DATA MINING LECTURE 8

The EM Algorithm

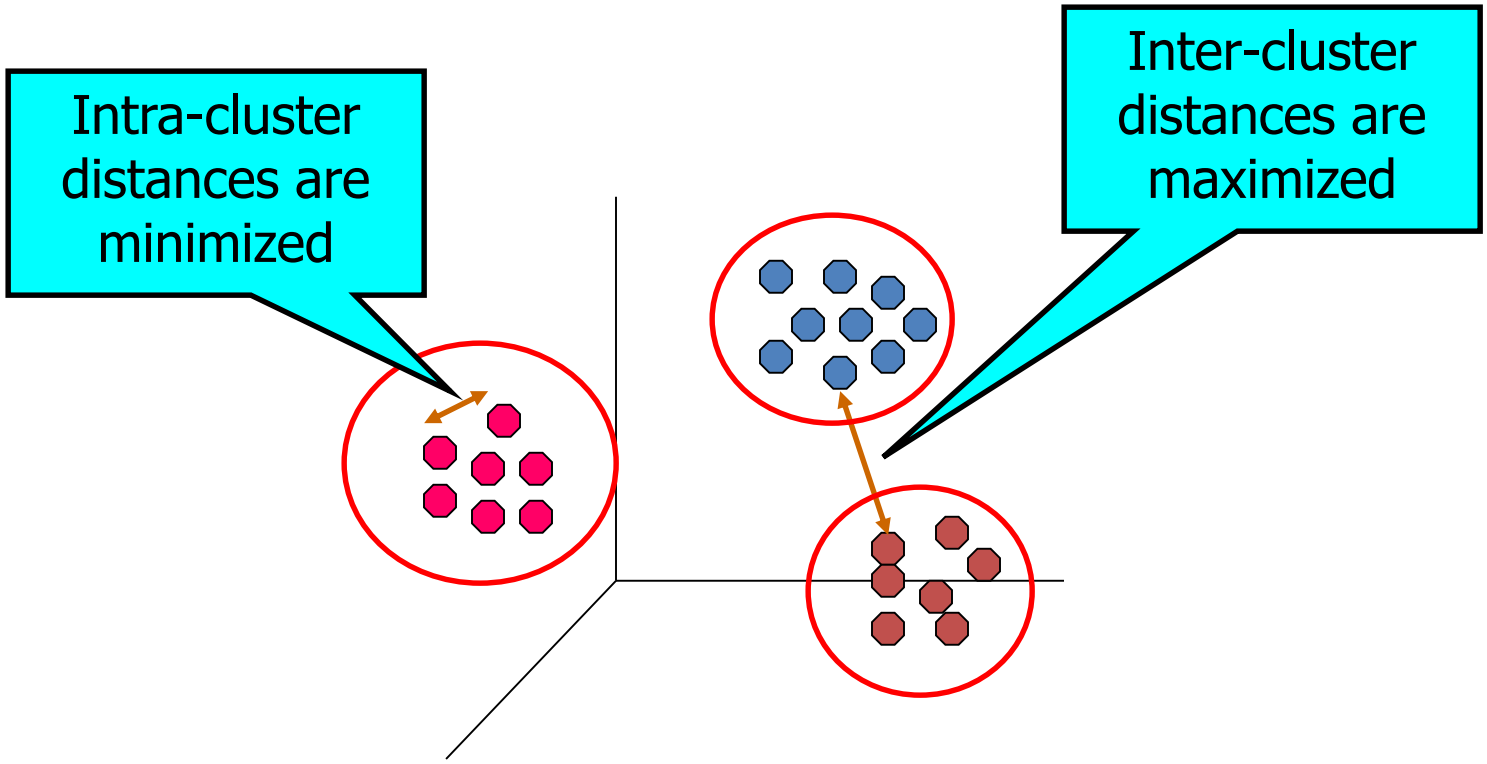
Clustering Validation

Sequence segmentation

CLUSTERING

What is a Clustering?

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



Clustering Algorithms

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

MIXTURE MODELS AND THE EM ALGORITHM

Model-based clustering

- In order to understand our data, we will assume that there is a generative process (a model) that creates/describes the data, and we will try to find the model that best fits the data.
  - Models of different complexity can be defined, but we will assume that our model is a distribution from which data points are sampled
  - Example: the data is the height of all people in Greece
- In most cases, a single distribution is not good enough to describe all data points: different parts of the data follow a different distribution
  - Example: the data is the height of all people in Greece and China
  - We need a mixture model
  - Different distributions correspond to different clusters in the data.

Gaussian Distribution

- Example: the data is the height of all people in Greece
  - Experience has shown that this data follows a Gaussian (Normal) distribution
  - Reminder: Normal distribution:

$$P(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

  - μ = mean, σ = standard deviation

Gaussian Model

- What is a model?
  - A Gaussian distribution is fully defined by the mean μ and the standard deviation σ
  - We define our model as the pair of parameters $\theta = (\mu, \sigma)$
- This is a general principle: a model is defined as a vector of parameters θ

Fitting the model

- We want to find the normal distribution that best fits our data
  - Find the best values for μ and σ
  - But what does best fit mean?

Maximum Likelihood Estimation (MLE)

- Suppose that we have a vector $X = (x_1, \dots, x_n)$ of values and we want to fit a Gaussian $N(\mu, \sigma)$ model to the data
- Probability of observing point x_i:

$$P(x_i) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

- Probability of observing all points (assume independence)

$$P(X) = \prod_{i=1}^{n} P(x_i) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

- We want to find the parameters $\theta = (\mu, \sigma)$ that maximize the probability $P(X|\theta)$

Maximum Likelihood Estimation (MLE)

- The probability $P(X|\theta)$ as a function of θ is called the Likelihood function

$$L(\theta) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x_i-\mu)^2}{2\sigma^2}}$$

- It is usually easier to work with the Log-Likelihood function

$$LL(\theta) = -\sum_{i=1}^{n} \frac{(x_i-\mu)^2}{2\sigma^2} - \frac{1}{2} n \log 2\pi - n \log \sigma$$

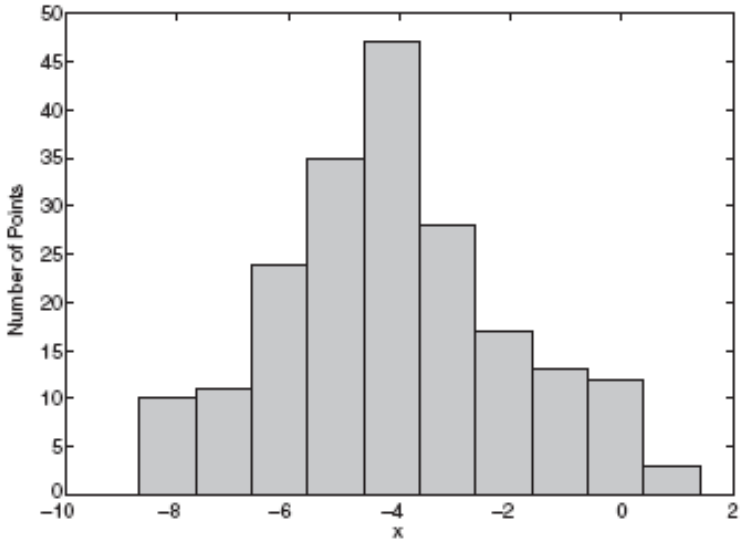
- Maximum Likelihood Estimation
  - Find parameters μ, σ that maximize $LL(\theta)$

$$\mu = \frac{1}{n}\sum_{i=1}^{n} x_i = \mu_X$$

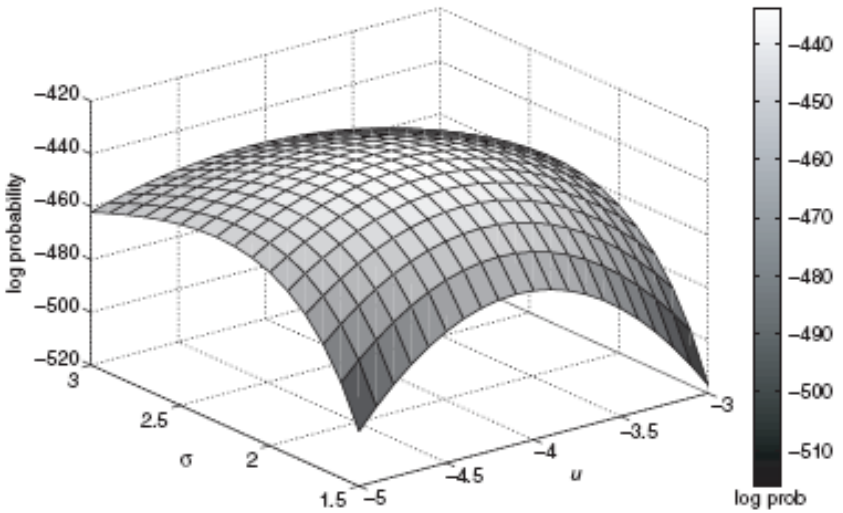
Sample Mean

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n} (x_i-\mu)^2 = \sigma_X^2$$

Sample Variance



(a) Histogram of 200 points from a Gaussian distribution.



(b) Log likelihood plot of the 200 points for different values of the mean and standard deviation.

Figure 9.3. 200 points from a Gaussian distribution and their log probability for different parameter values.

MLE

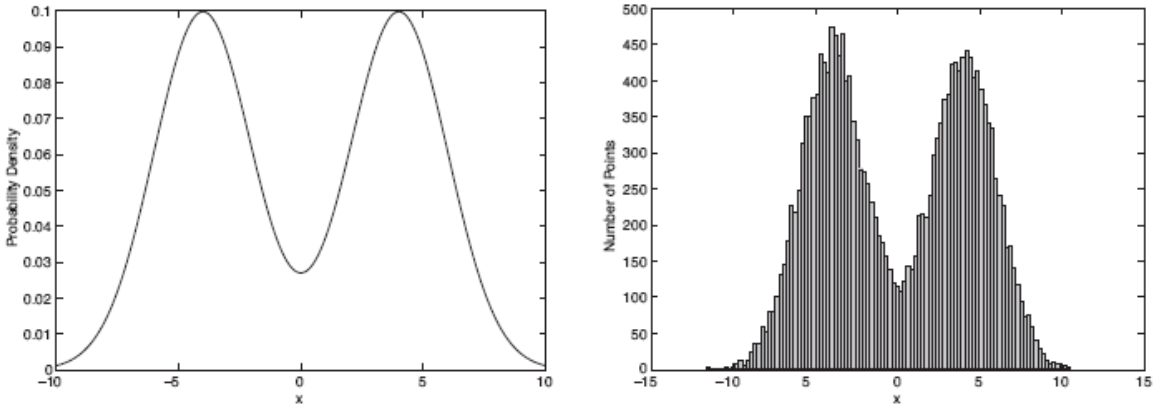
- Note: these are also the most likely parameters given the data

$$P(\theta|X) = \frac{P(X|\theta)P(\theta)}{P(X)}$$

- If we have no prior information about θ, or X, then maximizing $P(X|\theta)$ is the same as maximizing $P(\theta|X)$

Mixture of Gaussians

- Suppose that you have the heights of people from Greece and China and the distribution looks like the figure below (dramatization)



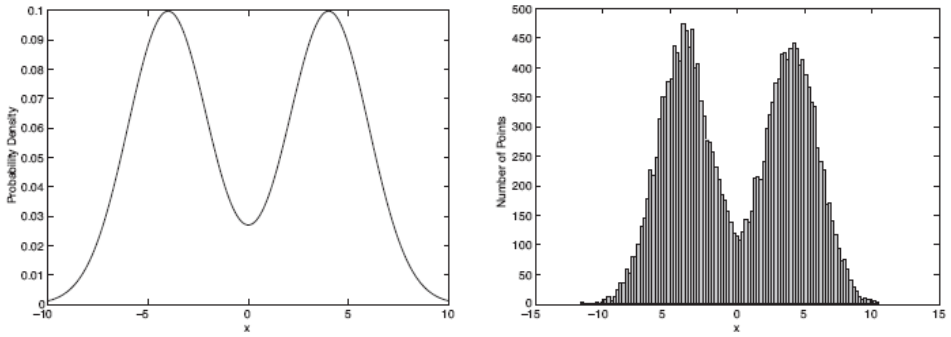
(a) Probability density function for the mixture model.

(b) 20,000 points generated from the mixture model.

Figure 9.2. Mixture model consisting of two normal distributions with means of -4 and 4, respectively. Both distributions have a standard deviation of 2.

Mixture of Gaussians

- In this case the data is the result of the mixture of two Gaussians
  - One for Greek people, and one for Chinese people
  - Identifying for each value which Gaussian is most likely to have generated it will give us a clustering.



(a) Probability density function for the mixture model.

(b) 20,000 points generated from the mixture model.

Figure 9.2. Mixture model consisting of two normal distributions with means of -4 and 4, respectively. Both distributions have a standard deviation of 2.

Mixture model

- A value x_i is generated according to the following process:
  - First select the nationality
    - With probability π_G select Greece, with probability π_C select China $(\pi_G + \pi_C = 1)$

We can also thing of this as a Hidden Variable Z that takes two values: Greece and China

  - Given the nationality, generate the point from the corresponding Gaussian
    - $P(x_i|\theta_G) \sim N(\mu_G, \sigma_G)$ if Greece
    - $P(x_i|\theta_C) \sim N(\mu_C, \sigma_C)$ if China

θ_G: parameters of the Greek distribution
θ_C: parameters of the China distribution

Mixture Model

- Our model has the following parameters

$$\Theta = (\pi_G, \pi_C, \mu_G, \sigma_G, \mu_C, \sigma_C)$$

Mixture probabilities

θ_G: parameters of the Greek distribution

θ_C: parameters of the China distribution

Mixture Model

- Our model has the following parameters

$$\Theta = (\pi_G, \pi_C, \mu_G, \sigma_G, \mu_C, \sigma_C)$$

Mixture probabilities | Distribution Parameters

- For value x_i, we have:

$$P(x_i|\Theta) = \pi_G P(x_i|\theta_G) + \pi_C P(x_i|\theta_C)$$

- For all values $X = (x_1, \ldots, x_n)$

$$P(X|\Theta) = \prod_{i=1}^{n} P(x_i|\Theta)$$

- We want to estimate the parameters that maximize the Likelihood of the data

Mixture Model

- Our model has the following parameters

$$\Theta = (\pi_G, \pi_C, \mu_G, \sigma_G, \mu_C, \sigma_C)$$

Mixture probabilities | Distribution Parameters

- For value x_i, we have:

$$P(x_i|\Theta) = \pi_G P(x_i|\theta_G) + \pi_C P(x_i|\theta_C)$$

- For all values $X = (x_1, \ldots, x_n)$

$$P(X|\Theta) = \prod_{i=1}^{n} P(x_i|\Theta)$$

- We want to estimate the parameters that maximize the Likelihood of the data

Mixture Models

- Once we have the parameters $\Theta = (\pi_G, \pi_C, \mu_G, \mu_C, \sigma_G, \sigma_C)$ we can estimate the membership probabilities $P(G|x_i)$ and $P(C|x_i)$ for each point x_i:
  - This is the probability that point x_i belongs to the Greek or the Chinese population (cluster)

Given from the Gaussian distribution $N(\mu_G, \sigma_G)$ for Greek

$$P(G|x_i) = \frac{P(x_i|G)P(G)}{P(x_i|G)P(G) + P(x_i|C)P(C)}$$

$$= \frac{P(x_i|\theta_G)\pi_G}{P(x_i|\theta_G)\pi_G + P(x_i|\theta_C)\pi_C}$$

EM (Expectation Maximization) Algorithm

- Initialize the values of the parameters in Θ to some random values
- Repeat until convergence
  - E-Step: Given the parameters Θ estimate the membership probabilities $P(G|x_i)$ and $P(C|x_i)$
  - M-Step: Compute the parameter values that (in expectation) maximize the data likelihood

$$\pi_C = \frac{1}{n}\sum_{i=1}^{n} P(C|x_i) \qquad \pi_G = \frac{1}{n}\sum_{i=1}^{n} P(G|x_i)$$

Fraction of population in G,C

$$\mu_C = \sum_{i=1}^{n} \frac{P(C|x_i)}{n * \pi_C} x_i \qquad \mu_G = \sum_{i=1}^{n} \frac{P(G|x_i)}{n * \pi_G} x_i$$

MLE Estimates if π's were fixed

$$\sigma_C^2 = \sum_{i=1}^{n} \frac{P(C|x_i)}{n * \pi_C} (x_i - \mu_C)^2 \qquad \sigma_G^2 = \sum_{i=1}^{n} \frac{P(G|x_i)}{n * \pi_G} (x_i - \mu_G)^2$$

Relationship to K-means

- E-Step: Assignment of points to clusters
  - K-means: hard assignment, EM: soft assignment
- M-Step: Computation of centroids
  - K-means assumes common fixed variance (spherical clusters)
  - EM: can change the variance for different clusters or different dimensions (ellipsoid clusters)
- If the variance is fixed then both minimize the same error function

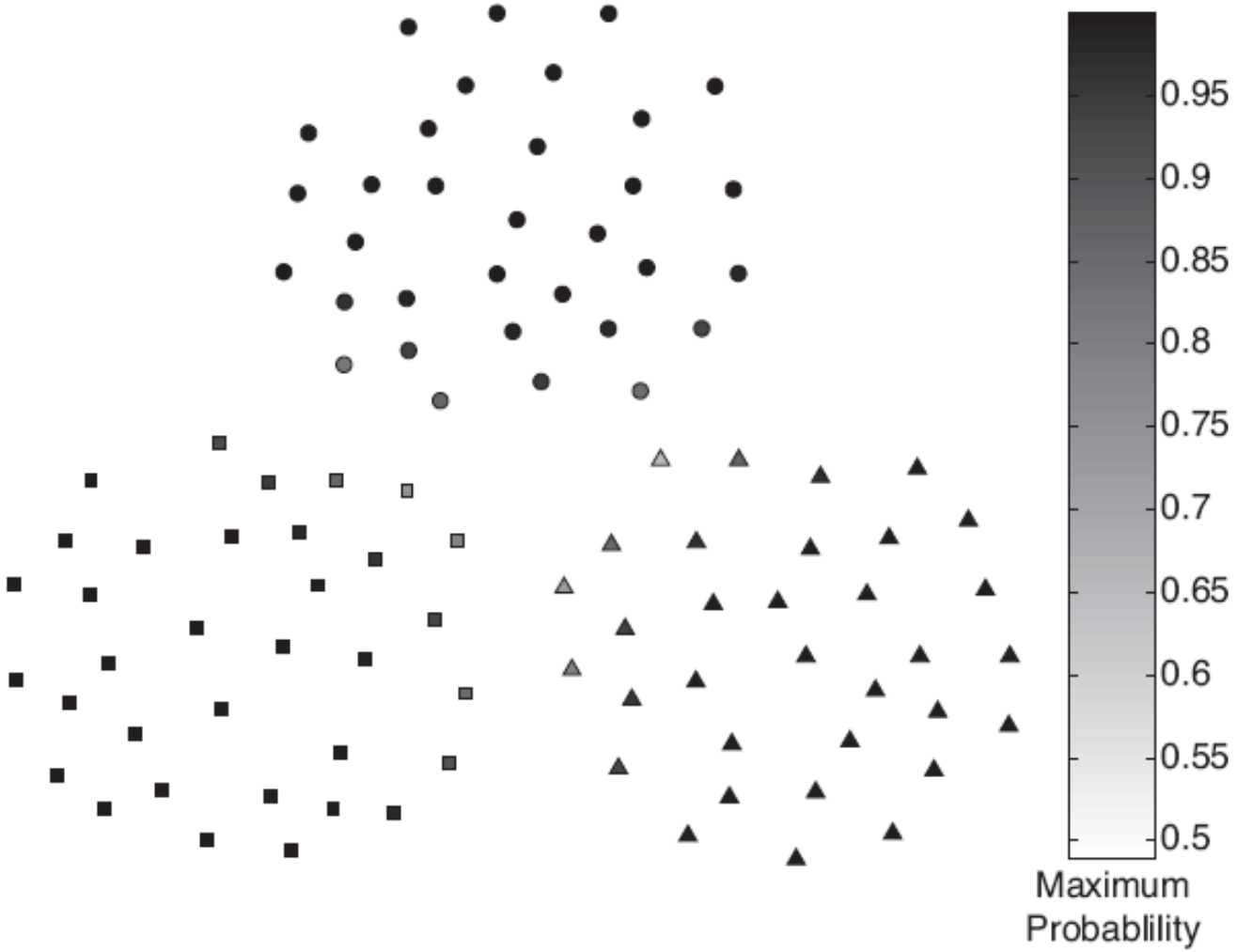


Figure 9.4. EM clustering of a two-dimensional point set with three clusters.

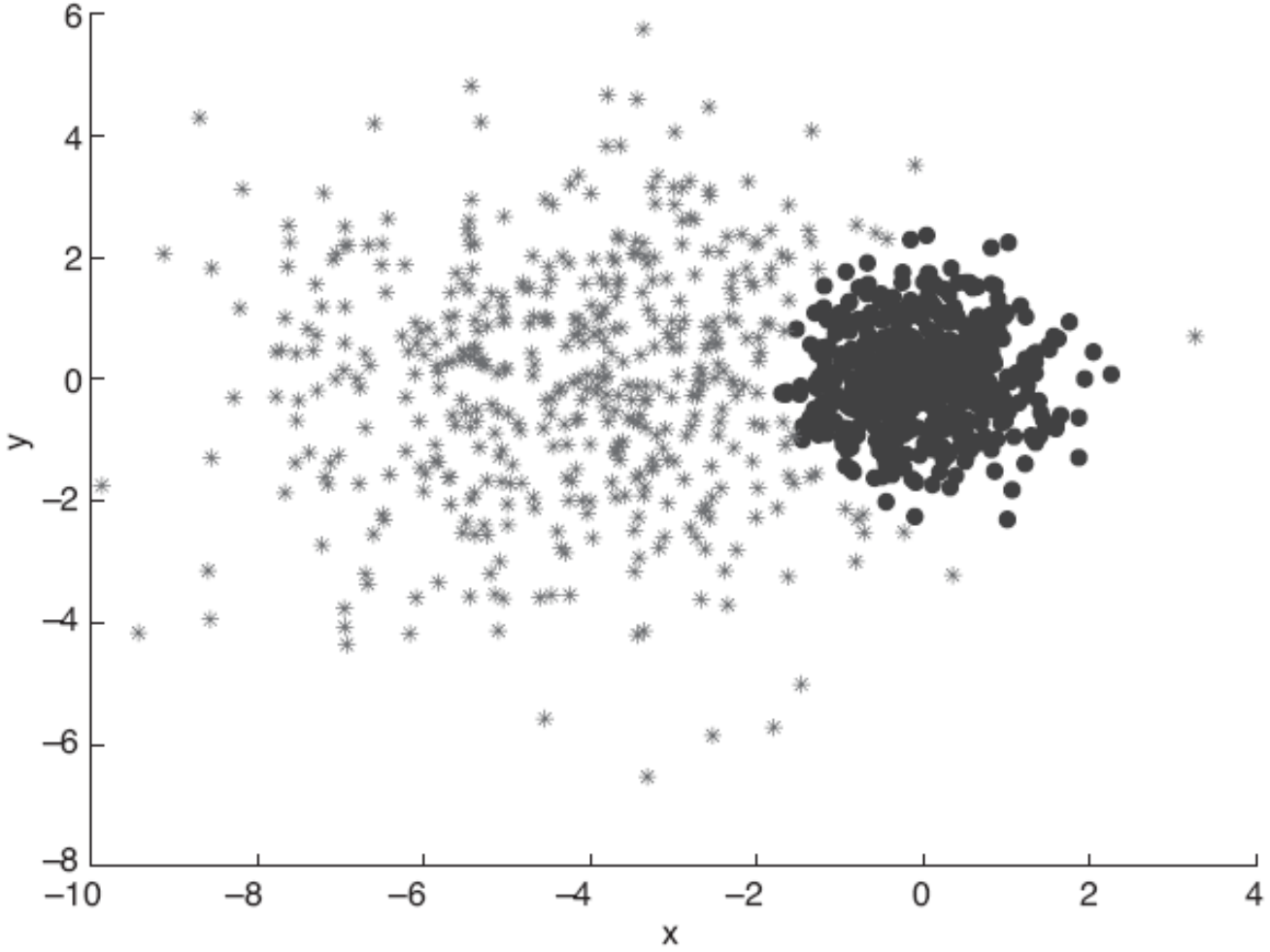
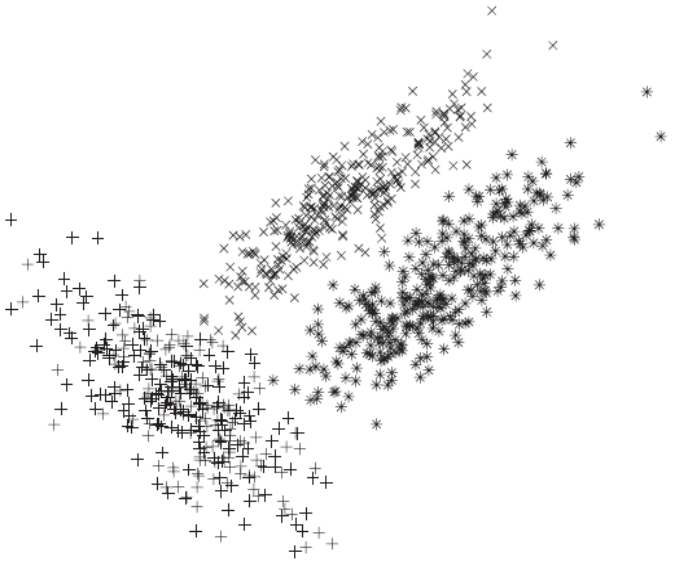
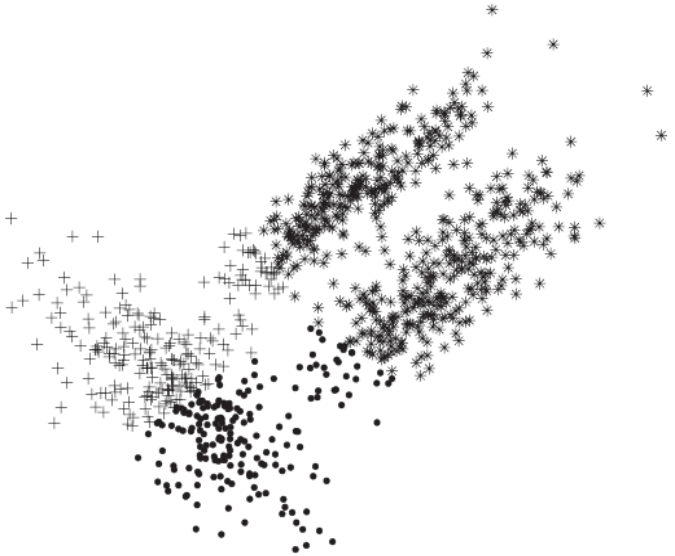


Figure 9.5. EM clustering of a two-dimensional point set with two clusters of differing density.



(a) Clusters produced by mixture model clustering.



(b) Clusters produced by K-means clustering.

Figure 9.6. Mixture model and K-means clustering of a set of two-dimensional points.

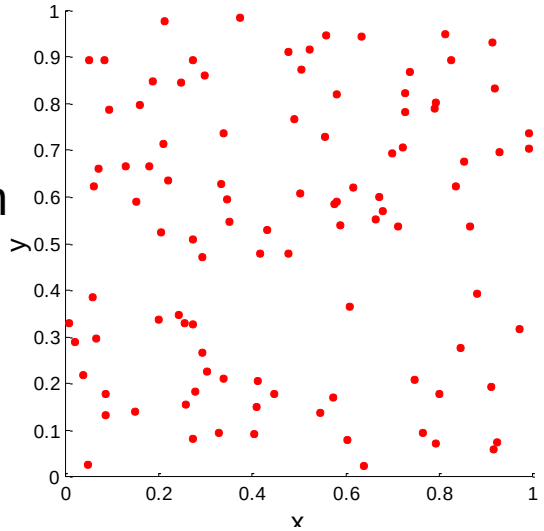
CLUSTERING EVALUATION

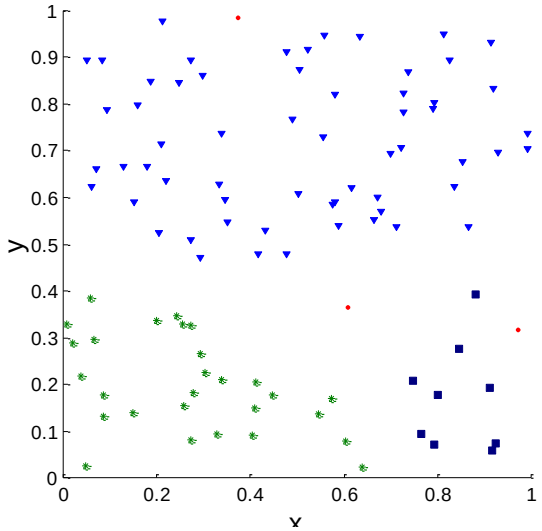
Clustering Evaluation

- How do we evaluate the "goodness" of the resulting clusters?
- But "clustering lies in the eye of the beholder"!
- Then why do we want to evaluate them?
  - To avoid finding patterns in noise
  - To compare clusterings, or clustering algorithms
  - To compare against a "ground truth"

Clusters found in Random Data

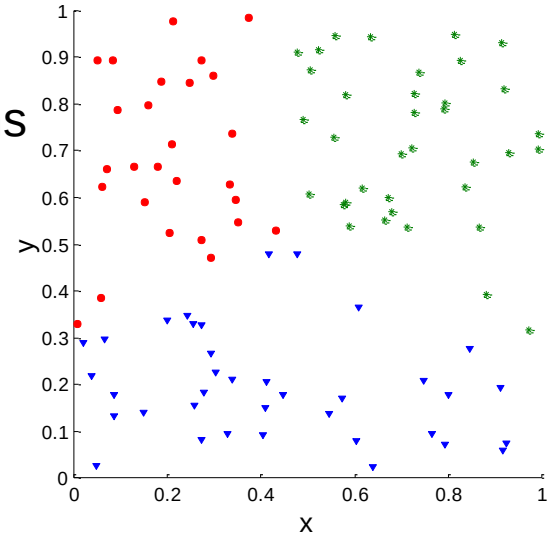
Random Points

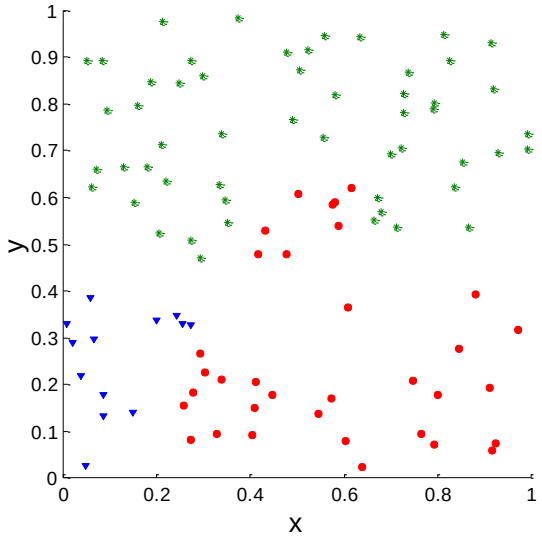




DBSCAN

K-means





Complete Link

Different Aspects of Cluster Validation

1. Determining the clustering tendency of a set of data, i.e., distinguishing whether non-random structure actually exists in the data.
2. Comparing the results of a cluster analysis to externally known results, e.g., to externally given class labels.
3. Evaluating how well the results of a cluster analysis fit the data *without* reference to external information.
   - Use only the data
4. Comparing the results of two different sets of cluster analyses to determine which is better.
5. Determining the 'correct' number of clusters.

For 2, 3, and 4, we can further distinguish whether we want to evaluate the entire clustering or just individual clusters.

Measures of Cluster Validity

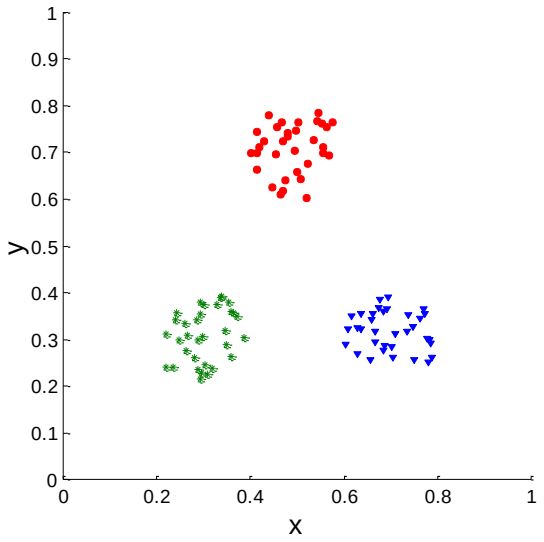
- Numerical measures that are applied to judge various aspects of cluster validity, are classified into the following three types.
  - External Index: Used to measure the extent to which cluster labels match externally supplied class labels.
    - E.g., entropy, precision, recall
  - Internal Index: Used to measure the goodness of a clustering structure without reference to external information.
    - E.g., Sum of Squared Error (SSE)
  - Relative Index: Used to compare two different clusterings or clusters.
    - Often an external or internal index is used for this function, e.g., SSE or entropy
- Sometimes these are referred to as criteria instead of indices
  - However, sometimes criterion is the general strategy and index is the numerical measure that implements the criterion.

Measuring Cluster Validity Via Correlation

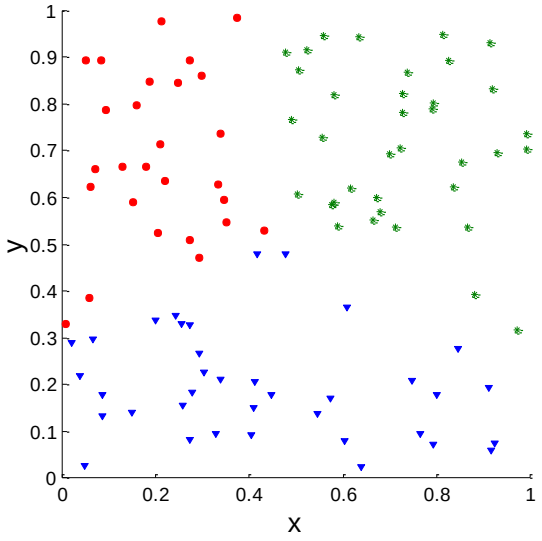
- Two matrices
  - Similarity or Distance Matrix
    - One row and one column for each data point
    - An entry is the similarity or distance of the associated pair of points
  - "Incidence" Matrix
    - One row and one column for each data point
    - An entry is 1 if the associated pair of points belong to the same cluster
    - An entry is 0 if the associated pair of points belongs to different clusters
- Compute the correlation between the two matrices
  - Since the matrices are symmetric, only the correlation between n(n-1) / 2 entries needs to be calculated.
- High correlation (positive for similarity, negative for distance) indicates that points that belong to the same cluster are close to each other.
- Not a good measure for some density or contiguity based clusters.

Measuring Cluster Validity Via Correlation

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



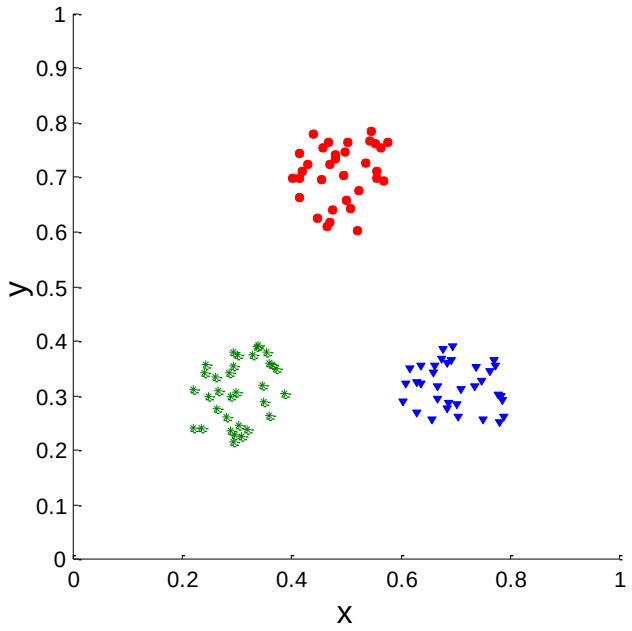
Corr = -0.9235

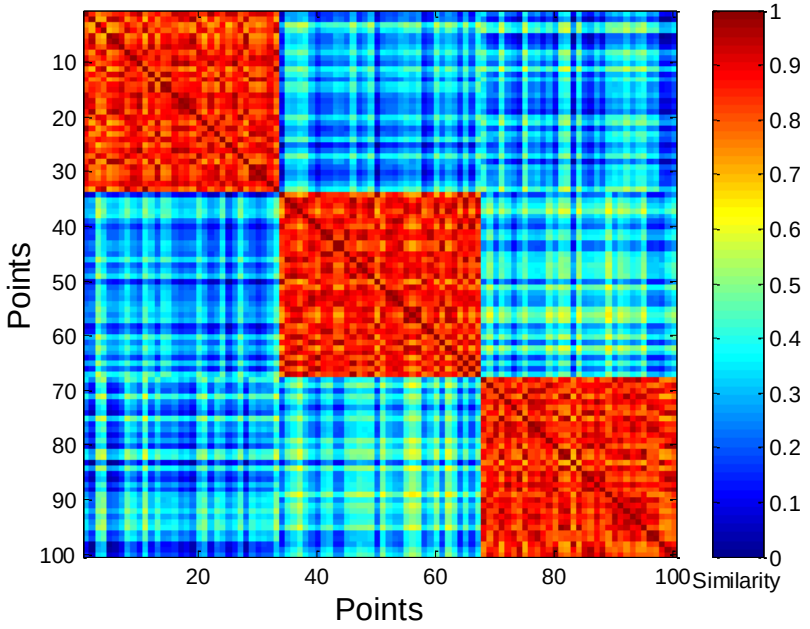


Corr = -0.5810

Using Similarity Matrix for Cluster Validation

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

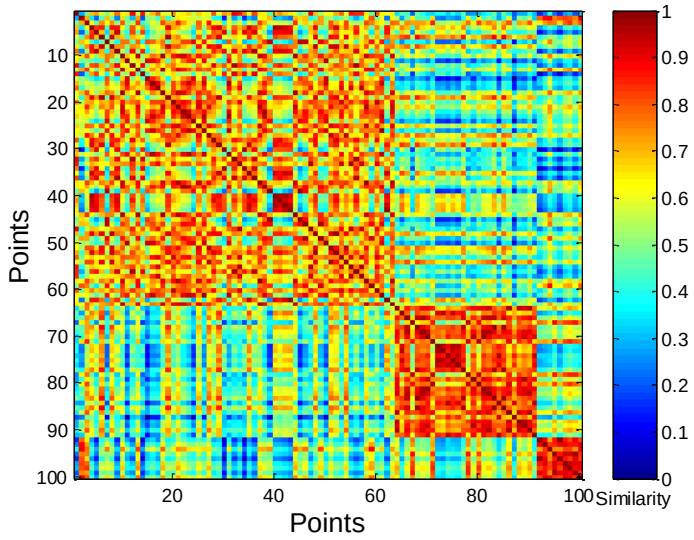


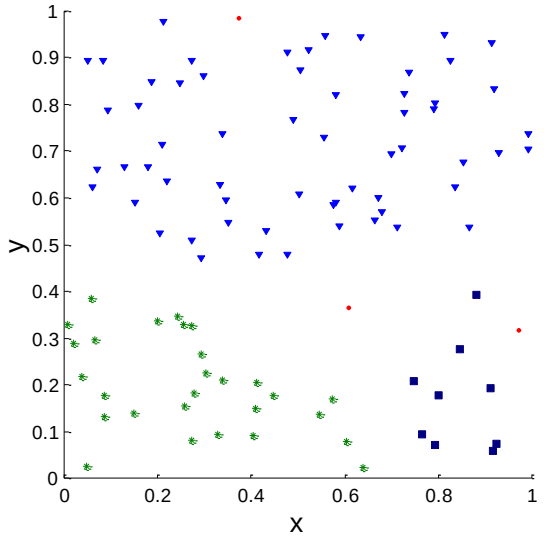


$$sim(i,j) = 1 - \frac{d_{ij} - d_{min}}{d_{max} - d_{min}}$$

Using Similarity Matrix for Cluster Validation

- Clusters in random data are not so crisp

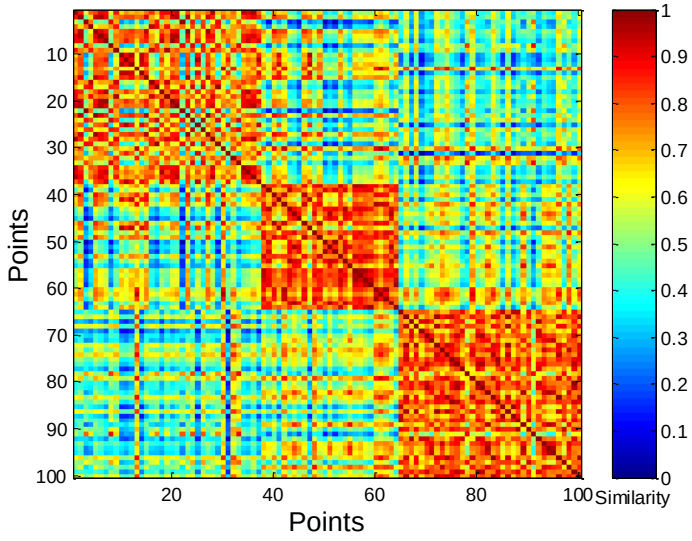


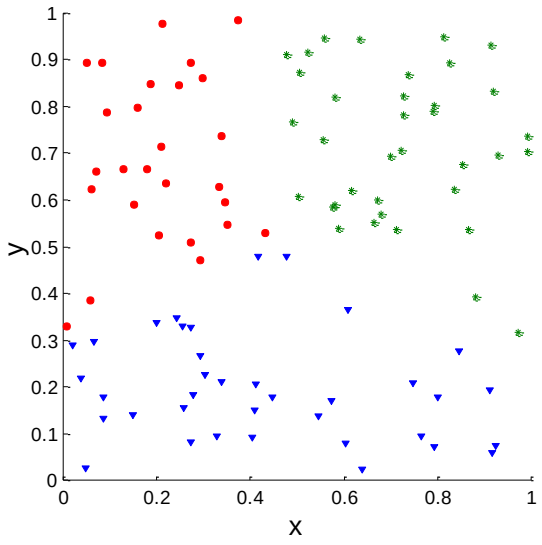


DBSCAN

Using Similarity Matrix for Cluster Validation

- Clusters in random data are not so crisp

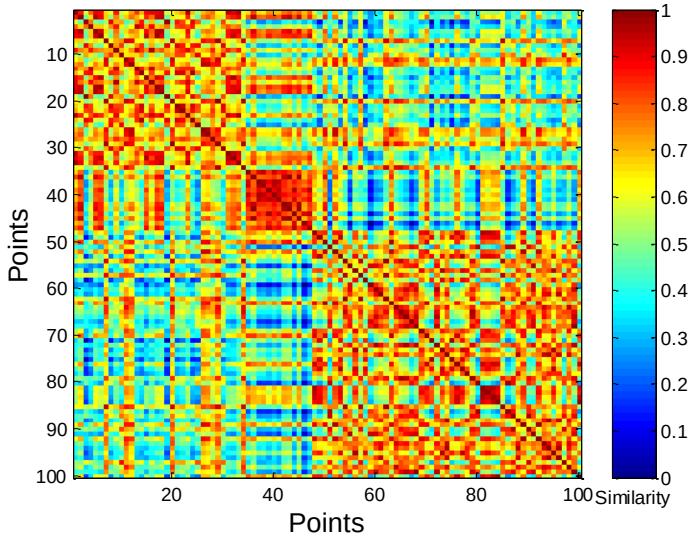


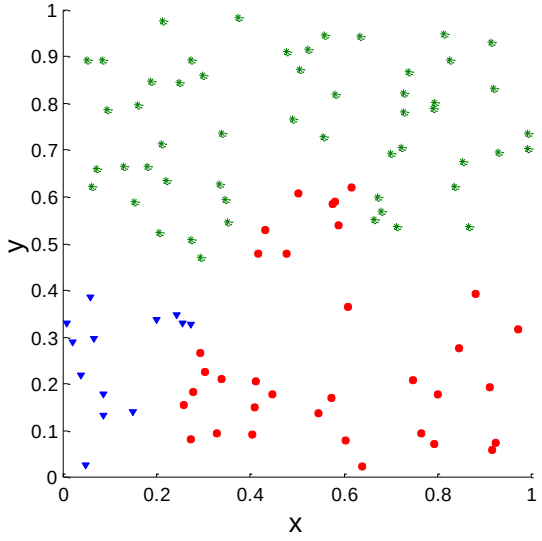


K-means

Using Similarity Matrix for Cluster Validation

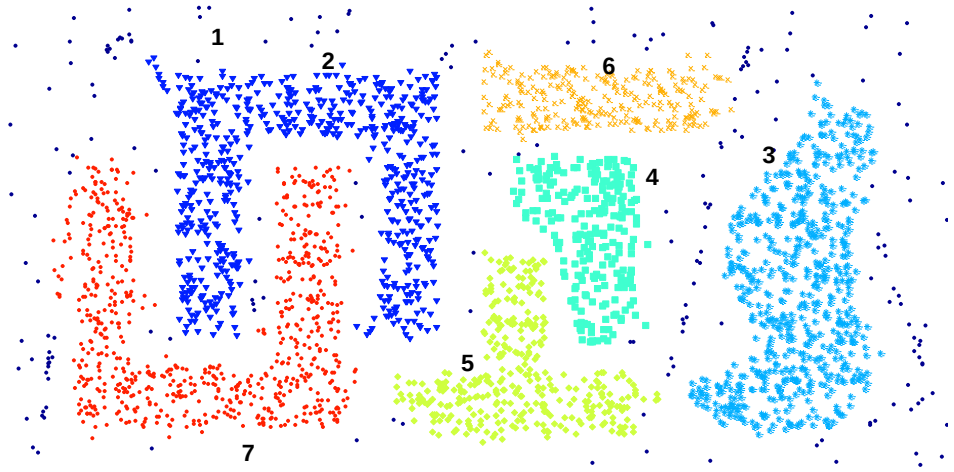
- Clusters in random data are not so crisp

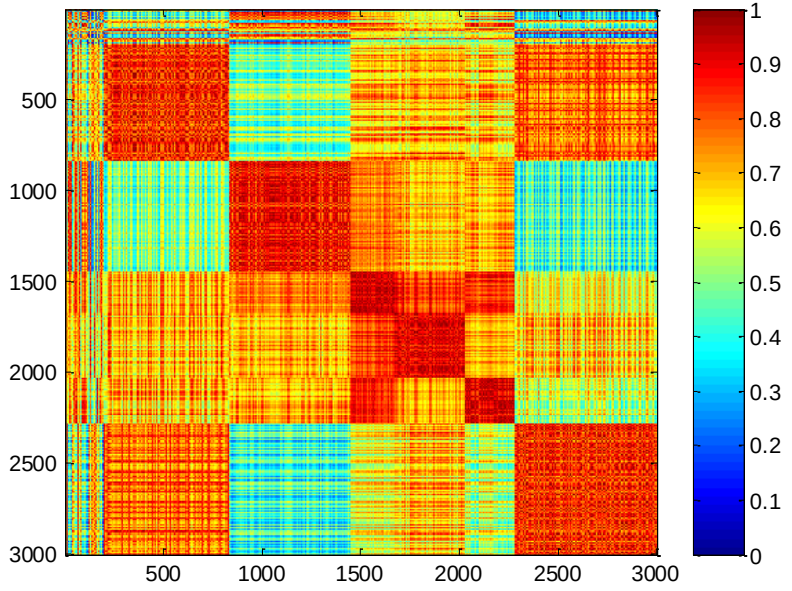




Complete Link

Using Similarity Matrix for Cluster Validation



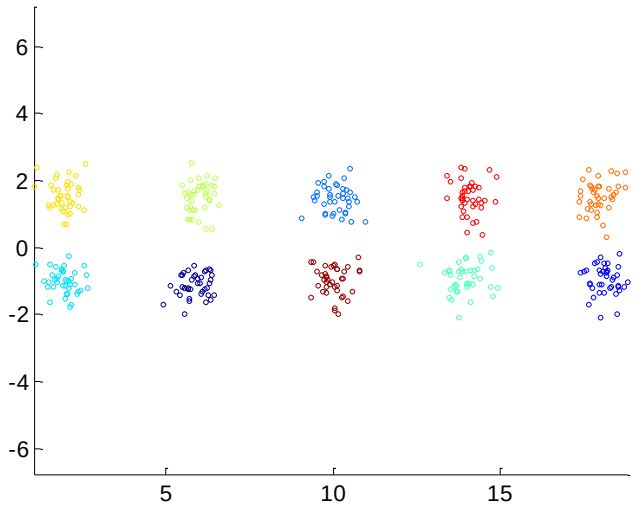


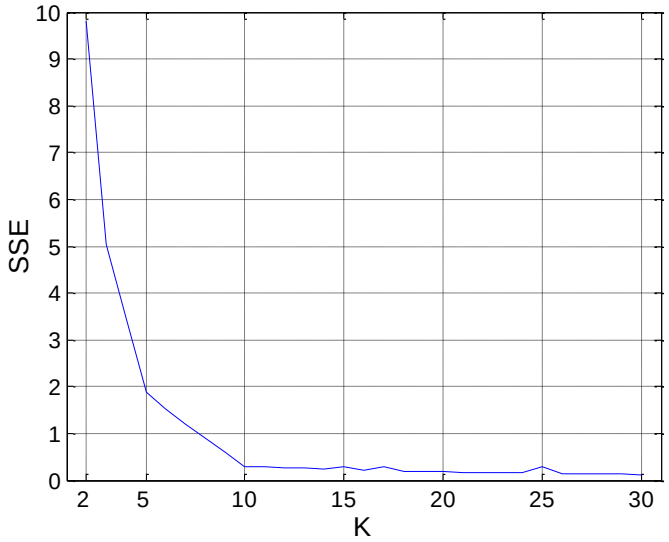
DBSCAN

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

Internal Measures: SSE

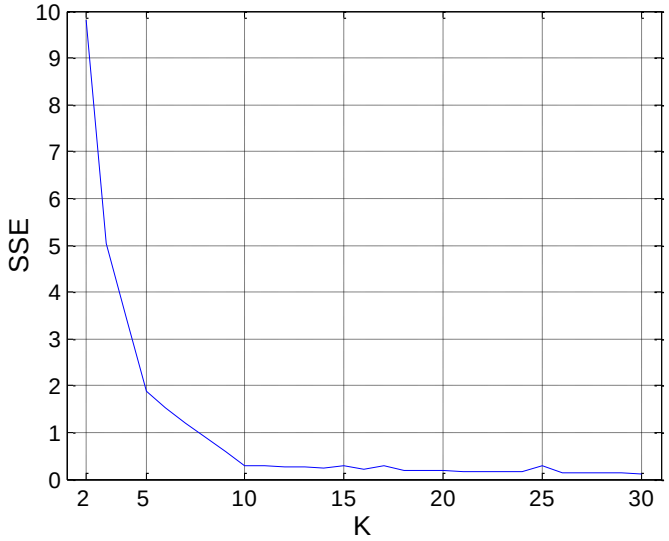
- Internal Index: Used to measure the goodness of a clustering structure without reference to external information
  - Example: SSE
- SSE is good for comparing two clusterings or two clusters (average SSE).
- Can also be used to estimate the number of clusters





Estimating the “right” number of clusters

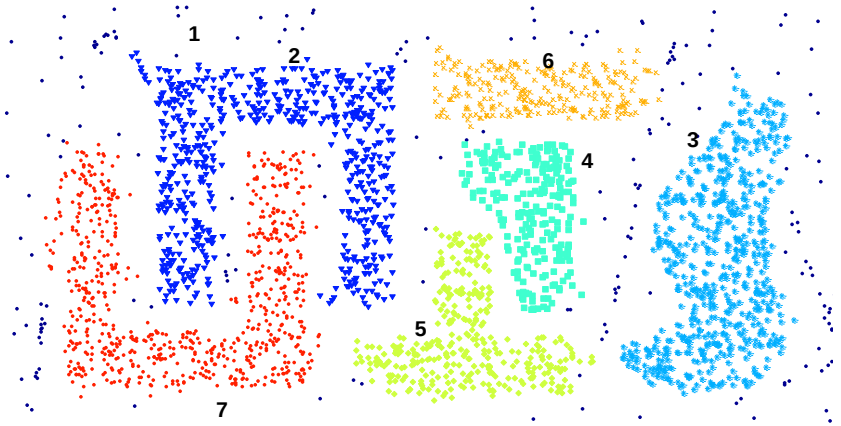
- Typical approach: find a “knee” in an internal measure curve.

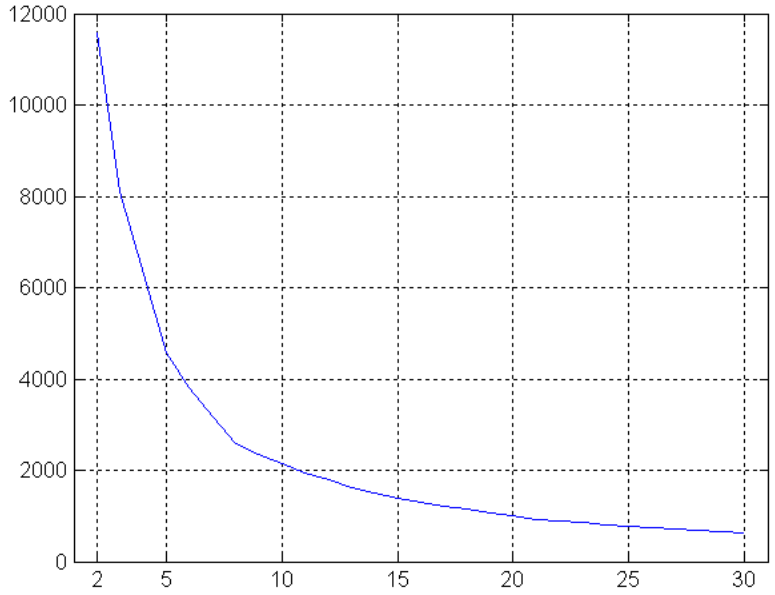


- Question: why not the k that minimizes the SSE?
  - Forward reference: minimize a measure, but with a “simple” clustering
- Desirable property: the clustering algorithm does not require the number of clusters to be specified (e.g., DBSCAN)

Internal Measures: SSE

- SSE curve for a more complicated data set





SSE of clusters found using K-means

Internal Measures: Cohesion and Separation

- Cluster Cohesion: Measures how closely related are objects in a cluster
- Cluster Separation: Measure how distinct or well-separated a cluster is from other clusters
- Example: Squared Error
  - Cohesion is measured by the within cluster sum of squares (SSE)

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

We want this to be small

  - Separation is measured by the between cluster sum of squares

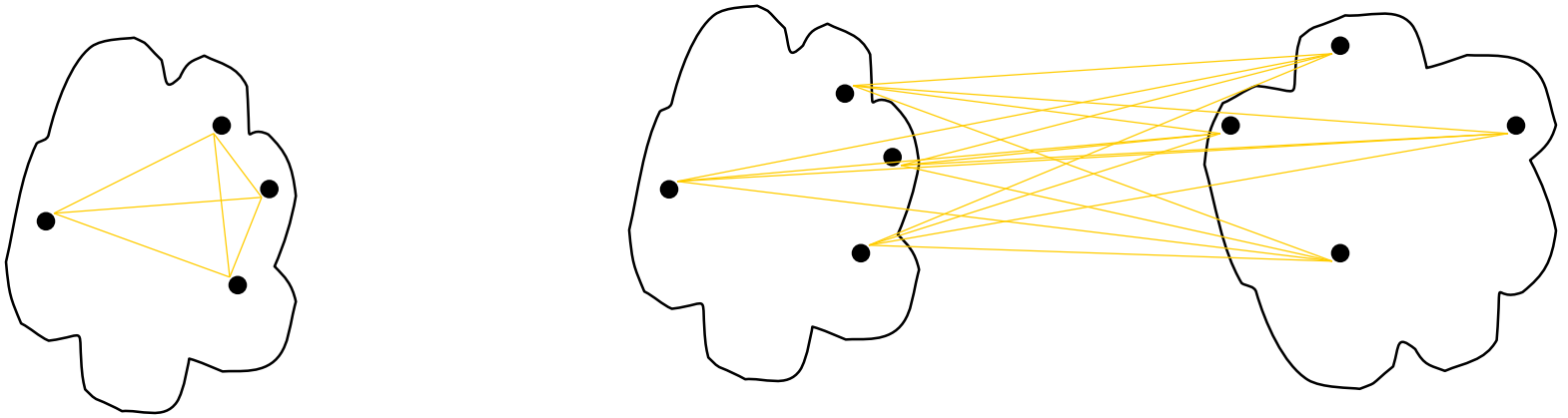
$$BSS = \sum_i m_i (c - c_i)^2$$

We want this to be large

    - Where m_i is the size of cluster i

Internal Measures: Cohesion and Separation

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



cohesion

separation

Internal measures – caveats

- Internal measures have the problem that the clustering algorithm did not set out to optimize this measure, so it is will not necessarily do well with respect to the measure.

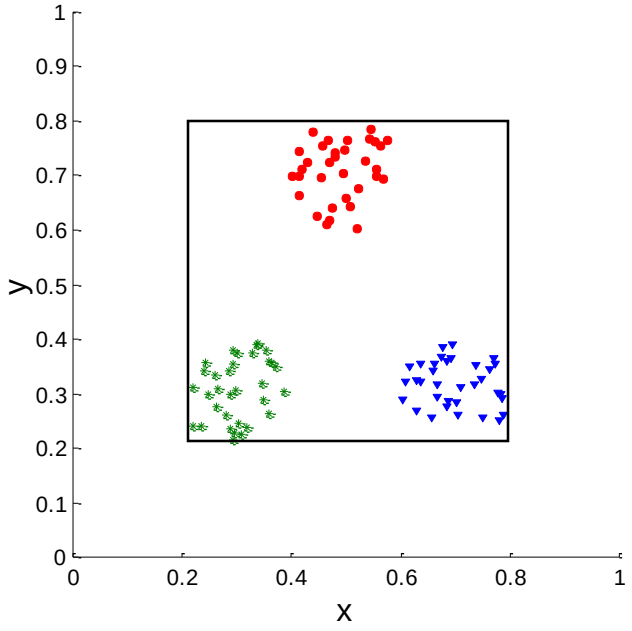
- An internal measure can also be used as an objective function for clustering

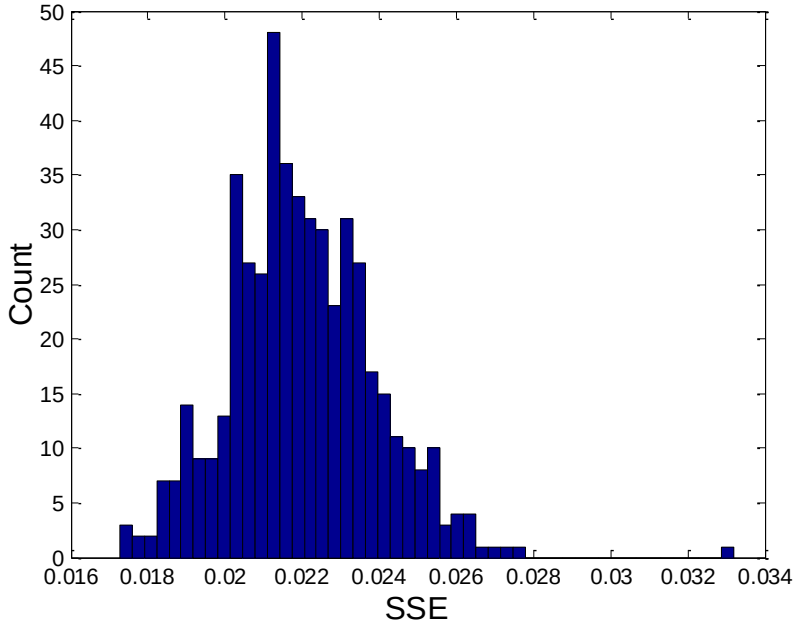
Framework for Cluster Validity

- Need a framework to interpret any measure.
  - For example, if our measure of evaluation has the value, 10, is that good, fair, or poor?
- Statistics provide a framework for cluster validity
  - The more "non-random" a clustering result is, the more likely it represents valid structure in the data
  - Can compare the values of an index that result from random data or clusterings to those of a clustering result.
    - If the value of the index is unlikely, then the cluster results are valid
- For comparing the results of two different sets of cluster analyses, a framework is less necessary.
  - However, there is the question of whether the difference between two index values is significant

Statistical Framework for SSE

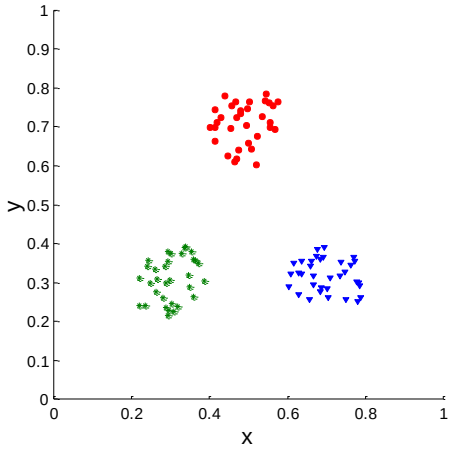
- Example
  - Compare SSE of 0.005 against three clusters in random data
  - Histogram of SSE for three clusters in 500 random data sets of 100 random points distributed in the range 0.2 – 0.8 for x and y
    - Value 0.005 is very unlikely

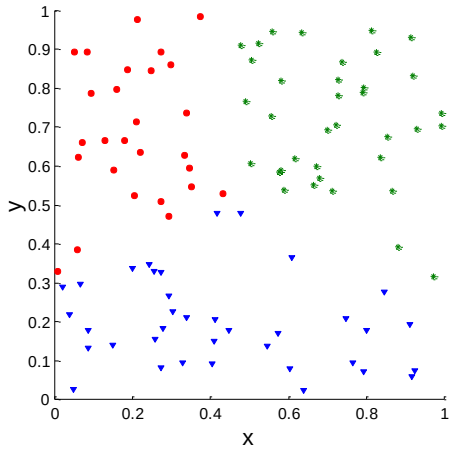


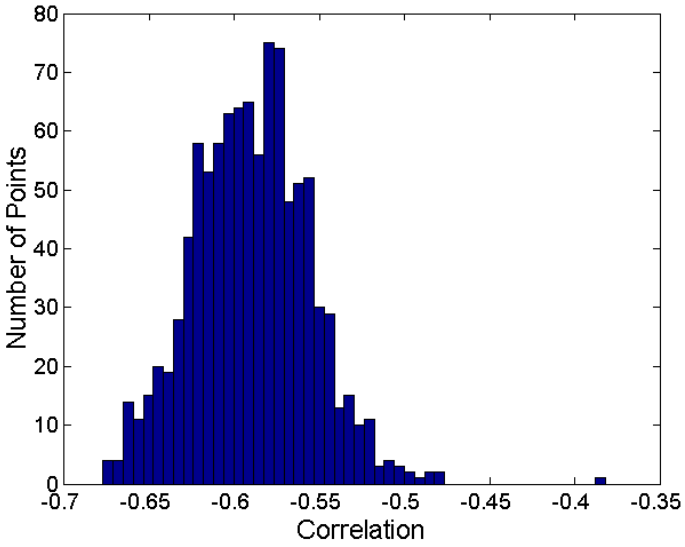


Statistical Framework for Correlation

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







Corr = -0.9235

Corr = -0.5810

Empirical p-value

- If we have a measurement v (e.g., the SSE value)
- ..and we have N measurements on random datasets
- …the empirical p-value is the fraction of measurements in the random data that have value less or equal than value v (or greater or equal if we want to maximize)
  - i.e., the value in the random dataset is at least as good as that in the real data
- We usually require that p-value ≤ 0.05
- Hard question: what is the right notion of a random dataset?

External Measures for Clustering Validity

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

Confusion matrix

- n = number of points
- m_i = points in cluster i
- c_j = points in class j
- n_{ij}= points in cluster i coming from class j
- $p_{ij} = n_{ij}/m_i$= probability of element from cluster i to be assigned in class j

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

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

Measures

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

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

Measures

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

- Precision:
  - Of cluster i with respect to class j: $Prec(i,j) = p_{ij}$
- Recall:
  - Of cluster i with respect to class j: $Rec(i,j) = \frac{n_{ij}}{c_j}$
- F-measure:
  - Harmonic Mean of Precision and Recall:

$$F(i,j) = \frac{2 * Prec(i,j) * Rec(i,j)}{Prec(i,j) + Rec(i,j)}$$

Measures

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

Precision/Recall for clusters and clusterings

- Assign to cluster i the class k_i such that $k_i = \arg\max_j n_{ij}$
- Precision:
  - Of cluster i: $Prec(i) = \frac{n_{ik_i}}{m_i}$
  - Of the clustering: $Prec(C) = \sum_i \frac{m_i}{n} Prec(i)$
- Recall:
  - Of cluster i: $Rec(i) = \frac{n_{ik_i}}{c_{k_i}}$
  - Of the clustering: $Rec(C) = \sum_i \frac{m_i}{n} Rec(i)$
- F-measure:
  - Harmonic Mean of Precision and Recall

Good and bad clustering

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

Purity: (0.94, 0.81, 0.85)
– overall 0.86
Precision: (0.94, 0.81, 0.85)
– overall 0.86
Recall: (0.85, 0.9, 0.85)
- overall 0.87

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

Purity: (0.38, 0.38, 0.38)
– overall 0.38
Precision: (0.38, 0.38, 0.38)
– overall 0.38
Recall: (0.35, 0.42, 0.38)
– overall 0.39

Another clustering

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

Cluster 1:

Purity: 1

Precision: 1

Recall: 0.35

External Measures of Cluster Validity: Entropy and Purity

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

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

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

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

Final Comment on Cluster Validity

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

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

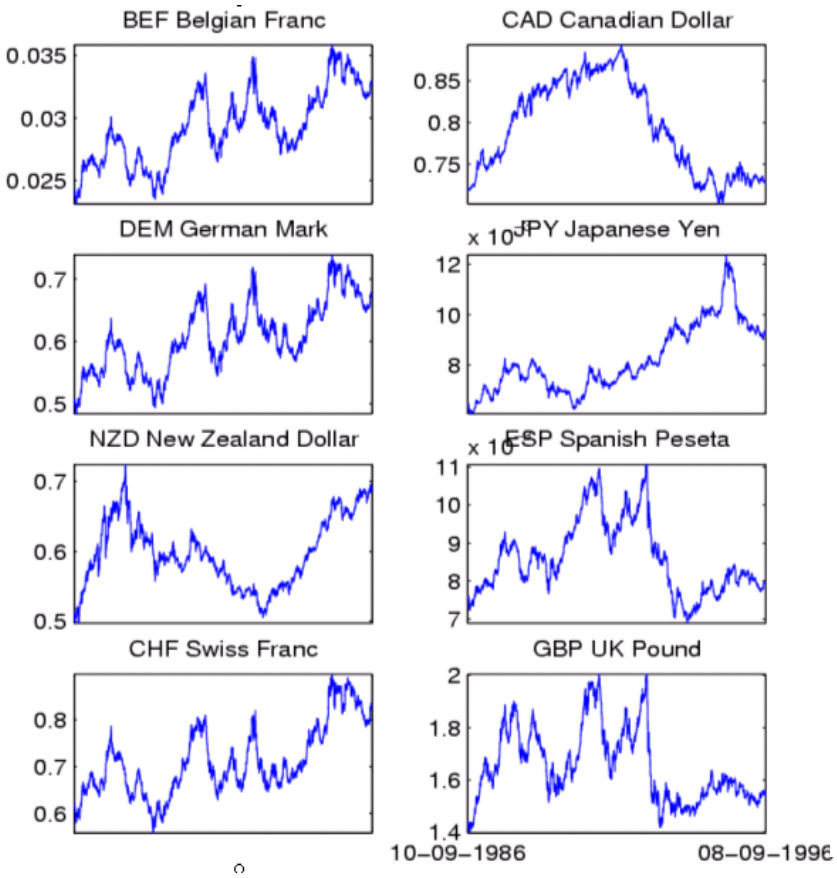
Algorithms for Clustering Data, Jain and Dubes

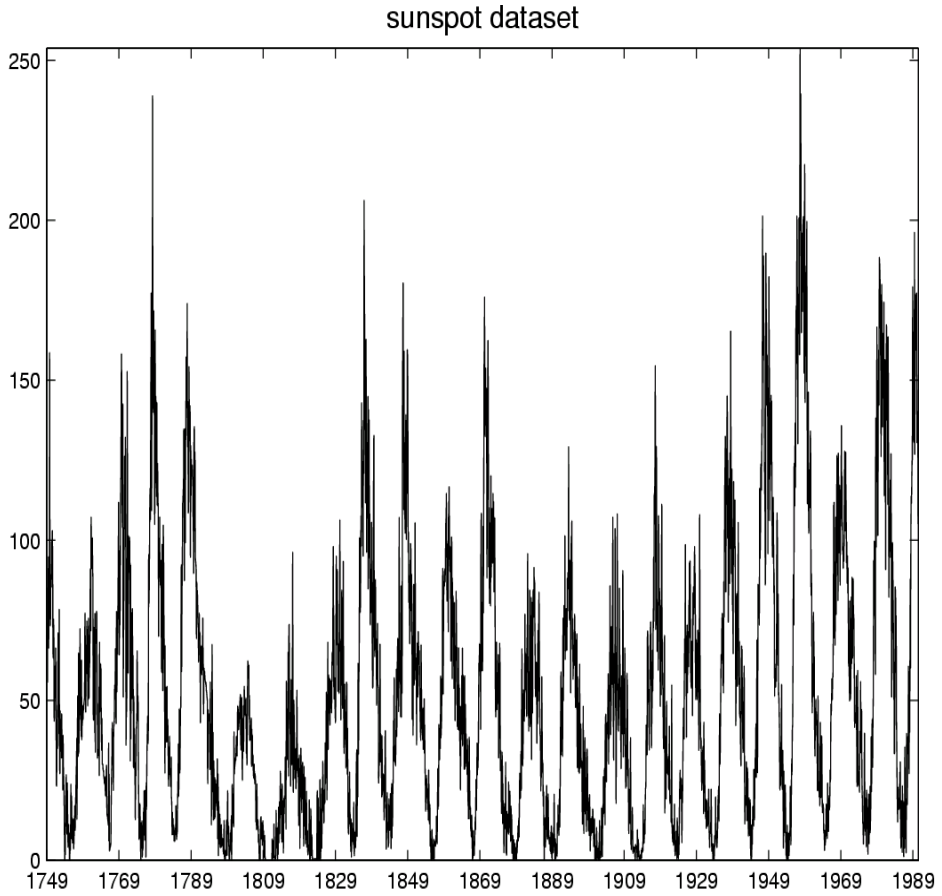
SEQUENCE SEGMENTATION

Sequential data

- Sequential data (or time series) refers to data that appear in a specific order.
  - The order defines a time axis, that differentiates this data from other cases we have seen so far
- Examples
  - The price of a stock (or of many stocks) over time
  - Environmental data (pressure, temperature, precipitation etc) over time
  - The sequence of queries in a search engine, or the frequency of a single query over time
  - The words in a document as they appear in order
  - A DNA sequence of nucleotides
  - Event occurrences in a log over time
  - Etc…
- Time series: usually we assume that we have a vector of numeric values that change over time.

Time-series data





- Financial time series, process monitoring…

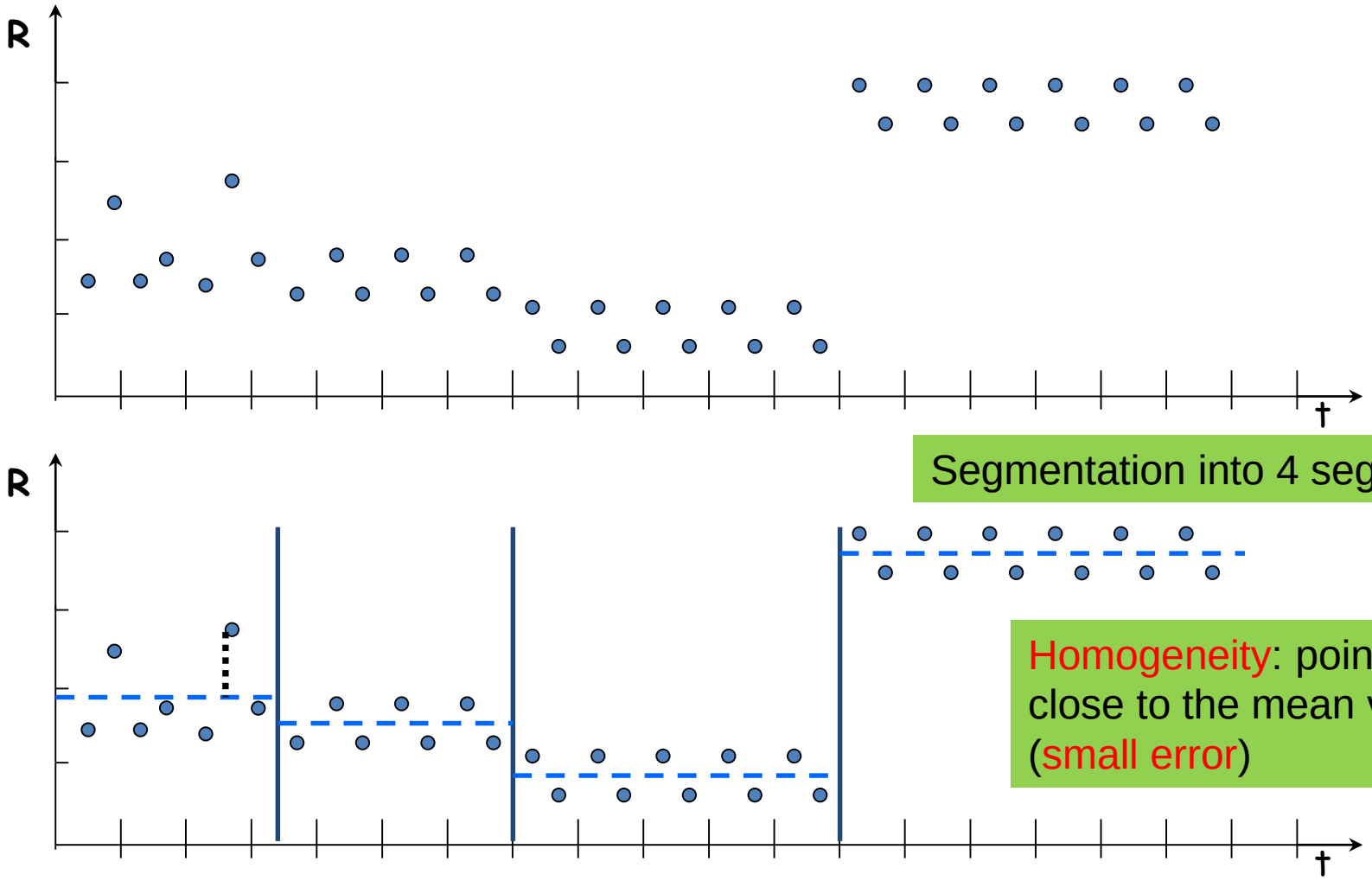
Time series analysis

- The addition of the time axis defines new sets of problems
  - Discovering periodic patterns in time series
  - Defining similarity between time series
  - Finding bursts, or outliers
- Also, some existing problems need to be revisited taking sequential order into account
  - Association rules and Frequent Itemsets in sequential data
  - Summarization and Clustering: Sequence Segmentation

Sequence Segmentation

- Goal: discover structure in the sequence and provide a concise summary
- Given a sequence T, segment it into K contiguous segments that are as homogeneous as possible
- Similar to clustering but now we require the points in the cluster to be contiguous
- Commonly used for summarization of histograms in databases

Example



Segmentation into 4 segments

Homogeneity: points are close to the mean value (small error)

Basic definitions

- Sequence $T = \{t_1, t_2, \ldots, tN\}$: an ordered set of N d-dimensional real points $t_i \in Rd$
- A K-segmentation S: a partition of T into K contiguous segments $\{s_1, s_2, \ldots, sK\}$.
  - Each segment $s \in S$ is represented by a single vector $\mu \in \mathbb{R}^d$(the representative of the segment -- same as the centroid of a cluster)
- Error E(S): The error of replacing individual points with representatives
  - Different error functions, define different representatives.
- Sum of Squares Error (SSE):

$$E(S) = \sum_{s \in S} \sum_{t \in s} (t - \mu_s)^2$$

- Representative of segment s with SSE: mean $\mu_s = \frac{1}{|s|} \sum_{t \in s} t$

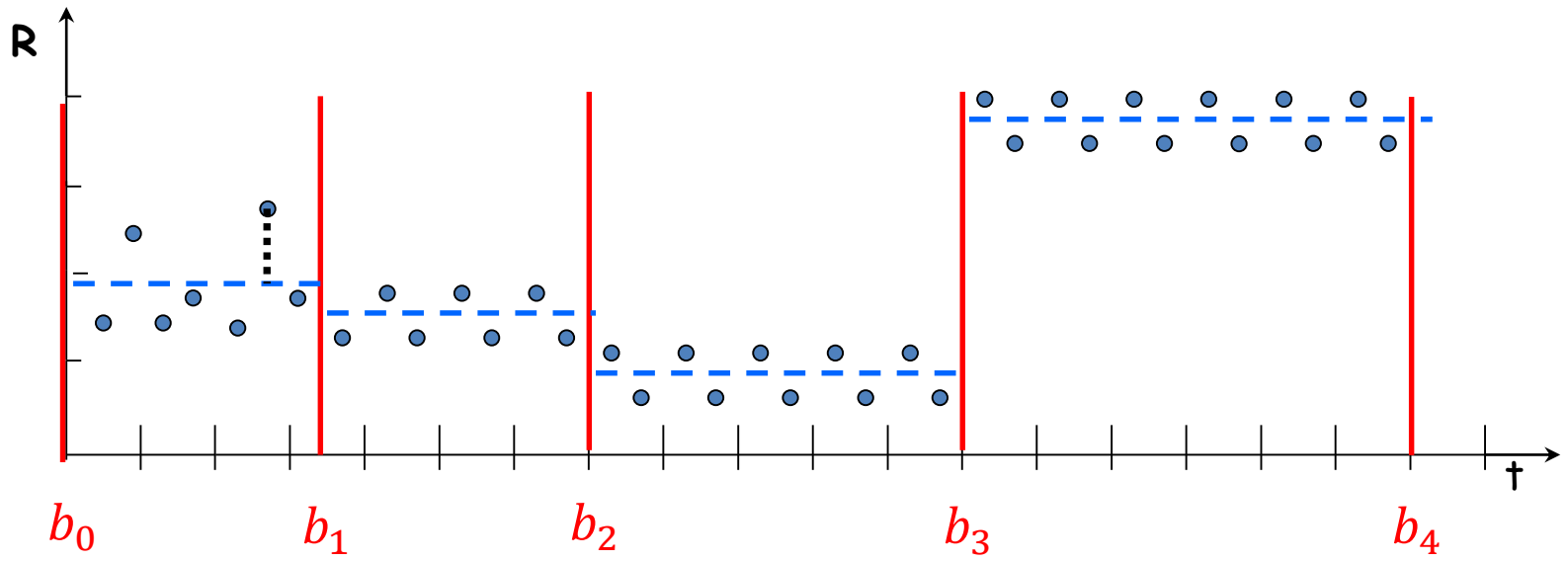
The K-segmentation problem

Given a sequence T of length N and a value K, find a K-segmentation $S = \{s_1, s_2, \ldots, s_K\}$ of T such that the SSE error E is minimized.

- Similar to K-means clustering, but now we need the points in the clusters to respect the order of the sequence.
  - This actually makes the problem easier.

Basic Definitions

- Observation: a K-segmentation S is defined by $K+1$ boundary points $b_0, b_1, \ldots, b_{K-1}, b_K$.



- $b_0 = 0, b_k = N + 1$ always.
  - We only need to specify $b_1, \ldots, b_{K-1}$

Optimal solution for the k-segmentation problem

- Bellman'61: The K-segmentation problem can be solved optimally using a standard dynamic programming algorithm

- Dynamic Programming:
  - Construct the solution of the problem by using solutions to problems of smaller size
    - Define the dynamic programming recursion
  - Build the solution bottom up from smaller to larger instances
    - Define the dynamic programming table that stores the solutions to the sub-problems

Rule of thumb

- Most optimization problems where order is involved can be solved optimally in polynomial time using dynamic programming.
  - The polynomial exponent may be large though

Dynamic Programming Recursion

- Terminology:
  - $T[1,n]$: subsequence $\{t_1, t_2, \ldots, t_n\}$ for $n \leq N$
  - $E(S[1,n],k)$: error of optimal segmentation of subsequence $T[1,n]$ with k segments for $k \leq K$

- Dynamic Programming Recursion:

$$E(S[1,n],k) = \min_{k \leq j \leq n-1} \left\{ E(S[1,j],k-1) + \sum_{j+1 \leq t \leq n} \left(t - \mu_{[j+1,n]}\right)^2 \right\}$$

Minimum over all possible placements of the last boundary point b_{k-1}

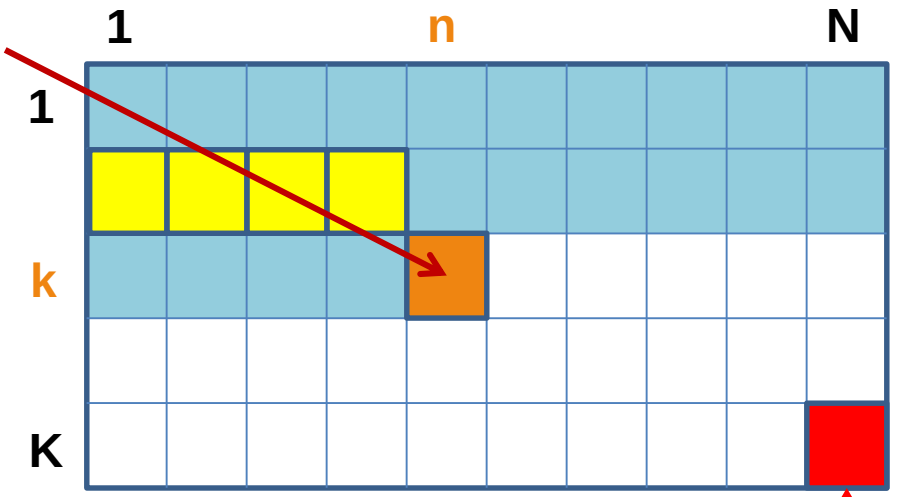
Error of optimal segmentation S[1,j] with k-1 segments

Error of k-th (last) segment when the last segment is [j+1,n]

Dynamic programming table

- Two−dimensional table $A[1 \ldots K, 1 \ldots N]$

$$A[k,n] = E(S[1,n],k)$$

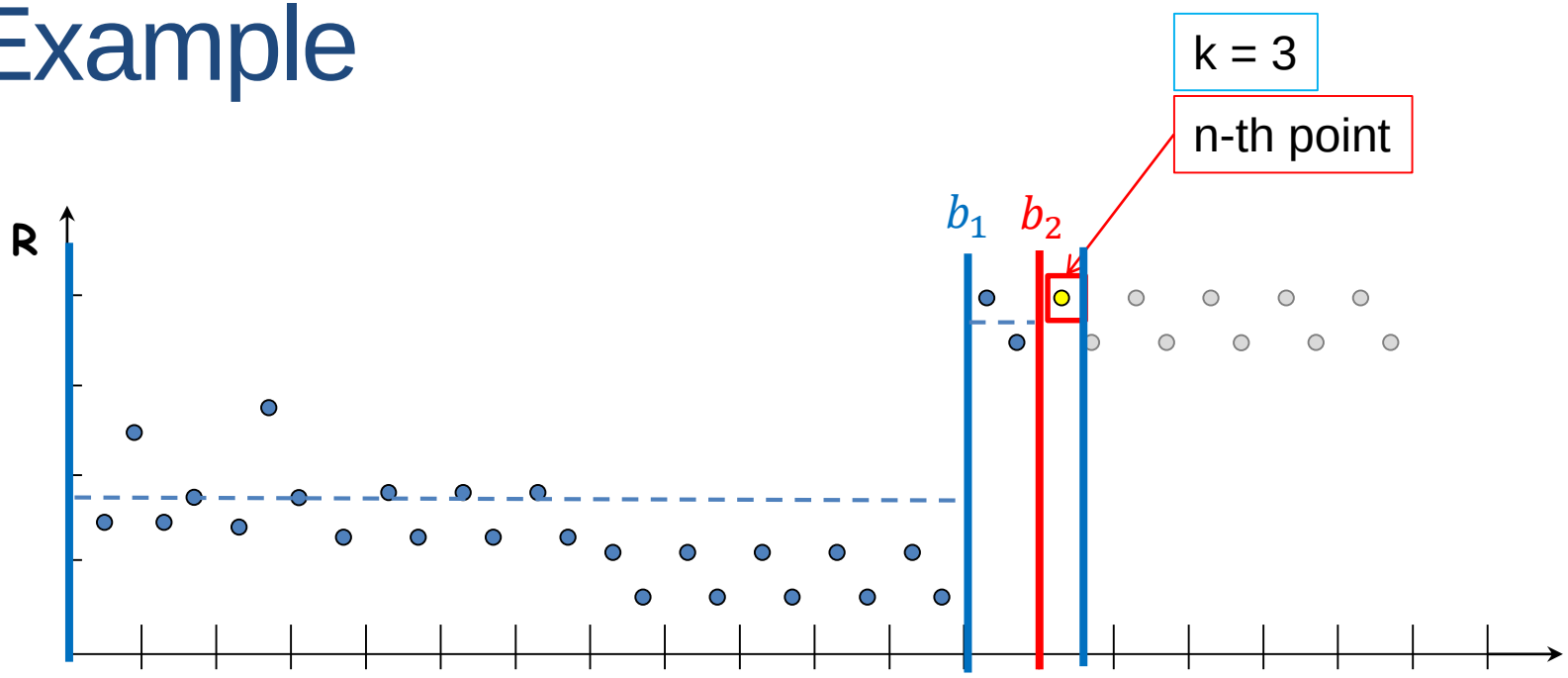


$$E(S[1,n],k) = \min_{k \le j \le n-1} \left\{ E(S[1,j],k-1) + \sum_{j+1 \le t \le n} \left(t - \mu_{[j+1,n]}\right)^2 \right\}$$

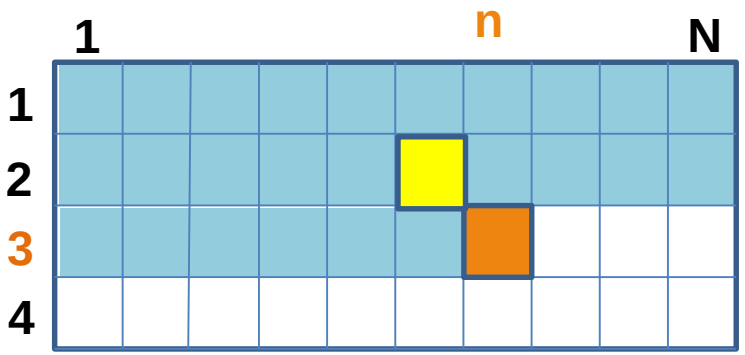
- Fill the table top to bottom, left to right.

Error of optimal K-segmentation

Example

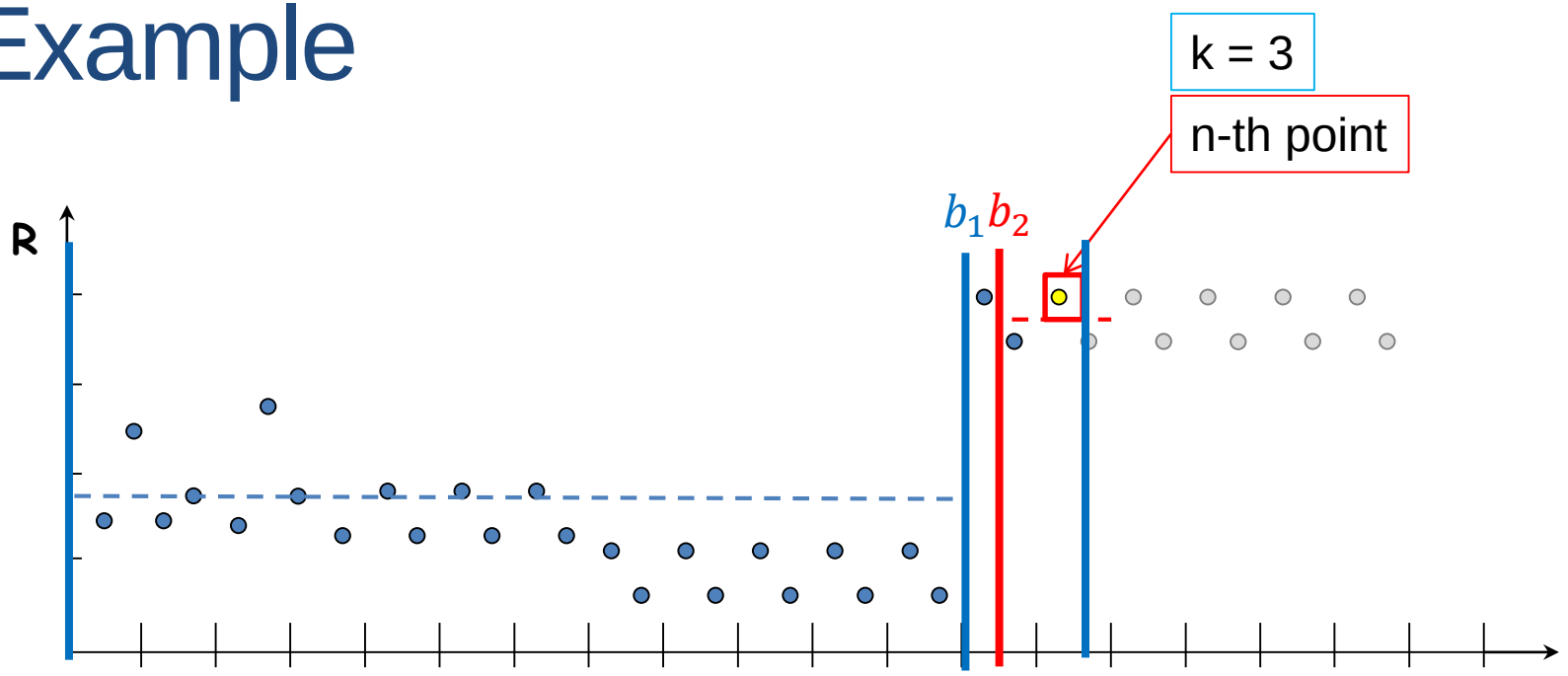


$$E(S[1,n],k) = \min_{k \le j \le n-1} \{ E(S[1,j],k-1) \}$$

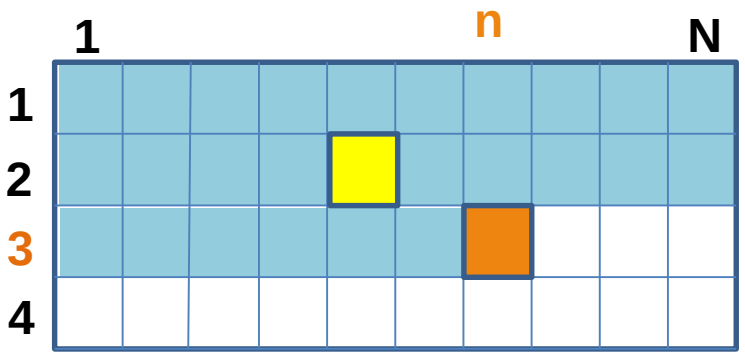


Where should we place boundary b_2 ?

Example

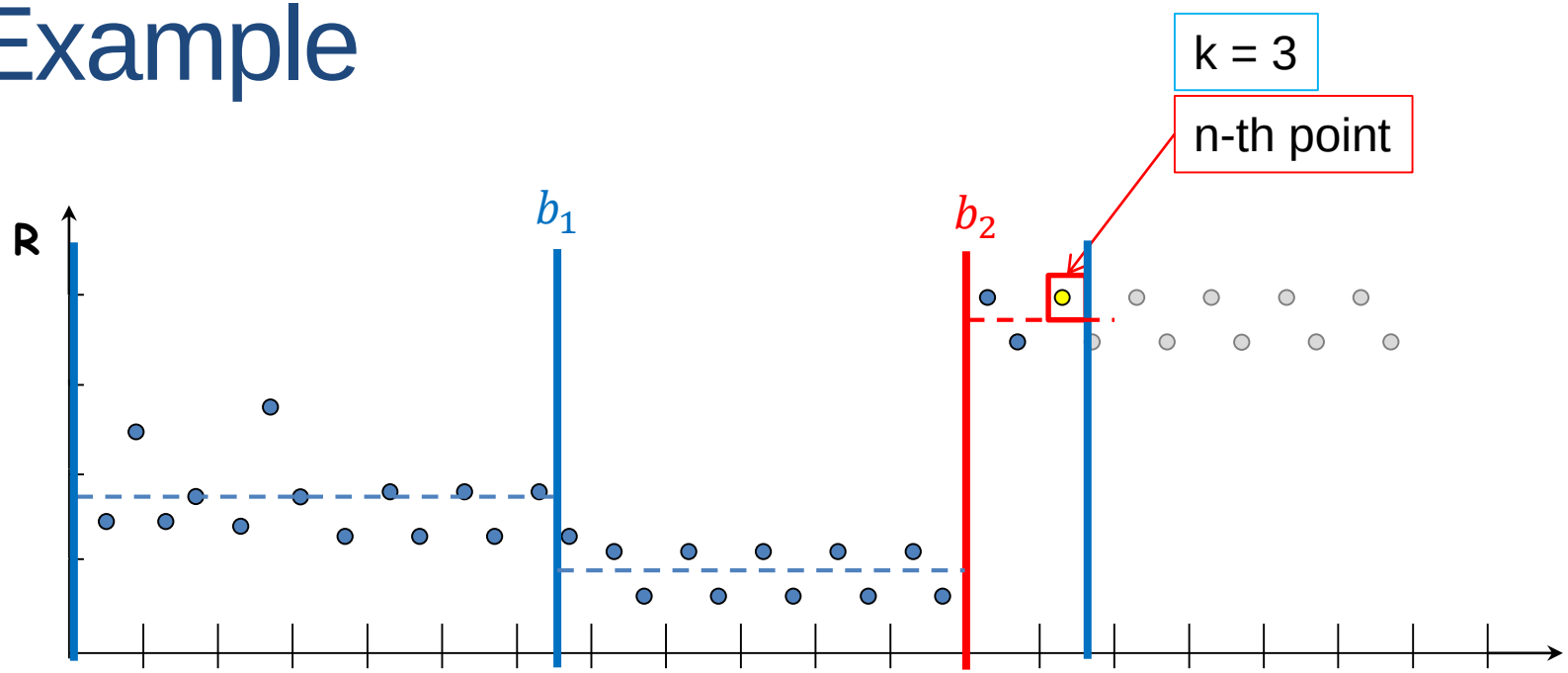


$$E(S[1,n],k) = \min_{k \le j \le n-1} \{ E(S[1,j],k-1) \}$$

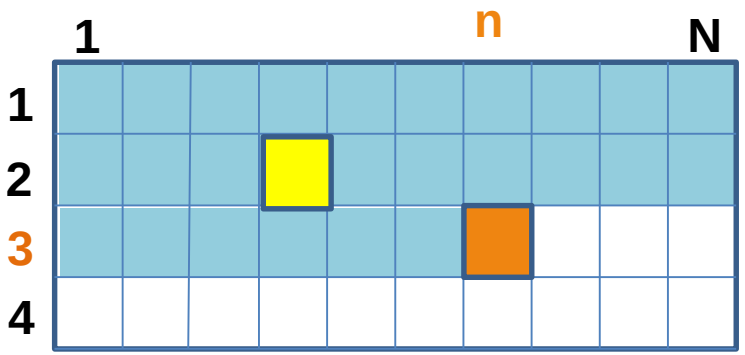


Where should we place boundary b_2 ?

Example

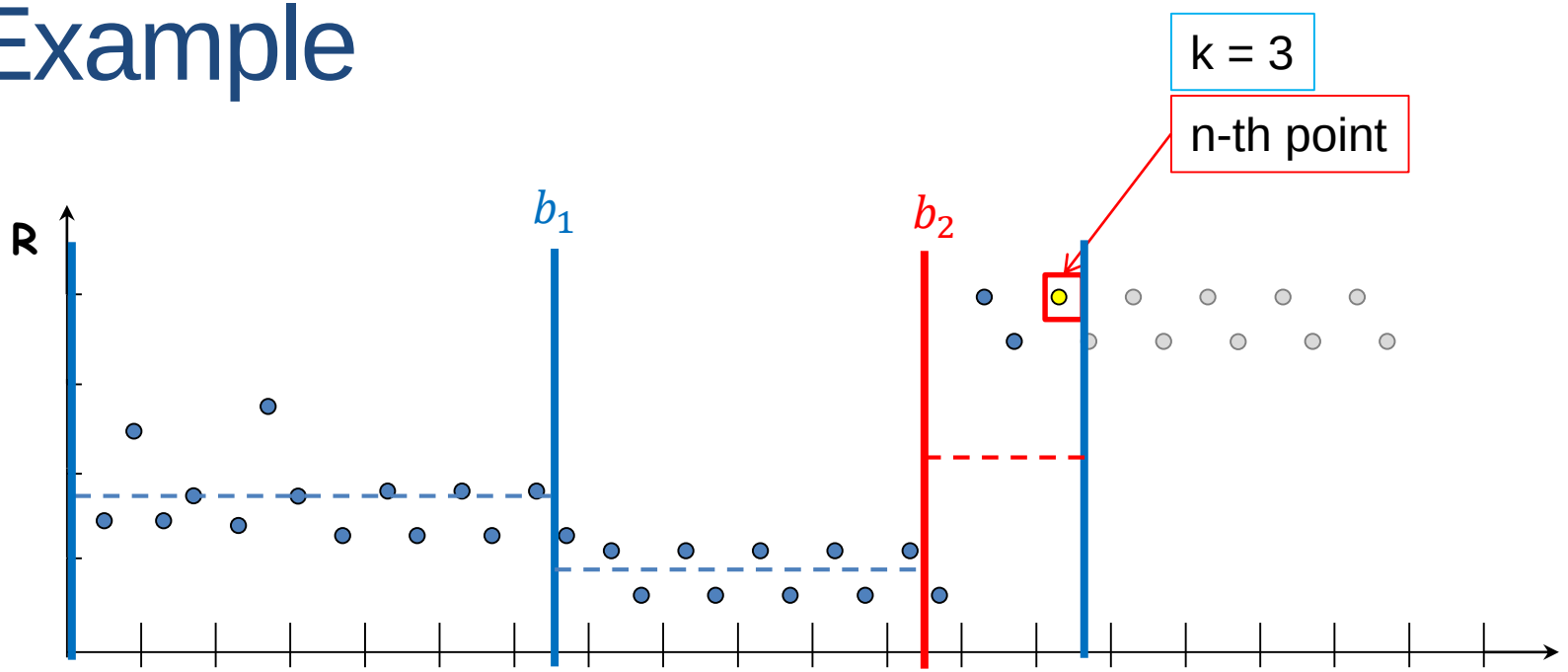


$$E(S[1,n],k) = \min_{k \le j \le n-1} \{ E(S[1,j],k-1) \}$$

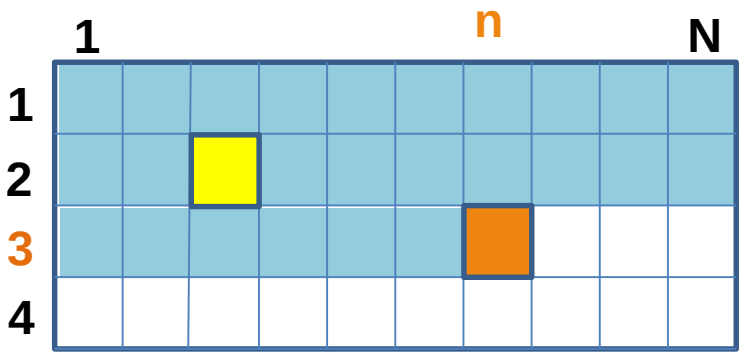


Where should we place boundary b_2 ?

Example

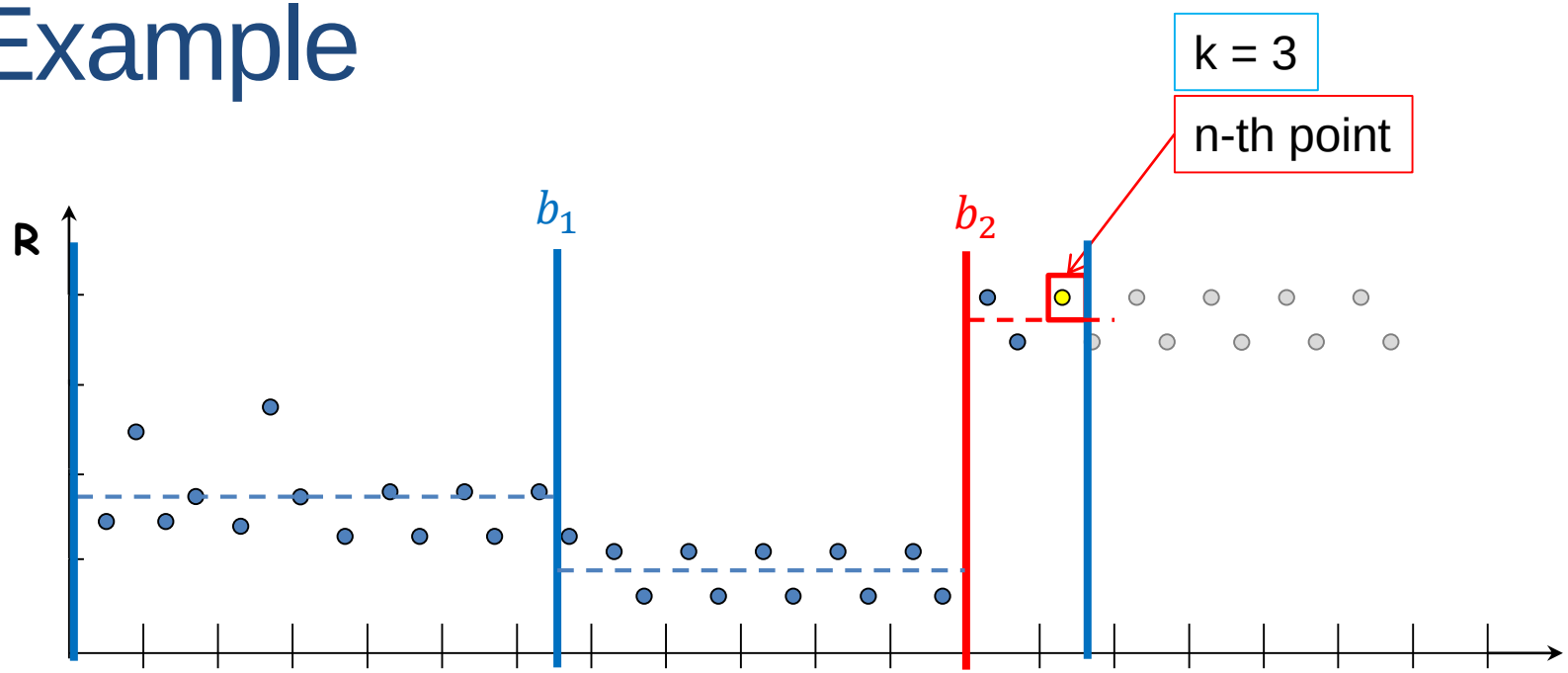


$$E(S[1,n],k) = \min_{k \le j \le n-1} \{ E(S[1,j],k-1) \}$$



Where should we place boundary b_2 ?

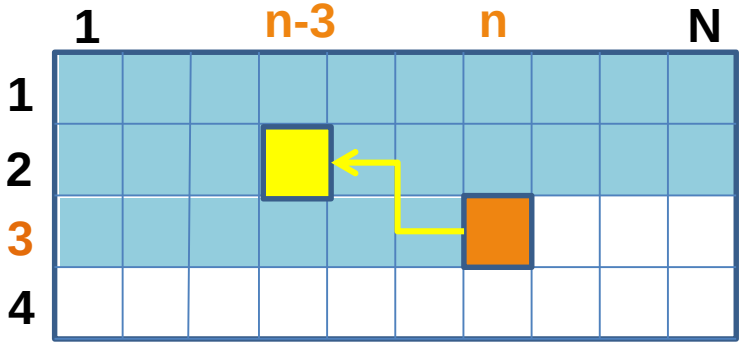
Example



Optimal segmentation S[1:n]

The cell $A[3, n]$ stores the error of the optimal solution 3-segmentation of $T[1, n]$

In the cell (or in a different table) we also store the position $n - 3$ of the boundary so we can trace back the segmentation



Dynamic-programming algorithm

- Input: Sequence **T**, length **N**, **K** segments, error function **E()**

  - For **i=1** to **N** //Initialize first row
    - **A[1,i]=E(T[1…i])** //Error when everything is in one cluster
  - For **k=1** to **K** // Initialize diagonal
    - **A[k,k] = 0** // Error when each point in its own cluster
  - For **k=2** to **K**
    - For **i=k+1** to **N**
      - **$A[k,i] = \min_{j<i}\{A[k-1,j]+E(T[j+1…i])\}$**

- To recover the actual segmentation (not just the optimal cost) store also the minimizing values **j**

Algorithm Complexity

- What is the complexity?
- NK cells to fill
- Computation per cell $E(S[1,n],k) = \min_{k \le j < n} \{E(S[1,j],k-1) + \sum_{j+1 \le t \le n}(t -$

Heuristics

- Top-down greedy (TD): O(NK)
  - Introduce boundaries one at the time so that you get the largest decrease in error, until K segments are created.
- Bottom-up greedy (BU): O(NlogN)
  - Merge adjacent points each time selecting the two points that cause the smallest increase in the error until K segments
- Local Search Heuristics: O(NKI)
  - Assign the breakpoints randomly and then move them so that you reduce the error

Other time series analysis

- Using signal processing techniques is common for defining similarity between series
  - Fast Fourier Transform
  - Wavelets
- Rich literature in the field