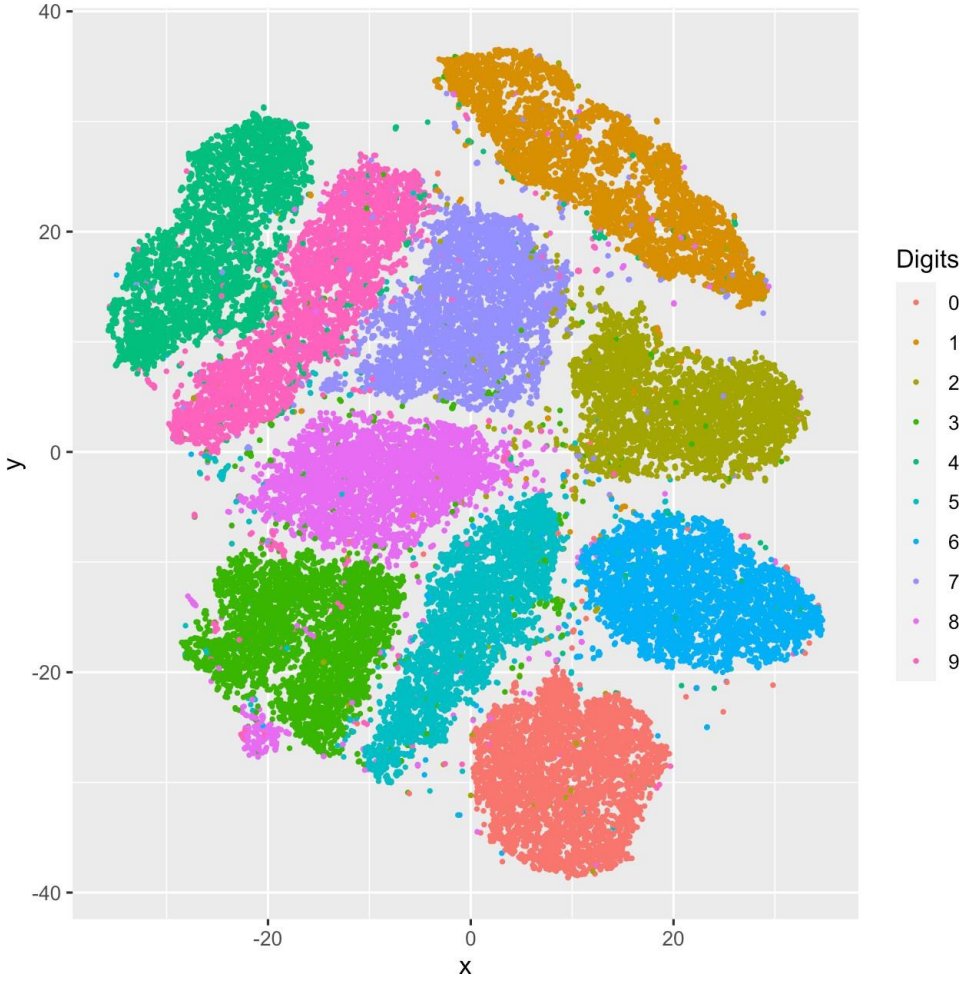
K-means: tuning number of clusters



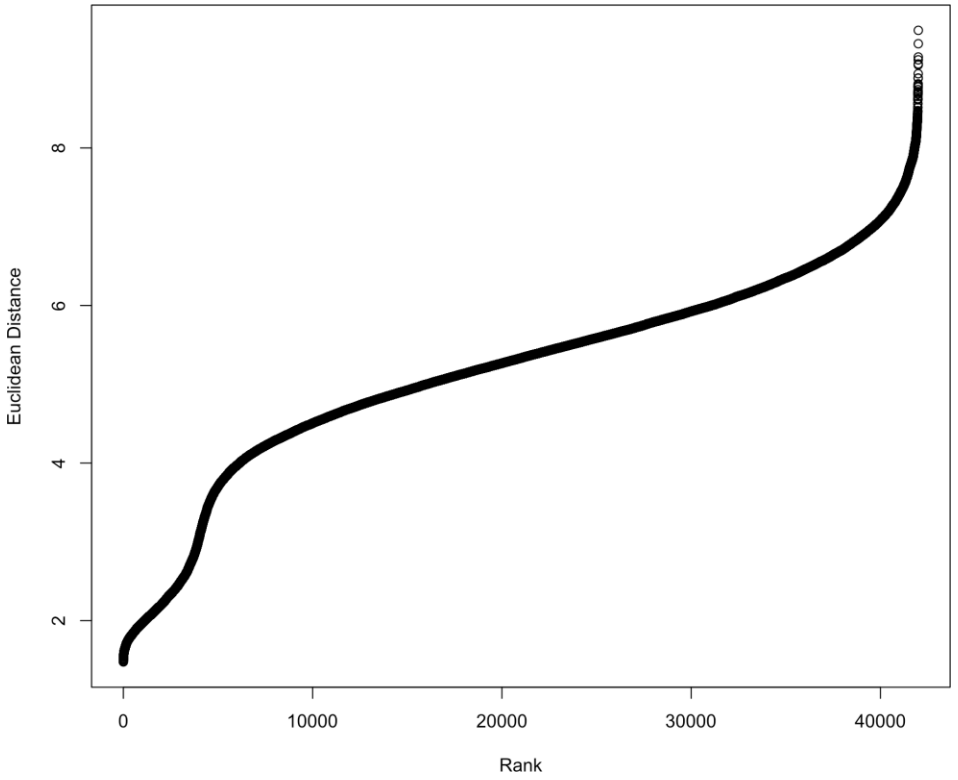
In the version shown in lecture, K = 14 was optimal. But this run of Gap statistic suggests 25 due to sampling variability.

K-means clustering

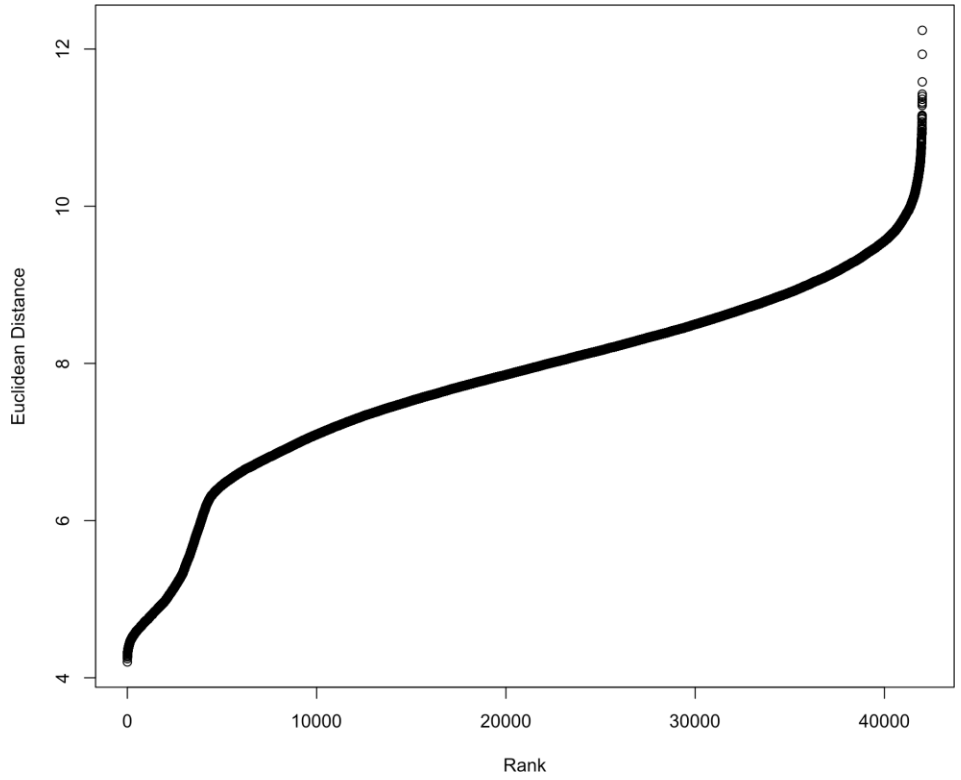




DBSCAN

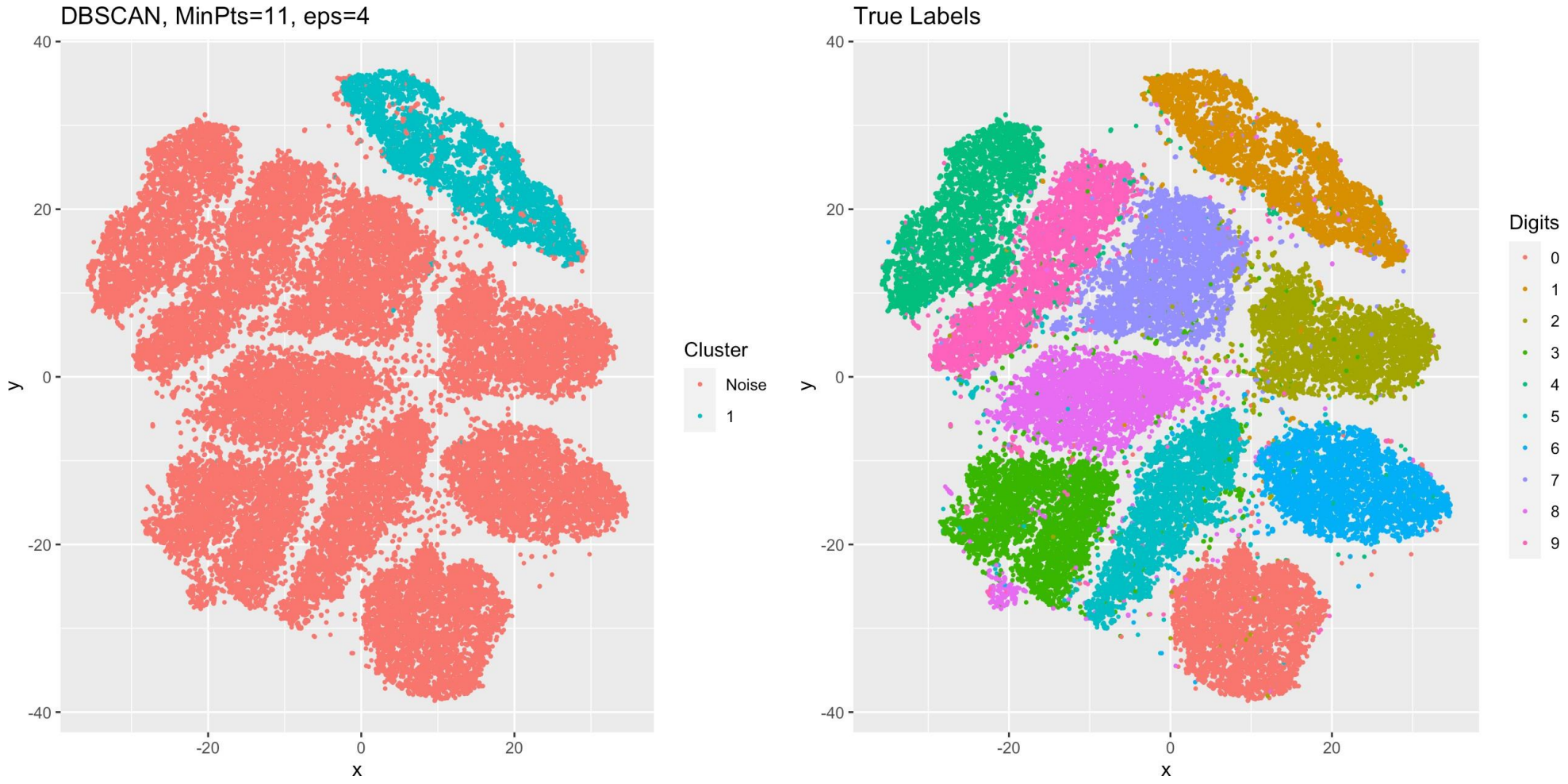


Sorted 10th nearest neighbor distances



Sorted 1300th nearest neighbor distances

DBSCAN clustering



Example 2: NCI Microarray Data (Elements of Statistical Learning, Ch. 14)

- This data set contains gene expression levels in cancer cells
  - 64 cancer cells of different types
  - 6830 genes

- We'll assume that the relative expression level is important for distinguish
  - Use cosine dissimilarity
  - Can input this into DBSCAN and t-SNE

Example 2: NCI Microarray data

After clustering: clustered regression

Clustering can be used for further analysis

Clustered regression:

- Clustering partitions the data into meaningful groups
- Regression models are fitted separately for each group
- Captures varying relationships across different groups

Key Benefits:

- Enhanced predictive performance (higher R^2)
- Better understanding of group-specific dynamics

Example: Boston housing dataset

Consider Boston housing dataset with 506 observations

Target variable

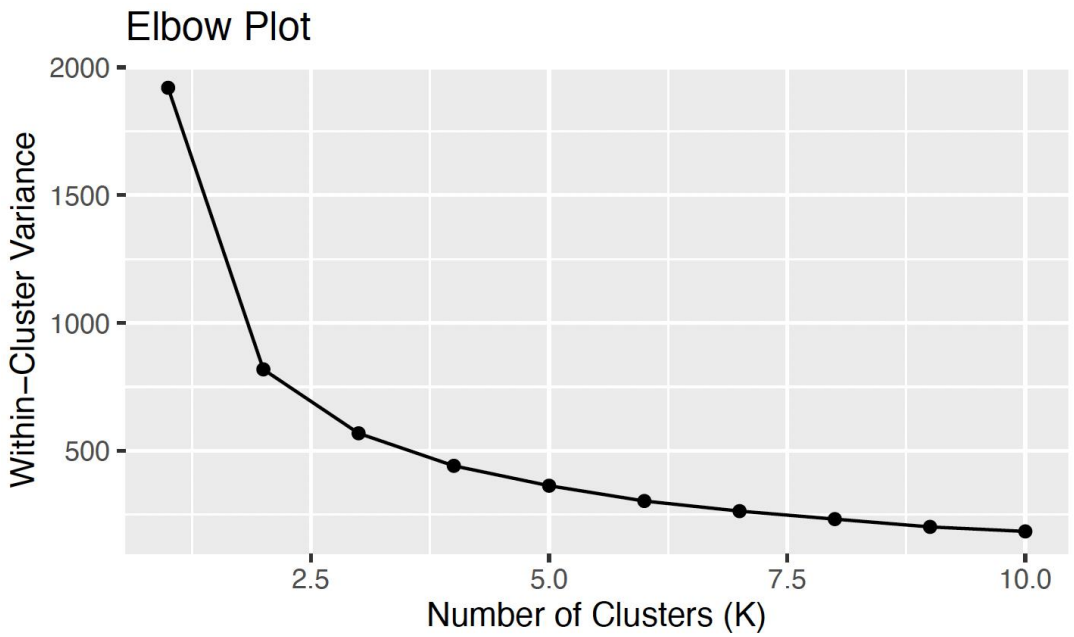
- *medv*: median value of owner-occupied housings (in $1000s)

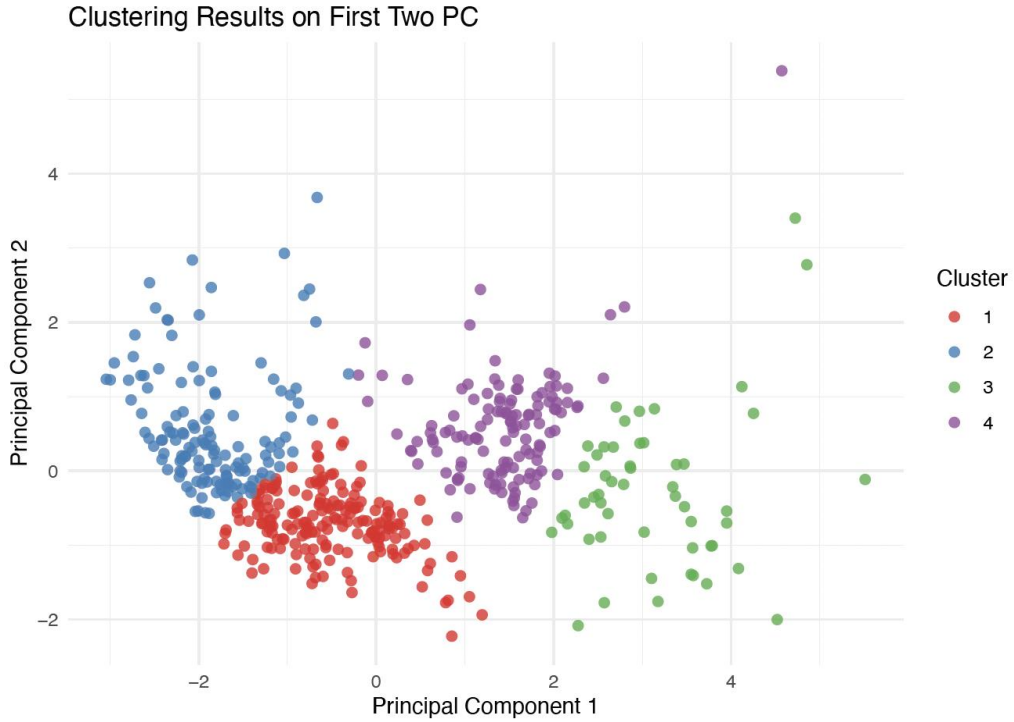
Predictors:

- *crim*: crime rate
- *nox*: levels of air pollution
- *rm*: number of rooms
- *lstat*: lower Status Population Percentage
- *indus*: Industrial Proportion

Example: Boston housing dataset

- K-means clustering based on first two principal components
- Select K=4 using elbow plot





Example: Boston housing dataset

- Improvement in R^2 by clustered regression: 0.6466 → 0.8176
- Sign of different relationships for different clusters