

DATA MINING CLUSTERING

The k-means algorithm

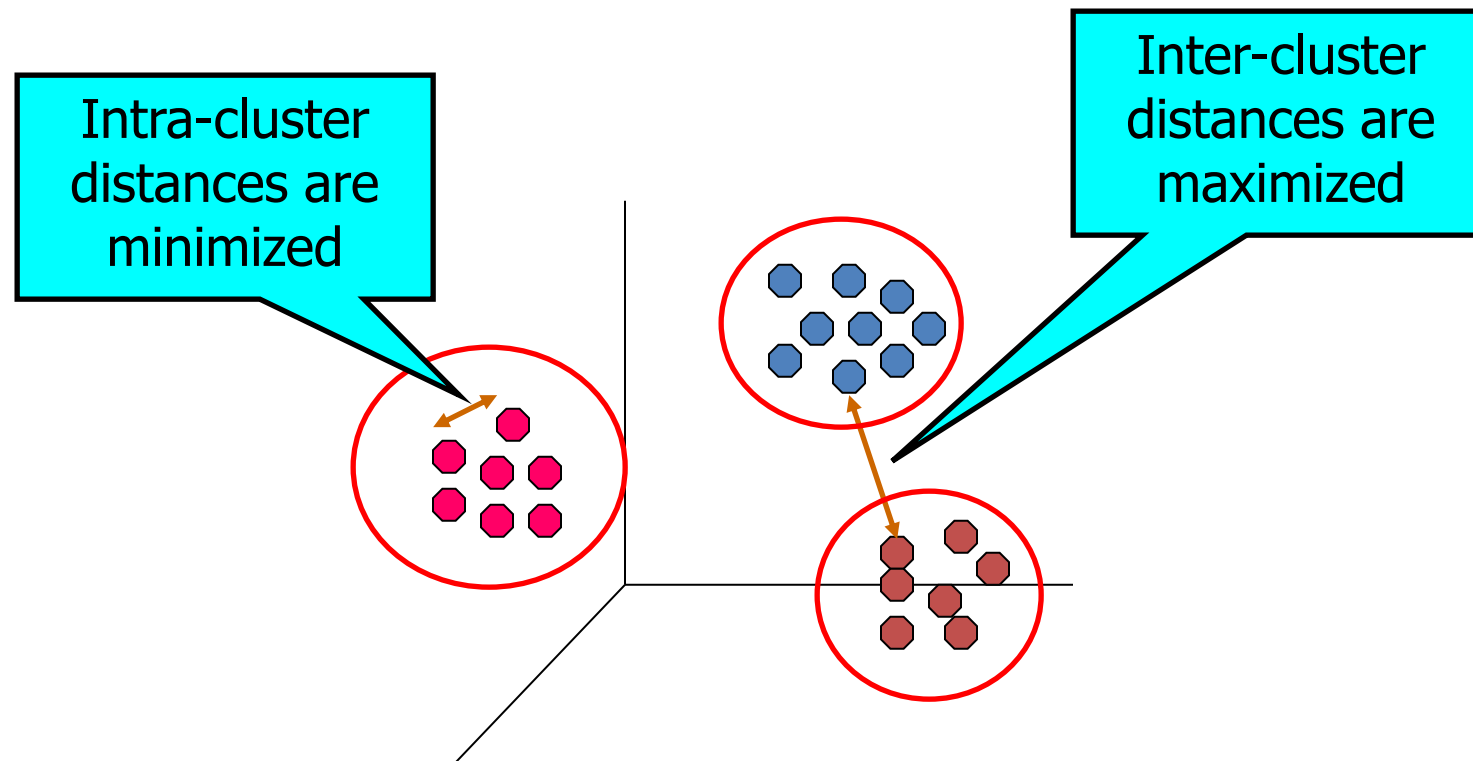
Hierarchical Clustering

The DBSCAN algorithm

Evaluation

What is a Clustering?

A **grouping** of objects such that the objects in a **group** (**cluster**) are similar (or related) to one another and different from (or unrelated to) the objects in other groups (clusters)



Why Cluster Analysis

- **Understanding**

- **Group** related **documents** for browsing, **genes and proteins** that have similar functionality, **stocks** with similar price fluctuations, **users** with same behavior

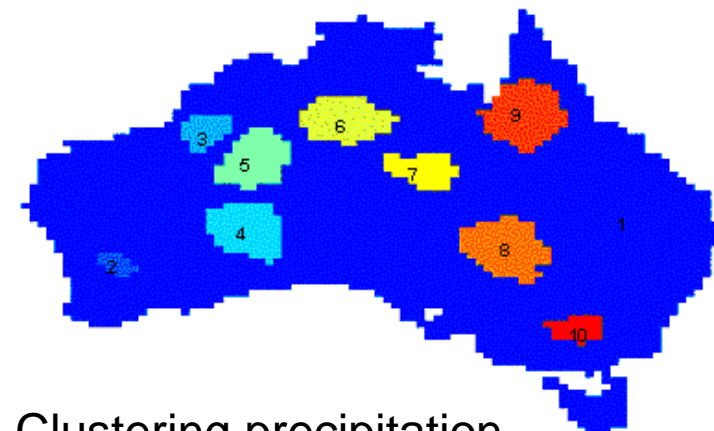
- **Summarization**

- Reduce the size of large data sets

- **Applications**

- Recommendation systems
- Search Personalization

| | <i>Discovered Clusters</i> | <i>Industry Group</i> |
|----------|---|-----------------------|
| 1 | Applied-Matl-DOWN,Bay-Network-DOWN,3-COM-DOWN, Cabletron-Sys-DOWN,CISCO-DOWN,HP-DOWN, DSC-Comm-DOWN,INTEL-DOWN,LSI-Logic-DOWN, Micron-Tech-DOWN,Texas-Inst-DOWN,Tellabs-Inc-Down, Natl-Semiconduct-DOWN,Oracl-DOWN,SGI-DOWN, Sun-DOWN | Technology1-DOWN |
| 2 | Apple-Comp-DOWN,Autodesk-DOWN,DEC-DOWN, ADV-Micro-Device-DOWN,Andrew-Corp-DOWN, Computer-Assoc-DOWN,Circuit-City-DOWN, Compaq-DOWN, EMC-Corp-DOWN, Gen-Inst-DOWN, Motorola-DOWN,Microsoft-DOWN,Scientific-Atl-DOWN | Technology2-DOWN |
| 3 | Fannie-Mae-DOWN,Fed-Home-Loan-DOWN, MBNA-Corp-DOWN,Morgan-Stanley-DOWN | Financial-DOWN |
| 4 | Baker-Hughes-UP,Dresser-Inds-UP,Halliburton-HLD-UP, Louisiana-Land-UP,Phillips-Petro-UP,Unocal-UP, Schlumberger-UP | Oil-UP |



Clustering precipitation in Australia

Early applications of cluster analysis

- John Snow, London 1854

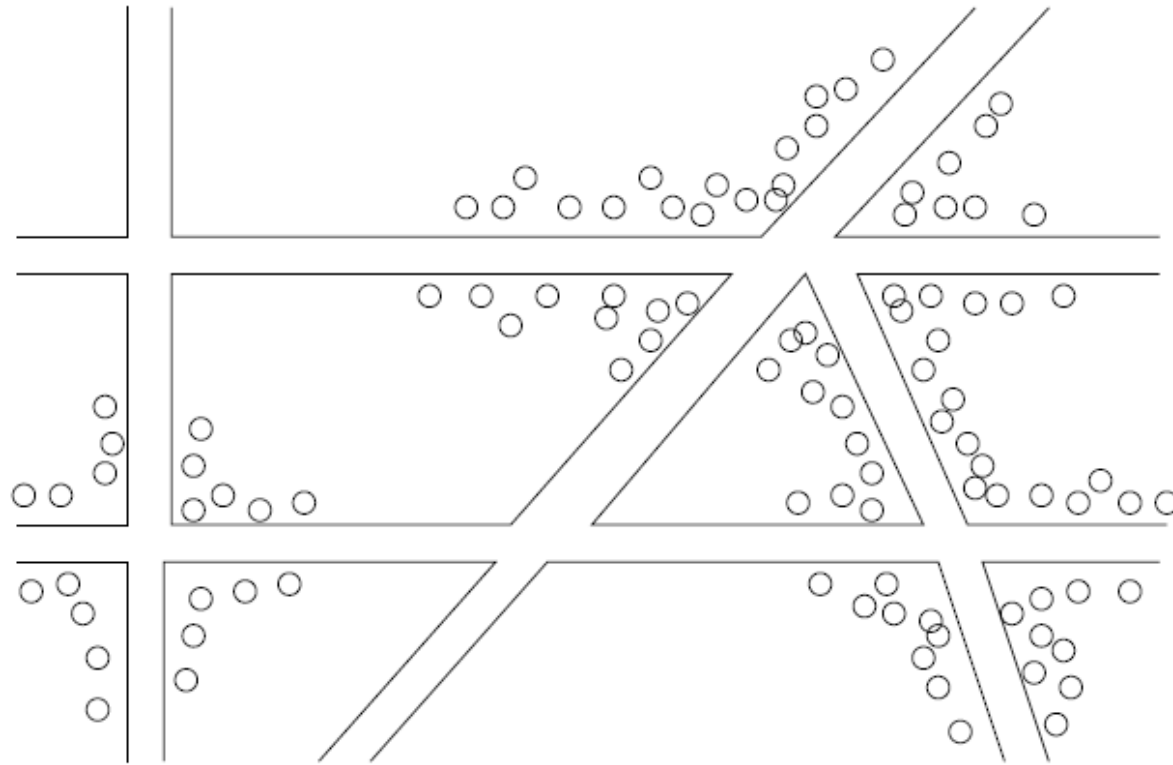
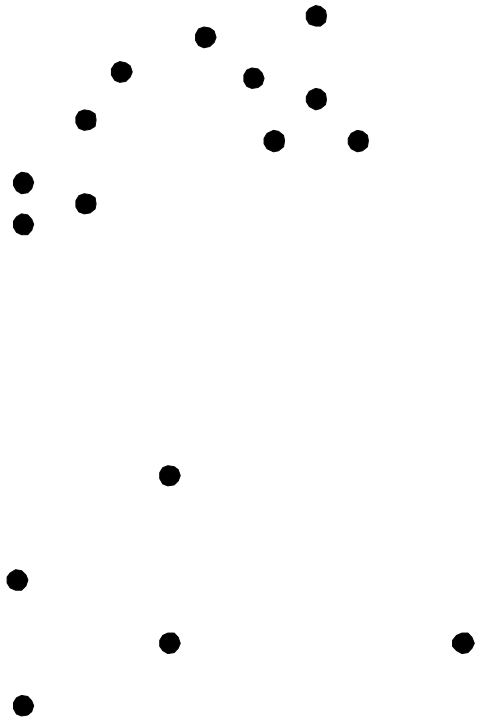


Figure 1.1: Plotting cholera cases on a map of London

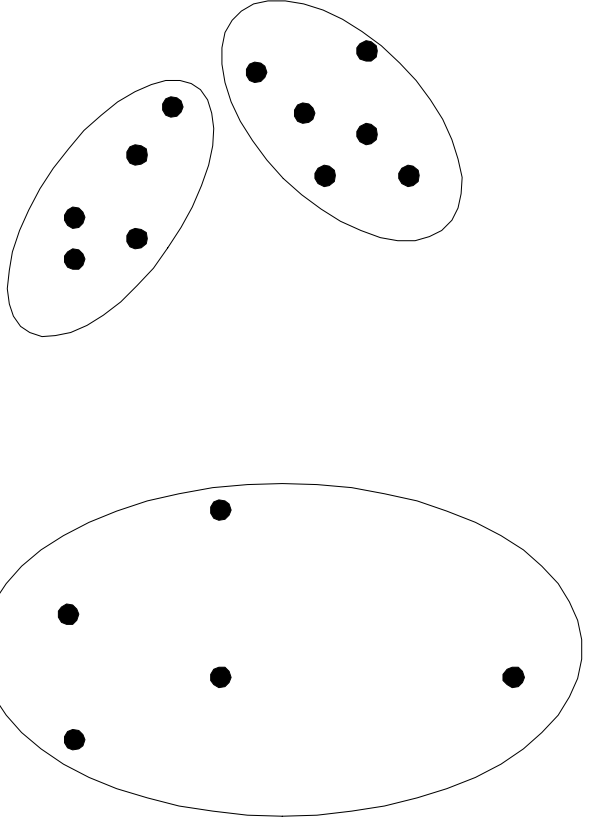
Types of Clusterings

- Important distinction between **hierarchical** and **partitional** sets of clusters
- **Partitional** Clustering
 - A division data objects into subsets (**clusters**) such that each data object is in exactly one subset
- **Hierarchical** clustering
 - A set of nested clusters organized as a hierarchical tree

Partitional Clustering

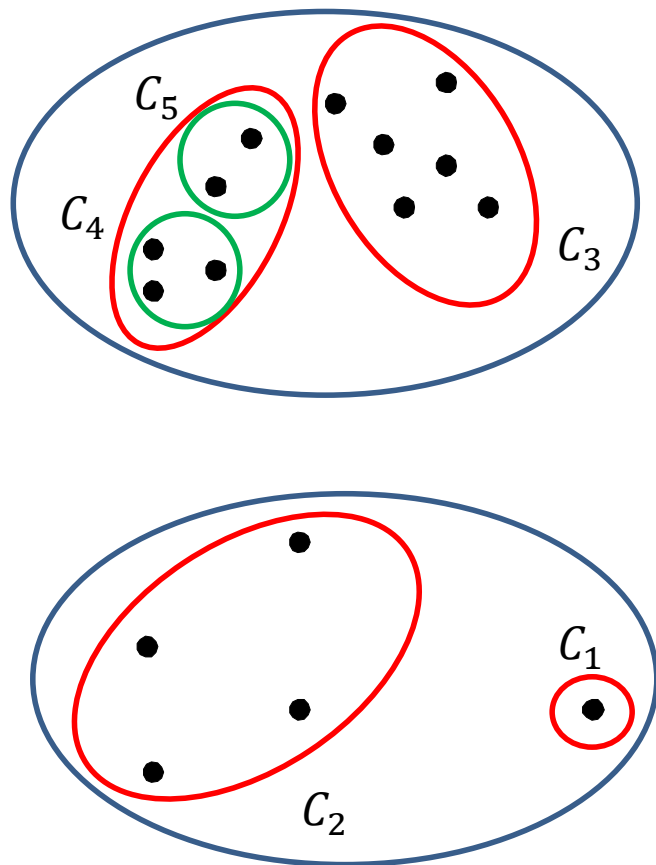


Original Points

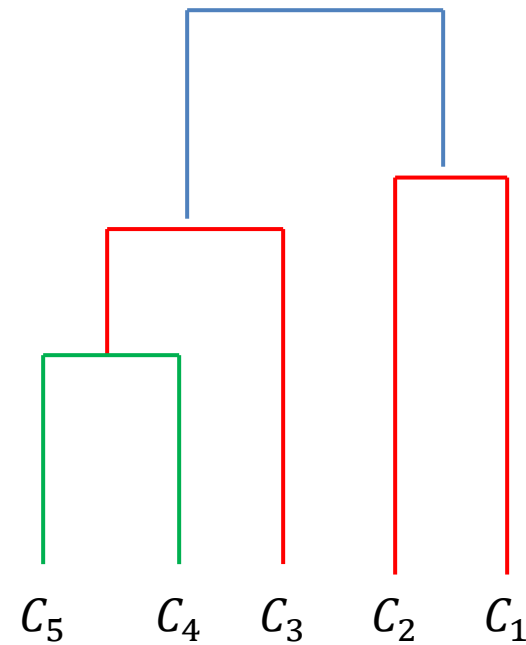


A Partitional Clustering

Hierarchical Clustering



Hierarchical Clustering



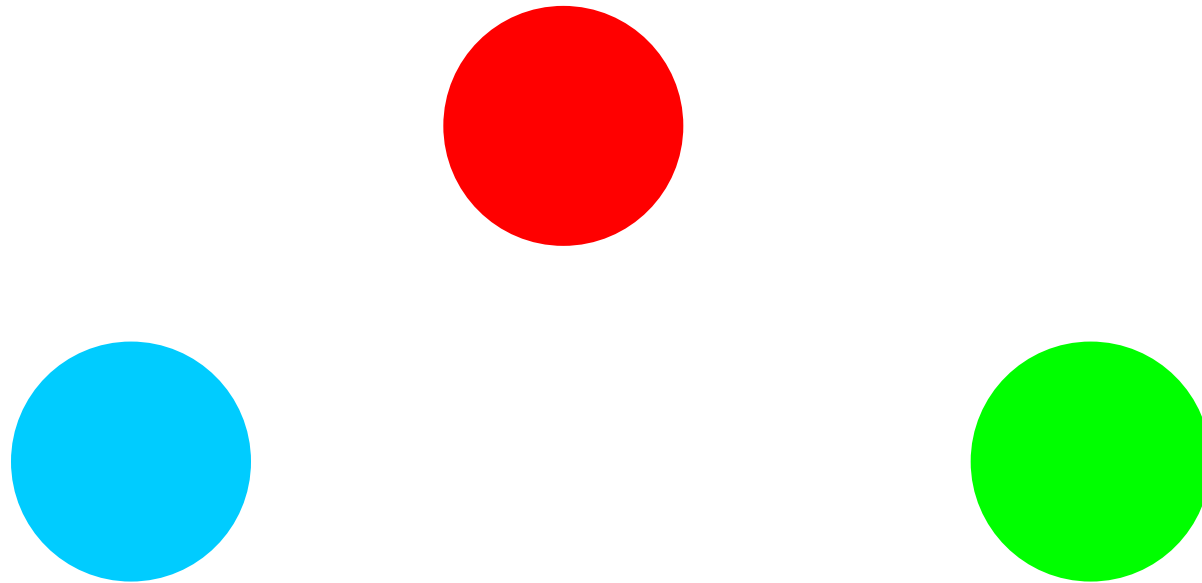
Hierarchical Clustering dendrogram

Other types of clustering

- **Exclusive** (or **non-overlapping**) versus **non-exclusive** (or **overlapping**)
 - In non-exclusive clusterings, points may belong to multiple clusters.
 - Points that belong to multiple classes, or 'border' points
- **Fuzzy** (or **soft**) versus **non-fuzzy** (or **hard**)
 - In fuzzy clustering, a point belongs to every cluster with some weight between 0 and 1
 - Weights usually must sum to 1 (often interpreted as **probabilities**)
- **Partial** versus **complete**
 - In some cases, we only want to cluster some of the data

Clustering objectives

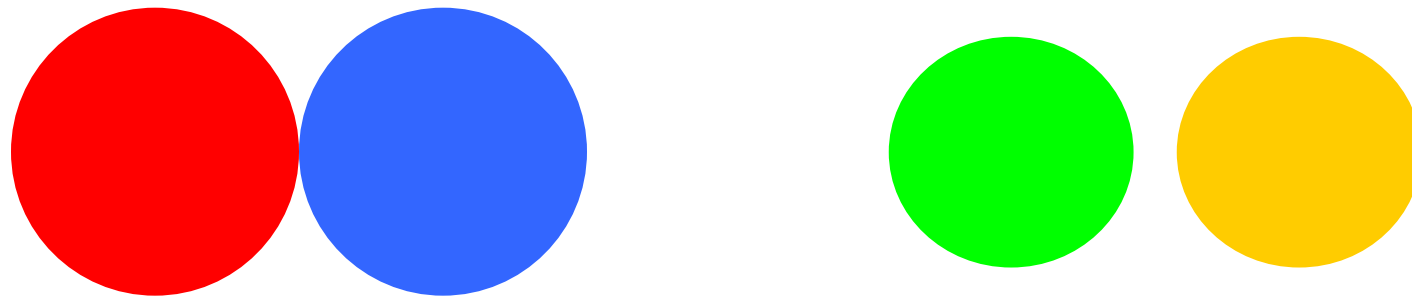
- **Well-Separated Clusters:**
 - A cluster is a set of points such that any point in a cluster is closer (or more similar) to every other point in the cluster than to any point not in the cluster.



3 well-separated clusters

Clustering objectives

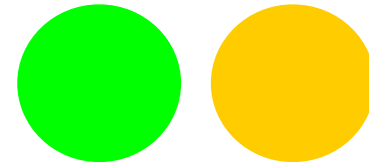
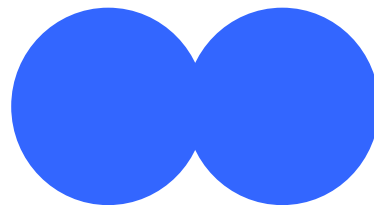
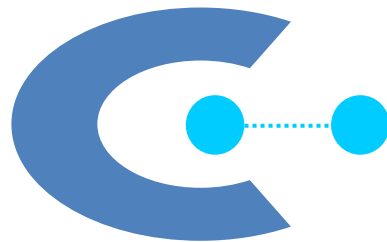
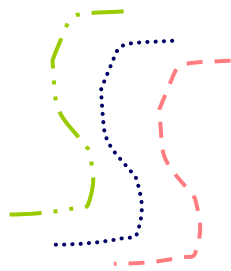
- Center-based Clusters:
 - A cluster is a set of objects such that an object in a cluster is **closer** (more **similar**) to the “center” of a cluster, than to the center of any other cluster
 - The center of a cluster is often a **centroid**, the minimizer of distances from all the points in the cluster, or a **medoid**, the most “representative” point of a cluster



4 center-based clusters

Clustering objectives

- **Contiguous Clusters** (Nearest neighbor or Transitive)
 - A cluster is a set of points such that a point in a cluster is closer (or more similar) to one or more other points in the cluster than to any point not in the cluster.

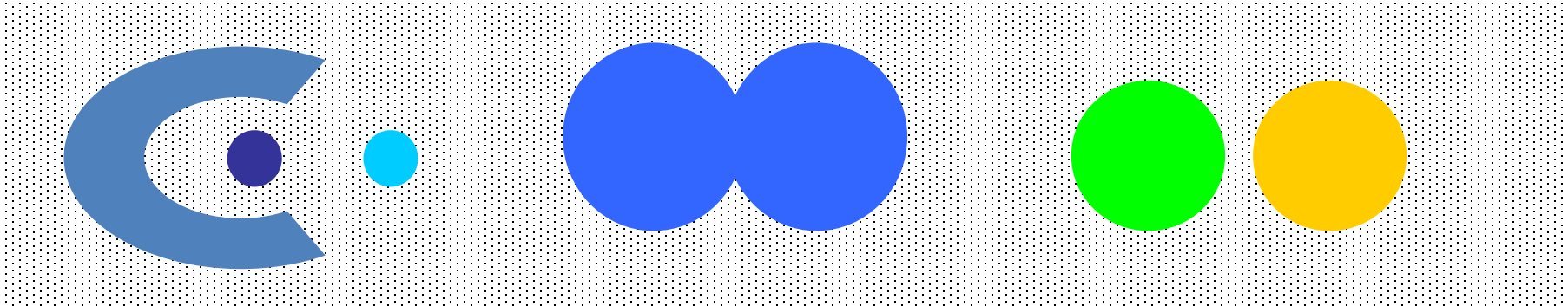


8 contiguous clusters

Clustering Objectives

- Density-based clusters

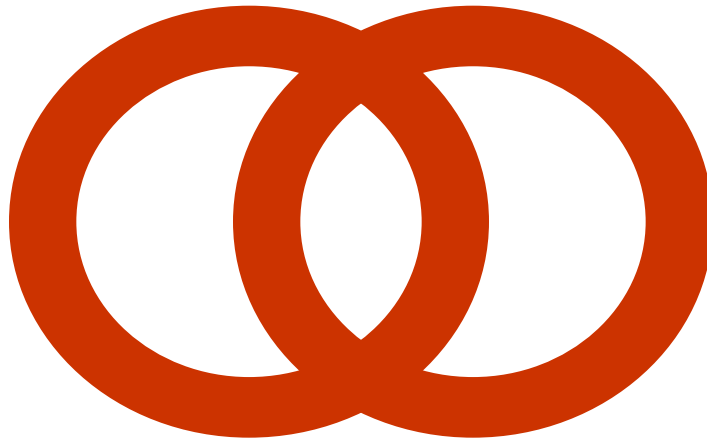
- A cluster is a dense region of points, which is separated by low-density regions, from other regions of high density.
- Used when the clusters are irregular or intertwined, and when noise and outliers are present.



6 density-based clusters

Clustering objectives

- **Shared Property or Conceptual Clusters**
 - Finds clusters that share some common property or represent a particular concept.



A cluster is defined as a set of points that lie on a circle

Clustering objectives

- Clustering as an **optimization problem**
 - Finds clusters that minimize or maximize an **objective function**.
 - Consider all possible ways of dividing the points into clusters and compute the '**goodness**' of each clustering using the objective function to find the best one.
 - Usually, finding the best is NP-hard (no polynomial algorithm).
 - Can have **global** or **local** objectives.
 - Hierarchical clustering algorithms typically have local objectives
 - Partitional algorithms typically have global objectives
 - A variation of the global objective function approach is to **fit** the data to a **parameterized (probabilistic) model**.
 - The **parameters** for the model are determined from the data, and they determine the clustering
 - E.g., **Mixture models** assume that the data is a 'mixture' of a number of statistical distributions.

Clustering Algorithms

- K-means and its variants
- Hierarchical clustering
- DBSCAN

K-MEANS

K-means Clustering

- Partitional clustering approach
- Each cluster is associated with a **centroid** (center point)
- Each point is assigned to the cluster with the **closest** centroid
- Number of clusters, **K**, must be specified
- The **objective** is to:
 - find **K centroids** and
 - the **assignment** of **points to clusters/centroids**
 - so as to **minimize the sum of distances** of the points to their respective **centroid**

K-means Clustering as an optimization problem

- **Problem:** Given a set X of n objects and an integer K , find a grouping of the points into K clusters $C = \{C_1, C_2, \dots, C_K\}$ with centroids $\{c_1, c_2, \dots, c_K\}$ that **minimizes** the cost function

$$Cost(C) = \sum_{i=1}^K \sum_{x \in C_i} dist(x, c_i)$$

Definition for a general distance function *dist*

- **Note:** We need to find **both** the **grouping** into clusters **and** the **centroids** per cluster.

K-means Clustering

- Most common definition is with euclidean distance, minimizing the **Sum of Squares Error (SSE)** – distance function
 - Sometimes K-means clustering is defined like that
- **Problem:** Given a set X of n points in a d -dimensional space and an integer K group the points into K clusters $C = \{C_1, C_2, \dots, C_K\}$ such that

$$Cost(C) = \sum_{i=1}^K \sum_{x \in C_i} (x - c_i)^2$$

Sum of Squares Error (SSE)

is **minimized**, where c_i is the **mean** of the points in cluster C_i

Complexity of the k-means problem

- **NP-hard** if the dimensionality of the data is at least 2 ($d \geq 2$)
 - Finding the best solution in polynomial time is infeasible
- For $d = 1$ the problem is solvable in polynomial time (how?)
- A simple iterative algorithm works quite well in practice

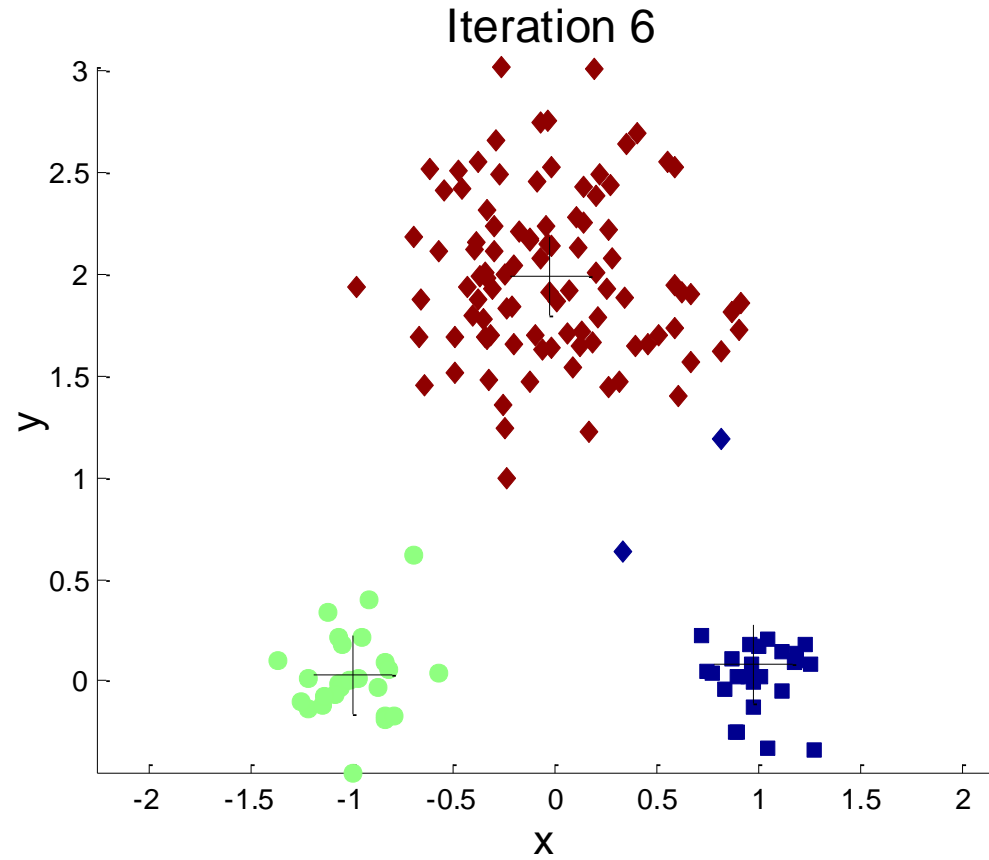
K-means Algorithm

- Also known as **Lloyd's algorithm**.
- K-means is sometimes synonymous with this algorithm

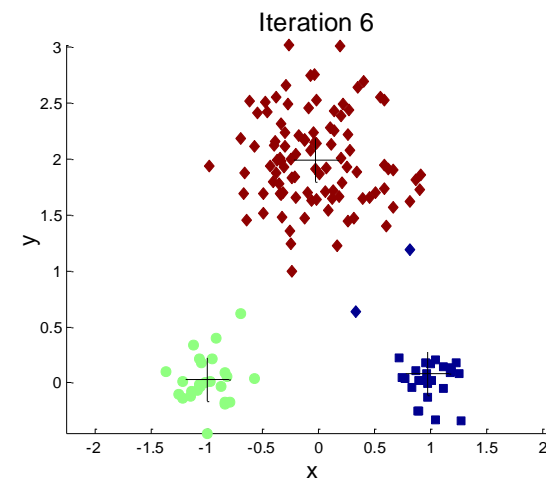
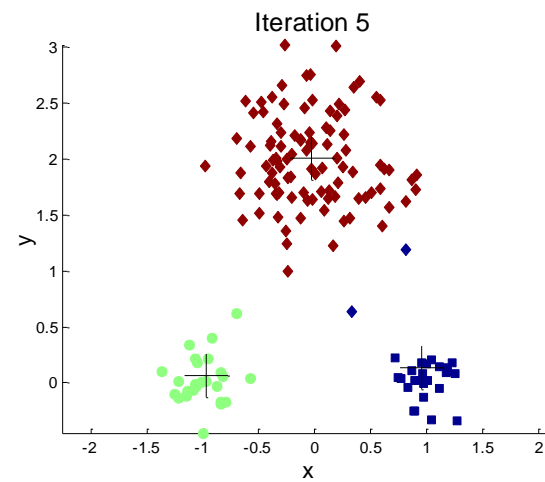
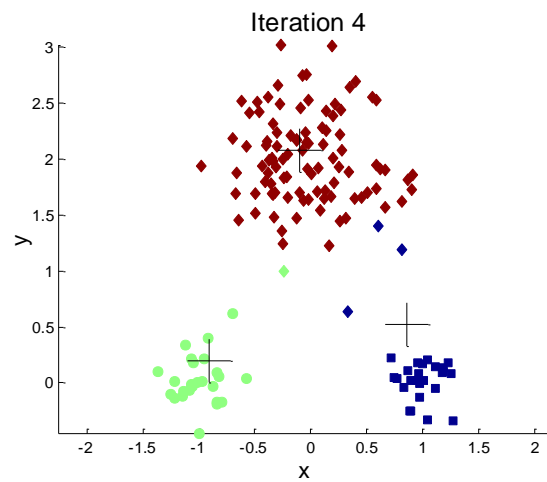
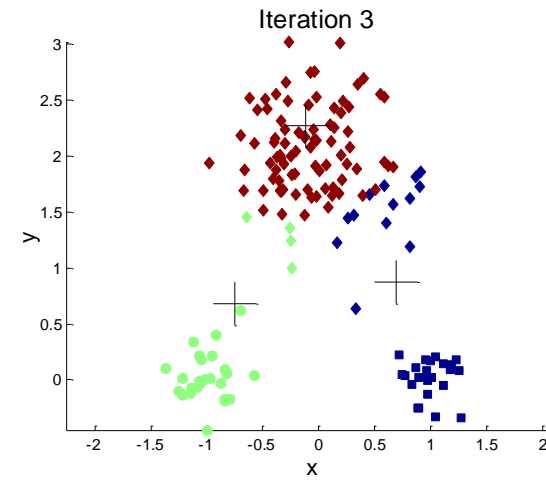
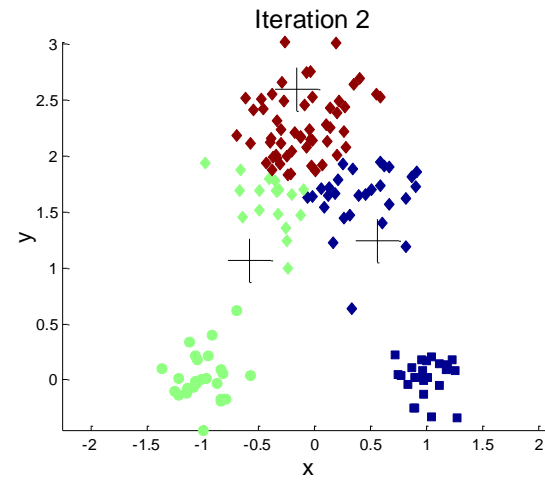
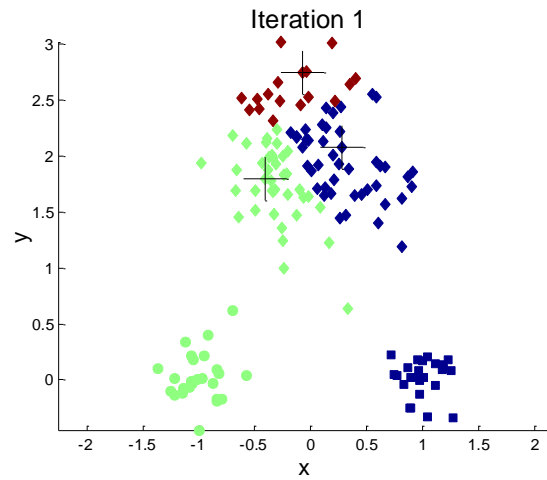
1. Select K points as the **initial centroids**
2. **repeat**
3. Form K clusters by assigning each point to the closest centroid
4. Compute the new **centroid*** of each cluster
5. **until** The centroids do not change

*The centroid of a set of points is the point that minimizes the sum of distances from the points in the set

Example



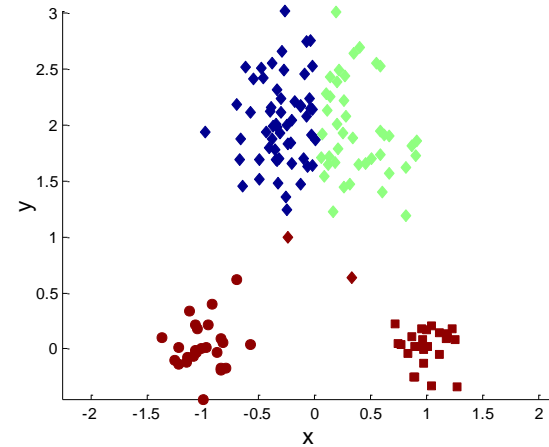
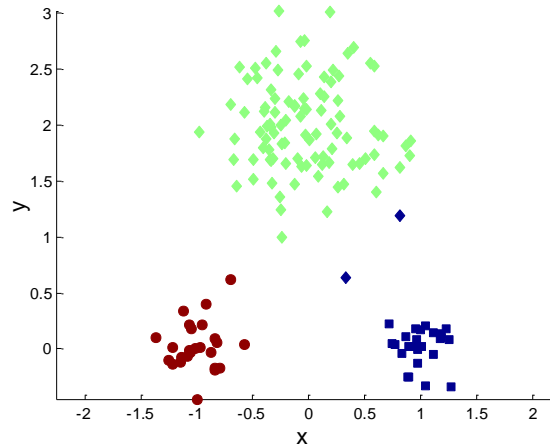
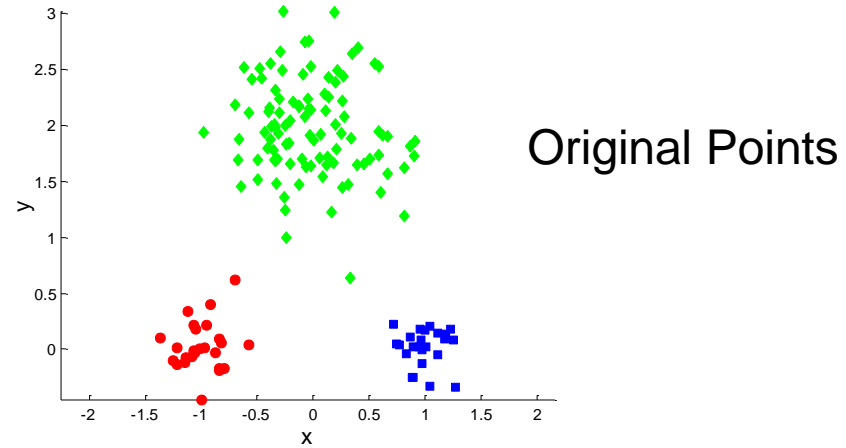
Example



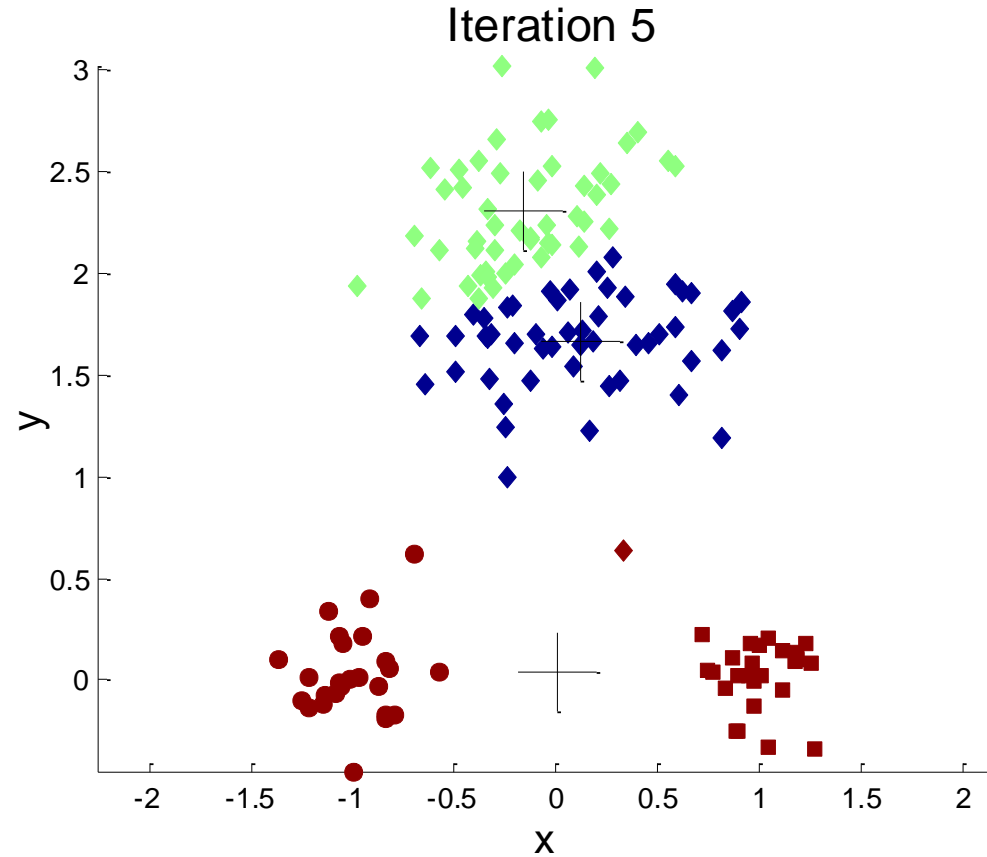
K-means Algorithm – Initialization

- Initial centroids are often chosen **randomly**.
 - Clusters produced vary from one run to another.

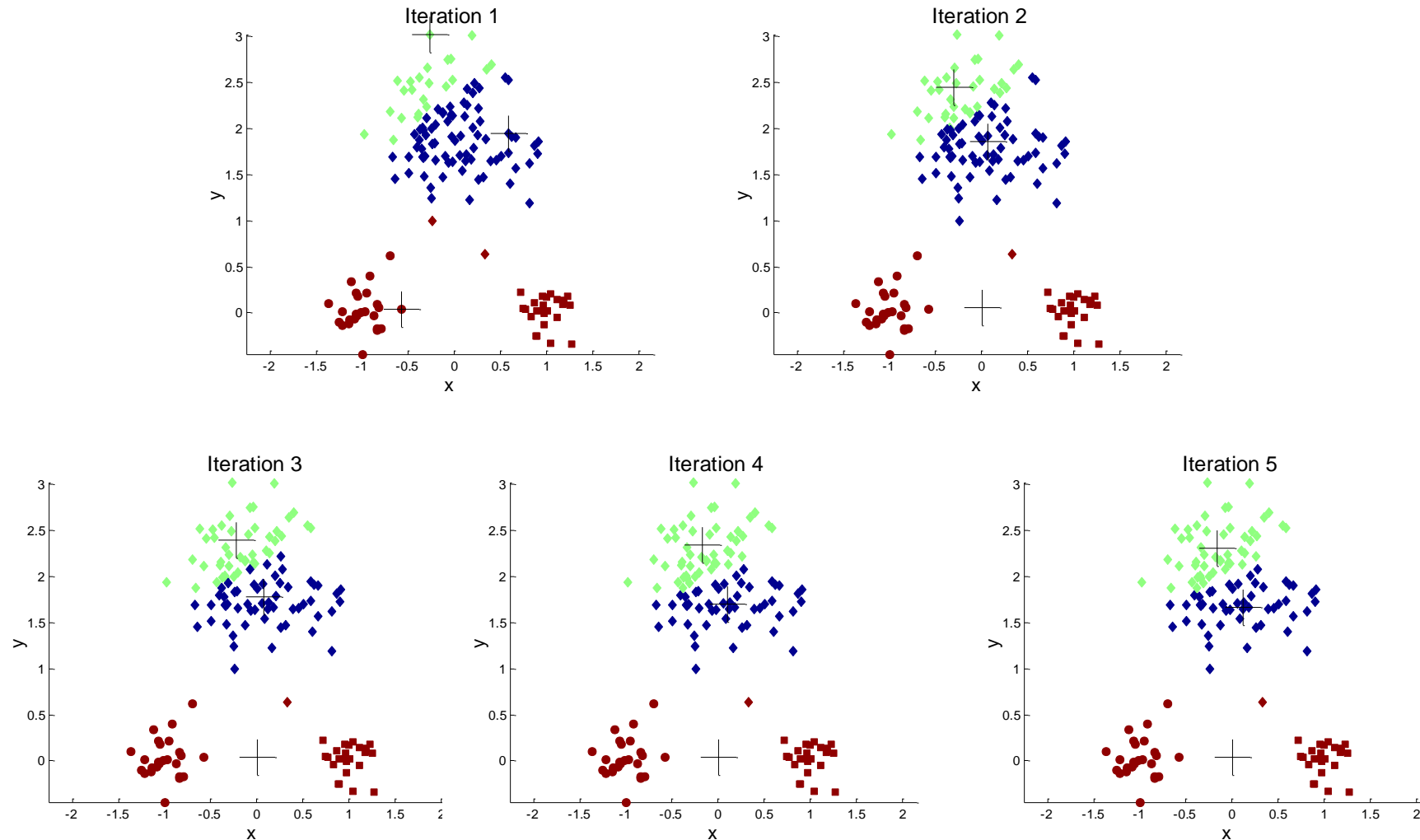
Two different K-means Clusterings



Importance of Choosing Initial Centroids



Importance of Choosing Initial Centroids ...



Dealing with Initialization

- Do **multiple runs** and select the clustering with the smallest error
- Select original set of points by methods other than random.
E.g., pick the most distant (from each other) points as cluster centers (**K-means++** algorithm)

K-means Algorithm – Centroids

- ‘**Closeness**’ is measured by some similarity or distance function
 - E.g., Euclidean distance (SSE), cosine similarity, correlation, etc.
- The **centroid** depends on the distance function
 - The **minimizer** for the distance function
- **Centroid:**
 - The **mean** of the points in the cluster for SSE, and cosine similarity
 - The **median** for Manhattan distance.
- Finding the centroid is not always easy
 - It can be an NP-hard problem for some distance functions
 - E.g., median for multiple dimensions

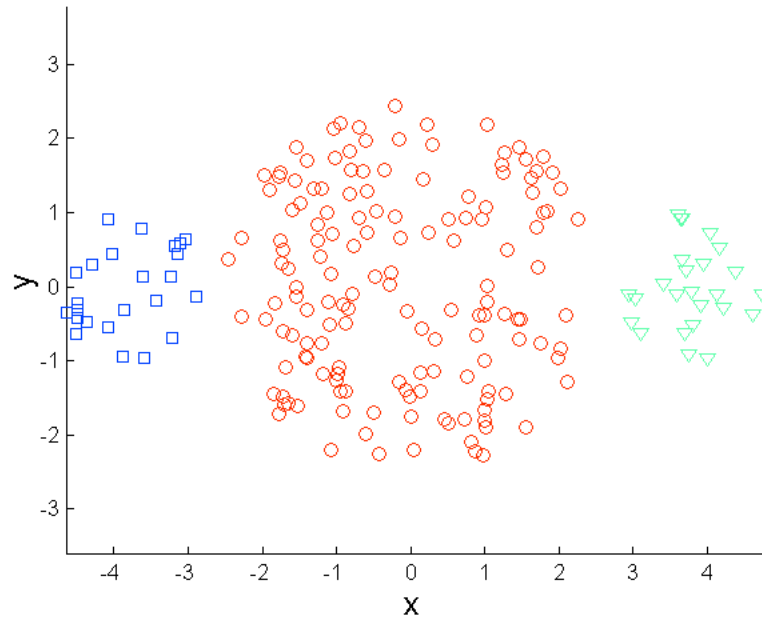
K-means Algorithm – Convergence

- K-means will **converge** for common similarity measures mentioned above.
 - Most of the convergence happens in the first few iterations.
 - Often the stopping condition is changed to ‘**Until relatively few points change clusters**’
- Complexity is $O(n \cdot K \cdot I \cdot d)$
 - n = number of points,
 - K = number of clusters,
 - I = number of iterations,
 - d = dimensionality
- In general, a fast and efficient algorithm

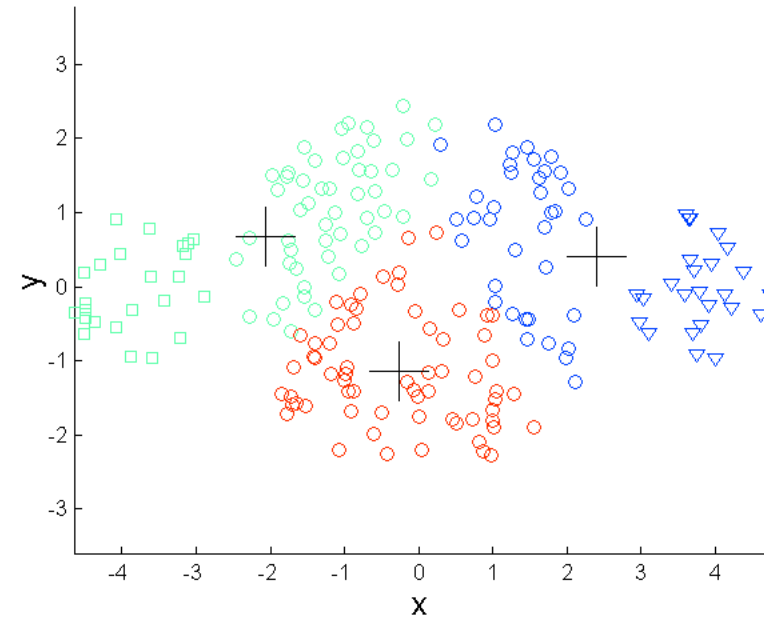
Limitations of K-means

- K-means has problems when clusters are of different:
 - sizes
 - densities
 - **non-globular** shapes
- K-means has problems when the data contains outliers.

Limitations of K-means: Differing Sizes

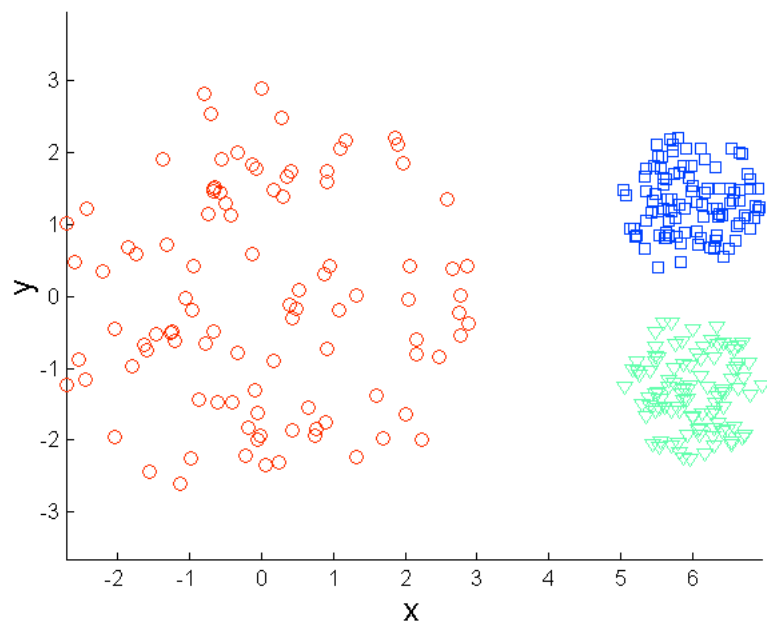


Original Points

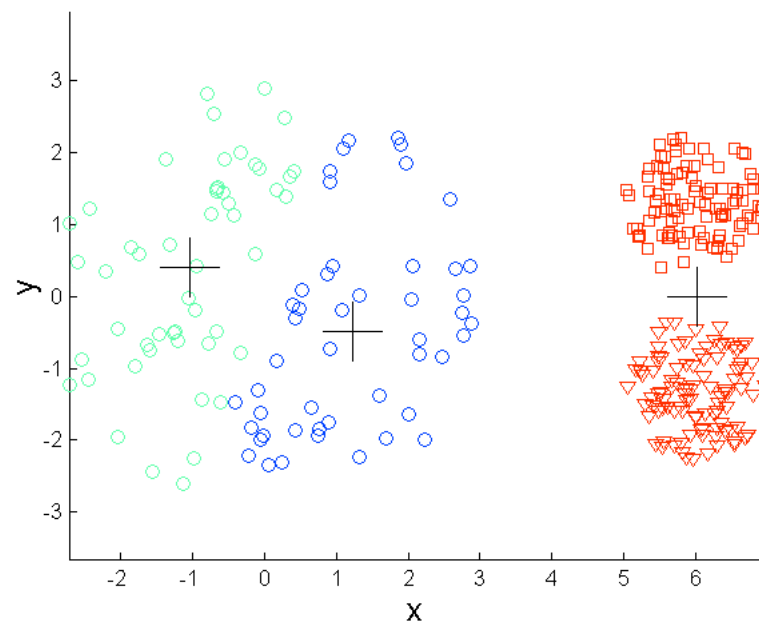


K-means (3 Clusters)

Limitations of K-means: Differing Density

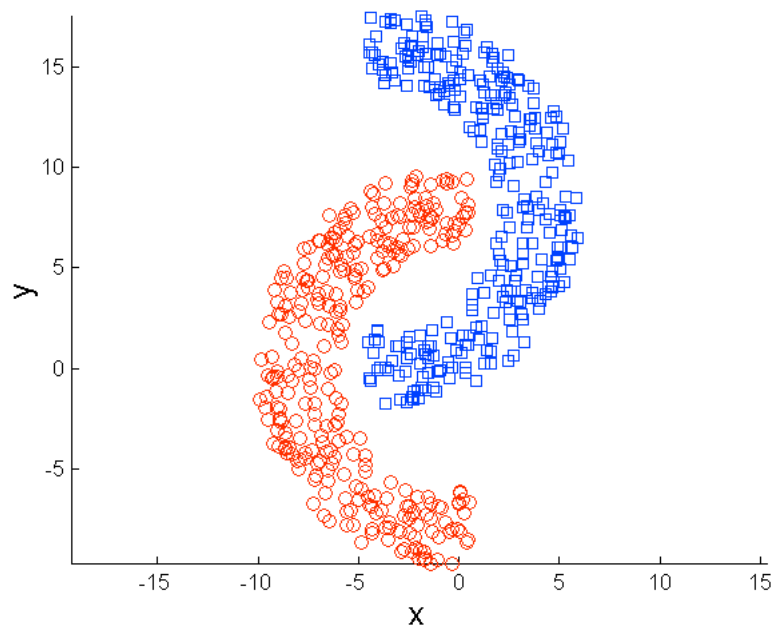


Original Points

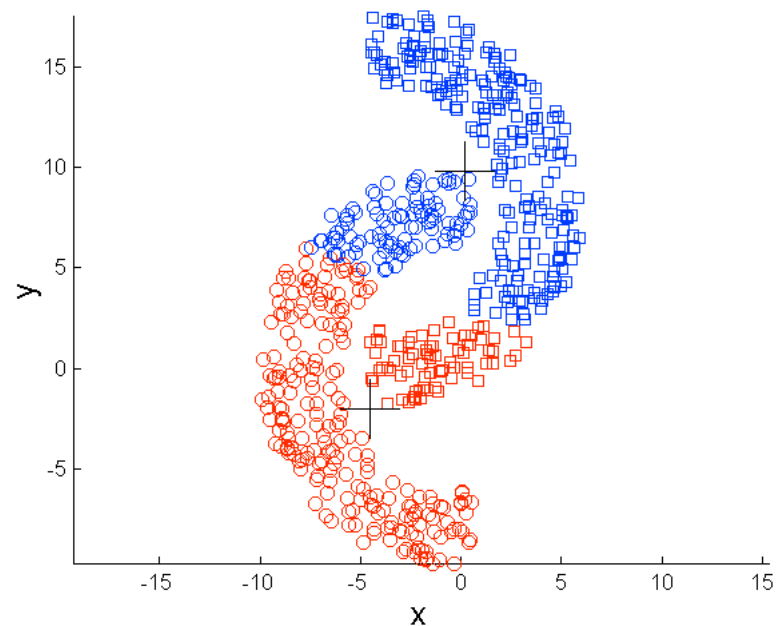


K-means (3 Clusters)

Limitations of K-means: Non-globular Shapes

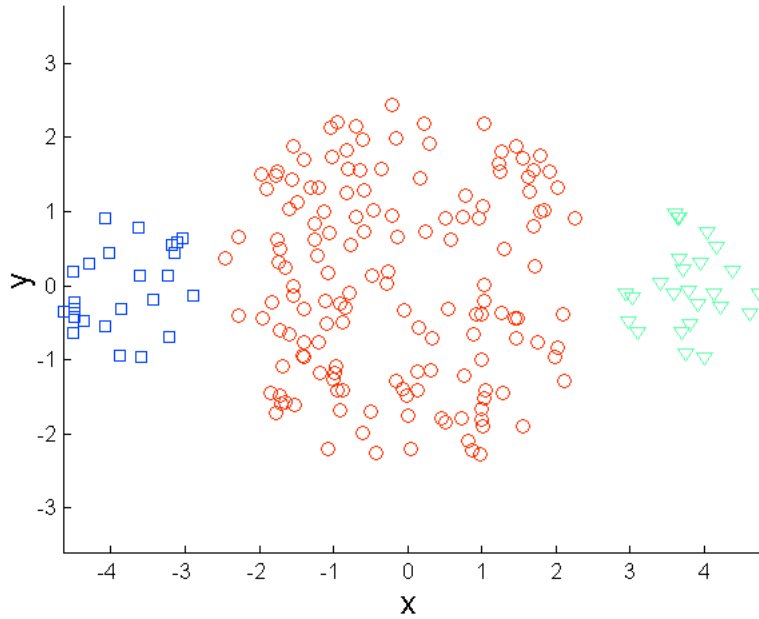


Original Points

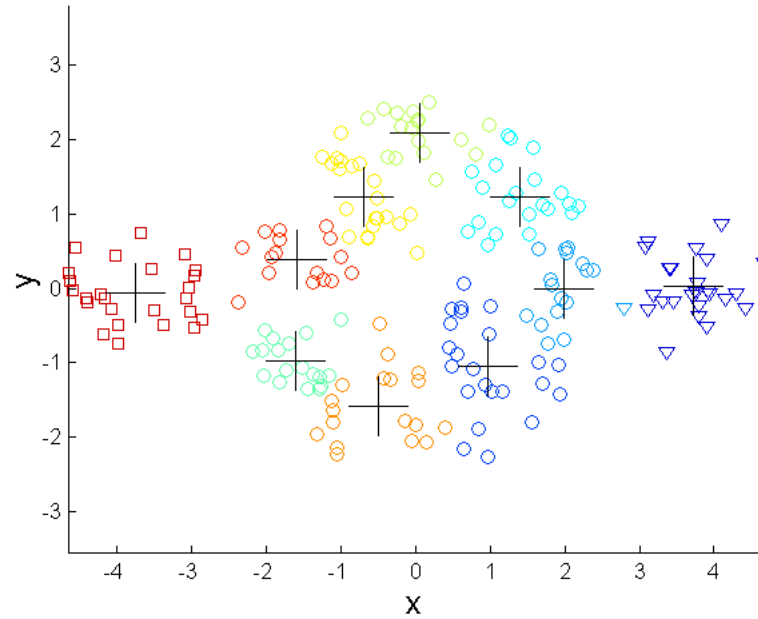


K-means (2 Clusters)

Overcoming K-means Limitations



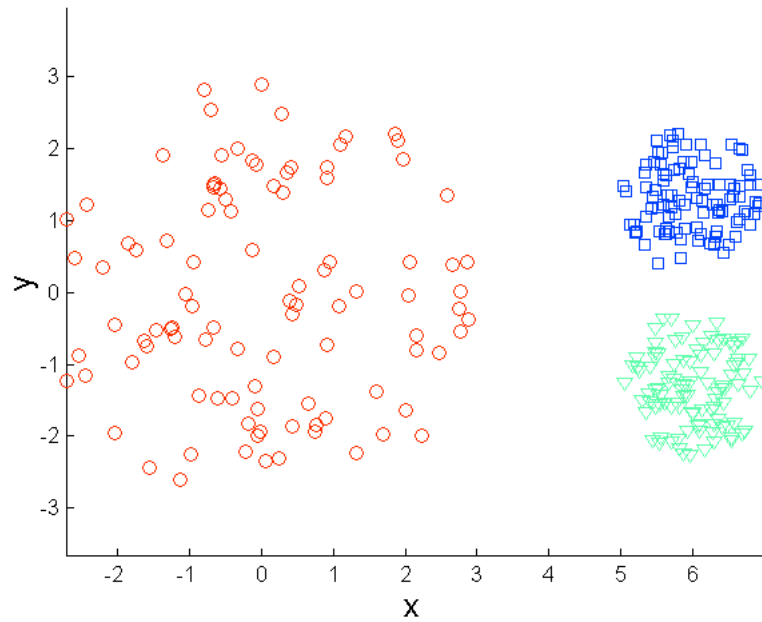
Original Points



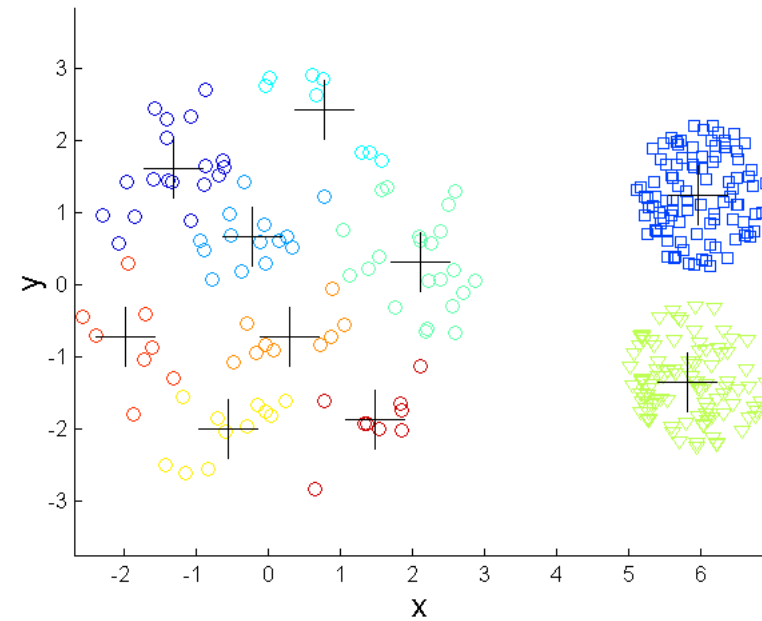
K-means Clusters

One solution is to use many clusters.
Find parts of clusters, but need to put together.

Overcoming K-means Limitations

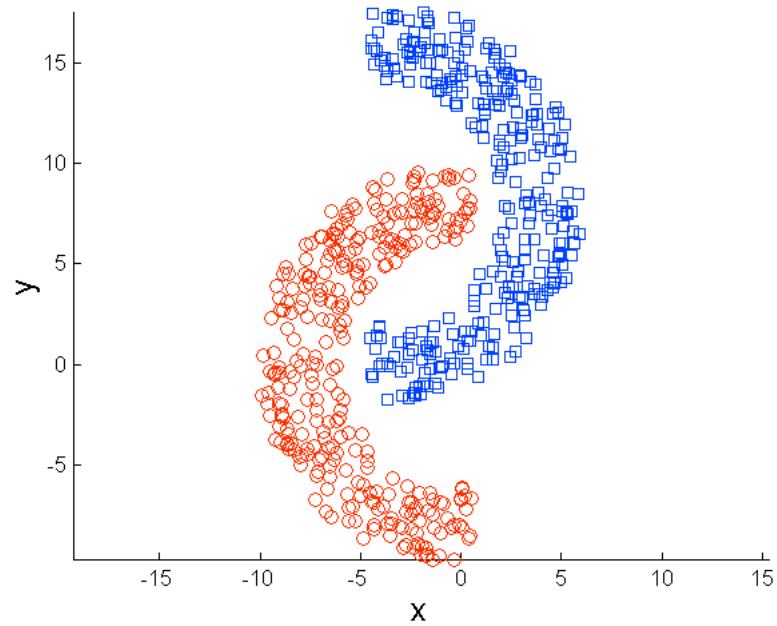


Original Points

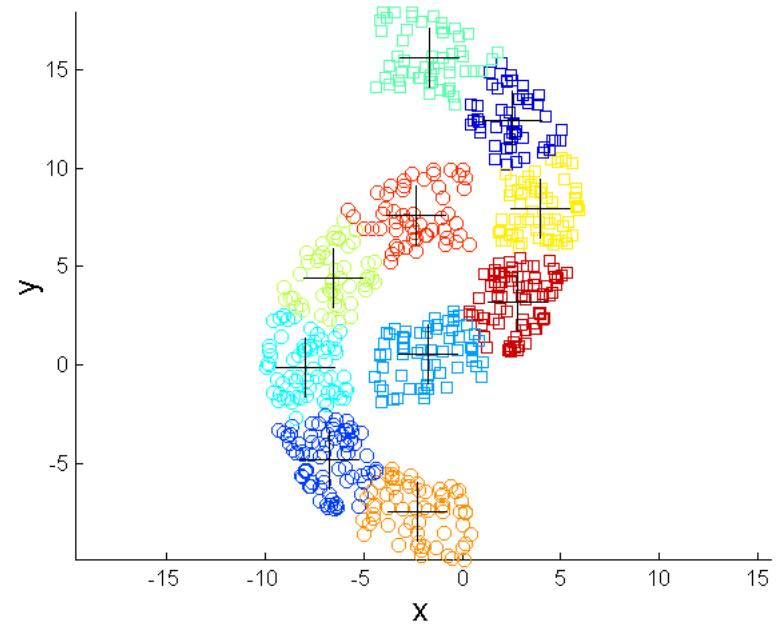


K-means Clusters

Overcoming K-means Limitations



Original Points



K-means Clusters

Variations

- **K-medoids**: Similar problem definition as in K-means, but the centroid of the cluster is defined to be one of the points in the cluster (the **medoid**).
- **K-centers**: Similar problem definition as in K-means, but the goal now is to minimize the maximum **diameter** of the clusters
 - diameter of a cluster is maximum distance between any two points in the cluster.

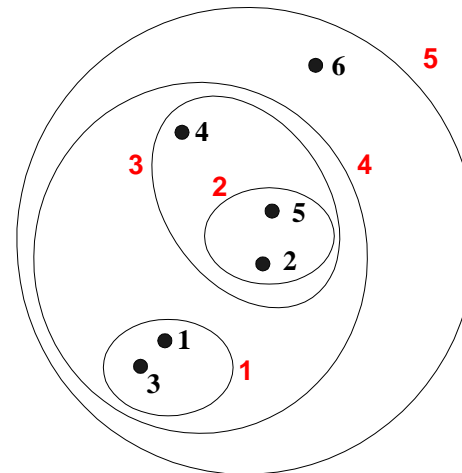
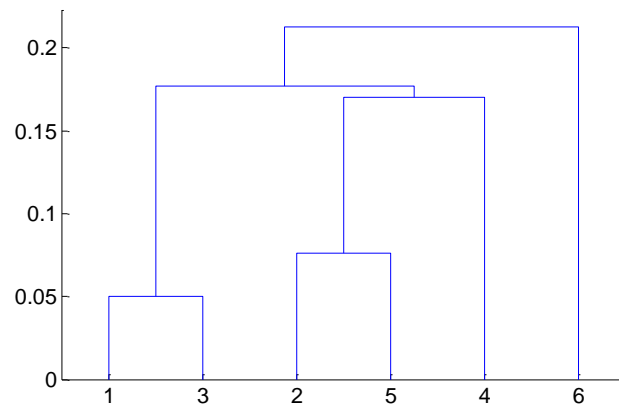
HIERARCHICAL CLUSTERING

Hierarchical Clustering

- Two main types of hierarchical clustering
 - **Agglomerative:**
 - Start with the points as individual clusters
 - At each step, merge the closest pair of clusters until only one cluster (or k clusters) left
 - **Divisive:**
 - Start with one, all-inclusive cluster
 - At each step, split a cluster until each cluster contains a point (or there are k clusters)
- Traditional hierarchical algorithms use a **similarity** or **distance matrix**
 - Merge or split one cluster at a time

Hierarchical Clustering

- Produces a set of nested clusters organized as a hierarchical tree
- Can be visualized as a **dendrogram**
 - A tree like diagram that records the sequences of merges or splits



Strengths of Hierarchical Clustering

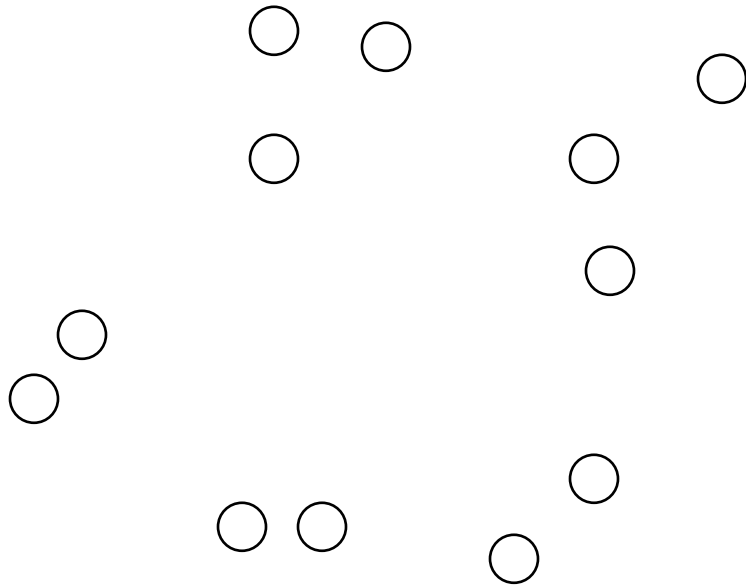
- Do not have to assume any particular number of clusters
 - Any desired number of clusters can be obtained by 'cutting' the dendrogram at the proper level
- Dendrograms **may** correspond to meaningful **taxonomies**
 - Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, ...)

Agglomerative Clustering Algorithm

- Most popular hierarchical clustering technique
- Basic algorithm is straightforward
 1. Compute the **proximity matrix**
 2. Let each data point be a cluster
 3. **Repeat**
 4. **Merge** the two **closest clusters**
 5. **Update** the proximity matrix
 6. **Until** only a single cluster remains
- Key operation is the computation of the **proximity of two clusters**
 - Different approaches to defining the **distance between clusters** distinguish the different algorithms

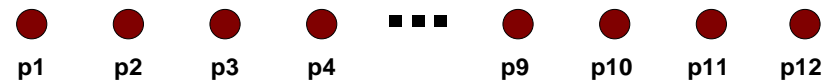
Starting Situation

- Start with **single-point clusters** and a proximity matrix **between points**



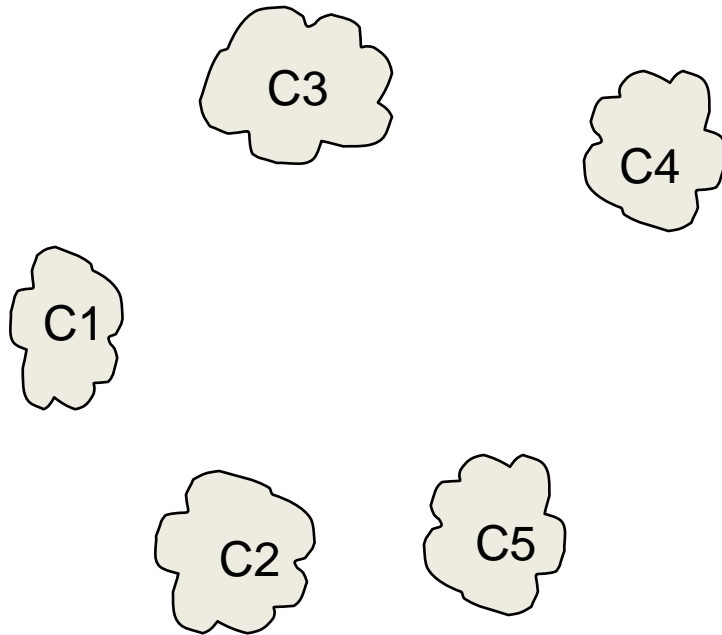
| | p1 | p2 | p3 | p4 | p5 | ... |
|----|----|----|----|----|----|-----|
| p1 | | | | | | |
| p2 | | | | | | |
| p3 | | | | | | |
| p4 | | | | | | |
| p5 | | | | | | |
| . | | | | | | |
| . | | | | | | |
| . | | | | | | |

Proximity Matrix



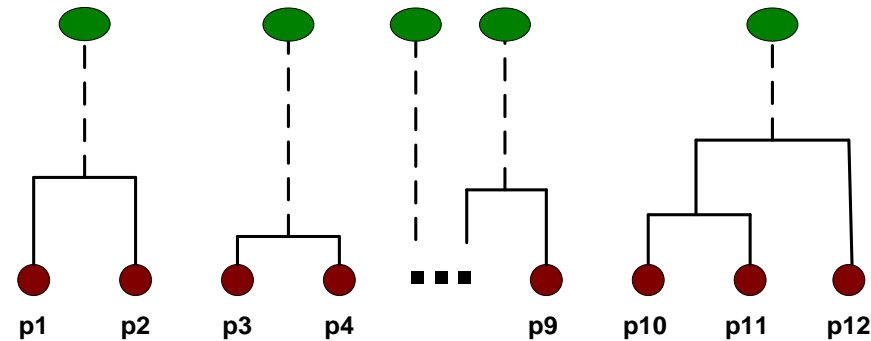
Intermediate Situation

- After some merging steps, we have some clusters and a proximity matrix **between clusters**



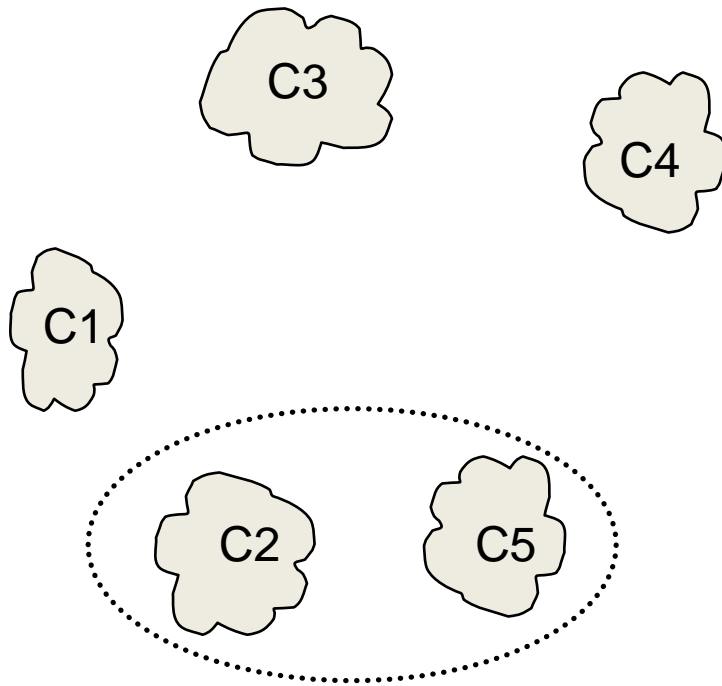
| | C1 | C2 | C3 | C4 | C5 |
|----|----|----|----|----|----|
| C1 | | | | | |
| C2 | | | | | |
| C3 | | | | | |
| C4 | | | | | |
| C5 | | | | | |

Proximity Matrix



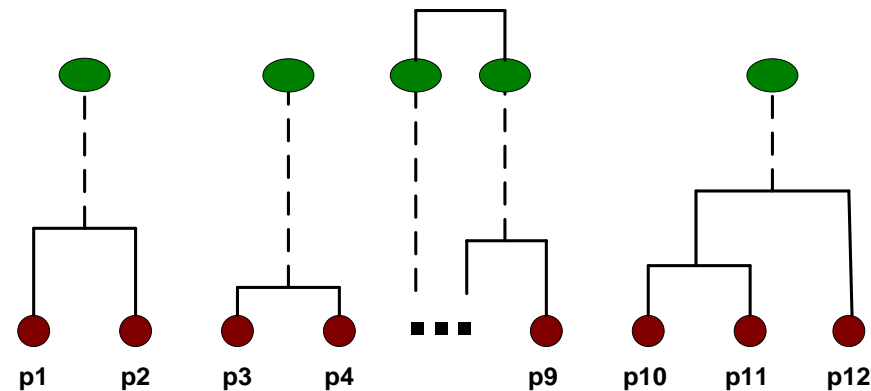
Intermediate Situation

- We want to merge the two closest clusters (C2 and C5) and update the proximity matrix.



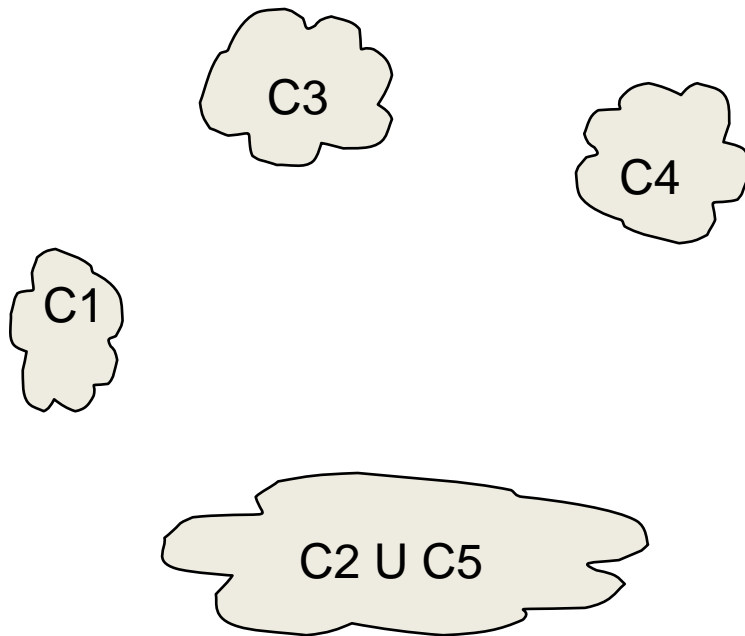
| | C1 | C2 | C3 | C4 | C5 |
|----|----|----|----|----|----|
| C1 | | | | | |
| C2 | | | | | |
| C3 | | | | | |
| C4 | | | | | |
| C5 | | | | | |

Proximity Matrix



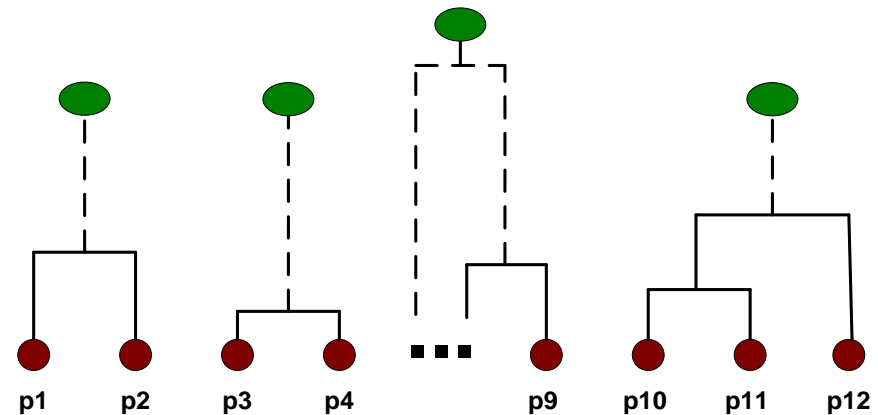
After Merging

- The question is “How do we update the proximity matrix?”

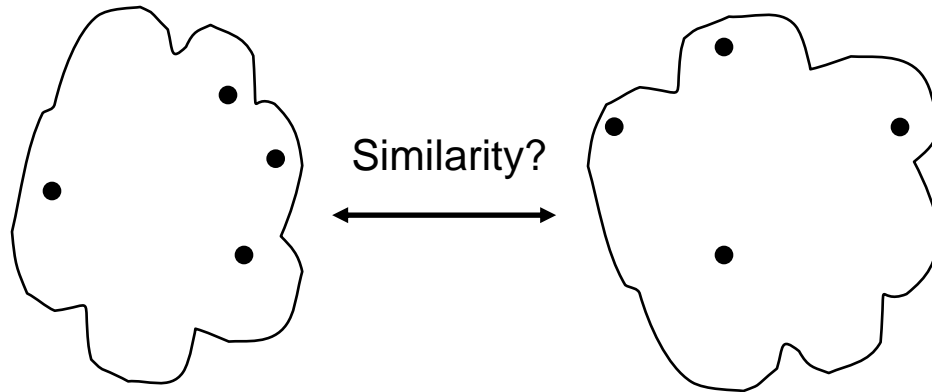


| | C1 | C2 U C5 | C3 | C4 |
|---------|----|---------------|----|----|
| C1 | | ? | | |
| C2 U C5 | ? | ? | ? | ? |
| C3 | | ? | | |
| C4 | | ? | | |

Proximity Matrix



How to Define Inter-Cluster Similarity

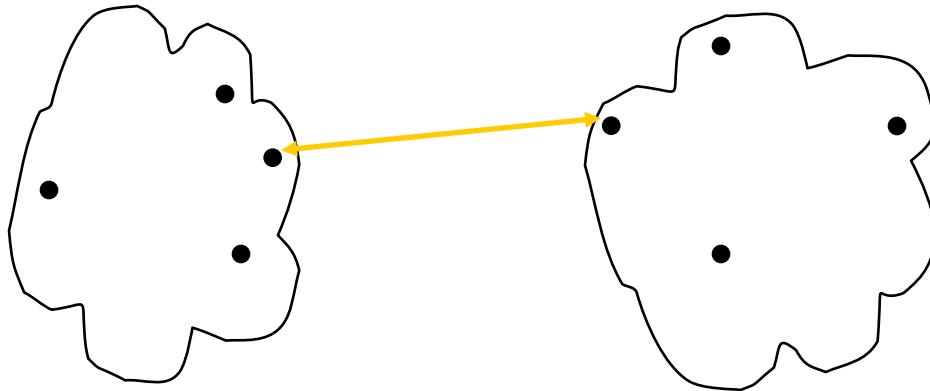


- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
 - Ward's Method uses squared error

| | p1 | p2 | p3 | p4 | p5 | ... |
|----|----|----|----|----|----|-----|
| p1 | | | | | | |
| p2 | | | | | | |
| p3 | | | | | | |
| p4 | | | | | | |
| p5 | | | | | | |
| . | | | | | | |
| . | | | | | | |
| . | | | | | | |

Proximity Matrix

How to Define Inter-Cluster Similarity

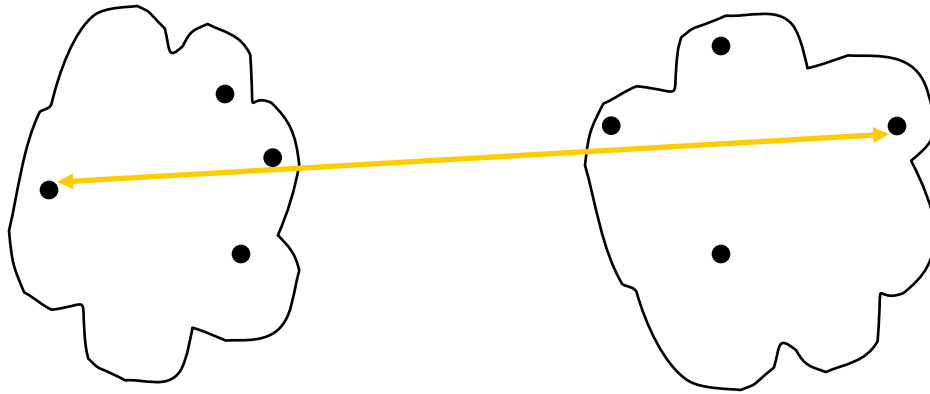


- **MIN**
- **MAX**
- **Group Average**
- **Distance Between Centroids**
- **Other methods driven by an objective function**
 - Ward's Method uses squared error

| | p1 | p2 | p3 | p4 | p5 | ... |
|----|----|----|----|----|----|-----|
| p1 | | | | | | |
| p2 | | | | | | |
| p3 | | | | | | |
| p4 | | | | | | |
| p5 | | | | | | |
| . | | | | | | |
| . | | | | | | |
| . | | | | | | |

Proximity Matrix

How to Define Inter-Cluster Similarity

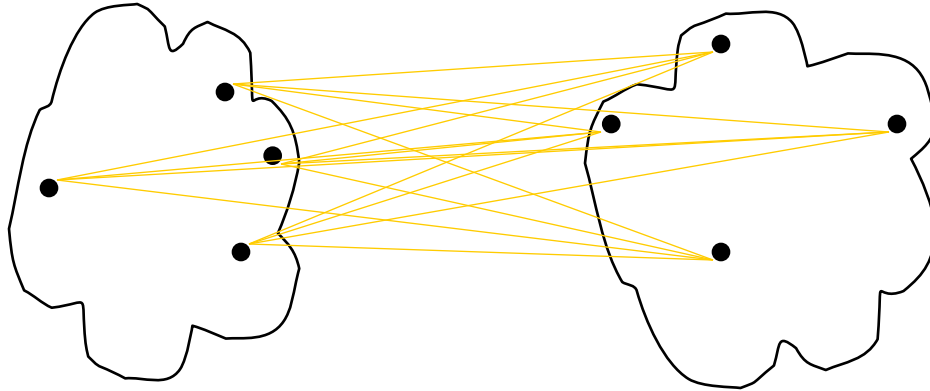


- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function
 - Ward's Method uses squared error

| | p1 | p2 | p3 | p4 | p5 | ... |
|----|----|----|----|----|----|-----|
| p1 | | | | | | |
| p2 | | | | | | |
| p3 | | | | | | |
| p4 | | | | | | |
| p5 | | | | | | |
| . | | | | | | |
| . | | | | | | |
| . | | | | | | |

Proximity Matrix

How to Define Inter-Cluster Similarity



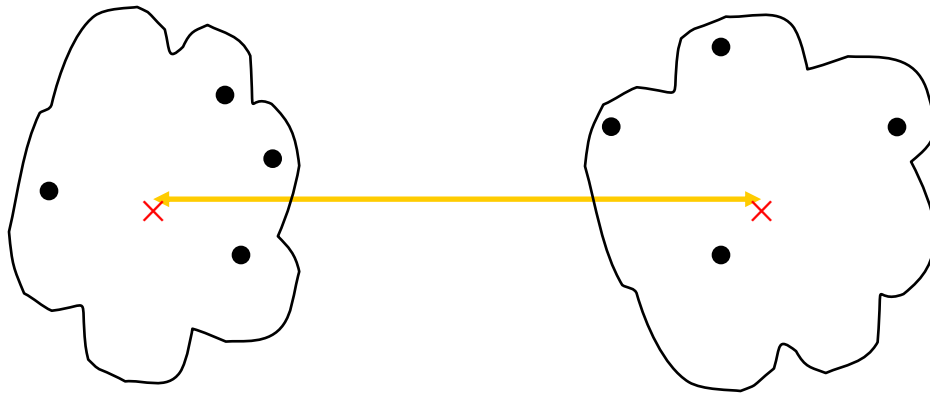
| | p1 | p2 | p3 | p4 | p5 | ... |
|----|----|----|----|----|----|-----|
| p1 | | | | | | |
| p2 | | | | | | |
| p3 | | | | | | |
| p4 | | | | | | |
| p5 | | | | | | |
| . | | | | | | |
| . | | | | | | |
| . | | | | | | |

Proximity Matrix

- MIN
- MAX
- **Group Average**
- Distance Between Centroids
- Other methods driven by an objective function
 - Ward's Method uses squared error

$$\text{proximity}(\text{Cluster}_i, \text{Cluster}_j) = \frac{\sum_{\substack{p_i \in \text{Cluster}_i \\ p_j \in \text{Cluster}_j}} \text{proximity}(p_i, p_j)}{|\text{Cluster}_i| * |\text{Cluster}_j|}$$

How to Define Inter-Cluster Similarity



| | p1 | p2 | p3 | p4 | p5 | ... |
|----|----|----|----|----|----|-----|
| p1 | | | | | | |
| p2 | | | | | | |
| p3 | | | | | | |
| p4 | | | | | | |
| p5 | | | | | | |
| . | | | | | | |
| . | | | | | | |
| . | | | | | | |

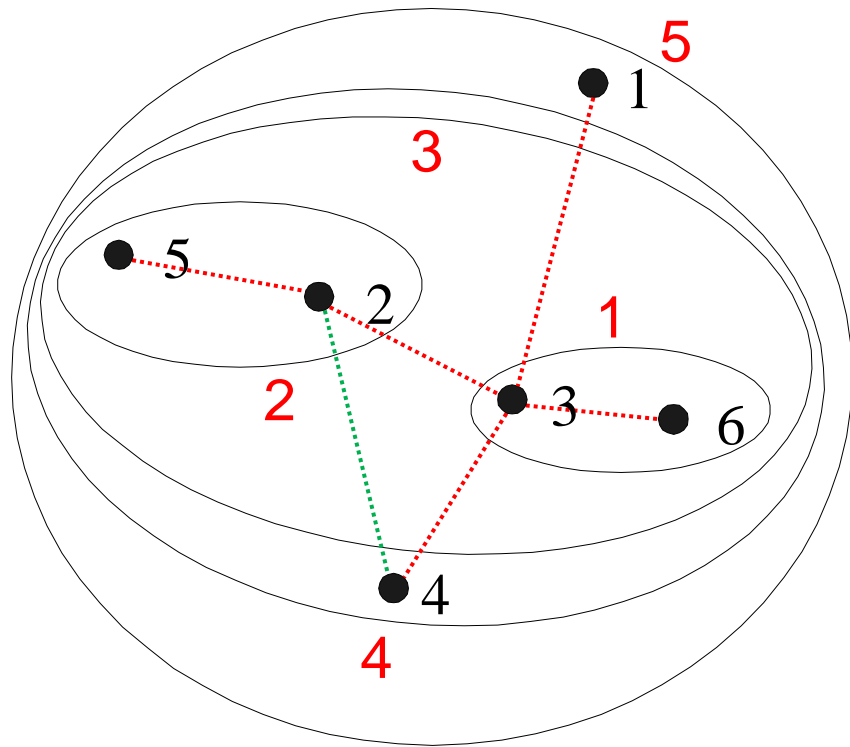
Proximity Matrix

- MIN
- MAX
- Group Average
- **Distance Between Centroids**
- Other methods driven by an objective function
 - Ward's Method uses squared error

Single Link – Complete Link

- Another way to view the processing of the hierarchical algorithm is that we create links between the **elements** in order of **increasing distance**
 - The MIN – **Single Link**, will merge two clusters when a **single pair** of elements between the two clusters is linked
 - The MAX – **Complete Linkage** will merge two clusters when **all pairs** of elements between the two clusters have been linked.

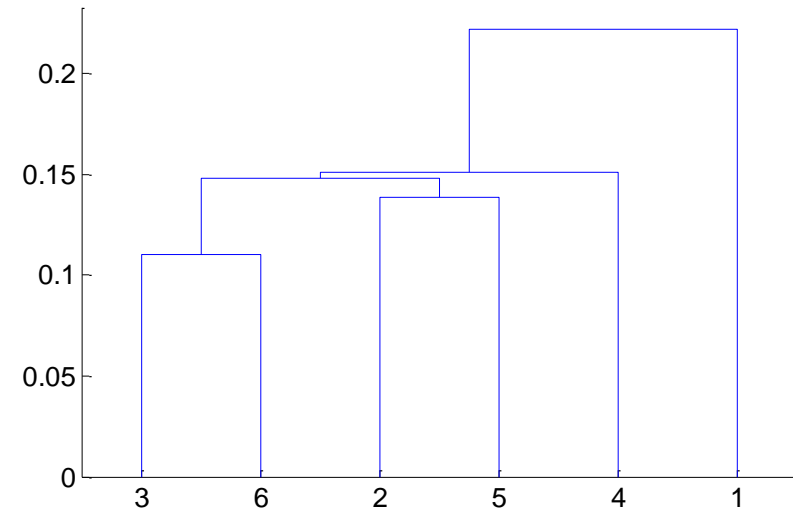
Hierarchical Clustering: MIN



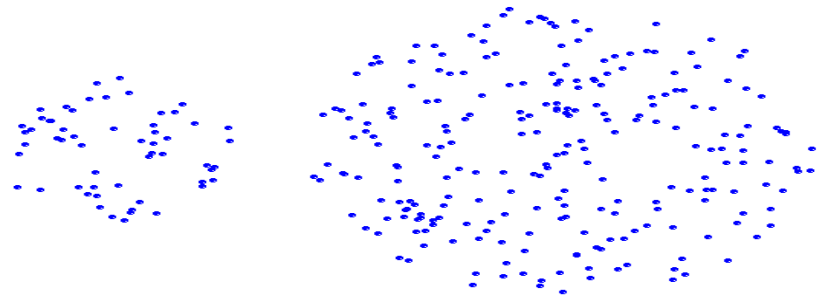
Nested Clusters

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|------------|------------|------------|------------|------------|------------|
| 1 | 0 | .24 | .22 | .37 | .34 | .23 |
| 2 | .24 | 0 | .15 | .20 | .14 | .25 |
| 3 | .22 | .15 | 0 | .15 | .28 | .11 |
| 4 | .37 | .20 | .15 | 0 | .29 | .22 |
| 5 | .34 | .14 | .28 | .29 | 0 | .39 |
| 6 | .23 | .25 | .11 | .22 | .39 | 0 |

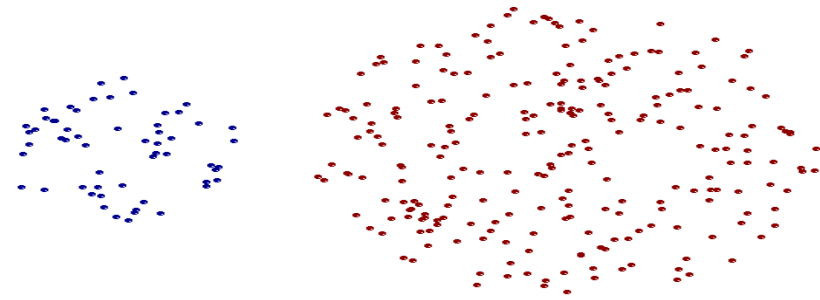
Dendrogram



Strength of MIN



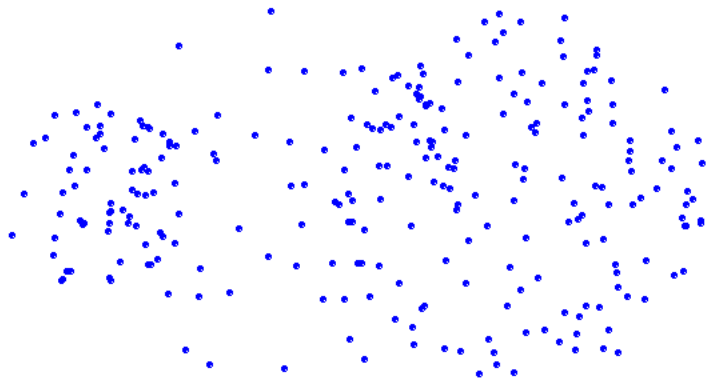
Original Points



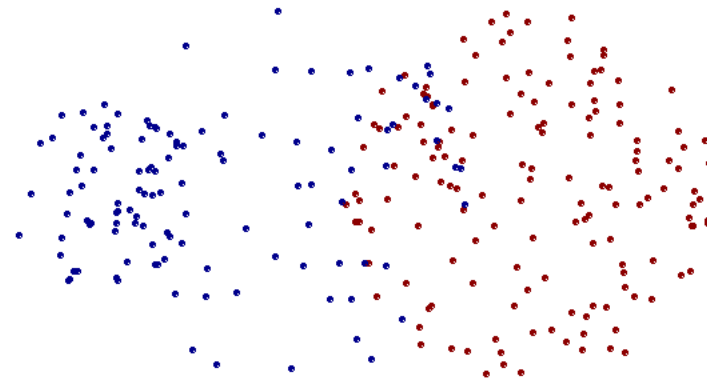
Two Clusters

- Can handle non-elliptical shapes

Limitations of MIN



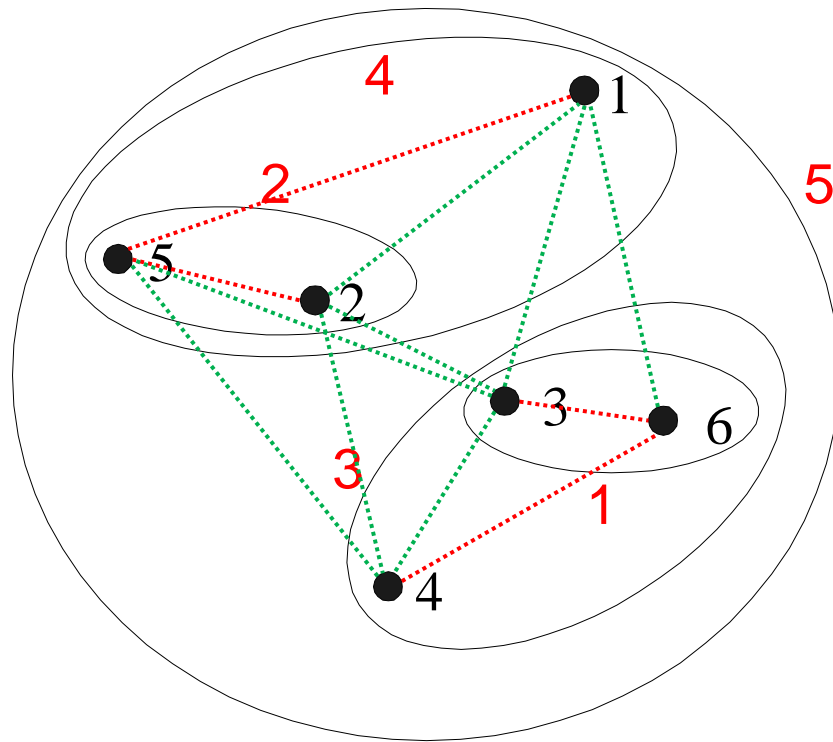
Original Points



Two Clusters

- Sensitive to noise and outliers

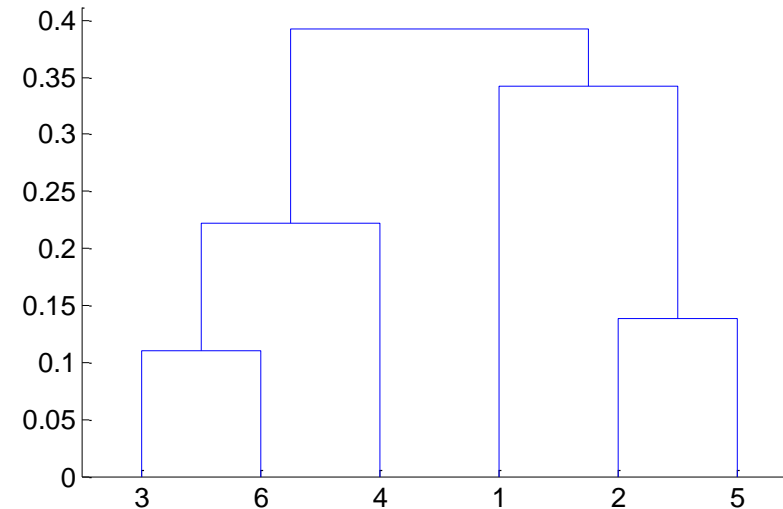
Hierarchical Clustering: MAX



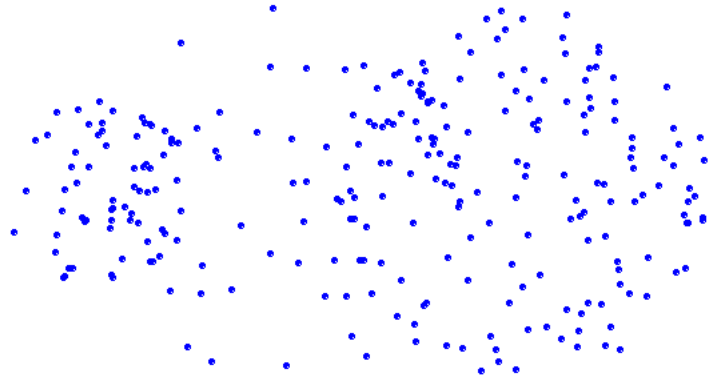
Nested Clusters

Dendrogram

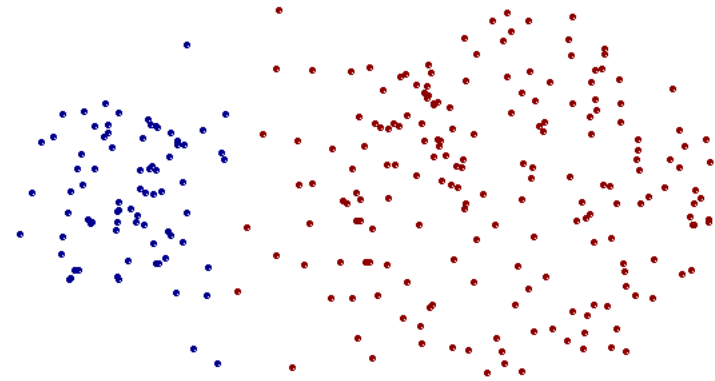
| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|-----|-----|-----|-----|-----|-----|
| 1 | 0 | .24 | .22 | .37 | .34 | .23 |
| 2 | .24 | 0 | .15 | .20 | .14 | .25 |
| 3 | .22 | .15 | 0 | .15 | .28 | .11 |
| 4 | .37 | .20 | .15 | 0 | .29 | .22 |
| 5 | .34 | .14 | .28 | .29 | 0 | .39 |
| 6 | .23 | .25 | .11 | .22 | .39 | 0 |



Strength of MAX



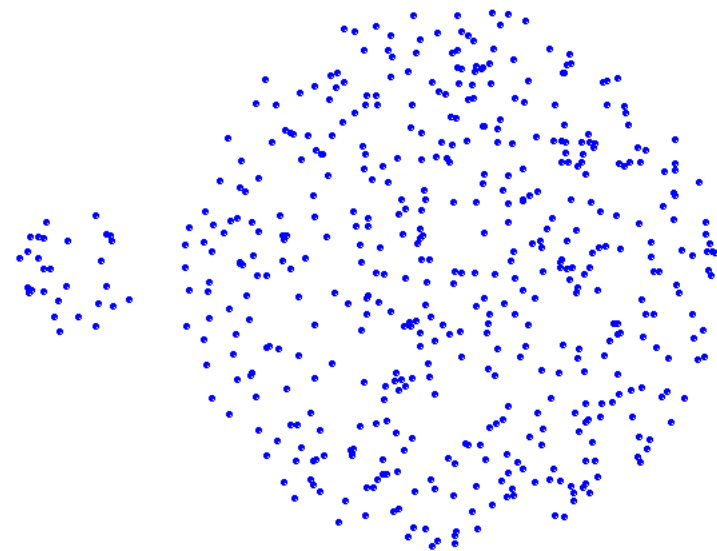
Original Points



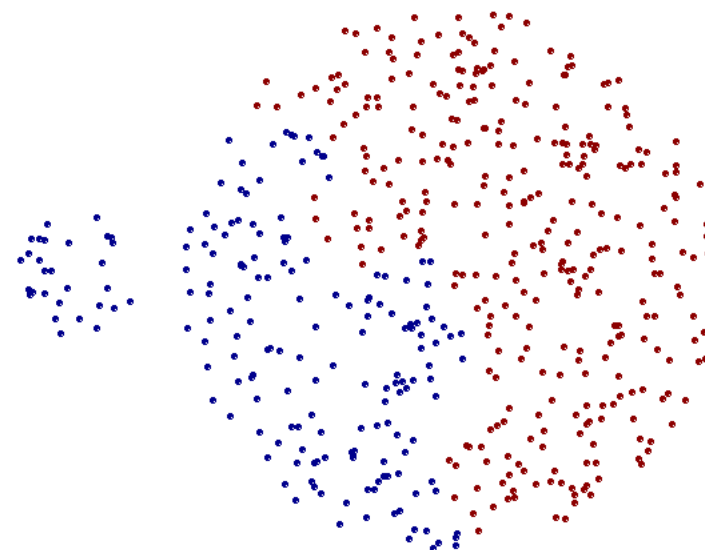
Two Clusters

- Less susceptible to noise and outliers

Limitations of MAX



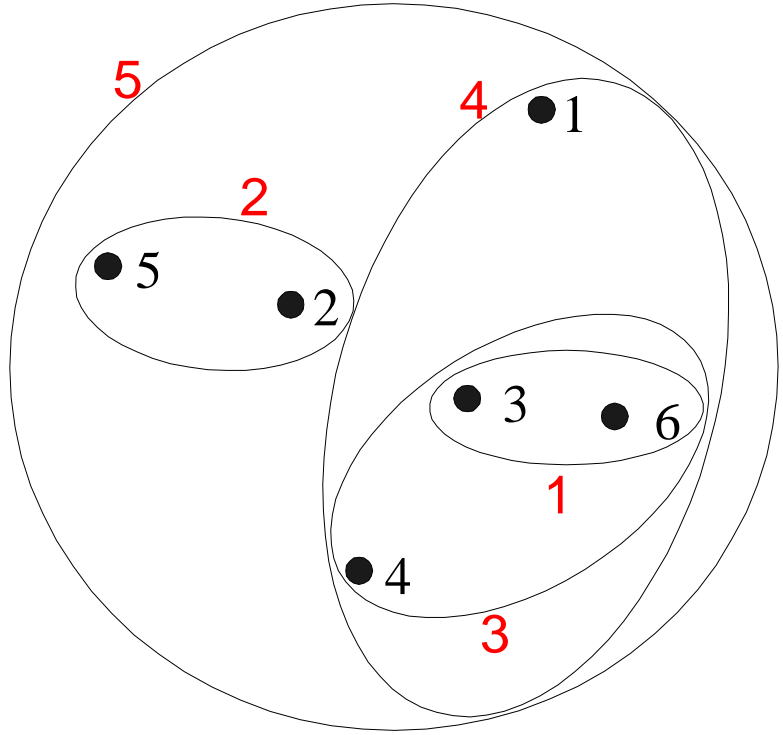
Original Points



Two Clusters

- Tends to break large clusters
- Biased towards globular clusters

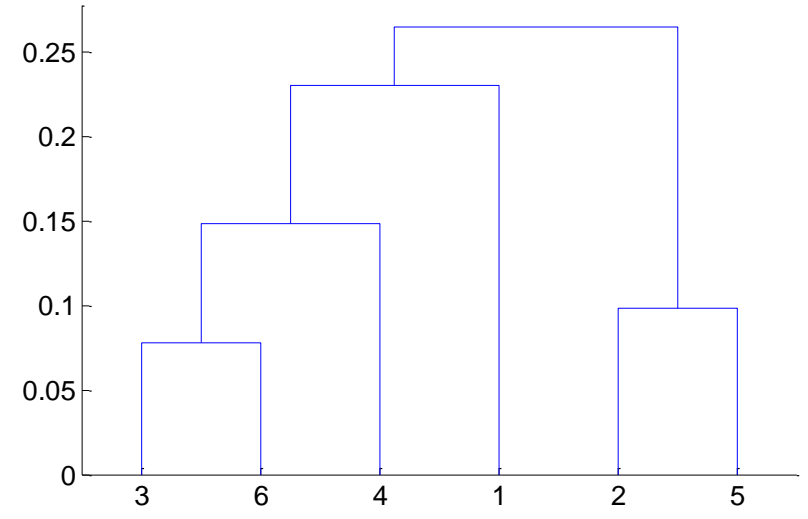
Hierarchical Clustering: Group Average



Nested Clusters

Dendrogram

| | 1 | 2 | 3 | 4 | 5 | 6 |
|---|-----|-----|-----|-----|-----|-----|
| 1 | 0 | .24 | .22 | .37 | .34 | .23 |
| 2 | .24 | 0 | .15 | .20 | .14 | .25 |
| 3 | .22 | .15 | 0 | .15 | .28 | .11 |
| 4 | .37 | .20 | .15 | 0 | .29 | .22 |
| 5 | .34 | .14 | .28 | .29 | 0 | .39 |
| 6 | .23 | .25 | .11 | .22 | .39 | 0 |



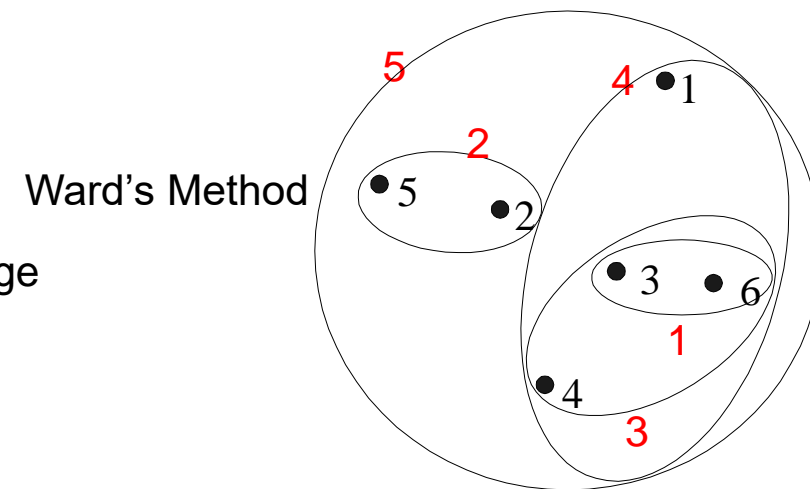
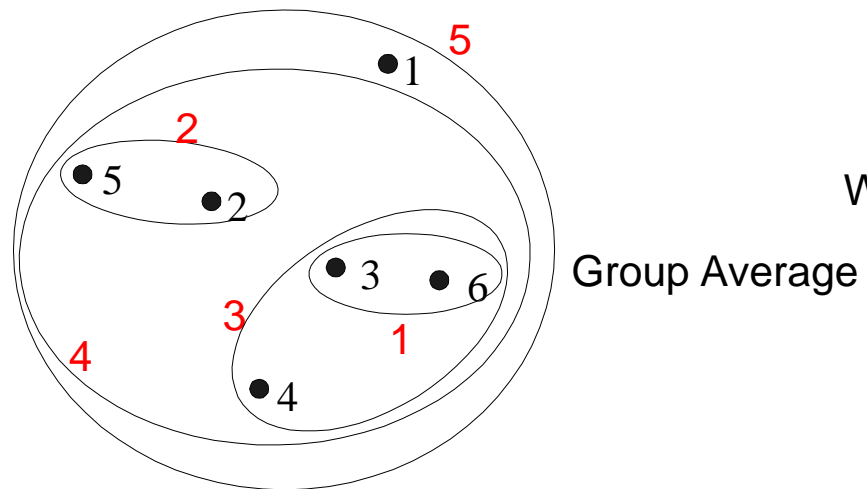
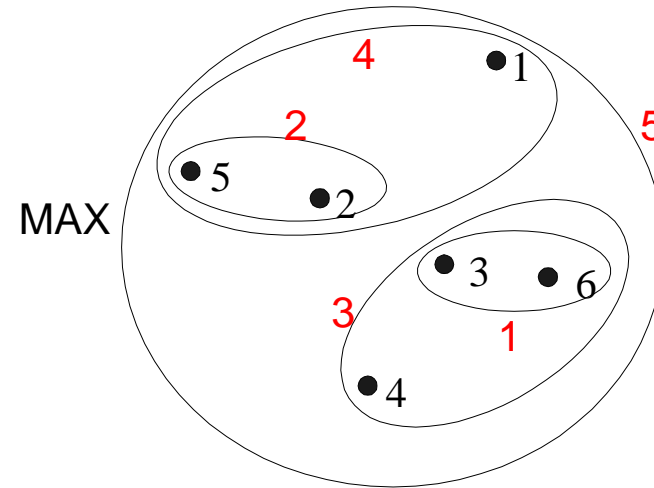
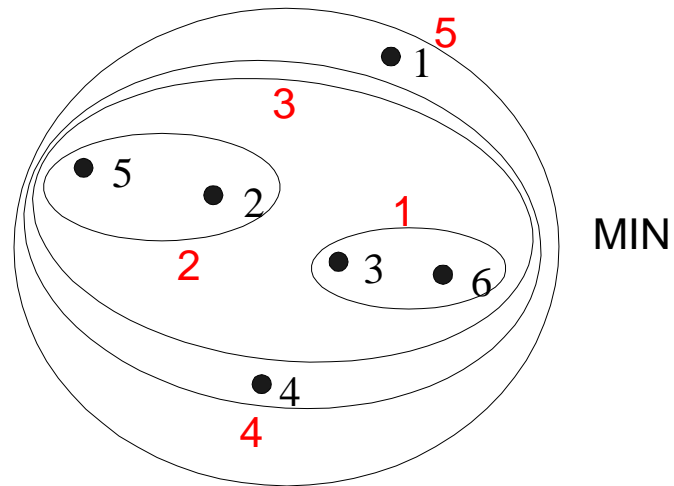
Hierarchical Clustering: Group Average

- Compromise between Single and Complete Link
- Strengths
 - Less susceptible to noise and outliers
- Limitations
 - Biased towards globular clusters

Cluster Similarity: Ward's Method

- Similarity of two clusters is based on the **increase** in **squared error (SSE)** when two clusters are merged
 - Similar to group average if distance between points is sum of squares distance
- Hierarchical analogue of K-means
 - Can be used to initialize K-means
- Less susceptible to noise and outliers
- Biased towards globular cluster

Hierarchical Clustering: Comparison



Hierarchical Clustering: Time and Space requirements

- $O(N^2)$ space since it uses the proximity matrix.
 - N is the number of points.
- $O(N^3)$ time in many cases
 - There are N steps and at each step the size, N^2 , proximity matrix must be updated and searched
 - Complexity can be reduced to $O(N^2 \log(N))$ time for some approaches

Hierarchical Clustering: Problems and Limitations

- Computational complexity in time and space
- Once a decision is made to combine two clusters, it cannot be undone
- No objective function is directly minimized
- Different schemes have problems with one or more of the following:
 - Sensitivity to noise and outliers
 - Difficulty handling different sized clusters and convex shapes
 - Breaking large clusters

DBSCAN

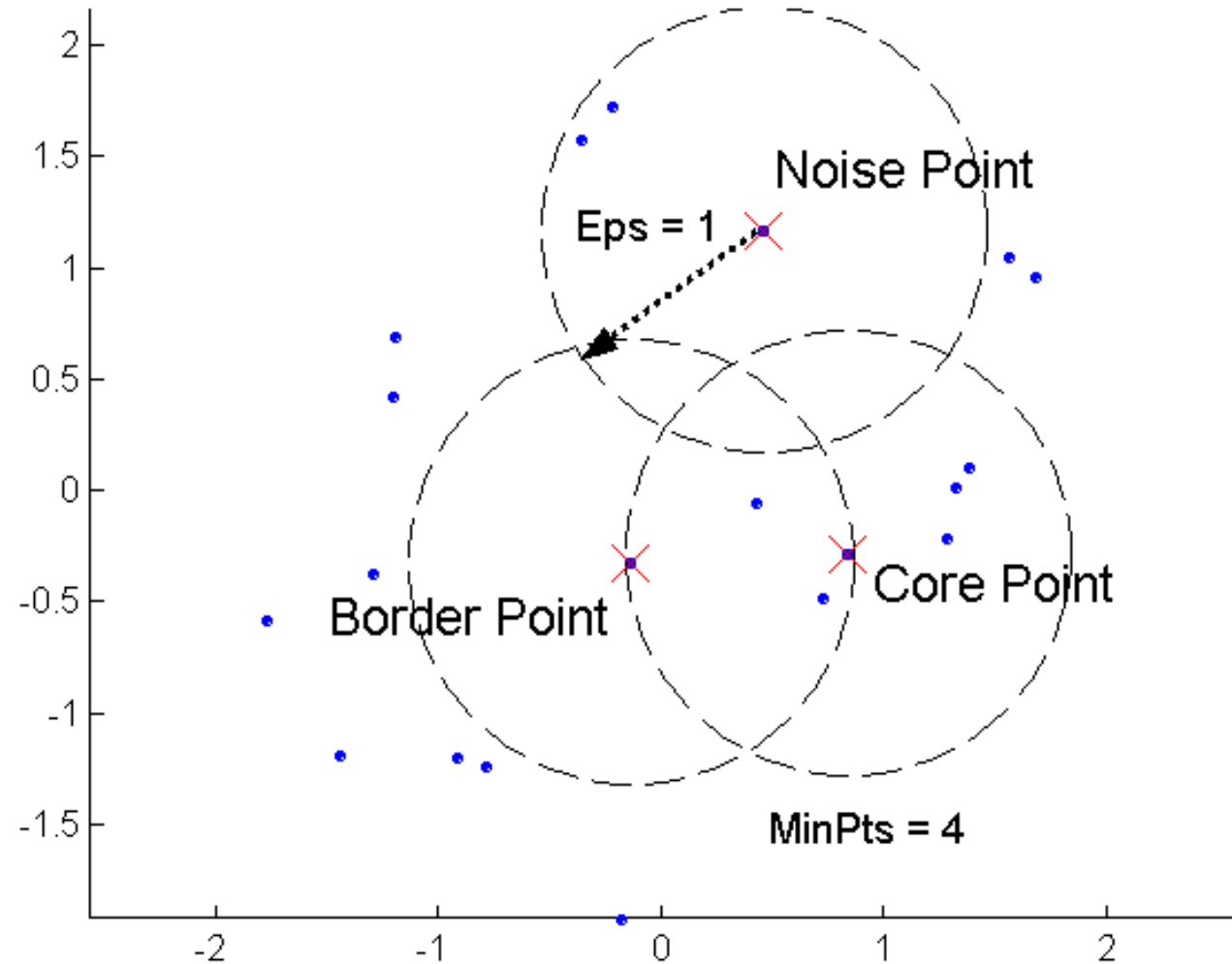
DBSCAN: Density-Based Clustering

- DBSCAN is a Density-Based Clustering algorithm
- Reminder: In density-based clustering we partition points into dense regions separated by not-so-dense regions.
- Important Questions:
 - How do we measure density?
 - What is a dense region?
- DBSCAN:
 - Density at point p : number of points within a circle of radius Eps
 - Dense Region: A circle of radius Eps that contains at least $MinPts$ points

DBSCAN

- Characterization of points
 - A point is a **core point** if it has more than a specified number of points (**MinPts**) within **Eps**
 - These points belong in a **dense region** and are at the **interior** of a cluster
 - A **border point** has fewer than **MinPts** within **Eps**, but is in the neighborhood of a **core** point.
 - A **noise point** is any point that is not a core point or a border point.

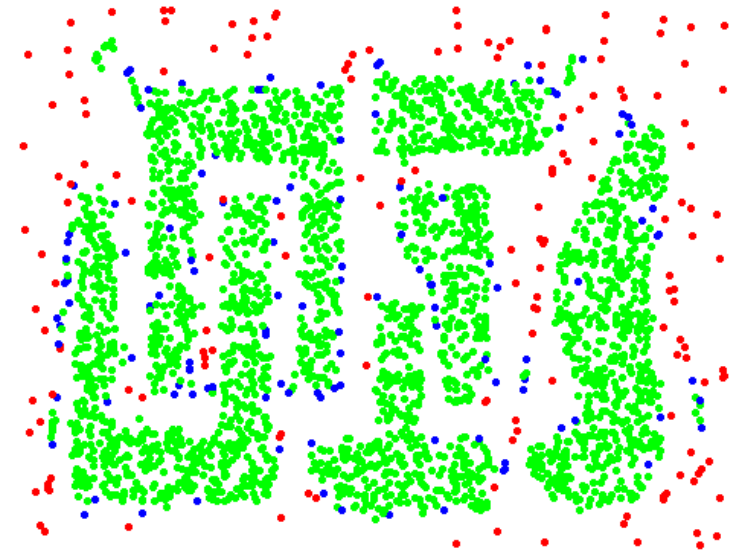
DBSCAN: Core, Border, and Noise Points



DBSCAN: Core, Border and Noise Points



Original Points



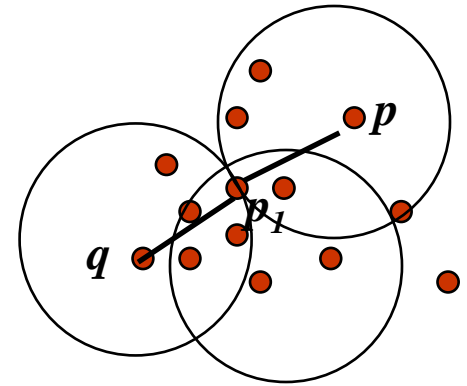
Point types: **core**, **border** and **noise**

Eps = 10, MinPts = 4

Density-Connected points

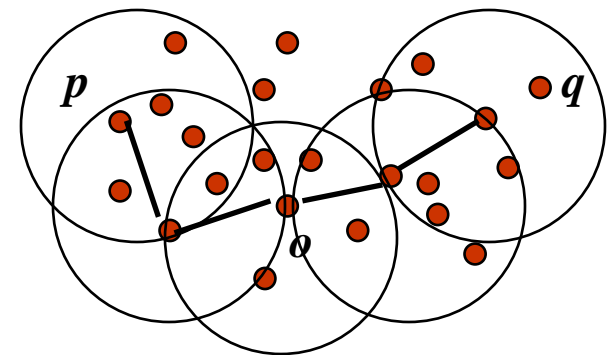
- Density edge

- We place an **edge** between two core points **q** and **p** if they are within distance **Eps**.



- Density-connected

- A point **p** is **density-connected** to a point **q** if there is a **path of edges** from **p** to **q**

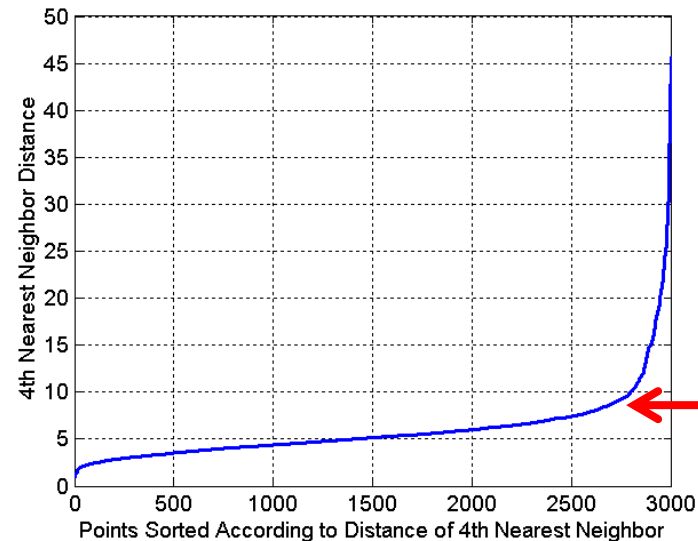


DBSCAN Algorithm

- Label points as **core**, **border** and **noise**
- Eliminate **noise** points
- For every **core** point **p** that has not been assigned to a cluster
 - Create a new cluster with the point **p** and all the points that are **density-connected** to **p**.
- Assign **border** points to the cluster of the closest core point.

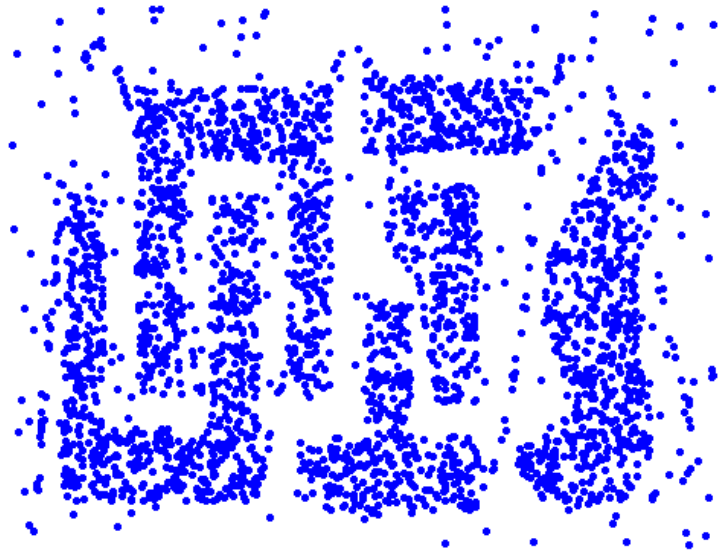
DBSCAN: Determining Eps and MinPts

- Idea: for points in a cluster, their k^{th} nearest neighbors are at roughly the same distance
- Noise points have the k^{th} nearest neighbor at farther distance
- So, plot sorted distance of every point to its k^{th} nearest neighbor
- Find the distance d where there is a “knee” in the curve
 - Eps = d , MinPts = k

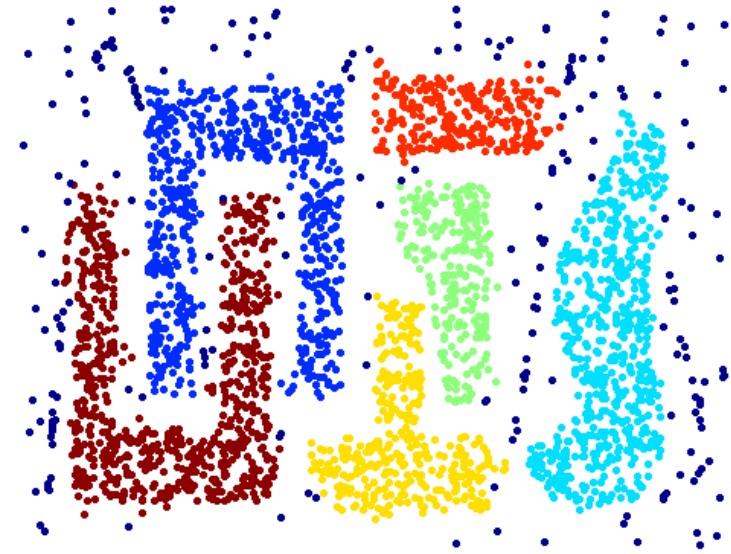


Eps ~ 7-10
MinPts = 4

When DBSCAN Works Well



Original Points



Clusters

- Resistant to Noise
- Can handle clusters of different shapes and sizes

DBSCAN: Sensitive to Parameters

Figure 8. DBScan results for DS1 with MinPts at 4 and Eps at (a) 0.5 and (b) 0.4.

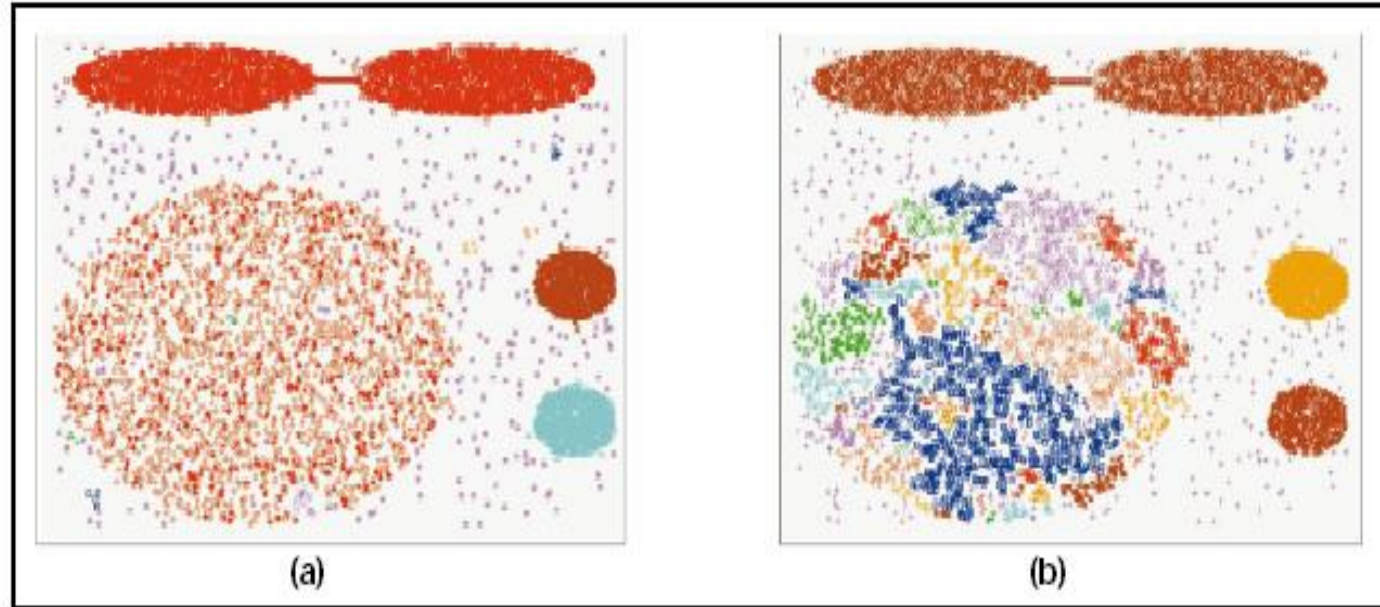
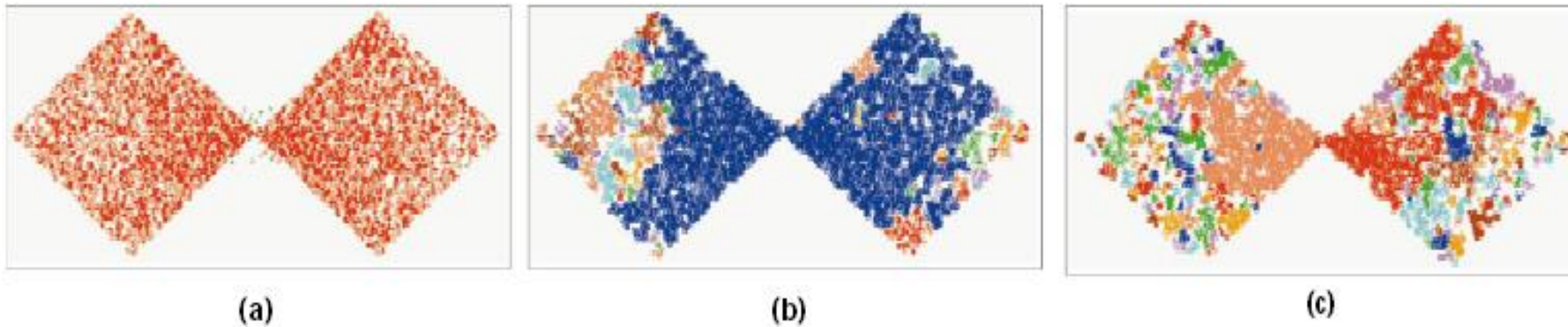
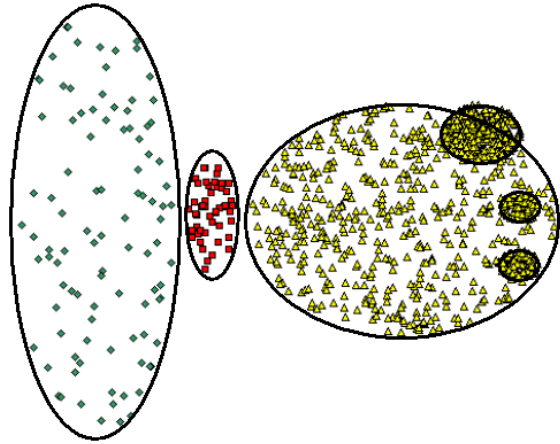


Figure 9. DBScan results for DS2 with MinPts at 4 and Eps at (a) 5.0, (b) 3.5, and (c) 3.0.

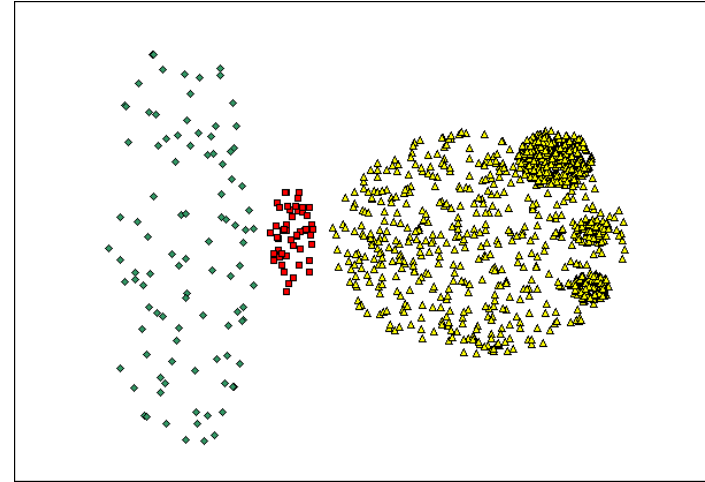


When DBSCAN Does NOT Work Well

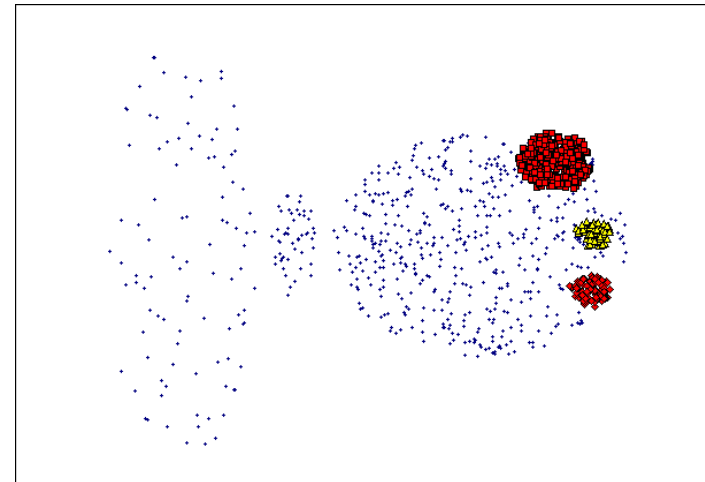


Original Points

- Varying densities
- High-dimensional data



(MinPts=4, Eps=9.75).



(MinPts=4, Eps=9.92)

Other algorithms

- **PAM, CLARANS**: Solutions for the **k-medoids** problem
- **BIRCH**: Constructs a **hierarchical tree** that acts a summary of the data, and then clusters the leaves.
- **MST**: Clustering using the **Minimum Spanning Tree**.
- **ROCK**: clustering **categorical data** by neighbor and link analysis
- **LIMBO, COOLCAT**: Clustering **categorical data** using **information theoretic** tools.
- **CURE**: **Hierarchical** algorithm uses different representation of the cluster
- **CHAMELEON**: **Hierarchical** algorithm uses **closeness and interconnectivity** for merging

CLUSTERING EVALUATION

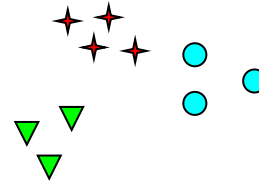
Clustering Evaluation

- We need to evaluate the “goodness” of the resulting clusters?
- But “clustering lies in the eye of the beholder”!

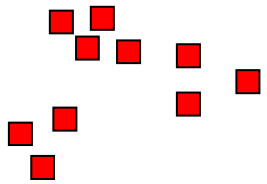
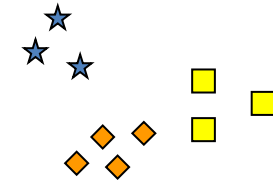
Quality of Clustering can be Ambiguous



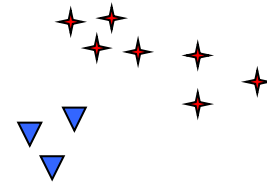
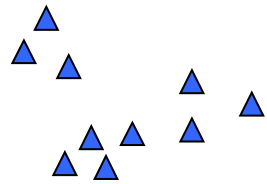
How many clusters?



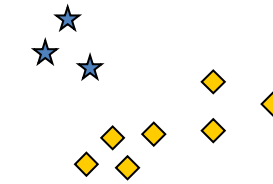
Six Clusters



Two Clusters



Four Clusters



Clustering Evaluation

- Then why do we want to evaluate them?
 - To avoid finding patterns in noise
 - To compare clusterings, or clustering algorithms
 - To compare against a “ground truth”

Different Aspects of Cluster Validation

1. **Internal Evaluation**: Evaluating how well the results of a cluster analysis fit the data *without* reference to external information.
 - Use only the data
2. Determining the **clustering tendency** of a set of data, i.e., distinguishing whether non-random structure actually exists in the data.
3. Determining the **'correct' number of clusters**.
4. **External Evaluation**: Comparing the results of a cluster analysis to externally known results, e.g., to externally given **class labels**.

Metrics for cluster and clustering validity

- For the following we will see some metrics for the clustering validity.
- The metrics can be applied for the evaluation of either a **cluster**, or a **clustering**
- In **cluster validity**, we **evaluate a group of points**, as to whether they were correctly placed together.
 - We usually check if the group is homogeneous (for some notion of homogeneity)
- In **clustering validity**, we **evaluate a collection of clusters**
 - We often use the (weighted) average of cluster validity
 - There are also metrics that look at the relationships between the clusters, e.g., how well-separated the clusters are.

CLUSTER VALIDITY WITH INTERNAL CRITERIA

Internal Measures

- **Internal Index:** A metric used to measure the goodness of a clustering structure without reference to external information
 - Example: Sum of Square Errors (SSE)
- SSE can be used to evaluate a cluster or a clustering:
 - For a cluster of points C_i , the SSE is:

$$SSE(C_i) = \sum_{x \in C_i} (x - c_i)^2, c_i = \text{centroid of cluster } C_i$$

- For a clustering $C = \{C_1, C_2, \dots, C_k\}$, the SSE is:

$$SSE(C) = \sum_i \sum_{x \in C_i} (x - c_i)^2, c_i = \text{centroid of cluster } C_i$$

- SSE can also be used to compare clusters, or clusterings

Cohesion and Separation

- In general, we evaluate clusters and clusterings based on **cohesion** and **separation**
 - **Cluster Cohesion**: Measures how closely related are objects in a cluster
 - **Cluster Separation**: Measure how distinct or well-separated a cluster is from other clusters

- Example: Squared Error

- **Cohesion** is measured by the **within cluster sum of squares** (SSE)

$$WSS = \sum_i \sum_{x \in C_i} (x - c_i)^2$$

We want this to be small

- **Separation** is measured by the **sum of square error of the centroids**

$$BSS = \sum_i m_i (c - c_i)^2$$

We want this to be large

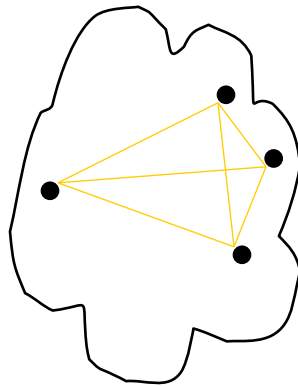
- Where m_i is the size of cluster i , c the **overall mean**. It also holds that:

$$BSS = \sum_{x \in C_i} \sum_{y \in C_j} (x - y)^2$$

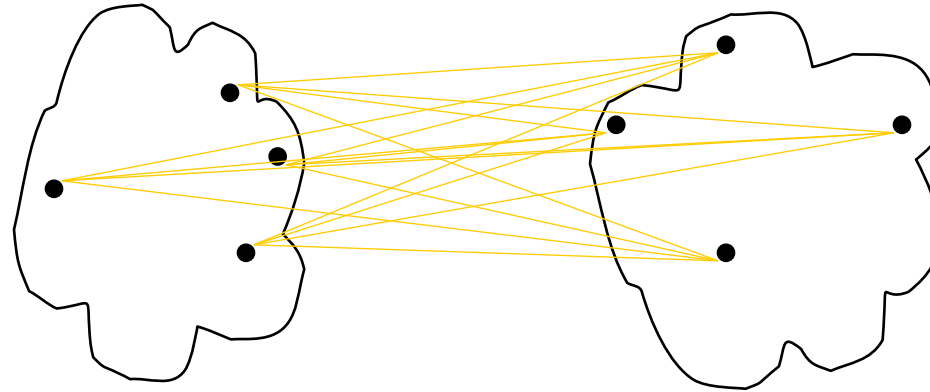
- Interesting observation: $WSS + BSS = \text{constant}$

Cohesion and Separation

- A **proximity-graph**-based approach can also be used for cohesion and separation.
 - Cluster cohesion is the sum of the weight of all links within a cluster.
 - Cluster separation is the sum of the weights between nodes in the cluster and nodes outside the cluster.



cohesion

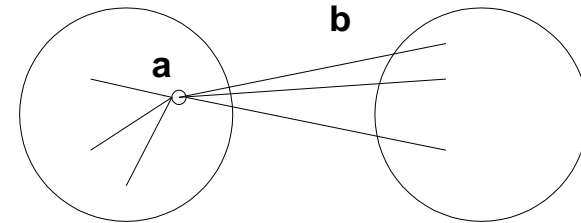


separation

Silhouette Coefficient

- Silhouette Coefficient combines ideas of both cohesion and separation, but for individual points, as well as clusters and clusterings
- For an individual **point i**
 - Calculate a_i = average distance of i to the points in its own cluster
 - Calculate b_i = min (over clusters) of the average distance of i to points in other clusters
 - The silhouette coefficient for a **point i** is then given by

$$s_i = 1 - a_i/b_i$$



- Typically, between 0 and 1, the closer to 1 the better.
 - Can be less than 0 but this is a problematic case
- We can calculate the Average Silhouette coefficient of the points for a cluster, or for a clustering

Silhouette Coefficient Example

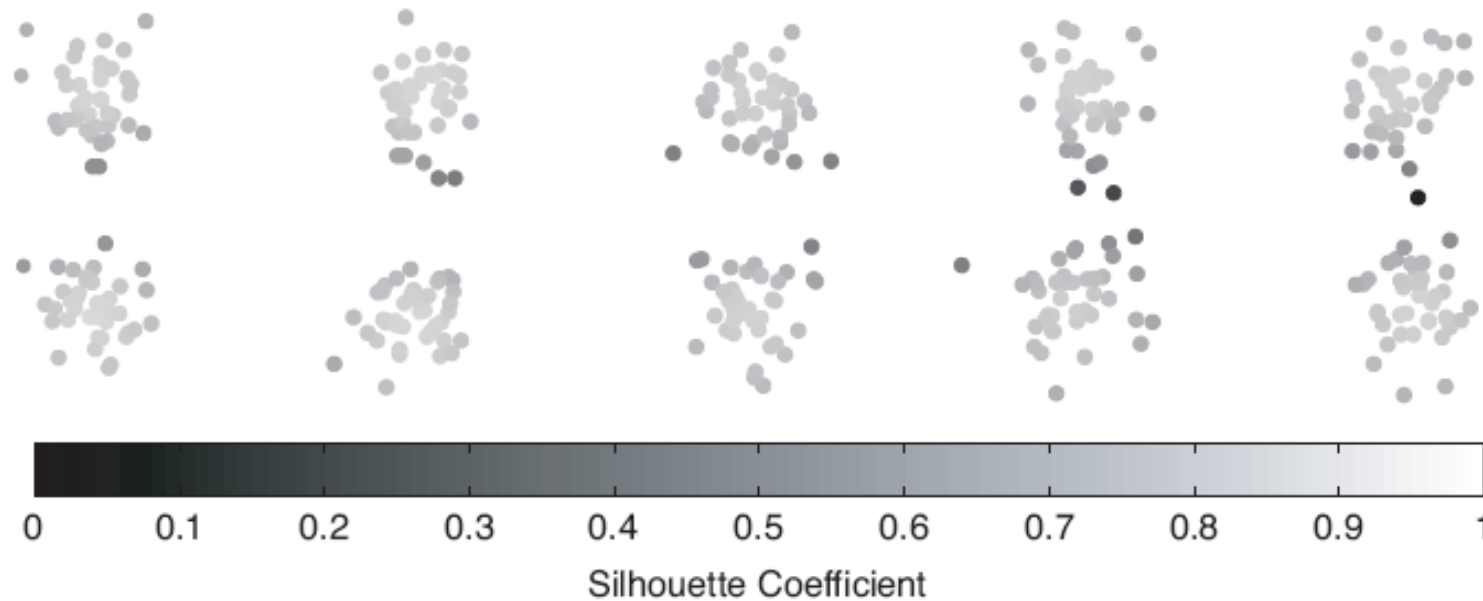


Figure 8.29. Silhouette coefficients for points in ten clusters.

Measuring Cluster Validity Via Correlation

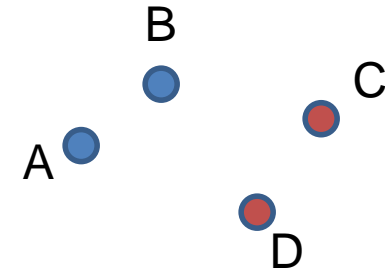
■ Two matrices

■ Similarity or Distance Matrix

- One row and one column for each data point
- An entry is the similarity or distance of the associated pair of points

■ “Incidence” Matrix

- One row and one column for each data point
- An entry is 1 if the associated pair of points belong to the same cluster
- An entry is 0 if the associated pair of points belongs to different clusters



$$D = \begin{matrix} & \begin{matrix} A & B & C & D \end{matrix} \\ \begin{matrix} A \\ B \\ C \\ D \end{matrix} & \begin{bmatrix} 0 & 0.9 & 2.2 & 1.5 \\ 0.9 & 0 & 1.2 & 1.7 \\ 2.2 & 1.2 & 0 & 1.1 \\ 1.5 & 1.7 & 1.1 & 0 \end{bmatrix} \end{matrix}$$

$$I = \begin{matrix} & \begin{matrix} A & B & C & D \end{matrix} \\ \begin{matrix} A \\ B \\ C \\ D \end{matrix} & \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \end{matrix}$$

Measuring Cluster Validity Via Correlation

- Compute the **correlation** between the two matrices

$$\text{CorrCoeff}(X, Y) = \frac{\sum_i (x_i - \mu_X)(y_i - \mu_Y)}{\sqrt{\sum_i (x_i - \mu_X)^2} \sqrt{\sum_i (y_i - \mu_Y)^2}}$$

- Since the matrices are symmetric, only the correlation between $n(n-1) / 2$ entries needs to be calculated.

- High** correlation (**positive** for similarity, **negative** for distance) indicates that points that belong to the same cluster are close to each other.
- Not a good measure for some density or contiguity-based clusters.

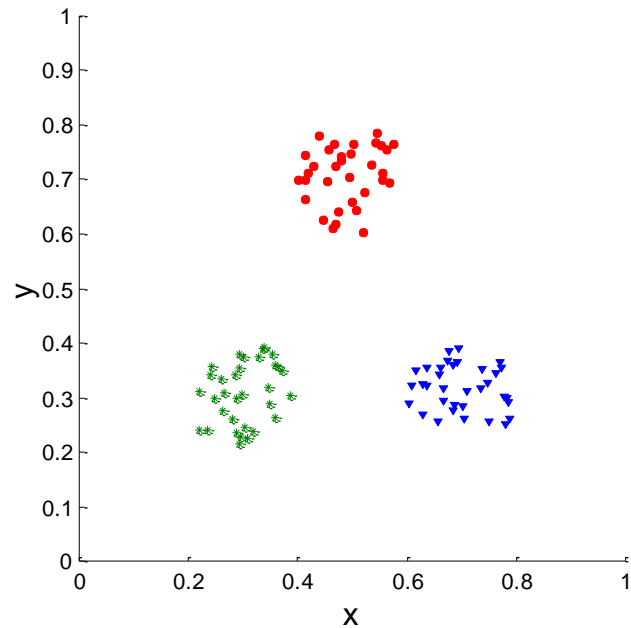
$$S = \begin{matrix} & \begin{matrix} A & B & C & D \end{matrix} \\ \begin{matrix} A \\ B \\ C \\ D \end{matrix} & \begin{bmatrix} 0 & 0.9 & 2.2 & 1.5 \\ 0.9 & 0 & 1.2 & 1.7 \\ 2.2 & 1.2 & 0 & 1.1 \\ 1.5 & 1.7 & 1.1 & 0 \end{bmatrix} \end{matrix}$$

$$I = \begin{matrix} & \begin{matrix} A & B & C & D \end{matrix} \\ \begin{matrix} A \\ B \\ C \\ D \end{matrix} & \begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix} \end{matrix}$$

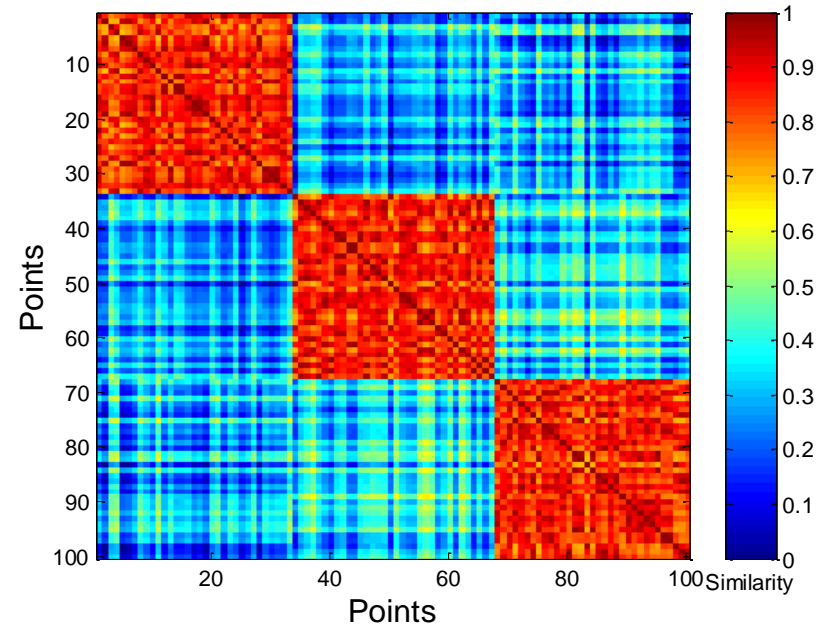
$$\begin{aligned} \text{CorrCoeff}([0.9, 2.2, 1.5, 1.2, 1.7, 1.1], \\ [1, 0, 0, 0, 0, 1]) \\ = -0.71 \end{aligned}$$

Using Similarity Matrix for Cluster Validation

- Order the **similarity** matrix with respect to cluster labels and inspect visually.



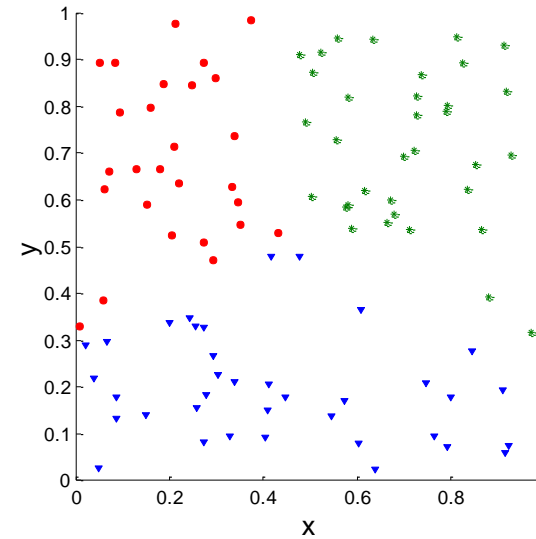
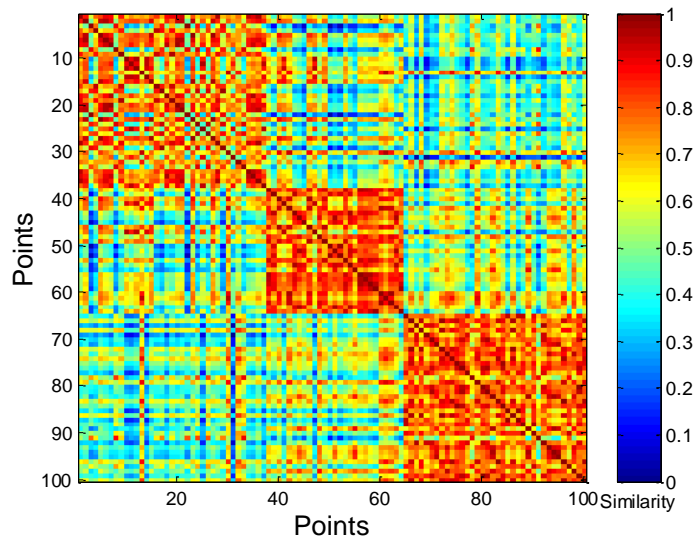
Corr = -0.9235



$$sim(i,j) = 1 - \frac{d_{ij} - d_{min}}{d_{max} - d_{min}}$$

Using Similarity Matrix for Cluster Validation

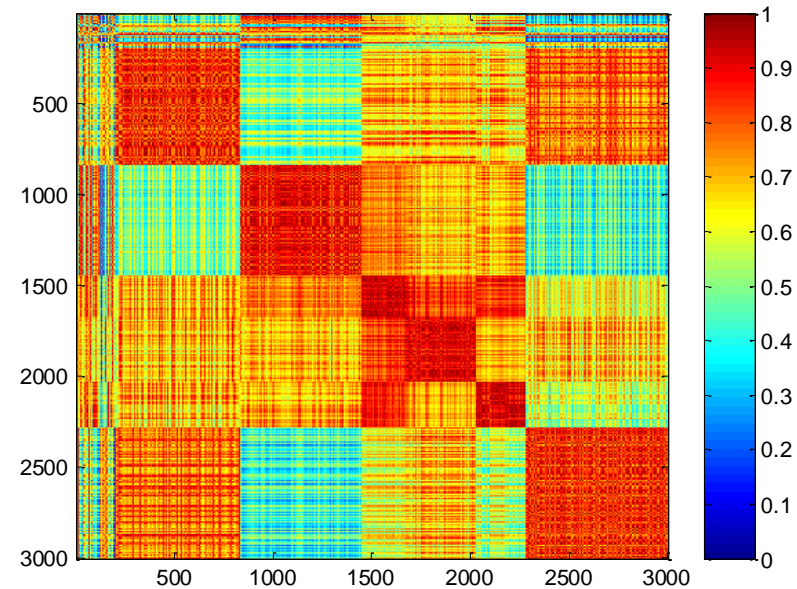
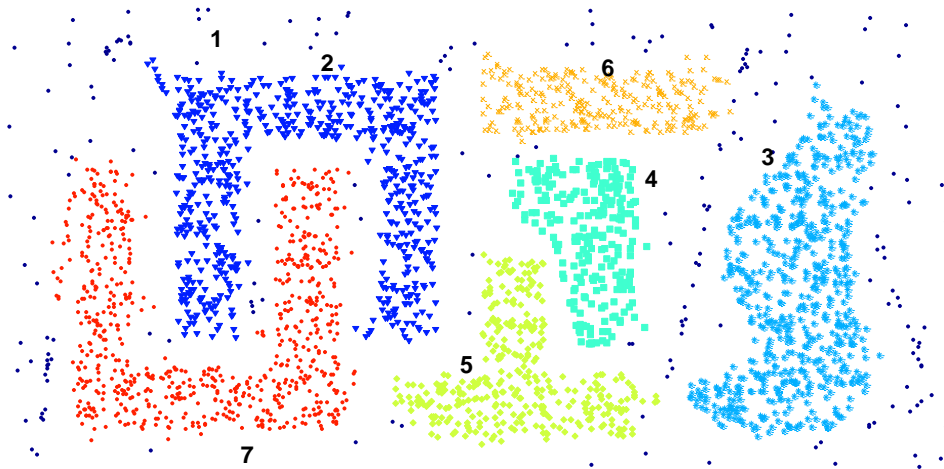
- Clusters in random data are not so crisp



Corr = -0.5810

K-means

Using Similarity Matrix for Cluster Validation



DBSCAN

- Clusters in more complicated figures are not well separated
- This technique can only be used for small datasets since it requires a quadratic computation

Internal measures – caveats

- Internal measures have the problem that the clustering algorithm **did not set out to optimize this measure**, so it will not necessarily do well with respect to the measure.
 - Essentially, we check whether one criterion correlates well with another
- An internal measure can also be used as an **objective function** for clustering
- The algorithm that optimizes this criterion is expected to do well.

STATISTICAL FRAMEWORK FOR CLUSTER(ING) VALIDITY

Framework for Cluster Validity

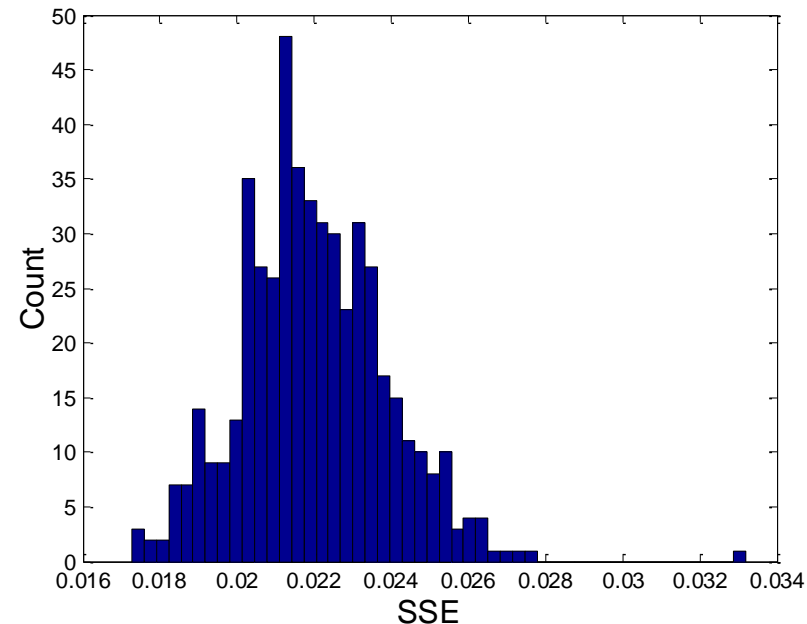
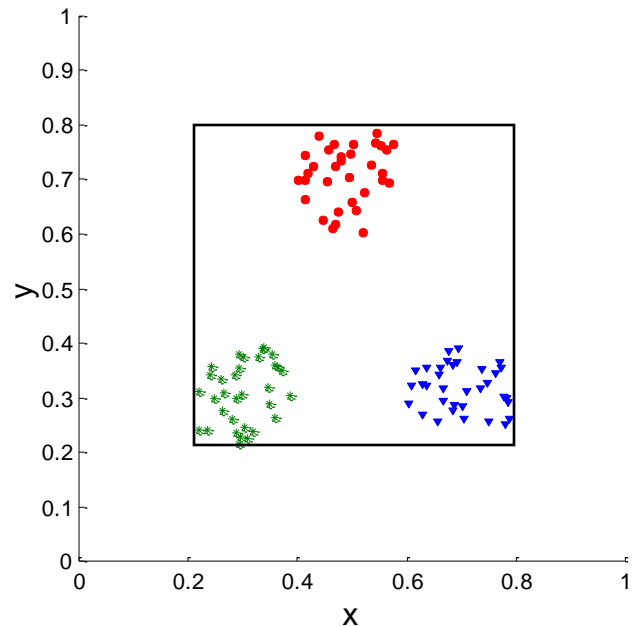
- Need a **framework** to interpret any measure.
 - For example, if our measure of evaluation has the value, 10, is that good, fair, or poor?
- **Statistics** provide a framework for cluster validity
 - The more “**non-random**” a clustering result is, the more likely it represents valid structure in the data
 - Can compare the index value for a clustering with the values of the index that result from **clustering random data**, or from **random clusterings**.
 - If the value of the index is **unlikely**, then the clustering results are valid
 - Comparing against clustering of random data tells us if there is valid clustering structure in the data
 - Comparing against random clusterings tells us if the clustering algorithm is meaningful
 - Although a random clustering is a weak alternative.
- For comparing the results of two different clusterings, a framework is less necessary, but we may want to know whether the difference between two index values is **significant**

Statistical Framework for SSE

- Example

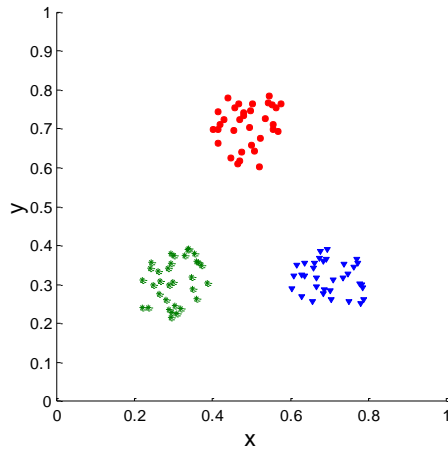
- Compare SSE of 0.005 against three clusters in random data
- Histogram of SSE for three clusters in 500 random data sets of 100 random points distributed in the range 0.2 – 0.8 for x and y
 - Value 0.005 is very unlikely

SSE = 0.005

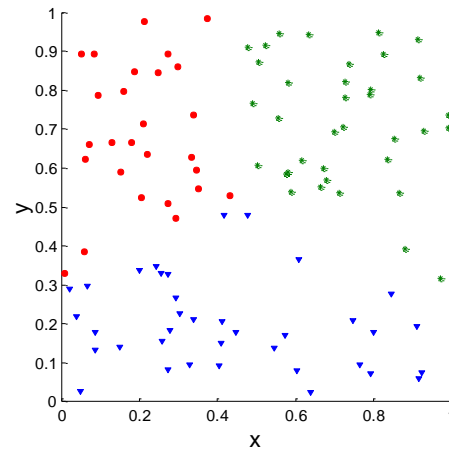


Statistical Framework for Correlation

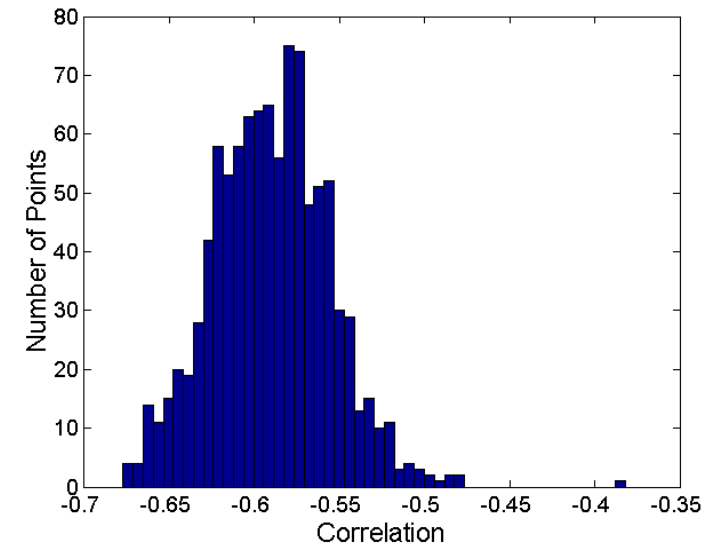
- Correlation of incidence and proximity matrices for the K-means clusterings of the following two data sets.



Corr = -0.9235



Corr = -0.5810



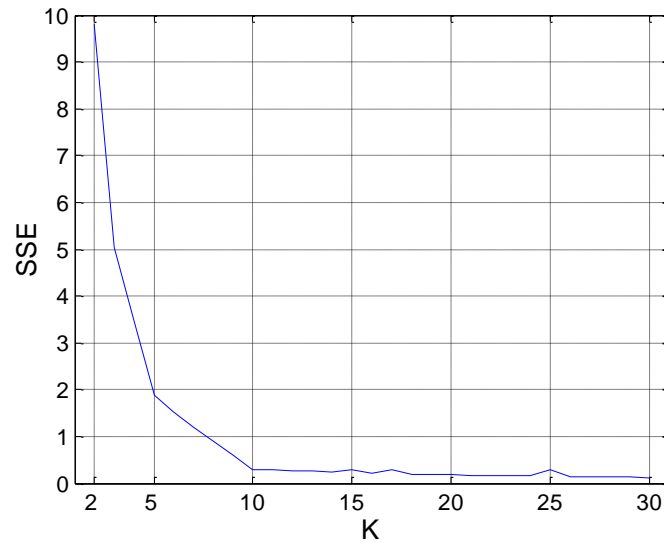
Empirical p-value

- What we do is similar to a **permutation test**:
- We have a **measurement v** (e.g., the SSE value)
- We compute **N** measurements on **random datasets**
- We compute the **empirical p-value** as the **fraction** of measurements in the random data that have value **less or equal** than value **v** (or greater or equal if we want to maximize)
 - i.e., the value in the random dataset is **at least as good** as that in the real data
- We usually require that **$p\text{-value} \leq 0.05$**
- **Hard question**: what is the right notion of a random dataset?

ESTIMATING THE “RIGHT” NUMBER OF CLUSTERS

Estimating the “right” number of clusters

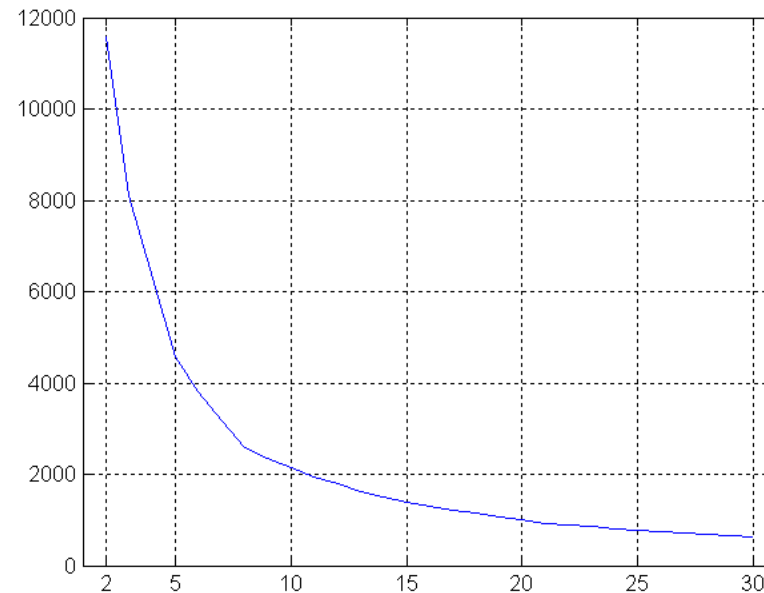
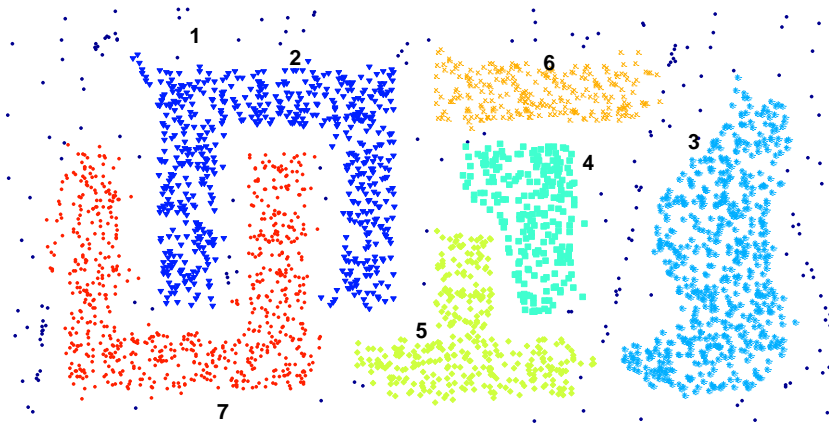
- Typical approach: find a “knee” in an internal measure curve.



- Question: why not the k that **minimizes** the SSE?
 - Forward reference: minimize a measure, but with a “**simple**” clustering
- **Desirable property**: the clustering algorithm does not require the number of clusters to be specified (e.g., DBSCAN)

Estimating the “right” number of clusters

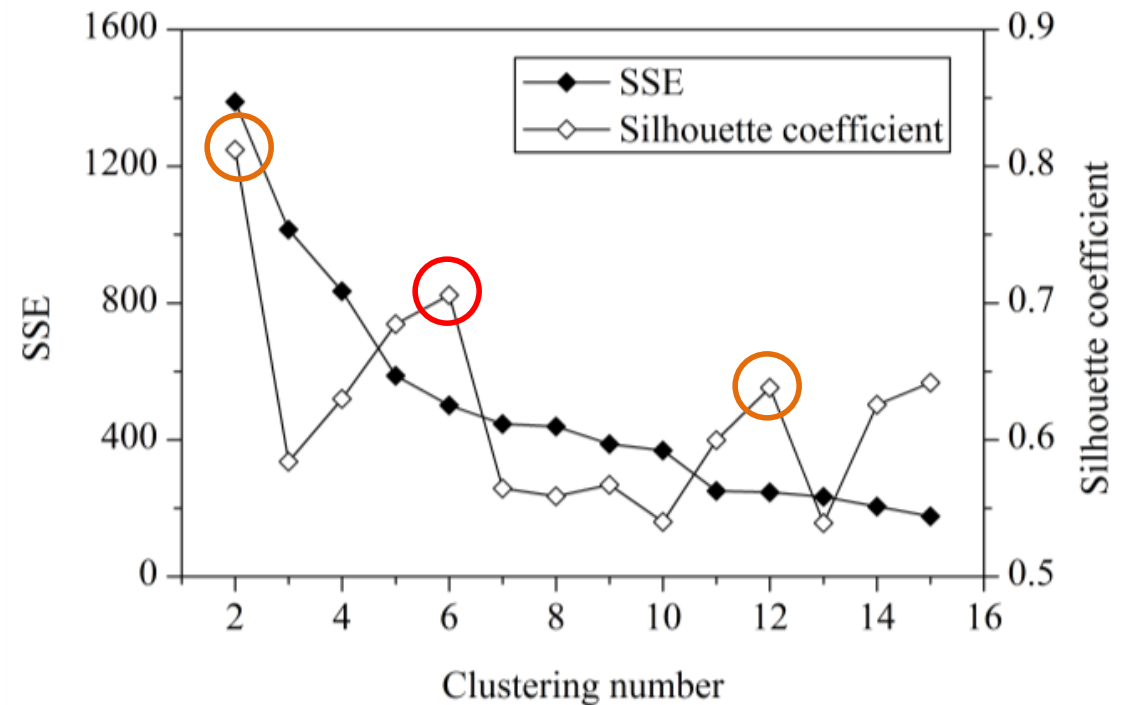
- SSE curve for a more complicated data set



SSE of clusters found using K-means

Estimating the “right” number of clusters

- A metric that is better suited for this task is the **average silhouette coefficient** which does not change monotonically with the number of clusters
- In this example 6 seems to be a good number of clusters since it has high silhouette coefficient and low SSE
- 2 has the highest silhouette coefficient but highest SSE.
- 12 could be another alternative



EVALUATION WITH EXTERNAL “GROUND TRUTH”

External Measures for Clustering Validity

- Assume that the data is **labeled** with some class labels
 - E.g., **documents** are classified into **topics**, **people** classified according to their **income**, **politicians** classified according to the **political party**.
 - This is called the “**ground truth**”
- In this case we want the clusters to be **homogeneous** with respect to classes
 - **Each cluster** should contain elements of **mostly one class**
 - **Each class** should ideally be assigned to a **single cluster**
- This does not always make sense
 - **Clustering** is not the same as **classification**
 - ...but this is what people use most of the time

Confusion/Contingency matrix

- Rows: clusters
- Columns: classes
- Entries: counts/probability of cluster-class pair
- n = number of points
- m_i = points in cluster i
- c_j = points in class j
- n_{ij} = points in cluster i coming from class j

- The confusion/contingency matrix is sometimes used for evaluation as is
 - It gives us the mapping between the clusters and ground truth classes

Confusion/Contingency matrix of clusters and classes (counts)

| | Class 1 | Class 2 | Class 3 | |
|-----------|----------|----------|----------|-------|
| Cluster 1 | n_{11} | n_{12} | n_{13} | m_1 |
| Cluster 2 | n_{21} | n_{22} | n_{23} | m_2 |
| Cluster 3 | n_{31} | n_{32} | n_{33} | m_3 |
| | c_1 | c_2 | c_3 | n |

Example

| | Class 1 | Class 2 | Class 3 | |
|-----------|---------|---------|---------|-----|
| Cluster 1 | 2 | 3 | 85 | 90 |
| Cluster 2 | 90 | 12 | 8 | 110 |
| Cluster 3 | 8 | 85 | 7 | 100 |
| | 100 | 100 | 100 | 300 |

Measures of cluster homogeneity

- Compute probabilities:

$$p_{ij} = \frac{n_{ij}}{m_i}$$

- The probability that a randomly selected point from cluster i comes from class j .
- Probabilities of rows sum to 1

- **Purity:**

- Of a cluster i : $p_i = \max_j p_{ij}$
- Of a clustering: $p(C) = \sum_{i=1}^K \frac{m_i}{n} p_i$

- **Entropy:**

- Of a cluster i : $e_i = -\sum_{j=1}^L p_{ij} \log p_{ij}$
 - Highest when uniform, zero when single class
- Of a clustering: $e = \sum_{i=1}^K \frac{m_i}{n} e_i$

| | Class 1 | Class 2 | Class 3 | |
|-----------|----------|----------|----------|-------|
| Cluster 1 | p_{11} | p_{12} | p_{13} | m_1 |
| Cluster 2 | p_{21} | p_{22} | p_{23} | m_2 |
| Cluster 3 | p_{31} | p_{32} | p_{33} | m_3 |
| | c_1 | c_2 | c_3 | n |

| | Class 1 | Class 2 | Class 3 | |
|-----------|---------|---------|---------|-----|
| Cluster 1 | 0.02 | 0.03 | 0.95 | 90 |
| Cluster 2 | 0.82 | 0.11 | 0.07 | 110 |
| Cluster 3 | 0.08 | 0.85 | 0.07 | 100 |
| | 100 | 100 | 100 | 300 |

Purity: (0.94, 0.81, 0.85)

– overall 0.86

Entropy: (0.33, 0.85, 0.76)

– overall 0.66

Precision-recall for cluster-class combinations

- **Precision** of **cluster i** with respect to **class j** : $Prec(i, j) = \frac{n_{ij}}{m_i} = p_{ij}$
 - Percentage of the **cluster i** that comes from **class j**
- **Recall** of **cluster i** with respect to **class j** : $Rec(i, j) = \frac{n_{ij}}{c_j}$
 - Percentage of **class j** that goes to **cluster i**

| | Class 1 | Class 2 | Class 3 | |
|-----------|---------|---------|---------|-----|
| Cluster 1 | 2 | 3 | 85 | 90 |
| Cluster 2 | 90 | 12 | 8 | 110 |
| Cluster 3 | 8 | 85 | 7 | 100 |
| | 100 | 100 | 100 | 300 |

| | Class 1 | Class 2 | Class 3 |
|-----------|---------|---------|---------|
| Cluster 1 | 0.02 | 0.03 | 0.95 |
| Cluster 2 | 0.82 | 0.11 | 0.07 |
| Cluster 3 | 0.08 | 0.85 | 0.07 |

Precision Table

| | Class 1 | Class 2 | Class 3 |
|-----------|---------|---------|---------|
| Cluster 1 | 0.02 | 0.03 | 0.85 |
| Cluster 2 | 0.90 | 0.12 | 0.08 |
| Cluster 3 | 0.08 | 0.85 | 0.07 |

Recall Table

Precision/Recall for clusters and clusterings

- Assign to cluster i the class k_i such that $k_i = \arg \max_j n_{ij}$

- **Precision:**

- Of cluster i : $Prec(i) = \frac{n_{ik_i}}{m_i}$

- Of the clustering: $Prec(C) = \sum_i \frac{m_i}{n} Prec(i)$

- **Recall:**

- Of cluster i : $Rec(i) = \frac{n_{ik_i}}{c_{k_i}}$

- Of the clustering: $Rec(C) = \sum_i \frac{m_i}{n} Rec(i)$

- **F-measure:**

- **Harmonic Mean** of Precision and Recall

| | Class 1 | Class 2 | Class 3 | |
|-----------|---------|---------|---------|-----|
| Cluster 1 | 2 | 3 | 85 | 90 |
| Cluster 2 | 90 | 12 | 8 | 110 |
| Cluster 3 | 8 | 85 | 7 | 100 |
| | 100 | 100 | 100 | 300 |

Precision: (0.94, 0.81, 0.85)

– overall 0.86

Recall: (0.85, 0.9, 0.85)

- overall 0.87

Good and bad clustering

| | Class 1 | Class 2 | Class 3 | |
|-----------|---------|---------|---------|-----|
| Cluster 1 | 2 | 3 | 85 | 90 |
| Cluster 2 | 90 | 12 | 8 | 110 |
| Cluster 3 | 8 | 85 | 7 | 100 |
| | 100 | 100 | 100 | 300 |

Purity: (0.94, 0.81, 0.85)

– overall 0.86

Precision: (0.94, 0.81, 0.85)

– overall 0.86

Recall: (0.85, 0.9, 0.85)

- overall 0.87

| | Class 1 | Class 2 | Class 3 | |
|-----------|---------|---------|---------|-----|
| Cluster 1 | 20 | 35 | 35 | 90 |
| Cluster 2 | 30 | 42 | 38 | 110 |
| Cluster 3 | 38 | 35 | 27 | 100 |
| | 100 | 100 | 100 | 300 |

Purity: (0.38, 0.38, 0.38)

– overall 0.38

Precision: (0.38, 0.38, 0.38)

– overall 0.38

Recall: (0.35, 0.42, 0.38)

– overall 0.39

Another clustering

| | Class 1 | Class 2 | Class 3 | |
|-----------|---------|---------|---------|-----|
| Cluster 1 | 0 | 0 | 35 | 35 |
| Cluster 2 | 50 | 77 | 38 | 165 |
| Cluster 3 | 38 | 35 | 27 | 100 |
| | 100 | 100 | 100 | 300 |

Cluster 1:

Purity: 1

Precision: 1

Recall: 0.35

External Measures of Cluster Validity: Entropy and Purity

Table 5.9. K-means Clustering Results for LA Document Data Set

| Cluster | Entertainment | Financial | Foreign | Metro | National | Sports | Entropy | Purity |
|---------|---------------|-----------|---------|-------|----------|--------|---------|--------|
| 1 | 3 | 5 | 40 | 506 | 96 | 27 | 1.2270 | 0.7474 |
| 2 | 4 | 7 | 280 | 29 | 39 | 2 | 1.1472 | 0.7756 |
| 3 | 1 | 1 | 1 | 7 | 4 | 671 | 0.1813 | 0.9796 |
| 4 | 10 | 162 | 3 | 119 | 73 | 2 | 1.7487 | 0.4390 |
| 5 | 331 | 22 | 5 | 70 | 13 | 23 | 1.3976 | 0.7134 |
| 6 | 5 | 358 | 12 | 212 | 48 | 13 | 1.5523 | 0.5525 |
| Total | 354 | 555 | 341 | 943 | 273 | 738 | 1.1450 | 0.7203 |

entropy For each cluster, the class distribution of the data is calculated first, i.e., for cluster j we compute p_{ij} , the ‘probability’ that a member of cluster j belongs to class i as follows: $p_{ij} = m_{ij}/m_j$, where m_j is the number of values in cluster j and m_{ij} is the number of values of class i in cluster j . Then using this class distribution, the entropy of each cluster j is calculated using the standard formula $e_j = \sum_{i=1}^L p_{ij} \log_2 p_{ij}$, where the L is the number of classes. The total entropy for a set of clusters is calculated as the sum of the entropies of each cluster weighted by the size of each cluster, i.e., $e = \sum_{j=1}^K \frac{m_j}{m} e_j$, where m_j is the size of cluster j , K is the number of clusters, and m is the total number of data points.

purity Using the terminology derived for entropy, the purity of cluster j , is given by $purity_j = \max p_{ij}$ and the overall purity of a clustering by $purity = \sum_{j=1}^K \frac{m_j}{m} purity_j$.

Final Comment on Cluster Validity

“The validation of clustering structures is the most difficult and frustrating part of cluster analysis.

Without a strong effort in this direction, cluster analysis will remain a black art accessible only to those true believers who have experience and great courage.”

Algorithms for Clustering Data, Jain and Dubes