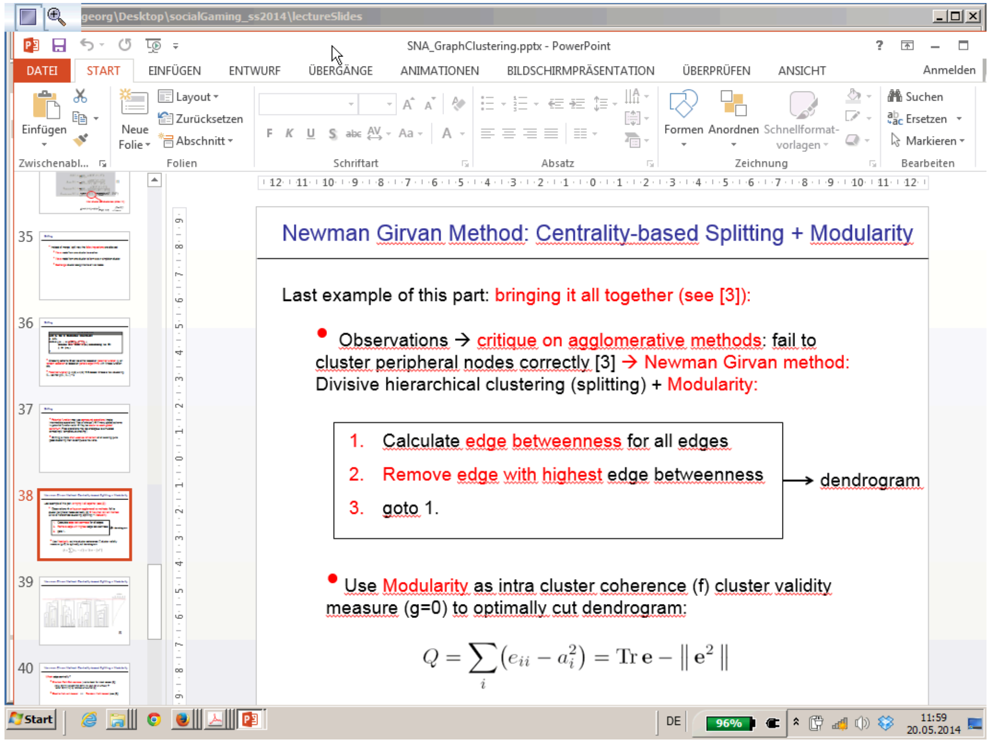
Script generated by TTT

Title: groh: profile1 (20.05.2014)

Date: Tue May 20 11:59:43 CEST 2014

Duration: 94:28 min

Pages: 59



Newman Girvan Method: Centrality-based Splitting + Modularity

Last example of this part: bringing it all together (see [3]):

- Observations → critique on agglomerative methods: fail to cluster peripheral nodes correctly [3] → Newman Girvan method: Divisive hierarchical clustering (splitting) + Modularity:

1. Calculate edge betweenness for all edges
2. Remove edge with highest edge betweenness
3. goto 1.

→ dendrogram

- Use Modularity as intra cluster coherence (f) cluster validity measure (g=0) to optimally cut dendrogram:

$$Q = \sum_i (e_{ii} - a_i^2) = \mathrm{Tr}\,\mathbf{e} - \|\mathbf{e}^2\|$$

Newman Girvan Method: Centrality-based Splitting + Modularity

Last example of this part: bringing it all together (see [3]):

- Observations → critique on agglomerative methods: fail to cluster peripheral nodes correctly [3] → Newman Girvan method: Divisive hierarchical clustering (splitting) + Modularity:

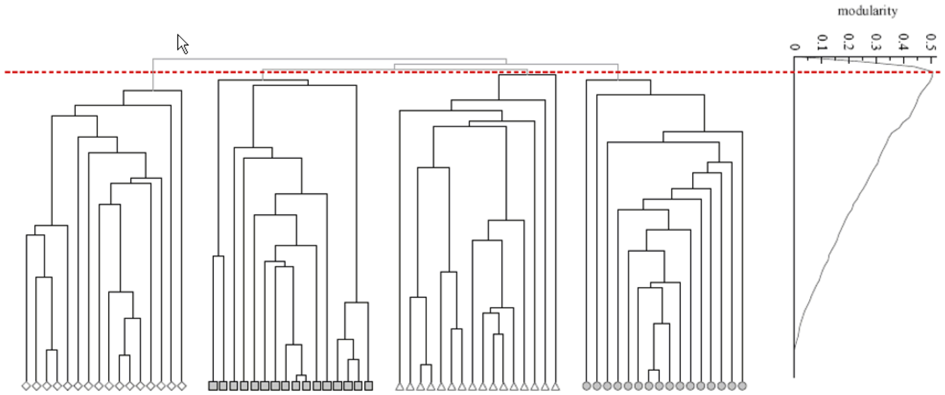
1. Calculate edge betweenness for all edges
2. Remove edge with highest edge betweenness
3. goto 1.

→ dendrogram

- Use Modularity as intra cluster coherence (f) cluster validity measure (g=0) to optimally cut dendrogram:

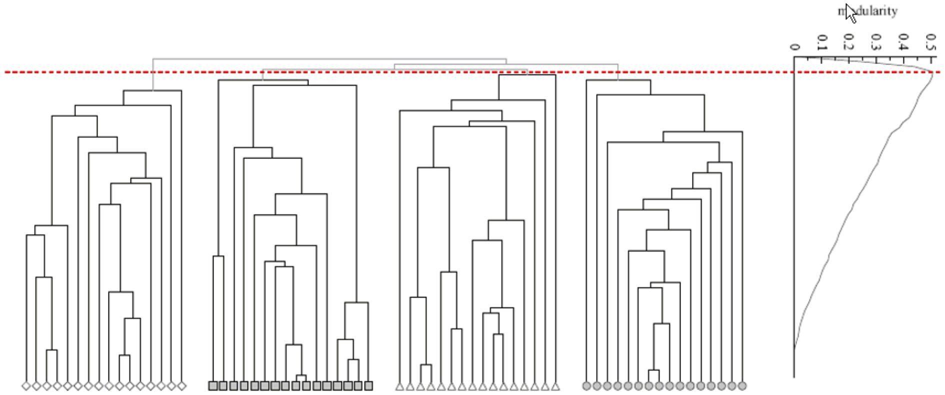
$$Q = \sum_i (e_{ii} - a_i^2) = \mathrm{Tr}\,\mathbf{e} - \|\mathbf{e}^2\|$$

Newman Girvan Method: Centrality-based Splitting + Modularity



[3]

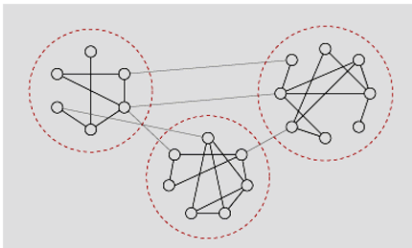
Newman Girvan Method: Centrality-based Splitting + Modularity



[3]

Newman Girvan Method: Centrality-based Splitting + Modularity

Modularity:

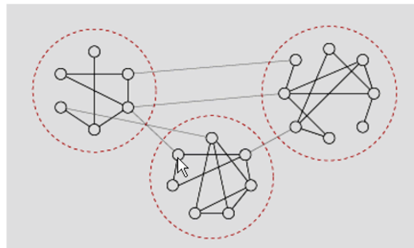


[3]

- k clusters → k x k symmetric matrix **e**: e_{ij} = |E(C_i,C_j)| / |E| : fraction of edges between communities
- $\mathrm{Tr}\,\mathbf{e} = \sum_i e_{ii}$: fraction of edges within communities
- $a_i = \sum_j e_{ij}$: fraction of edges that connect to cluster C_i
- Random network (keep a_i fixed): $e_{ij}^{\mathrm{rnd}} = a_i a_j \rightarrow e_{ii}^{\mathrm{rnd}} = a_i^2$
- f: Compare (→difference) real with rnd → $Q = \sum_i (e_{ii} - a_i^2) = \mathrm{Tr}\,\mathbf{e} - \| \mathbf{e}^2 \|$

Newman Girvan Method: Centrality-based Splitting + Modularity

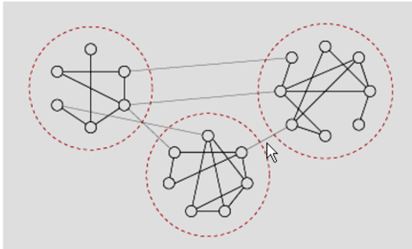
Modularity:



[3]

- k clusters → k x k symmetric matrix **e**: e_{ij} = |E(C_i,C_j)| / |E| : fraction of edges between communities
- $\mathrm{Tr}\,\mathbf{e} = \sum_i e_{ii}$: fraction of edges within communities
- $a_i = \sum_j e_{ij}$: fraction of edges that connect to cluster C_i
- Random network (keep a_i fixed): $e_{ij}^{\mathrm{rnd}} = a_i a_j \rightarrow e_{ii}^{\mathrm{rnd}} = a_i^2$
- f: Compare (→difference) real with rnd → $Q = \sum_i (e_{ii} - a_i^2) = \mathrm{Tr}\,\mathbf{e} - \| \mathbf{e}^2 \|$

Newman Girvan Method: Centrality-based Splitting + Modularity

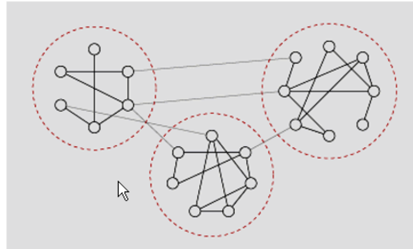


[3]

Modularity:

- k clusters → k x k symmetric matrix **e**: e_{ij} = |E(C_i,C_j)| / |E| : fraction of edges between communities
- $\mathrm{Tr}\,\mathbf{e} = \sum_i' e_{ii}$: fraction of edges within communities
- $a_i = \sum_j e_{ij}$: fraction of edges that connect to cluster C_i
- Random network (keep a_i fixed): $e_{ij}^{\text{rnd}} = a_i a_j \rightarrow e_{ii}^{\text{rnd}} = a_i^2$
- f: Compare (→difference)

real with rnd → $Q = \sum_i (e_{ii} - a_i^2) = \mathrm{Tr}\,\mathbf{e} - \|\mathbf{e}^2\|$



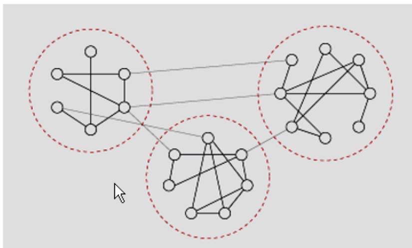
Newman Girvan Method: Centrality-based Splitting + Modularity

[3]

Modularity:

- k clusters → k x k symmetric matrix **e**: e_{ij} = |E(C_i,C_j)| / |E| : fraction of edges between communities
- $\mathrm{Tr}\,\mathbf{e} = \sum_i' e_{ii}$: fraction of edges within communities
- $a_i = \sum_j e_{ij}$: fraction of edges that connect to cluster C_i
- Random network (keep a_i fixed): $e_{ij}^{\text{rnd}} = a_i a_j \rightarrow e_{ii}^{\text{rnd}} = a_i^2$
- f: Compare (→difference)

real with rnd → $Q = \sum_i (e_{ii} - a_i^2) = \mathrm{Tr}\,\mathbf{e} - \|\mathbf{e}^2\|$



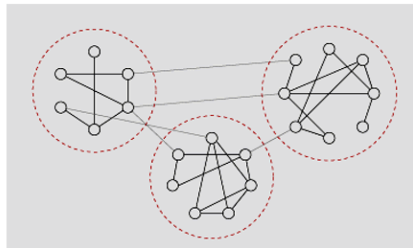
Newman Girvan Method: Centrality-based Splitting + Modularity

[3]

Modularity:

- k clusters → k x k symmetric matrix **e**: e_{ij} = |E(C_i,C_j)| / |E| : fraction of edges between communities
- $\mathrm{Tr}\,\mathbf{e} = \sum_i' e_{ii}$: fraction of edges within communities
- $a_i = \sum_j e_{ij}$: fraction of edges that connect to cluster C_i
- Random network (keep a_i fixed): $e_{ij}^{\text{rnd}} = a_i a_j \rightarrow e_{ii}^{\text{rnd}} = a_i^2$
- f: Compare (→difference)

real with rnd → $Q = \sum_i (e_{ii} - a_i^2) = \mathrm{Tr}\,\mathbf{e} - \|\mathbf{e}^2\|$



Newman Girvan Method: Centrality-based Splitting + Modularity

[3]

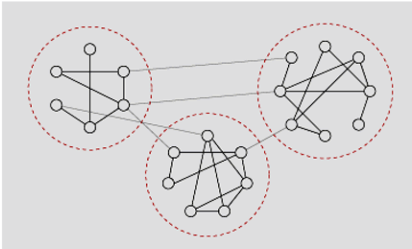
Modularity:

- k clusters → k x k symmetric matrix **e**: e_{ij} = |E(C_i,C_j)| / |E| : fraction of edges between communities
- $\mathrm{Tr}\,\mathbf{e} = \sum_i' e_{ii}$: fraction of edges within communities
- $a_i = \sum_j e_{ij}$: fraction of edges that connect to cluster C_i
- Random network (keep a_i fixed): $e_{ij}^{\text{rnd}} = a_i a_j \rightarrow e_{ii}^{\text{rnd}} = a_i^2$
- f: Compare (→difference)

real with rnd → $Q = \sum_i (e_{ii} - a_i^2) = \mathrm{Tr}\,\mathbf{e} - \|\mathbf{e}^2\|$

Newman Girvan Method: Centrality-based Splitting + Modularity

Modularity:

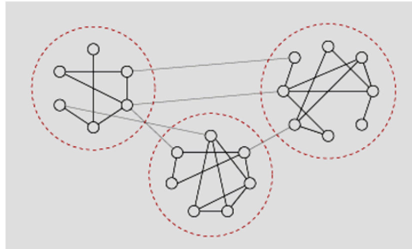


[3]

- k clusters → k x k symmetric matrix **e**: e_{ij} = |E(C_i,C_j)| / |E| : fraction of edges between communities
- $\mathrm{Tr}\,\mathbf{e} = \sum_i' e_{ii}$: fraction of edges within communities
- $a_i = \sum_j e_{ij}$: fraction of edges that connect to cluster C_i
- Random network (keep a_i fixed): $e_{ij}^{\text{rnd}} = a_i a_j \rightarrow e_{ii}^{\text{rnd}} = a_i^2$
- f: Compare (→difference)
  real with rnd → $$Q = \sum_i (e_{ii} - a_i^2) = \mathrm{Tr}\,\mathbf{e} - \|\mathbf{e}^2\|$$

Newman Girvan Method: Centrality-based Splitting + Modularity

Modularity:



[3]

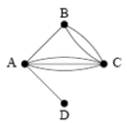
- k clusters → k x k symmetric matrix **e**: e_{ij} = |E(C_i,C_j)| / |E| : fraction of edges between communities
- $\mathrm{Tr}\,\mathbf{e} = \sum_i' e_{ii}$: fraction of edges within communities
- $a_i = \sum_j e_{ij}$: fraction of edges that connect to cluster C_i
- Random network (keep a_i fixed): $e_{ij}^{\text{rnd}} = a_i a_j \rightarrow e_{ii}^{\text{rnd}} = a_i^2$
- f: Compare (→difference)
  real with rnd → $$Q = \sum_i (e_{ii} - a_i^2) = \mathrm{Tr}\,\mathbf{e} - \|\mathbf{e}^2\|$$

Newman Girvan Method: Centrality-based Splitting + Modularity

Modularity:

- In [1]: different notion (not keeping a_i fixed): $$\sum_{i=1}^{k}\left(|E(C_i)| - m\frac{|C_i|\cdot(|C_i|-1)|}{n\cdot(n-1)}\right)$$

- In [4]: Newman's version for weighted graphs: idea: use multiple edges to model weights

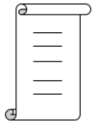


$$\equiv \begin{array}{cc} \begin{matrix} A & B & C & D \end{matrix} & \\ \begin{pmatrix} 0 & 1 & 3 & 1 \\ 1 & 0 & 2 & 0 \\ 3 & 2 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} & \begin{matrix} A \\ B \\ C \\ D \end{matrix} \end{array}$$

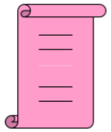
Data Mining:
Metric Clustering

Finding Clusters in Profiles

- Node profiles may contain:
  Personal data:
  name, age, sex, role-description, skill-description etc.,
  contextual personal data
  location, velocity, current emotional state etc.



- Edge profiles may contain:
  Averaged data:
  general type of relation, average strength of relation, etc..
  Time-dependent or contextual data:
  record of relation instantiations (with time & space information), momentary state of relation (weight, state of instantiation, etc.) etc.



Finding Clusters in Profiles

Examples for profile elements that can be embedded in metric spaces:

- Location & Velocity: Metric space: ($\mathbb{R}^3$, || . ||)
- Text describing Interests: Metric space: ($\mathbb{R}^{|Voc|}$, || . ||) where Voc denotes the Vocabulary of the text.

"I like to dance samba, bake pizza, watch tv and plant trees in the garden. I also like to bake cakes." ⇒

I	2
like	2
to	2
dance	1
samba	1
bake	2
pizza	1
watch	1
tv	1
and	1
plant	1
trees	1
in	1
the	1
garden	1
also	1
cakes	1

*Often: Instead of term-frequency (tf) alone: use term-frequency * inverse document frequency (idf); idf = log (#of docs where t occurs / #of docs)*

Finding Clusters in Profiles

Examples for profile elements that can be embedded in metric spaces:

- Location & Velocity: Metric space: ($\mathbb{R}^3$, || . ||)
- Text describing Interests: Metric space: ($\mathbb{R}^{|Voc|}$, || . ||) where Voc denotes the Vocabulary of the text.

"I like to dance samba, bake pizza, watch tv and plant trees in the garden. I also like to bake cakes." ⇒

I	2
like	2
to	2
dance	1
samba	1
bake	2
pizza	1
watch	1
tv	1
and	1
plant	1
trees	1
in	1
the	1
garden	1
also	1
cakes	1

*Often: Instead of term-frequency (tf) alone: use term-frequency * inverse document frequency (idf); idf = log (#of docs where t occurs / #of docs)*

Finding Clusters in Profiles

- How do we compute clusters in metric spaces?
- Group models: How do we compute socially meaningful clusters in metric spaces (and thus avoid quasi-groups)?
- First some notations / basics:
  - In graph clustering we had: A graph clustering **C**={C_1, C_2, ..., C_K} is a partion of V into non-empty subsets C_k
  - Now: clustering $\mathscr{C} : \mathcal{X} \to \mathcal{I}$: mapping of a metric value space $\mathcal{X}$ to a set of cluster indices $\mathcal{I}$
  - Clusterings can be:
    - exclusive or non-exclusive
    - crisp or fuzzy
    - hierarchical or non-hierarchical

Metric variant of Single / Complete link clustering

- Metric variant of Single / Complete link clustering: Hierarchical, crisp, non-overlapping

- Completely analogous to graph clustering case: Start with singletons and on each level of the dendrogram merge two clusters with minimal distance (cost)

  - Single link:

$$d(\mathcal{C}_{k_1}, \mathcal{C}_{k_2}) = \min_{\{n_1, n_2 | x_{n_1} \in \mathcal{C}_{k_1} \wedge x_{n_2} \in \mathcal{C}_{k_2}\}} ||x_{n_1} - x_{n_2}||$$

  - Complete link:

$$d(\mathcal{C}_{k_1}, \mathcal{C}_{k_2}) = \max_{\{n_1, n_2 | x_{n_1} \in \mathcal{C}_{k_1} \wedge x_{n_2} \in \mathcal{C}_{k_2}\}} ||x_{n_1} - x_{n_2}||$$

Metric variant of Single / Complete link clustering

- Metric variant of Single / Complete link clustering: Hierarchical, crisp, non-overlapping

- Completely analogous to graph clustering case: Start with singletons and on each level of the dendrogram merge two clusters with minimal distance (cost)

  - Single link:

$$d(\mathcal{C}_{k_1}, \mathcal{C}_{k_2}) = \min_{\{n_1, n_2 | x_{n_1} \in \mathcal{C}_{k_1} \wedge x_{n_2} \in \mathcal{C}_{k_2}\}} ||x_{n_1} - x_{n_2}||$$

  - Complete link:

$$d(\mathcal{C}_{k_1}, \mathcal{C}_{k_2}) = \max_{\{n_1, n_2 | x_{n_1} \in \mathcal{C}_{k_1} \wedge x_{n_2} \in \mathcal{C}_{k_2}\}} ||x_{n_1} - x_{n_2}||$$

Metric variant of Single / Complete link clustering

- Metric variant of Single / Complete link clustering: Hierarchical, crisp, non-overlapping

- Completely analogous to graph clustering case: Start with singletons and on each level of the dendrogram merge two clusters with minimal distance (cost)

  - Single link:

$$d(\mathcal{C}_{k_1}, \mathcal{C}_{k_2}) = \min_{\{n_1, n_2 | x_{n_1} \in \mathcal{C}_{k_1} \wedge x_{n_2} \in \mathcal{C}_{k_2}\}} ||x_{n_1} - x_{n_2}||$$

  - Complete link:

$$d(\mathcal{C}_{k_1}, \mathcal{C}_{k_2}) = \max_{\{n_1, n_2 | x_{n_1} \in \mathcal{C}_{k_1} \wedge x_{n_2} \in \mathcal{C}_{k_2}\}} ||x_{n_1} - x_{n_2}||$$

K-Means Clustering

- General idea (also valid in graph clustering): Optimize objective function that formalizes clustering paradigm.

- K-Means: Optimize intra cluster coherence:

  - Describe cluster C_k by prototype μ_k; prototype need not be an actual pattern (If so, algorithm works with slight modifications as well)

  - Determine cluster for each pattern x_n by nearest neighbour rule:

$$\mathscr{C}(x_n) = k_a \leftrightarrow ||x_n - \mu_{k_a}|| = \min_i ||x_n - \mu_k||$$

K-Means Clustering

- K-Means: Optimize intra cluster coherence:
  - Find prototypes by optimizing objective function modeling intra cluster coherence as mean square error

$$J_{\mathrm{SQE}} = \sum_{k=1}^{K} \sum_{\{n | x_n \in \mathcal{C}_k\}} ||x_n - \mu_k||^2$$

$$\frac{\mathrm{d}J_{\mathrm{SQE}}}{\mathrm{d}\mu_k} \overset{!}{=} 0 \quad \Longrightarrow \quad \mu^k = \frac{1}{|\mathcal{C}_k|} \sum_{\{n | x_n \in \mathcal{C}_k\}} x_n$$

  - → cluster prototypes are barycenters („centers of gravity") of their clusters.

K-Means Clustering

- K-Means: Optimize intra cluster coherence:
  - Find prototypes by optimizing objective function modeling intra cluster coherence as mean square error

$$J_{\mathrm{SQE}} = \sum_{k=1}^{K} \sum_{\{n | x_n \in \mathcal{C}_k\}} ||x_n - \mu_k||^2$$

$$\frac{\mathrm{d}J_{\mathrm{SQE}}}{\mathrm{d}\mu_k} \overset{!}{=} 0 \quad \Longrightarrow \quad \mu^k = \frac{1}{|\mathcal{C}_k|} \sum_{\{n | x_n \in \mathcal{C}_k\}} x_n$$

  - → cluster prototypes are barycenters („centers of gravity") of their clusters.

K-Means Clustering

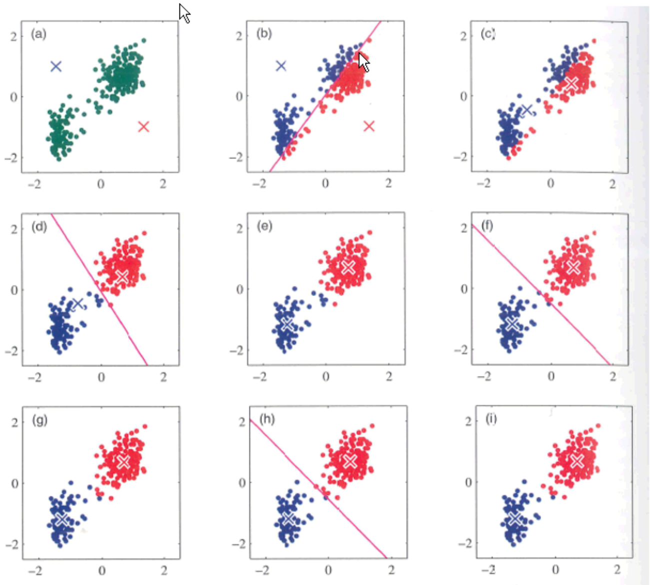
- K-Means: Optimize intra cluster coherence:
  - Find prototypes by optimizing objective function modeling intra cluster coherence as mean square error

$$J_{\mathrm{SQE}} = \sum_{k=1}^{K} \sum_{\{n | x_n \in \mathcal{C}_k\}} ||x_n - \mu_k||^2$$

$$\frac{\mathrm{d}J_{\mathrm{SQE}}}{\mathrm{d}\mu_k} \overset{!}{=} 0 \quad \Longrightarrow \quad \mu^k = \frac{1}{|\mathcal{C}_k|} \sum_{\{n | x_n \in \mathcal{C}_k\}} x_n$$

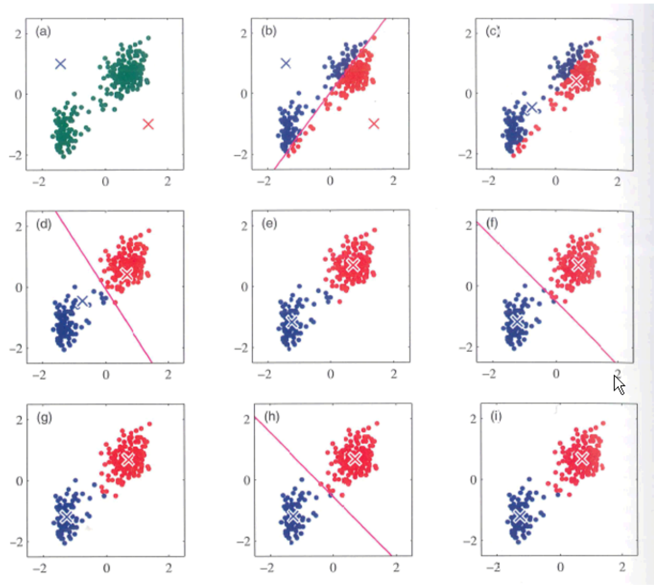
  - → cluster prototypes are barycenters („centers of gravity") of their clusters.

K-Means Clustering



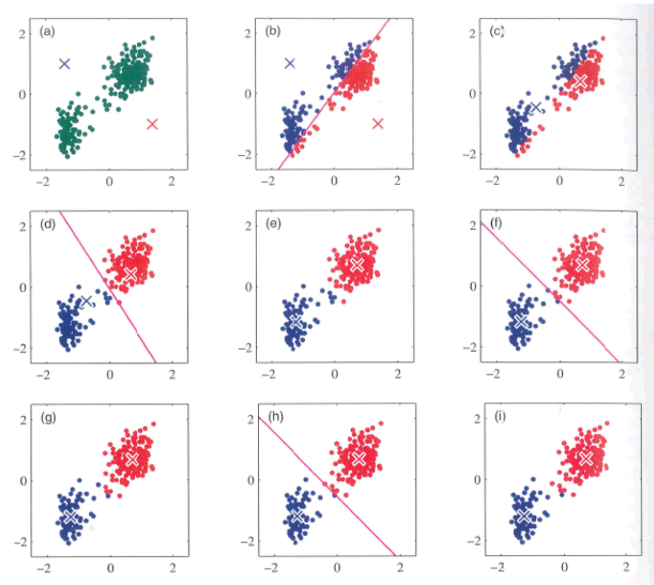
[3]

K-Means Clustering



[3]

K-Means Clustering



[3]

K-Means Clustering

- Interesting aspect: How do we determine correct number k of clusters? (Same problem with graph clustering: where to cut dendrogram?)
- Answer: Compute for every k clusterings; chose the best clustering with a cluster quality measure
- Cluster quality measures for metric case: (countless variants exist in literature; for an overview: e.g. [2]) (Objective functions modeling clustering paradigm):
  - Dunn-Index
  - Entropy based indices
  - ....

K-Means Clustering

- Dunn Index:

$$D = \min_{k_1 \in [1,K]} \left(\min_{k_2 \in [1,K]} \left(\frac{d_1(\mathcal{C}_{k_1}, \mathcal{C}_{k_2})}{\max_{k_3 \in [1,K]} d_2(\mathcal{C}_{k_3})} \right) \right)$$

where $d_1(\mathcal{C}_{k_1}, \mathcal{C}_{k_2})$ is the distance function between two clusters defined by

$$d_1(\mathcal{C}_{k_1}, \mathcal{C}_{k_2}) = \min_{\{(n_1,n_2) \mid x_{n_1} \in \mathcal{C}_{k_1} \wedge x_{n_2} \in \mathcal{C}_{k_2}\}} ||x_{n_1} - x_{n_2}||$$

(that is the single link distance from SAHN).
The "diameter" d_2 of the clusters is defined by

$$d_2(\mathcal{C}_i) = \max_{\{(n_1,n_2) \mid x_{n_1} \in \mathcal{C}_i \wedge x_{n_2} \in \mathcal{C}_i\}} ||x_{n_1} - x_{n_2}||$$

[7]

- Dunn Index:

$$D = \min_{k_1 \in [1,K]} \left(\min_{k_2 \in [1,K]} \left(\frac{d_1(\mathcal{C}_{k_1}, \mathcal{C}_{k_2})}{\max_{k_3 \in [1,K]} d_2(\mathcal{C}_{k_3})} \right) \right)$$

where $d_1(\mathcal{C}_{k_1}, \mathcal{C}_{k_2})$ is the distance function between two clusters defined by

$$d_1(\mathcal{C}_{k_1}, \mathcal{C}_{k_2}) = \min_{\{(n_1,n_2) | x_{n_1} \in \mathcal{C}_{k_1} \wedge x_{n_2} \in \mathcal{C}_{k_2}\}} ||x_{n_1} - x_{n_2}||$$

(that is the single link distance from SAHN).
The "diameter" d_2 of the clusters is defined by

$$d_2(\mathcal{C}_i) = \max_{\{(n_1,n_2) | x_{n_1} \in \mathcal{C}_i \wedge x_{n_2} \in \mathcal{C}_i\}} ||x_{n_1} - x_{n_2}||$$

K-Means Clustering

- Dunn Index:

$$D = \min_{k_1 \in [1,K]} \left(\min_{k_2 \in [1,K]} \left(\frac{d_1(\mathcal{C}_{k_1}, \mathcal{C}_{k_2})}{\max_{k_3 \in [1,K]} d_2(\mathcal{C}_{k_3})} \right) \right)$$

where $d_1(\mathcal{C}_{k_1}, \mathcal{C}_{k_2})$ is the distance function between two clusters defined by

$$d_1(\mathcal{C}_{k_1}, \mathcal{C}_{k_2}) = \min_{\{(n_1,n_2) | x_{n_1} \in \mathcal{C}_{k_1} \wedge x_{n_2} \in \mathcal{C}_{k_2}\}} ||x_{n_1} - x_{n_2}||$$

(that is the single link distance from SAHN).
The "diameter" d_2 of the clusters is defined by

$$d_2(\mathcal{C}_i) = \max_{\{(n_1,n_2) | x_{n_1} \in \mathcal{C}_i \wedge x_{n_2} \in \mathcal{C}_i\}} ||x_{n_1} - x_{n_2}||$$

[7]

K-Means Clustering

- Dunn Index:

$$D = \min_{k_1 \in [1,K]} \left(\min_{k_2 \in [1,K]} \left(\frac{d_1(\mathcal{C}_{k_1}, \mathcal{C}_{k_2})}{\max_{k_3 \in [1,K]} d_2(\mathcal{C}_{k_3})} \right) \right)$$

where $d_1(\mathcal{C}_{k_1}, \mathcal{C}_{k_2})$ is the distance function between two clusters defined by

$$d_1(\mathcal{C}_{k_1}, \mathcal{C}_{k_2}) = \min_{\{(n_1,n_2) | x_{n_1} \in \mathcal{C}_{k_1} \wedge x_{n_2} \in \mathcal{C}_{k_2}\}} ||x_{n_1} - x_{n_2}||$$

(that is the single link distance from SAHN).
The "diameter" d_2 of the clusters is defined by

$$d_2(\mathcal{C}_i) = \max_{\{(n_1,n_2) | x_{n_1} \in \mathcal{C}_i \wedge x_{n_2} \in \mathcal{C}_i\}} ||x_{n_1} - x_{n_2}||$$

[7]

K-Means Clustering

- Dunn Index:

$$D = \min_{k_1 \in [1,K]} \left(\min_{k_2 \in [1,K]} \left(\frac{d_1(\mathcal{C}_{k_1}, \mathcal{C}_{k_2})}{\max_{k_3 \in [1,K]} d_2(\mathcal{C}_{k_3})} \right) \right)$$

where $d_1(\mathcal{C}_{k_1}, \mathcal{C}_{k_2})$ is the distance function between two clusters defined by

$$d_1(\mathcal{C}_{k_1}, \mathcal{C}_{k_2}) = \min_{\{(n_1,n_2) | x_{n_1} \in \mathcal{C}_{k_1} \wedge x_{n_2} \in \mathcal{C}_{k_2}\}} ||x_{n_1} - x_{n_2}||$$

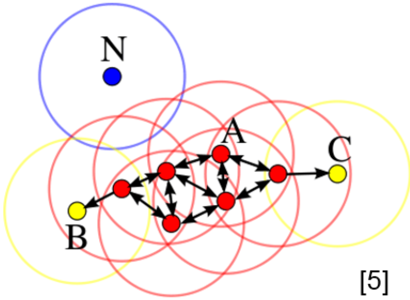
(that is the single link distance from SAHN).
The "diameter" d_2 of the clusters is defined by

$$d_2(\mathcal{C}_i) = \max_{\{(n_1,n_2) | x_{n_1} \in \mathcal{C}_i \wedge x_{n_2} \in \mathcal{C}_i\}} ||x_{n_1} - x_{n_2}||$$

[7]

DBSCAN

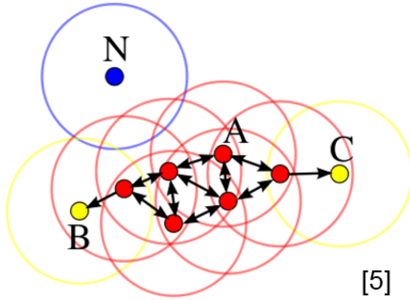
- K-Means is „OK“ as cluster algorithm, but has certain disadvantages:
  - favors spherical clusters
  - need to know K
  - no notion of noise
- Alternative → DBSCAN [4]
  (de facto state of the art):
  - Idea: Two parameters: minPt, ε
  - Rough idea: iterate:
    visit previously unseen pattern x:
      if in ε-neighborhood {x‘} of x: |{x‘}|≥ minPt then
        start new cluster: include x and {x‘} and those of their
        ε-neighborhoods {x‘‘} that are dense enough (|{x‘‘}|≥
        minPt), etc.
      else: x is noise



[5]

DBSCAN

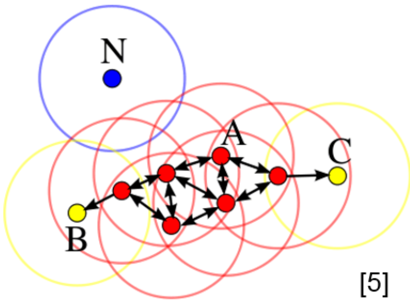
- K-Means is „OK“ as cluster algorithm, but has certain disadvantages:
  - favors spherical clusters
  - need to know K
  - no notion of noise
- Alternative → DBSCAN [4]
  (de facto state of the art):
  - Idea: Two parameters: minPt, ε
  - Rough idea: iterate:
    visit previously unseen pattern x:
      if in ε-neighborhood {x‘} of x: |{x‘}|≥ minPt then
        start new cluster: include x and {x‘} and those of their
        ε-neighborhoods {x‘‘} that are dense enough (|{x‘‘}|≥
        minPt), etc.
      else: x is noise



[5]

DBSCAN

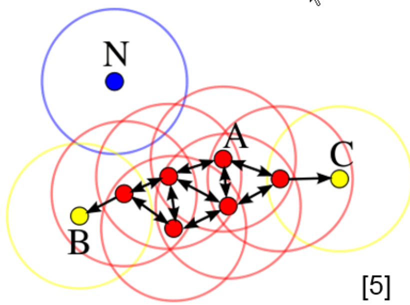
- K-Means is „OK“ as cluster algorithm, but has certain disadvantages:
  - favors spherical clusters
  - need to know K
  - no notion of noise
- Alternative → DBSCAN [4]
  (de facto state of the art):
  - Idea: Two parameters: minPt, ε
  - Rough idea: iterate:
    visit previously unseen pattern x:
      if in ε-neighborhood {x‘} of x: |{x‘}|≥ minPt then
        start new cluster: include x and {x‘} and those of their
        ε-neighborhoods {x‘‘} that are dense enough (|{x‘‘}|≥
        minPt), etc.
      else: x is noise



[5]

DBSCAN

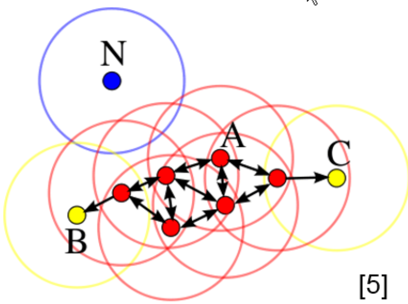
- K-Means is „OK“ as cluster algorithm, but has certain disadvantages:
  - favors spherical clusters
  - need to know K
  - no notion of noise
- Alternative → DBSCAN [4]
  (de facto state of the art):
  - Idea: Two parameters: minPt, ε
  - Rough idea: iterate:
    visit previously unseen pattern x:
      if in ε-neighborhood {x‘} of x: |{x‘}|≥ minPt then
        start new cluster: include x and {x‘} and those of their
        ε-neighborhoods {x‘‘} that are dense enough (|{x‘‘}|≥
        minPt), etc.
      else: x is noise



[5]

DBSCAN

- K-Means is „OK“ as cluster algorithm, but has certain disadvantages:
  - favors spherical clusters
  - need to know K
  - no notion of noise
- Alternative → DBSCAN [4]
  (de facto state of the art):
  - Idea: Two parameters: minPt, ε
  - Rough idea: iterate:
    visit previously unseen pattern x:
      if in ε-neighborhood {x‘} of x: |{x‘}|≥ minPt then
        start new cluster: include x and {x‘} and those of their
        ε-neighborhoods {x‘‘} that are dense enough (|{x‘‘}|≥
        minPt), etc.
      else: x is noise



[5]

K-Means Clustering

Example Application: Clustering locations

- Problem: How do we distinguish socially relevant clusters (candidates for groups) from quasi groups?
  - Compute clusterings over period of time: Good candidates: clusters that appear over and over again, clusters that appear periodically
  - Establish threshold for distance in clusters: Human “social distance”: A few meters (if groups are very small); few tens of meters (if groups are medium sized)
  - Include velocities: If divergent → no group

K-Means Clustering

- K-Means: Optimize intra cluster coherence:
  - Find prototypes by optimizing objective function modeling intra cluster coherence as mean square error

$$J_{\mathrm{SQE}} = \sum_{k=1}^{K} \sum_{\{n | x_n \in \mathcal{C}_k\}} ||x_n - \mu_k||^2$$

$$\frac{\mathrm{d}J_{\mathrm{SQE}}}{\mathrm{d}\mu_k} \stackrel{!}{=} 0 \quad \Longrightarrow \quad \mu^k = \frac{1}{|\mathcal{C}_k|} \sum_{\{n | x_n \in \mathcal{C}_k\}} x_n$$

  - → cluster prototypes are barycenters („centers of gravity“) of their clusters.

Fuzzy C-Means Clustering

- K-Means was a crisp algorithm. Now: fuzzy variant
- Reformulate K-Means objective function with membership matrix
r_{nk}: Membership of pattern x_n in class C_k

$$J_{SQE} = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} ||x_n - \mu_k||^2$$

- optimization criterion

$$\mathrm{d}J_{SQE}/\mathrm{d}\mu_k = 0$$

- together with non-overlaping constraint

$$\forall n (\exists k (r_{nk} = 1) \wedge ((k' \neq k) \rightarrow (r_{nk'} = 0)))$$

leads to well known K-Means

$$\mu_k = \sum_{n=1}^{N} r_{nk} x_n / \sum_{n=1}^{N} r_{nk} = (1/|\mathcal{C}_k|) \sum_{n | x_n \in \mathcal{C}_k} x_n$$

Fuzzy C-Means Clustering

- Now modify objective function to:

$$J_{GSQE} = \sum_{n=1}^{N} \sum_{k=1}^{K} (r_{nk})^m ||x_n - \mu_k||^2$$

- Exponent m models degree of fuzzyness:
  m → 1 : K-Means (crisp case);
  m → ∞ : r_{nk} → 1/K (where K is the number of clusters)

- Optimize the obj. fct. under the conditions:

$$\forall\, x_n \quad : \quad \sum_{k=1}^{K} \alpha_k(x_n) = \sum_{k=1}^{K} r_{nk} = 1$$

$$\forall\, \mathcal{C}_k \quad : \quad \sum_{n=1}^{N} \alpha_k(x_n) = \sum_{n=1}^{N} r_{nk} > 0$$

Fuzzy C-Means Clustering

- Result:

$$r_{nk} = \Big(\sum_{k'=1}^{K} \big(\frac{||x_n - \mu_k||}{||x_n - \mu_{k'}||}\big)^{\frac{2}{m-1}}\Big)^{-1} \qquad (☺)$$

$$\mu_k = \frac{\sum_{n=1}^{N} r_{nk}^m x_n}{\sum_{n=1}^{N} r_{nk}} \qquad (☺☺)$$

- the result assumes that no patterns and prototypes coincide

$$\forall\, n, k : \quad ||x_n - \mu_k|| \neq 0$$

if they do coincide, set $r_{nk} = 1$ for $x_n = \mu_k$ and $r_{nk} = 0$ for $x_n \neq \mu_k$

Fuzzy C-Means Clustering

- Result:

$$r_{nk} = \Big(\sum_{k'=1}^{K} \big(\frac{||x_n - \mu_k||}{||x_n - \mu_{k'}||}\big)^{\frac{2}{m-1}}\Big)^{-1} \qquad (☺)$$

$$\mu_k = \frac{\sum_{n=1}^{N} r_{nk}^m x_n}{\sum_{n=1}^{N} r_{nk}} \qquad (☺☺)$$

- the result assumes that no patterns and prototypes coincide

$$\forall\, n, k : \quad ||x_n - \mu_k|| \neq 0$$

if they do coincide, set $r_{nk} = 1$ for $x_n = \mu_k$ and $r_{nk} = 0$ for $x_n \neq \mu_k$

Fuzzy C-Means Clustering

- Result:

$$r_{nk} = \Big(\sum_{k'=1}^{K} \big(\frac{||x_n - \mu_k||}{||x_n - \mu_{k'}||}\big)^{\frac{2}{m-1}}\Big)^{-1} \qquad (☺)$$

$$\mu_k = \frac{\sum_{n=1}^{N} r_{nk}^m x_n}{\sum_{n=1}^{N} r_{nk}} \qquad (☺☺)$$

- the result assumes that no patterns and prototypes coincide

$$\forall\, n, k : \quad ||x_n - \mu_k|| \neq 0$$

if they do coincide, set $r_{nk} = 1$ for $x_n = \mu_k$ and $r_{nk} = 0$ for $x_n \neq \mu_k$

Fuzzy C-Means Clustering

- Limit m → ∞ gives:

$$r_{nk} \overset{m\to\infty}{\longrightarrow} \frac{1}{\sum_{k'=1}^{K} 1} = \frac{1}{K}$$

- Limit m → 1 we get the nearest neighbor rule (K-Means) because:

$$r_{nk} = 1/((\textstyle\sum_{k'\neq k}(\frac{||x_n-\mu_k||}{||x_n-\mu_{k'}||})^{\frac{2}{m-1}}) + 1)$$

in the limit m→1 the first sum in the denominator becomes ∞ if

$$||x_n - \mu_k|| \neq \min_{1\leq k'\leq K} ||x_n - \mu_{k'}||$$

and it becomes 0 if

$$||x_n - \mu_k|| = \min_{1\leq k'\leq K} ||x_n - \mu_{k'}||$$

Fuzzy C-Means Clustering

- Limit m → ∞ gives:

$$r_{nk} \overset{m\to\infty}{\longrightarrow} \frac{1}{\sum_{k'=1}^{K} 1} = \frac{1}{K}$$

- Limit m → 1 we get the nearest neighbor rule (K-Means) because:

$$r_{nk} = 1/((\textstyle\sum_{k'\neq k}(\frac{||x_n-\mu_k||}{||x_n-\mu_{k'}||})^{\frac{2}{m-1}}) + 1)$$

in the limit m→1 the first sum in the denominator becomes ∞ if

$$||x_n - \mu_k|| \neq \min_{1\leq k'\leq K} ||x_n - \mu_{k'}||$$

and it becomes 0 if

$$||x_n - \mu_k|| = \min_{1\leq k'\leq K} ||x_n - \mu_{k'}||$$

Fuzzy C-Means Clustering

- Limit m → ∞ gives:

$$r_{nk} \overset{m\to\infty}{\longrightarrow} \frac{1}{\sum_{k'=1}^{K} 1} = \frac{1}{K}$$

- Limit m → 1 we get the nearest neighbor rule (K-Means) because:

$$r_{nk} = 1/((\textstyle\sum_{k'\neq k}(\frac{||x_n-\mu_k||}{||x_n-\mu_{k'}||})^{\frac{2}{m-1}}) + 1)$$

in the limit m→1 the first sum in the denominator becomes ∞ if

$$||x_n - \mu_k|| \neq \min_{1\leq k'\leq K} ||x_n - \mu_{k'}||$$

and it becomes 0 if

$$||x_n - \mu_k|| = \min_{1\leq k'\leq K} ||x_n - \mu_{k'}||$$

Fuzzy C-Means Clustering

- Limit m → ∞ gives:

$$r_{nk} \overset{m\to\infty}{\longrightarrow} \frac{1}{\sum_{k'=1}^{K} 1} = \frac{1}{K}$$

- Limit m → 1 we get the nearest neighbor rule (K-Means) because:

$$r_{nk} = 1/((\textstyle\sum_{k'\neq k}(\frac{||x_n-\mu_k||}{||x_n-\mu_{k'}||})^{\frac{2}{m-1}}) + 1)$$

in the limit m→1 the first sum in the denominator becomes ∞ if

$$||x_n - \mu_k|| \neq \min_{1\leq k'\leq K} ||x_n - \mu_{k'}||$$

and it becomes 0 if

$$||x_n - \mu_k|| = \min_{1\leq k'\leq K} ||x_n - \mu_{k'}||$$

Gaussian Mixture Models

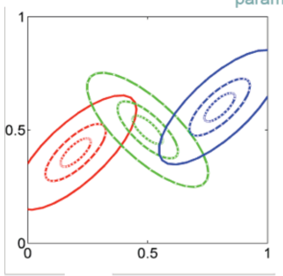
- Fuzzy C-Means is “OK” as a non-crisp clustering alg. but (as K-Means) favors spherical clusters → better approaches

- Example: Gaussian Mixture Models (GMM)

  - Linear combination of Gaussians

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k) \quad \text{where} \quad \sum_{k=1}^{K} \pi_k = 1, \quad 0 \leqslant \pi_k \leqslant 1$$

parameters to be estimated



[6]

Gaussian Mixture Models

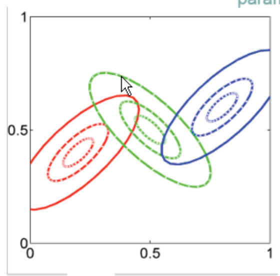
- Fuzzy C-Means is “OK” as a non-crisp clustering alg. but (as K-Means) favors spherical clusters → better approaches

- Example: Gaussian Mixture Models (GMM)

  - Linear combination of Gaussians

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k) \quad \text{where} \quad \sum_{k=1}^{K} \pi_k = 1, \quad 0 \leqslant \pi_k \leqslant 1$$

parameters to be estimated



[6]

Gaussian Mixture Models

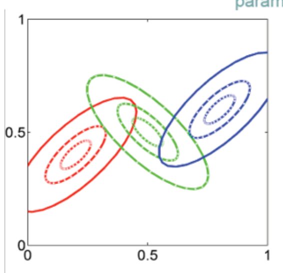
- Fuzzy C-Means is “OK” as a non-crisp clustering alg. but (as K-Means) favors spherical clusters → better approaches

- Example: Gaussian Mixture Models (GMM)

  - Linear combination of Gaussians

$$p(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k) \quad \text{where} \quad \sum_{k=1}^{K} \pi_k = 1, \quad 0 \leqslant \pi_k \leqslant 1$$

parameters to be estimated



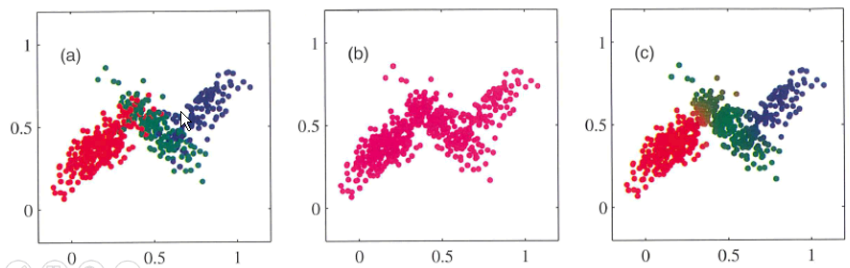
[6]

GMM-Basics

- Responsibilities

$$\gamma(z_k) \equiv p(z_k = 1|\mathbf{x}) = \frac{p(z_k = 1)p(\mathbf{x}|z_k = 1)}{\sum_{j=1}^{K} p(z_j = 1)p(\mathbf{x}|z_j = 1)} = \frac{\pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)}.$$

- Example

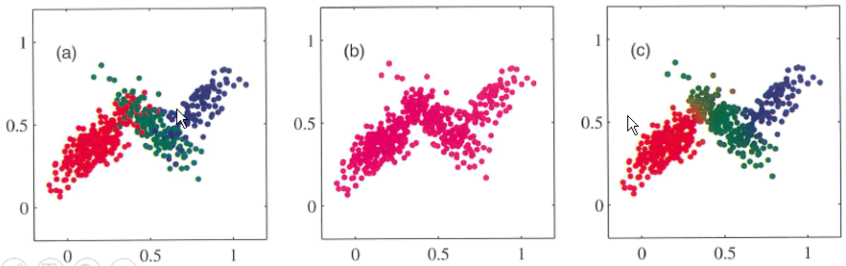


GMM-Basics

- Responsibilities

$$\gamma(z_k) \equiv p(z_k=1|\mathbf{x}) = \frac{p(z_k=1)p(\mathbf{x}|z_k=1)}{\sum_{j=1}^{K} p(z_j=1)p(\mathbf{x}|z_j=1)} = \frac{\pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)}.$$

- Example

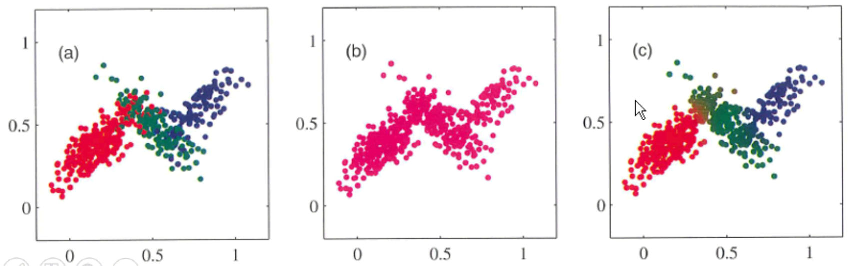


GMM-Basics

- Responsibilities

$$\gamma(z_k) \equiv p(z_k=1|\mathbf{x}) = \frac{p(z_k=1)p(\mathbf{x}|z_k=1)}{\sum_{j=1}^{K} p(z_j=1)p(\mathbf{x}|z_j=1)} = \frac{\pi_k \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)}.$$

- Example



Machine Learning

Example: $x \in \mathbb{R}^n$ and $p(x|\Theta)$ is one multivariate Gaussian)

$$p(\mathbf{x}|\Theta) = \mathcal{N}(\mathbf{x}|\underbrace{\boldsymbol{\mu},\boldsymbol{\Sigma}}_{\theta}) = \frac{1}{(2\pi)^{D/2}}\frac{1}{|\boldsymbol{\Sigma}|^{1/2}}\exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$

- log likelihood: (use base e)

$$\ln p(\mathbf{X}|\Theta) = \ln p(\mathbf{X}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = -\frac{ND}{2}\ln(2\pi) - \frac{N}{2}\ln|\boldsymbol{\Sigma}| - \frac{1}{2}\sum_{n=1}^{N}(\mathbf{x}_n-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_n-\boldsymbol{\mu})$$

- Maximum log likelihood:

$$\Theta_{ML} = argmax_{\Theta} \log p(X|\Theta) \Rightarrow \nabla_{\Theta}\left(\sum_i \log p(x_i|\Theta)\right) \overset{!}{=} 0$$

$$\boldsymbol{\mu}_{\mathrm{ML}}: \frac{\partial}{\partial \boldsymbol{\mu}} \ln p(\mathbf{X}|\mu,\boldsymbol{\Sigma}) = 0 \qquad \boldsymbol{\mu}_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n$$

$$\boldsymbol{\Sigma}_{\mathrm{ML}}: \frac{\partial}{\partial \boldsymbol{\Sigma}} \ln p(\mathbf{X}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = 0 \Rightarrow \boldsymbol{\Sigma}_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n-\boldsymbol{\mu}_{\mathrm{ML}})(\mathbf{x}_n-\boldsymbol{\mu}_{\mathrm{ML}})^{\mathrm{T}}$$

Machine Learning

Example: $x \in \mathbb{R}^n$ and $p(x|\Theta)$ is one multivariate Gaussian)

$$p(\mathbf{x}|\Theta) = \mathcal{N}(\mathbf{x}|\underbrace{\boldsymbol{\mu},\boldsymbol{\Sigma}}_{\theta}) = \frac{1}{(2\pi)^{D/2}}\frac{1}{|\boldsymbol{\Sigma}|^{1/2}}\exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$

- log likelihood: (use base e)

$$\ln p(\mathbf{X}|\Theta) = \ln p(\mathbf{X}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = -\frac{ND}{2}\ln(2\pi) - \frac{N}{2}\ln|\boldsymbol{\Sigma}| - \frac{1}{2}\sum_{n=1}^{N}(\mathbf{x}_n-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\mathbf{x}_n-\boldsymbol{\mu})$$

- Maximum log likelihood:

$$\Theta_{ML} = argmax_{\Theta} \log p(X|\Theta) \Rightarrow \nabla_{\Theta}\left(\sum_i \log p(x_i|\Theta)\right) \overset{!}{=} 0$$

$$\boldsymbol{\mu}_{\mathrm{ML}}: \frac{\partial}{\partial \boldsymbol{\mu}} \ln p(\mathbf{X}|\mu,\boldsymbol{\Sigma}) = 0 \qquad \boldsymbol{\mu}_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n$$

$$\boldsymbol{\Sigma}_{\mathrm{ML}}: \frac{\partial}{\partial \boldsymbol{\Sigma}} \ln p(\mathbf{X}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = 0 \Rightarrow \boldsymbol{\Sigma}_{\mathrm{ML}} = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{x}_n-\boldsymbol{\mu}_{\mathrm{ML}})(\mathbf{x}_n-\boldsymbol{\mu}_{\mathrm{ML}})^{\mathrm{T}}$$