Clustering

Markus Götz, KIT

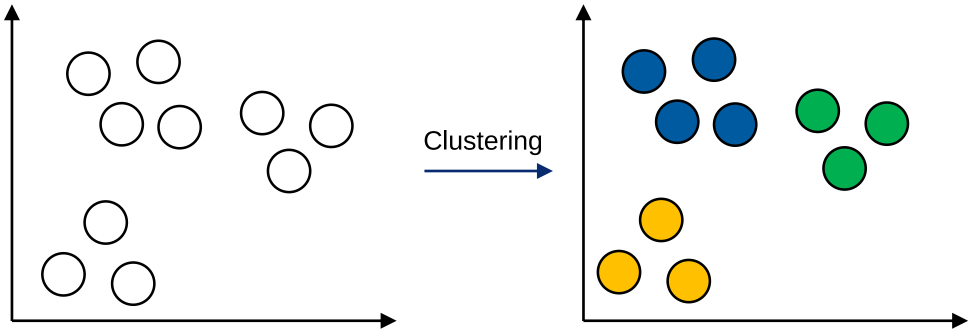
Agenda

- **Introduction**
  - Basics, terminology, similarity
- **Methods**
  - K-Means
  - DBSCAN
  - Self-organizing Maps (SOMs)
- **Cluster Validation**
  - SSE
  - Silhouette Coefficient
  - Dunn-Index
- **Summary**

Introduction

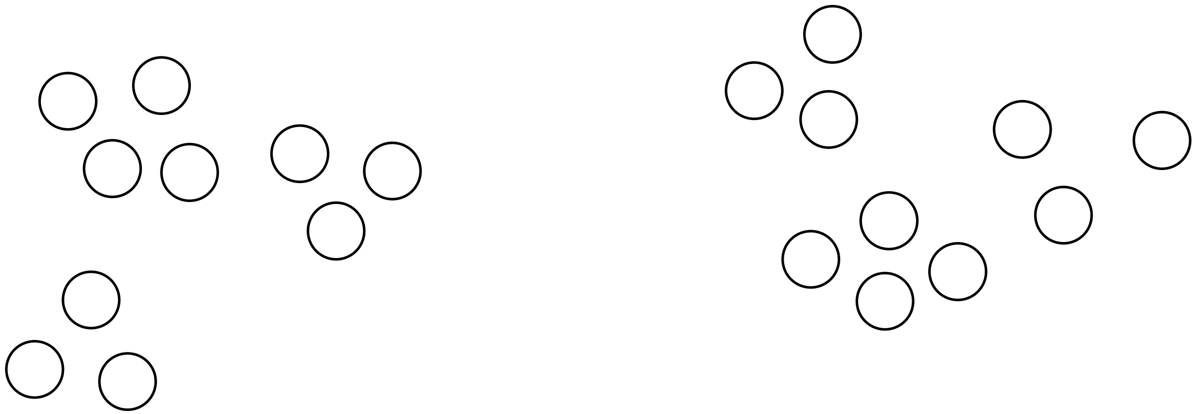
- **Unsupervised** machine learning
- Identify (disjoint) data **groupings**
- Utilizes **similarity** of items

- Typical application scenarios
  - Data exploration **(working hypothesis)**
  - Segmentation
  - Label generation for classification



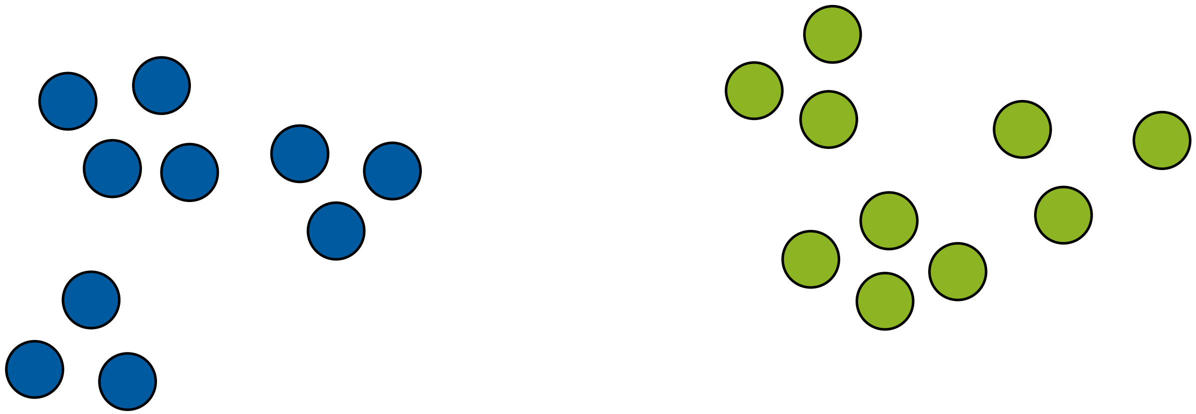
Introduction

What is a cluster?



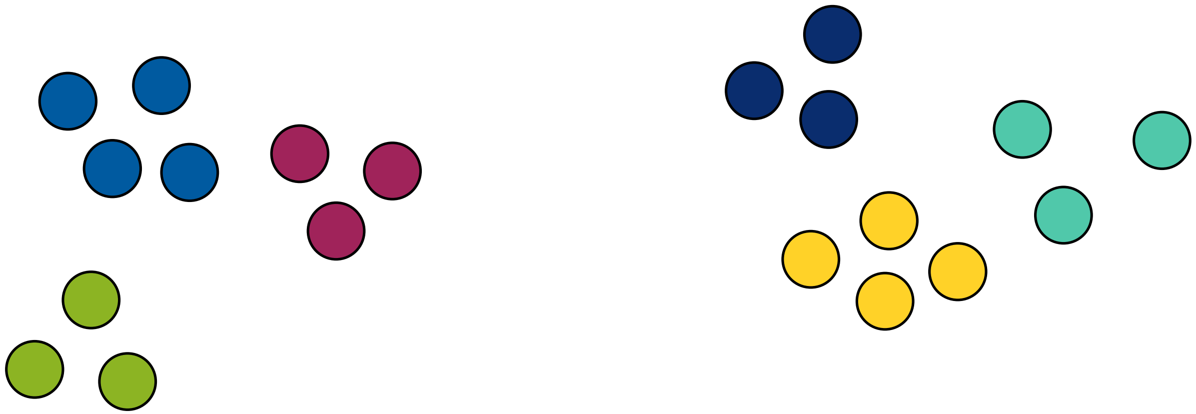
Introduction

What is a cluster?

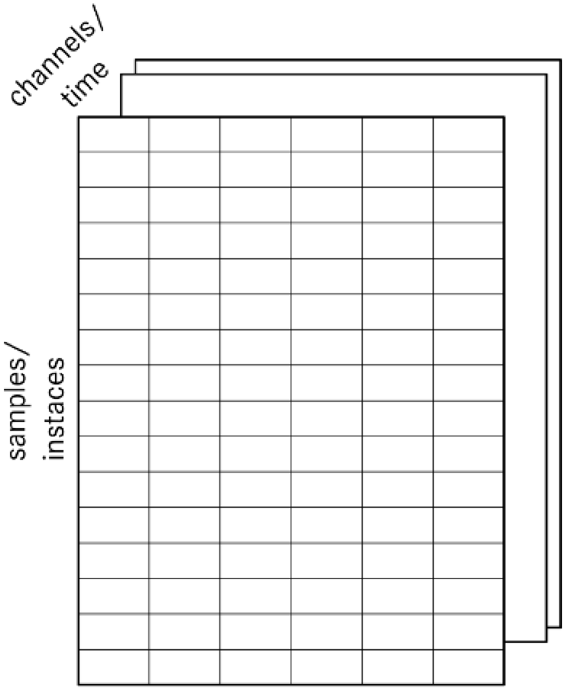


Introduction

What is a cluster?



Terminology



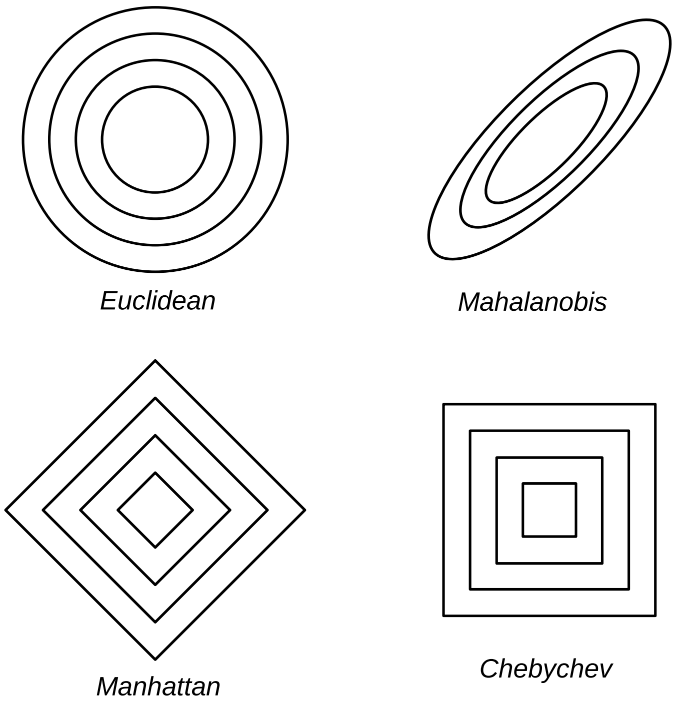
- **Samples**/instances
  - Examples: a measurement, ensemble members, an image
- **Features** or attributes are properties
  - Examples: surface temperature, surface coordinate, pixel color
- **Channels**
  - Examples: image color channels, spectral bands, time series

Measuring Similarity

… or rather dissimilarity

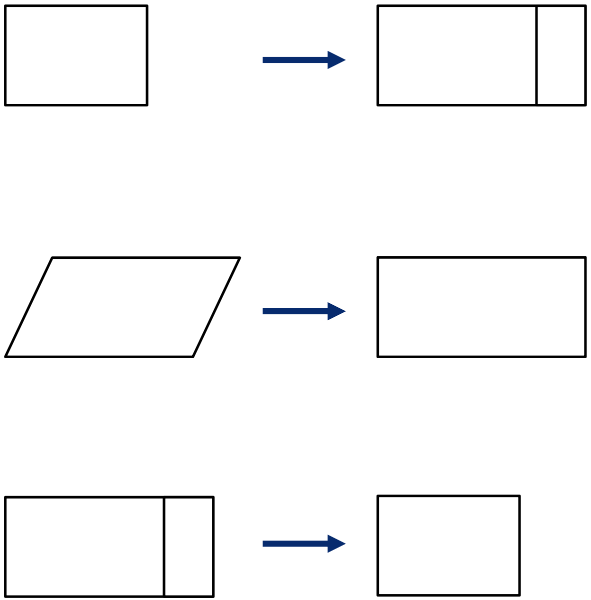
- Expressing similarity often hard
- Dissimilarity/**distances** alternative

- Minkowski distances
  - Manhattan (L_1)
  - **Euclidean** distances (L_2)
  - Arbitrary ($L_p = \sqrt[p]{\sum |x-y|^p}$)
- Other distances
  - Geodesic
  - **Mahalanobis**
  - Chebychev



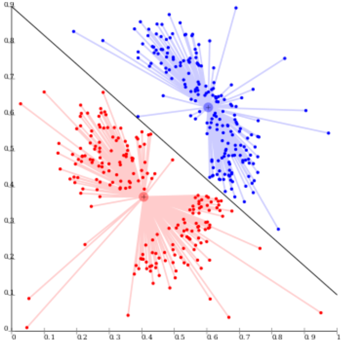
Preprocessing

Normalization and Feature Engineering

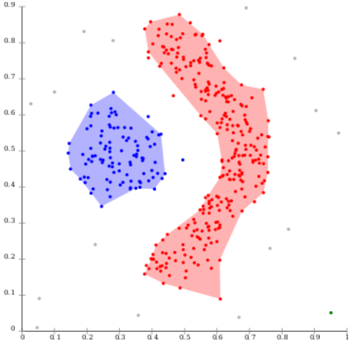


- **Feature Engineering**
  - Adding descriptive derived features
  - Mainly domain knowledge
- **Normalization**
  - Distance measures require same scales
  - $[0,1]$, standardization, unit length
- **Feature Reduction**
  - „Curse of dimensionality“
  - Achieve interpretability
  - Approaches: PCA, Autoencoder, …

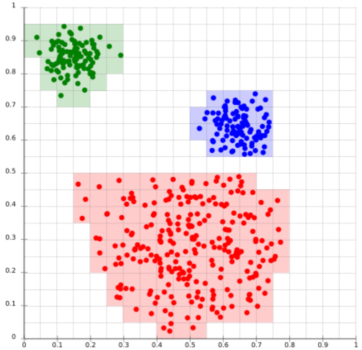
Clustering Approaches



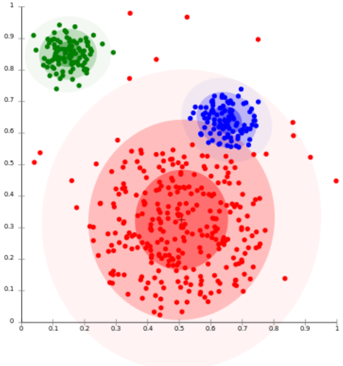
Centroid-based



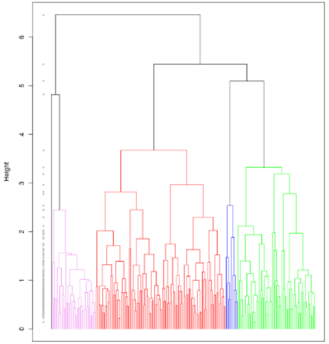
Density-based



Grid-based



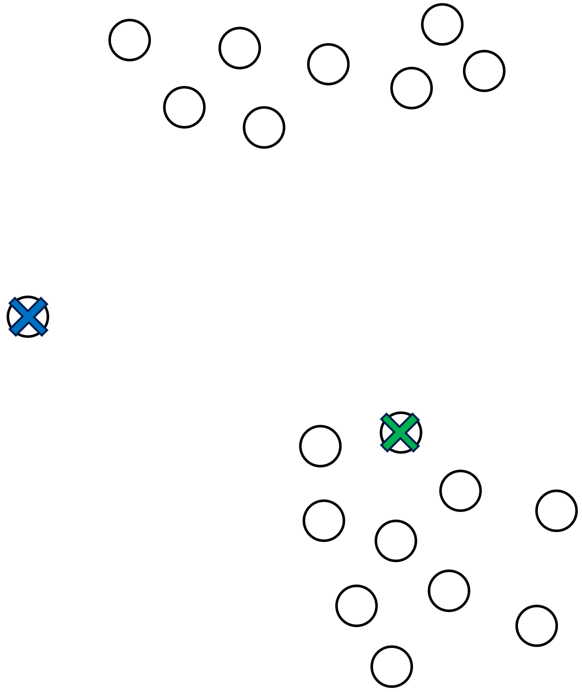
Distribution-based



Connectivity-based

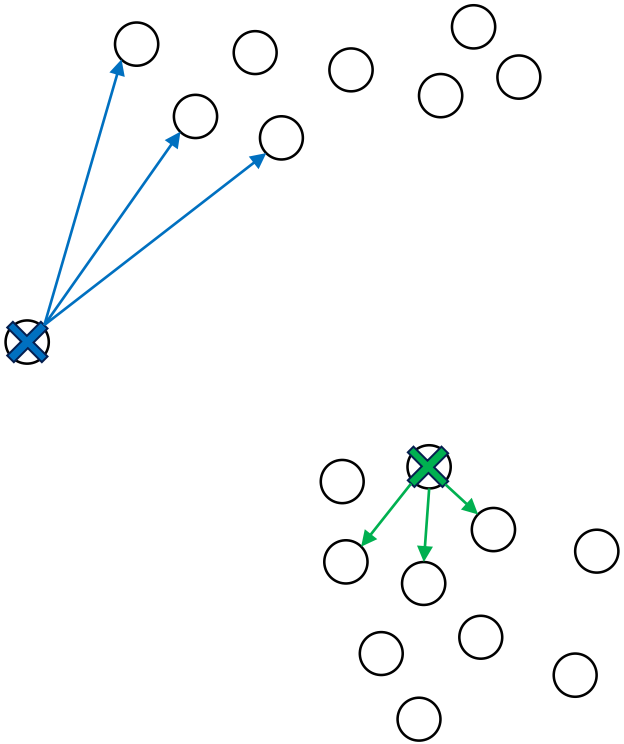
K-Means

- Core idea: ***k* clusters** around centroids
- Iterative minimization
  - $\arg\min_{C} \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \bar{x}\|^2$
  - Other matrics possible
- Algorithm sketch
  - Choose *k* centroids
  - For each points calculate distance to centroids
  - Assign point to **closest centroid**
  - Estimate new centroids as **mean** of points
  - Repeat until **convergence**



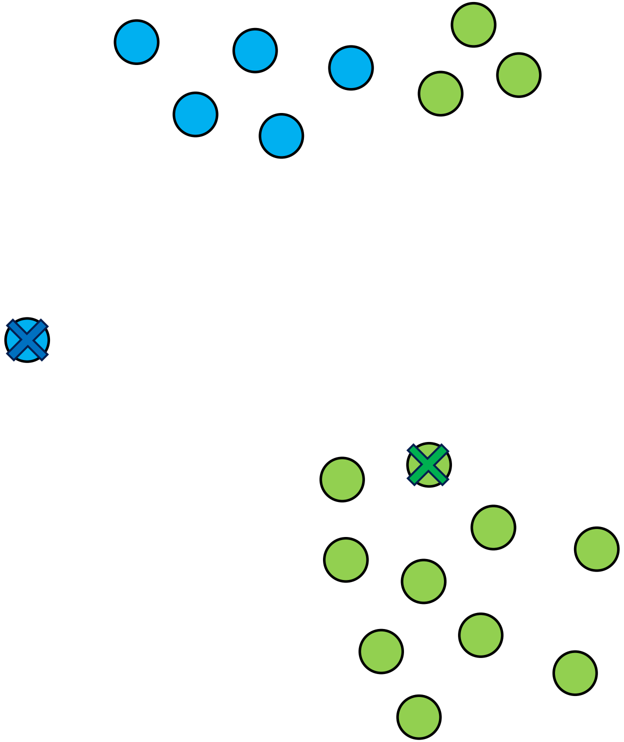
K-Means

Example



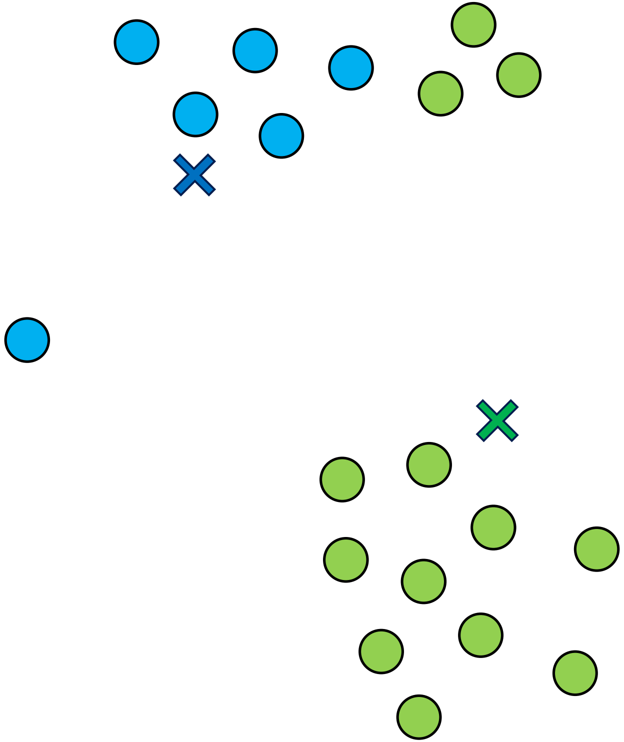
K-Means

Example



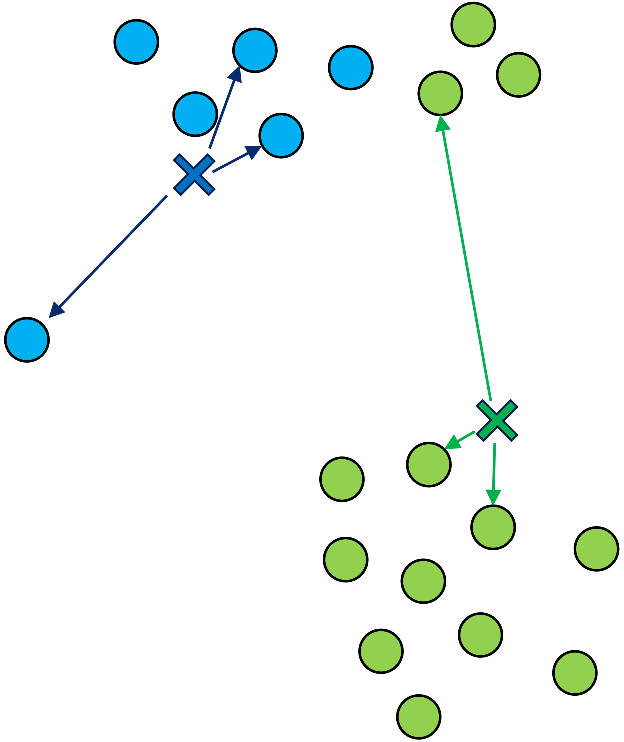
K-Means

Example



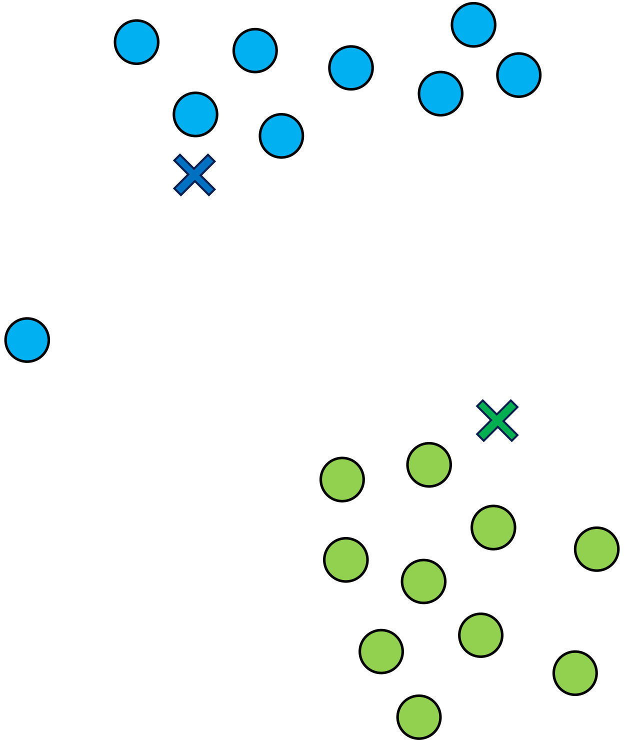
K-Means

Example



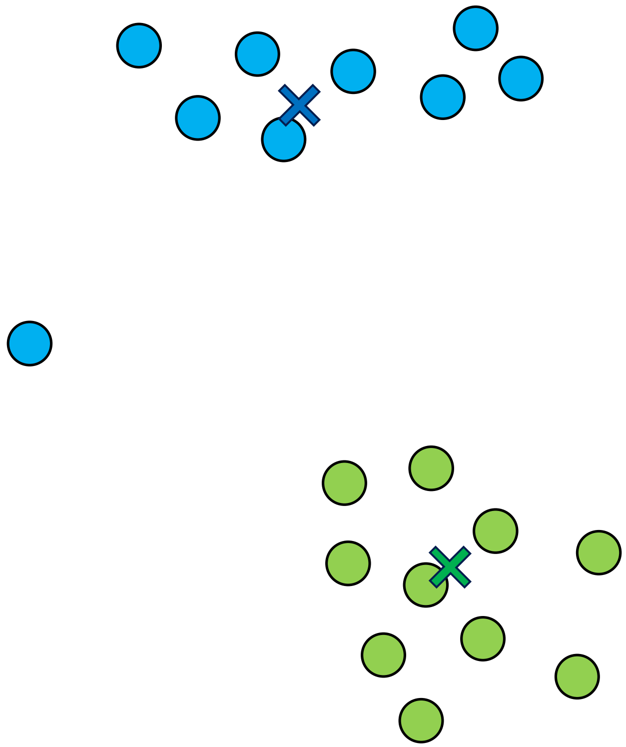
K-Means

Example



K-Means

Example

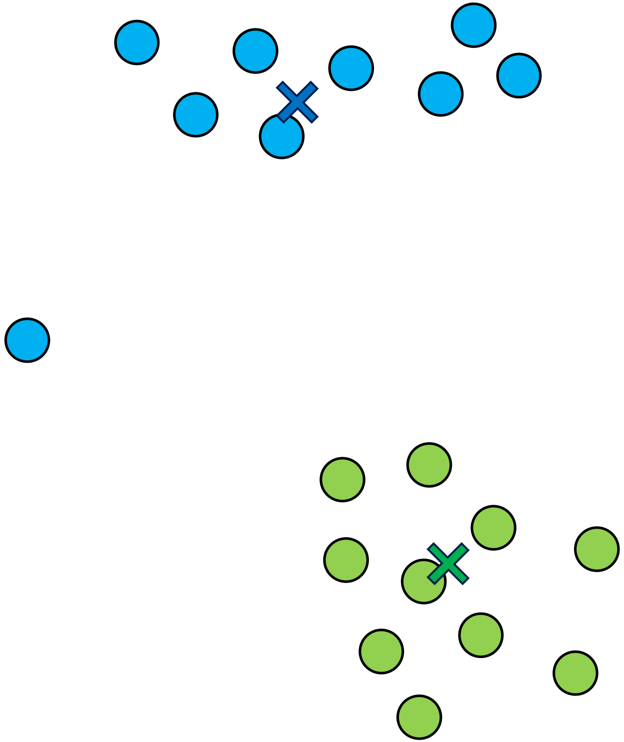


K-Means

- Centroid selection
  - Random sampling
  - Explicit specification
  - Heuristics (e.g. K-Means++)

- Estimating *k*
  - Domain knowledge
  - Multiple runs, **„elbow"-method**

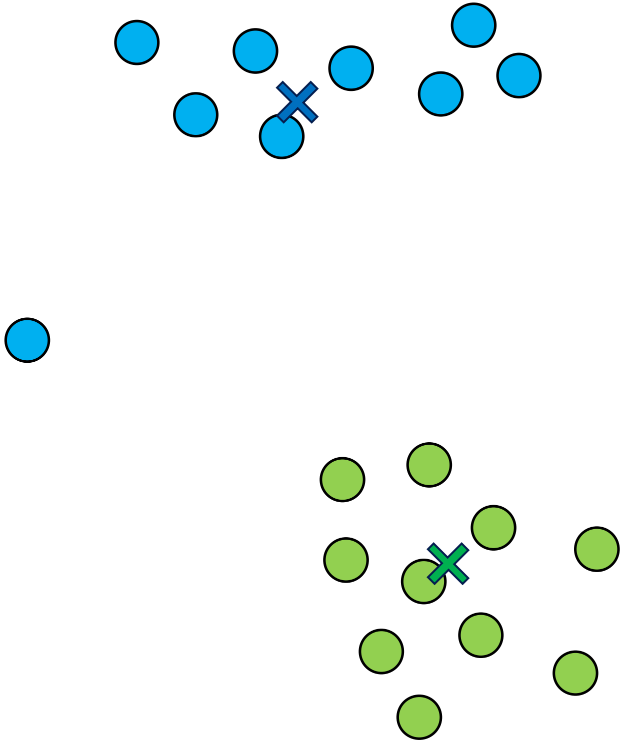
- Determining convergence
  - Centroid **movement** below **threshold** (ε)
  - Upper **iterations** bound



K-Means

Discussion

- Algorithmic properties
  - (Hyper-)**globular** clusters
  - Each point guaranteed to be in cluster
  - Susceptible to outliers (due to mean)
- Computational properties
  - **Non-deterministic,**
  - Time complexity: $\mathcal{O}(n \times k \times i)$
  - Space complexity: $\mathcal{O}(n + k)$
- Trivial to parallelize
- Extensions: k-mediods, fuzzy C-Means, batched



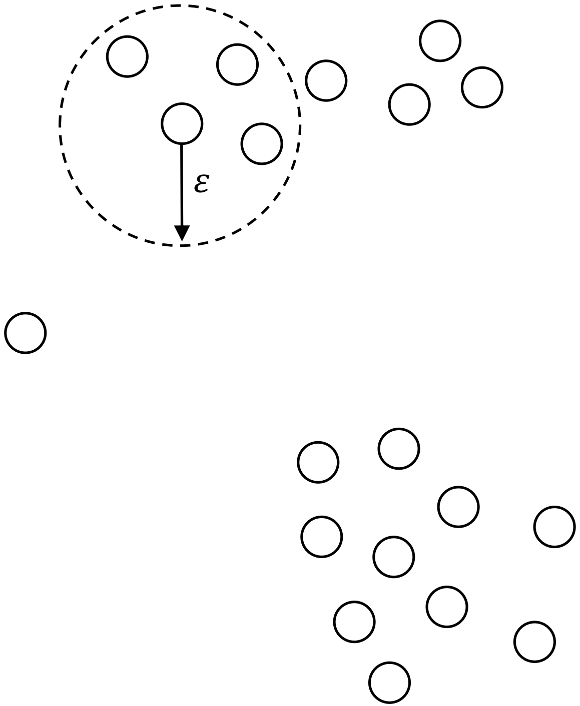
Demo

DBSCAN

- Core idea: **dense regions** are clusters

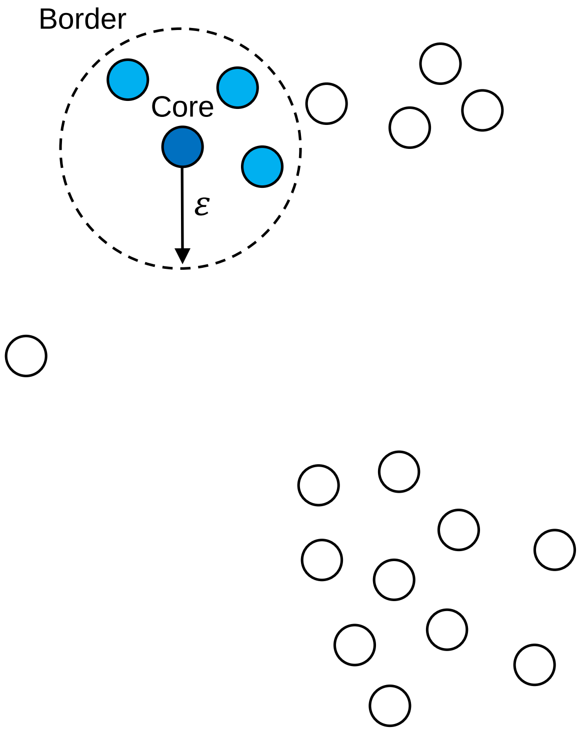
- Two parameters
  - $minPts$ – spatial search radius
  - ε – density threshold

- Algorithm sketch
  - For each point **perform spatial search**
  - If density criterion **fulfilled**, recursive **expansion**
  - Else **noise** identified
  - Continue with unvisited points



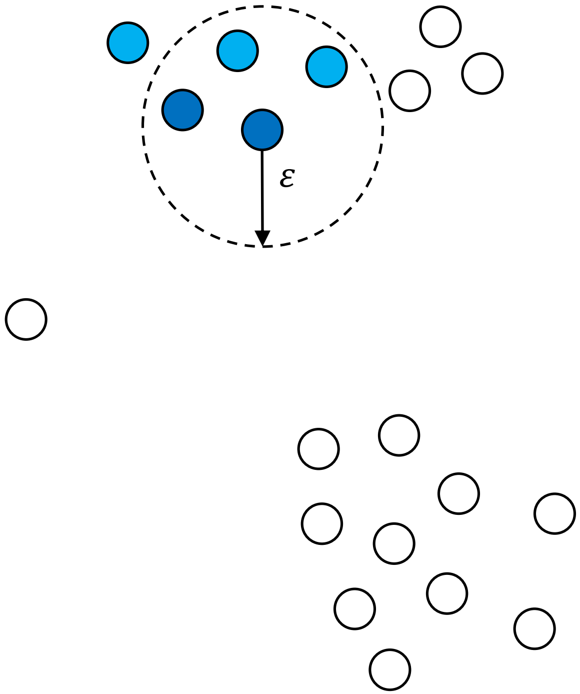
DBSCAN

Example



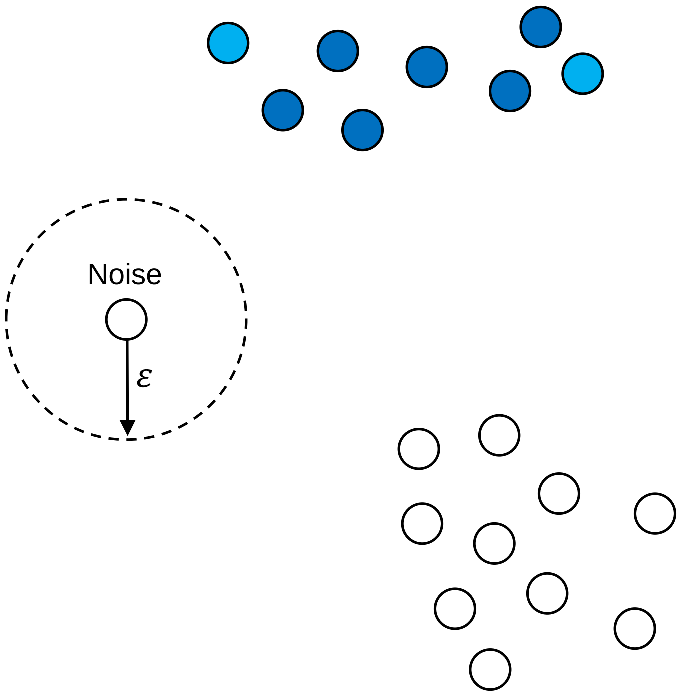
DBSCAN

Example



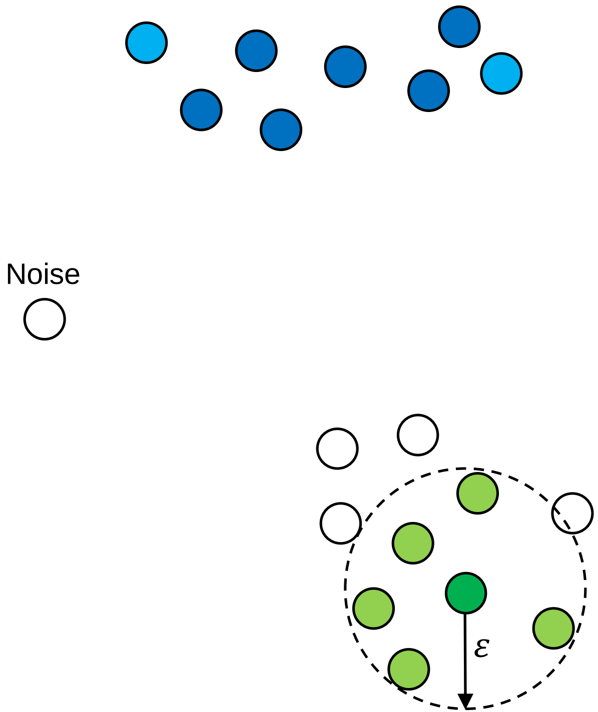
DBSCAN

Example



DBSCAN

Example



DBSCAN

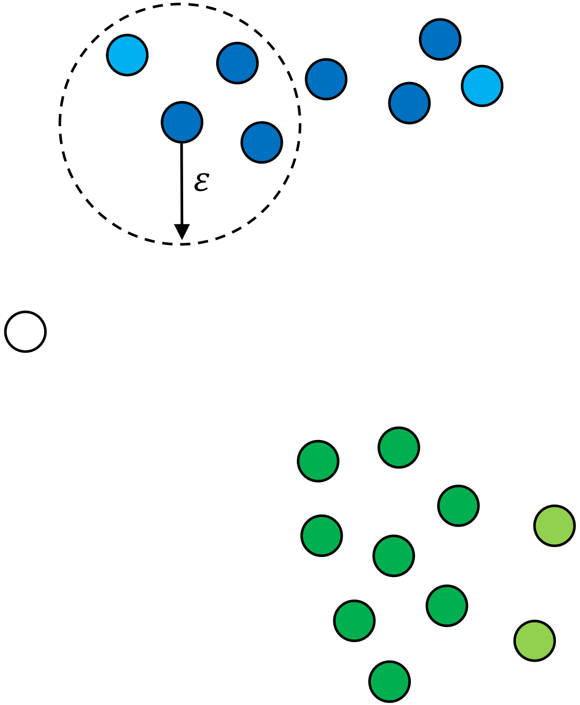
Discussion

- Algorithmic properties
  - Detects **noise**
  - Cluster **count** may be apriori **unknown**
  - **Arbitrary shapes**, except „bow ties“

- Computational properties
  - **Deterministic**
  - Time complexity: $\mathcal{O}(n \times \log(n))$
  - Space complexity: $\mathcal{O}(n)$

- Parallelized for Minkowski distances

- Extensions: SUBCLU, HDBSCAN, ST-DBSCAN



Demo

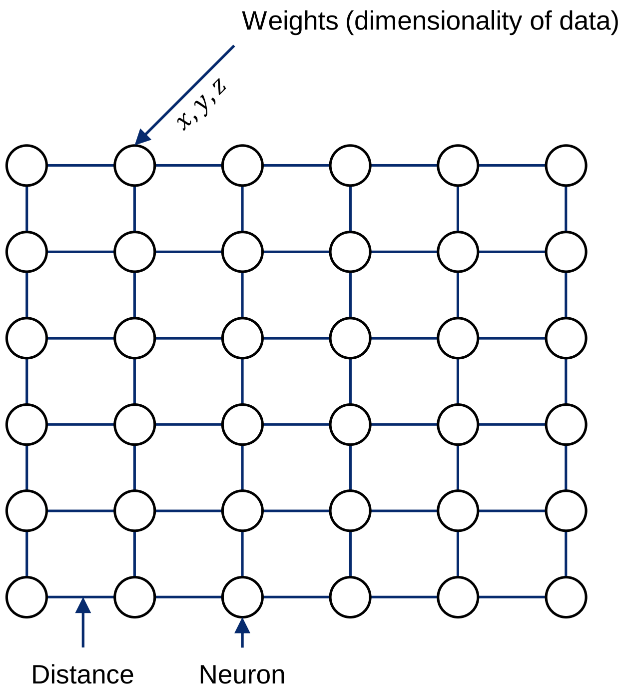
Self-organizing Maps (SOMs)

…or Kohonen-Network

- Core idea: „I am **not** a **clustering algorithm**“
  - **Dimensionality reduction** algorithm
  - Map data to **discrete, quantized** grid
  - Inherent structure enables clustering

- Form of **artificial neural network**
  - Unsupervised model
  - Not a gradient optimizer
  - Instead: **competitive learning**

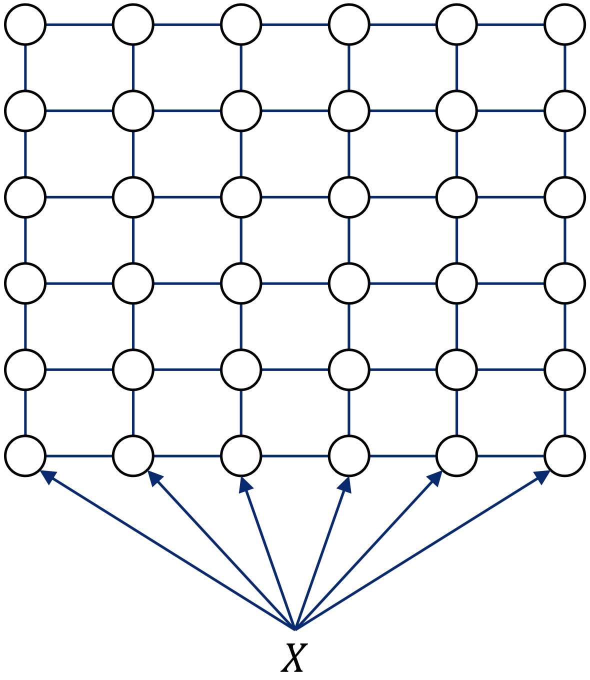
- Maintains high-dimensional topology



Self-organizing Maps (SOMs)

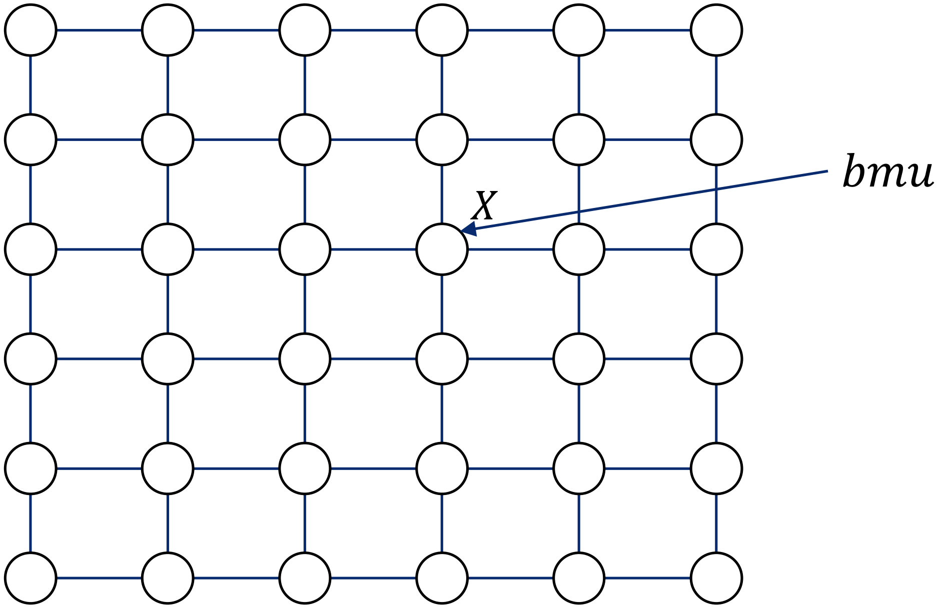
…or Kohonen-Network

- Highly flexible **toolkit**
  - Here: 2D, rectangular base form
  - Fixed grid-size, linear decays
- Algorithm sketch
  - **Randomly initialize** quantization weights
  - Determine **best-matching unit** (bmu) for samples X
  - **Update** all **weights** (gaussian distance to bmu)
  - $W_i(s+1) = W_i(s) + l(s) * r(bmu, s, i) * \left(X - W_i(s)\right)$
  - Decay learning-rate l and radius r
  - Repeat until convergence or **epoch** count reached



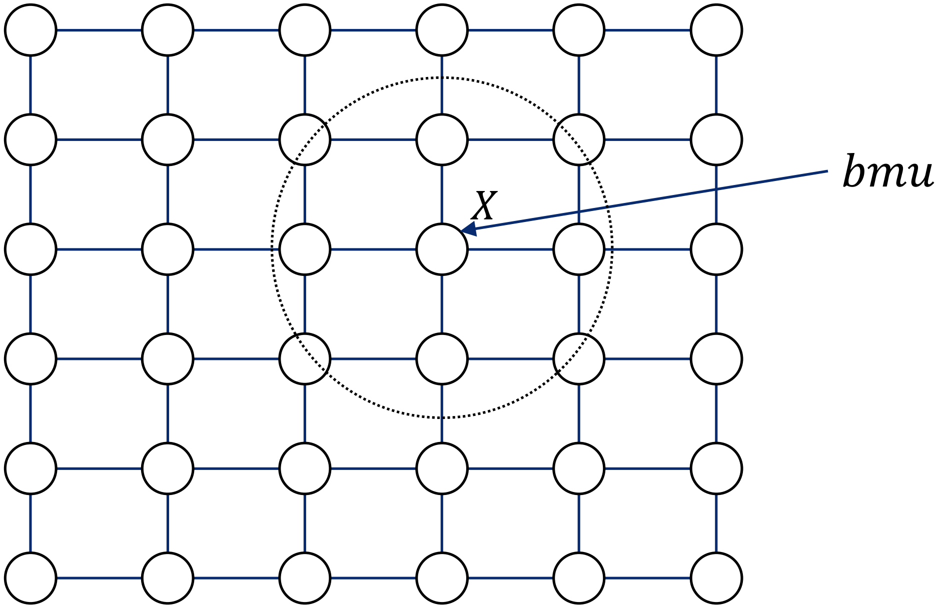
Self-organizing Maps (SOMs)

Example



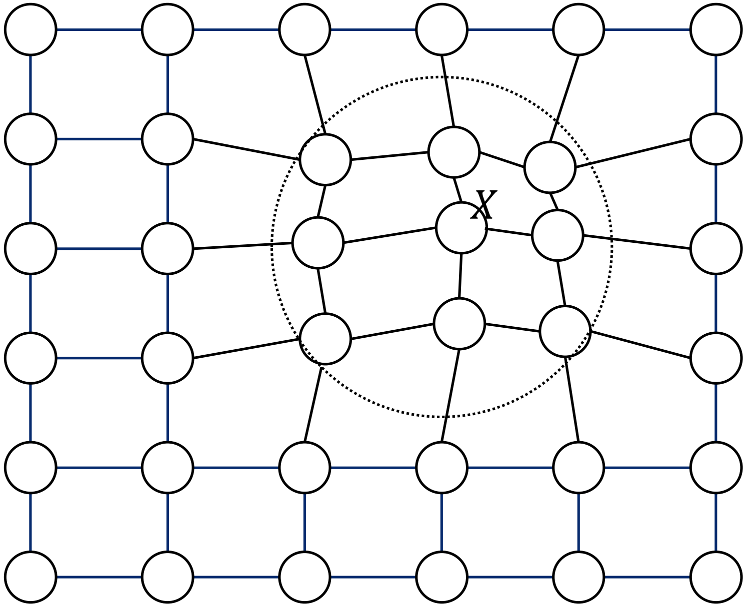
Self-organizing Maps (SOMs)

Example



Self-organizing Maps (SOMs)

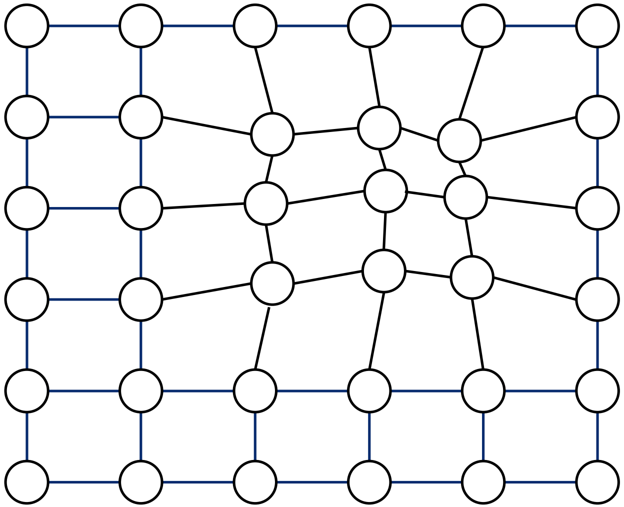
Example



Self-organizing Maps (SOMs)

Discussion

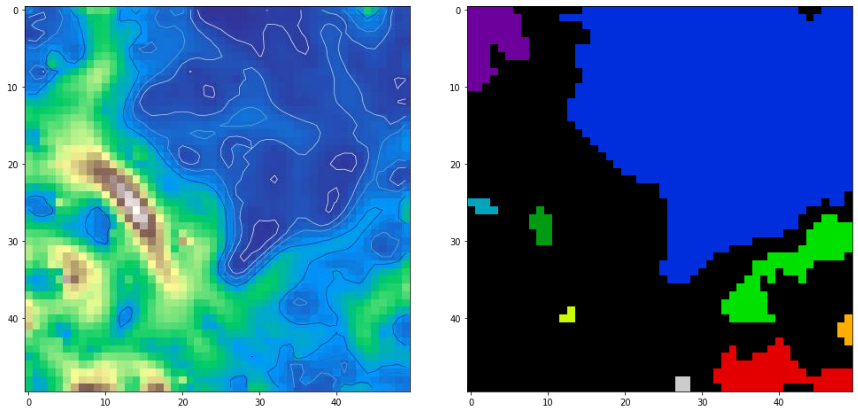
- Algorithmic properties
  - Topology-preserving, discrete quantization
  - Density-matching and feature selective
  - On-the-fly training (e.g. streams)
- Computational properties
  - **Expensive training**
  - Time complexity: $\mathcal{O}(e \times n \times \log(n))$
  - Space complexity: $\mathcal{O}(w \times h \times \cdots \times d)$
- Highly parallelizable
- Extensions: hexagonal grid, Growing SOMs



Self-organizing Maps (SOMs)

U-Matrix and Clustering

- U-Matrix computation and visualization
  - Matrix with shape of SOM
  - Each position maps **internal weight distances**
  - Usually **immediate neighbor** average
  - Visualization as SOM-dimensional image



- Cluster analysis
  - **Standard clustering** on neurons vectors
  - Threshold **connected-component labeling**
  - **Image processing** on U-Matrix
  - Map data items to bmu index, look up cluster map

Demo

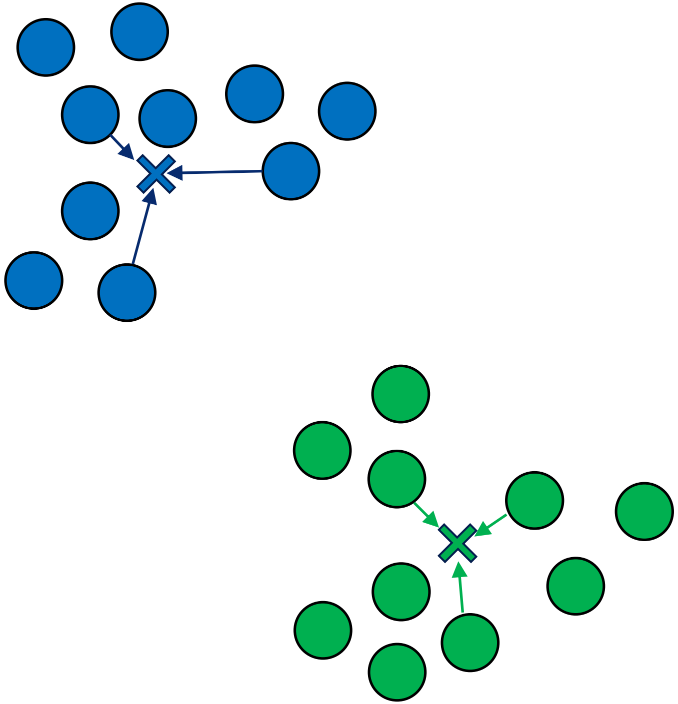
Cluster Validation

- Quantify clustering quality
- Compare different clustering algorithms

- **Domain** knowledge
- **External** measures
  - Compare to **ground truth** (labels)
  - May be more suitable classification task
- **Internal** measures
  - Works on data only, no reference
  - Based of **cohesion** and **separation**

Cluster Validation

Sum of Squared Errors (SSE)



- Measures distances to cluster nucleus
- Considers cluster **cohesion only**
- Purely **relative** measure
  - $SSE = \sum_{i=1}^{k} \sum_{p \in C_i} (p - \bar{p})^2$
  - $SD = \sum_{i=1}^{k} \sum_{p \in C_i} distance(p, \bar{p})$
- Tends to favors small, globular clusters

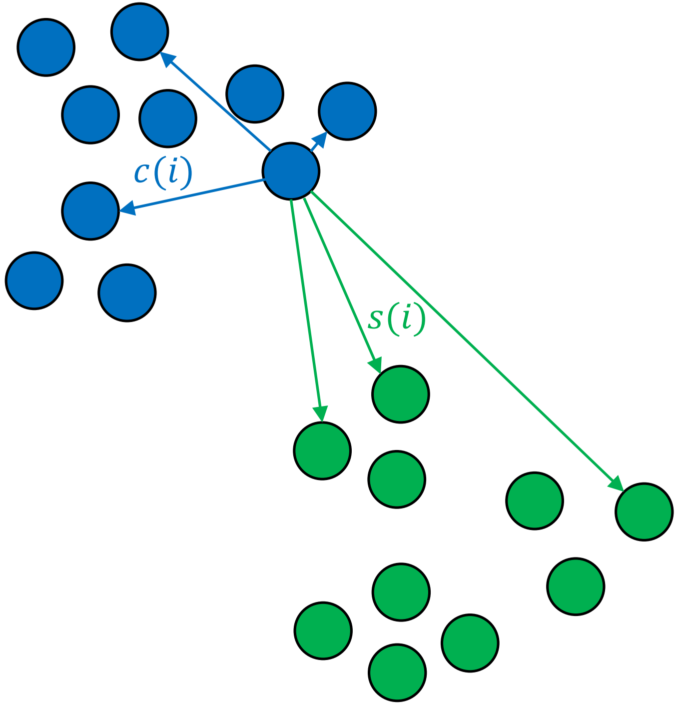
Cluster Validation

Silhouette Coefficient

- Balances separation (s) and cohesion (c)

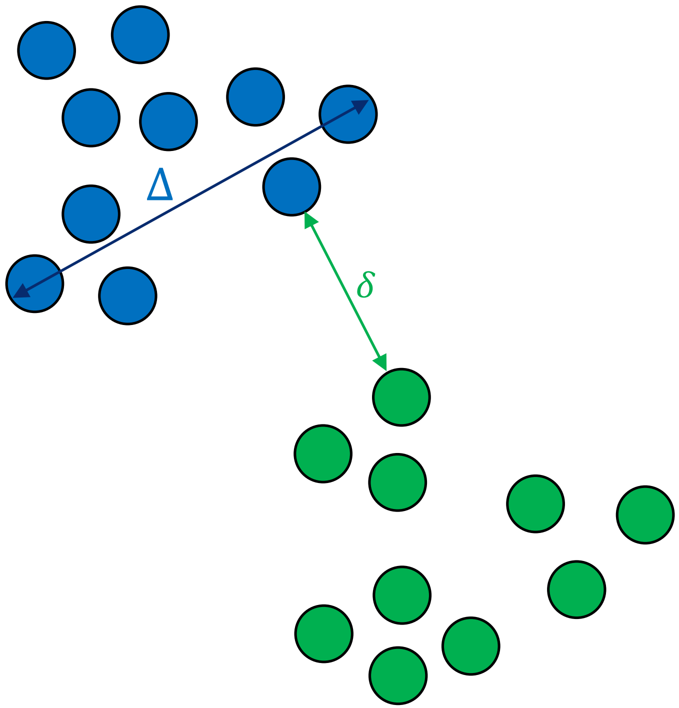
- For each data point i
  - $c(i) \triangleq$ average all-to-all intra-cluster distance
  - $s(i) \triangleq$ minimal average other-cluster distance
  - $sc(i) = \frac{s(i)-c(i)}{\max(s(i),c(i))}$

- Global $\overline{sc}$ allows to judge entire clustering
- Favors well separated clusters



Cluster Validation

Dunn-Index



- Global **worst-case** view of clustering
- Purely **relative measure**
  - Globally loosest cohesion (Δ)
  - Overall smallest separation (δ)
  - Dunn-Index is cohesion-separation-fraction
  - $\Delta_i = \max_{p,q \in C_i} distance(p,q)$
  - $\delta(C_i, C_j) = \min_{p \in C_i, q \in C_j} distance(p,q)$
  - $DI_m = \frac{\min_{1 \le i < j \le m} \delta(C_i, C_j)}{\max_{1 \le k \le m} \Delta_k}$
- Tends to favor many small clusters

Demo

Summary

- Visited clustering topics
  - Basics (terminology, similarity, preprocessing)
  - Algorithms
  - Internal result validation

- **Take-aways**
  - Cluster analysis is complex topic
  - Analysis quality depends on selected method

- **Invitation**: clustering application discussion

Discussion