Clustering II

COMP5318 Machine Learning and Data Mining

semester 1, 2023, week 10

Irena Koprinska

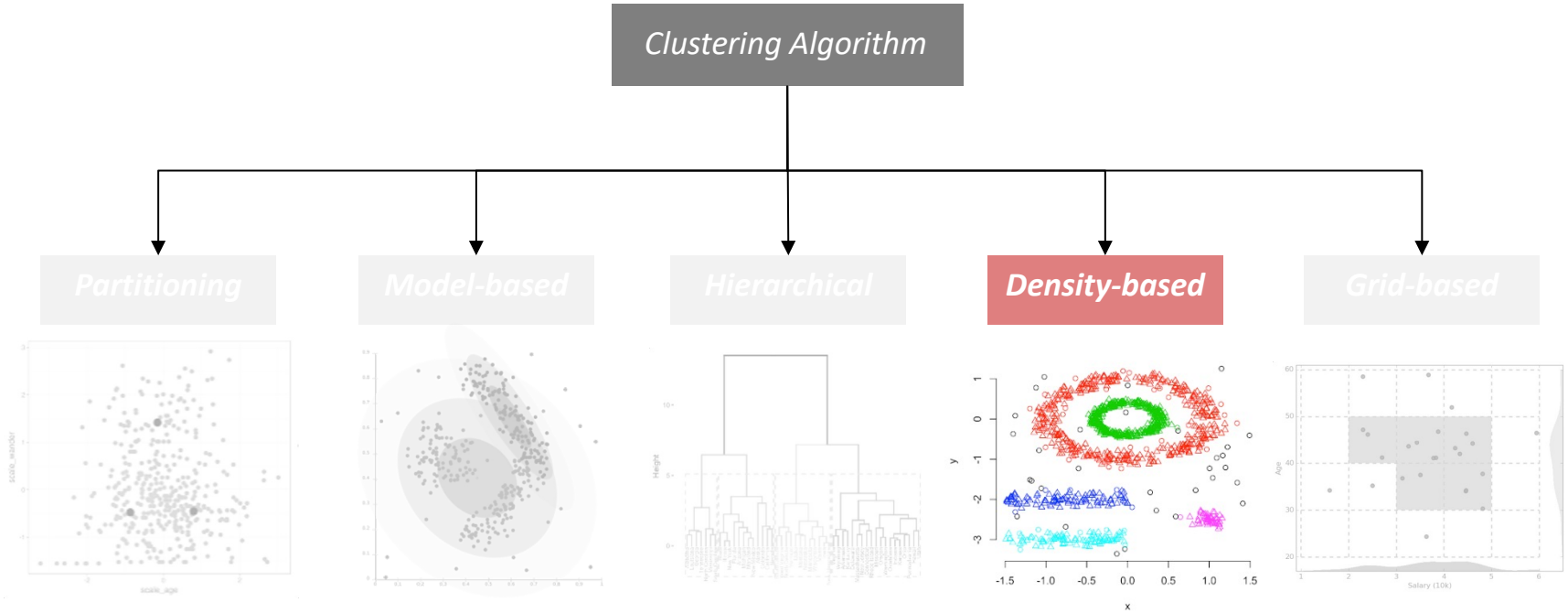
Reference: Tan ch.7.4-7.5, 8.3, Witten ch.4.8, Müller & Guido: ch.3.5, Geron: ch.9

THE UNIVERSITY OF SYDNEY

Outline

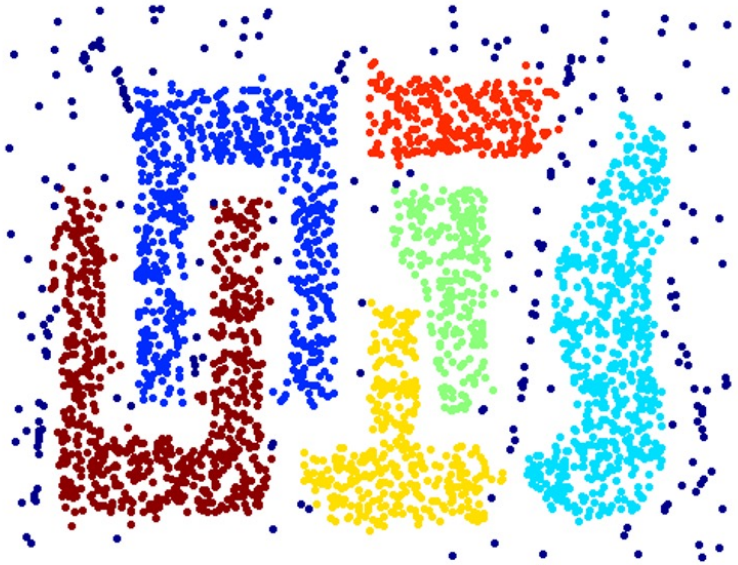
- Density-based clustering: DBSCAN
- Grid-based clustering
- Evaluating clustering results

Density-based: DBSCAN



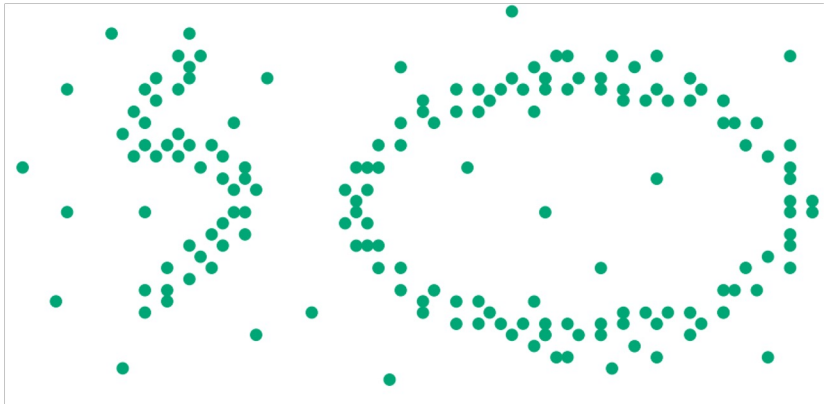
DBSCAN

- DBSCAN – **D**ensity-**B**ased **S**patial **C**lustering of **A**pplications with **N**oise
- Clusters are regions of high density, separated from one another by regions with low density



DBSCAN

- In contrast to K-Means which finds circular clusters, DBSCAN can find clusters with arbitrary and complex shape, e.g. S shape, ovals, half circles

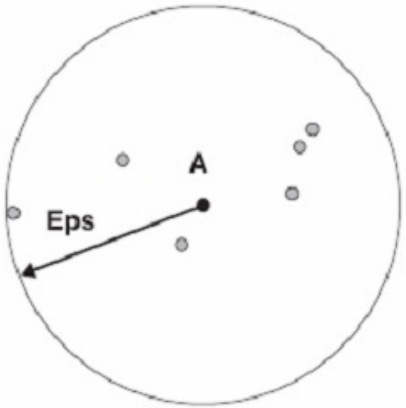


We will apply DBSCAN to the moons dataset during the tutorial!

Main idea

- If point A is in a cluster, the density of the points around it should be higher than a threshold
  - We need to define density and neighborhood of a point

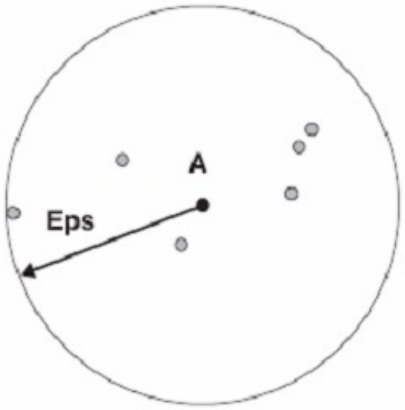
- Neighborhood of a point A = the area within a radius Eps
- Density of a point A = the number of points in the neighborhood of A, including the point A
- Density threshold MinPts – the minimum number of points in the Eps neighborhood of a point



What is the density of A?

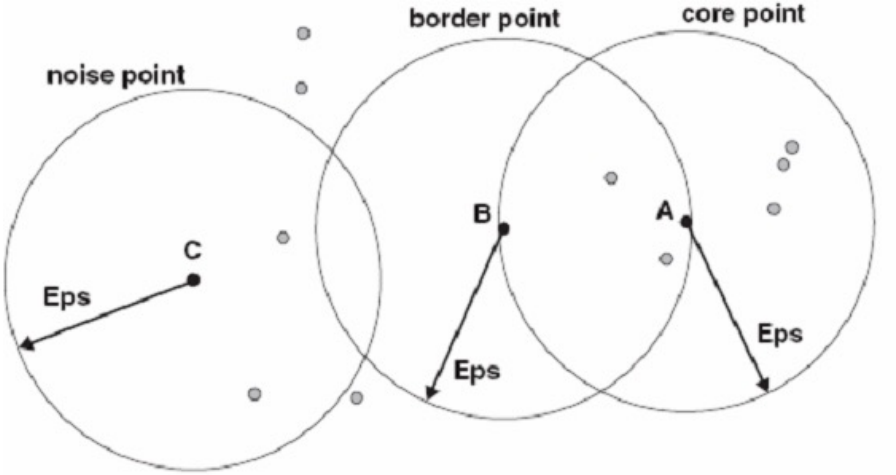
Density depends on Eps

- The density of a point depends on the neighborhood Eps
  - Eps too big - all points will have a density of *m* (*m* =number of points in the dataset)
  - Eps is too small – all points will have a density of 1



Three types of points in DBSCAN

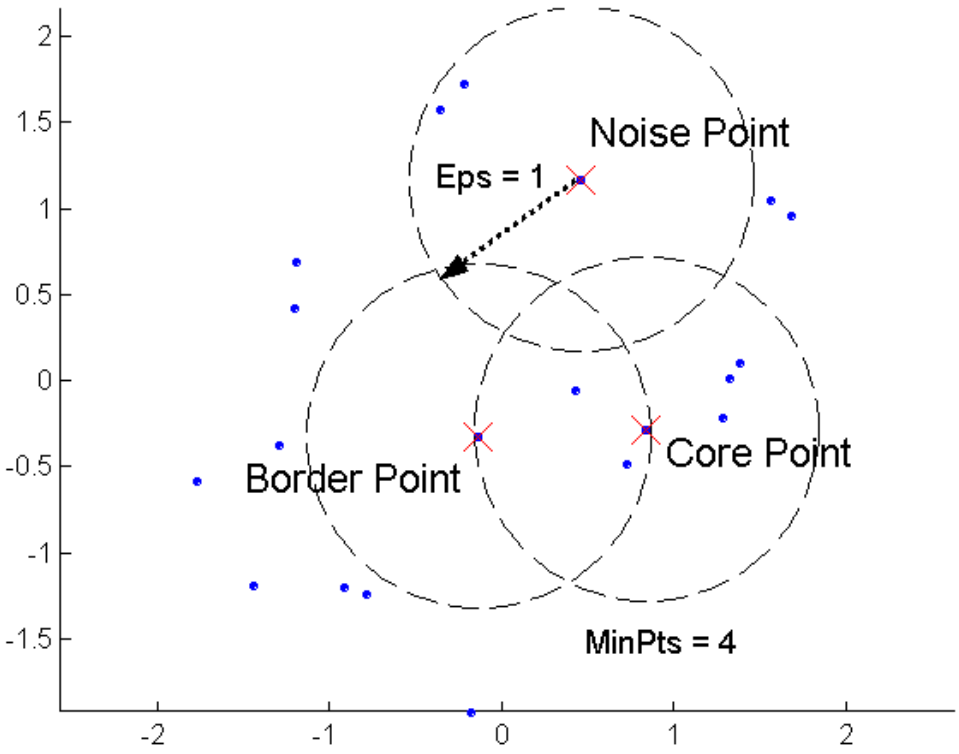
- There are 3 types of points in DBSCAN: core, border and noise
- Given Eps and MinPts, a point X is a:
  - Core point – if its Eps neighborhood contains at least MinPts points. Core points are in the interior of a density-based cluster.
  - Border point – if it is not a core point but falls within the neighborhood of a core point. A border point may fall in the neighborhood of several core points.
  - Noise point – if it is not a core or a border point



For Eps and MinPts =7:
A – core
B – border
C – noise

Core, border and noise points - example

- Another example showing the 3 types of points:



DBSCAN algorithm

1. Label points as core, border and noise
2. Discard noise points
3. Cluster the remaining points as follows
   - Any 2 core points within Eps of each other are put in the same cluster
   - Any border point from the neighborhood of a core point is put in the same cluster as the core point
   - Ties need to be resolved when a border point belongs to the neighborhood of more than 1 core point

Example

- Given are 5 items and their distance matrix. What clusters will DBSCAN find for Eps=2 and MinPts=3?

	A	B	C	D	E
A	0	1	4	5	6
B	1	0	2	6	7
C	4	2	0	3	4
D	5	6	3	0	1
E	6	7	4	1	0

Hint:

1. Find neighborhoods using Eps<=2
2. Label points using MinPts>=3
3. Find clusters

Solution

- Given are 5 items and their distance matrix. What clusters will DBSCAN find for Eps=2 and MinPts=3?

	A	B	C	D	E
A	0	1	4	5	6
B	1	0	2	6	7
C	4	2	0	3	4
D	5	6	3	0	1
E	6	7	4	1	0

The neighborhood of A contains 2 points: A and B; don't forget to include the point itself

1. Find the neighborhoods for Eps<=2:

N(A)={A, B}

N(B)={B, A, C}

N(C)={C, B}

N(D)={D, E}

N(E)={E, D}

2. Label points as core, border and noise

MinPts>=3: core, border or noise?

A: border or noise?

B: core

C: border or noise?

D: border or noise?

E: border or noise?

3. Find clusters

{A,B,C} - 1 cluster only

Example 2

- Given are 5 items and their distance matrix. What clusters will DBSCAN find for Eps=1 and MinPts=2?

	A	B	C	D	E
A	0	1	4	5	6
B	1	0	2	6	7
C	4	2	0	3	4
D	5	6	3	0	1
E	6	7	4	1	0

Hint:

1. Find neighborhoods using Eps<=1
2. Label points using MinPts>=2
3. Find clusters

Algorithm 8.4 DBSCAN algorithm.

1: Label all points as core, border, or noise points.
2: Eliminate 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.

Solution

- Given are 5 items and their distance matrix. What clusters will DBSCAN find for Eps=1 and MinPts=2?

	A	B	C	D	E
A	0	1	4	5	6
B	1	0	2	6	7
C	4	2	0	3	4
D	5	6	3	0	1
E	6	7	4	1	0

1. Find the neighborhoods for Eps<=1:

N(A)={A, B}

N(B)={B, A}

N(C)={C}

N(D)={D, E}

N(E)={E, D}

2. Label points as core, border and noise

MinPts>=2: core, border or noise?

A: core

B: core

C: border or noise?

D: core

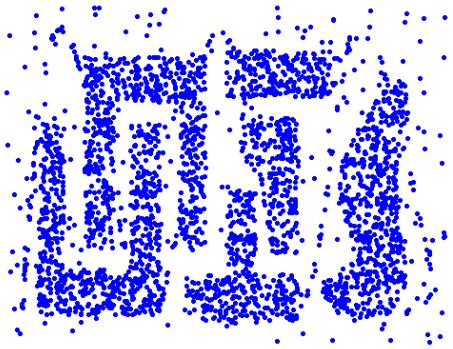
E: core

3. Find clusters

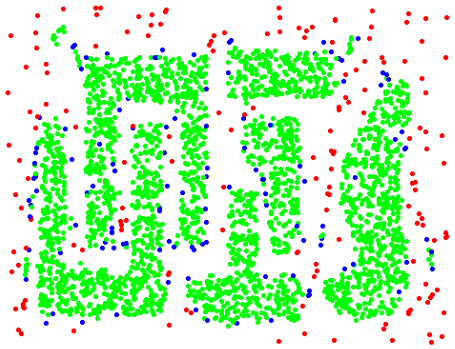
{A,B}, {D,E} – 2 clusters

How many clusters?

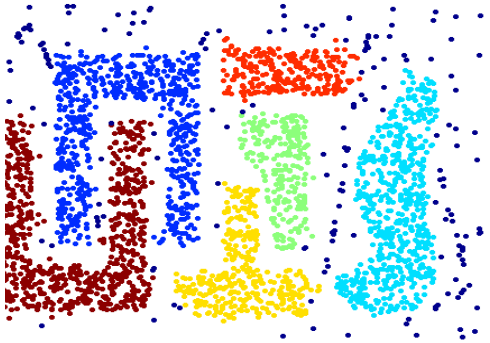
- Different number of clusters depending on Eps and MinPts



Original points

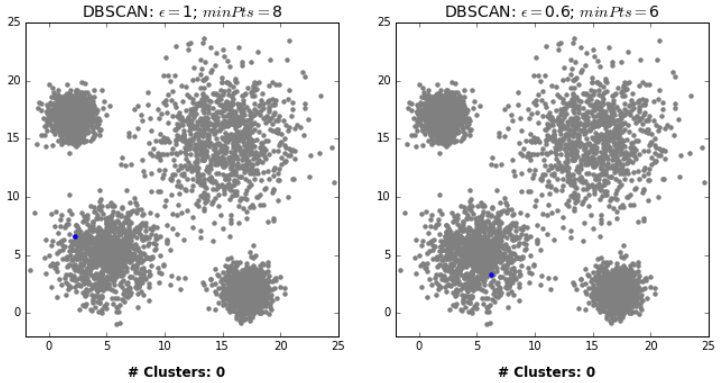


1 cluster



6 clusters

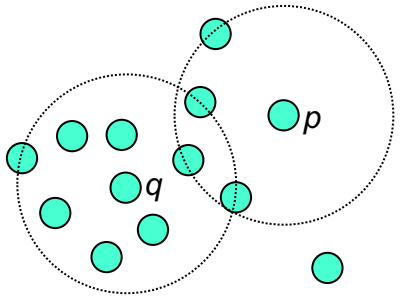
core, border and noise points



https://dashee87.github.io/images/DBSCAN_search.gif

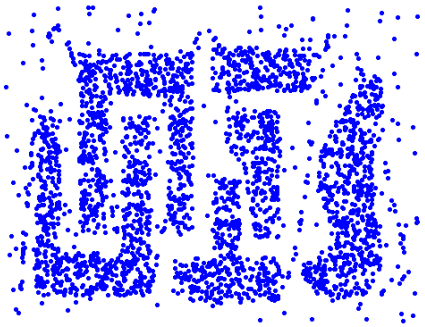
How to select Eps and MinPts?

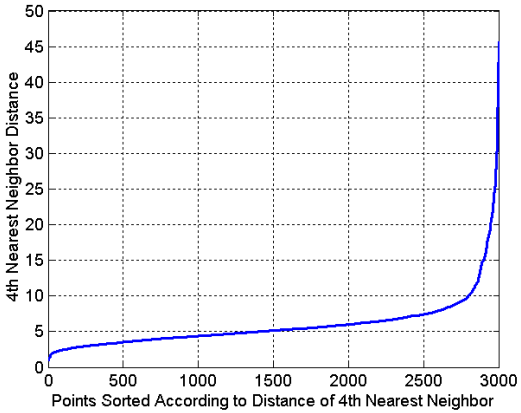
- If Eps is too big, then all points will be core points
- If Eps is too small, then all points will be noise



Method for selecting Eps and MinPts

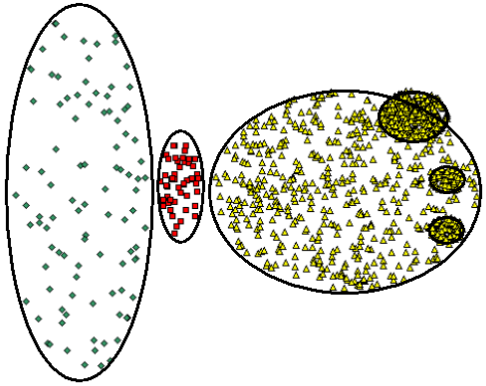
- Examine the distance from a point to its k-th nearest neighbor (k-dist)
  - For points that belong to some cluster, k-dist will be small (if k is not larger than the cluster size)
  - For points that are not in a cluster, such as noise points, k-dist will be large
- Hence, we can compute the k-dist for all data points for some k (e.g. for k=4), sort them in increasing order and plot the graph. A sharp increase in k-dist indicates a suitable distance value for Eps and MinPts can be set to k.
- If we do this, the points for which k-dist is less than Eps will be labelled as core points, while the others will be labelled as noise or border points
- However, this method requires k



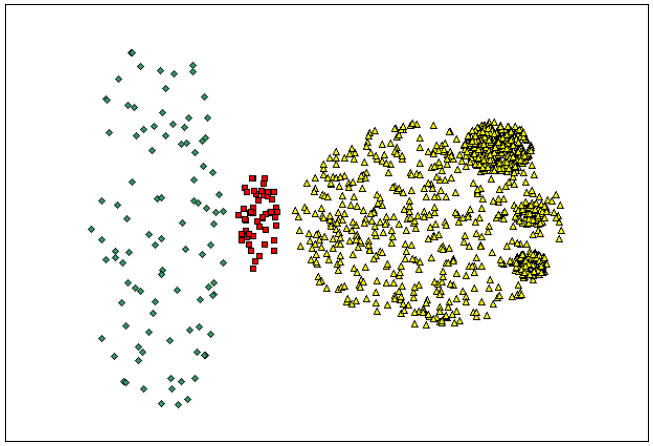


DBSCAN with clusters of varying density

- Cannot handle well clusters with widely varying density
- Different results depending on the parameters Eps and MinPts

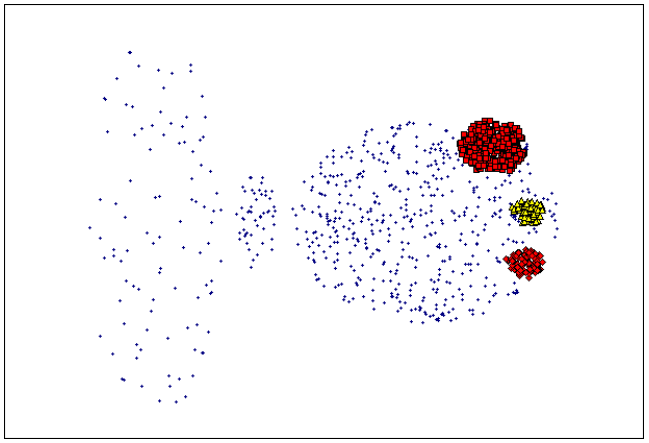


Original points



DBSCAN

MinPts=4, Eps=9.75



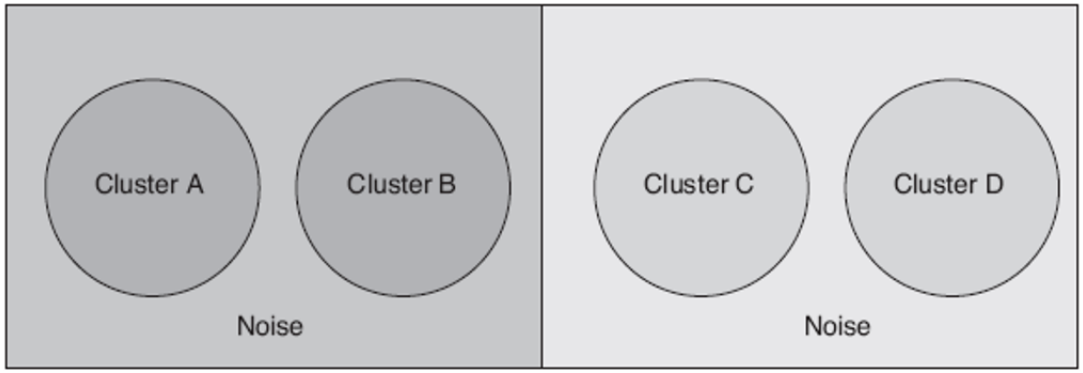
DBSCAN

MinPts=4, Eps=9.92

DBSCAN with clusters of varying density (2)

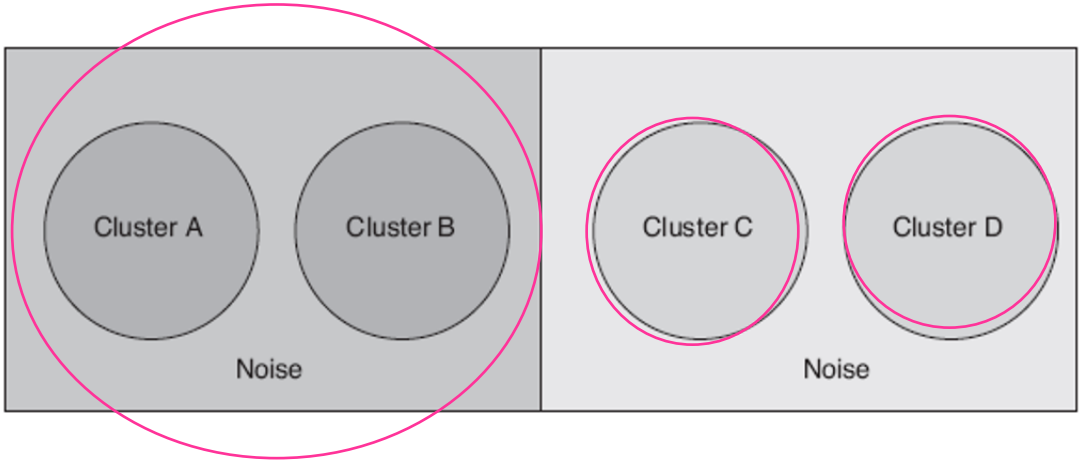
- Another example
  - 4 clusters: A, B, C and D, embedded in noise
  - The darker the color, the higher the density
  - The noise around A and B has the same density as clusters C and D

- How many clusters will DBSCAN find?



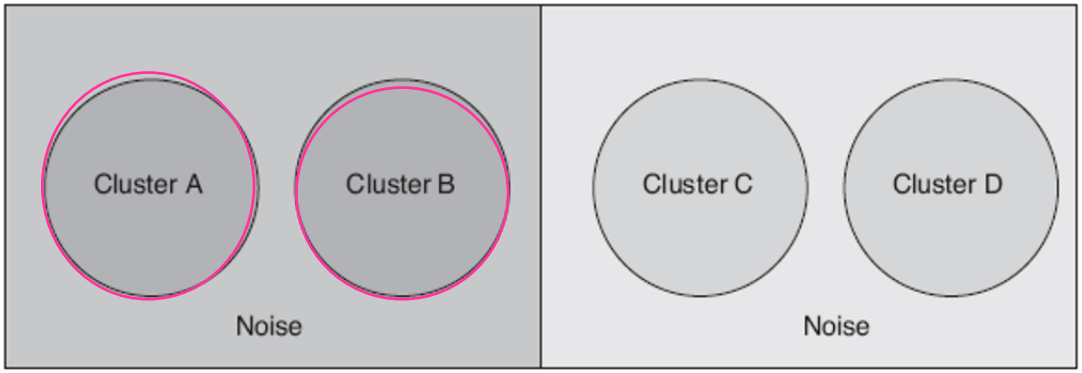
How many clusters?

- For a fixed MinPts, if Eps is high (big enough) so that C and D are located as separate clusters and the points around them as noise, then A, B and the points around them will become 1 cluster
- => There will be 3 clusters: {C}, {D}, {A,B, noise around them}



How many clusters? (2)

- For a fixed MinPts, if Eps is low (small enough) so that A and B can be located as separate clusters and the points around them are marked as noise, then C, D and the points around them will be marked as noise
- => There will be 2 clusters: {A}, {B}



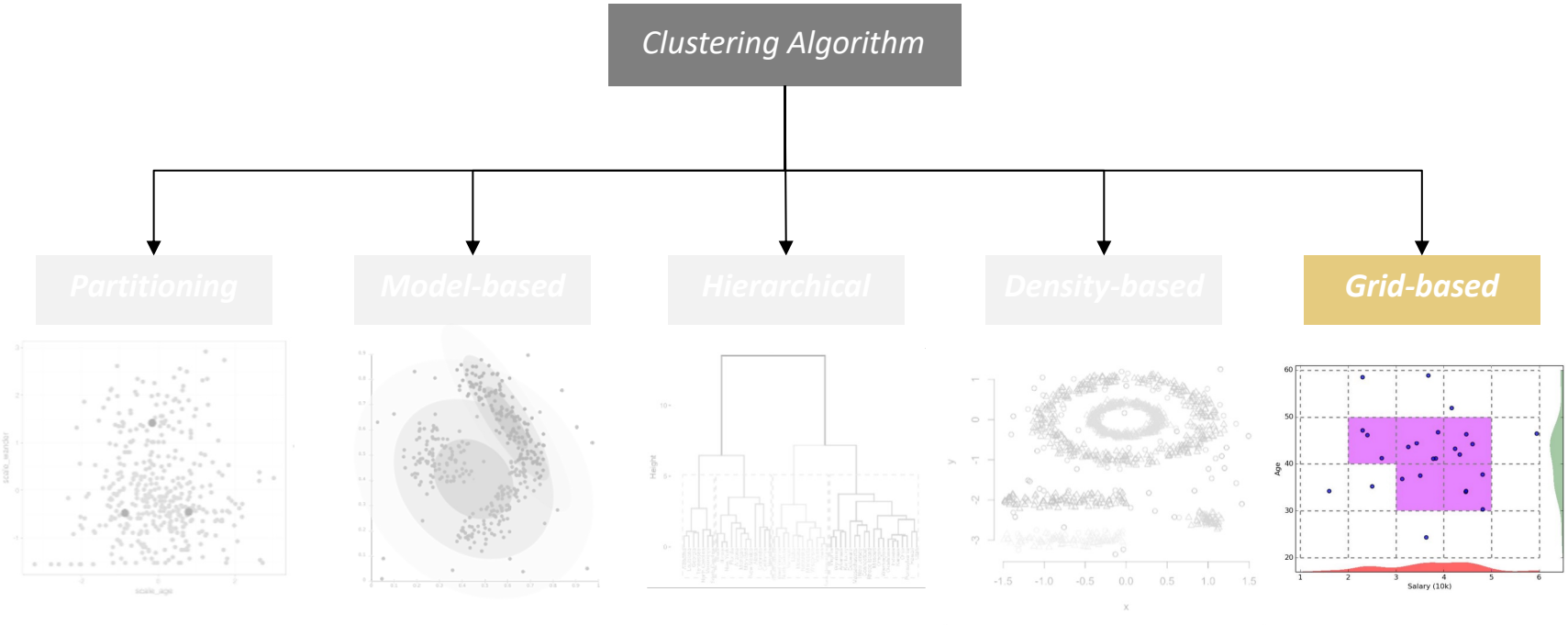
DBSCAN – time and space complexity

- Time – $O(n^2)$, n - number of points
  - Can be reduced to $O(n \log n)$ in low dimensional spaces using kd-trees
- Space – $O(n)$
  - Requires to keep a small amount of information for each point – its cluster and type: core, border or noise

DBSCAN – strengths and weaknesses

- Strengths:
  - Can form clusters of arbitrary shapes and different sizes => can find clusters that K-means cannot
  - Does not require the number of clusters to be pre-specified (but this is done indirectly by specifying Eps and MinPts)
  - Resistant to noise as it uses a density-based definition of a cluster and labels points as noise
- Weaknesses:
  - Does not work well for clusters with widely varying density
  - Does not work well for high dimensional data – more difficult to define density
  - Sensitive to the input parameters Eps and MinPts
  - Eps and MinPts may be difficult to determine – heuristic approaches to determine their values

Grid-based clustering

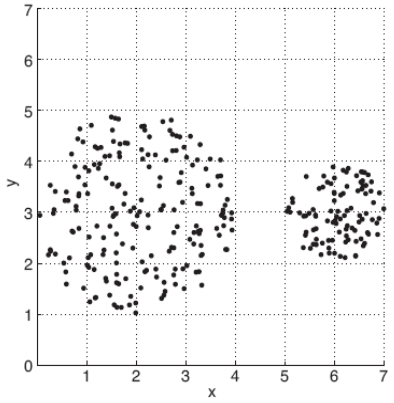


Grid-based clustering – main idea

- Grid-based clustering is a density-based clustering using a grid
- Main idea: break the data into grid cells and then form clusters from the cells that are dense enough

Algorithm 9.4 Basic grid-based clustering algorithm.

1: Define a set of grid cells.
2: Assign objects to the appropriate cells and compute the density of each cell.
3: Eliminate cells having a density below a specified threshold, τ.
4: Form clusters from contiguous (adjacent) groups of dense cells.



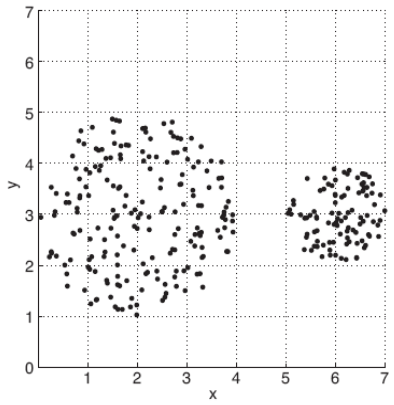
0	0	0	0	0	0	0
0	0	0	0	0	0	0
4	17	18	6	0	0	0
14	14	13	13	0	18	27
11	18	10	21	0	24	31
3	20	14	4	0	0	0
0	0	0	0	0	0	0

Defining grid cells

- Split the values of each attribute into intervals, creating grid cells
- Common approach: split the values of each attribute into equal width intervals
  - The resulting grid cells will have the same volume
  - The density of the cell will be the number of points in the cell
- More sophisticated approaches
  - Break the values of an attribute into intervals so that each interval contains an equal number of points (equal frequency intervals)
  - Use clustering to determine the intervals
  - Initially break the attribute values into a larger number of equal width intervals and then combine intervals with similar density
- The grid cell greatly impacts the clustering results – different grid cells, different clustering results

Density of grid cells

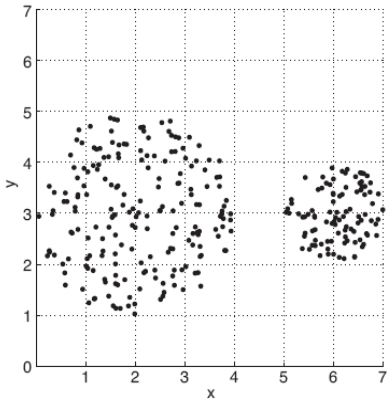
- Density of a grid cell = number of points / volume of the cell
  - i.e. the density is the number of points per amount of space
- Examples:
  - 1-dim: number of road signs per kilometer
  - 2-dim: number of kangaroos per square kilometer of habitat
  - 3-dim – number of molecules of a gas per cubic centimeter
- When we use the grid cells with the same volume, the number of points per cell is a direct measure of the cell's density



0	0	0	0	0	0	0
0	0	0	0	0	0	0
4	17	18	6	0	0	0
14	14	13	13	0	18	27
11	18	10	21	0	24	31
3	20	14	4	0	0	0
0	0	0	0	0	0	0

- The grid cells have equal volume (area)
- => The counts for the grid cells can be considered as the density of the cells

Forming clusters from dense grid cells



0	0	0	0	0	0	0
0	0	0	0	0	0	0
4	17	18	6	0	0	0
14	14	13	13	0	18	27
11	18	10	21	0	24	31
3	20	14	4	0	0	0
0	0	0	0	0	0	0

- How many clusters here?
- 2

- Forming clusters from adjacent groups of dense cells is straightforward
- We need to define “adjacent”
  - E.g. does a cell on a 2-dim grid have 4 or 8 adjacent cells?
- Cells on the boundaries are likely to be not dense enough, so they will be discarded and parts of the cluster will be lost
  - E.g. if the density threshold is 9, 4 parts of the big cluster will be lost
- We can modify the algorithm to address these issues

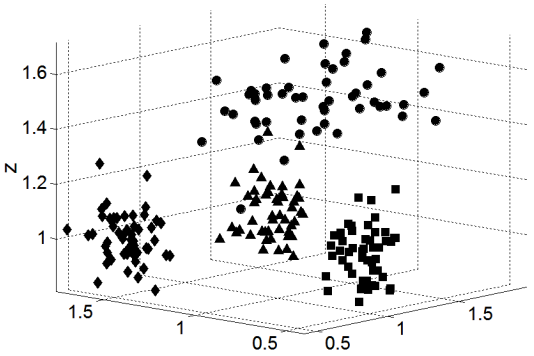
Grid-based clustering – strength and limitations

- Strengths – efficient algorithm:
  - Given a partitioning of each attribute, a single pass through the data can determine the grid cell of every object and the count of every cell
  - Time and space complexity of defining the grid, assigning each object to a cell and computing the density of each sell: O(*m*), *m* – number of points
  - If adjacent cells can be efficiently accessed using a search tree, the entire clustering process can be highly efficient – time complexity: O(*m* log *m*)
- Limitations
  - Dependent on the grid partitioning
  - Dependent on the choice of the density threshold *t*
    - if *t* is too high – clusters will be lost
    - if *t* is too low – 2 clusters that should be separate, may be joined
  - Does not perform well with clusters of different densities and noise – not possible to find a single *t* that works well for all parts of the data space
  - Problems with cells at the boundaries – they may be ignored (insufficient density)

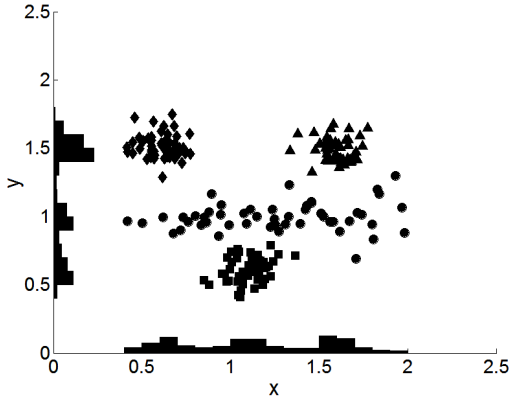
Grid-based, subspace clustering

- The clustering algorithms we studied so far find clusters by using all attributes
- However, the data may cluster well using only some of the attributes and may be randomly distributed using the remaining attributes
- Also, different clusters may exist in different dimensions (subsets of attributes)

- Original 3-dim space: 3 clusters (diamonds, triangles and squares); the circles don't look like a cluster

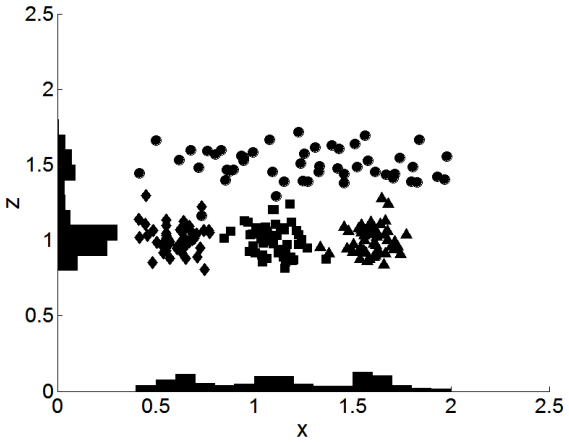


- 2-dim subspace xy (z is ignored) – again 3 clusters
- Shows also the histograms (point distribution with respect to x and y)
- Consider only 1-dim subspace: x - 3 clusters, y - 3 clusters

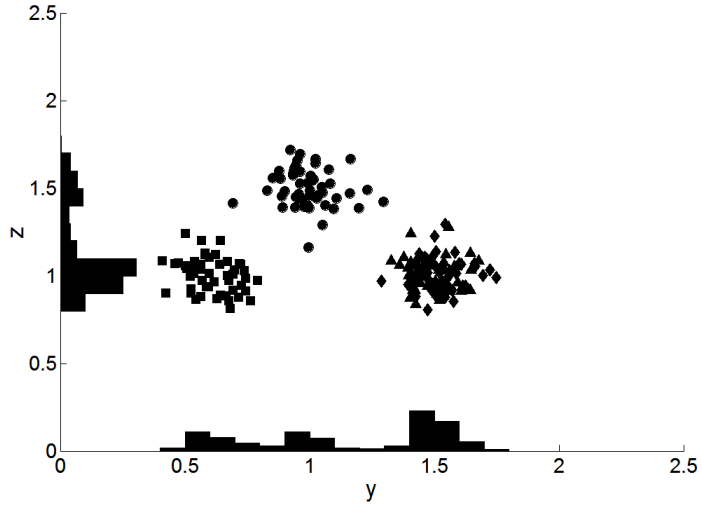


Clusters in subspaces

- Different number of clusters in the 2-dim and 1-dim subspaces
  - 2-dim: xz



  - 2-dim: yz



CLIQUE algorithm

- CLUQUE (**CL**ustering **I**n **QUE**st) – dimension-growth, subspace grid-based clustering
  - The clustering starts in 1-dimensional subspaces and grows to the higher-dimensional subspaces
  - It finds the subspace of the highest dimensionality where high-density clusters exist
  - As in the basic grid-based clustering, a cluster is a set of contiguous (touching) dense cells
  - At each subspace, the data is partitioned into rectangular cells and the dense cells are identified based on a threshold
  - To efficiently find the dense cells at k-dim subspace, it uses the Apriori property from association rule mining and only considers the dense cells from the previous lower dimensional subspace (k-1). There is no need to consider the non-dense cells at (k-1)-dim; this reduces the search space.

Apriori (monotonicity) principle

- Apriori principle (alco called monotonicity principle)
  - If a set of points forms a density-based cluster in k-dim space, then the same set of points is also part of a density-based cluster in all possible subsets of those dimensions

- =>If a k-dim cell is dense, then its projections in (k-1)-dim space are also dense
- Hence, if a (k-1)-dim cell is not dense, then its k-dim cell is also not dense
- Hence, to generate candidate dense cells in k–dim space we only need to consider the dense cells in (k-1)-dim space

CLIQUE algorithm - pseudocode

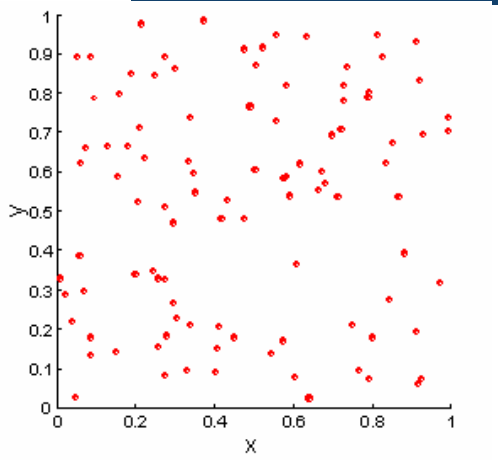
- Provides an efficient technique for searching for clusters in subspaces
- As with all grid-based clustering algorithms:
  - The quality of the results depends on the choice of parameters – number and width of the grid cells and density threshold
  - Does not work well for clusters with widely differing densities, since the threshold is fixed

Algorithm 9.5 CLIQUE.

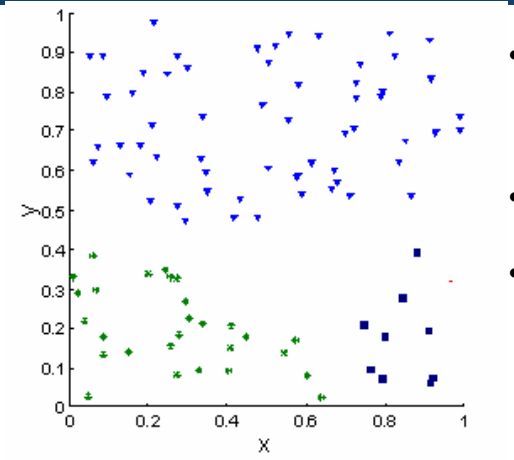
1: Find all the dense areas in the one-dimensional spaces corresponding to each attribute. This is the set of dense one-dimensional cells.
2: $k \leftarrow 2$
3: **repeat**
4: Generate all candidate dense k-dimensional cells from dense $(k-1)$-dimensional cells.
5: Eliminate cells that have fewer than ξ points.
6: $k \leftarrow k + 1$
7: **until** There are no candidate dense k-dimensional cells.
8: Find clusters by taking the union of all adjacent, high-density cells.
9: Summarize each cluster using a small set of inequalities that describe the attribute ranges of the cells in the cluster.

Evaluating clustering results

Clusters found in random data

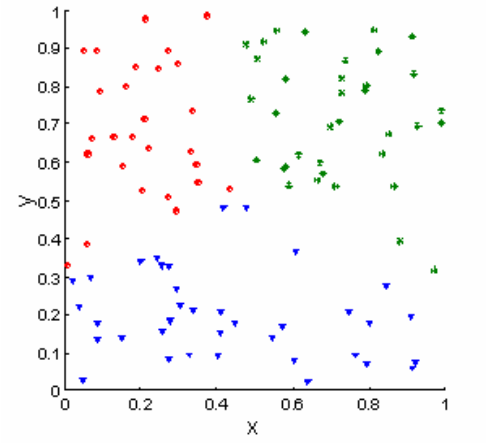


Random points

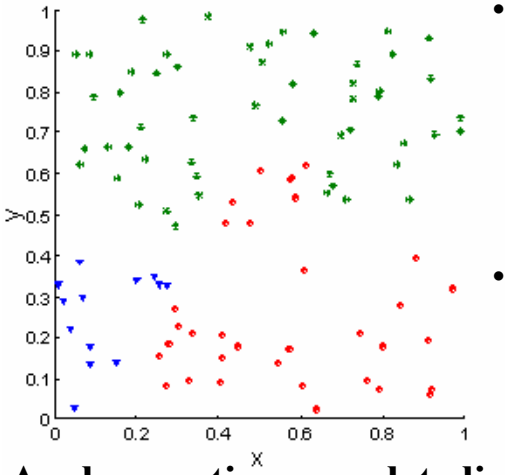


DBSCAN

- 1000 data points, randomly generated
- DBSCAN found 3 clusters
- We set K-means and agglomerative also to find 3 clusters



K-means



Agglomerative complete link

- All clustering algorithms will find clusters even if data is random and has no natural clusters
- We need to evaluate the resulting clusters as we don't want to find patterns in noise

How good is our clustering?

- 1) Evaluating clustering quality using unsupervised measures
  - High similarity within a cluster, low similarity between clusters
  - These measures are also called internal
- 2) Evaluating clustering quality using supervised measures
  - Comparing the clustering results with "ground truth" (correct cluster labels provided by an expert
  - These measures are also called external

Related topics:

- 3) Determining the correct number of clusters
- 4) Determining the clustering tendency of the data – is there a non-random structure in the data?

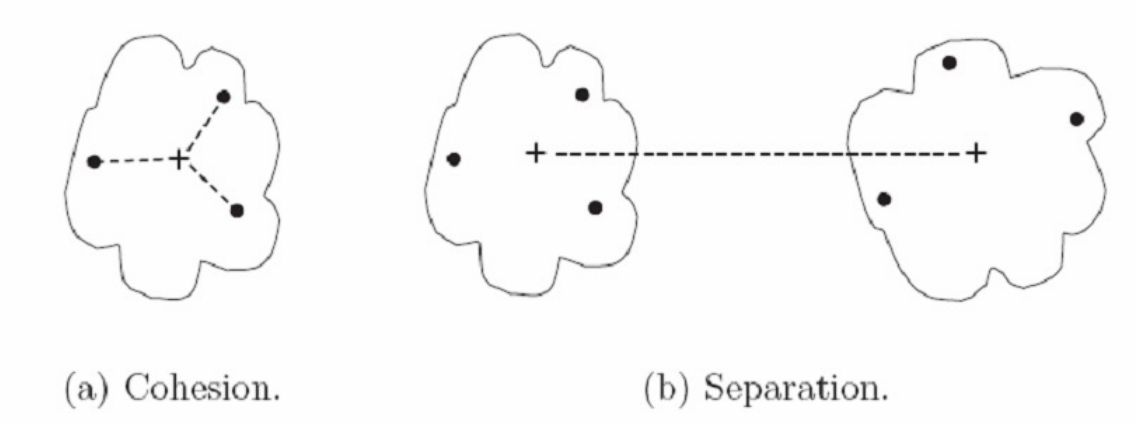
1) Evaluating clustering quality using unsupervised measures

- Method 1: Measuring cohesion and separation
  - Cohesion and separation alone
  - Combining cohesion and separation: Silhouette coefficient
- Method 2: Correlation between 2 similarity matrices
  - matrix 1: derived from the distance matrix
  - matrix 2: derived from the clustering results
  - Are the similar items in the same cluster?
- Method 3: Visual inspection of sorted similarity matrix

Unsupervised method 1: Cohesion and separation

Good clustering

- A good clustering produces clusters with
  - high cohesion = high similarity within the cluster
  - high separation = low similarity between the clusters



(a) Cohesion. (b) Separation.

- Cohesion and separation are measured using distance measures

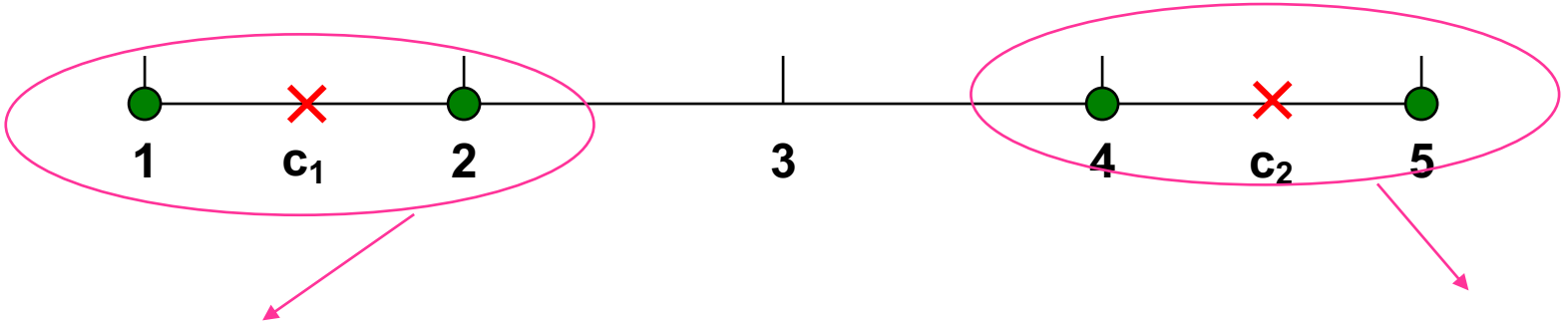
Cohesion

- Cohesion of a cluster K_i:

$$cohesion(K_i) = \sum_{x \in K_i} dist(x, c_i)$$

c_i is the cluster centroid, x are all the objects in this cluster

- Example: 2 clusters K1 and K2:



Using the Manhattan distance:
cohesion(K1) =|1-1.5|+|2-1.5|=1

cohesion(K2)=|4-4.5|+|5-4.5|=1

- Cohesion of the whole clustering - overall cohesion over all clusters K_i:

cohesion =cohesion(K1)+cohesion(K2)=2

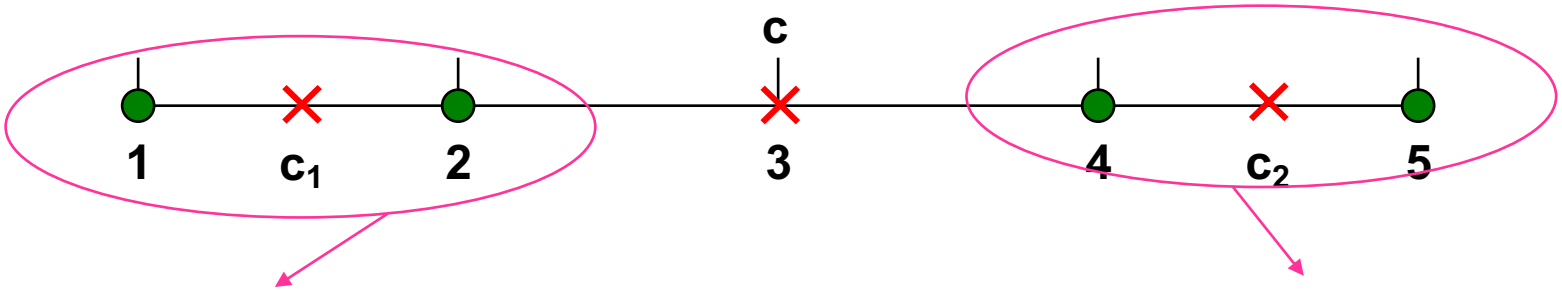
Separation

- Separation of a cluster K_i from the other clusters:

$$separation(K_i) = dist(c_i, c)$$

c is the overall centroid (all points considered)

$\mathbf{c_i}$ are the cluster centroids, i=1,2



separation(K1) =|1.5-3|=1.5

separation(K2)=|4.5-3|=1.5

- Separation of the clustering - overall separation weighted by the size of each cluster $|K_i|$:

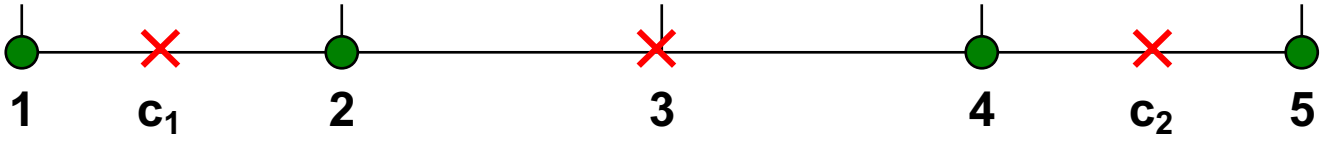
$$separation = \sum_{i=1}^{k} |K_i| dist(c_i, c)$$

- Sum of the separations of the individual clusters, weighed by their size

separation =2*|1.5-3|+2*|4.5-3|=6

Cohesion and separation using squared distance

- We can use squared distances to express cohesion and separation:
  - SSE = distance within a cluster $$SSE = \sum_{i=1}^{k} \sum_{\mathbf{x} \in K_i} (c_i, \mathbf{x})^2$$
  - BSE = distance between clusters $$BSE = \sum_{i=1}^{k} |K_i| (c_i, c)^2$$



k=2 clusters

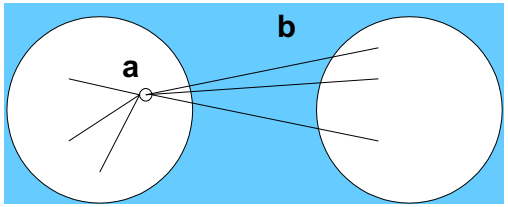
({1,2}, {3,4})

SSE = (1-1.5)^2 + (2-1.5)^2 + (4-4.5)^2 + (5-4.5)^2 = 1

BSE = 2*(3-1.5)^2 + 2*(4.5-3)^2 = 9

Silhouette coefficient

- Combines cohesion and separation
- Calculated for a point, cluster and clustering



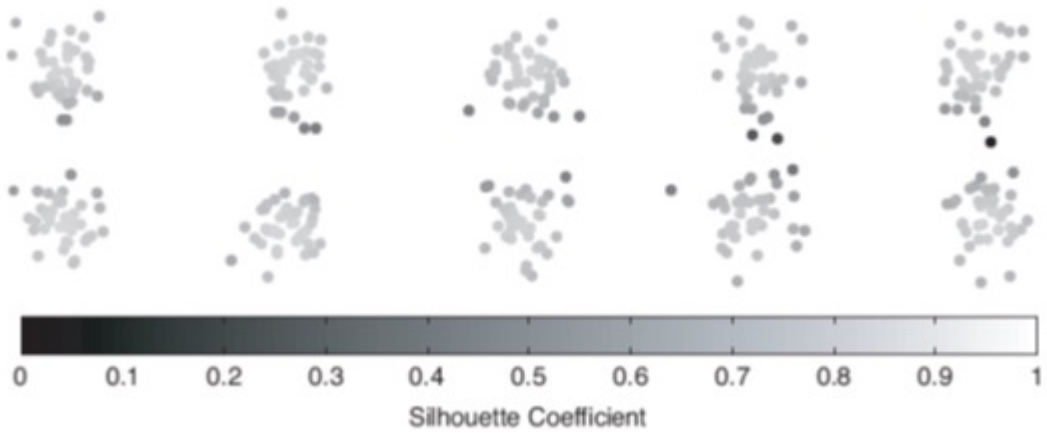
- Silhouette coefficient for a point i:
  - a_i = the average distance from i to all points in its cluster (cohesion)
  - b_i = the average distance from i to all points in another cluster (separation). Find the minimum value with respect to all clusters. (minimum average distance to points in another cluster)
  - Silhouette coefficient $s_i = (b_i-a_i)/max(a_i,b_i)$

- Values between -1 and 1; the higher the better
  - max value=1 when $a_i=0$ and $b_i>a_i$ ($s_i=b_i/b_i$, high cohesion & high separation)
  - negative values: $b_i<a_i$ – bad quality clustering (separation<cohesion)

Silhouette coefficient (2)

- Silhouette coefficient for a cluster
  - average of the Silhouette coefficients for all points in the cluster

- Silhouette coefficient for a clustering
  - average of the Silhouette coefficients for all clusters



- Example: Silhouette coefficient calculated for all points in the 10 clusters
- Darker color = lower Silhouette coefficient

Unsupervised method 2: Correlation between two similarity matrices

Correlation between similarity matrices

- Finds if the similar items (the items which are close to each other) are in the same cluster
- Example:
  - 4 items were clustered into 2 clusters {P1, P2} and {P3, P4}
  - Similarity matrix:

	P1	P2	P3	P4
P1	1	0.8	0.65	0.55
P2		1	0.7	0.6
P3			1	0.9
P4				1

- Task: Evaluate the clustering quality using correlation
- Idea: Compute the correlation between the given similarity matrix and the similarity matrix defined by the clustering (value =0, if 2 items are not in the same cluster, value =1, if they are in the same cluster)

	P1	P2	P3	P4
P1	1	1	0	0
P2		1	0	0
P3			1	1
P4				1

- High correlation: good clustering (similar items are in the same cluster)
- Low correlation: poor clustering (similar items are not in the same cluster)

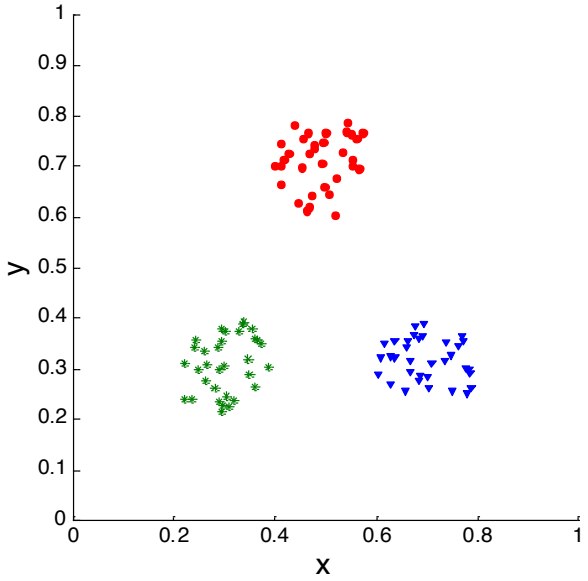
Correlation method - summary

- Finds the correlation between 2 similarity matrices
  - matrix1 computed from the distance matrix
  - matrix2 computed from the clustering results
- The higher the correlation, the better the quality of the clustering
- Matrix1 - computed from the distance matrix
  - different ways to define similarity; in general the higher the distance, the smaller the similarity, e.g.:
  - sim=1 - (d - d_min)/(d_max - d_min)
- Matrix2 – based on the cluster labels obtained by the clustering
  - values of 0 and 1 only
  - ij entry = 0 if items i and j are from different clusters
  - ij entry = 1 if items i and j are from the same cluster
- Revision: correlation measures a linear relationship between numeric attributes, see lecture 1b; range = [-1, 1], -1: perfect negative correlation, +1: perfect positive, 0- no correlation

Evaluating Clustering Quality Based on Correlation – Example

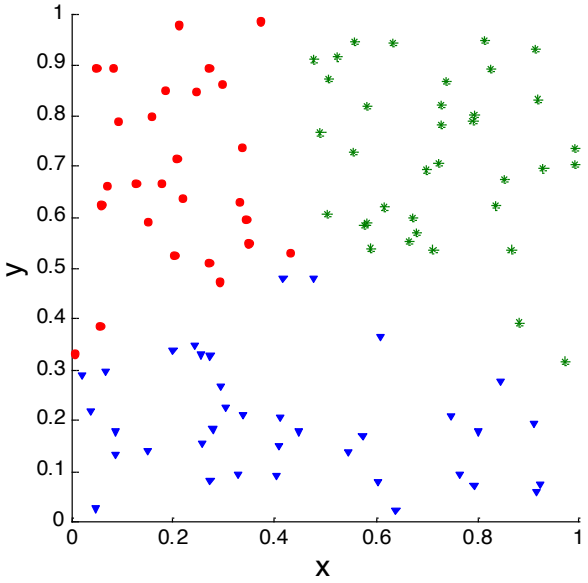
- Example for K-means clustering

data with 3 well separated clusters



Corr = -0.9235

random data – not well separated clusters

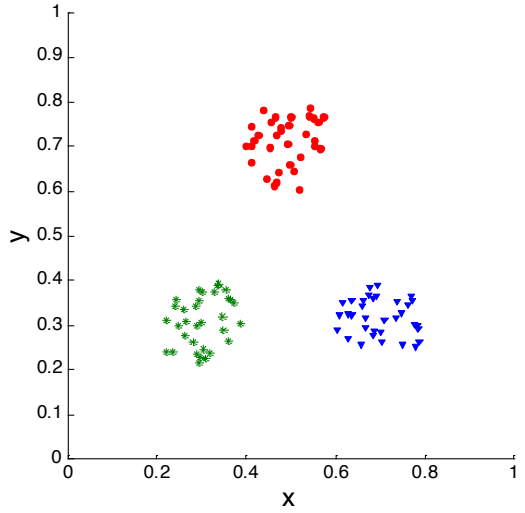


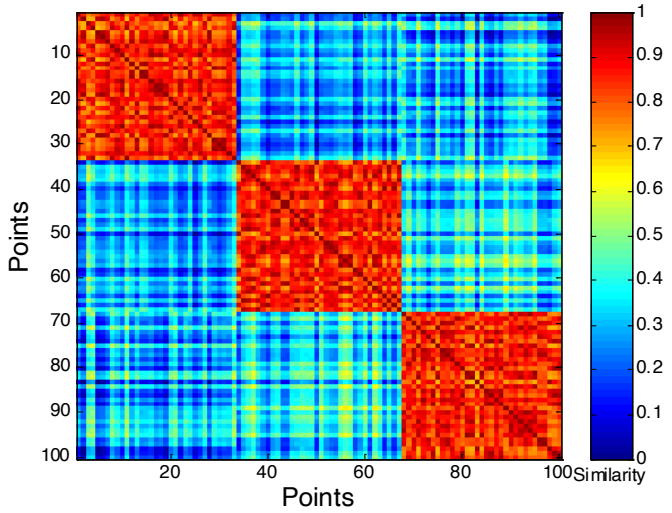
Corr = -0.5810

Unsupervised method 3: Visual inspection of similarity matrix

Visual inspection of the similarity matrix

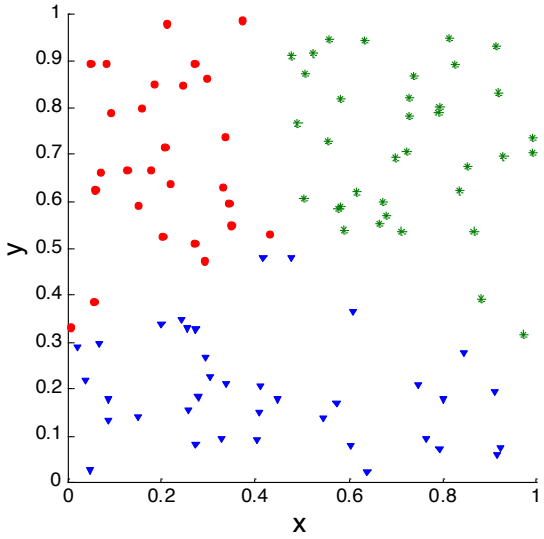
- Order the points based on their cluster
  - Points from cluster 1, points from cluster 2, etc.
- Plot the similarity matrix using coloring based on the similarity
- Well defined blocks along the main diagonal indicate good clustering
  - I.e. items from the same cluster are similar to each other

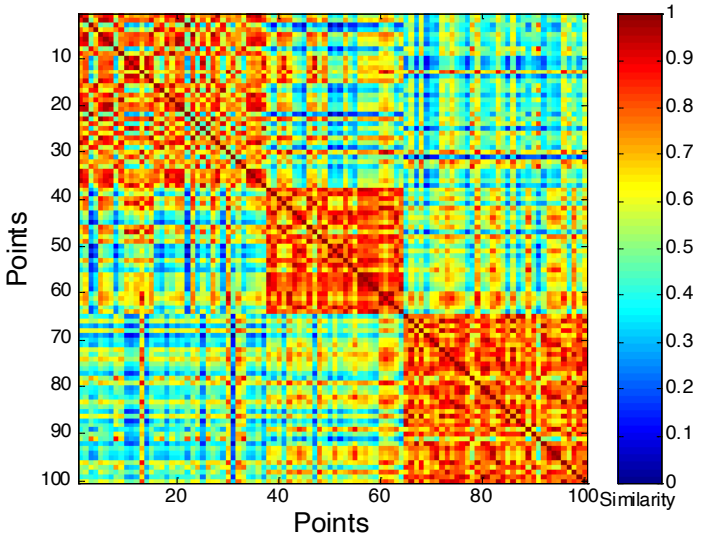




Example: K-means

- Clustering random data: weak block diagonal patterns

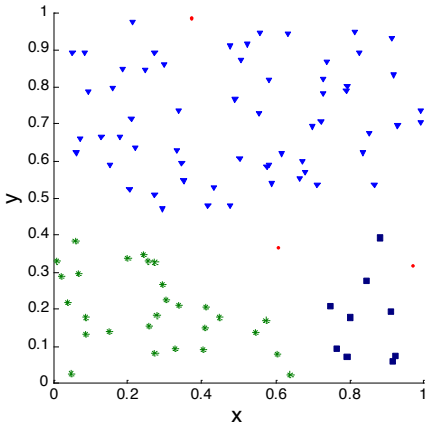


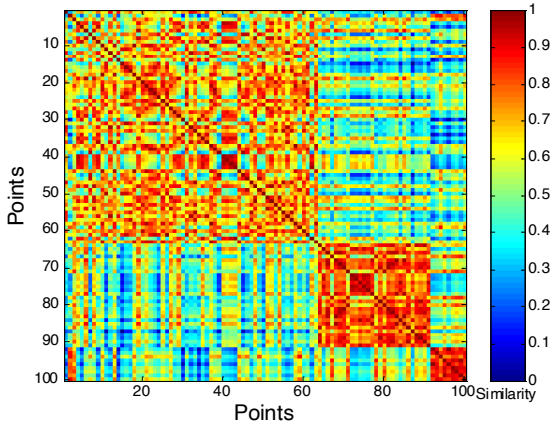


K-means

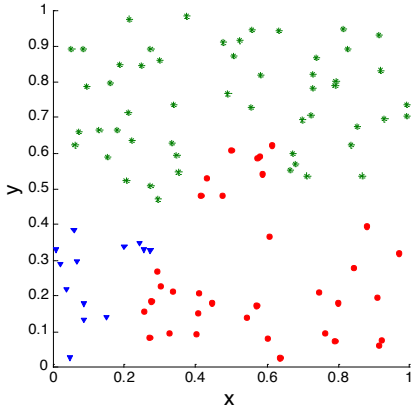
Example: DBSCAN and agglomerative

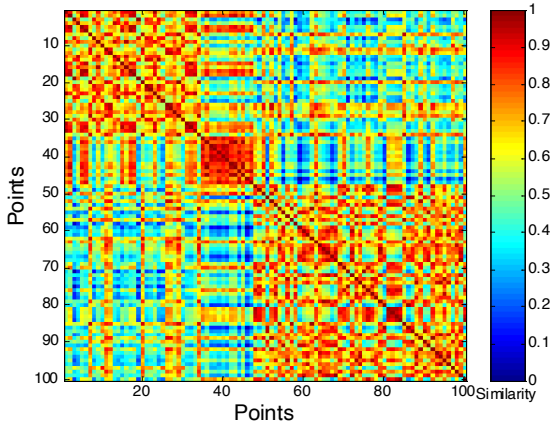
- The same: weak block diagonal patterns when clustering random data





DBSCAN





Agglomerative complete link

2) Evaluating clustering quality using supervised measures

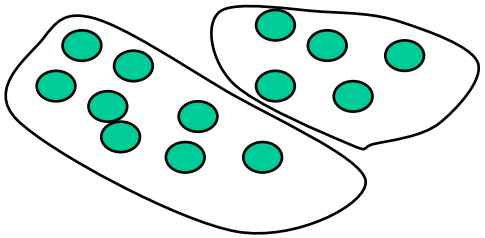
- Classification oriented measures
  - Entropy, purity
  - Accuracy, precision, recall, F1 measure
- Similarity oriented
  - Correlation between the obtained and true similarity matrices

2) Evaluating clustering quality using supervised measures

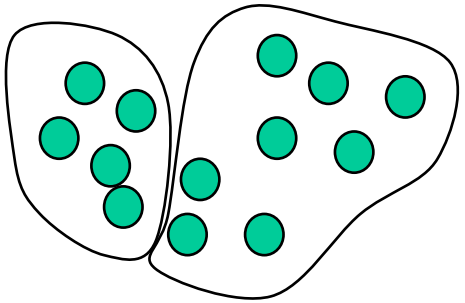
- Supervised – an expert provides the true labels of the items (their correct cluster); called ground truth
- Idea: measure the difference between the true labels and the ones obtained by the clustering algorithm
  - Small difference: good clustering, big difference: bad clustering
- Example:

1) Clustering result:

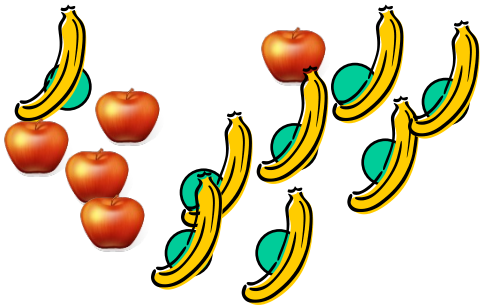
A



B



2) Now the expert has provided the true labels – apples and bananas:

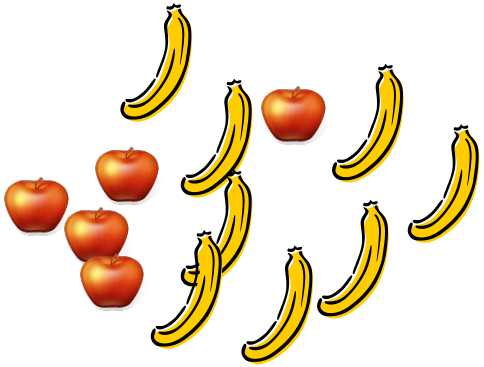


3) Which clustering is better: A or B?

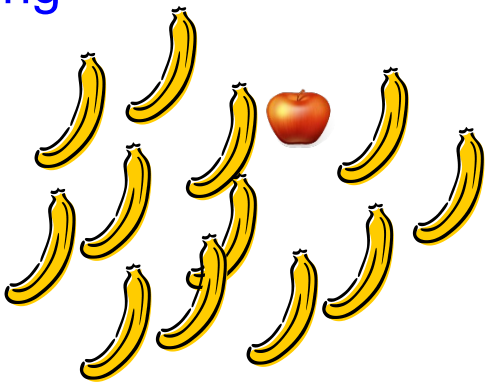
Supervised Method 1: Entropy and purity

Entropy and Purity for Clustering Quality Evaluation

- Two complementary measures – high entropy means low purity and vice versa
- Measure the homogeneity of a set of examples with respect to their class label
- We use them to measure the homogeneity of each cluster, and based on this the homogeneity of the overall clustering



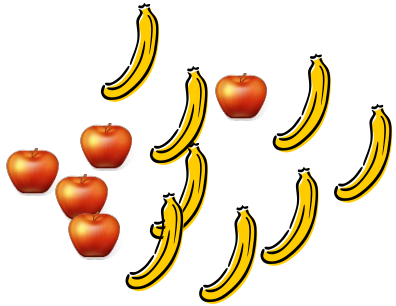
A cluster with 5 apples and 8 bananas – high entropy and low purity



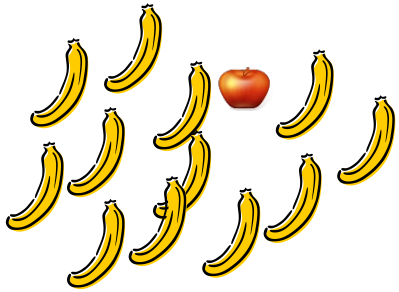
A cluster with 1 apple and 12 bananas – low entropy and high purity

Entropy

- Entropy of a cluster K: $entropy(K) = -\sum_i P_i . \log_2 P_i$
  - i – index over the class labels (ground truth), e.g. banana and apples
  - P_i - proportion of examples that belong to class I, base of log is typically 2
- The smaller the entropy, the greater the purity of the cluster
- Revision: see lecture 5a



$$entropy(K) = -\frac{5}{13}\log\frac{5}{13} - \frac{8}{13}\log\frac{8}{13} = high$$



$$entropy(K) = -\frac{1}{13}\log\frac{1}{13} - \frac{12}{13}\log\frac{12}{13} = low$$

- Entropy of the clustering = sum of the entropies of the individual clusters weighted by the size of each cluster

$$overall\_entropy = \sum_{i=1}^{k} \frac{m_i}{m} entropy(K_i)$$

Num. of examples in each cluster/total number of examples

Purity

- Similar to entropy; also measures homogeneity of a cluster K (set of examples) with respect to their class label

$$purity(K) = \max_i Pi$$ compare with $$entropy(K) = -\sum_i P_i . \log_2 P_i$$

- Purity of the clustering

$$overall\_purity = \sum_{i=1}^{k} \frac{m_i}{m} purity(K_i)$$

- The higher the purity, the better the clustering

Cluster evaluation using entropy and purity - steps

1. Cluster the data
2. Label each point with its correct class label
3. Calculate entropy and purity – for each cluster and then the overall
4. If most of the items in a cluster are from the same class, entropy will be low and purity will be high, i.e. good clustering

Entropy and purity for evaluating clustering quality - example

- 3204 newspaper articles; K-means with the cosine similarity measure was used to cluster them; number of clusters: k=6
- The articles belong to 6 classes: Entertainment, Financial, Foreign, Metro, National and Sports (ground truth, not used in clustering but we will use it for the evaluation of the quality of clustering)
- Clustering results – how many papers from each class were in each cluster:

Cluster	Enter-tainment	Financial	Foreign	Metro	National	Sports
1	3	5	40	506	96	27
2	4	7	280	29	39	2
3	1	1	1	7	4	671
4	10	162	3	119	73	2
5	331	22	5	70	13	23
6	5	358	12	212	48	13
Total	354	555	341	943	273	738

- Task: Use entropy and purity to evaluate the quality of the clustering

Solution

- Entropy and purity for cluster K1
  - Total number of news articles in K1: 677

$$entropy(K_1) = -\frac{3}{677}\log\frac{3}{677} - \frac{5}{677}\log\frac{5}{677} - \frac{40}{677}\log\frac{40}{677} - \frac{506}{677}\log\frac{506}{677} - \frac{96}{677}\log\frac{96}{677} - \frac{27}{677}\log\frac{27}{677} = 1.227$$

$$purity(K_1) = \max\left(\frac{3}{677}, \frac{5}{677}, \frac{40}{677}, \frac{506}{677}, \frac{96}{677}, \frac{27}{677}\right) = 0.7474$$

- Entropy and purity for the whole clustering:

$$overall\_entropy = \frac{677}{3204}1.227 + ... = 1.145$$

$$overall\_purity = \frac{677}{3204}0.7474 + ... = 0.7203$$

Cluster	Entertainment	Financial	Foreign	Metro	National	Sports
1	3	5	40	506	96	27
2	4	7	280	29	39	2
3	1	1	1	7	4	671
4	10	162	3	119	73	2
5	331	22	5	70	13	23
6	5	358	12	212	48	13
Total	354	555	341	943	273	738

Is this a good clustering?

Supervised Method 2: Correlation between obtained and ideal similarity matrix

Correlation between cluster similarity matrices

- Calculate the correlation between the obtained cluster similarity matrix and the ideal cluster similarity matrix
- The ideal similarity matrix is based on the labels provided by the expert
- Binary matrices: ij entry =1 if items i and j belong to the same class, 0 otherwise

- As these matrices are binary, instead of calculating the correlation, we can also calculate other measures, e.g. the Simple Matching Coefficient (SMC) (also called Rand statistic) or the Jaccard coefficient – see lecture week 1b:
- SMC (Rand statistic) = (f11+f00)/(f01+f10+f11+f00)
- Jaccard = f11/(f01+f10+f11)

Example

- Given: 5 points p1,..,p5 were clustered into 2 clusters by a clustering algorithm: K1={p1,p2,p3}, K2={p4,p5}. The ground truth labels provided by an expert are L1={p1,p2}, L2={p3,p4,p5}.
- Task: Evaluate the quality of the clustering using correlation
- Correlation between binary vectors can be calculated using the Phi coefficient (https://en.wikipedia.org/wiki/Phi_coefficient)
  - phi=f11*f00-f10*f01/sqrt[(f11+f10)*(f01+f00)*(f00+f10)*(f01+f11)]
- Solution: phi([1,1,0,0,1,0,0,0,0,1],[1,0,0,0,0,0,0,1,1,1])=
  = (2*4-2*2)/sqrt(4*6*6*4)=4/24=1/6=0.17

Obtained

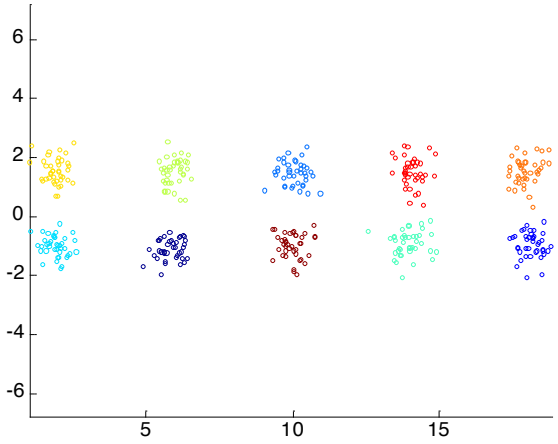
Point	p1	p2	p3	p4	p5
p1	1	1	1	0	0
p2	1	1	1	0	0
p3	1	1	1	0	0
p4	0	0	0	1	1
p5	0	0	0	1	1

Ideal

Point	p1	p2	p3	p4	p5
p1	1	1	0	0	0
p2	1	1	0	0	0
p3	0	0	1	1	1
p4	0	0	1	1	1
p5	0	0	1	1	1

3) Determining the Number of Clusters

- How to determine a good number of clusters?



- Elbow method
  - Run the clustering algorithm for several k, plot SSE or other unsupervised measure vs number of clusters
  - Look for distinct knee (drop) or peak = good number of clusters

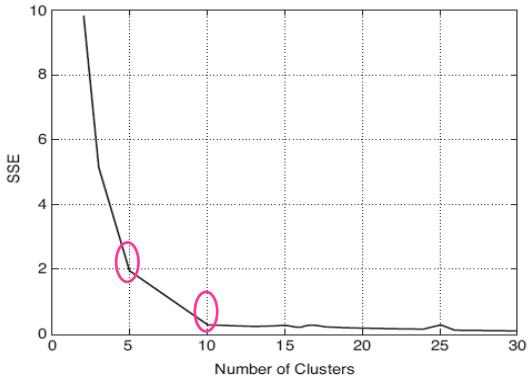


Figure 8.32. SSE versus number of clusters for the data of Figure 8.29.

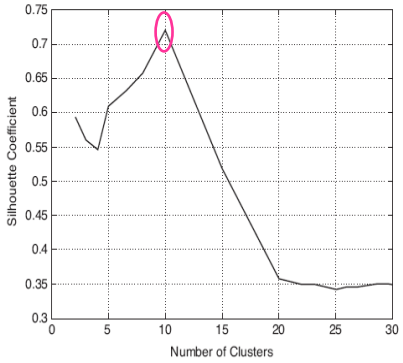


Figure 8.33. Average silhouette coefficient versus number of clusters for the data of Figure 8.29.

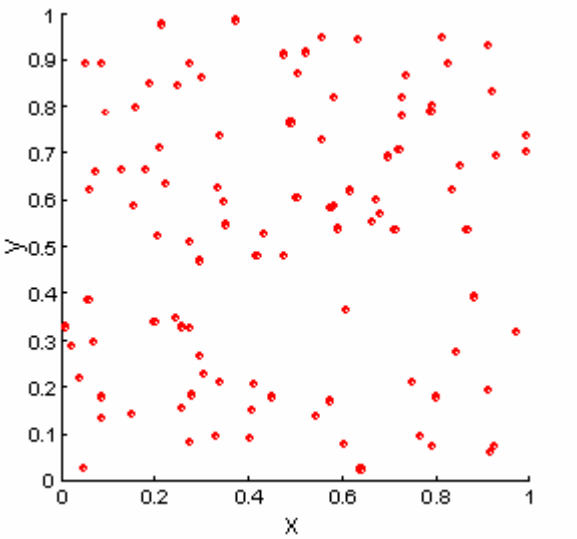
- See also the tutorial exercises!

5) Clustering Tendency

- Does a dataset have clusters?
- Method 1: cluster the data, evaluate the result using some clustering quality measures
- Method 2: measure clustering tendency before clustering
  - Is the data random or not? (spatially random)

Clustering Tendency Using Hopkins Statistic

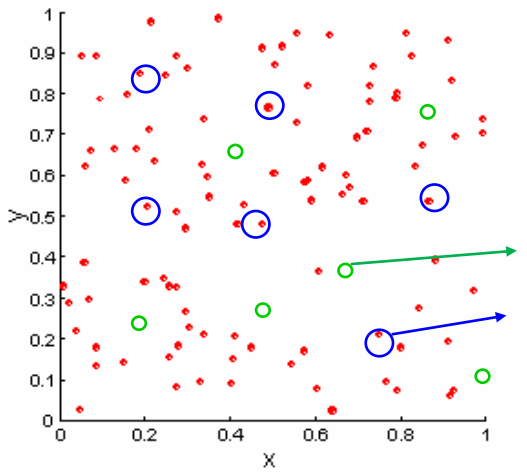
- Given: a dataset of *m* points
- Task: estimate the clustering tendency (spatial randomness) – is the data randomly distributed or not?



Hopkins statistic - procedure

- Generate 2 sets of *p* points (*p*<<*m*, heuristic: *p*=0.1**m*):
  - Set 1: *p* random points
  - Set 2: *p* actual points from the dataset - sample randomly *p* points of the given *m* points
- For each point in both sets, find the closest point in the original dataset, i.e. find the nearest neighbor point
  - Let the distances to the nearest neighbor points are u_i for set 1 (newly generated points) and w_i for set 2 (original points)
- Hopkins statistic H:

$$H = \frac{\sum_{i=1}^{p} u_i}{\sum_{i=1}^{p} u_i + \sum_{i=1}^{p} w_i}$$

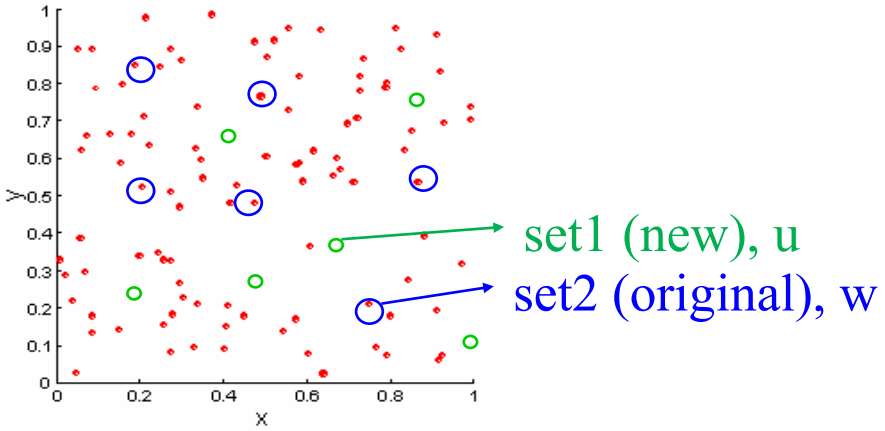


- Note that there is a mistake in the textbook (Tan, p.589) - w_i is in the numerator, it should be u_i

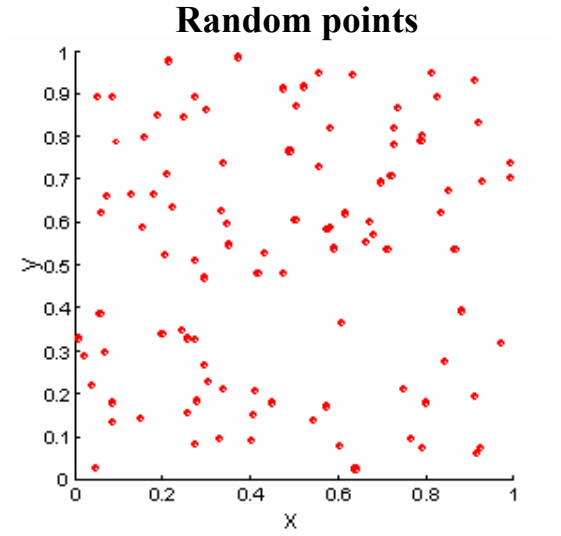
Hopkins statistic - interpretation

- The Hopkins statistic H compares nearest neighbors of a sample of: (1) original points (randomly selected) and (2) random points
- Interpretation of the value:
  - Near 1: data is highly clustered as u is big and w is small. Neighbors of random points are farther (big u) than neighbors of actual points (small w).
  - Near 0.5: set1 and set2 have roughly the same nearest neighbor distances => the original data is similar to the random data => the given dataset is spatially random, i.e. it doesn't cluster well
  - Near 0 - data is neither clustered nor random, it is evenly spaced

$$H = \frac{\sum_{i=1}^{p} u_i}{\sum_{i=1}^{p} u_i + \sum_{i=1}^{p} w_i}$$

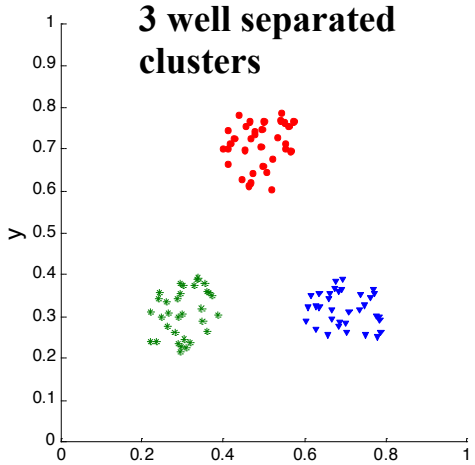


Agglomerative hierarchical clustering



- H computed and averaged over 100 runs
- Results: mean value +/- standard deviation

- H=0.56 +/- 0.03 – near 0.5 => spatially random data



H=0.95 +/- 0.006 – near 1 => highly clustered data

Clustering evaluation - summary

- How good is our clustering?
- Unsupervised evaluation
  - Cohesion and separation alone or combined (e.g. Silhouette coefficient)
  - Correlation between 2 similarity matrices - 1) derived from the distance matrix and 2) from the clustering results
  - Visual inspection of similarity matrix (sorted)
- Supervised
  - Entropy, purity (+ recall, precision, F1, accuracy)
  - Correlation between obtained and actual cluster matrices (ground truth and clustering results)
- Determining the number of clusters – Elbow method
- Determining the clustering tendency - Hopkins statistic