Cluster Analysis and Anomaly Detection

IE500 Data Mining

UNIVERSITY OF MANNHEIM
Data and Web Science Group

Outline

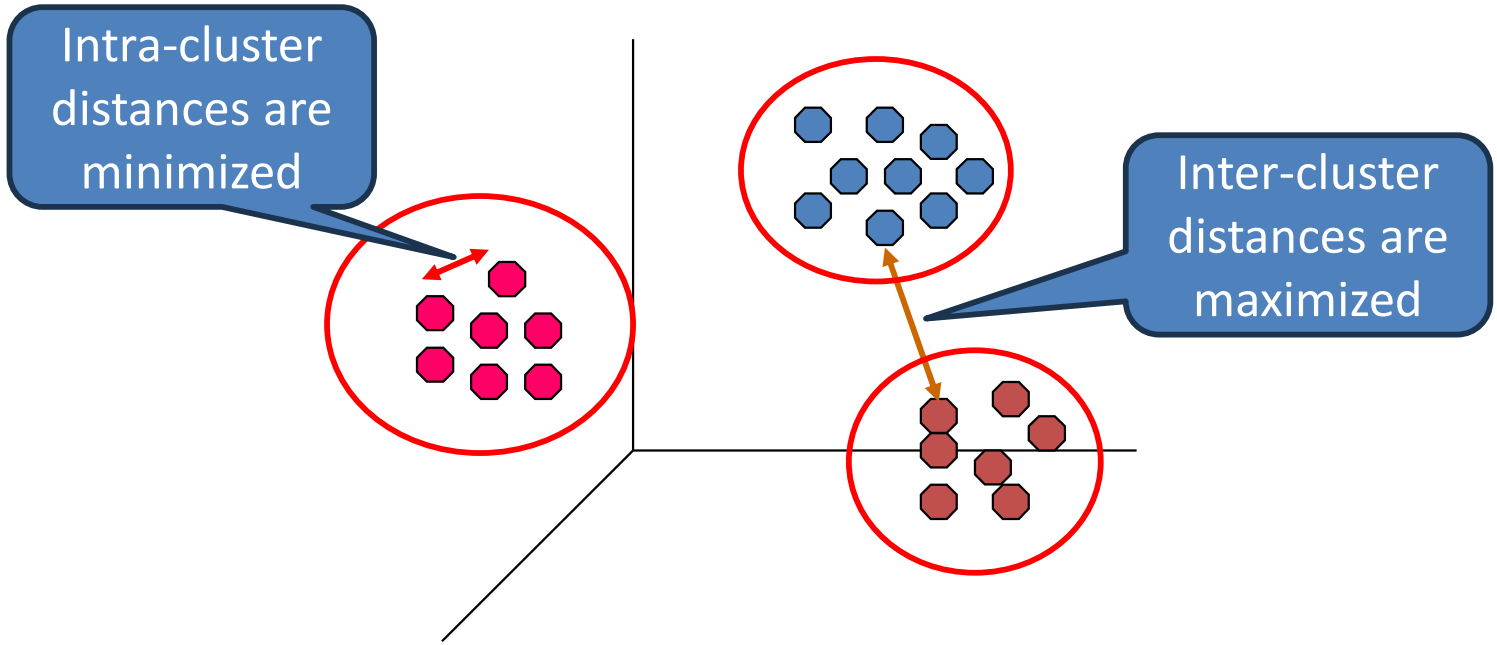
UNIVERSITY OF MANNHEIM
Data and Web Science Group

1. What is Cluster Analysis?
2. K-Means Clustering
   - K-Means and K-Medoids
3. Density-based Clustering
   - DBSCAN
4. Proximity Measures
   - Single Attributes
   - Multiple Attributes
5. Anomaly Detection
   - Statistical Approaches
   - Distance-based Approaches
   - Density-based Approaches

1. What is Cluster Analysis?

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Finding groups of objects such that the objects in a group
  - will be similar to one another
  - and different from the objects in other groups
- Goal: Get a better understanding of the data



Cluster Analysis as Unsupervised Learning

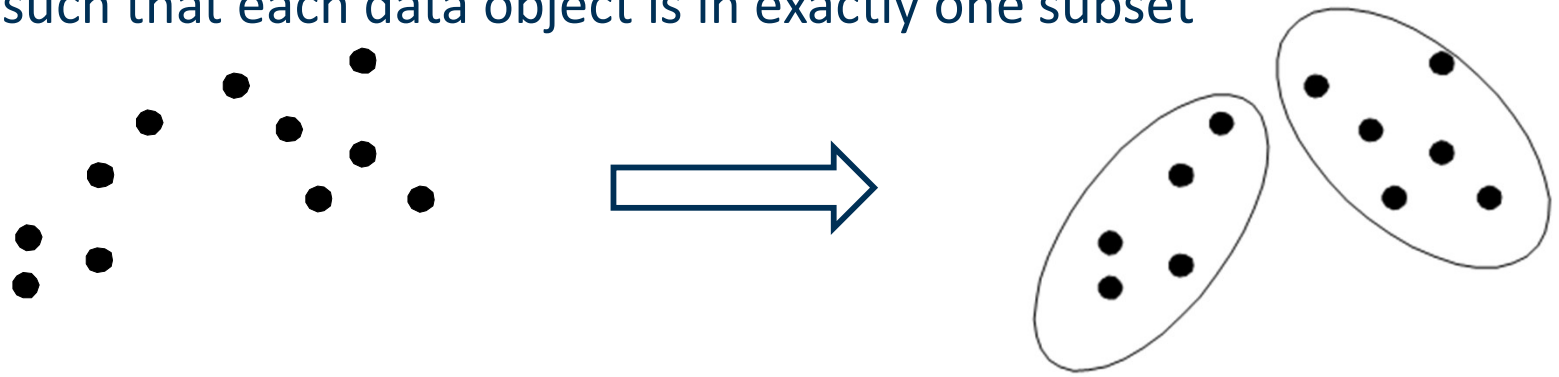
UNIVERSITY OF MANNHEIM
Data and Web Science Group

- **Supervised learning**: Discover patterns in the data that relate data attributes with a target (class) attribute
  - The set of classes is known before
  - Class attributes are usually provided by human annotators
  - Patterns are used for prediction of the target attribute for new data

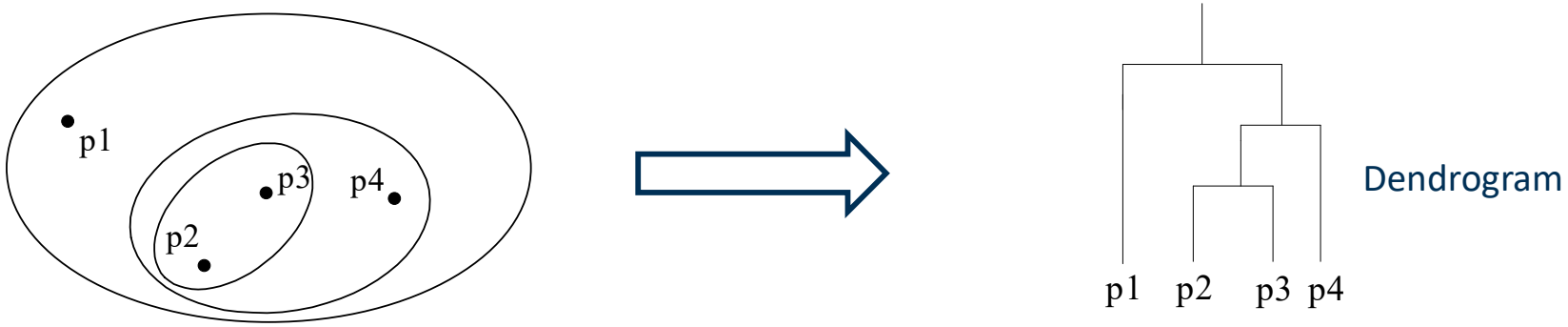
- **Unsupervised learning**: The data has no target attribute
  - We want to explore the data to find some intrinsic structures in it
  - The set of classes/clusters is not known before
  - Cluster Analysis and Association Rule Mining are unsupervised learning tasks

Types of Clusterings

- **Partitional Clustering**
  - A division of data objects into non-overlapping subsets (clusters) such that each data object is in exactly one subset



- **Hierarchical Clustering (see online lecture)**
  - A set of nested clusters organized as a hierarchical tree



Aspects of Cluster Analysis

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- **A clustering algorithm**
  - Partitional algorithms
  - Density-based algorithms
  - Hierarchical algorithms
  - …
- **A proximity (similarity, or dissimilarity) metric**
  - Euclidean distance
  - Cosine similarity
  - Data type-specific similarity metrics
  - Domain-specific similarity metrics
- **Clustering quality**
  - Intra-clusters distance $\Rightarrow$ minimized
  - Inter-clusters distance $\Rightarrow$ maximized
  - The clustering should be useful with regard to the goal of the analysis

The Notion of a Cluster is Ambiguous

How many clusters do you see?

Six Clusters

Two Clusters

Four Clusters

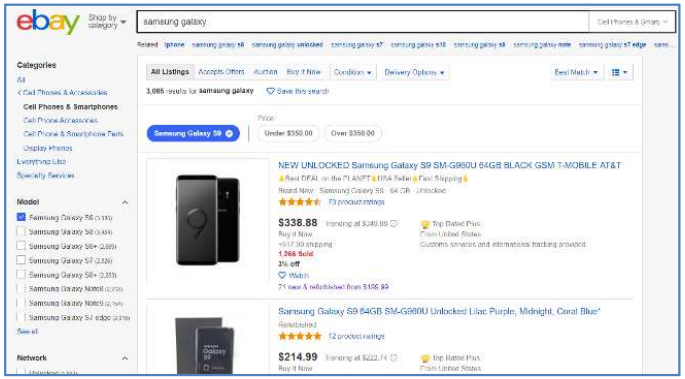
The usefulness of a clustering depends on the **goal of the analysis**

Example Applications

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Market Segmentation
  - goal: Identify groups of similar customers
  - level of granularity depends on the task at hand

- E-Commerce
  - identify offers of the same product on electronic markets



- Image Recognition
  - identify parts of an image that belong to the same object

2. K-Means Clustering

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Partitional clustering algorithm
- 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 beforehand

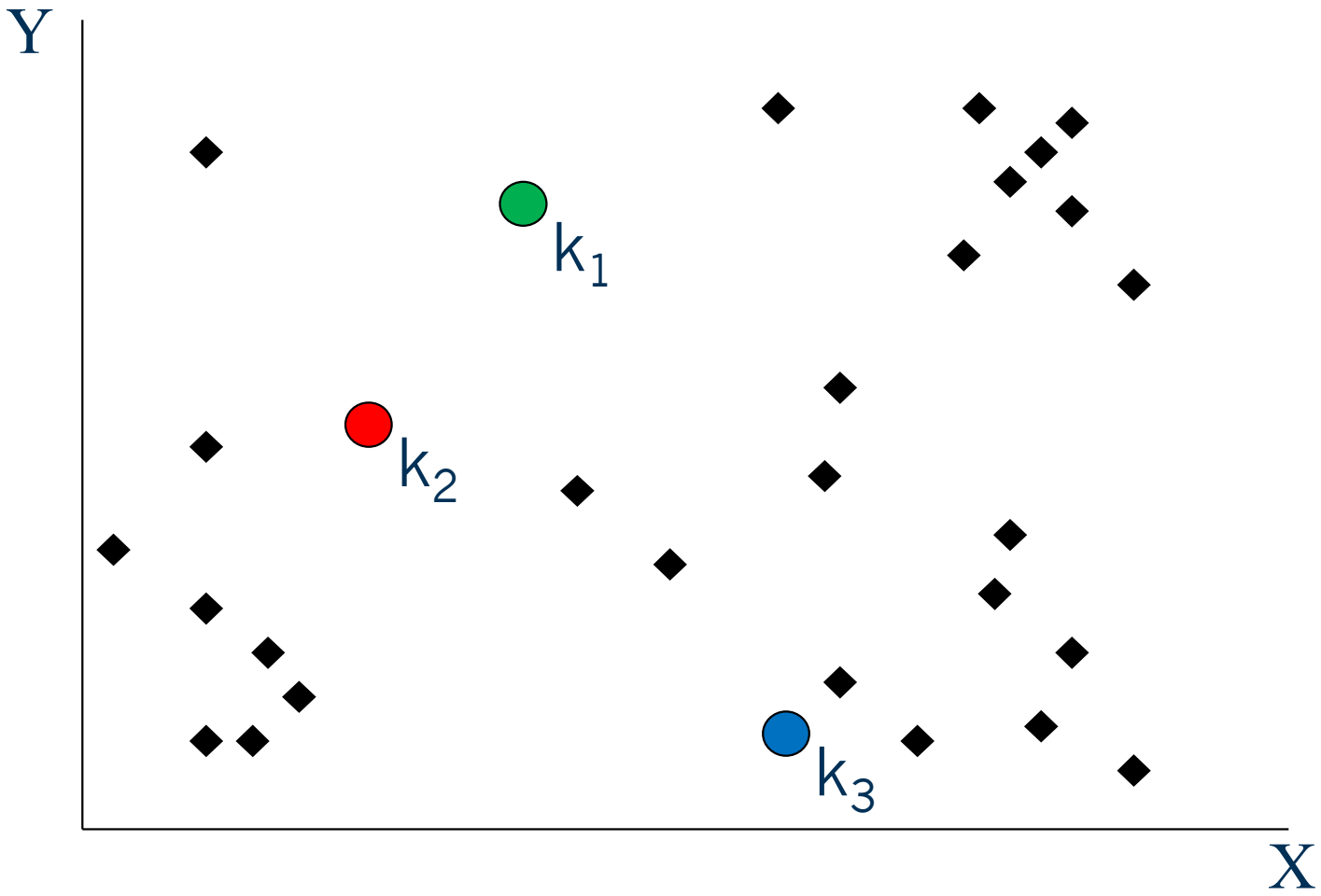
- Algorithm:

1: Select K points as the initial centroids.
2: **repeat**
3: Form K clusters by assigning all points to the closest centroid.
4: Recompute the centroid of each cluster.
5: **until** The centroids don't change

K-Means Example, Step 1

UNIVERSITY OF MANNHEIM
Data and Web Science Group

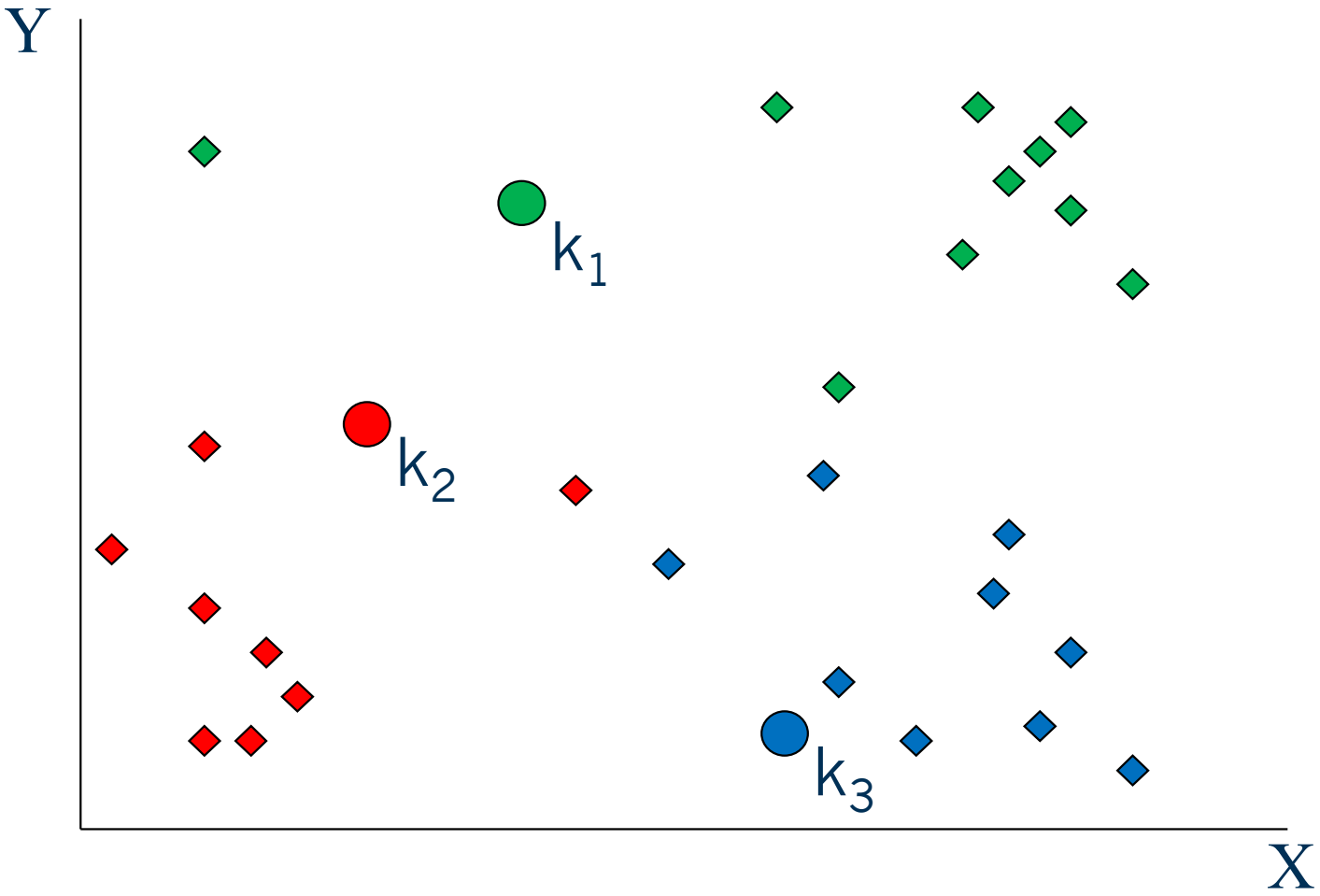
- Randomly pick 3 initial centroids



K-Means Example, Step 3

UNIVERSITY OF MANNHEIM
Data and Web Science Group

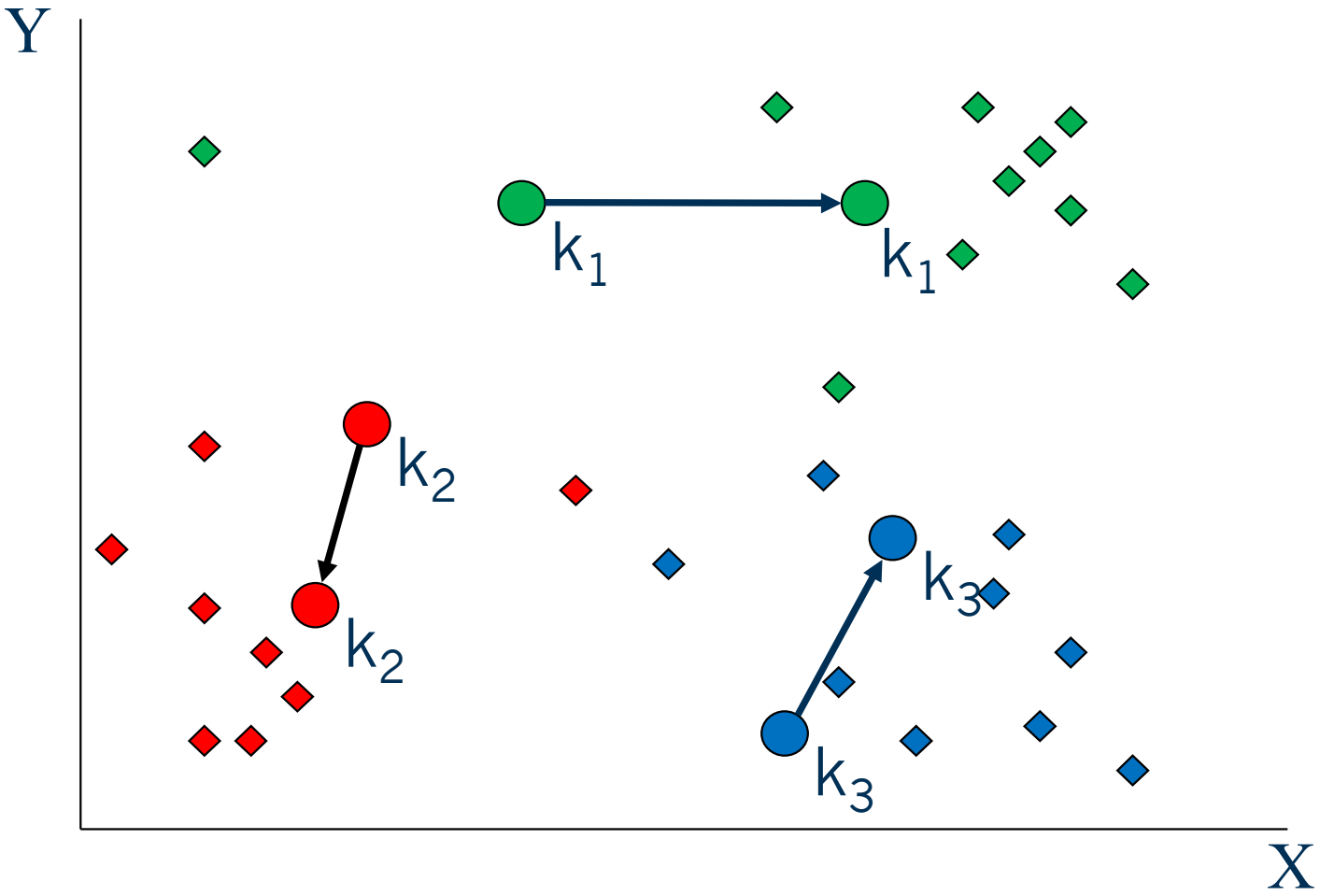
- Assign each point to the closest centroid



K-Means Example, Step 4

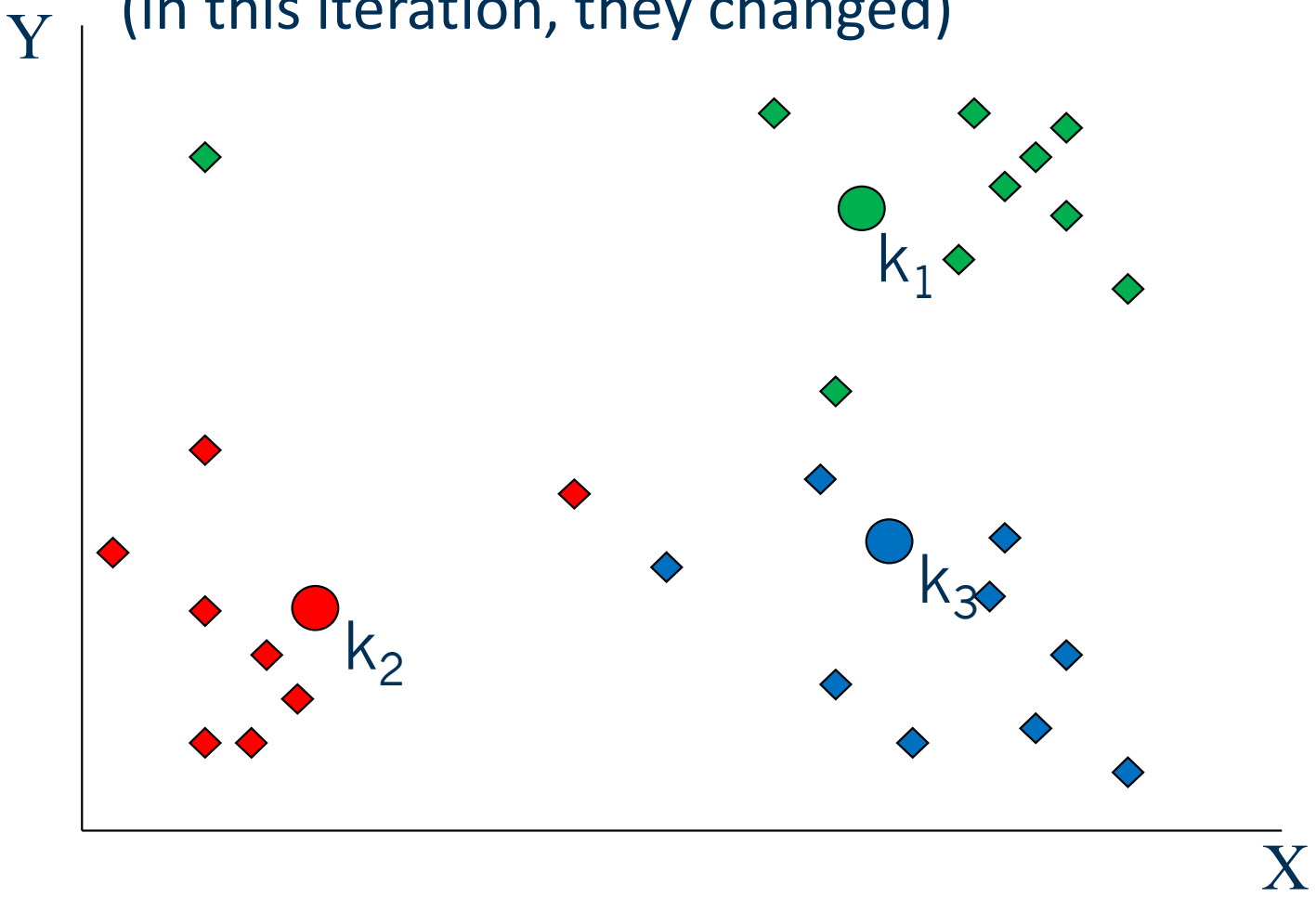
UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Move each centroid to the mean of each cluster



K-Means Example, Step 5

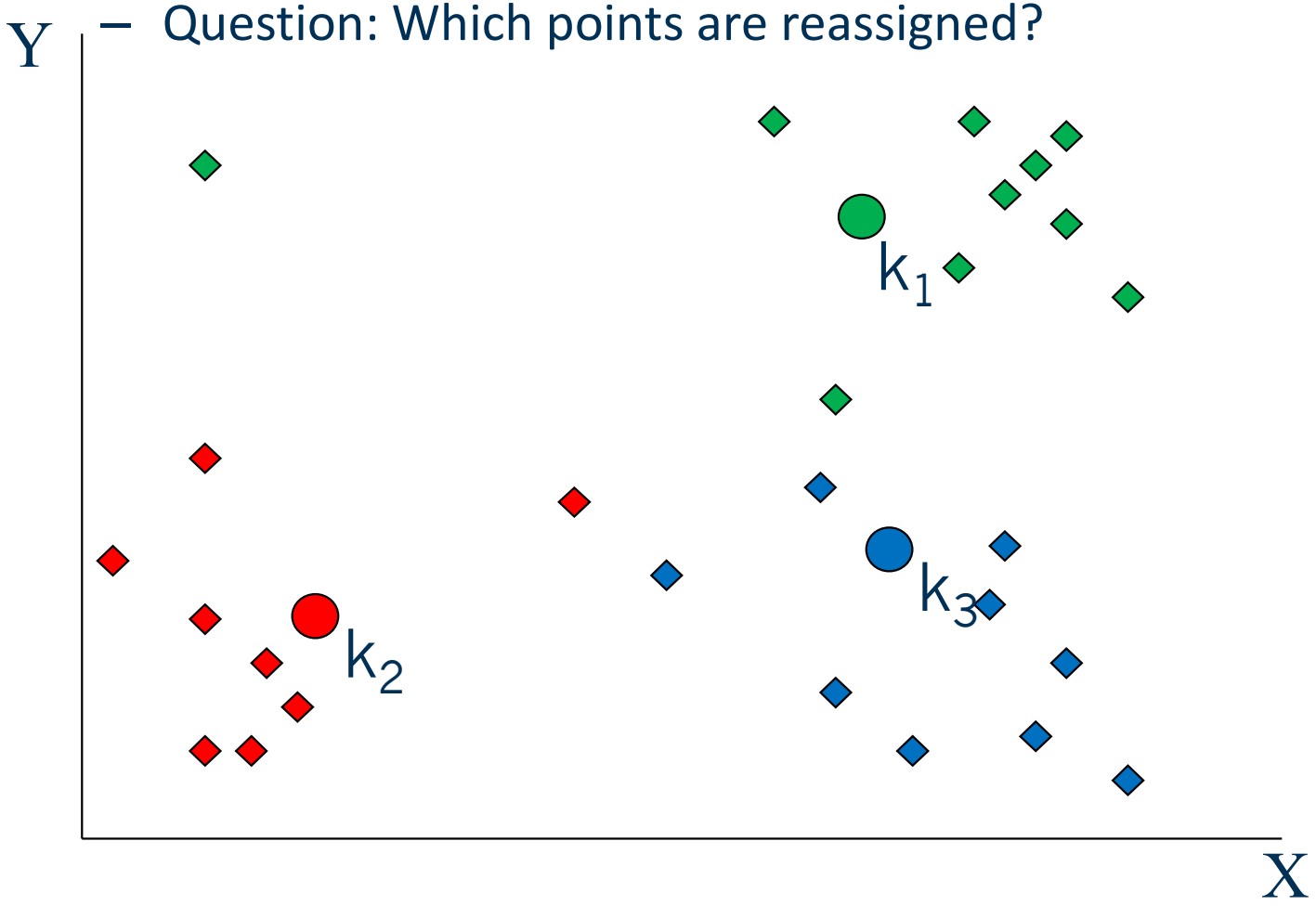
- Repeat until the centroids don't change
  (in this iteration, they changed)



K-Means Example, Repeated Step 3

UNIVERSITY OF MANNHEIM
Data and Web Science Group

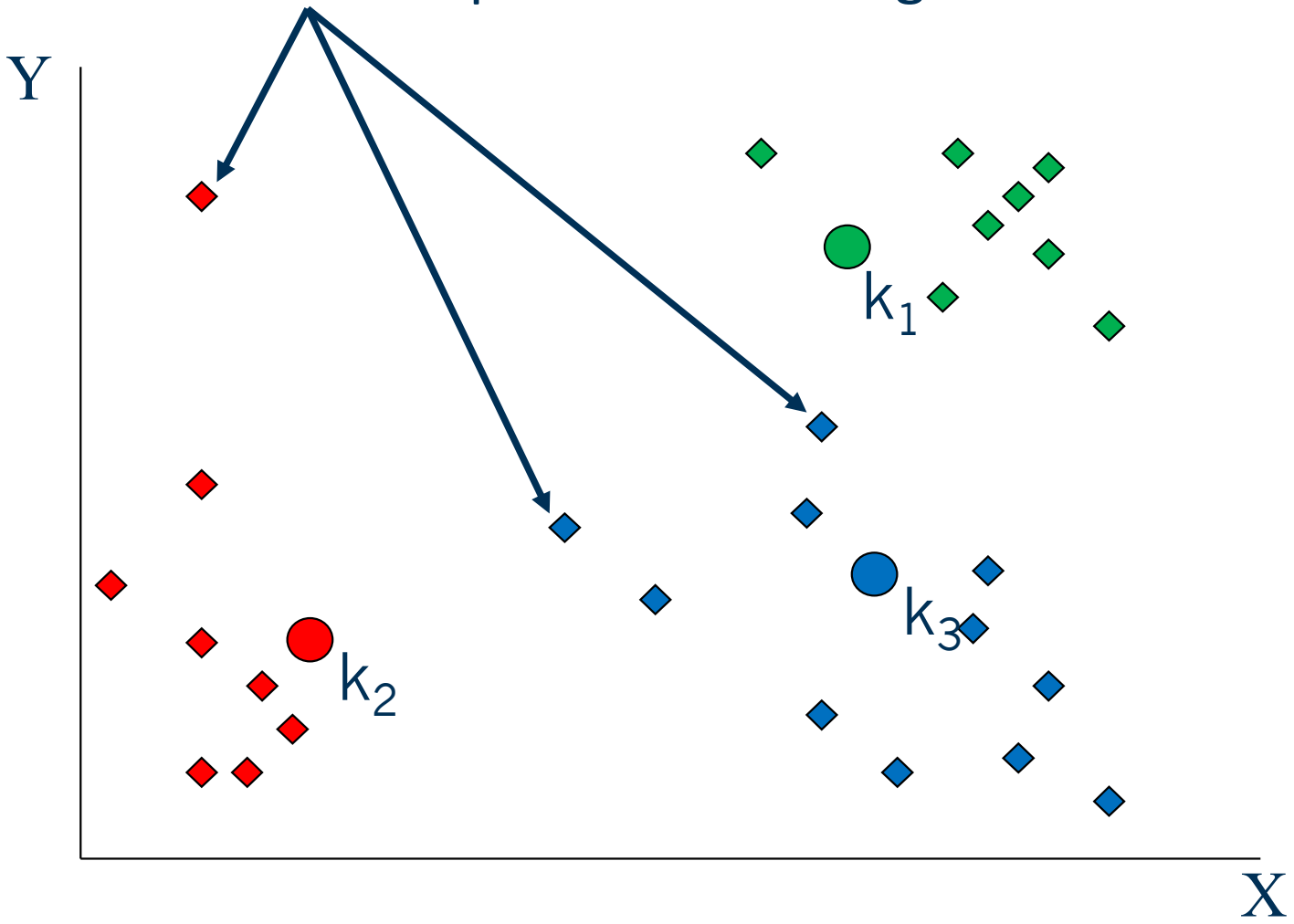
- Reassign points if they are now closer to a different centroid
  - Question: Which points are reassigned?



K-Means Example, Repeated Step 3

UNIVERSITY OF MANNHEIM
Data and Web Science Group

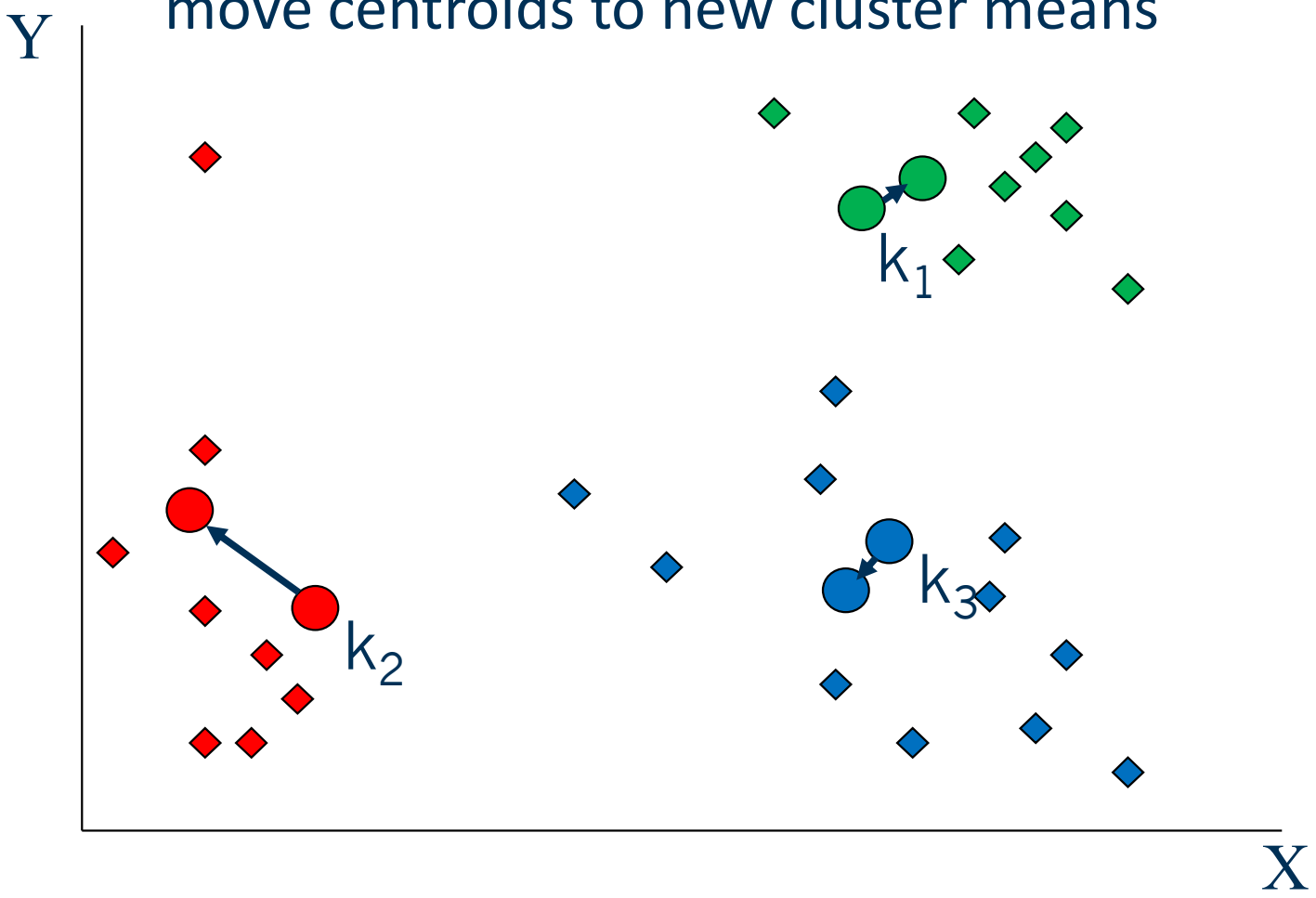
- Answer: Three points are reassigned



K-Means Example, Repeated Step 4

UNIVERSITY OF MANNHEIM
Data and Web Science Group

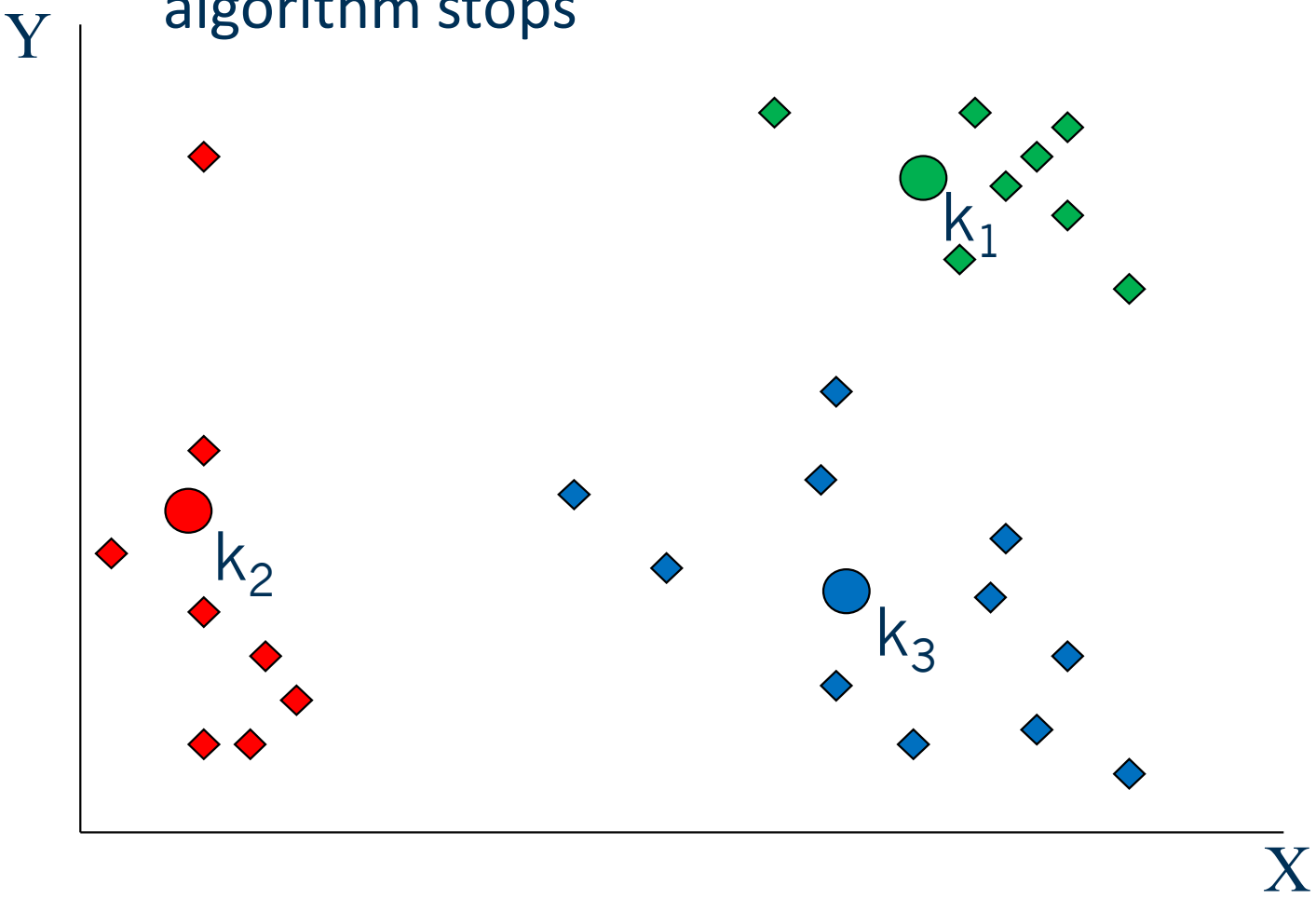
- Re-compute cluster means and move centroids to new cluster means



K-Means Example, Repeated Step 5

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- No points are reassigned, thus centroids don't change and algorithm stops



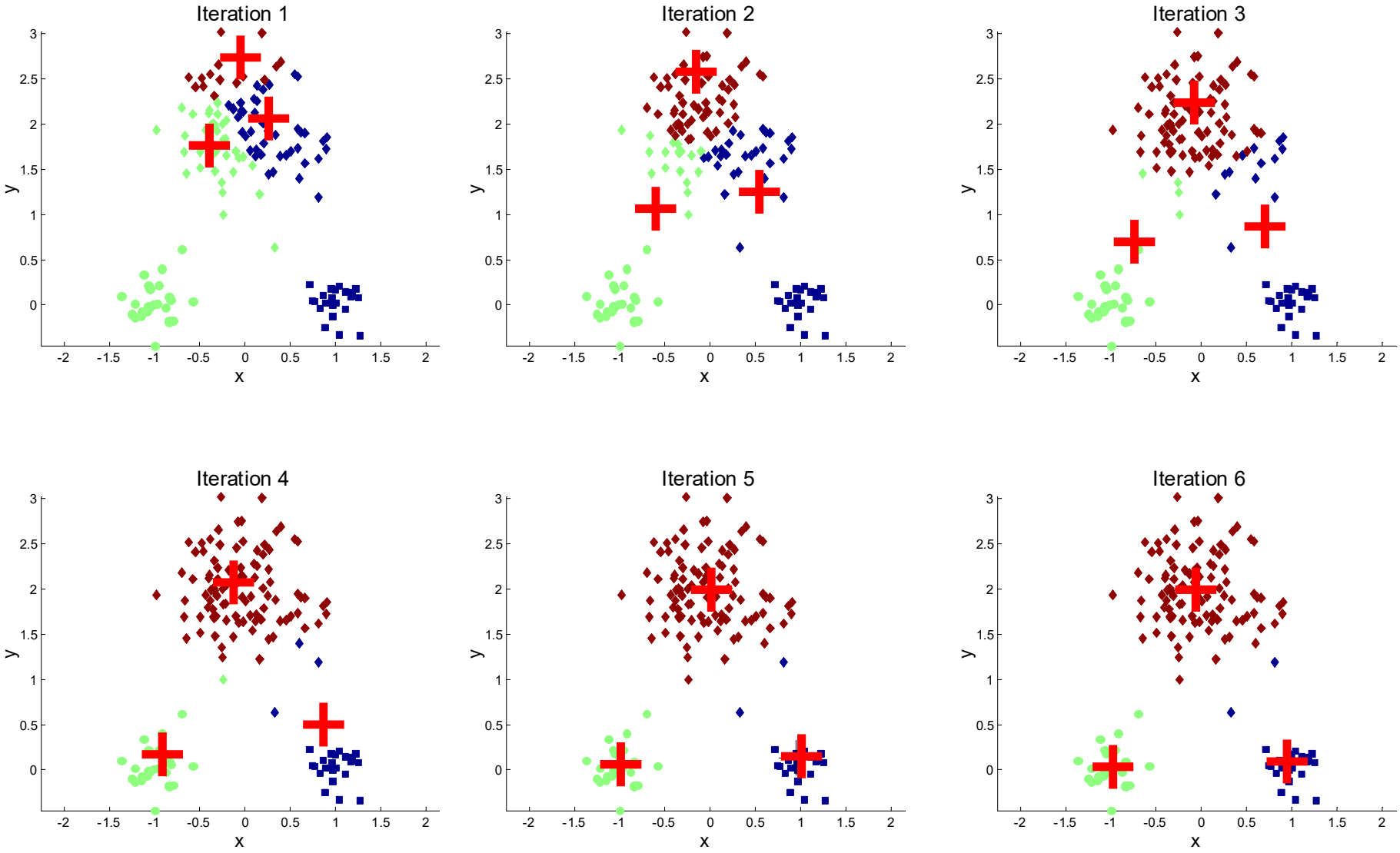
Convergence Criteria

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Default convergence criterion
  - No (or minimum) change of centroids
- Alternative convergence criteria
  - No (or minimum) re-assignments of data points to different clusters
  - Stop after x iterations
  - Minimum decrease in the sum of squared error (SSE)
    - See next slide

K-Means Clustering – Second Example

UNIVERSITY OF MANNHEIM
Data and Web Science Group



Evaluating K-Means Clusterings

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Widely used cohesion measure: **Sum of Squared Error (SSE)**
  - For each point, the error is the distance to the nearest centroid
  - To get SSE, we square these errors and sum them

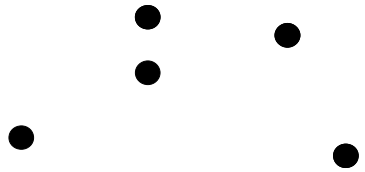
$$SSE = \sum_{j=1}^{k} \sum_{x \in C_j} dist(x, m_j)^2$$

    - C_j is the j-th cluster
    - m_j is the centroid of cluster C_j
      (the **m**ean vector of all the data points in C_j)
    - $dist(x, m_j)$ is the distance between data point x and centroid m_j
- Given several clusterings (= groupings),
  we should prefer the one with the smallest SSE

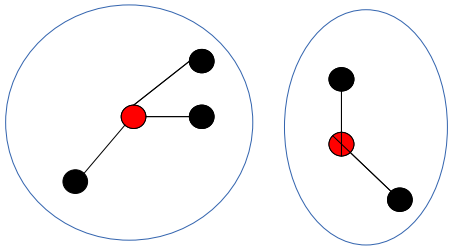
Illustration: Sum of Squared Error

UNIVERSITY OF MANNHEIM
Data and Web Science Group

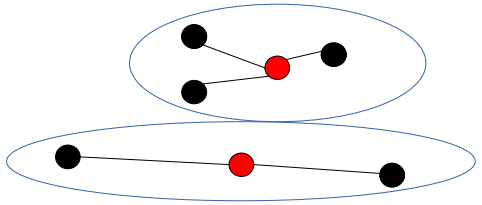
- Clustering problem given:



- Good clustering
  - small distances to centroids



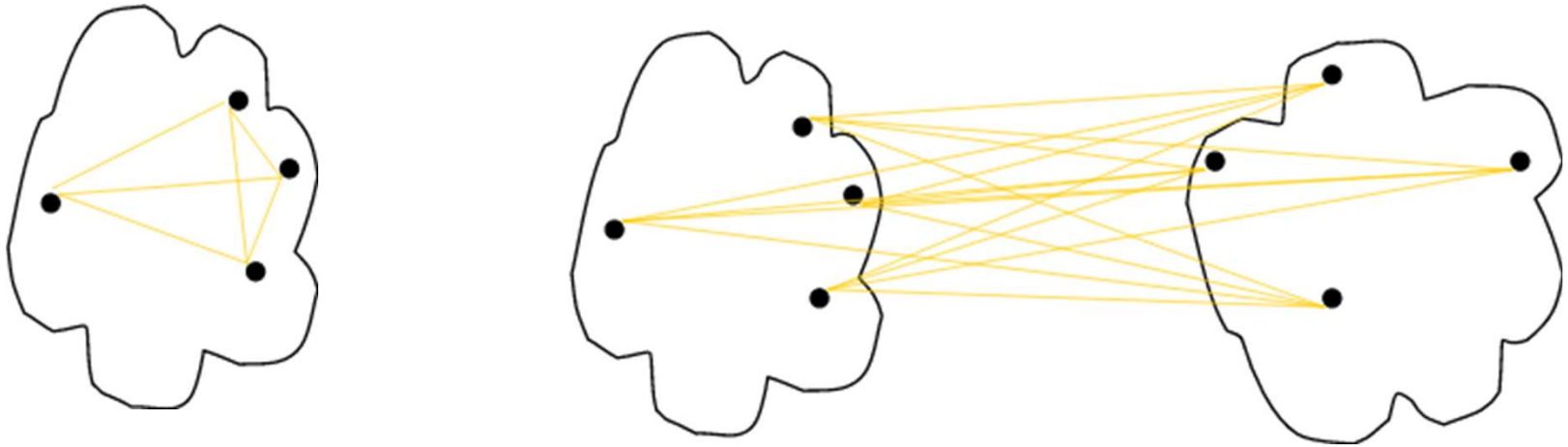
- Not so good clustering
  - larger distances to centroids



Cluster Evaluation using the Silhouette Coefficient

UNIVERSITY OF MANNHEIM
Data and Web Science Group

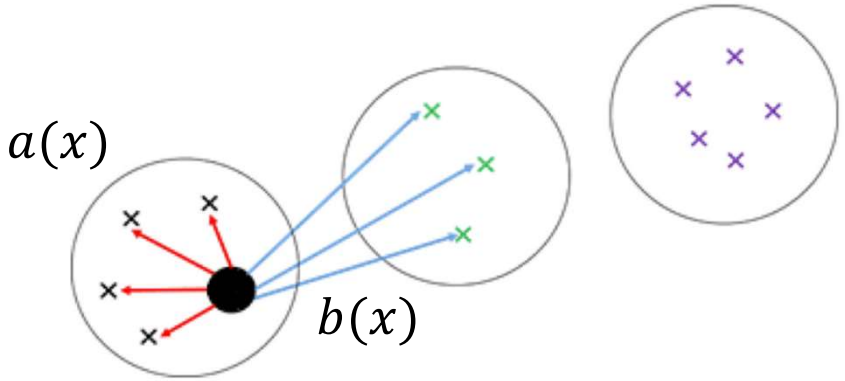
- Recap: we want to maximize
  - **Cohesion:** measures how closely related are objects in a cluster
  - **Separation:** measure how distinct or well-separated a cluster is from other clusters



Cohesion

Separation

Silhouette Coefficient



- Cohesion $a(x)$: Average distance of x to all other examples **in the same cluster**
- Separation $b(x)$: Average distance of x to the examples **in nearest neighboring cluster.**

  Silhouette $s(x)$:

$$s(x) = \frac{b(x) - a(x)}{\max\{a(x), b(x)\}} \qquad s(x) \in [-1,1]$$

-1 = bad
0 = indifferent
1 = good

- Silhouette coefficient (SC):

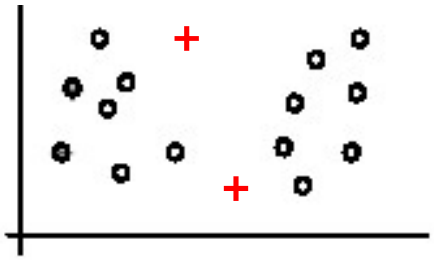
$$SC = \frac{1}{N}\sum_{i=1}^{N} s(x_i)$$

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.silhouette_score.html

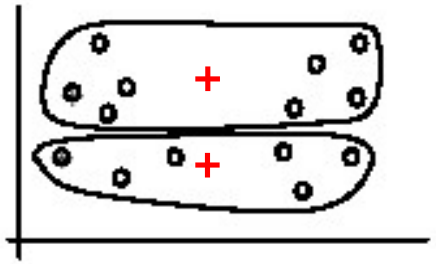
Weaknesses of K-Means: Initial Seeds

UNIVERSITY OF MANNHEIM
Data and Web Science Group

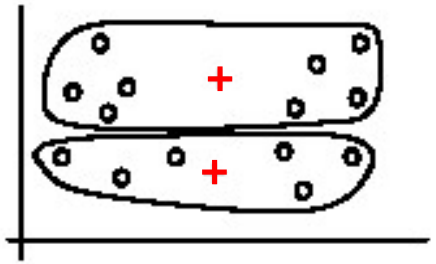
- Clustering results may vary significantly depending on initial choice of seeds (**number** and **position** of seeds)



(A). Random selection of seeds (centroids)



(B). Iteration 1

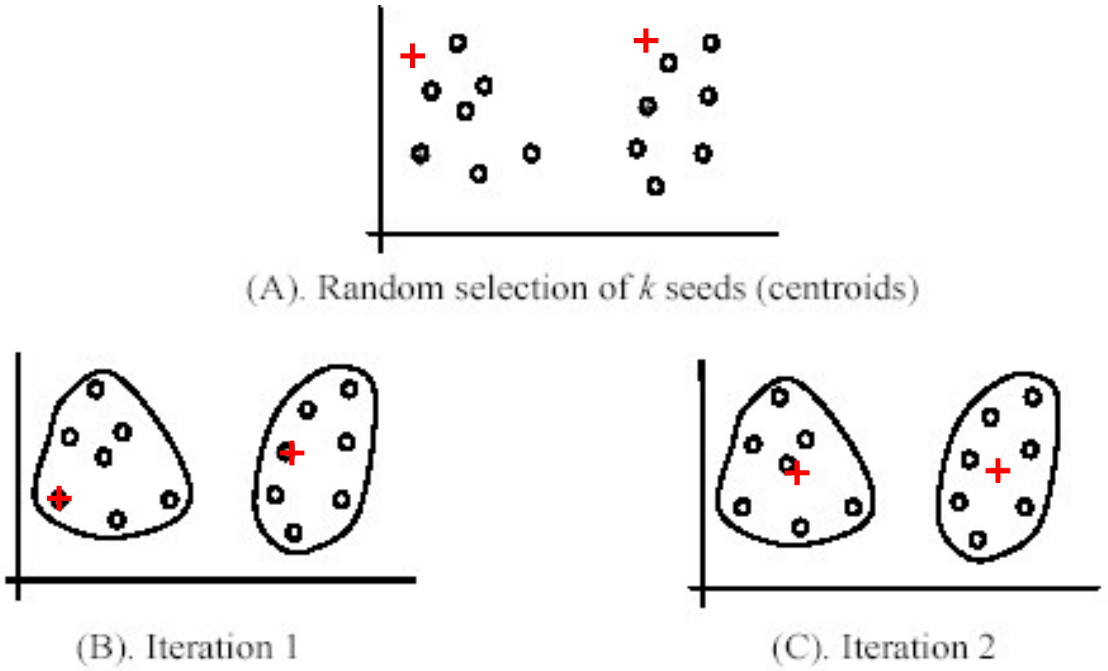


(C). Iteration 2

Weaknesses of K-Means: Initial Seeds

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- If we use **different seeds**, we get good results



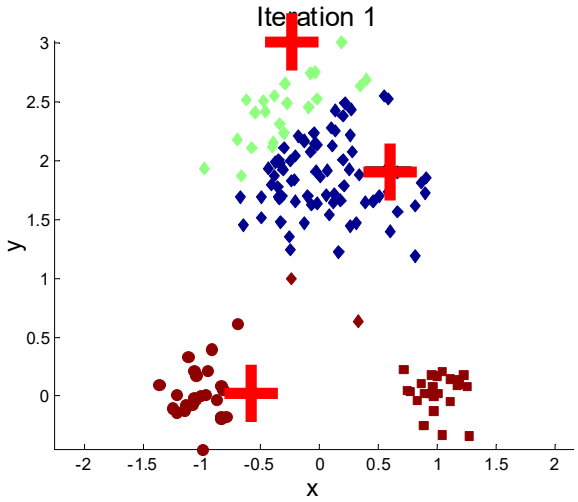
(A). Random selection of k seeds (centroids)

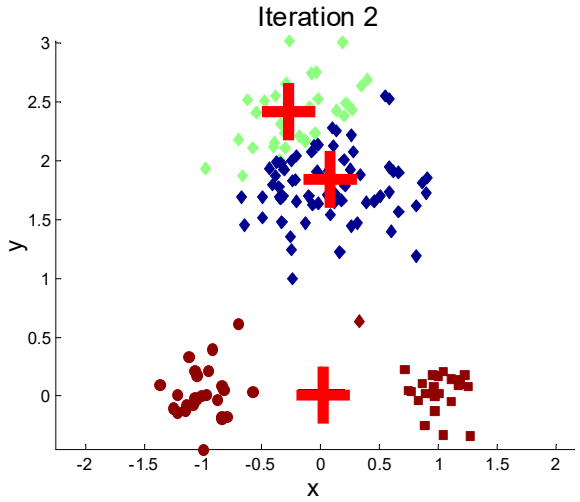
(B). Iteration 1

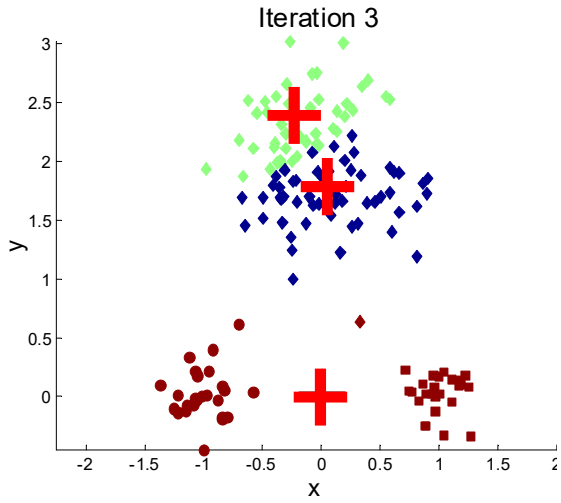
(C). Iteration 2

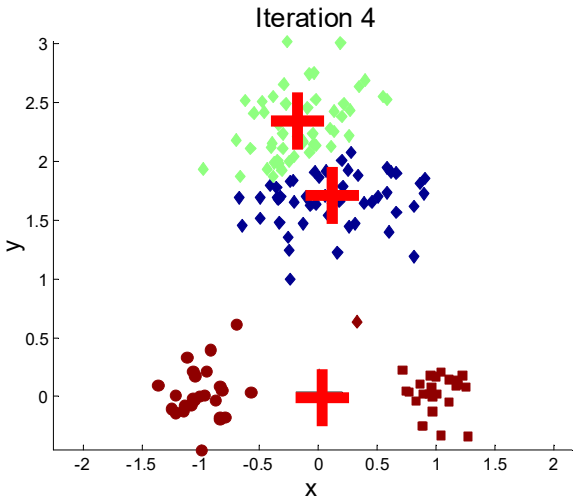
Bad Initial Seeds – Second Example

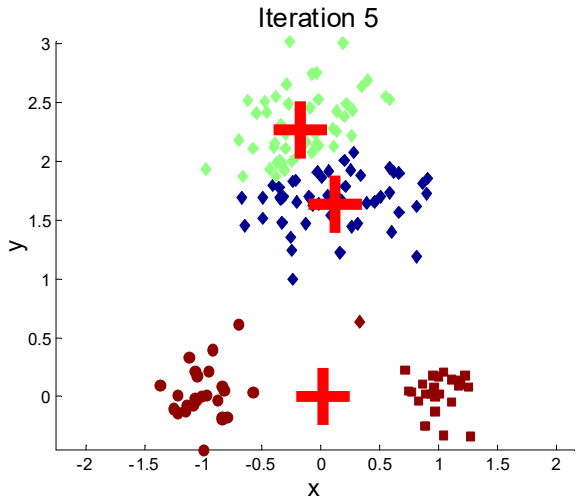
UNIVERSITY OF MANNHEIM
Data and Web Science Group







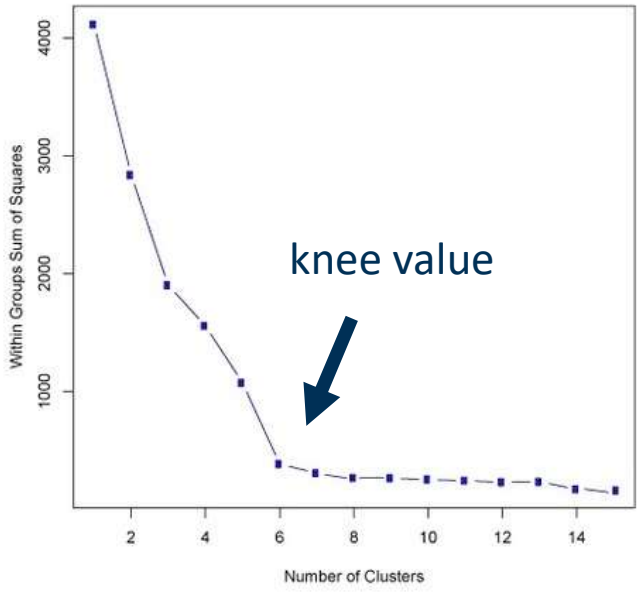




Improving the Clustering Results

UNIVERSITY OF MANNHEIM
Data and Web Science Group

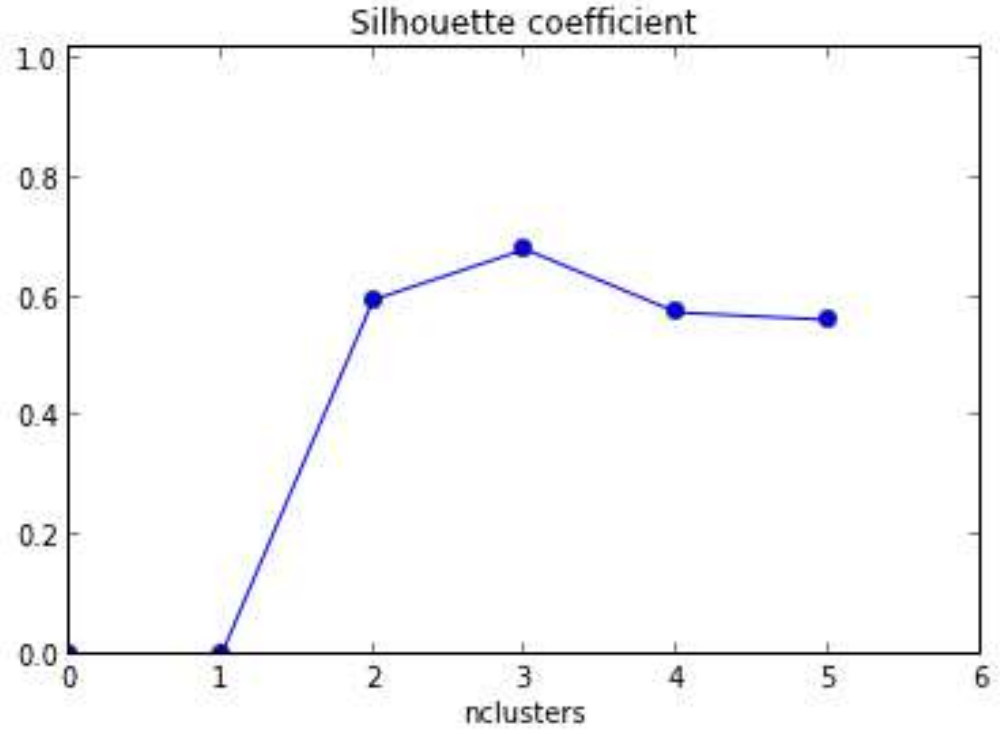
- Restart a number of times with different random seeds
  - chose the resulting clustering with the smallest sum of squared error (SSE)
- Run k-means with different values of k
  - The SSE for different values of k cannot directly be compared
    - Think: what happens for k → number of examples?
  - Workarounds
    - Choose k where SSE improvement decreases (knee value of k)
    - Employ X-Means
      - Variation of K-Means algorithm that automatically determines k
      - Starts with small k, then splits large clusters until improvement decreases



Selecting k using the Silhouette Coefficient

UNIVERSITY OF MANNHEIM
Data and Web Science Group

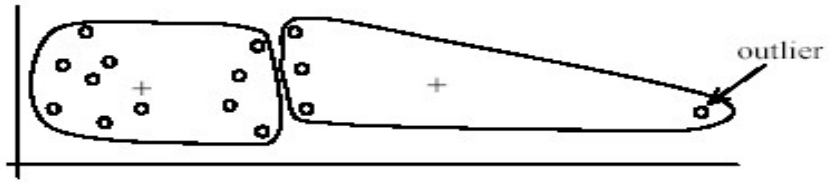
1. Run k-means with different k values
2. Plot the Silhouette Coefficient
3. Pick the best (i.e., highest) silhouette coefficient



Weaknesses of K-Means: Problems with Outliers

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Possible remedy:
  - Remove data points far away from centroids
  - To be safe: monitor these possible outliers over a few iterations and then decide to remove them
- Other remedy: random sampling
  - After determining the centroids based on random samples, assign the rest of the data points (also improves runtime performance)



(A): Undesirable clusters

(B): Ideal clusters

K-Medoids

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- K-Medoids is a K-Means variation that uses the **medians** of each cluster instead of the mean
  - Medoids are the **most central existing data points** in each cluster
- K-Medoids is more robust against outliers as the median is not affected by extreme values:
  - Mean and Median of 1, 3, 5, 7, 9 is **5**
  - Mean of 1, 3, 5, 7, 1009 is **205**
  - Median of 1, 3, 5, 7, 1009 is **5**

K-Means Clustering Summary

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- **Advantages**
  - Simple, understandable
  - Efficient time complexity: O(n * K * I * d) where
    - n = number of points
    - K = number of clusters
    - I = number of iterations
    - d = number of attributes
- **Disadvantages**
  - Need to determine number of clusters
  - All items are forced into a cluster
  - Sensitive to
    - Outliers
    - Initial seeds

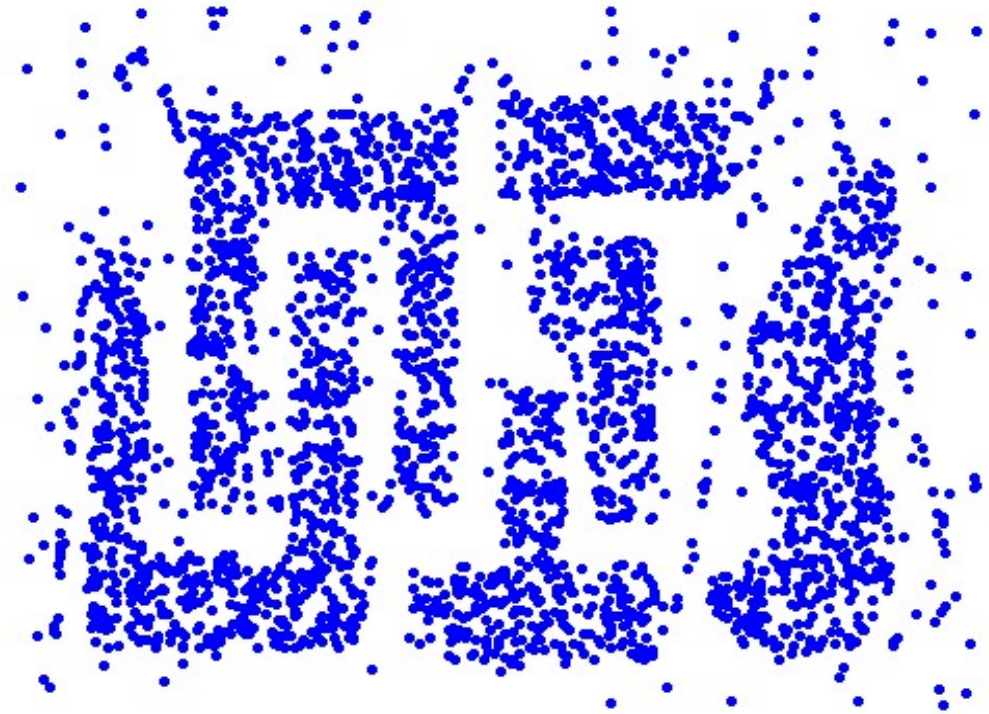
Python

```python
# import KMeans
from sklearn.cluster import KMeans

# create clusterer
estimator = KMeans(n_clusters = 3)

# create clustering
cluster_ids = estimator.fit_predict(dataset[['Att1', 'Att2']])
```

3. Density-based Clustering

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Challenging use case for K-Means because
  - Problem 1: Non-globular shapes
  - Problem 2: Outliers / noise points

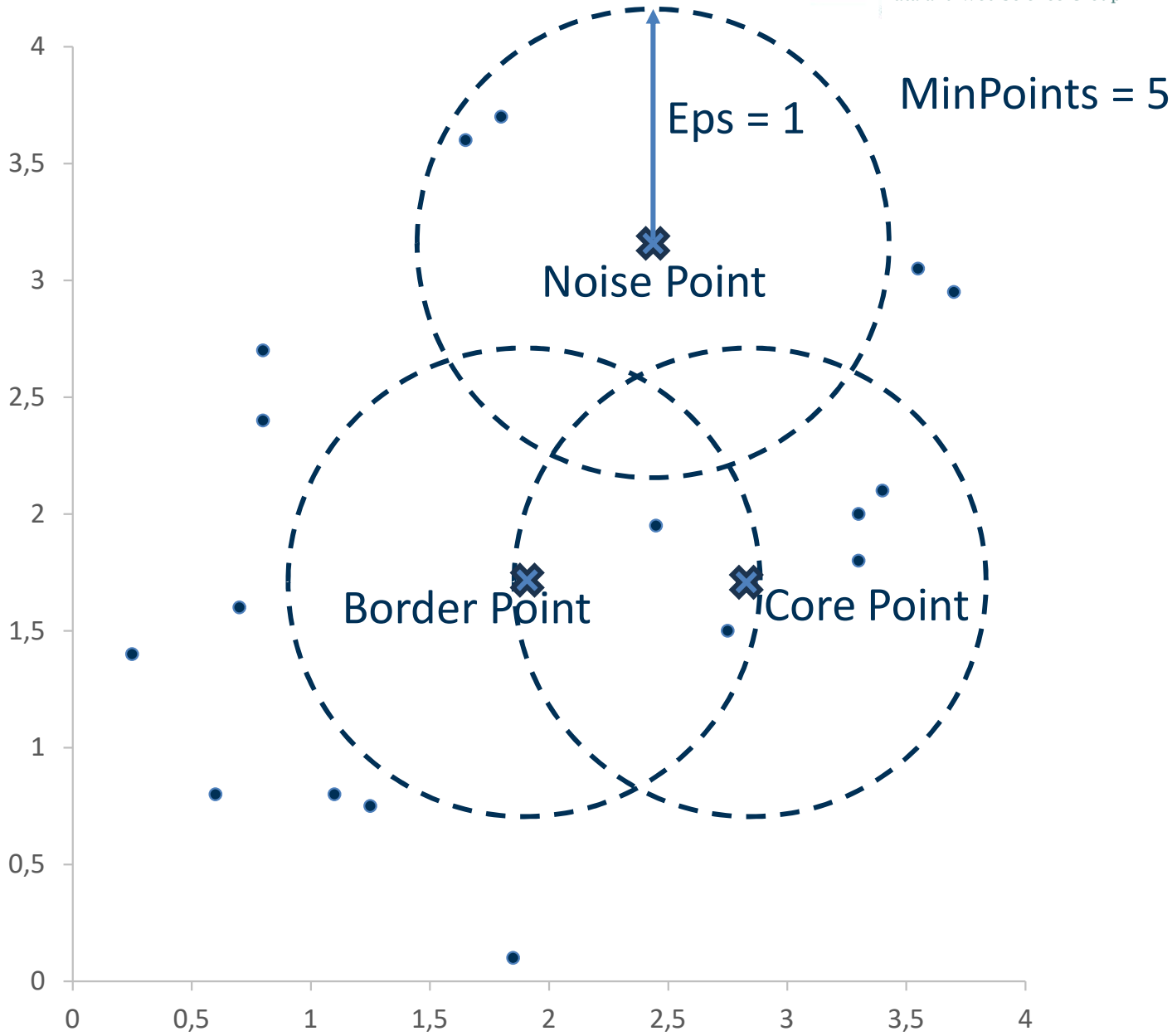


DBSCAN

UNIVERSITY OF MANNHEIM
Data and Web Science Group

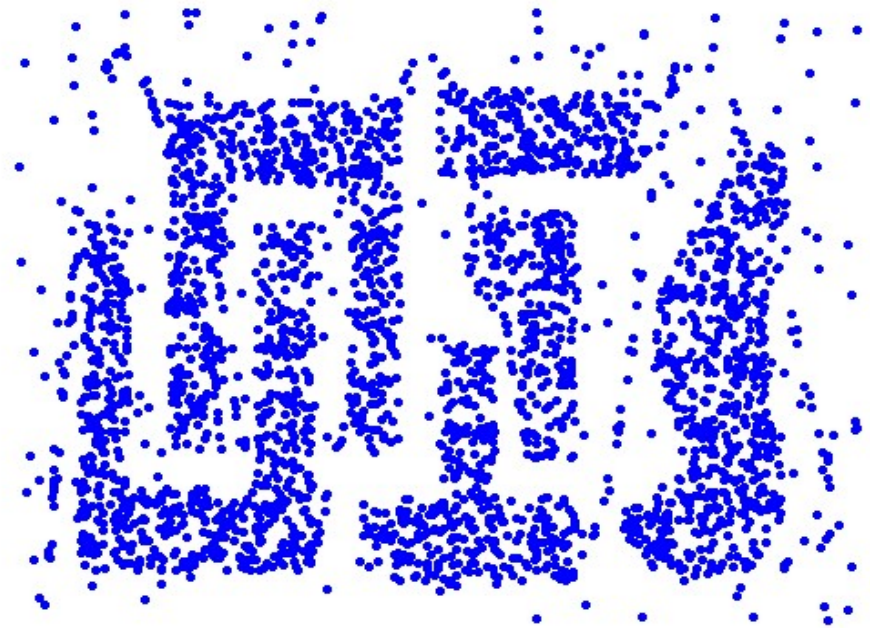
- DBSCAN is a density-based algorithm
  - **Density** = number of points within a specified radius Epsilon (**Eps**)
- Divides data points into three classes:
  - A point is a **core point** if it has at least a specified number of neighboring points (***MinPts***) within the specified radius *Eps*
    - the point itself is counted as well
    - these points form the interior of a dense region (cluster)
  - A **border point** has fewer points 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

Examples of Core, Border, and Noise Points

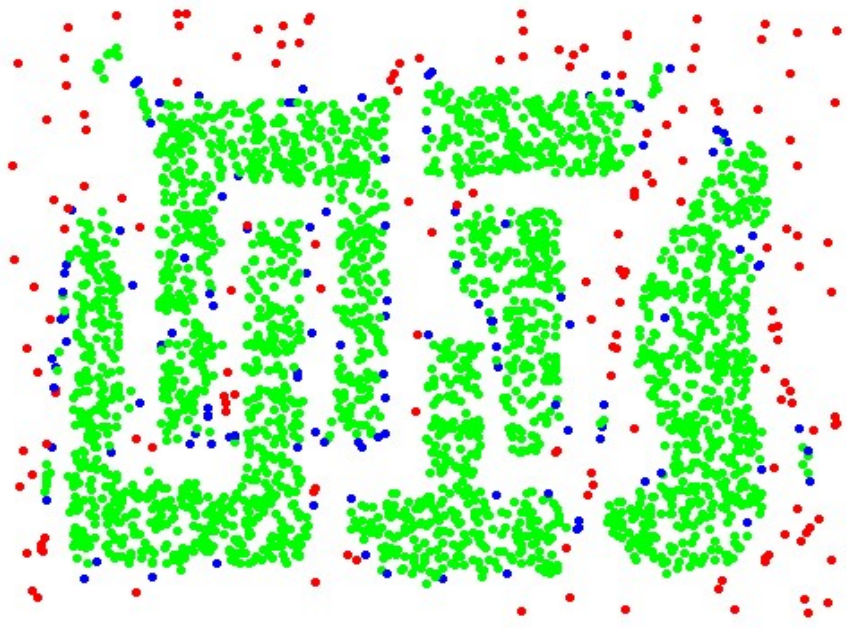


Examples of Core, Border, and Noise Points

UNIVERSITY OF MANNHEIM
Data and Web Science Group



1) Original Points



2) Point types:
core, border and noise

DBSCAN Algorithm

UNIVERSITY OF MANNHEIM
Data and Web Science Group

Eliminates noise points and returns clustering of the remaining points:

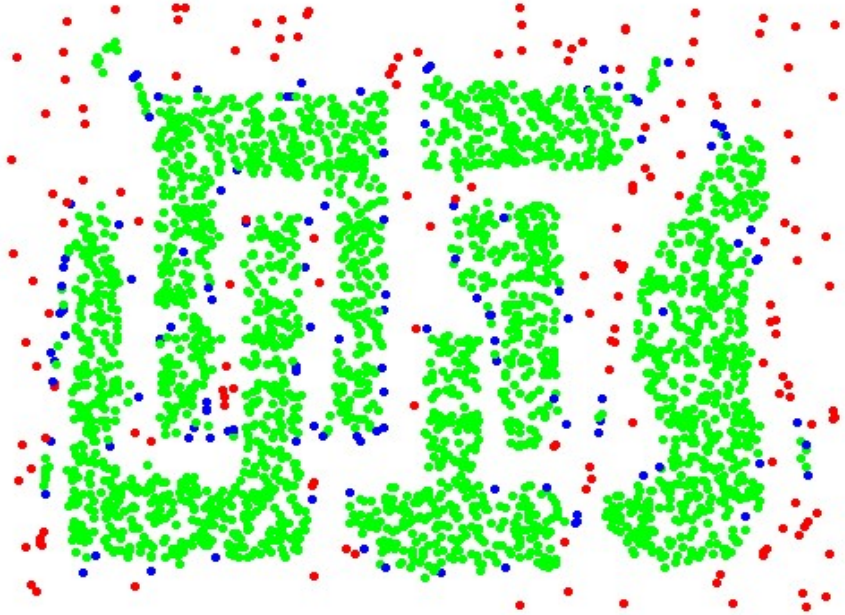
1. Label all points as core, border, or noise points
2. Eliminate all noise points
3. Put an edge between all core points that are within Eps of each other
4. Make each group of connected core points into a separate cluster
5. Assign each border point to one of the clusters of its associated core points
   - as a border point can be at the border of multiple clusters
   - use voting if core points belong to different clusters
   - if equal vote, than assign border point randomly

Time complexity: O(n log n)

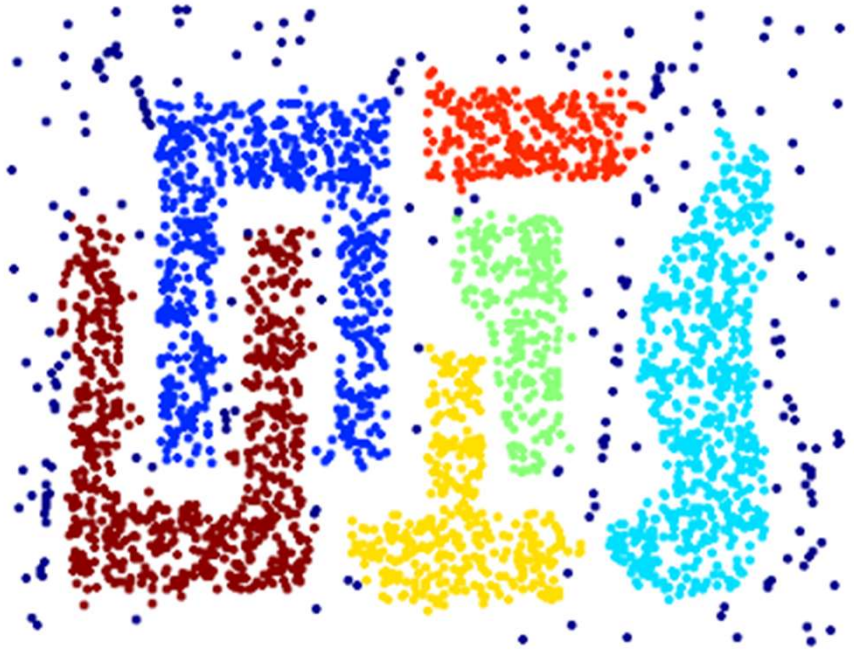
- dominated by neighborhood search for each point using an index

Examples Clusters

1) Original Points



2) Point types:
core, border and noise

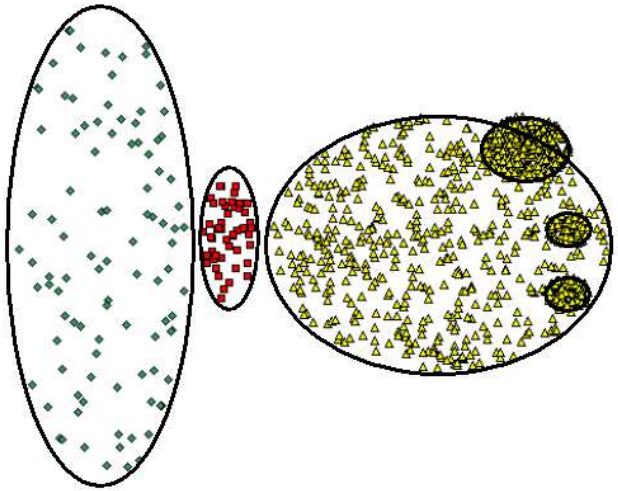


3) Clusters

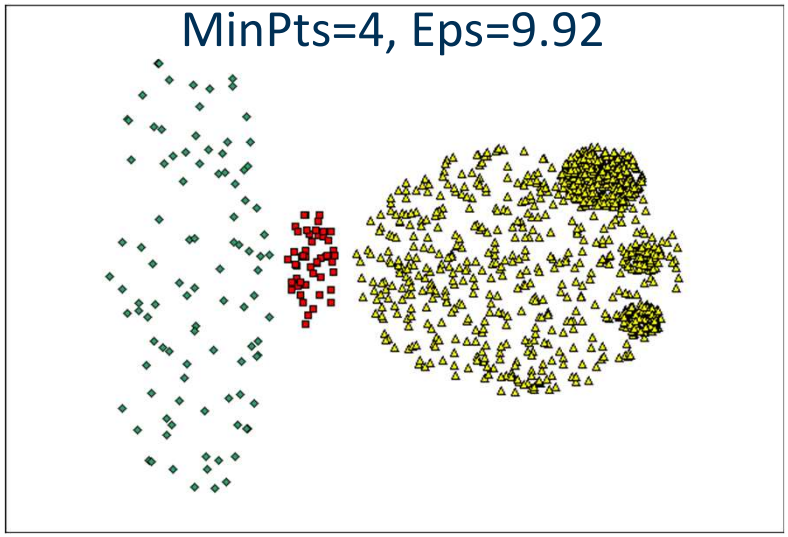
When DBSCAN Does NOT Work Well

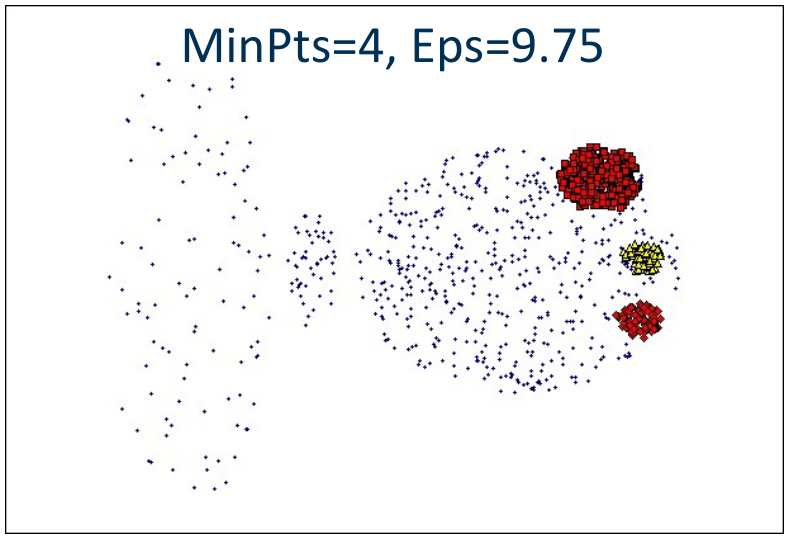
UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Varing densities!



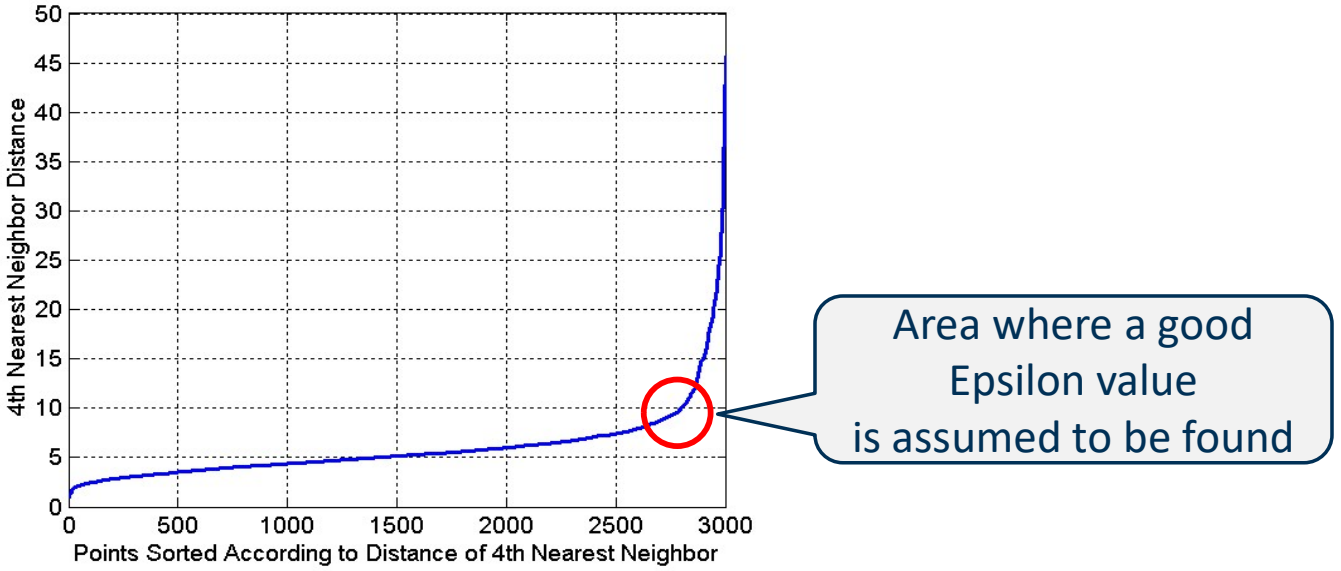
Original Points





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
- Plot sorted distance of every point to its k^{th} nearest neighbor



DBSCAN in Python

UNIVERSITY OF MANNHEIM
Data and Web Science Group

Python

```python
# import DBSCAN
from sklearn.cluster import DBSCAN

# create the clusterer
clusterer = DBSCAN(min_samples=3, eps=1.5, metric='euclidean')

# create the clusters
clusters = clusterer.fit_predict(dataset[['Att1', 'Att2']])
```

4. Proximity Measures

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- So far, we have seen different clustering algorithms
  - All of which rely on proximity (distance, similarity, ...) measures
- Now, we discuss proximity measures in more detail
- A wide range of different measures is used depending on the requirements of the application
- **Similarity**
  - Numerical measure of how alike two data objects are
  - Range [0,1]
- **Dissimilarity / Distance**
  - Numerical measure of how different are two data objects
  - Minimum dissimilarity is often 0, upper limit varies
- We distinguish proximity measures for single attributes and measures for multidimensional data points (records)

Proximity of Single Attributes

UNIVERSITY OF MANNHEIM
Data and Web Science Group

Attribute Type	Dissimilarity	Similarity
Nominal	$d = \begin{cases} 0 & \text{if } p = q \\ 1 & \text{if } p \neq q \end{cases}$	$s = \begin{cases} 1 & \text{if } p = q \\ 0 & \text{if } p \neq q \end{cases}$
Ordinal	$d = \frac{\lvert p-q \rvert}{n-1}$ (values mapped to integers 0 to $n-1$, where n is the number of values)	$s = 1 - \frac{\lvert p-q \rvert}{n-1}$
Interval or Ratio	$d = \lvert p - q \rvert$	$s = -d$, $s = \frac{1}{1+d}$ or $s = 1 - \frac{d - min\_d}{max\_d - min\_d}$

Similarity and dissimilarity for simple attributes

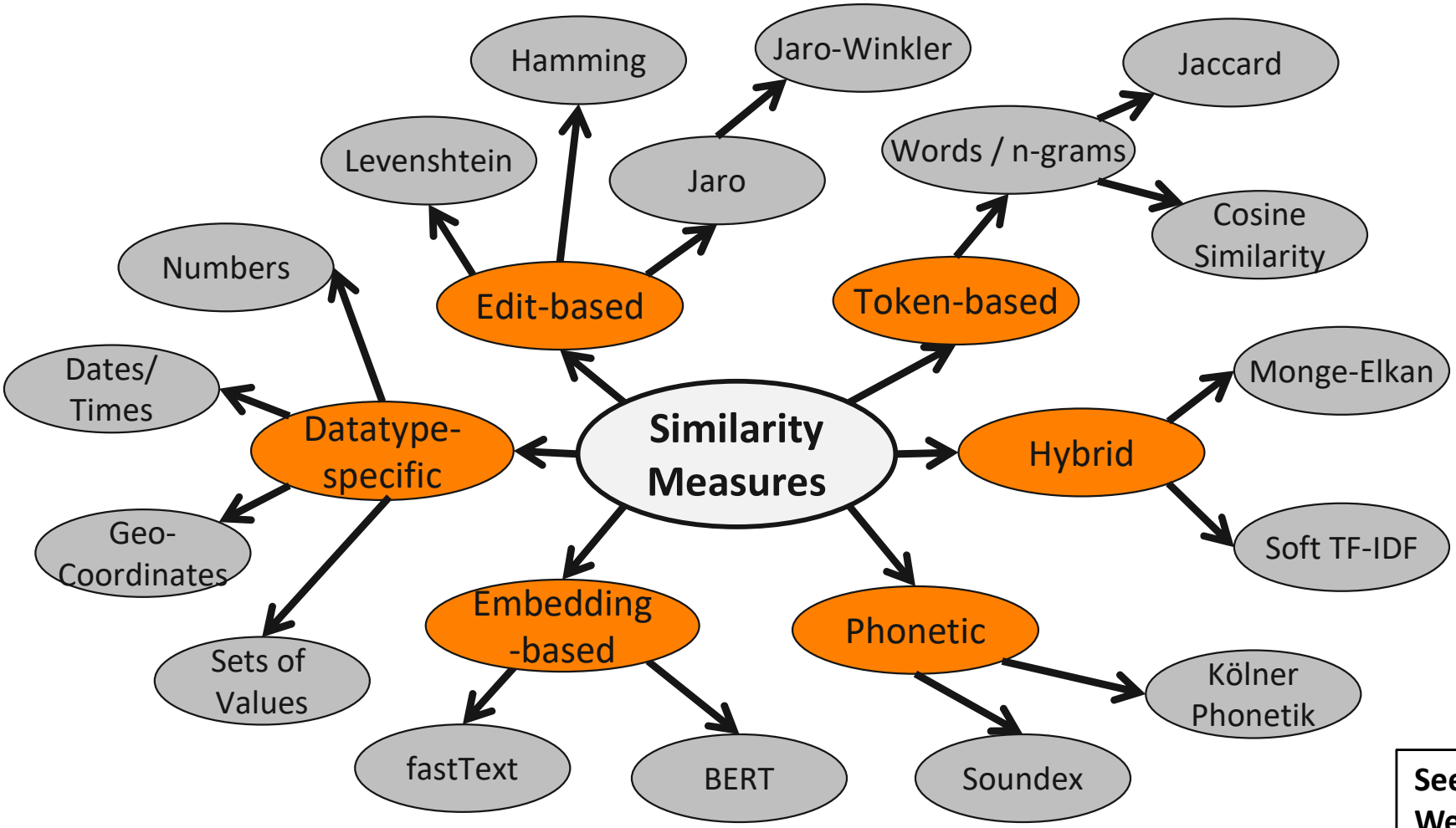
p and q are attribute values for two data objects

Levenshtein Distance

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Measures the dissimilarity of **two strings**
- Measures the **minimum number of edits** needed to transform one string into the other
- Allowed edit operations:
  - **Insert** a character into the string
  - **Delete** a character from the string
  - **Replace** one character with a different character
- Examples:
  - levensthein('table', 'cable') = 1 (1 substitution)
  - levensthein('Doe, Jane', 'Jane Doe') = 8 (7 substitution, 1 deletion)

Further String Similarity Metrics



See: IE 670 Web Data Integration

Proximity of Multidimensional Data Points

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- All measures discussed so far cover the proximity of single attribute values
- But we usually have data points with many attributes
  - e.g., age, height, weight, sex...
- Thus, we need proximity measures for data points taking multiple attributes/dimensions into account
- The suitability of a measure depends on
  1. the scale of the attributes
  2. the specific use case

Multiple Continuous Attributes

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Euclidean Distance (L_2 - norm)
  - $dist = \sqrt{\sum_{k=1}^{n}(p_k - q_k)^2}$
  - n is the number of dimensions (attributes) and p_k and q_k are the k-th attributes of data points p and q
- Manhattan distance (L_1 - norm)
  - $dist = \sum_{k=1}^{n} |p_k - q_k|$
  - Minimum distance to go from one crossing to another
    - in a squared city (like Manhattan)
- More general (L_p - norm)
  - $dist = \sqrt[p]{\sum_{k=1}^{n} |p_k - q_k|^p}$

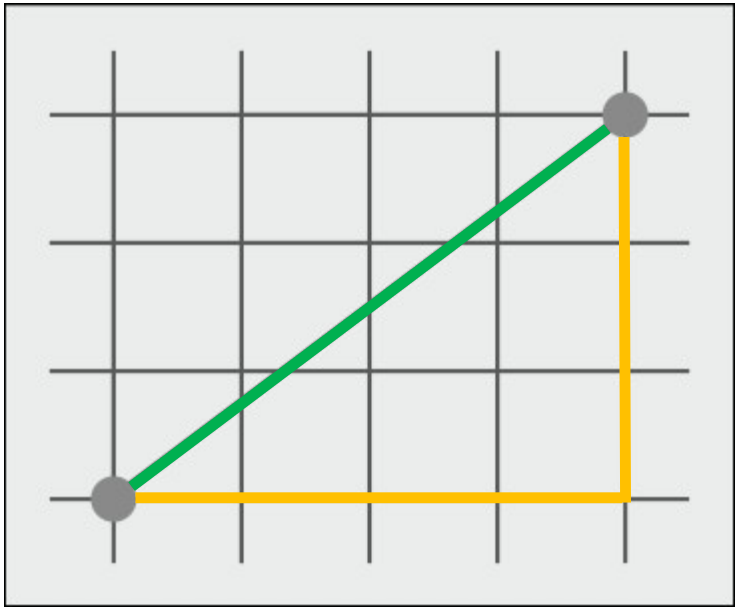
$$= \left(\sum_{k=1}^{n} |p_k - q_k|^p\right)^{\frac{1}{p}}$$

Squaring increases the impact of large deviations

Example:

$L_2 = \sqrt{4^2 + 3^2} = 5$

$L_1 = 4+3=7$



Normalization

- Attributes should be normalized so that all attributes can have **equal (or application-specific) impact** on the distance
- Consider the following pair of data points
  - x_i: (0.1, 20) and x_j: (0.9, 720).

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{(0.9 - 0.1)^2 + (720 - 20)^2} = 700.000457$$

- The distance is almost completely dominated by (720-20) = 700
- Solution: Scale attributes to have a **common value range**, for instance [0,1]

Scaling in Python

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- StandardScaler $z = \frac{x-\mu}{\sigma}$
  - Standardize features by removing the mean and scaling to unit variance
- MinMaxScaler $X_{norm} = \frac{X-X_{min}}{X_{max}-X_{min}}$
  - Transform features by scaling each feature to a given range e.g. [0,1]

Python

```python
# import min-max scaler
from sklearn import preprocessing.MinMaxScaler()

# create scaler
scaler = MinMaxScaler()

# normalize the relevant attributes
dataset[['Att1', 'Att2']] = scaler.fit_transform(dataset[['Att1', 'Att2']])
```

Similarity of Binary Attributes

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Common situation is that objects, *p* and *q*, have only binary attributes
  - products in shopping basket
  - courses attended by students

- We compute similarities using the following quantities:

  M_{11} = the number of attributes where p was 1 and q was 1
  M_{00} = the number of attributes where p was 0 and q was 0
  M_{01} = the number of attributes where p was 0 and q was 1
  M_{10} = the number of attributes where p was 1 and q was 0

Symmetric Binary Attributes

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- A binary attribute is **symmetric** if both of its states (0 and 1) have equal importance, and carry the same weights, e.g., male and female
- Similarity measure: **Simple Matching Coefficient**

$$SMC(\mathbf{x}_i, \mathbf{x}_j) = \frac{M_{11} + M_{00}}{M_{01} + M_{10} + M_{11} + M_{00}}$$

Number of matches / number of all attributes values

Asymmetric Binary Attributes

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Asymmetric: If one of the states is more important than the other
  - by convention, state 1 represents the more important state
  - 1 is typically the rare or infrequent state
  - examples: shopping baskets, word vectors
- Similarity measure: **Jaccard Coefficient**

$$J(\mathbf{x}_i, \mathbf{x}_j) = \frac{M_{11}}{M_{01} + M_{10} + M_{11}}$$

Number of 11 matches / number of **not-both-zero attributes** values

SMC versus Jaccard: Example

UNIVERSITY OF MANNHEIM
Data and Web Science Group

p = 1 0 0 0 0 0 0 0 0 0
q = 0 0 0 0 0 0 1 0 0 1

Interpretation of the example:
Customer p bought item 1
Customer q bought item 7 and 10

M_{11} = 0 (the number of attributes where p was 1 and q was 1)
M_{00} = 7 (the number of attributes where p was 0 and q was 0)
M_{01} = 2 (the number of attributes where p was 0 and q was 1)
M_{10} = 1 (the number of attributes where p was 1 and q was 0)

$$SMC = (M_{11} + M_{00})/(M_{01} + M_{10} + M_{11} + M_{00}) = (0+7) / (2+1+0+7) = 0.7$$

$$J = (M_{11}) / (M_{01} + M_{10} + M_{11}) = 0 / (2 + 1 + 0) = 0$$

SMC versus Jaccard: Question

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Which of the two measures would you use …
- …for a dating agency?
  - hobbies
  - favorite bands
  - favorite movies
  - …
- …for the Wahl-O-Mat?
  - (dis-)agreement with political statements
  - recommendation for voting

Using Weights to Combine Similarities

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- You may not want to treat all attributes the same
  - use weights w_k which are between 0 and 1 and sum up to 1
  - weights are set according to the importance of the attributes

- Example: **Weighted Euclidean Distance**

$$dist(\mathbf{x}_i, \mathbf{x}_j) = \sqrt{w_1(x_{i1} - x_{j1})^2 + w_2(x_{i2} - x_{j2})^2 + \ldots + w_r(x_{ir} - x_{jr})^2}$$

How to Choose a good Clustering Algorithm?

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- "Best" algorithm depends on
  1. the analytical goals of the specific use case
  2. the distribution of the data
- Normalization, feature selection, distance measure, and parameter settings have equally high influence on results
- Due to these complexities, the common practice is to
  1. run several algorithms using different distance measures, feature subsets and parameter settings, and
  2. then visualize and interpret the results based on knowledge about the application domain as well as the goals of the analysis

5. Anomaly Detection

UNIVERSITY OF MANNHEIM
Data and Web Science Group

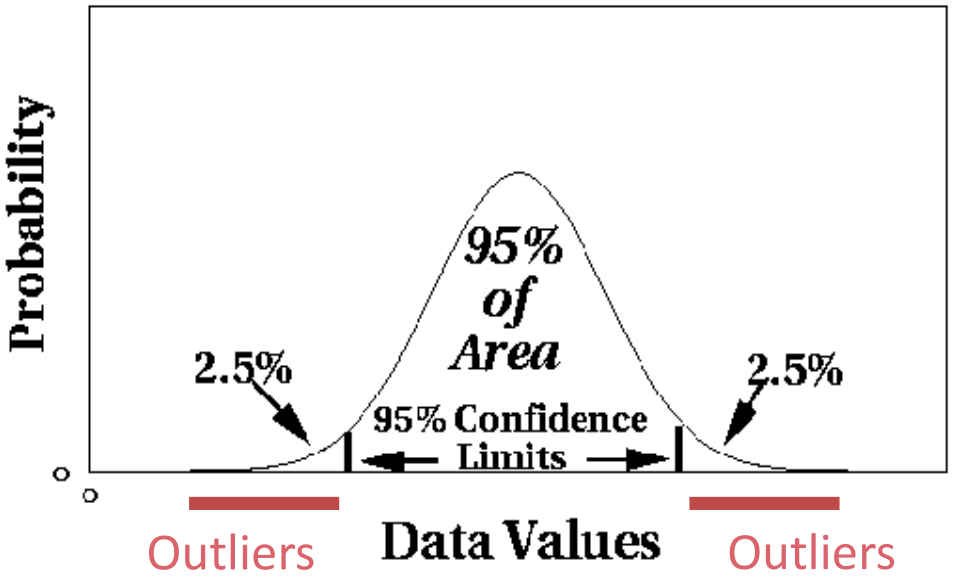
- Also known as “Outlier Detection”
- Automatically identify data points
  that are somehow different from the rest
- Working assumption:
  - There are considerably more “normal” observations than “abnormal” observations (outliers/anomalies) in the data
- Methods:
  1. Statistical Approaches (IQR, MAD)
  2. Distance-based Approaches
  3. Density-based Approaches
  4. Clustering-based Approaches

5.1 Statistical Approaches

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Assume a parametric model describing the distribution of the data (e.g., normal distribution)
- Determine outliers using Z-score
  - e.g. mark a point as an outlier if $| z | > 3$

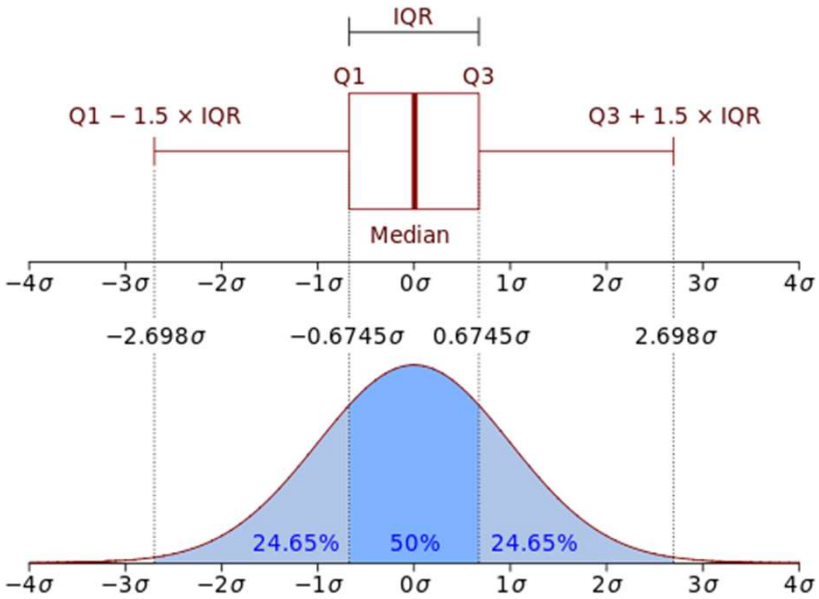
$$z = \frac{x - \mu}{\sigma}$$



Interquartile Range (IQR)

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Divides data in quartiles
  - relies on data sample
  - does not require distribution
- Definitions:
  - Q1: x ≥ Q1 holds for 75% of all x
  - Q3: x ≥ Q3 holds for 25% of all x
  - IQR = Q3-Q1
- Outlier detection:
  - All values outside
    [Q1-1.5*IQR ; Q3+1.5*IQR]
- Example:
  - 0,1,1,3,3,5,7,42 → median=3, Q1=1, Q3=7 → IQR = 6
  - Allowed interval: [1-1.5*6 ; 7+1.5*6] = [-8 ; 16]



Thus, 42 is an outlier

Median Absolute Deviation (MAD)

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- MAD is the median deviation from the median of a sample, i.e.

$$\tilde{X} = median(X)$$
$$MAD = median(|X_i - \tilde{X}|)$$

- MAD can be used for outlier detection
  - All values that are k*MAD away from the median are considered to be outliers e.g., k=3
- Example:
  - X = 5, 1, 42, 0, 1, 3, 7
  - $median(X)$: 0,1,1,**3**,5,7,42 → $\tilde{X} = 3$
  - $|X_i - \tilde{X}|$ (Deviations): 3,2,2,0,2,4,39
  - MAD 0,2,2,**2**,3,4,39→ MAD = 2
  - Allowed interval: [3-3*2 ; 3+3*2] = [-3;9]

Thus, 42 is an outlier

Outliers vs. Extreme Values

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- So far, we have looked at extreme values only
  - but outliers can occur as non-extremes
  - in that case, methods like IQR and MAD fail
- IQR on the example below:
  - Q2 (Median) is 0
  - Q1 is -1, Q3 is 1

    → IQR = 2

    → everything outside [-4,+4] is an outlier

    → there are no outliers in this example

5.2 Distance-based Approaches

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Nearest-neighbor based
  - Compute the distance between every pair of data points
  - There are various ways to define outliers:
    1. Data points for which there are fewer than p neighboring points within a distance D
    2. The top n data points whose distance to the k^{th} nearest neighbor is largest
    3. The top n data points whose average distance to the k nearest neighbors is largest
  - Implementation: PyOD
    - https://pyod.readthedocs.io/en/latest/

5.3 Density-based Approachs

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- For each point, compute the density of its local neighborhood
- **Local outlier factor (LOF)** of a point A is the ratio of average density of A's neighbors to density of A
  - Core cluster points: LOF ≈ 1
  - Border point: LOF slightly > 1
  - **Outlier: LOF ≫ 1**
- Advantage: Works if regions have different densities as calculation is local

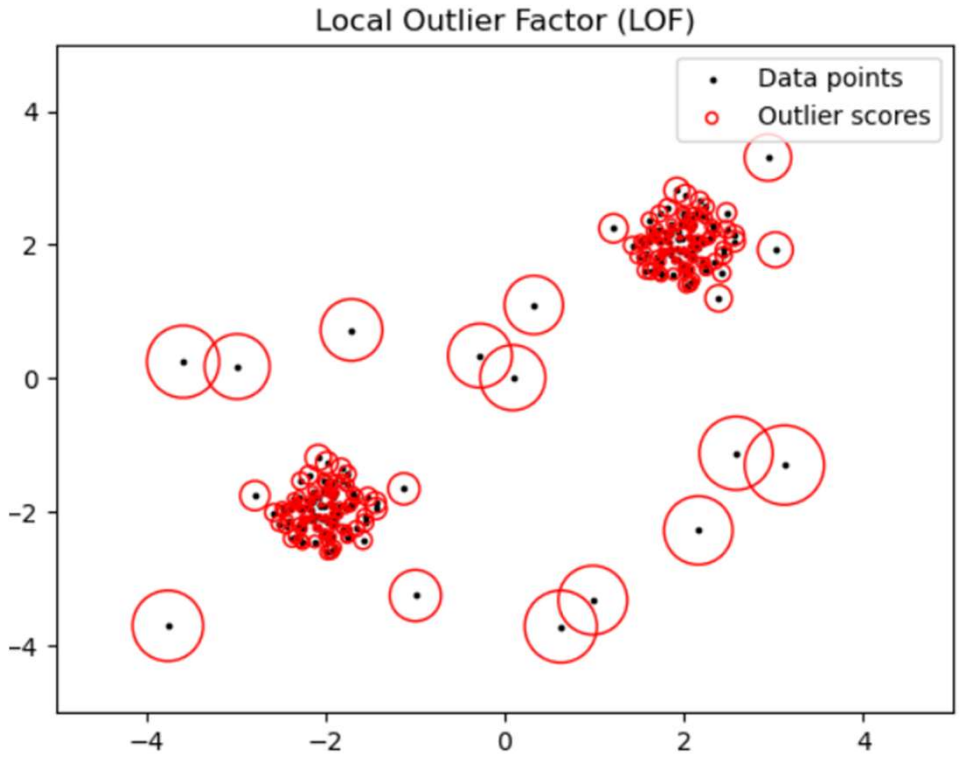
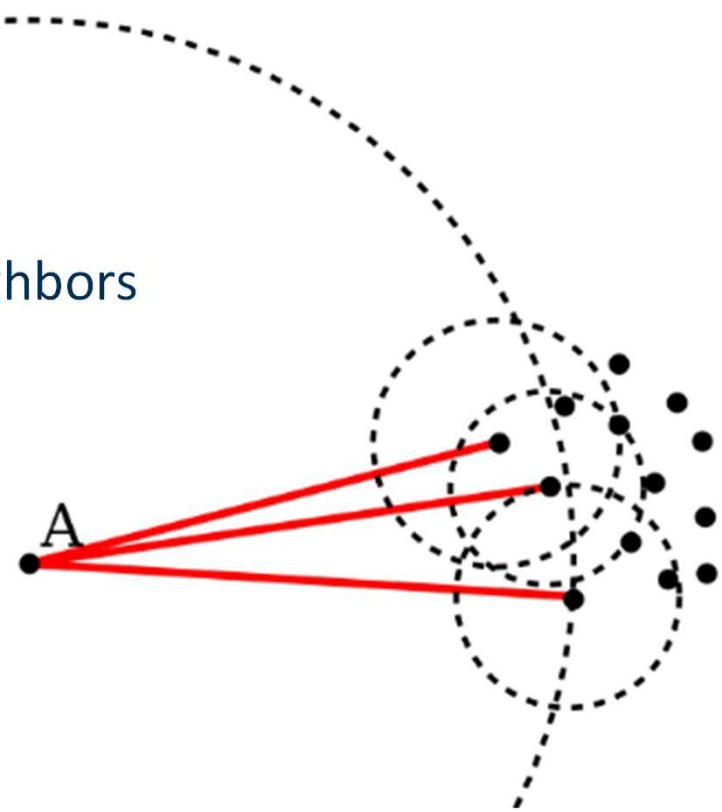


Illustration: Local Outlier Factor

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Using 3 nearest neighbors
- We compute
  - the density of A's neighborhood as the inverse of the average distance to the neighbors
  - the average density of A's neighbor's neighborhoods
- If the density of A is lower than the neighbors' density
  - A might be an outlier

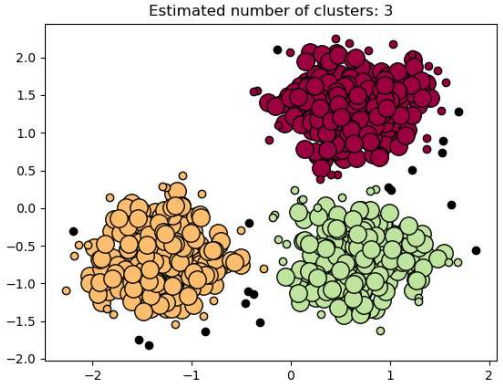


http://commons.wikimedia.org/wiki/File:LOF-idea.svg

DBSCAN for Outlier Detection

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- DBSCAN directly identifies noise points
  - These are outliers not belonging to any cluster
    - In scikit-learn: label -1
  - Allows for performing outlier detection directly

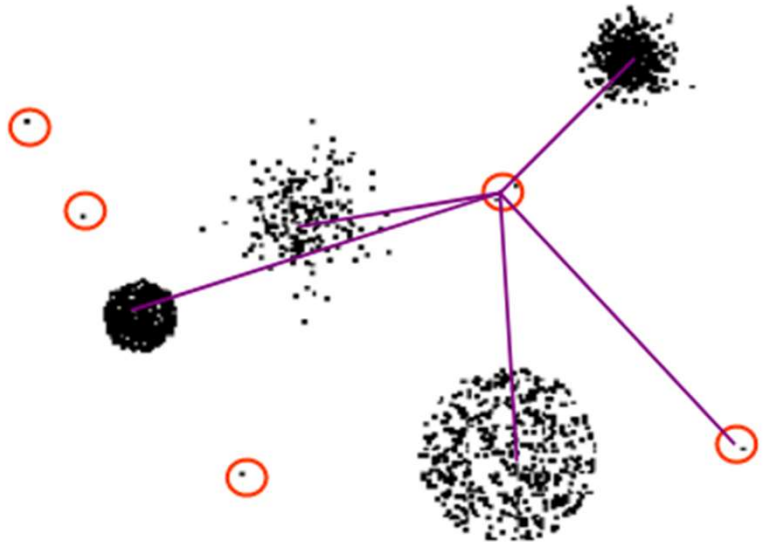



```
# Apply DBSCAN with eps=0.5 and min_samples=5
dbscan = DBSCAN(eps=0.1, min_samples=5)
dbscan.fit(X)

# Identify the noise points
noise_mask = dbscan.labels_ == -1
print(noise_mask)

# Remove the noise points from the dataframe
X = X[~noise_mask]
```

5.4 Clustering-based Outlier Detection

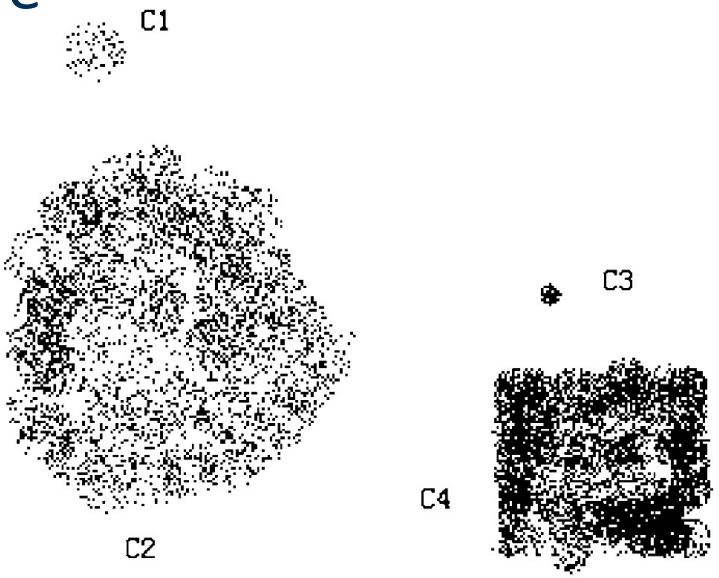
UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Basic idea:
  - Cluster the data into groups of different density
  - Choose points in small cluster as candidate outliers
  - Compute the distance between candidate points and non-candidate clusters
  - If candidate points are far from all other non-candidate points, they are outliers



Clustering-based Local Outlier Factor

- Idea: anomalies are data points that are
  - In a very small cluster or
  - Far away from other clusters



- CBLOF is run on clustered data
- Assigns a score based on
  - The size of the cluster a data point is in
  - The distance of the data point to the next large cluster

Clustering-based Local Outlier Factor

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- General process:
  - First, run a clustering algorithm (of your choice)
  - Then, apply CBLOF

Package PyOD

- Result: data points with outlier score

```
from sklearn.cluster import KMeans
from pyod.models.cblof import CBLOF
# clustering
clust = KMeans()
# outlier detection
detector = CBLOF(n_clusters=8,clustering_estimator=clust)
detector.fit(X)
# removal
noise_mask =detector.predict(X) == 1
X = X[~noise_mask]
```

Online Lecture

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- This week‘s additional material is about **Hierarchical Clustering**

- A part of the **exercise** is about Hierarchical Clustering

Week	Wednesday (Offline Lecture, Room A5, B243)	Online Lecture (see Ilias Course)	Thursday (Exercise, Room B6, A104)
11.02.2026	Introduction to Data Mining (PDF, 4 MB)	Nearest Centroids	Introduction to Python (Solution)
18.02.2026	Classification 1 (PDF, 3 MB)	Ensembles	Classification 1 (Solution)
25.02.2026	Classification 2 (PDF, 3 MB)	Comparing Classifiers	Classification 2 (Solution)
04.03.2026	Regression (PDF, 3 MB)	Time Series	Regression (Solution)
11.03.2026	Cluster Analysis and Anomalies	Hierarchical Clustering	Cluster Analysis
18.03.2026	Association Analysis and Subgroup Discovery	Multi Modal Data	Association Analysis
25.03.2026	Introduction to Student Projects		Time Series
	- Easter Break -		

Literature for this Slideset

UNIVERSITY OF MANNHEIM
Data and Web Science Group

- Pang-Ning Tan, Michael Steinbach, Karpatne,Vipin Kumar: Introduction to Data Mining. 2nd Edition. Pearson.
- Chapter 5: Cluster Analysis
  - Chapter 5.2: K-Means
  - Chapter 5.3: Agglomerative Hierarchical Clustering
  - Chapter 5.4: DBSCAN
- Chapter 9: Anomaly Detection
- Chapter 2.4: Measures of Similarity and Dissimilarity