

Unsupervised machine learning. Clustering

Alexey Zaytsev

Slides mostly by Evgeny Burnaev and Maxim Panov

Skoltech, Moscow, Russia

Skoltech

Skolkovo Institute of Science and Technology



Skoltech
Skolkovo Institute of Science and Technology

1 Overview

2 Hierarchical clustering

3 K-means

4 DBSCAN

5 Cluster validity

6 Mixture Models

7 Community detection

1 Overview

2 Hierarchical clustering

3 K-means

4 DBSCAN

5 Cluster validity

6 Mixture Models

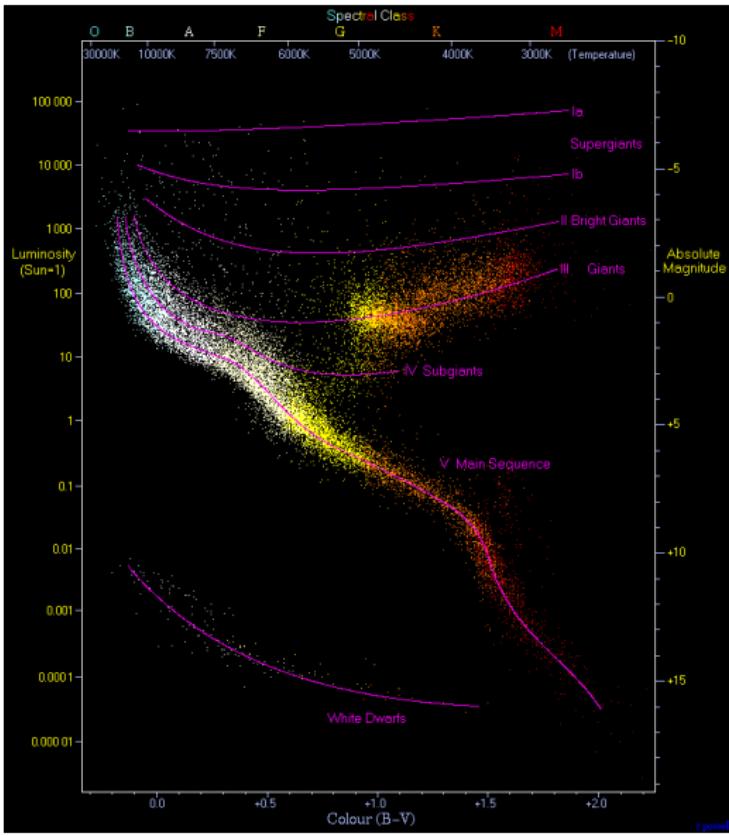
7 Community detection

- Spectral clustering
- Community detection problems
- Modularity
- BigCLAM

Clustering Problem: stars

Solved via the Hertzsprung–Russell diagram based on two features:

- Absolute magnitudes or luminosities
- Colour (B-V): Effective temperature



- A catalog of 2 billion “sky objects” represents objects by their radiation in 7 dimensions (frequency bands).
- **Problem:** Cluster into similar objects, e.g., galaxies, nearby stars, quasars, etc.
- **Sloan Digital Sky Survey:**



- A catalog of 2 billion “sky objects” represents objects by their radiation in 7 dimensions (frequency bands).
- **Problem:** Cluster into similar objects, e.g., galaxies, nearby stars, quasars, etc.
- **Sloan Digital Sky Survey:**



- A catalog of 2 billion “sky objects” represents objects by their radiation in 7 dimensions (frequency bands).
- **Problem:** Cluster into similar objects, e.g., galaxies, nearby stars, quasars, etc.
- **Sloan Digital Sky Survey:**



Finding topics:

- Represent a document by a vector (x_1, x_2, \dots, x_k) , where $x_i = 1$ iff the i -th word (in some order) appears in the document
- Documents with similar sets of words may be about the same topic

Finding topics:

- Represent a document by a vector (x_1, x_2, \dots, x_k) , where $x_i = 1$ iff the i -th word (in some order) appears in the document
- Documents with similar sets of words may be about the same topic

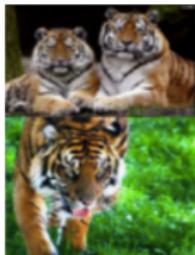
Clustering Problem: Images



What is a cluster?

Goal: partitioning data in maximally homogeneous, maximally distinguished subsets.

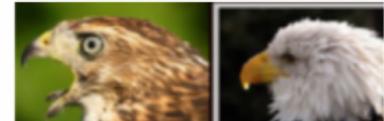
- **Internal criterion:** members of the cluster should be similar to each other (**inter-cluster compactness**).



tigers



whales



raptors

What is a cluster?

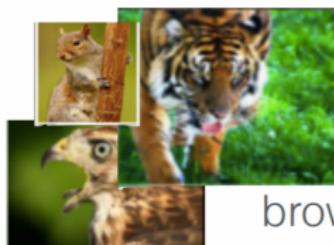
Goal: partitioning data in maximally homogeneous, maximally distinguished subsets.

- **External criterion:** objects outside the cluster should be dissimilar from the objects inside the cluster (**intra-cluster distance**).



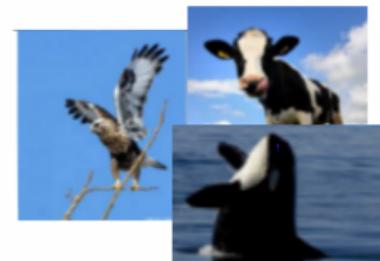
Goal: partitioning data in maximally homogeneous, maximally distinguished subsets.

- **Internal criterion:** members of the cluster should be similar to each other (**inter-cluster compactness**).
- **External criterion:** objects outside the cluster should be dissimilar from the objects inside the cluster (**intra-cluster distance**).



green
brown & white

blue
black
&
white



1 Overview

2 Hierarchical clustering

3 K-means

4 DBSCAN

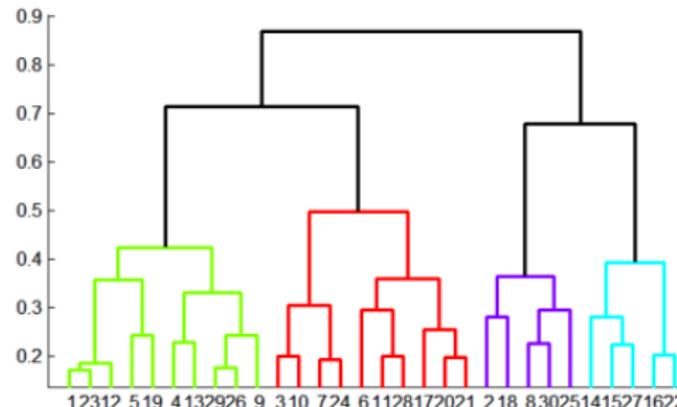
5 Cluster validity

6 Mixture Models

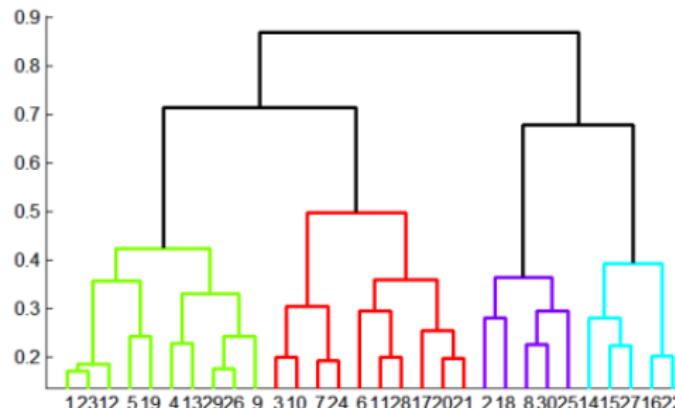
7 Community detection

- Spectral clustering
- Community detection problems
- Modularity
- BigCLAM

- **Agglomerative (bottom up):**
 - Initially, each point is a cluster;
 - Repeatedly combine the two “nearest” clusters into one.
- **Divisive (top down):**
 - Start with one cluster and recursively split it.



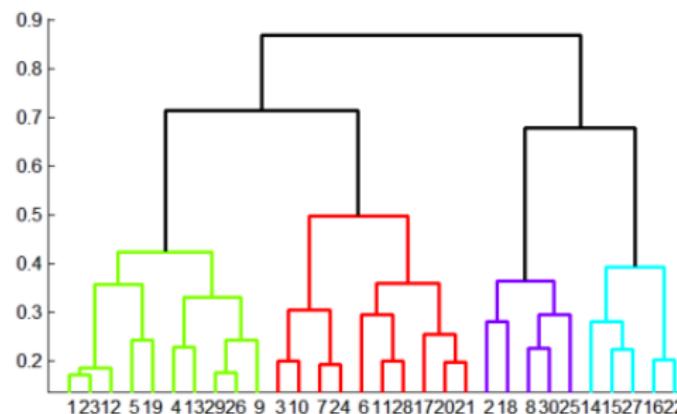
- **Agglomerative (bottom up):**
 - Initially, each point is a cluster;
 - Repeatedly combine the two “nearest” clusters into one.
- **Divisive (top down):**
 - Start with one cluster and recursively split it.



Key operation: Repeatedly combine two nearest clusters.

Three important questions:

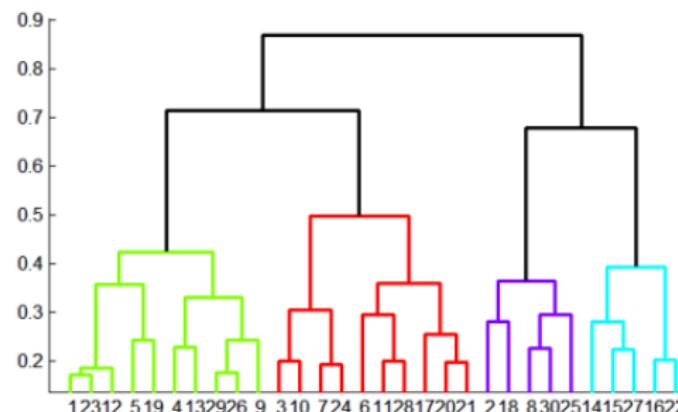
- ➊ How do you represent a cluster of more than one point?
- ➋ How do you determine the “nearness” of clusters?
- ➌ When to stop combining clusters?



Key operation: Repeatedly combine two nearest clusters.

Three important questions:

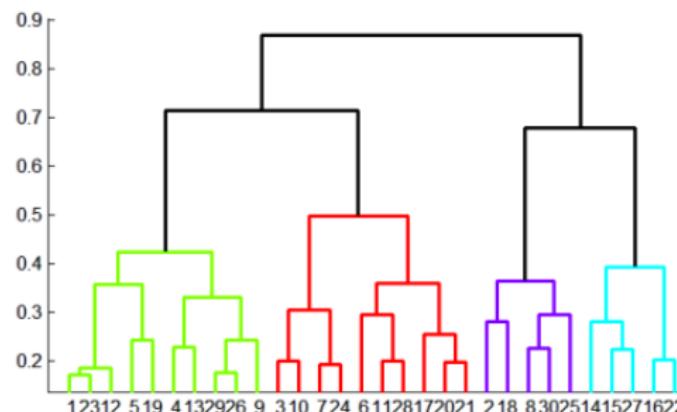
- ① How do you represent a cluster of more than one point?
- ② How do you determine the “nearness” of clusters?
- ③ When to stop combining clusters?



Key operation: Repeatedly combine two nearest clusters.

Three important questions:

- ① How do you represent a cluster of more than one point?
- ② How do you determine the “nearness” of clusters?
- ③ When to stop combining clusters?



Key operation: Repeatedly combine two nearest clusters

- ➊ How do you represent a cluster of more than one point?
 - **Key problem:** As you merge clusters, how do you represent the “location” of each cluster, to tell which pair of clusters is closest?
 - **Euclidean case:** each cluster has a centroid = average of its points
- ➋ How do you determine the “nearness” of clusters?
 - Measure cluster distances by distances of centroids

Key operation: Repeatedly combine two nearest clusters

- ➊ How do you represent a cluster of more than one point?
 - **Key problem:** As you merge clusters, how do you represent the “location” of each cluster, to tell which pair of clusters is closest?
 - **Euclidean case:** each cluster has a centroid = average of its points
- ➋ How do you determine the “nearness” of clusters?
 - Measure cluster distances by distances of centroids

Minkowski family of distances:

$$D(x, y) = \sqrt[p]{|x_1 - y_1|^p + |x_2 - y_2|^p + \cdots + |x_n - y_n|^p}.$$

It can be checked that for any $p \geq 1$:

- $D(x, y) \geq 0$,
- $D(x, x) = 0$,
- $D(x, y) = D(y, x)$,
- $D(x, y) \leq D(x, z) + D(z, y)$.

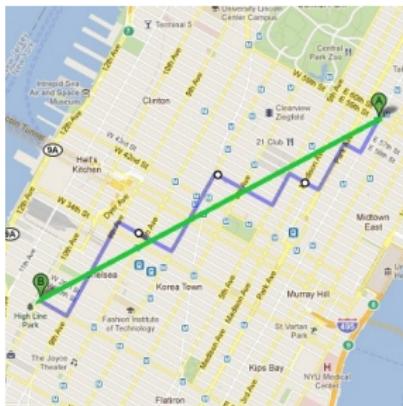
Minkowski family of distances:

$$D(x, y) = \sqrt[p]{|x_1 - y_1|^p + |x_2 - y_2|^p + \cdots + |x_n - y_n|^p}.$$

In case of $p = 1$:

$$D(x, y) = |x_1 - y_1| + |x_2 - y_2| + \cdots + |x_n - y_n|.$$

It is nicknamed **Manhattan distance** (blue):



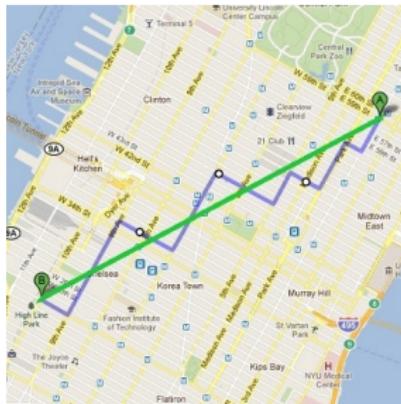
Minkowski family of distances:

$$D(x, y) = \sqrt[p]{|x_1 - y_1|^p + |x_2 - y_2|^p + \cdots + |x_n - y_n|^p}.$$

In case of $p = 2$:

$$\sqrt{|x_1 - y_1|^2 + |x_2 - y_2|^2 + \cdots + |x_n - y_n|^2}.$$

Euclidean distance (green):



1 Overview

2 Hierarchical clustering

3 K-means

4 DBSCAN

5 Cluster validity

6 Mixture Models

7 Community detection

- Spectral clustering
- Community detection problems
- Modularity
- BigCLAM

- Randomly **initialize** k centers:

$$\mu^0 = (\mu_1^0, \dots, \mu_k^0).$$

- **Classify:** Assign each point $j \in \{1, \dots, m\}$ to nearest center:

$$z^j = \arg \min_i \|\mathbf{x}_j - \mu_i^t\|_2^2.$$

- **Recenter:** μ_i becomes centroid of its points:

$$\mu_i^{t+1} = \arg \min_{\mu} \sum_{j:z^j=i} \|\mathbf{x}_j - \mu\|_2^2.$$

- Randomly **initialize** k centers:

$$\mu^0 = (\mu_1^0, \dots, \mu_k^0).$$

- **Classify:** Assign each point $j \in \{1, \dots, m\}$ to nearest center:

$$z^j = \arg \min_i \| \mathbf{x}_j - \mu_i^t \|_2^2.$$

- **Recenter:** μ_i becomes centroid of its points:

$$\mu_i^{t+1} = \arg \min_{\mu} \sum_{j: z^j = i} \| \mathbf{x}_j - \mu \|_2^2.$$

- Randomly **initialize** k centers:

$$\mu^0 = (\mu_1^0, \dots, \mu_k^0).$$

- **Classify:** Assign each point $j \in \{1, \dots, m\}$ to nearest center:

$$z^j = \arg \min_i \| \mathbf{x}_j - \mu_i^t \|_2^2.$$

- **Recenter:** μ_i becomes centroid of its points:

$$\mu_i^{t+1} = \arg \min_{\mu} \sum_{j: z^j = i} \| \mathbf{x}_j - \mu \|_2^2.$$

- Randomly **initialize** k centers:

$$\mu^0 = (\mu_1^0, \dots, \mu_k^0).$$

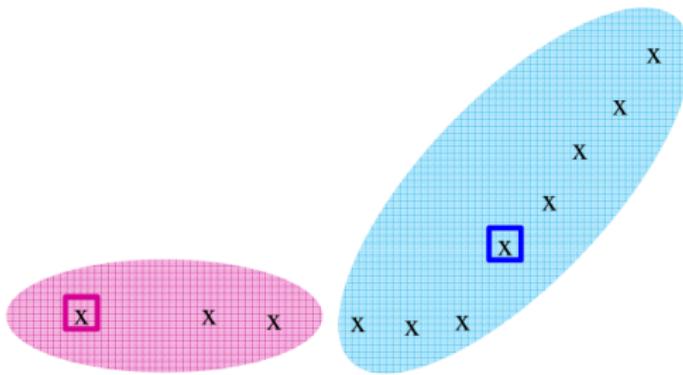
- **Classify:** Assign each point $j \in \{1, \dots, m\}$ to nearest center:

$$z^j = \arg \min_i \| \mathbf{x}_j - \mu_i^t \|_2^2.$$

- **Recenter:** μ_i becomes centroid of its points:

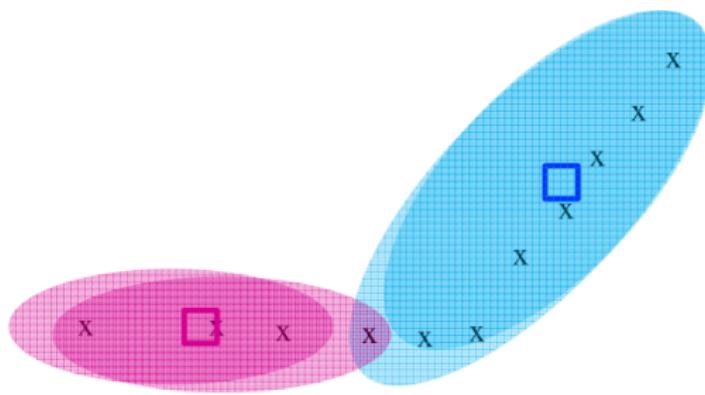
$$\mu_i^{t+1} = \arg \min_{\mu} \sum_{j: z^j = i} \| \mathbf{x}_j - \mu \|_2^2.$$

Equivalent to μ_i average of its points!



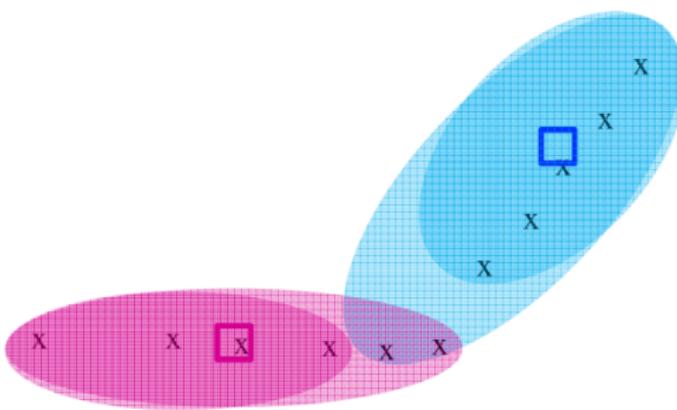
X ... data point
◻ ... centroid

Clusters after round 1



X ... data point
□ ... centroid

Clusters after round 2



X ... data point
□ ... centroid

Clusters at the end

Assumptions:

- Known number of clusters;
- Clusters of approximately the same size and density;
- Spherical form.

GMM improvements:

- Gaussian mixture models is an extension of K-means for ellipsoidal clusters and Gaussian mixture likelihood to be optimized.
- The optimization algorithm presented above is a variant of the famous EM algorithm.

1 Overview

2 Hierarchical clustering

3 K-means

4 DBSCAN

5 Cluster validity

6 Mixture Models

7 Community detection

- Spectral clustering
- Community detection problems
- Modularity
- BigCLAM

Non-parametric density-based method based on distances between points

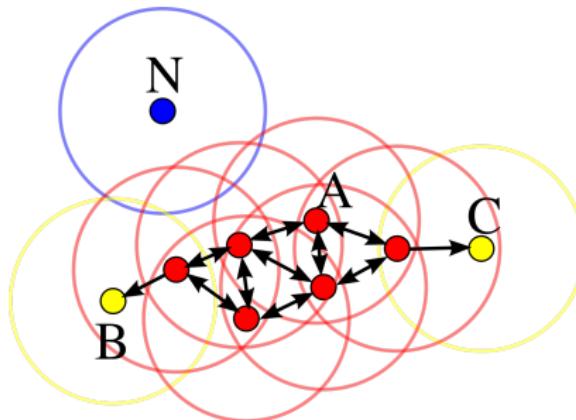
DBSCAN



k-means



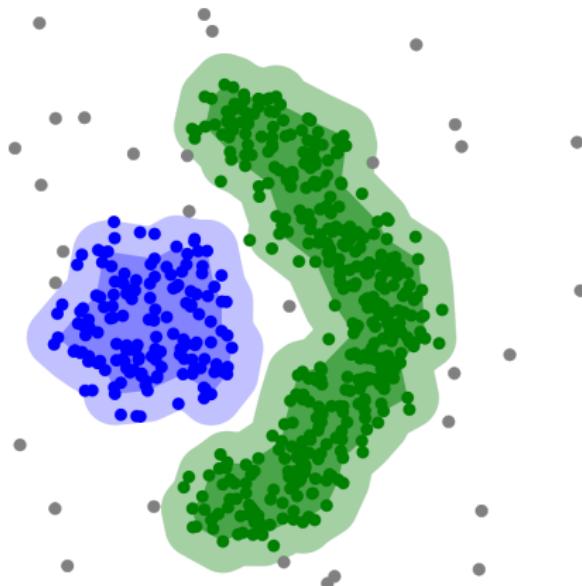
We are looking for core points that have at least k points within ε distance. All other points are either related to core points or outliers.



Hyperparameters:

- Distance function
- Radius
- Minimum number of nearest neighbours for the core points

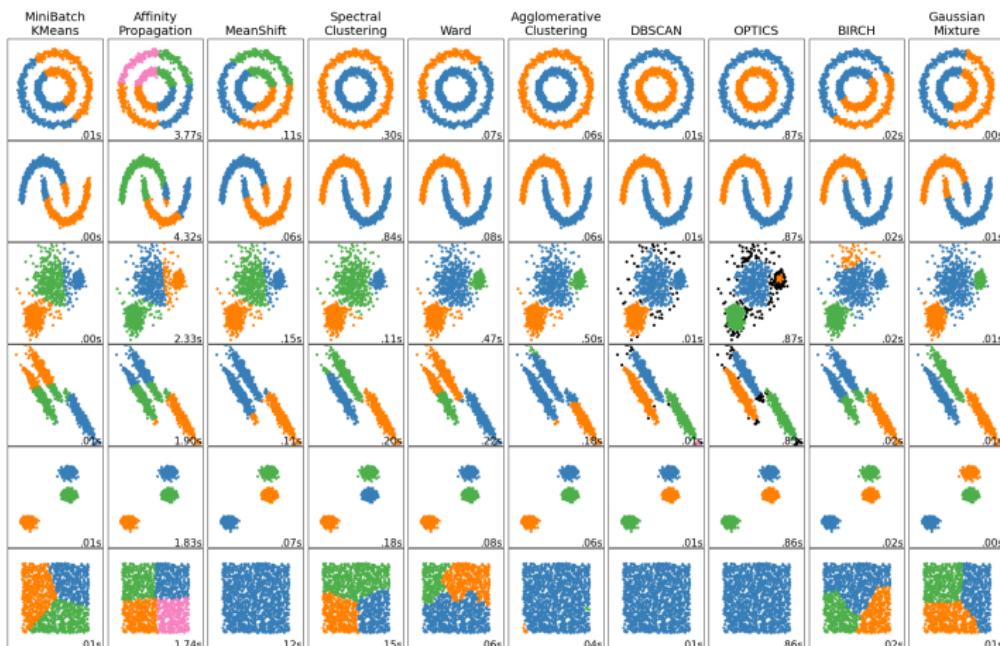
1. Find the core points. The core points are the points with at least k neighbors.
2. Find the connected components of core points. We ignore all other points. The components are clusters.
3. Assign each non-core point to a nearby cluster. Otherwise, it is an outlier.



Such clusterization is impossible for K-means or GMM.

How to select the best approach?

There are many clustering algorithms, because we are unsure on what we want as the output



1 Overview

2 Hierarchical clustering

3 K-means

4 DBSCAN

5 Cluster validity

6 Mixture Models

7 Community detection

- Spectral clustering
- Community detection problems
- Modularity
- BigCLAM

- For supervised classification we have a variety of measures to evaluate how good our model is: accuracy, precision, recall.
- For cluster analysis, the analogous question is how to evaluate the goodness of the resulting clusters?
- We need measures to compare:
 - clustering algorithms;
 - two sets of clusters.

- For supervised classification we have a variety of measures to evaluate how good our model is: accuracy, precision, recall.
- For cluster analysis, the analogous question is how to evaluate the goodness of the resulting clusters?
- We need measures to compare:
 - clustering algorithms;
 - two sets of clusters.

- For supervised classification we have a variety of measures to evaluate how good our model is: accuracy, precision, recall.
- For cluster analysis, the analogous question is how to evaluate the goodness of the resulting clusters?
- We need measures to compare:
 - clustering algorithms;
 - two sets of clusters.

- **External measure:** used to measure the extent to which cluster labels match externally supplied class labels.
 - Example: entropy
- **Internal measure:** used to measure the goodness of a clustering structure without respect to external information.
 - Example: Sum of Squared Errors (SSE)

- **External measure:** used to measure the extent to which cluster labels match externally supplied class labels.
 - Example: entropy
- **Internal measure:** used to measure the goodness of a clustering structure without respect to external information.
 - Example: Sum of Squared Errors (SSE)

- Partition obtained: $P = \{P_1, \dots, P_K\}$;
- External (true) partition: $C = \{C_1, \dots, C_K\}$;
- m_{ij} is a number of objects in $P_i \cap C_j$;
- $m_{i\cdot} = \sum_{j=1}^K m_{ij}$.

- Partition obtained: $P = \{P_1, \dots, P_K\}$;
- External (true) partition: $C = \{C_1, \dots, C_K\}$;
- m_{ij} is a number of objects in $P_i \cap C_j$;
- $m_{i\cdot} = \sum_{j=1}^K m_{ij}$.

- Partition obtained: $P = \{P_1, \dots, P_K\}$;
- External (true) partition: $C = \{C_1, \dots, C_K\}$;
- m_{ij} is a number of objects in $P_i \cap C_j$;
- $m_{i \cdot} = \sum_{j=1}^K m_{ij}$.

- Partition obtained: $P = \{P_1, \dots, P_K\}$;
- External (true) partition: $C = \{C_1, \dots, C_K\}$;
- m_{ij} is a number of objects in $P_i \cap C_j$;
- $m_{i\cdot} = \sum_{j=1}^K m_{ij}$.

- Distribution of objects in P_i :

$$\frac{m_{i1}}{m_{i\cdot}}, \frac{m_{i2}}{m_{i\cdot}}, \dots, \frac{m_{in}}{m_{i\cdot}}.$$

- Entropy (impurity) of the distribution is

$$\sum_{j=1}^K \frac{m_{ij}}{m_{i\cdot}} \log \frac{m_{ij}}{m_{i\cdot}}.$$

- Overall entropy:

$$E = - \sum_{i=1}^K \frac{m_{i\cdot}}{m} \sum_{j=1}^K \frac{m_{ij}}{m_{i\cdot}} \log \frac{m_{ij}}{m_{i\cdot}}.$$

- The lower the entropy, the better the clustering.

- Distribution of objects in P_i :

$$\frac{m_{i1}}{m_{i\cdot}}, \frac{m_{i2}}{m_{i\cdot}}, \dots, \frac{m_{in}}{m_{i\cdot}}.$$

- Entropy (impurity) of the distribution is

$$\sum_{j=1}^K \frac{m_{ij}}{m_{i\cdot}} \log \frac{m_{ij}}{m_{i\cdot}}.$$

- Overall entropy:

$$E = - \sum_{i=1}^K \frac{m_{i\cdot}}{m} \sum_{j=1}^K \frac{m_{ij}}{m_{i\cdot}} \log \frac{m_{ij}}{m_{i\cdot}}.$$

- The lower the entropy, the better the clustering.

- Distribution of objects in P_i :

$$\frac{m_{i1}}{m_{i\cdot}}, \frac{m_{i2}}{m_{i\cdot}}, \dots, \frac{m_{in}}{m_{i\cdot}}.$$

- Entropy (impurity) of the distribution is

$$\sum_{j=1}^K \frac{m_{ij}}{m_{i\cdot}} \log \frac{m_{ij}}{m_{i\cdot}}.$$

- Overall entropy:

$$E = - \sum_{i=1}^K \frac{m_{i\cdot}}{m} \sum_{j=1}^K \frac{m_{ij}}{m_{i\cdot}} \log \frac{m_{ij}}{m_{i\cdot}}.$$

- The lower the entropy, the better the clustering.

- Distribution of objects in P_i :

$$\frac{m_{i1}}{m_{i\cdot}}, \frac{m_{i2}}{m_{i\cdot}}, \dots, \frac{m_{in}}{m_{i\cdot}}.$$

- Entropy (impurity) of the distribution is

$$\sum_{j=1}^K \frac{m_{ij}}{m_{i\cdot}} \log \frac{m_{ij}}{m_{i\cdot}}.$$

- Overall entropy:

$$E = - \sum_{i=1}^K \frac{m_{i\cdot}}{m} \sum_{j=1}^K \frac{m_{ij}}{m_{i\cdot}} \log \frac{m_{ij}}{m_{i\cdot}}.$$

- The lower the entropy, the better the clustering.

- Given: to which P_i an object x belongs
 - This fact tells us something about the true class
 - The amount information is measured by

$$MI = \sum_{i,j=1}^K p_{ij} \log \frac{p_{ij}}{p_i p_j},$$

where $p_{ij} = \frac{m_{ij}}{m}$, $p_i = \frac{m_{i\cdot}}{m}$, $p_j = \frac{m_{\cdot j}}{m}$.

- Higher mutual information implies a higher clustering quality

- Given: to which P_i an object x belongs
- This fact tells us something about the true class
- The amount information is measured by

$$MI = \sum_{i,j=1}^K p_{ij} \log \frac{p_{ij}}{p_i p_j},$$

where $p_{ij} = \frac{m_{ij}}{m}$, $p_i = \frac{m_{i\cdot}}{m}$, $p_j = \frac{m_{\cdot j}}{m}$.

- Higher mutual information implies a higher clustering quality

- Given: to which P_i an object x belongs
- This fact tells us something about the true class
- The amount information is measured by

$$MI = \sum_{i,j=1}^K p_{ij} \log \frac{p_{ij}}{p_i p_j},$$

where $p_{ij} = \frac{m_{ij}}{m}$, $p_i = \frac{m_{i\cdot}}{m}$, $p_j = \frac{m_{\cdot j}}{m}$.

- Higher mutual information implies a higher clustering quality

- Given: to which P_i an object x belongs
- This fact tells us something about the true class
- The amount information is measured by

$$MI = \sum_{i,j=1}^K p_{ij} \log \frac{p_{ij}}{p_i p_j},$$

where $p_{ij} = \frac{m_{ij}}{m}$, $p_i = \frac{m_{i\cdot}}{m}$, $p_j = \frac{m_{\cdot j}}{m}$.

- Higher mutual information implies a higher clustering quality

$$J = \frac{a}{a + b + c},$$

where

- a is the number of pairs of points with the same label in C and assigned to the same cluster in P ;
- b is the number of pairs with the same label, but in different clusters;
- c is the number of pairs in the same cluster, but with different class labels.

$$J = \frac{a}{a + b + c},$$

where

- a is the number of pairs of points with the same label in C and assigned to the same cluster in P ;
- b is the number of pairs with the same label, but in different clusters;
- c is the number of pairs in the same cluster, but with different class labels.

$$J = \frac{a}{a + b + c},$$

where

- a is the number of pairs of points with the same label in C and assigned to the same cluster in P ;
- b is the number of pairs with the same label, but in different clusters;
- c is the number of pairs in the same cluster, but with different class labels.

$$J = \frac{a}{a + b + c},$$

where

- a is the number of pairs of points with the same label in C and assigned to the same cluster in P ;
- b is the number of pairs with the same label, but in different clusters;
- c is the number of pairs in the same cluster, but with different class labels.

The index produces a result in the range $[0, 1]$, where a value of 1.0 indicates that C and P are identical

$$RI = \frac{a + d}{a + b + c + d},$$

where

- a, b, c are as above
- d denotes the number of pairs with a different label in C that were assigned to a different cluster in P
- The index produces a result in the range $[0, 1]$, where a value of 1.0 indicates that C and P are identical.
- A high value for this measure generally indicates a high level of agreement between a clustering and the true classes.

$$RI = \frac{a + d}{a + b + c + d},$$

where

- a, b, c are as above
- d denotes the number of pairs with a different label in C that were assigned to a different cluster in P
- The index produces a result in the range $[0, 1]$, where a value of 1.0 indicates that C and P are identical.
- A high value for this measure generally indicates a high level of agreement between a clustering and the true classes.

$$RI = \frac{a + d}{a + b + c + d},$$

where

- a, b, c are as above
- d denotes the number of pairs with a different label in C that were assigned to a different cluster in P
- The index produces a result in the range $[0, 1]$, where a value of 1.0 indicates that C and P are identical.
- A high value for this measure generally indicates a high level of agreement between a clustering and the true classes.

$$RI = \frac{a + d}{a + b + c + d},$$

where

- a, b, c are as above
- d denotes the number of pairs with a different label in C that were assigned to a different cluster in P
- The index produces a result in the range $[0, 1]$, where a value of 1.0 indicates that C and P are identical.
- A high value for this measure generally indicates a high level of agreement between a clustering and the true classes.

Consider an i -th individual point

- $a(i)$ = average distance of the i -th point to the points in its cluster.
- $b(i)$ = min (average) distance of the i -th point to points in other clusters.
- The silhouette coefficient for the point is then given by

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}.$$

- Property: $-1 \leq s(i) \leq 1$.

Consider an i -th individual point

- $a(i)$ = average distance of the i -th point to the points in its cluster.
- $b(i)$ = min (average) distance of the i -th point to points in other clusters.
- The silhouette coefficient for the point is then given by

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}.$$

- Property: $-1 \leq s(i) \leq 1$.

Consider an i -th individual point

- $a(i)$ = average distance of the i -th point to the points in its cluster.
- $b(i)$ = min (average) distance of the i -th point to points in other clusters.
- The silhouette coefficient for the point is then given by

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}.$$

- Property: $-1 \leq s(i) \leq 1$.

Consider an i -th individual point

- $a(i)$ = average distance of the i -th point to the points in its cluster.
- $b(i)$ = min (average) distance of the i -th point to points in other clusters.
- The silhouette coefficient for the point is then given by

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}.$$

- Property: $-1 \leq s(i) \leq 1$.

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

- If $s(i)$ is close to 1, sample i is well-clustered and it was assigned to a very appropriate cluster
- If $s(i)$ is close to zero, sample i could be assigned to another closest cluster as well, and the sample lies equally far away from both clusters
- If $s(i)$ is close to -1 , sample i is misclassified.
- We can consider average $\frac{\sum_i s(i)}{m}$ of $s(i)$ for all objects in the whole dataset:
 - the larger it is, the better the clustering is

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

- If $s(i)$ is close to 1, sample i is well-clustered and it was assigned to a very appropriate cluster
- If $s(i)$ is close to zero, sample i could be assigned to another closest cluster as well, and the sample lies equally far away from both clusters
- If $s(i)$ is close to -1 , sample i is misclassified.
- We can consider average $\frac{\sum_i s(i)}{m}$ of $s(i)$ for all objects in the whole dataset:
 - the larger it is, the better the clustering is

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

- If $s(i)$ is close to 1, sample i is well-clustered and it was assigned to a very appropriate cluster
- If $s(i)$ is close to zero, sample i could be assigned to another closest cluster as well, and the sample lies equally far away from both clusters
- If $s(i)$ is close to -1 , sample i is misclassified.
- We can consider average $\frac{\sum_i s(i)}{m}$ of $s(i)$ for all objects in the whole dataset:
 - the larger it is, the better the clustering is

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

- If $s(i)$ is close to 1, sample i is well-clustered and it was assigned to a very appropriate cluster
- If $s(i)$ is close to zero, sample i could be assigned to another closest cluster as well, and the sample lies equally far away from both clusters
- If $s(i)$ is close to -1 , sample i is misclassified.
- We can consider average $\frac{\sum_i s(i)}{m}$ of $s(i)$ for all objects in the whole dataset:
 - the larger it is, the better the clustering is

1 Overview

2 Hierarchical clustering

3 K-means

4 DBSCAN

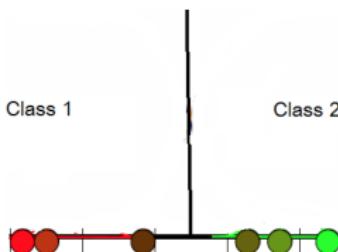
5 Cluster validity

6 Mixture Models

7 Community detection

- Spectral clustering
- Community detection problems
- Modularity
- BigCLAM

- K-means performs hard assignment
 - Each data point is assigned to one and only one cluster
 - Appropriate for points close to cluster centroids
 - Not appropriate for points midway between the two cluster centroids



- K-means performs hard assignment
- Each data point is assigned to one and only one cluster
- Appropriate for points close to cluster centroids
- Not appropriate for points midway between the two cluster centroids



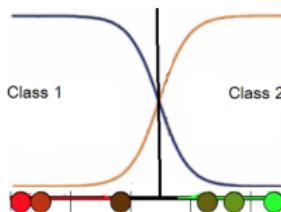
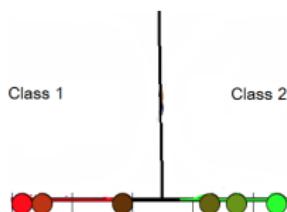
- K-means performs hard assignment
- Each data point is assigned to one and only one cluster
- Appropriate for points close to cluster centroids
- Not appropriate for points midway between the two cluster centroids



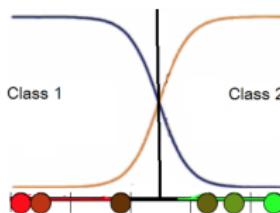
- K-means performs hard assignment
- Each data point is assigned to one and only one cluster
- Appropriate for points close to cluster centroids
- Not appropriate for points midway between the two cluster centroids



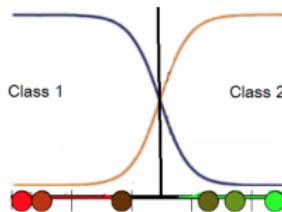
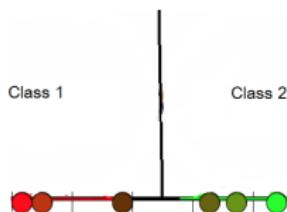
- Each data point is (partially) assigned to clusters with certain probabilities
- One point might be (partially) assigned to multiple clusters
- The midway point is assigned to either cluster with probability 0.5

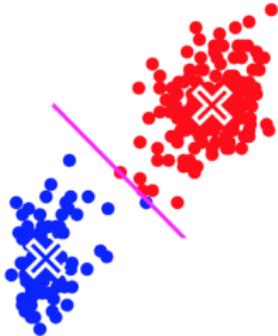


- Each data point is (partially) assigned to clusters with certain probabilities
- One point might be (partially) assigned to multiple clusters
- The midway point is assigned to either cluster with probability 0.5

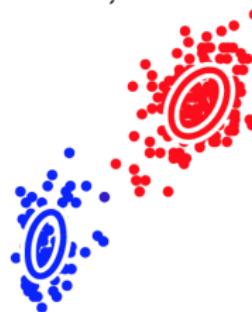


- Each data point is (partially) assigned to clusters with certain probabilities
- One point might be (partially) assigned to multiple clusters
- The midway point is assigned to either cluster with probability 0.5

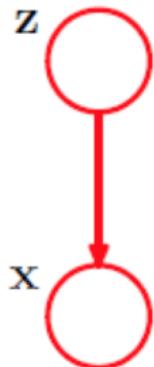




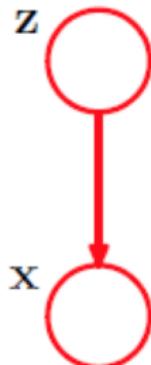
Hard Assignment



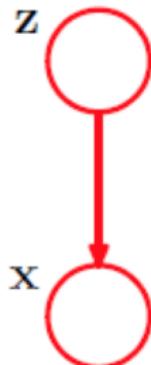
Soft Assignment
(Ellipses: contour of probability functions).



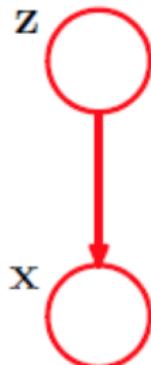
- K clusters: $1, 2, \dots, K$
- Randomly drawn object
 - \mathbf{x} : attribute values of the object, observed.
 - z : class of the object, latent variable (not observed)
- $P(z)$: distribution of z
 - $\pi_k = P(z = k)$: probability that the object is from class k
- $p(\mathbf{x}|z)$: conditional distribution of attribute values
 - $p(\mathbf{x}|z = k)$: distribution for objects from class k



- K clusters: $1, 2, \dots, K$
- Randomly drawn object
 - \mathbf{x} : attribute values of the object, observed.
 - z : class of the object, latent variable (not observed)
- $P(z)$: distribution of z
 - $\pi_k = P(z = k)$: probability that the object is from class k
- $p(\mathbf{x}|z)$: conditional distribution of attribute values
 - $p(\mathbf{x}|z = k)$: distribution for objects from class k



- K clusters: $1, 2, \dots, K$
- Randomly drawn object
 - \mathbf{x} : attribute values of the object, observed.
 - z : class of the object, latent variable (not observed)
- $P(z)$: distribution of z
 - $\pi_k = P(z = k)$: probability that the object is from class k
- $p(\mathbf{x}|z)$: conditional distribution of attribute values
 - $p(\mathbf{x}|z = k)$: distribution for objects from class k



- K clusters: $1, 2, \dots, K$
- Randomly drawn object
 - \mathbf{x} : attribute values of the object, observed.
 - z : class of the object, latent variable (not observed)
- $P(z)$: distribution of z
 - $\pi_k = P(z = k)$: probability that the object is from class k
- $p(\mathbf{x}|z)$: conditional distribution of attribute values
 - $p(\mathbf{x}|z = k)$: distribution for objects from class k

- Distribution of data:

$$p(\mathbf{x}) = \sum_{k=1}^K P(z=k)p(\mathbf{x}|z=k) = \sum_{k=1}^K \pi_k p(\mathbf{x}|z=k)$$

- It is a mixture of the distributions for individual classes
- Each $p(\mathbf{x}|z=k)$ is a component in the mixture
- Gaussian mixtures: each component is a Gaussian distribution

- Distribution of data:

$$p(\mathbf{x}) = \sum_{k=1}^K P(z=k)p(\mathbf{x}|z=k) = \sum_{k=1}^K \pi_k p(\mathbf{x}|z=k)$$

- It is a mixture of the distributions for individual classes
- Each $p(\mathbf{x}|z=k)$ is a component in the mixture
- Gaussian mixtures: each component is a Gaussian distribution

- Distribution of data:

$$p(\mathbf{x}) = \sum_{k=1}^K P(z=k)p(\mathbf{x}|z=k) = \sum_{k=1}^K \pi_k p(\mathbf{x}|z=k)$$

- It is a mixture of the distributions for individual classes
- Each $p(\mathbf{x}|z=k)$ is a component in the mixture
- Gaussian mixtures: each component is a Gaussian distribution

- Distribution of data:

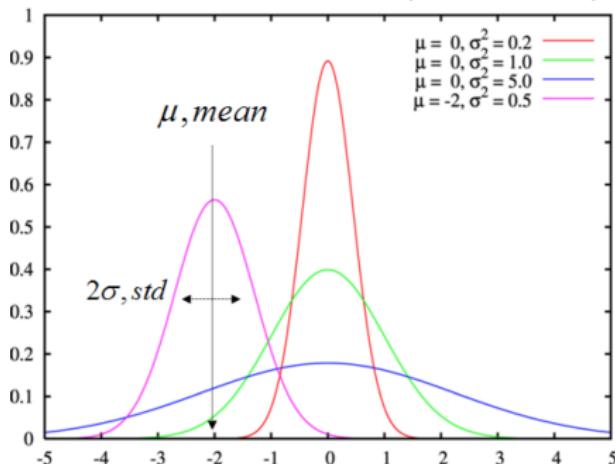
$$p(\mathbf{x}) = \sum_{k=1}^K P(z=k)p(\mathbf{x}|z=k) = \sum_{k=1}^K \pi_k p(\mathbf{x}|z=k)$$

- It is a mixture of the distributions for individual classes
- Each $p(\mathbf{x}|z=k)$ is a component in the mixture
- Gaussian mixtures: each component is a Gaussian distribution

- Gaussian distribution: $\mathcal{N}(x|\mu, \sigma)$

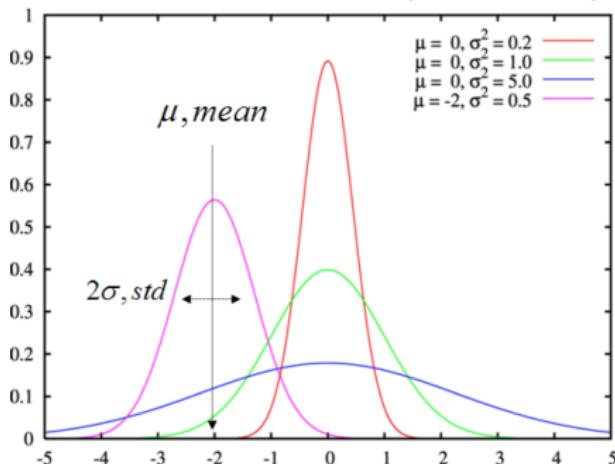
- Probability density function:

$$p(x|\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{(x-\mu)^2}{2\sigma^2}\right]$$



- Gaussian distribution: $\mathcal{N}(x|\mu, \sigma)$
- Probability density function:

$$p(\mathbf{x}|\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{(x - \mu)^2}{2\sigma^2}\right]$$

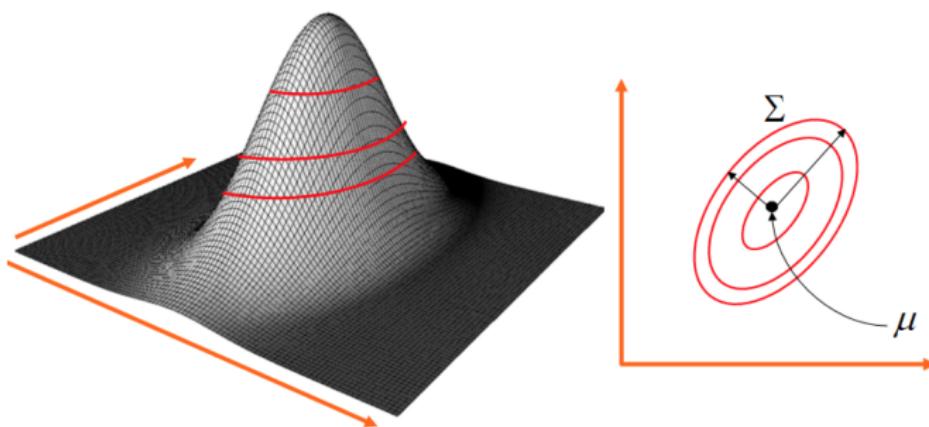


$$p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^d \det(\boldsymbol{\Sigma})}} \exp \left[-\frac{(\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1} (\mathbf{x} - \boldsymbol{\mu})}{2} \right],$$

where

- d : dimension;
- \mathbf{x} : vector of d random variables, representing data;
- $\boldsymbol{\mu}$: vector of means;
- $\boldsymbol{\Sigma}$: covariance matrix.

- μ : center of contour lines
- Σ : orientation and size of contour lines

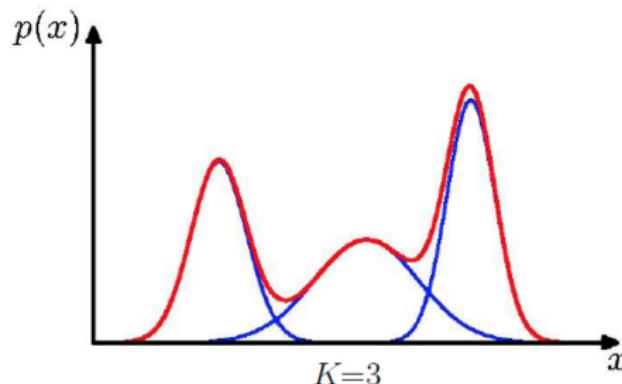


- Mixture distribution:

$$p(\mathbf{x}) = \sum_{k=1}^K \pi_k p(\mathbf{x}|z=k).$$

- Each component is a Gaussian distribution:

$$p(\mathbf{x}|z=k) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k).$$

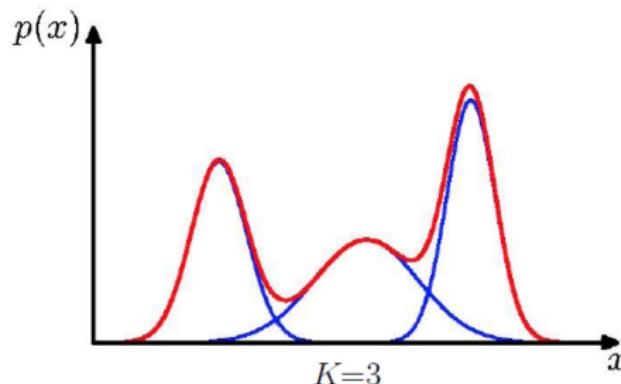


- Mixture distribution:

$$p(\mathbf{x}) = \sum_{k=1}^K \pi_k p(\mathbf{x}|z=k).$$

- Each component is a Gaussian distribution:

$$p(\mathbf{x}|z=k) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k).$$

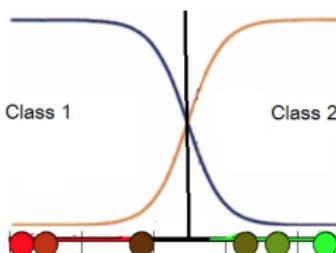


- Given mixture model:

$$p(\mathbf{x}) = \sum_{k=1}^K P(z=k)p(\mathbf{x}|z=k) = \sum_{k=1}^K \pi_k p(\mathbf{x}|z=k)$$

- Object \mathbf{x} belongs to class k with probability

$$P(z=k|\mathbf{x}) = \frac{P(z=k)p(\mathbf{x}|z=k)}{p(\mathbf{x})} = \frac{\pi_k p(\mathbf{x}|z=k)}{\sum_{s=1}^K \pi_s p(\mathbf{x}|z=s)}$$

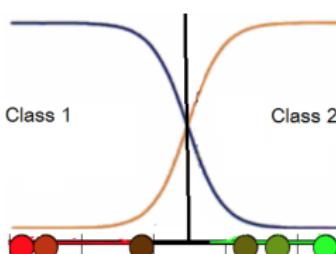


- Given mixture model:

$$p(\mathbf{x}) = \sum_{k=1}^K P(z=k)p(\mathbf{x}|z=k) = \sum_{k=1}^K \pi_k p(\mathbf{x}|z=k)$$

- Object \mathbf{x} belongs to class k with probability

$$P(z=k|\mathbf{x}) = \frac{P(z=k)p(\mathbf{x}|z=k)}{p(\mathbf{x})} = \frac{\pi_k p(\mathbf{x}|z=k)}{\sum_{s=1}^K \pi_s p(\mathbf{x}|z=s)}$$



- Given
 - unlabeled data $\{\mathbf{x}_1, \dots, \mathbf{x}_m\}$;
 - number of clusters K
- Find a K -component Gaussian mixture model:
 - mixing coefficients $\{\pi_1, \dots, \pi_K\}$;
 - components parameters $\{(\mu_k, \Sigma_k)\}_{k=1}^K$by maximizing data likelihood

- Given
 - unlabeled data $\{\mathbf{x}_1, \dots, \mathbf{x}_m\}$;
 - number of clusters K
- Find a K -component Gaussian mixture model:
 - mixing coefficients $\{\pi_1, \dots, \pi_K\}$;
 - components parameters $\{(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)\}_{k=1}^K$by maximizing data likelihood

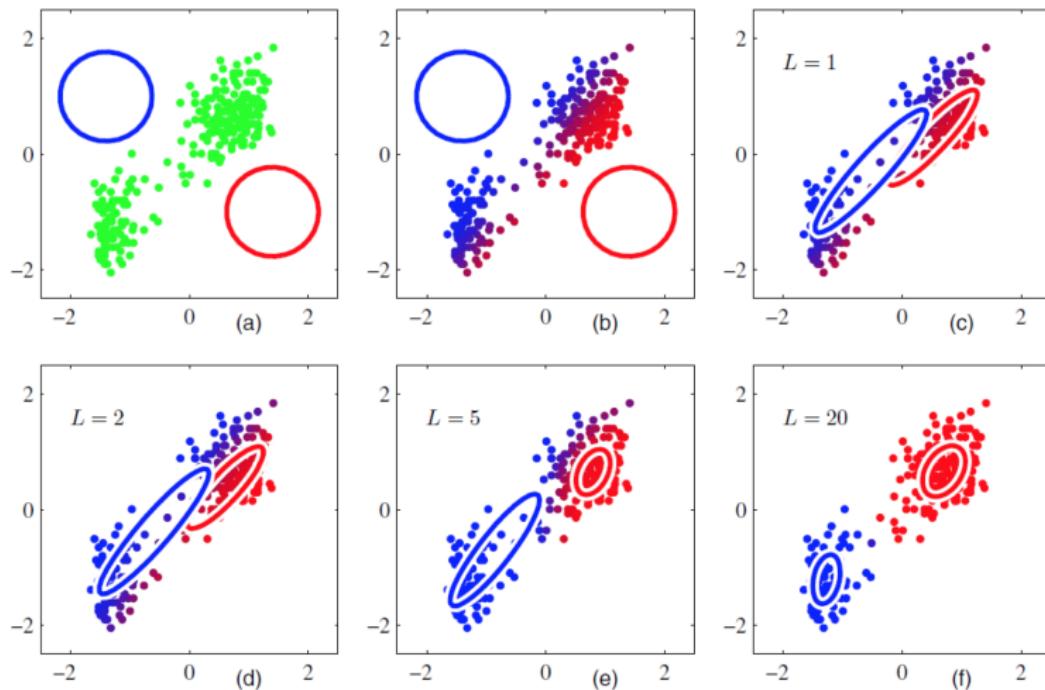
- Choose initial values for $\pi_k, \mu_k, \Sigma_k, k = 1, \dots, K$
- repeat:
 - Expectation: for each training example x_n
 - (a) Compute $r_{nk} = P(z = k|x_n), k = 1, \dots, K$
 - (b) Break data into K parts according to the probabilities
$$x_n[r_{nk}], k = 1, \dots, K$$
 - (c) Assign each part $x_n[r_{nk}]$ to the corresponding cluster k
 - Maximization: Re-estimate $\pi_k, \mu_k, \Sigma_k, k = 1, \dots, K$
- Until convergence

- Choose initial values for $\pi_k, \mu_k, \Sigma_k, k = 1, \dots, K$
- repeat:
 - Expectation: for each training example \mathbf{x}_n
 - (a) Compute $r_{nk} = P(z = k | \mathbf{x}_n), k = 1, \dots, K$
 - (b) Break data into K parts according to the probabilities
$$\mathbf{x}_n[r_{nk}], k = 1, \dots, K$$
 - (c) Assign each part $\mathbf{x}_n[r_{nk}]$ to the corresponding cluster k
 - Maximization: Re-estimate $\pi_k, \mu_k, \Sigma_k, k = 1, \dots, K$
- Until convergence

- Choose initial values for $\pi_k, \mu_k, \Sigma_k, k = 1, \dots, K$
- repeat:
 - Expectation: for each training example \mathbf{x}_n
 - (a) Compute $r_{nk} = P(z = k | \mathbf{x}_n), k = 1, \dots, K$
 - (b) Break data into K parts according to the probabilities
$$\mathbf{x}_n[r_{nk}], k = 1, \dots, K$$
 - (c) Assign each part $\mathbf{x}_n[r_{nk}]$ to the corresponding cluster k
 - Maximization: Re-estimate $\pi_k, \mu_k, \Sigma_k, k = 1, \dots, K$
- Until convergence

- Choose initial values for $\pi_k, \mu_k, \Sigma_k, k = 1, \dots, K$
- repeat:
 - Expectation: for each training example \mathbf{x}_n
 - (a) Compute $r_{nk} = P(z = k | \mathbf{x}_n), k = 1, \dots, K$
 - (b) Break data into K parts according to the probabilities
$$\mathbf{x}_n[r_{nk}], k = 1, \dots, K$$
 - (c) Assign each part $\mathbf{x}_n[r_{nk}]$ to the corresponding cluster k
 - Maximization: Re-estimate $\pi_k, \mu_k, \Sigma_k, k = 1, \dots, K$
- Until convergence

- Choose initial values for $\pi_k, \mu_k, \Sigma_k, k = 1, \dots, K$
- repeat:
 - Expectation: for each training example \mathbf{x}_n
 - (a) Compute $r_{nk} = P(z = k | \mathbf{x}_n), k = 1, \dots, K$
 - (b) Break data into K parts according to the probabilities
$$\mathbf{x}_n[r_{nk}], k = 1, \dots, K$$
 - (c) Assign each part $\mathbf{x}_n[r_{nk}]$ to the corresponding cluster k
 - Maximization: Re-estimate $\pi_k, \mu_k, \Sigma_k, k = 1, \dots, K$
- Until convergence



- Main computations are in step (a):

$$\begin{aligned} r_{nk} &= P(z = k | \mathbf{x}_n) \\ &= \frac{P(z = k)p(\mathbf{x}_n | z = k)}{\sum_{k=1}^K P(z = k)p(\mathbf{x}_n | z = k)} \\ &= \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{s=1}^K \pi_s \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_s, \boldsymbol{\Sigma}_s)}. \end{aligned}$$

- r_{nk} are often called responsibilities.

- Main computations are in step (a):

$$\begin{aligned} r_{nk} &= P(z = k | \mathbf{x}_n) \\ &= \frac{P(z = k)p(\mathbf{x}_n | z = k)}{\sum_{k=1}^K P(z = k)p(\mathbf{x}_n | z = k)} \\ &= \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{s=1}^K \pi_s \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_s, \boldsymbol{\Sigma}_s)}. \end{aligned}$$

- r_{nk} are often called responsibilities.

- Total number of examples, assigned to cluster k :

$$m_k = \sum_{i=1}^m r_{ki}$$

- Re-estimate π_k, μ_k, Σ_k :

$$\pi_k^{new} = \frac{m_k}{m},$$

$$\mu_k^{new} = \frac{1}{m_k} \sum_{i=1}^m r_{ki} \mathbf{x}_i,$$

$$\Sigma_k^{new} = \frac{1}{m_k} \sum_{i=1}^m r_{ki} (\mathbf{x}_i - \mu_k^{new})(\mathbf{x}_i - \mu_k^{new})^T$$

- Total number of examples, assigned to cluster k :

$$m_k = \sum_{i=1}^m r_{ki}$$

- Re-estimate π_k, μ_k, Σ_k :

$$\pi_k^{new} = \frac{m_k}{m},$$

$$\mu_k^{new} = \frac{1}{m_k} \sum_{i=1}^m r_{ki} \mathbf{x}_i,$$

$$\Sigma_k^{new} = \frac{1}{m_k} \sum_{i=1}^m r_{ki} (\mathbf{x}_i - \mu_k^{new})(\mathbf{x}_i - \mu_k^{new})^T$$

- Choose initial values for π_k , μ_k , Σ_k .
- Repeat until convergence
 - Expectation: for each training example x_n compute

$$r_{nk} = \frac{\pi_k \mathcal{N}(x_n | \mu_k, \Sigma_k)}{\sum_{s=1}^K \pi_s \mathcal{N}(x_n | \mu_s, \Sigma_s)}, k = 1, 2, \dots, K.$$

- Maximization: Re-estimate π_k , μ_k , Σ_k :

$$\pi_k^{new} = \frac{m_k}{m}, \quad \mu_k^{new} = \frac{1}{m_k} \sum_{i=1}^m r_{ki} x_i,$$

$$\Sigma_k^{new} = \frac{1}{m_k} \sum_{i=1}^m r_{ki} (x_i - \mu_k^{new})(x_i - \mu_k^{new})^T,$$

where $m_k = \sum_{i=1}^m r_{ki}$

- Choose initial values for π_k , μ_k , Σ_k .
- Repeat until convergence

— Expectation: for each training example \mathbf{x}_n compute

$$r_{nk} = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{s=1}^K \pi_s \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_s, \boldsymbol{\Sigma}_s)}, k = 1, 2, \dots, K.$$

— Maximization: Re-estimate π_k , μ_k , Σ_k :

$$\pi_k^{new} = \frac{m_k}{m}, \quad \boldsymbol{\mu}_k^{new} = \frac{1}{m_k} \sum_{i=1}^m r_{ki} \mathbf{x}_i,$$

$$\boldsymbol{\Sigma}_k^{new} = \frac{1}{m_k} \sum_{i=1}^m r_{ki} (\mathbf{x}_i - \boldsymbol{\mu}_k^{new})(\mathbf{x}_i - \boldsymbol{\mu}_k^{new})^T,$$

where $m_k = \sum_{i=1}^m r_{ki}$

- Choose initial values for π_k , μ_k , Σ_k .
- Repeat until convergence
 - Expectation: for each training example \mathbf{x}_n compute

$$r_{nk} = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{s=1}^K \pi_s \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_s, \boldsymbol{\Sigma}_s)}, \quad k = 1, 2, \dots, K.$$

- Maximization: Re-estimate π_k , μ_k , Σ_k :

$$\pi_k^{new} = \frac{m_k}{m}, \quad \boldsymbol{\mu}_k^{new} = \frac{1}{m_k} \sum_{i=1}^m r_{ki} \mathbf{x}_i,$$

$$\boldsymbol{\Sigma}_k^{new} = \frac{1}{m_k} \sum_{i=1}^m r_{ki} (\mathbf{x}_i - \boldsymbol{\mu}_k^{new})(\mathbf{x}_i - \boldsymbol{\mu}_k^{new})^T,$$

where $m_k = \sum_{i=1}^m r_{ki}$

- Choose initial values for π_k , $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$.
- Repeat until convergence
 - Expectation: for each training example \mathbf{x}_n compute

$$r_{nk} = \frac{\pi_k \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{s=1}^K \pi_s \mathcal{N}(\mathbf{x}_n | \boldsymbol{\mu}_s, \boldsymbol{\Sigma}_s)}, \quad k = 1, 2, \dots, K.$$

- Maximization: Re-estimate π_k , $\boldsymbol{\mu}_k$, $\boldsymbol{\Sigma}_k$:

$$\pi_k^{new} = \frac{m_k}{m}, \quad \boldsymbol{\mu}_k^{new} = \frac{1}{m_k} \sum_{i=1}^m r_{ki} \mathbf{x}_i,$$

$$\boldsymbol{\Sigma}_k^{new} = \frac{1}{m_k} \sum_{i=1}^m r_{ki} (\mathbf{x}_i - \boldsymbol{\mu}_k^{new})(\mathbf{x}_i - \boldsymbol{\mu}_k^{new})^T,$$

where $m_k = \sum_{i=1}^m r_{ki}$

- Let $\boldsymbol{\pi} = (\pi_1, \dots, \pi_K)$, $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_K)$, $\boldsymbol{\Sigma} = (\boldsymbol{\Sigma}_1, \dots, \boldsymbol{\Sigma}_K)$
- Specification of $\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$ defines a probability density functions over features \mathbf{x} :

$$p(\mathbf{x}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

- Data: $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_m\}$
- Log-Likelihood as function of model parameters:

$$l(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \log \prod_{i=1}^m p(\mathbf{x}_i|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{i=1}^m \log p(\mathbf{x}_i|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

- Let $\boldsymbol{\pi} = (\pi_1, \dots, \pi_K)$, $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_K)$, $\boldsymbol{\Sigma} = (\boldsymbol{\Sigma}_1, \dots, \boldsymbol{\Sigma}_K)$
- Specification of $\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$ defines a probability density functions over features \mathbf{x} :

$$p(\mathbf{x}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

- Data: $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_m\}$
- Log-Likelihood as function of model parameters:

$$l(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \log \prod_{i=1}^m p(\mathbf{x}_i|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{i=1}^m \log p(\mathbf{x}_i|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

- Let $\boldsymbol{\pi} = (\pi_1, \dots, \pi_K)$, $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_K)$, $\boldsymbol{\Sigma} = (\boldsymbol{\Sigma}_1, \dots, \boldsymbol{\Sigma}_K)$
- Specification of $\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$ defines a probability density functions over features \mathbf{x} :

$$p(\mathbf{x}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

- Data: $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_m\}$
- Log-Likelihood as function of model parameters:

$$l(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \log \prod_{i=1}^m p(\mathbf{x}_i|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{i=1}^m \log p(\mathbf{x}_i|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

- Let $\boldsymbol{\pi} = (\pi_1, \dots, \pi_K)$, $\boldsymbol{\mu} = (\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_K)$, $\boldsymbol{\Sigma} = (\boldsymbol{\Sigma}_1, \dots, \boldsymbol{\Sigma}_K)$
- Specification of $\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$ defines a probability density functions over features \mathbf{x} :

$$p(\mathbf{x}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{k=1}^K \pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

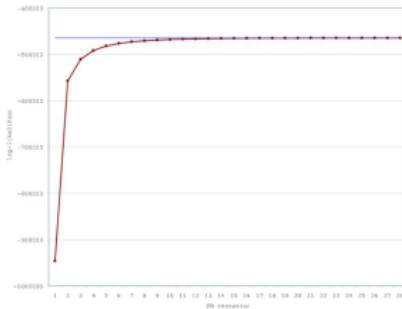
- Data: $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_m\}$
- Log-Likelihood as function of model parameters:

$$l(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \log \prod_{i=1}^m p(\mathbf{x}_i|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \sum_{i=1}^m \log p(\mathbf{x}_i|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

- EM aims at computing the maximum likelihood estimation (MLE) of the parameters

$$(\boldsymbol{\pi}^*, \boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*) = \arg \max_{\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}} l(\mathbf{X} | \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

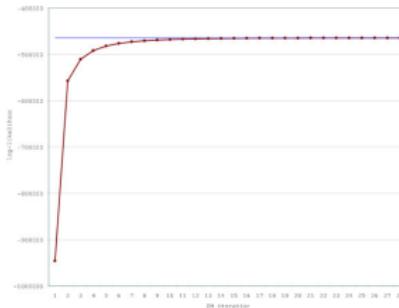
- Let $l(t)$ be the log-likelihood after iteration t
- The series $l(1), l(2), \dots$ increases monotonically with t
- Terminate EM when $l(t+1) - l(t)$ falls below a threshold



- EM aims at computing the maximum likelihood estimation (MLE) of the parameters

$$(\boldsymbol{\pi}^*, \boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*) = \arg \max_{\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}} l(\mathbf{X} | \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

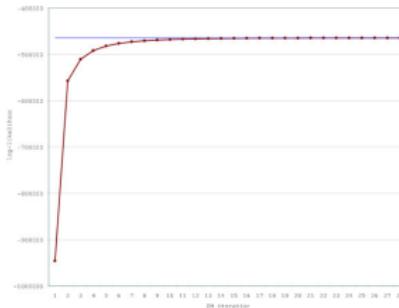
- Let $l(t)$ be the log-likelihood after iteration t
- The series $l(1), l(2), \dots$ increases monotonically with t
- Terminate EM when $l(t+1) - l(t)$ falls below a threshold



- EM aims at computing the maximum likelihood estimation (MLE) of the parameters

$$(\boldsymbol{\pi}^*, \boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*) = \arg \max_{\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}} l(\mathbf{X} | \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

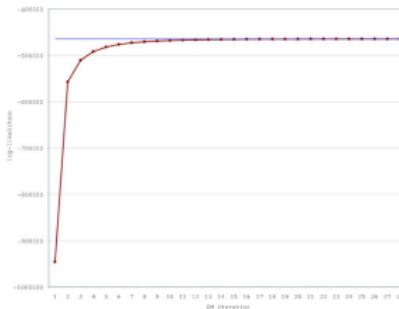
- Let $l(t)$ be the log-likelihood after iteration t
- The series $l(1), l(2), \dots$ increases monotonically with t
- Terminate EM when $l(t+1) - l(t)$ falls below a threshold



- EM aims at computing the maximum likelihood estimation (MLE) of the parameters

$$(\boldsymbol{\pi}^*, \boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*) = \arg \max_{\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}} l(\mathbf{X} | \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

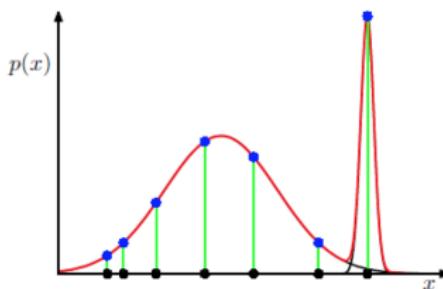
- Let $l(t)$ be the log-likelihood after iteration t
- The series $l(1), l(2), \dots$ increases monotonically with t
- Terminate EM when $l(t+1) - l(t)$ falls below a threshold



- The maximum value of log-likelihood might be infinite:

$$l(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \log \prod_{i=1}^m p(\mathbf{x}_i|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

- Such singularity in likelihood function happens often in case of outliers and repeated points

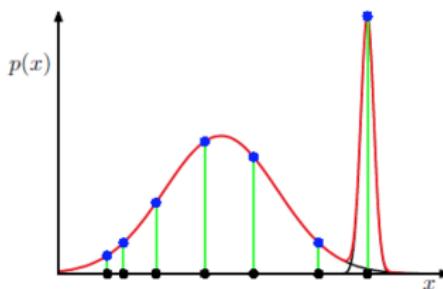


- Solution: Bound the eigenvalues of covariance matrix
- To avoid local maximum: multiple restart

- The maximum value of log-likelihood might be infinite:

$$l(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \log \prod_{i=1}^m p(\mathbf{x}_i|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

- Such singularity in likelihood function happens often in case of outliers and repeated points

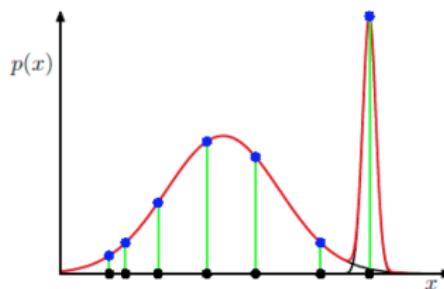


- Solution: Bound the eigenvalues of covariance matrix
- To avoid local maximum: multiple restart

- The maximum value of log-likelihood might be infinite:

$$l(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \log \prod_{i=1}^m p(\mathbf{x}_i|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

- Such singularity in likelihood function happens often in case of outliers and repeated points

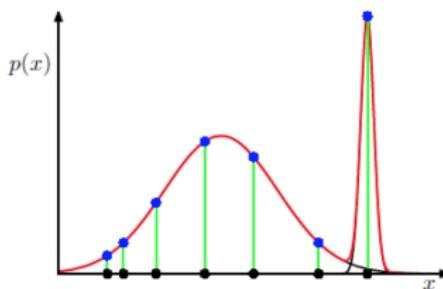


- Solution: Bound the eigenvalues of covariance matrix
- To avoid local maximum: multiple restart

- The maximum value of log-likelihood might be infinite:

$$l(\mathbf{X}|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \log \prod_{i=1}^m p(\mathbf{x}_i|\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

- Such singularity in likelihood function happens often in case of outliers and repeated points



- Solution: Bound the eigenvalues of covariance matrix
- To avoid local maximum: multiple restart

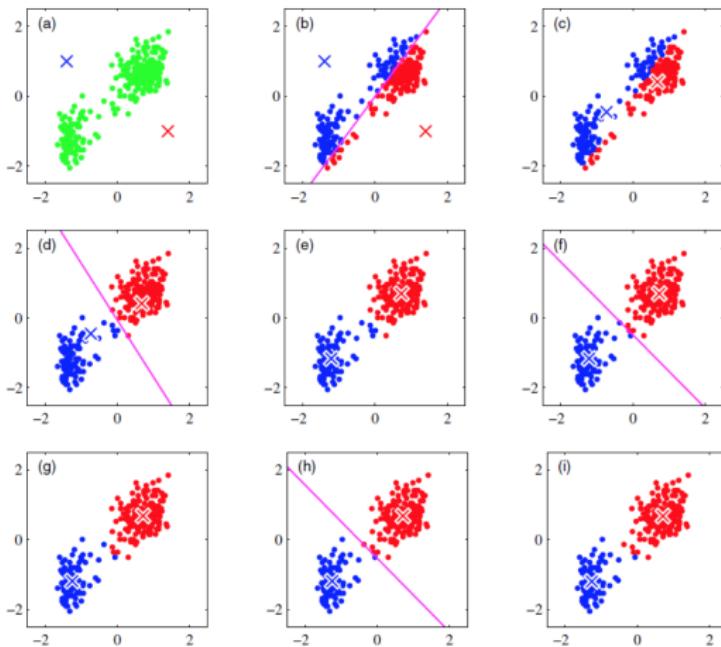
Hard Assignment:

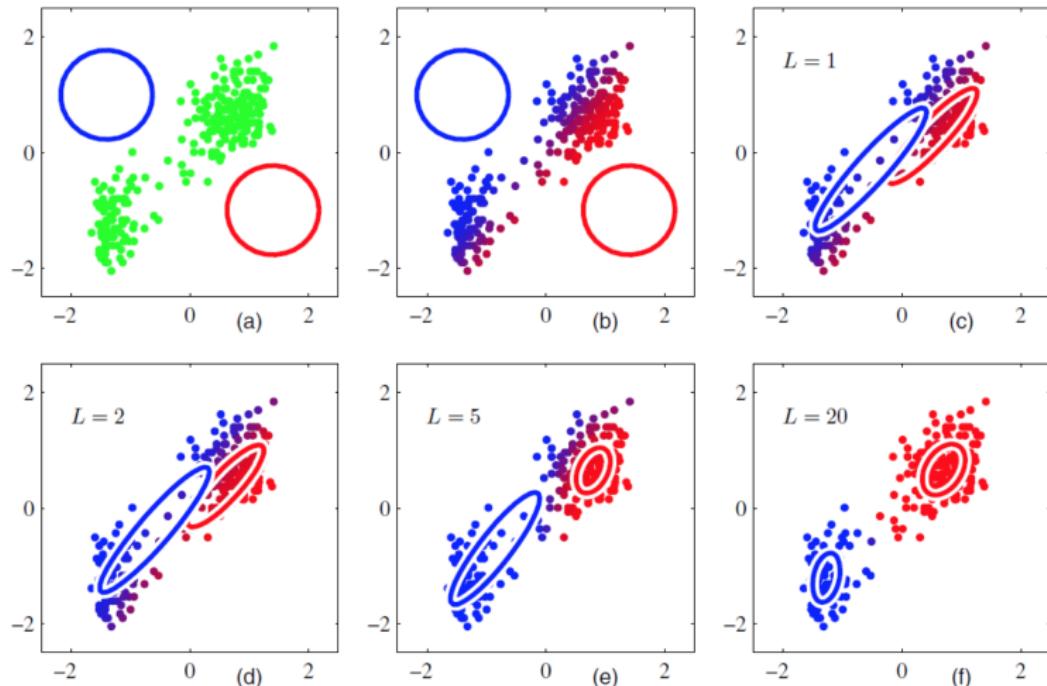
- Select K points as the initial centroids
- Repeat
 - for K clusters by assigning all points to the closest centroid;
 - recompute the centroid of each cluster
- until the centroids don't change

Soft Assignment:

- Choose initial values for π_k , μ_k , Σ_k
- Repeat
 - Expectation:
 - (a) Compute $r_{nk} = P(z = k | \mathbf{x}_n)$ for $k = 1, \dots, K$;
 - (b) Break it into K fractional examples according to the probabilities;
 - (c) Assign each fractional examples to the corresponding cluster k
 - Maximization: Re-estimate π_k , μ_k , Σ_k
- until convergence

K-means example





PROBABLY NO...

1 Overview

2 Hierarchical clustering

3 K-means

4 DBSCAN

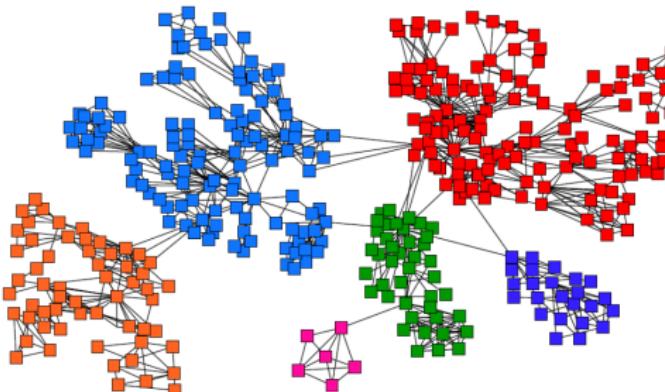
5 Cluster validity

6 Mixture Models

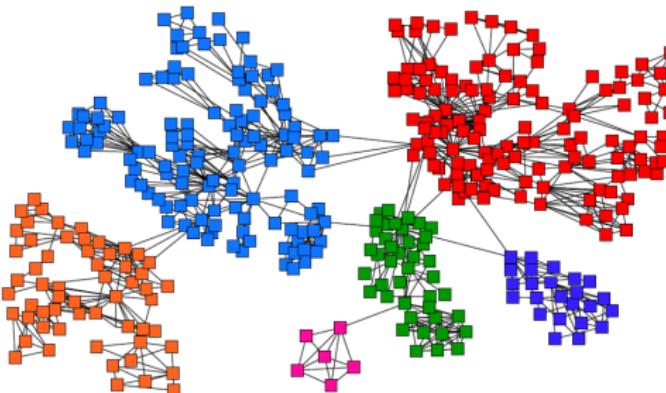
7 Community detection

- Spectral clustering
- Community detection problems
- Modularity
- BigCLAM

- **Graph $G(E, V)$:**
 - Nodes v_j
 - Edge weights $w_{ij} > 0$
- **Problem:** Want to partition graph such that edges between groups have low weights



- **Graph $G(E, V)$:**
 - Nodes v_j
 - Edge weights $w_{ij} > 0$
- **Problem:** Want to partition graph such that edges between groups have low weights



Types of graphs:

- **ϵ -neighborhood:**

- Only include edges with distances $< \epsilon$;
- Treat as unweighted: $w_{ij} = Const$

- **k-NN:**

- Connect v_i and v_j if v_j is a k-NN of v_i
- Weighted by similarity $w_{ij} = s_{ij}$
- Directed or undirected

- **Mutual k-NN:**

- Same as k-NN, but only include mutual k-NN

Types of graphs:

- **ϵ -neighborhood:**

- Only include edges with distances $< \epsilon$;
- Treat as unweighted: $w_{ij} = Const$

- **k-NN:**

- Connect v_i and v_j if v_j is a k-NN of v_i
- Weighted by similarity $w_{ij} = s_{ij}$
- Directed or undirected

- **Mutual k-NN:**

- Same as k-NN, but only include mutual k-NN

Types of graphs:

- **ϵ -neighborhood:**

- Only include edges with distances $< \epsilon$;
- Treat as unweighted: $w_{ij} = Const$

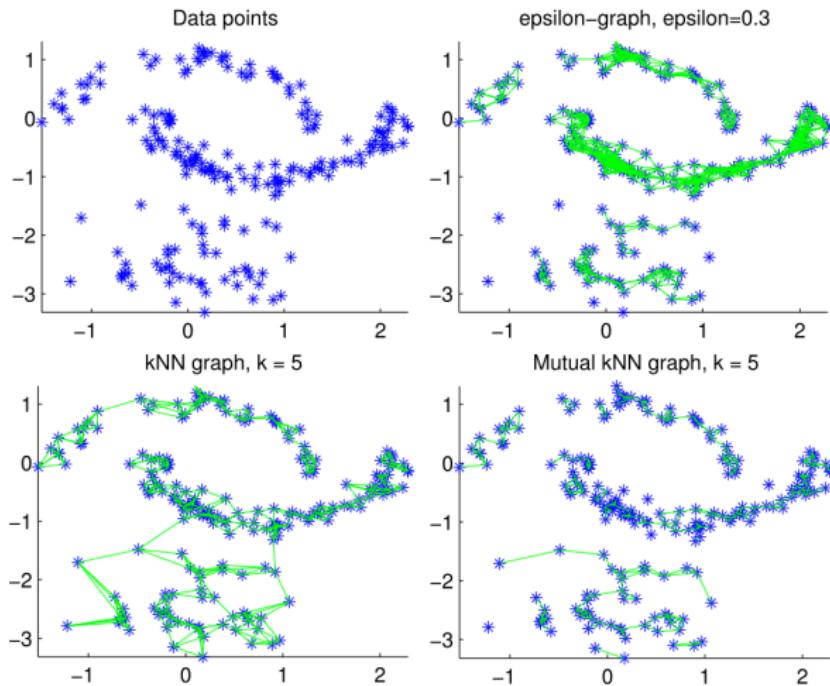
- **k-NN:**

- Connect v_i and v_j if v_j is a k-NN of v_i
- Weighted by similarity $w_{ij} = s_{ij}$
- Directed or undirected

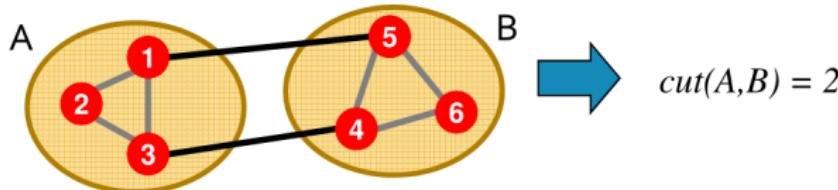
- **Mutual k-NN:**

- Same as k-NN, but only include mutual k-NN

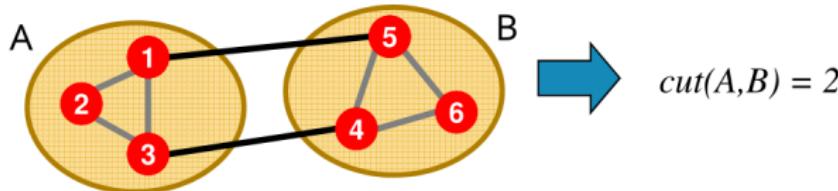
Similarity graphs



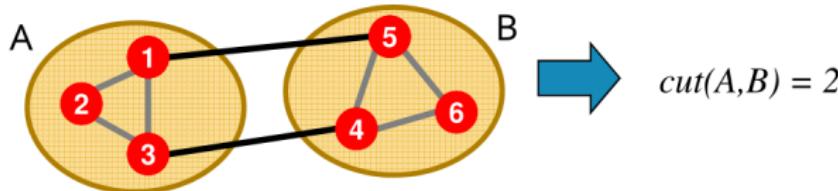
- **Problem:** Partition graph such that edges between groups have low weights
- **Define:** $W(A, B) = \sum_{i \in A, j \in B} w_{ij}$
- **MinCut problem:** $Cut(A_1, \dots, A_k) = \sum_{i=1}^k W(A_i, \bar{A}_i)$
- **Choose:** $A_1, \dots, A_k = \arg \min_{A_1, \dots, A_k} Cut(A_1, \dots, A_k)$



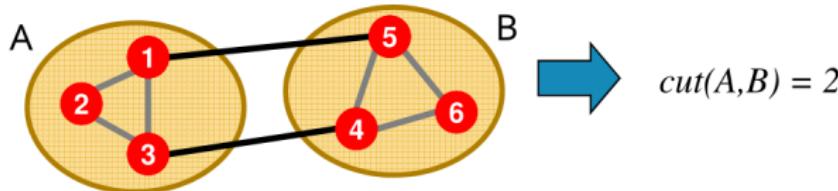
- **Problem:** Partition graph such that edges between groups have low weights
- **Define:** $W(A, B) = \sum_{i \in A, j \in B} w_{ij}$
- **MinCut problem:** $Cut(A_1, \dots, A_k) = \sum_{i=1}^k W(A_i, \overline{A}_i)$
- **Choose:** $A_1, \dots, A_k = \arg \min_{A_1, \dots, A_k} Cut(A_1, \dots, A_k)$



- **Problem:** Partition graph such that edges between groups have low weights
- **Define:** $W(A, B) = \sum_{i \in A, j \in B} w_{ij}$
- **MinCut problem:** $Cut(A_1, \dots, A_k) = \sum_{i=1}^k W(A_i, \overline{A}_i)$
- **Choose:** $A_1, \dots, A_k = \arg \min_{A_1, \dots, A_k} Cut(A_1, \dots, A_k)$



- **Problem:** Partition graph such that edges between groups have low weights
- **Define:** $W(A, B) = \sum_{i \in A, j \in B} w_{ij}$
- **MinCut problem:** $Cut(A_1, \dots, A_k) = \sum_{i=1}^k W(A_i, \overline{A}_i)$
- **Choose:** $A_1, \dots, A_k = \arg \min_{A_1, \dots, A_k} Cut(A_1, \dots, A_k)$



1 Overview

2 Hierarchical clustering

3 K-means

4 DBSCAN

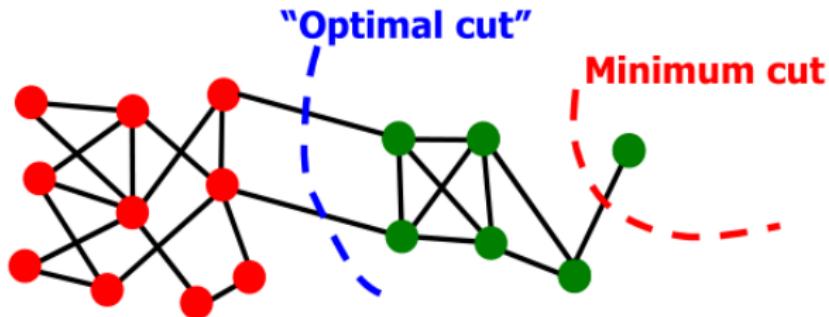
5 Cluster validity

6 Mixture Models

7 Community detection

- Spectral clustering
- Community detection problems
- Modularity
- BigCLAM

Problem: MinCut favors isolated clusters.



Solution:

- Ratio cuts (RatioCut);
- Normalized cuts (Ncut);
- Lead to “balanced” clusters.

Two measures of size of a subset:

- Cardinality:

$$|A| = \# \text{ of vertices in } A$$

- Volume:

$$vol(A) = \sum_{i \in A} \sum_{j=1}^N w_{ij}$$

Two measures of size of a subset:

- Cardinality:

$$|A| = \# \text{ of vertices in } A$$

- Volume:

$$vol(A) = \sum_{i \in A} \sum_{j=1}^N w_{ij}$$

- Ratio cuts (RatioCut)

- $k = 2$: $\text{RatioCut}(A, \bar{A}) = \text{Cut}(A, \bar{A}) \left(\frac{1}{|A|} + \frac{1}{|\bar{A}|} \right)$.

- General k : $\text{RatioCut}(A_1, \dots, A_k) = \frac{1}{2} \sum_{i=1}^k \frac{\text{Cut}(A_i, \bar{A}_i)}{|A_i|}$.

- Normalized cuts (Ncut)

- $k = 2$: $\text{NCut}(A, \bar{A}) = \text{Cut}(A, \bar{A}) \left(\frac{1}{\text{Vol}(A)} + \frac{1}{\text{Vol}(\bar{A})} \right)$.

- General k : $\text{NCut}(A_1, \dots, A_k) = \frac{1}{2} \sum_{i=1}^k \frac{\text{Cut}(A_i, \bar{A}_i)}{\text{Vol}(A_i)}$

- Problem is NP-hard!

- We need to look at relaxation (solution = Spectral clustering)

- Ratio cuts (RatioCut)

- $k = 2$: $\text{RatioCut}(A, \bar{A}) = \text{Cut}(A, \bar{A}) \left(\frac{1}{|A|} + \frac{1}{|\bar{A}|} \right).$

- General k : $\text{RatioCut}(A_1, \dots, A_k) = \frac{1}{2} \sum_{i=1}^k \frac{\text{Cut}(A_i, \bar{A}_i)}{|A_i|}.$

- Normalized cuts (Ncut)

- $k = 2$: $\text{NCut}(A, \bar{A}) = \text{Cut}(A, \bar{A}) \left(\frac{1}{\text{Vol}(A)} + \frac{1}{\text{Vol}(\bar{A})} \right).$

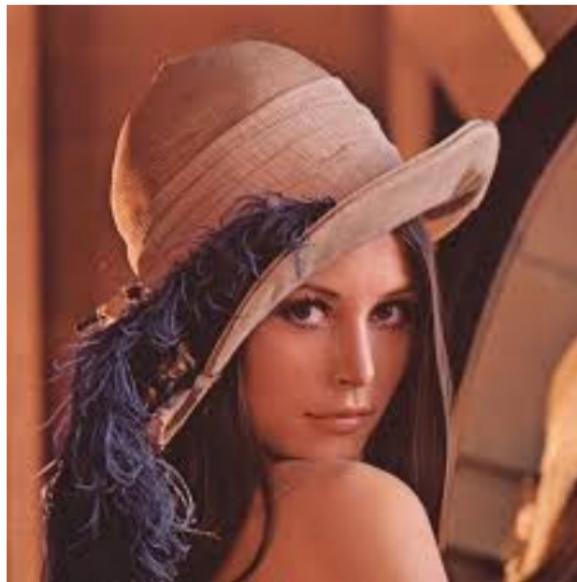
- General k : $\text{NCut}(A_1, \dots, A_k) = \frac{1}{2} \sum_{i=1}^k \frac{\text{Cut}(A_i, \bar{A}_i)}{\text{Vol}(A_i)}$

- Problem is NP-hard!

- We need to look at relaxation (solution = Spectral clustering)

- Ratio cuts (RatioCut)
 - $k = 2$: $\text{RatioCut}(A, \bar{A}) = \text{Cut}(A, \bar{A}) \left(\frac{1}{|A|} + \frac{1}{|\bar{A}|} \right)$.
 - General k : $\text{RatioCut}(A_1, \dots, A_k) = \frac{1}{2} \sum_{i=1}^k \frac{\text{Cut}(A_i, \bar{A}_i)}{|A_i|}$.
- Normalized cuts (Ncut)
 - $k = 2$: $\text{NCut}(A, \bar{A}) = \text{Cut}(A, \bar{A}) \left(\frac{1}{\text{Vol}(A)} + \frac{1}{\text{Vol}(\bar{A})} \right)$.
 - General k : $\text{NCut}(A_1, \dots, A_k) = \frac{1}{2} \sum_{i=1}^k \frac{\text{Cut}(A_i, \bar{A}_i)}{\text{Vol}(A_i)}$
- Problem is NP-hard!
- We need to look at relaxation (solution = Spectral clustering)

Segmentation of Lena



Spectral clustering: discretize, 28.62s



1 Overview

2 Hierarchical clustering

3 K-means

4 DBSCAN

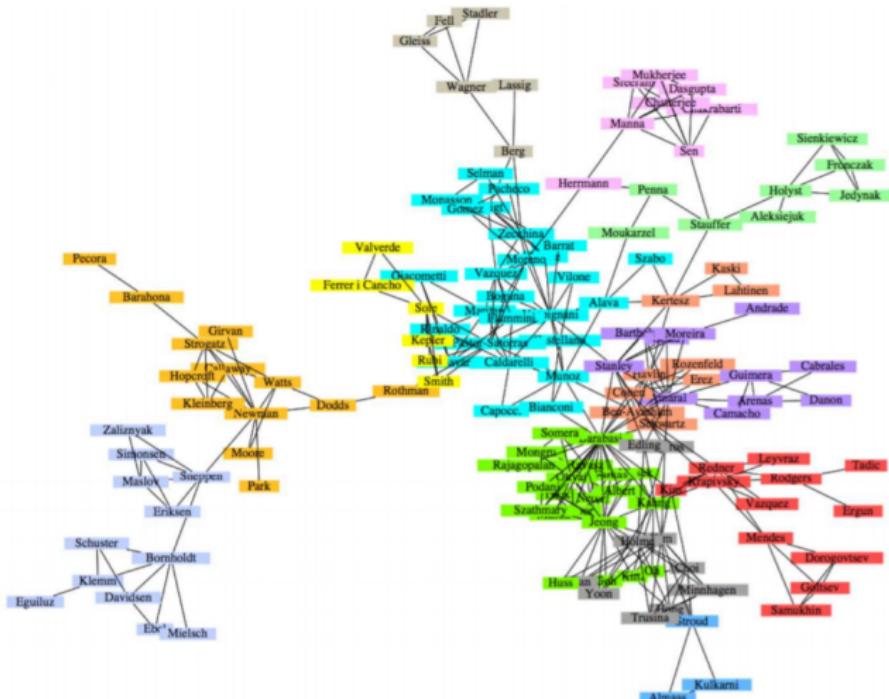
5 Cluster validity

6 Mixture Models

7 Community detection

- Spectral clustering
- Community detection problems
- Modularity
- BigCLAM

Citation network



Objects of study:

- social networks,
- citation/co-authorship networks,
- designing network protocols,
- biological networks,
- ...

1 Overview

2 Hierarchical clustering

3 K-means

4 DBSCAN

5 Cluster validity

6 Mixture Models

7 Community detection

- Spectral clustering
- Community detection problems
- Modularity
- BigCLAM

$$Q = \frac{1}{2m} \sum_{(i,j) \in E} \left(w_{ij} - \frac{d_i d_j}{2m} \right) \delta(C_i, C_j),$$

where

- d_i is a degree of node i ;
- C_i is a community of node i ;
- $\delta(C_i, C_j)$ is a delta function;
- $m = |E|$ is a total number of edges in a graph

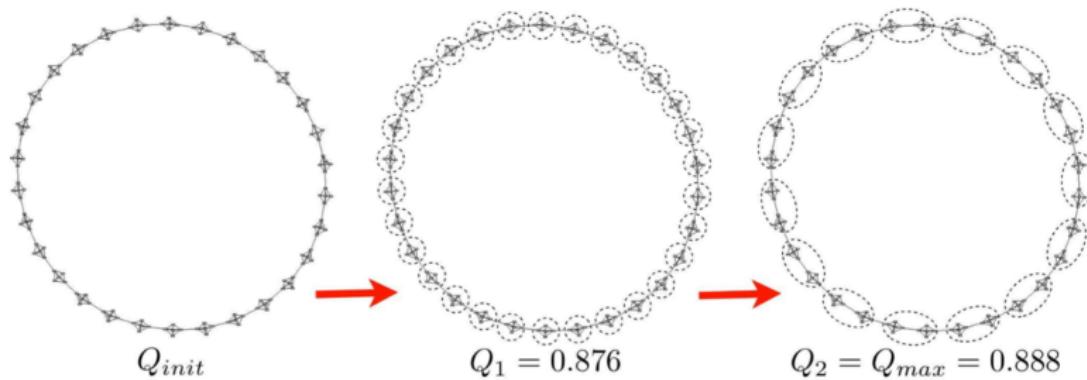
Interpretation: difference between the fraction of edges inside the community and its expectation in random graph with fixed node degrees.

Modularity:

$$Q = \frac{1}{2m} \sum_{(i,j) \in E} \left(w_{ij} - \frac{d_i d_j}{2m} \right) \delta(C_i, C_j),$$

- **Idea:** optimize modularity (discrete optimization problem).
- **Efficient implementation:** Louvain community detection algorithm.
- **Problem:** low resolution.

Low resolution of modularity



1 Overview

2 Hierarchical clustering

3 K-means

4 DBSCAN

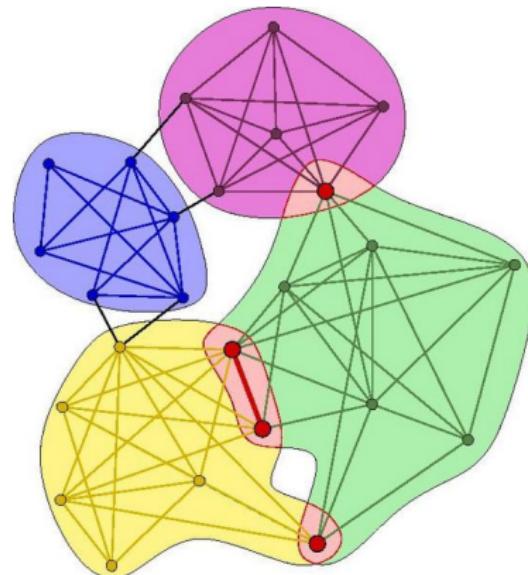
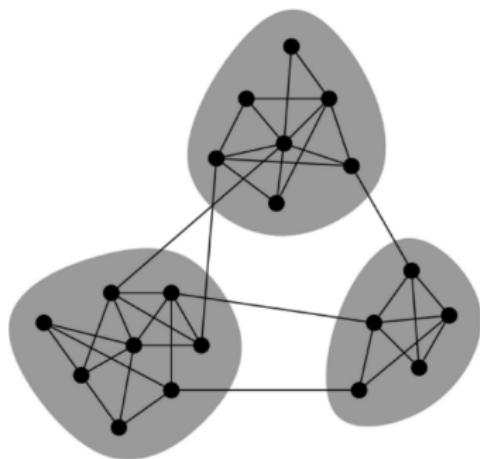
5 Cluster validity

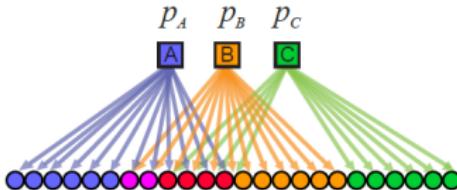
6 Mixture Models

7 Community detection

- Spectral clustering
- Community detection problems
- Modularity
- BigCLAM

Non-overlapping vs. overlapping communities





AGM generates the links:

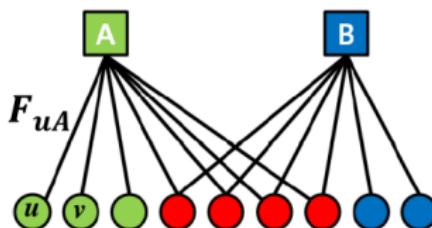
- For each pair of nodes in community A we connect them with probability p_A .
- The overall edge probability is:

$$P(u, v) = 1 - \prod_{c \in M_u \cap M_v} (1 - p_c),$$

where M_z is a set of communities node z belongs to.

- If u, v share no communities: $P(u, v) = \epsilon$.

Relaxation: Memberships have strengths



- F_{uA} : The membership strength of node u to community A ($F_{uA} = 0$: no membership).
- Each community A links nodes independently:

$$P_A(u, v) = 1 - \exp\{-F_{uA}F_{vA}\}.$$

- Community membership strength matrix: $F = (F_{uA}, u \in V, A \in \mathcal{C})$.
- Community A links nodes u, v independently:

$$P_A(u, v) = 1 - \exp\{-F_{uA} \cdot F_{vA}\}.$$

- Then probability at least one common C links them:

$$P(u, v) = 1 - \prod_{C \in \mathcal{C}} (1 - P_C(u, v)) = 1 - \exp\{-F_u F_v^T\}.$$

Task: Given a network $G(V, E)$, estimate F .

- Find F that maximizes the likelihood:

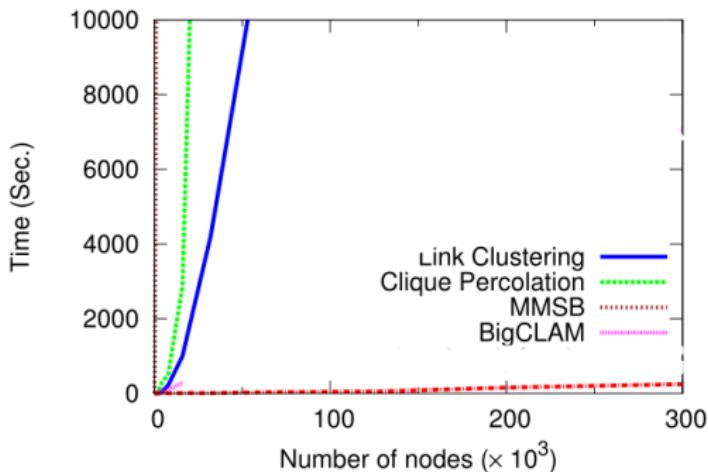
$$\arg \max_F \prod_{(u,v) \in E} P(u,v) \prod_{(u,v) \notin E} (1 - P(u,v)),$$

where $P(u,v) = 1 - \exp\{-F_u F_v^T\}$.

Goal: Find F that maximizes log-likelihood:

$$\ell(F) = \sum_{(u,v) \in E} \log(1 - \exp\{-F_u F_v^T\}) - \sum_{(u,v) \notin E} F_u F_v^T.$$

Note: Non-convex optimization problem!



- BigCLAM takes 5 minutes for 300k node nets (other methods take 10 days).
- Can process networks with 100M edges!