Statistical clustering methods

School of Electrical and Computer Engineering

University of Tehran

Erfan Darzi

erfandarzi@ut.ac.ir

* Describe the pairwise distance via a graph
* Describe the pairwise distance via a graph – Clustering can be obtained via graph cut

* Describe the pairwise distance via a graph – Clustering can be obtained via graph cut
* Describe the pairwise distance via a graph

– Clustering can be obtained via graph cut

Cut by class label

Cut by cluster label

Recap: external validation

* Given class label Ω on each instance – Rand index

𝑤𝑤

𝑖𝑖

=

w

j

𝑤𝑤

𝑖𝑖

≠

w

j

𝑐𝑐

𝑖𝑖

=

𝑐𝑐

𝑗𝑗

20

20

𝑐𝑐

𝑖𝑖

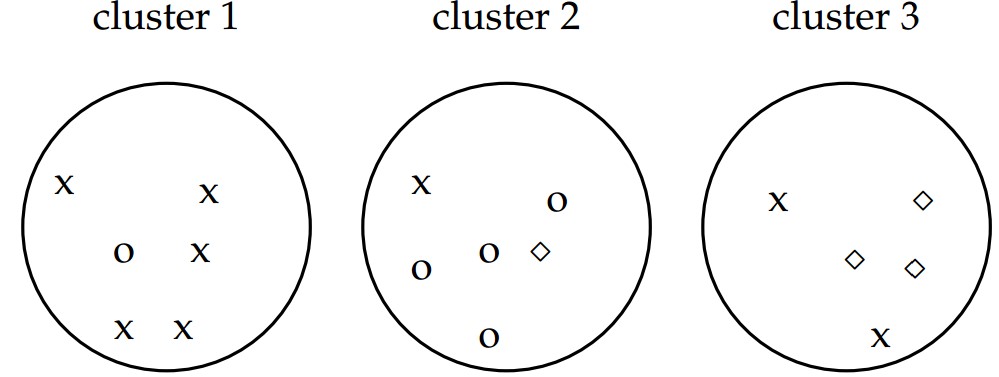
≠

𝑐𝑐

𝑗𝑗

24

72



𝑇𝑇𝑇𝑇

+

𝐹𝐹

𝑇𝑇

=

6

2

+

6

2

+

5

2

=

40

𝑇𝑇𝑇𝑇

=

5

2

+

4

2

+

3

2

+

2

2

=

20

# *k*-means Clustering

Hongning Wang

CS@UVa

# Today’s lecture

* *k*-means clustering
  + A typical partitional clustering algorithm
  + Convergence property
* Expectation Maximization algorithm

– Gaussian mixture model

# Partitional clustering algorithms

• Partition instances into exactly *k* nonoverlapping clusters – Flat structure clustering

– Users need to specify the cluster size *k*

–

Task: identify the partition of

*k*

clusters that

optimize the chosen partition criterion



# Partitional clustering algorithms

* Partition instances into exactly *k* nonoverlapping clusters **Optimize this in an alternative way**

– Typical criterion Inter-cluster distanceIntra-cluster distance

* max ∑𝑖𝑖≠j 𝑑𝑑 𝑐𝑐𝑖𝑖, 𝑐𝑐𝑗𝑗 −𝐶𝐶∑𝑖𝑖 𝜎𝜎𝑖𝑖

– Optimal solution: enumerate every possible partition of size *k* and return the one maximizes the criterion

Let’s approximate this! ***Unfortunately, this is NP-hard!***

# *k*-means algorithm

Input: cluster size *k*, instances {𝑥𝑥𝑖𝑖}𝑁𝑁𝑖𝑖=1, distance metric 𝑑𝑑(⋅,⋅)

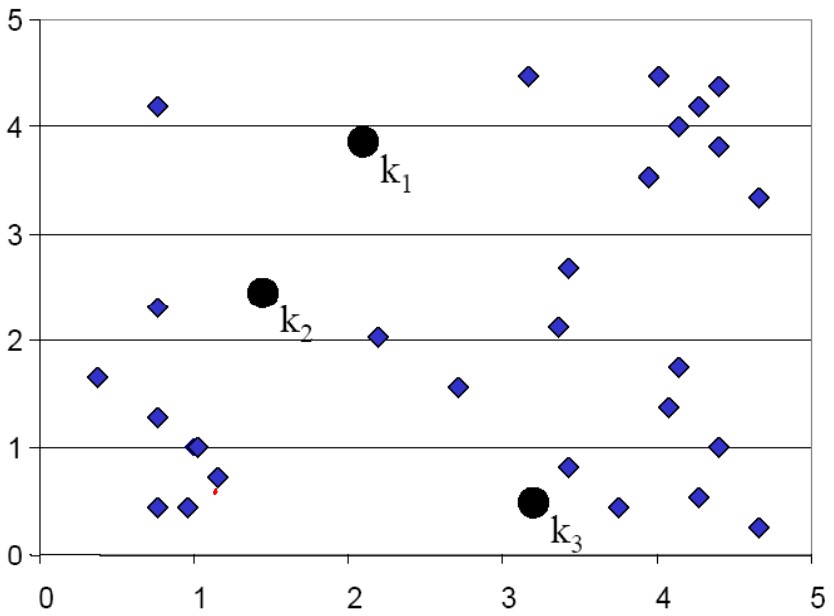
Output: cluster membership assignments {𝑧𝑧𝑖𝑖}𝑁𝑁𝑖𝑖=1

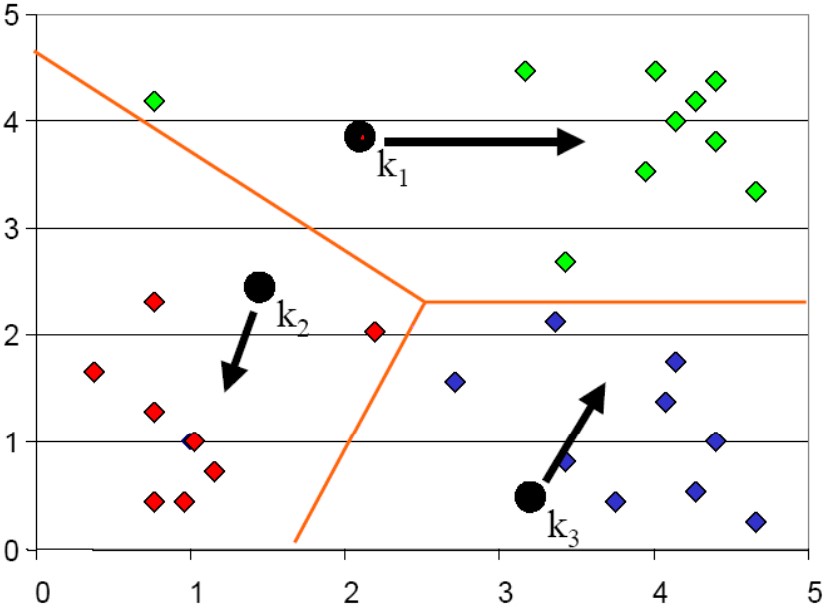
1. Initialize *k* cluster centroids {𝑐𝑐𝑖𝑖}𝑘𝑘𝑖𝑖=1 (randomly if no domain knowledge is available)
2. Repeat until no instance changes its cluster membership: – Decide the cluster membership of instances by assigning them to the nearest cluster centroid

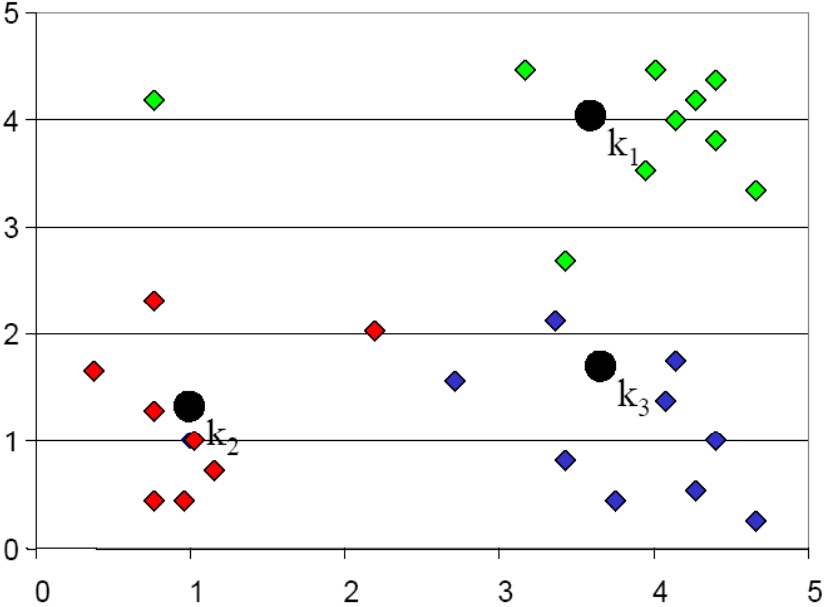
𝑧𝑧𝑖𝑖 = 𝑎𝑎𝑎𝑎𝑎𝑎𝑎𝑎𝑎𝑎𝑛𝑛𝑘𝑘𝑑𝑑(𝑐𝑐𝑘𝑘, 𝑥𝑥𝑖𝑖) *Minimize intra distance*

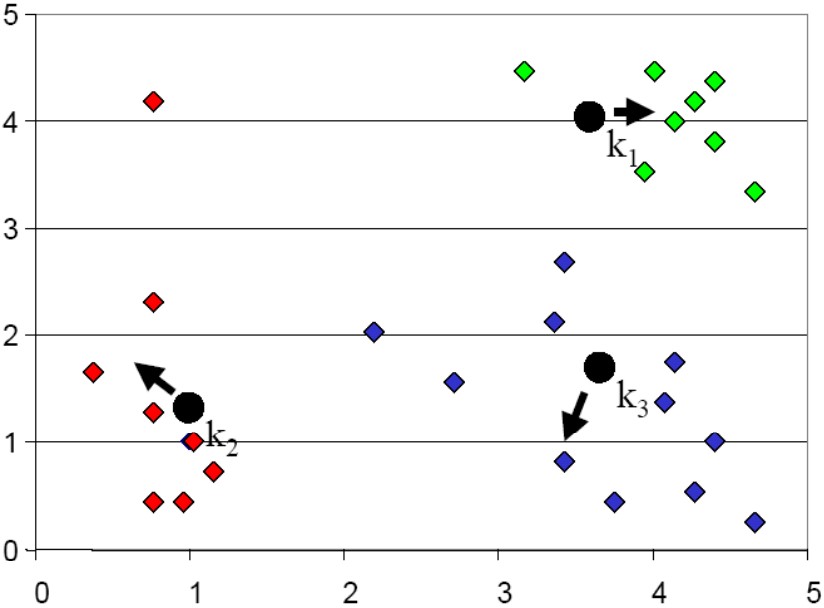
– Update the *k* cluster centroids based on the assigned cluster membership

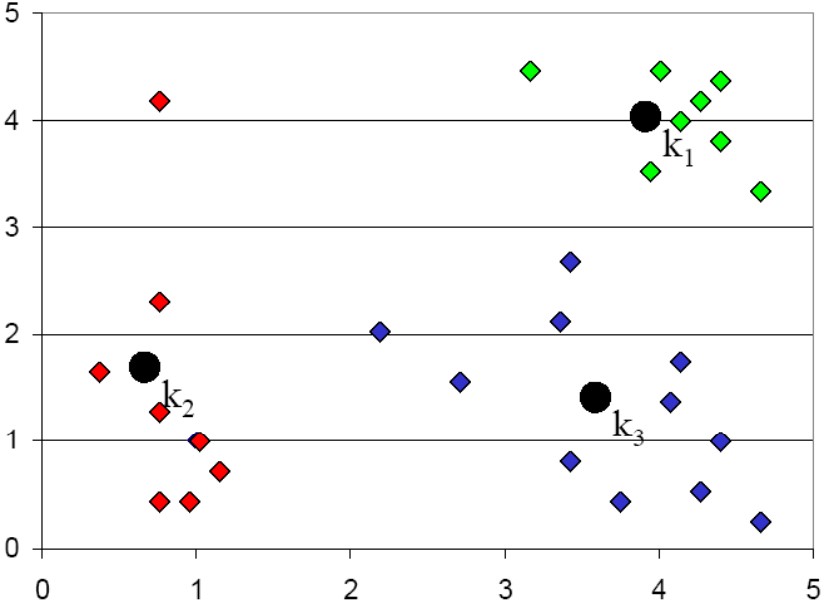
𝑐𝑐𝑘𝑘 =∑∑𝑖𝑖𝑖𝑖𝛿𝛿𝛿𝛿𝑧𝑧(𝑧𝑧𝑖𝑖𝑖𝑖==𝑐𝑐𝑐𝑐𝑘𝑘𝑘𝑘𝑥𝑥)𝑖𝑖 *Maximize inter distance*



**Voronoi diagram**





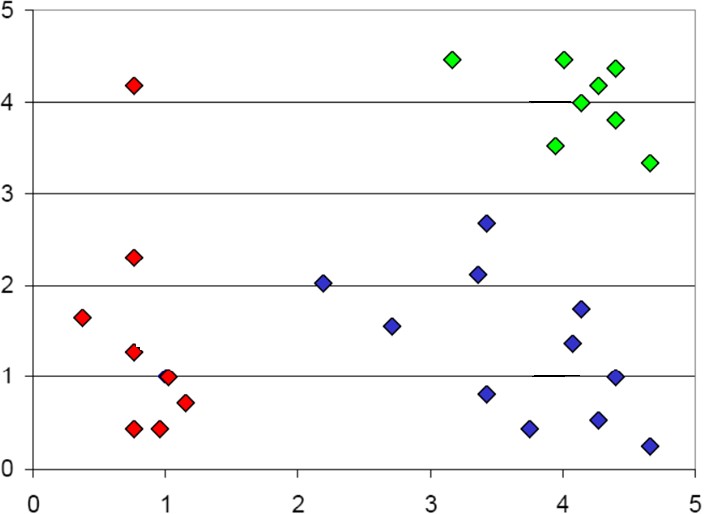


# Complexity analysis

* Decide cluster membership
  + 𝑂𝑂(𝑘𝑘𝑛𝑛) *Don’t forget the complexity of*
* Compute cluster centroid *distance computation, e.g.,* 𝑂𝑂(𝑉𝑉) *for Euclidean distance* – 𝑂𝑂(𝑛𝑛)
* Assume *k*-means stops after 𝑙𝑙 iterations
  + 𝑂𝑂(𝑘𝑘𝑛𝑛𝑙𝑙)

# Convergence property

* Why will *k*-means stop?
  + Answer: it is a special version of Expectation Maximization (EM) algorithm, and EM is guaranteed to converge
  + However, it is only guaranteed to converge to local optimal, since *k*-means (EM) is a greedy algorithm
* The density model of 𝑝𝑝 𝑥𝑥 is multi-modal • Each mode represents a sub-population
  + E.g., unimodal Gaussian for each group



Mixture model

𝑝𝑝

𝑥𝑥

=

𝑧𝑧

𝑝𝑝

𝑥𝑥

𝑧𝑧

𝑝𝑝

(

𝑧𝑧

)

Unimodal distribution

𝑝𝑝

(

𝑥𝑥

|

𝑧𝑧

=

1)

𝑝𝑝

(

𝑥𝑥

|

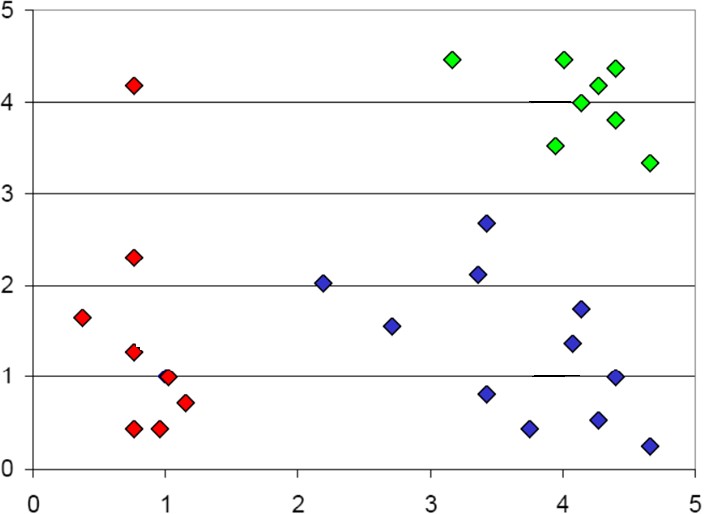
𝑧𝑧

=

2)

𝑝𝑝(𝑥𝑥|𝑧𝑧 3)

* If 𝑧𝑧 is known for every 𝑥𝑥
  + Estimating 𝑝𝑝(𝑧𝑧) and 𝑝𝑝(𝑥𝑥|𝑧𝑧) is easy
* Maximum likelihood estimation
* This is Naïve Bayes



Mixture model

𝑝𝑝

𝑥𝑥

=

𝑧𝑧

𝑝𝑝

𝑥𝑥

𝑧𝑧

𝑝𝑝

(

𝑧𝑧

)

Unimodal distribution

𝑝𝑝

(

𝑥𝑥

|

𝑧𝑧

=

1)

𝑝𝑝

(

𝑥𝑥

|

𝑧𝑧

=

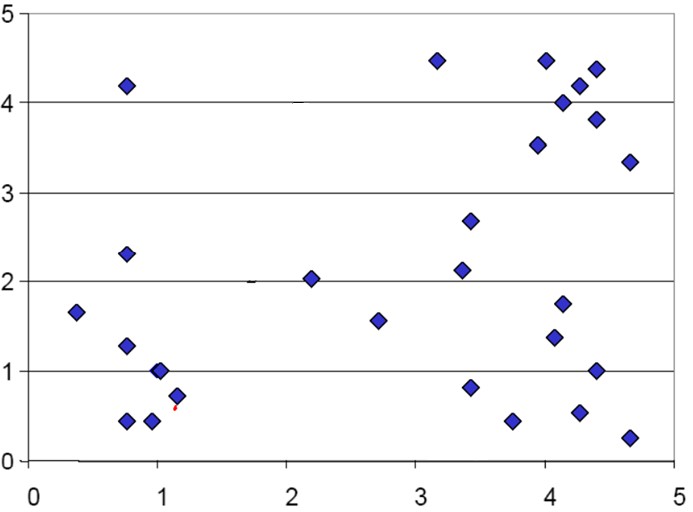
2)

𝑝𝑝(𝑥𝑥|𝑧𝑧 3)

* But 𝑧𝑧 is unknown for all 𝑥𝑥 Usually a constrained optimization problem
  + Estimating 𝑝𝑝(𝑧𝑧) and 𝑝𝑝(𝑥𝑥|𝑧𝑧) is generally hard
* max ∑𝑖𝑖 log ∑𝑧𝑧𝑖𝑖 𝑝𝑝 𝑥𝑥𝑖𝑖 𝑧𝑧𝑖𝑖, 𝛽𝛽 𝑝𝑝(𝑧𝑧𝑖𝑖|𝛼𝛼)

𝛼𝛼,𝛽𝛽

– Appeal to the Expectation Maximization algorithm



Mixture model

𝑝𝑝

𝑥𝑥

=

𝑧𝑧

𝑝𝑝

𝑥𝑥

𝑧𝑧

𝑝𝑝

(

𝑧𝑧

)

Unimodal distribution

𝑝𝑝

𝑥𝑥

𝑧𝑧

=

1

?

𝑝𝑝

𝑥𝑥

𝑧𝑧

=

2

?

𝑝𝑝

𝑥𝑥

𝑧𝑧

3

?

# Introduction to EM

* Parameter estimation

– All data is observable • Maximum likelihood estimator

* Optimize the analytic form of 𝐿𝐿 𝜃𝜃 = log𝑝𝑝(𝑋𝑋|𝜃𝜃)

– Missing/unobservable data*E.g. cluster membership*

* Data: X (observed) + Z (hidden)
* Likelihood: 𝐿𝐿 𝜃𝜃 = log ∑𝑧𝑧 𝑝𝑝 𝑋𝑋, 𝑍𝑍 𝜃𝜃
* Approximate it!

*Most of cases are intractable*

# Background knowledge

• Jensen's inequality

– For any convex function 𝑓𝑓(𝑥𝑥) and positive weights

𝜆𝜆

,

𝑓𝑓

𝑖𝑖

𝜆𝜆

𝑖𝑖

𝑥𝑥

𝑖𝑖

≤

𝑖𝑖

𝜆𝜆

𝑖𝑖

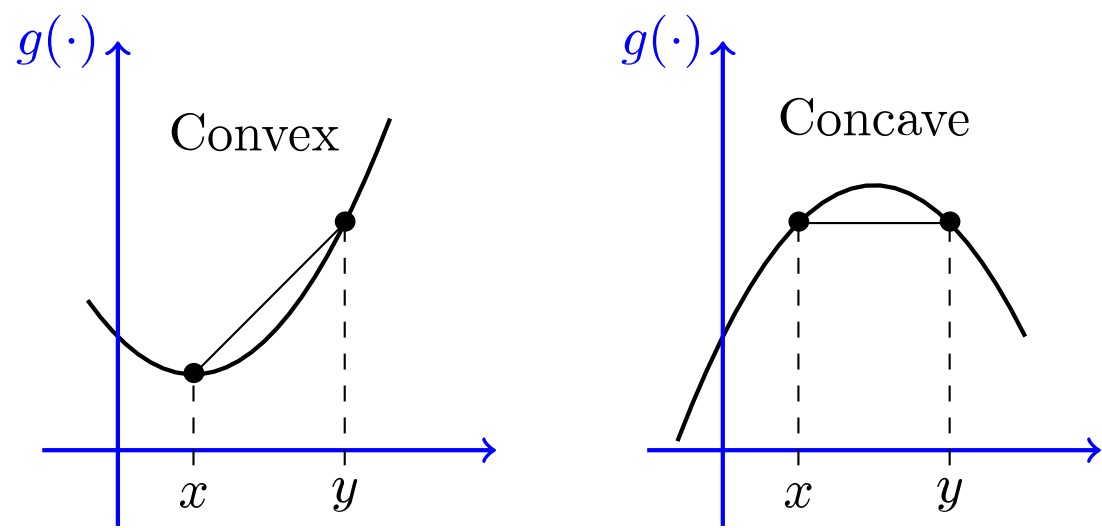
𝑓𝑓

(

𝑥𝑥

𝑖𝑖

)



𝑖𝑖

𝜆𝜆

𝑖𝑖

=

1

# Expectation Maximization

• Maximize data likelihood function by pushing the lower bound Proposal distributions for 𝑍𝑍

–

𝐿𝐿

𝜃𝜃

=

log

∑

𝑍𝑍

𝑝𝑝

𝑋𝑋

,Z

𝜃𝜃

=

log

∑

𝑍𝑍

𝑞𝑞

𝑍𝑍

𝑝𝑝

𝑋𝑋

,Z

𝜃𝜃

𝑞𝑞

𝑍𝑍

≥

𝑍𝑍

𝑞𝑞

𝑍𝑍

log

𝑝𝑝

𝑋𝑋

,

𝑍𝑍

𝜃𝜃

−

𝑍𝑍

𝑞𝑞

𝑍𝑍

log

𝑞𝑞

(

𝑍𝑍

)

Jensen's inequality

𝑓𝑓

𝐸𝐸

𝑥𝑥

≥

𝐸𝐸

[

𝑓𝑓

(

𝑥𝑥

)]

***Lower bound!***

**Components we need to tune when optimizing** 𝑳𝑳 𝜽𝜽 **:** 𝒒𝒒(𝒁𝒁) **and** 𝜽𝜽!

# Intuitive understanding of EM

*Data likelihood p(X|*

θ

*)*

*Lower bound*

Easier to optimize, guarantee

to improve data likelihood

θ

* Optimize the lower bound w.r.t. 𝑞𝑞(𝑍𝑍)

–

𝐿𝐿

𝜃𝜃

≥

∑

𝑍𝑍

𝑞𝑞

𝑍𝑍

𝑙𝑙𝑙𝑙

𝑎𝑎

𝑝𝑝

𝑋𝑋

,

𝑍𝑍

𝜃𝜃

−

∑

𝑍𝑍

𝑞𝑞

𝑍𝑍

𝑙𝑙𝑙𝑙

𝑎𝑎

𝑞𝑞

(

𝑍𝑍

)

=

𝑍𝑍

𝑞𝑞

𝑍𝑍

log

𝑝𝑝

𝑍𝑍

𝑋𝑋

,

𝜃𝜃

+

log

𝑝𝑝

(

𝑋𝑋

|

𝜃𝜃

)

−

𝑍𝑍

𝑞𝑞

𝑍𝑍

log

𝑞𝑞

(

𝑍𝑍

)

=

𝑍𝑍

𝑞𝑞

𝑍𝑍

log

𝑝𝑝

𝑍𝑍

𝑋𝑋

,

𝜃𝜃

𝑞𝑞

(

𝑍𝑍

)

+

log

𝑝𝑝

(

𝑋𝑋

|

𝜃𝜃

)

negative KL-divergence between 𝑞𝑞(𝑍𝑍) and 𝑝𝑝 𝑍𝑍 𝑋𝑋, 𝜃𝜃 Constant with respect to 𝑞𝑞(𝑍𝑍)

𝑇𝑇(𝑥𝑥)

𝐾𝐾𝐿𝐿(𝑇𝑇| 𝑄𝑄 = 𝑇𝑇 𝑥𝑥 log 𝑑𝑑𝑥𝑥

𝑄𝑄(𝑥𝑥)

* Optimize the lower bound w.r.t. 𝑞𝑞(𝑍𝑍)
  + 𝐿𝐿 𝜃𝜃 ≥−𝐾𝐾𝐿𝐿𝑞𝑞(𝑍𝑍)||𝑝𝑝 𝑍𝑍 𝑋𝑋, 𝜃𝜃 + 𝐿𝐿(𝜃𝜃)
  + KL-divergence is non-negative, and equals to zero i.f.f.

𝑞𝑞 𝑍𝑍 = 𝑝𝑝 𝑍𝑍 𝑋𝑋, 𝜃𝜃

* + A step further: when 𝑞𝑞 𝑍𝑍 = 𝑝𝑝 𝑍𝑍 𝑋𝑋, 𝜃𝜃, we will get 𝐿𝐿 𝜃𝜃 ≥ 𝐿𝐿(𝜃𝜃), i.e., the lower bound is tight!
  + Other choice of 𝑞𝑞 𝑍𝑍 cannot lead to this tight bound, but might reduce computational complexity
  + **Note**: calculation of 𝑞𝑞 𝑍𝑍 is based on current 𝜃𝜃
* Optimize the lower bound w.r.t. 𝑞𝑞(𝑍𝑍)
  + Optimal solution: 𝑞𝑞 𝑍𝑍 = 𝑝𝑝𝑍𝑍|𝑋𝑋, 𝜃𝜃𝑡𝑡

Posterior distribution of 𝑍𝑍 given current model 𝜃𝜃𝑡𝑡

*In k-means: this corresponds to assigning*

*instance* 𝑥𝑥𝑖𝑖 *to its closest cluster centroid* 𝑐𝑐𝑘𝑘

𝑧𝑧𝑖𝑖 = 𝑎𝑎𝑎𝑎𝑎𝑎𝑎𝑎𝑎𝑎𝑛𝑛𝑘𝑘𝑑𝑑(𝑐𝑐𝑘𝑘, 𝑥𝑥𝑖𝑖)

* Optimize the lower bound w.r.t. 𝜃𝜃
  + 𝐿𝐿 𝜃𝜃 ≥ ∑𝑍𝑍𝑝𝑝 𝑍𝑍|𝑋𝑋, 𝜃𝜃𝑡𝑡 log 𝑝𝑝 𝑋𝑋, 𝑍𝑍 𝜃𝜃 −

∑𝑍𝑍𝑝𝑝 𝑍𝑍|𝑋𝑋, 𝜃𝜃𝑡𝑡 log 𝑝𝑝 𝑍𝑍|𝑋𝑋, 𝜃𝜃𝑡𝑡 Constant w.r.t. 𝜃𝜃

* + 𝜃𝜃𝑡𝑡+1 = 𝑎𝑎𝑎𝑎𝑎𝑎𝑎𝑎𝑎𝑎𝑥𝑥𝜃𝜃 ∑𝑍𝑍𝑝𝑝 𝑍𝑍|𝑋𝑋, 𝜃𝜃𝑡𝑡 𝑙𝑙𝑙𝑙𝑎𝑎𝑝𝑝𝑋𝑋, 𝑍𝑍 𝜃𝜃

= 𝑎𝑎𝑎𝑎𝑎𝑎𝑎𝑎𝑎𝑎𝑥𝑥𝜃𝜃𝐸𝐸𝑍𝑍|𝑋𝑋,𝜃𝜃𝑡𝑡 log 𝑝𝑝(𝑋𝑋, 𝑍𝑍|𝜃𝜃)

**Expectation of complete data likelihood**

*In k-means, we are not computing the expectation, but the most probable*

*configuration, and then* 𝑐𝑐𝑘𝑘 = ∑∑𝑖𝑖𝑖𝑖𝛿𝛿𝛿𝛿𝑧𝑧(𝑧𝑧𝑖𝑖=𝑖𝑖=𝑐𝑐𝑐𝑐𝑘𝑘𝑘𝑘𝑥𝑥)𝑖𝑖

# Expectation Maximization

• EM tries to iteratively maximize likelihood

* “Complete” likelihood: 𝐿𝐿𝑐𝑐 𝜃𝜃 = log𝑝𝑝(𝑋𝑋, Z|𝜃𝜃)
* Starting from an initial guess θ(0),
  1. **E-step**: compute the expectation of the complete likelihood

𝑄𝑄𝜃𝜃; 𝜃𝜃𝑡𝑡 = E𝑍𝑍|𝑋𝑋,𝜃𝜃𝑡𝑡 𝐿𝐿𝑐𝑐 𝜃𝜃 = 𝑝𝑝 𝑍𝑍 𝑋𝑋, 𝜃𝜃𝑡𝑡 log p X, Z 𝜃𝜃

𝑍𝑍

* 1. **M-step**: compute θ(t+1) by maximizing the Q-function

𝜃𝜃𝑡𝑡+1 = 𝑎𝑎𝑎𝑎𝑎𝑎𝑎𝑎𝑎𝑎𝑥𝑥𝜃𝜃𝑄𝑄𝜃𝜃; 𝜃𝜃𝑡𝑡 ***Key step!***

# An intuitive understanding of EM

*Data likelihood p(X|*

θ

*)*

*current guess*

*Lower bound*

*)*

*Q function*

*(*

*next guess*

In

*k*

means

-

•

E-step: identify the cluster

membership -

𝑝𝑝

𝑧𝑧

𝑥𝑥

,

𝑐𝑐

•

M-step: update

𝑐𝑐

by

𝑝𝑝

𝑧𝑧

𝑥𝑥

,

𝑐𝑐

θ

*E-step = computing the lower bound M-step = maximizing the lower bound*

*30*

# Convergence guarantee

• Proof of EM

log 𝑝𝑝 𝑋𝑋 𝜃𝜃 = log 𝑝𝑝𝑍𝑍, 𝑋𝑋 𝜃𝜃 − log 𝑝𝑝(𝑍𝑍|𝑋𝑋, 𝜃𝜃)

Taking expectation with respect to 𝑝𝑝(𝑍𝑍|𝑋𝑋, 𝜃𝜃𝑡𝑡) of both sides: log 𝑝𝑝 𝑋𝑋 𝜃𝜃 = 𝑝𝑝(𝑍𝑍|𝑋𝑋, 𝜃𝜃𝑡𝑡) log 𝑝𝑝 𝑍𝑍, 𝑋𝑋 𝜃𝜃 − 𝑝𝑝(𝑍𝑍|𝑋𝑋, 𝜃𝜃𝑡𝑡) log 𝑝𝑝(𝑍𝑍|𝑋𝑋, 𝜃𝜃)

𝑍𝑍𝑍𝑍

= 𝑄𝑄𝜃𝜃; 𝜃𝜃𝑡𝑡 + 𝐻𝐻(𝜃𝜃; 𝜃𝜃𝑡𝑡) Cross-entropy

Then the change of log data likelihood between EM iteration is:

log 𝑝𝑝 𝑋𝑋 𝜃𝜃 − log 𝑝𝑝(𝑋𝑋|𝜃𝜃𝑡𝑡) = 𝑄𝑄 𝜃𝜃; 𝜃𝜃𝑡𝑡 + 𝐻𝐻 𝜃𝜃; 𝜃𝜃𝑡𝑡 −𝑄𝑄 𝜃𝜃𝑡𝑡; 𝜃𝜃𝑡𝑡−𝐻𝐻(𝜃𝜃𝑡𝑡; 𝜃𝜃𝑡𝑡)

By Jensen’s inequality, we know 𝐻𝐻 𝜃𝜃; 𝜃𝜃𝑡𝑡 ≥𝐻𝐻 𝜃𝜃𝑡𝑡; 𝜃𝜃𝑡𝑡 , that means log 𝑝𝑝 𝑋𝑋 𝜃𝜃 − log 𝑝𝑝(𝑋𝑋|𝜃𝜃𝑡𝑡) ≥𝑄𝑄 𝜃𝜃; 𝜃𝜃𝑡𝑡 −𝑄𝑄 𝜃𝜃𝑡𝑡; 𝜃𝜃𝑡𝑡 ≥ 0

*M-step guarantee this*

# What is not guaranteed

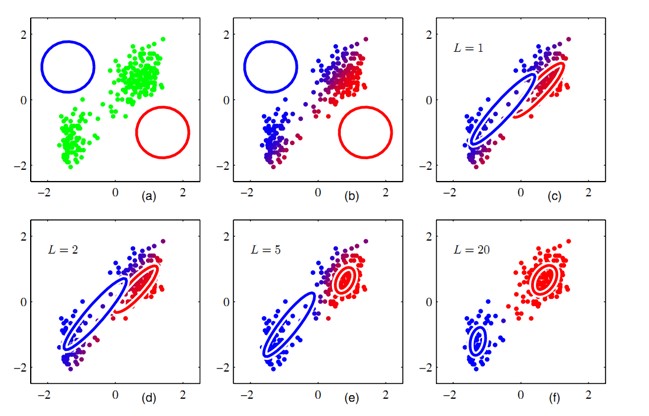
* Global optimal is not guaranteed!
  + Likelihood: 𝐿𝐿 𝜃𝜃 = log ∑𝑍𝑍𝑝𝑝𝑋𝑋, Z 𝜃𝜃 is non-convex in most of case
  + EM boils down to a greedy algorithm
* Alternative ascent
* Generalized EM
  + E-step: 𝑞𝑞 𝑍𝑍 = argmin𝑞𝑞(𝑍𝑍)𝐾𝐾𝐿𝐿𝑞𝑞(𝑍𝑍)||𝑝𝑝 𝑍𝑍 𝑋𝑋, 𝜃𝜃𝑡𝑡

𝑡𝑡

* + M-step: choose 𝜃𝜃 that improves 𝑄𝑄𝜃𝜃; 𝜃𝜃

*k*-means v.s. Gaussian Mixture

* If we use Euclidean distance in *k*-means
  + We have explicitly assumed 𝑝𝑝(𝑥𝑥|𝑧𝑧) is Gaussian
  + Gaussian Mixture Model (GMM)



•

𝑝𝑝

𝑥𝑥

𝑧𝑧

=

𝑁𝑁

𝜇𝜇

𝑧𝑧

,

Σ

𝑧𝑧

•

𝑝𝑝

𝑧𝑧

=

𝛼𝛼

𝑧𝑧

1

𝜋𝜋

2

𝑘𝑘

Σ

𝑧𝑧

𝑒𝑒

−

1

𝑥𝑥−𝜇𝜇

𝑧𝑧

𝑇𝑇

Σ

z

−1

𝑥𝑥−𝜇𝜇

𝑧𝑧

Multinomial

𝑇𝑇

𝑥𝑥

𝑧𝑧

=

1

𝜋𝜋

2

𝑒𝑒

−

𝑥𝑥−𝜇𝜇

𝑧𝑧

𝑇𝑇

(

𝑥𝑥−𝜇𝜇

𝑧𝑧

)

2

In

*k*

-

means, we assume

equal variance across

clusters, so we don’t

need to estimate them

We do not

consider cluster

size in

*k*

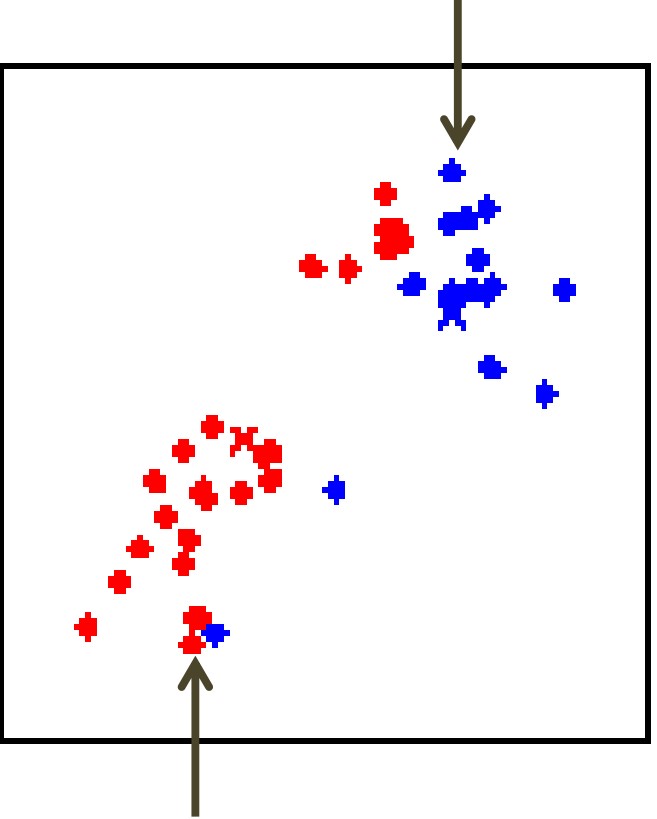
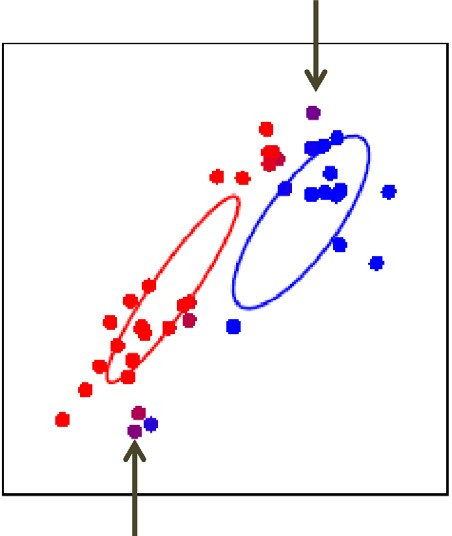
-

means

*k*-means v.s. Gaussian Mixture

* Soft v.s., hard posterior assignment

GMM *k*-means



# *k*-means in practice

* Extremely fast and scalable – One of the most popularly used clustering methods
* Top 10 data mining algorithms – ICDM 2006
* Can be easily parallelized
  + Map-Reduce implementation
* Mapper: assign each instance to its closest centroid – Reducer: update centroid based on the cluster membership
* Sensitive to initialization
  + Prone to local optimal

Better initialization: *k*-means++

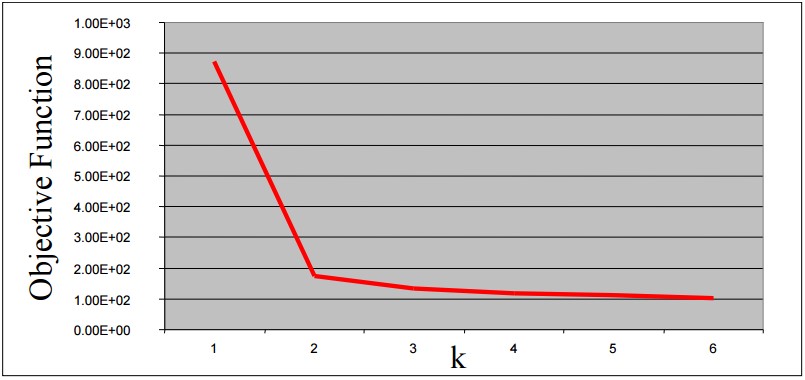
1. Choose the first cluster center at uniformly random
2. Repeat until all *k* centers have been found
   * For each instance compute Dx = min𝑑𝑑(𝑥𝑥, 𝑐𝑐𝑘𝑘) k
   * Choose a new cluster center with probability

𝑝𝑝 𝑥𝑥 ∝ 𝐷𝐷𝑥𝑥2  *new center should be far away from existing centers*

1. Run *k*-means with selected centers as initialization

How to determine *k*

* Vary 𝑘𝑘 to optimize clustering criterion
  + Internal v.s. external validation
  + Cross validation
* Abrupt change in objective function



# How to determine *k*

* Vary 𝑘𝑘 to optimize clustering criterion
  + Internal v.s. external validation
  + Cross validation
* Abrupt change in objective function
* Model selection criterion – penalizing too many clusters – AIC, BIC

What you should know

* *k*-means algorithm
* An alternative greedy algorithm
* Convergence guarantee
  + EM algorithm
* Hard clustering v.s., soft clustering
  + *k*-means v.s., GMM

# Today’s reading

* Introduction to Information Retrieval – Chapter 16: Flat clustering
* 16.4 *k*-means
* 16.5 Model-based clustering