Clustering.
Unsupervised Learning

硬聚类：每个点只可能是一种聚类.

Maria-Florina Balcan

04/06/2015

Reading:

- Chapter 14.3: Hastie, Tibshirani, Friedman.

Additional resources:

- Center Based Clustering: A Foundational Perspective.
  Awasthi, Balcan. Handbook of Clustering Analysis. 2015.

Clustering, Informal Goals

Goal: Automatically partition unlabeled data into groups of similar datapoints.

Question: When and why would we want to do this?

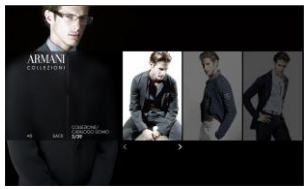
Useful for:

- Automatically organizing data. 数据管理.
- Understanding hidden structure in data.
- Preprocessing for further analysis.
    - Representing high-dimensional data in a low-dimensional space (e.g., for visualization purposes).

Applications (Clustering comes up everywhere…)

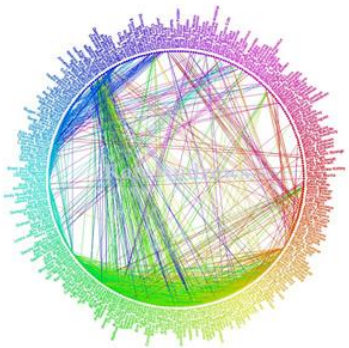
- Cluster news articles or web pages or search results by topic.



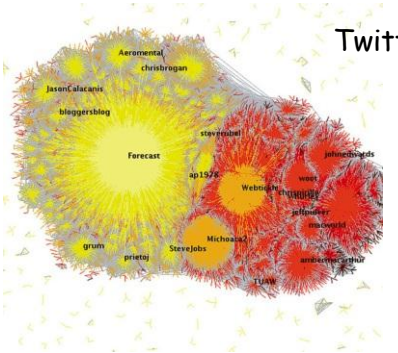


- Cluster protein sequences by function or genes according to expression profile.


- Cluster users of social networks by interest (community detection).



Facebook network



Twitter Network

Applications (Clustering comes up everywhere…)

- Cluster customers according to purchase history.

- Cluster galaxies or nearby stars (e.g. Sloan Digital Sky Survey)

- Image segmentation (clustering pixels)

Clustering

Today:

- Objective based clustering
- Hierarchical clustering
- Mention overlapping clusters

[March 4th: EM-style algorithm for clustering for mixture of Gaussians (specific probabilistic model).]

Objective Based Clustering

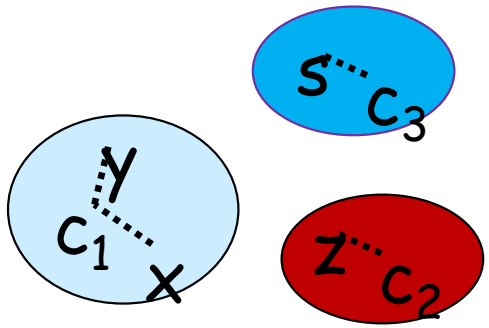
Input: A set S of n points, also a distance/dissimilarity measure specifying the distance d(x,y) between pairs (x,y).

两个样本点之间的距离.

E.g., # keywords in common, edit distance, wavelets coef., etc.

Goal: output a partition of the data.

- k-means: find center pts $c_1, c_2, \ldots, c_k$ to minimize $\sum_{i=1}^{n} \min_{j\in\{1,\ldots,k\}} d^2(\mathbf{x^i}, \mathbf{c_j})$
- k-median: find center pts $\mathbf{c_1}, \mathbf{c_2}, \ldots, \mathbf{c_k}$ to minimize $\sum_{i=1}^{n} \min_{j\in\{1,\ldots,k\}} d(\mathbf{x^i}, \mathbf{c_j})$



- K-center: find partition to minimize the maxim radius

Euclidean k-means Clustering

Input: A set of n datapoints $\mathbf{x^1}, \mathbf{x^2}, \ldots, \mathbf{x^n}$ in $\mathrm{R^d}$

target #clusters k

Output: k representatives $\boldsymbol{c_1}, \mathbf{c_2}, \ldots, \boldsymbol{c_k} \in \mathrm{R^d}$

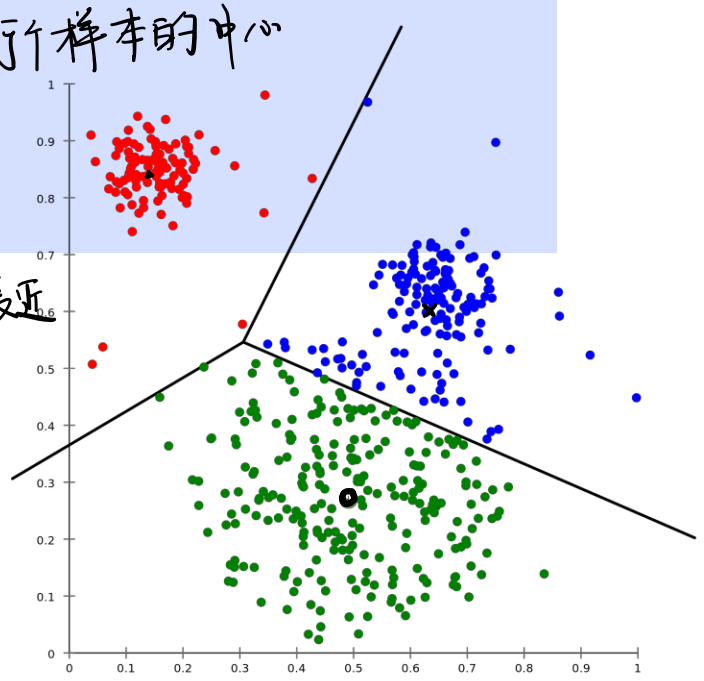
Objective: choose $\boldsymbol{c_1}, \mathbf{c_2}, \ldots, \boldsymbol{c_k} \in \mathrm{R^d}$ to minimize

$$\sum_{i=1}^{n} \min_{j \in \{1,\ldots,k\}} \left|\left|\mathbf{x^i} - \mathbf{c_j}\right|\right|^2$$

第j个样本的中心

但最开始未知 c_j，未知 i 和哪个点最近

⇒ 非凸且NP-hard ⇒ 依赖初始化



Euclidean k-means Clustering

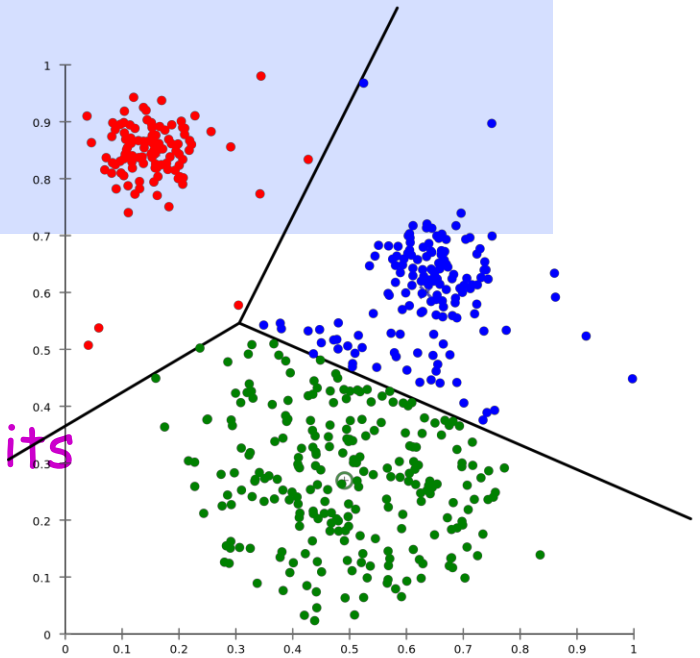
Input: A set of n datapoints $\mathbf{x^1}, \mathbf{x^2}, \ldots, \mathbf{x^n}$ in $\mathrm{R^d}$
target #clusters k

Output: k representatives $\boldsymbol{c_1}, \mathbf{c_2}, \ldots, \boldsymbol{c_k} \in \mathrm{R^d}$

Objective: choose $\boldsymbol{c_1}, \mathbf{c_2}, \ldots, \boldsymbol{c_k} \in \mathrm{R^d}$ to minimize

$$\sum_{i=1}^{n} \min_{j \in \{1,\ldots,k\}} \left|\left|\mathbf{x^i} - \mathbf{c_j}\right|\right|^2$$

Natural assignment: each point assigned to its closest center, leads to a Voronoi partition.



Euclidean k-means Clustering

简化目标函数：

定义 r_{ij}：x^i 属于 C_j 的指示变量，为0或取一或0.

$\Rightarrow$ 矩阵 R：$\{0,1\}^{n\times k}$ n行k列的矩阵.

约束：$R \times \mathbb{1}_k = \mathbb{1}_n$ （$\mathbb{1}_k$：全为1的k行列向量.）

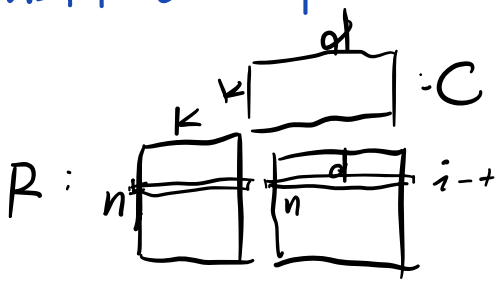
则目标函数可以化成 $\min_{R,C} \sum_{i=1}^{n}\sum_{j=1}^{k} r_{ij}\|x^i - C_j\|^2$ s.t. $R\times\mathbb{1}_k = \mathbb{1}_n$, $R\in\{0,1\}^{n\times k}$

还是一个NP-hard且非凸. 依赖于一个 C_j 的初始化.

矩阵化（去掉求和符号）：$\min_{R,C}\|X - RC\|^2$

C：所有的 C_j 组成的矩阵，$k\times d$.（有k个类，维度是d）

→ 矩阵范数：$\|A\|_F = \sqrt{Tr(A^TA)}$



$\Rightarrow C = C_j$

只有第j个元素是1（第j列是1其它均为0）

只有一个聚类.

只有k=1时是非闭式解.（均值）

Euclidean k-means Clustering

Input: A set of n datapoints $\mathbf{x^1}, \mathbf{x^2}, \ldots, \mathbf{x^n}$ in $\mathrm{R^d}$

target #clusters k

Output: k representatives $\boldsymbol{c_1}, \mathbf{c_2}, \ldots, \boldsymbol{c_k} \in \mathrm{R^d}$

Objective: choose $\boldsymbol{c_1}, \mathbf{c_2}, \ldots, \boldsymbol{c_k} \in \mathrm{R^d}$ to minimize

$$\sum_{i=1}^{n} \min_{j \in \{1,\ldots,k\}} \left|\left|\mathbf{x^i} - \mathbf{c_j}\right|\right|^2$$

Computational complexity:

NP hard: even for $k = 2$ [Dagupta'08] or

$d = 2$ [Mahajan-Nimbhorkar-Varadarajan09]

There are a couple of easy cases…

An Easy Case for k-means: k=1

Input: A set of n datapoints $\mathbf{x^1}, \mathbf{x^2}, \ldots, \mathbf{x^n}$ in R^d

Output: $\boldsymbol{c} \in R^d$ to minimize $\sum_{i=1}^{n} \left|\left|\mathbf{x^i} - \mathbf{c}\right|\right|^2$

Solution: The optimal choice is $\boldsymbol{\mu} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x^i}$

Idea: bias/variance like decomposition

$$\frac{1}{n}\sum_{i=1}^{n} \left|\left|\mathbf{x^i} - \mathbf{c}\right|\right|^2 = \left|\left|\boldsymbol{\mu} - \mathbf{c}\right|\right|^2 + \frac{1}{n}\sum_{i=1}^{n} \left|\left|\mathbf{x^i} - \boldsymbol{\mu}\right|\right|^2$$

Quiz

Avg k-means cost wrt c

Avg k-means cost wrt μ

So, the optimal choice for $\mathbf{c}$ is $\boldsymbol{\mu}$.

Common Heuristic in Practice: The Lloyd's method

[Least squares quantization in PCM, Lloyd, IEEE Transactions on Information Theory, 1982]

Input: A set of n datapoints $\mathbf{x^1}, \mathbf{x^2}, \ldots, \mathbf{x^n}$ in R^d

Initialize centers $\boldsymbol{c_1}, \mathbf{c_2}, \ldots, \boldsymbol{c_k} \in \mathrm{R}^d$ and clusters $\mathrm{C}_1, \mathrm{C}_2, \ldots, \mathrm{C}_k$ in any way.

Repeat until there is no further change in the cost.

E-step 给定当前的c判断属于哪个类别.

- For each j: $\mathrm{C}_j \leftarrow \{x \in S$ whose closest center is $\mathbf{c}_j\}$

M-step MLE优化θ

- For each j: $\mathbf{c}_j \leftarrow$ mean of C_j

K-means 假设：所有θ都是一样的. (3个聚类形状是一样的，且是一个凸集).

若是瑞士卷分布：  则需用 Spectral Clustering (K-means和GM均无法分类完全)

Common Heuristic in Practice: The Lloyd's method

[Least squares quantization in PCM, Lloyd, IEEE Transactions on Information Theory, 1982]

Input: A set of n datapoints $\mathbf{x^1}, \mathbf{x^2}, \ldots, \mathbf{x^n}$ in $\mathrm{R^d}$

Initialize centers $\boldsymbol{c_1}, \mathbf{c_2}, \ldots, \boldsymbol{c_k} \in \mathrm{R^d}$ and clusters $\mathrm{C_1}, \mathrm{C_2}, \ldots, \mathrm{C_k}$ in any way.

Repeat until there is no further change in the cost.

- For each j: $\mathrm{C_j} \leftarrow \{x \in S$ whose closest center is $\mathbf{c_j}\}$
- For each j: $\mathbf{c_j} \leftarrow$ mean of $\mathrm{C_j}$

Holding $\boldsymbol{c_1}, \mathbf{c_2}, \ldots, \boldsymbol{c_k}$ fixed, pick optimal $\mathrm{C_1}, \mathrm{C_2}, \ldots, \mathrm{C_k}$

Holding $\mathrm{C_1}, \mathrm{C_2}, \ldots, \mathrm{C_k}$ fixed, pick optimal $\boldsymbol{c_1}, \mathbf{c_2}, \ldots, \boldsymbol{c_k}$

保证3每一步都减小3目标函数

E-step: $D^{(t)} \leftarrow \arg\min \|X - R C^{(t-1)}\|^2$

假设 $QCR(R,C) = \min \|X - RC\|$ 则 $QCR^{(t)}(C^{(t-1)}) \le QCR^{(t-1)}(C^{(t-1)})$

t-th iteration: M-step: $C^{(t)} \leftarrow \arg\min_C \|X - R^{(t)}C\|_F^2 \Rightarrow Q(R^{(t)}, C^{(t)}) \le Q(R^{(t)}, C^{(t-1)})$ (交替优化)

Common Heuristic: The Lloyd's method

Input: A set of n datapoints $\mathbf{x^1}, \mathbf{x^2}, \ldots, \mathbf{x^n}$ in R^d

Initialize centers $\mathbf{c_1}, \mathbf{c_2}, \ldots, \mathbf{c_k} \in R^d$ and clusters $C_1, C_2, \ldots, C_k$ in any way.

Repeat until there is no further change in the cost.

- For each j: $C_j \leftarrow \{x \in S$ whose closest center is $\mathbf{c_j}\}$
- For each j: $\mathbf{c_j} \leftarrow$ mean of C_j

Note: it always converges.

- the cost always drops and
- there is only a finite #s of Voronoi partitions
  (so a finite # of values the cost could take)

Initialization for the Lloyd's method

Input: A set of n datapoints $\mathbf{x^1}, \mathbf{x^2}, \ldots, \mathbf{x^n}$ in R^d

Initialize centers $\mathbf{c_1}, \mathbf{c_2}, \ldots, \mathbf{c_k} \in R^d$ and

clusters $C_1, C_2, \ldots, C_k$ in any way.

Repeat until there is no further change in the cost.

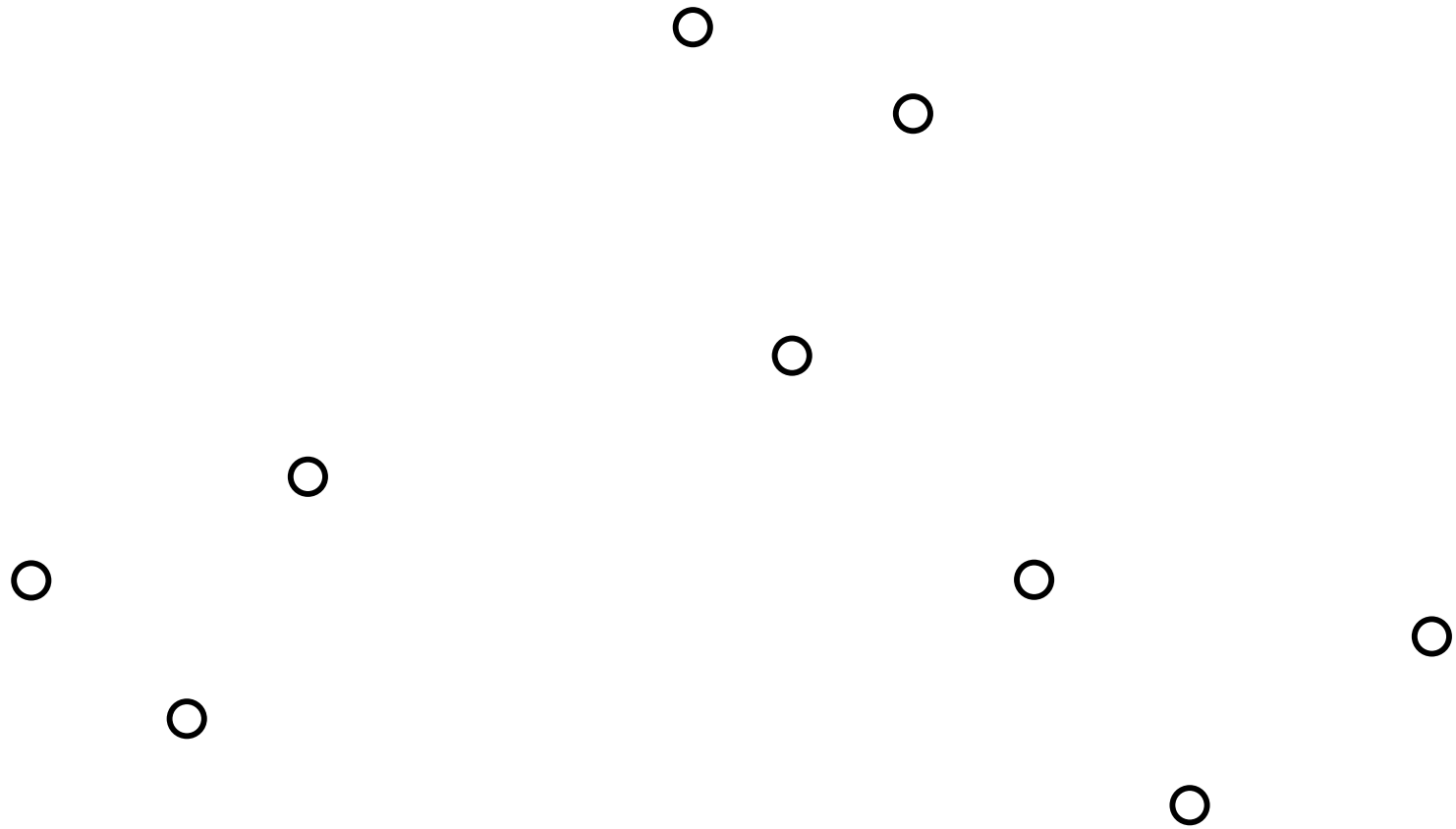
- For each j: $C_j \leftarrow \{x \in S$ whose closest center is $\mathbf{c_j}\}$
- For each j: $\mathbf{c_j} \leftarrow$ mean of C_j

- Initialization is crucial (how fast it converges, quality of solution output)
- Discuss techniques commonly used in practice
  - Random centers from the datapoints (repeat a few times) 随机初始化
  - Furthest traversal 最远点遍历.
  - K-means ++ (works well and has provable guarantees)

Lloyd's method: Random Initialization

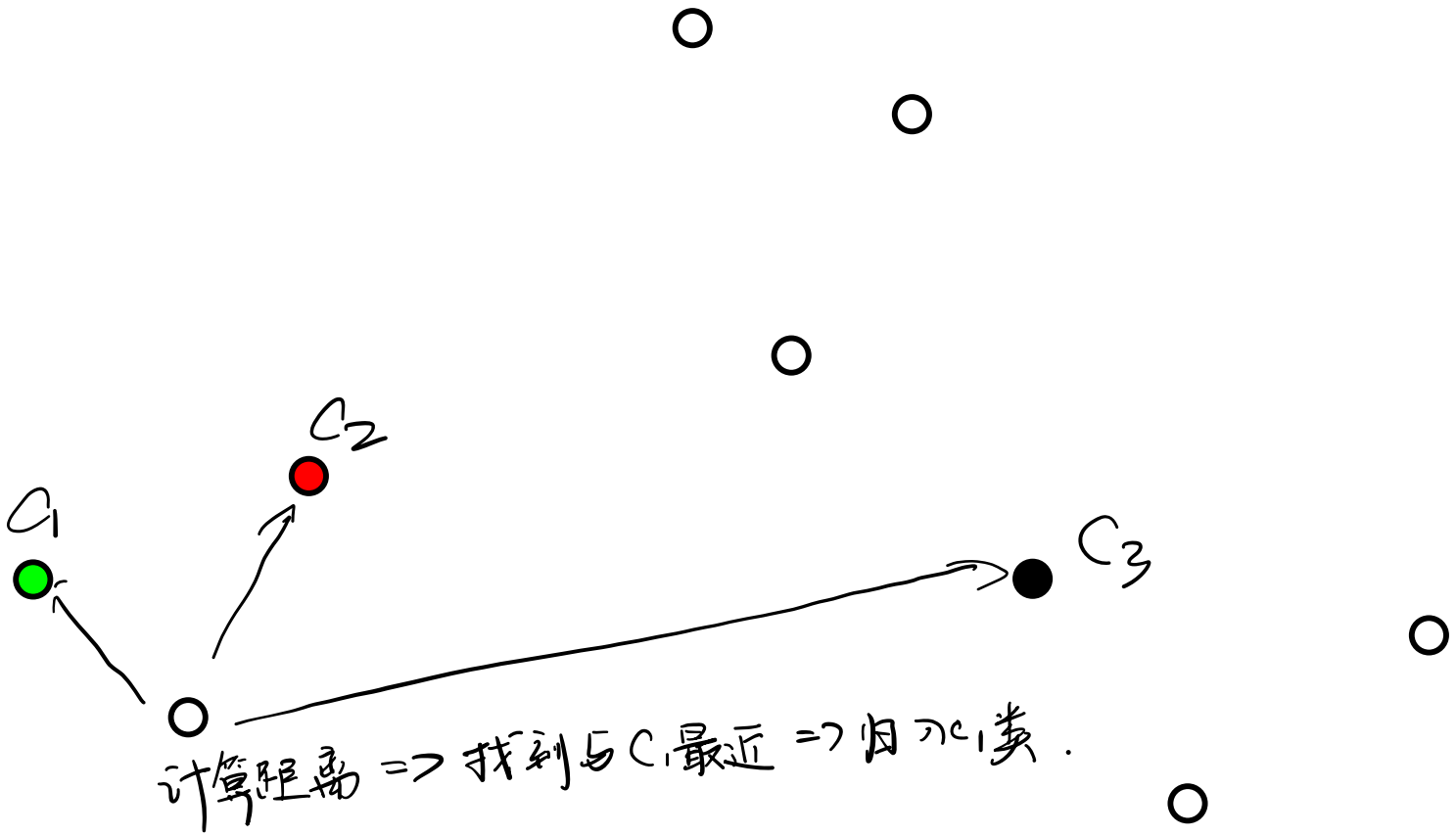
Lloyd's method: Random Initialization

Example: Given a set of datapoints



Lloyd's method: Random Initialization

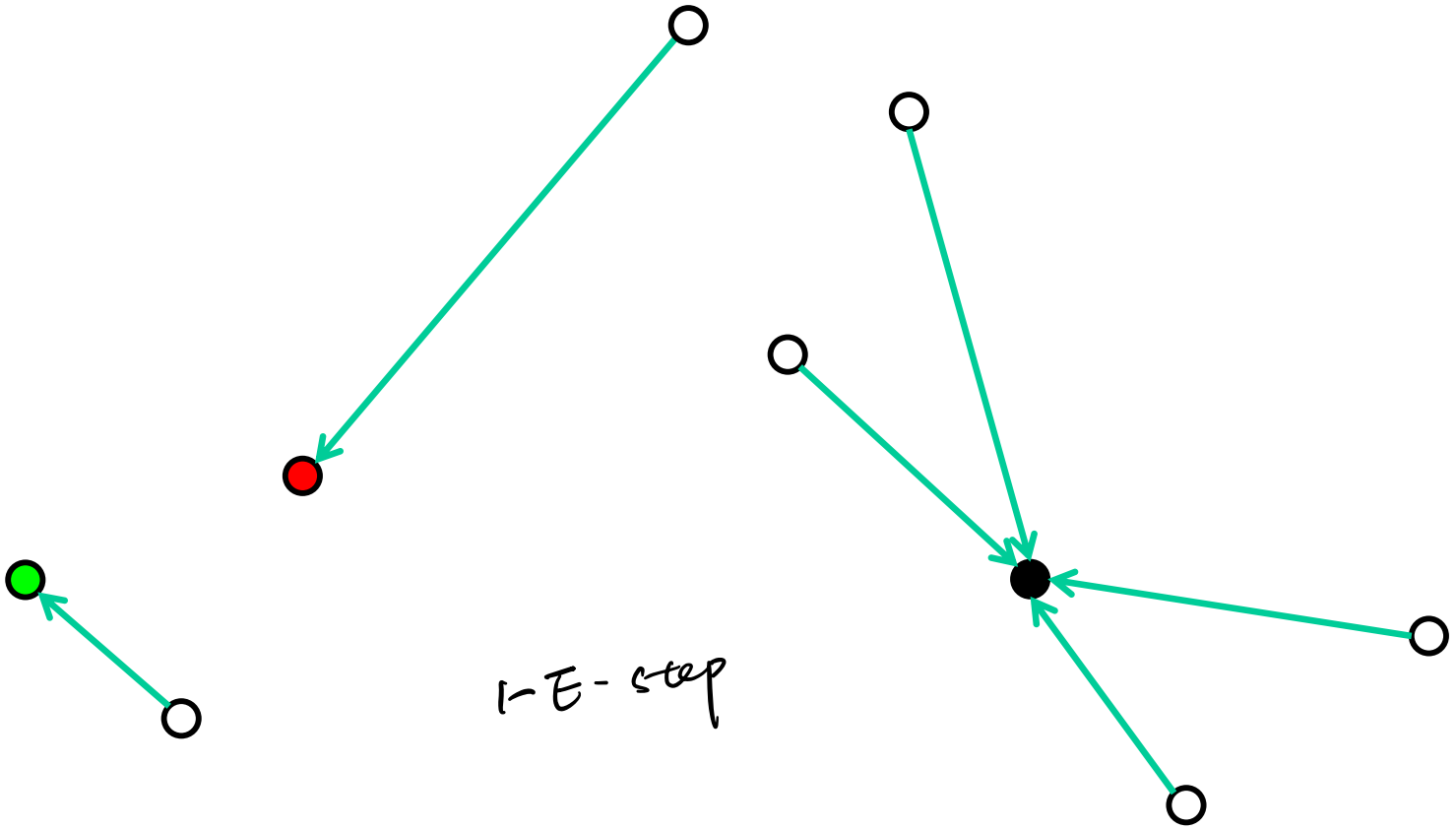
Select initial centers at random



Random Initialize

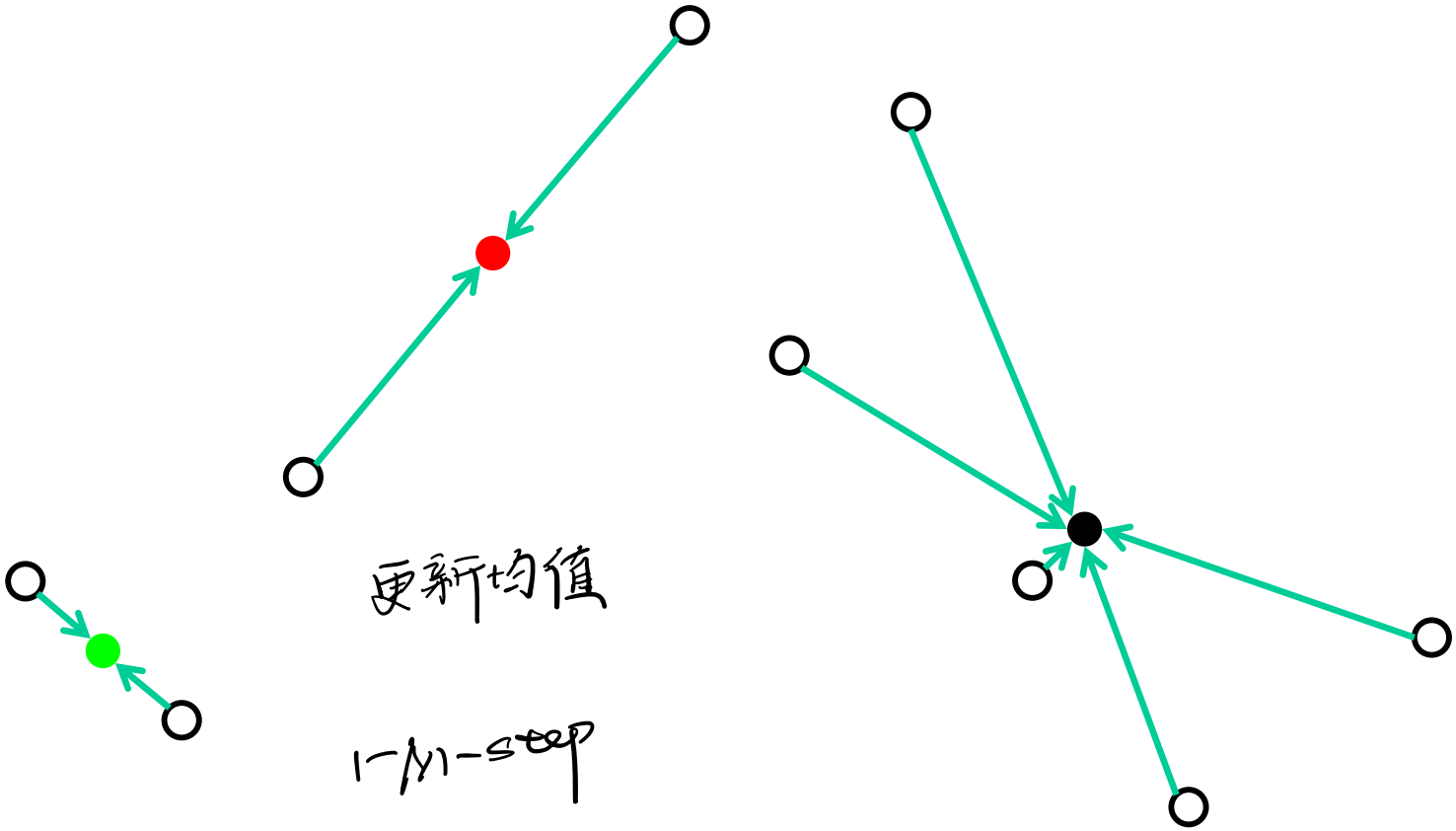
Lloyd's method: Random Initialization

Assign each point to its nearest center



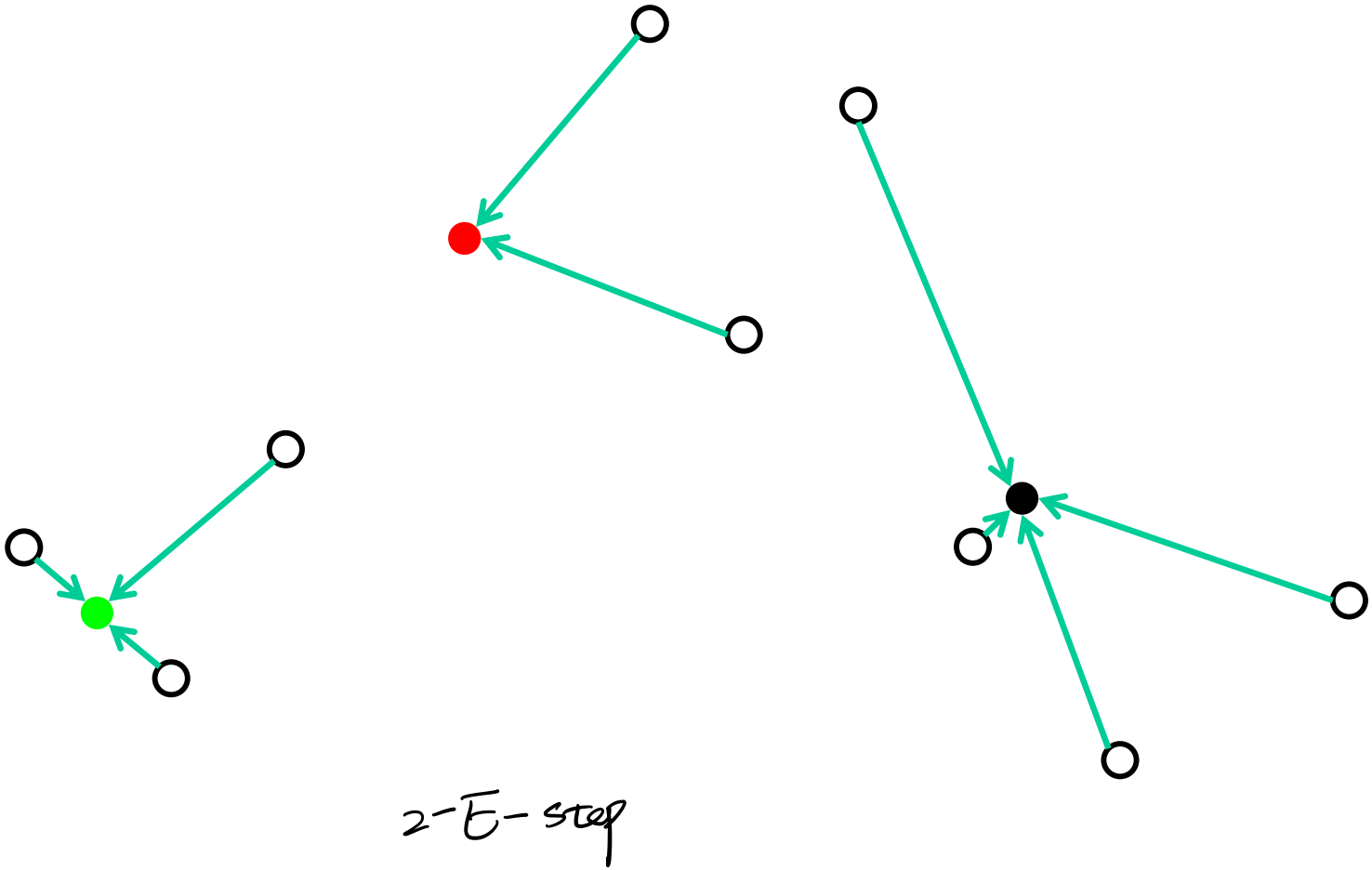
Lloyd's method: Random Initialization

Recompute optimal centers given a fixed clustering



Lloyd's method: Random Initialization

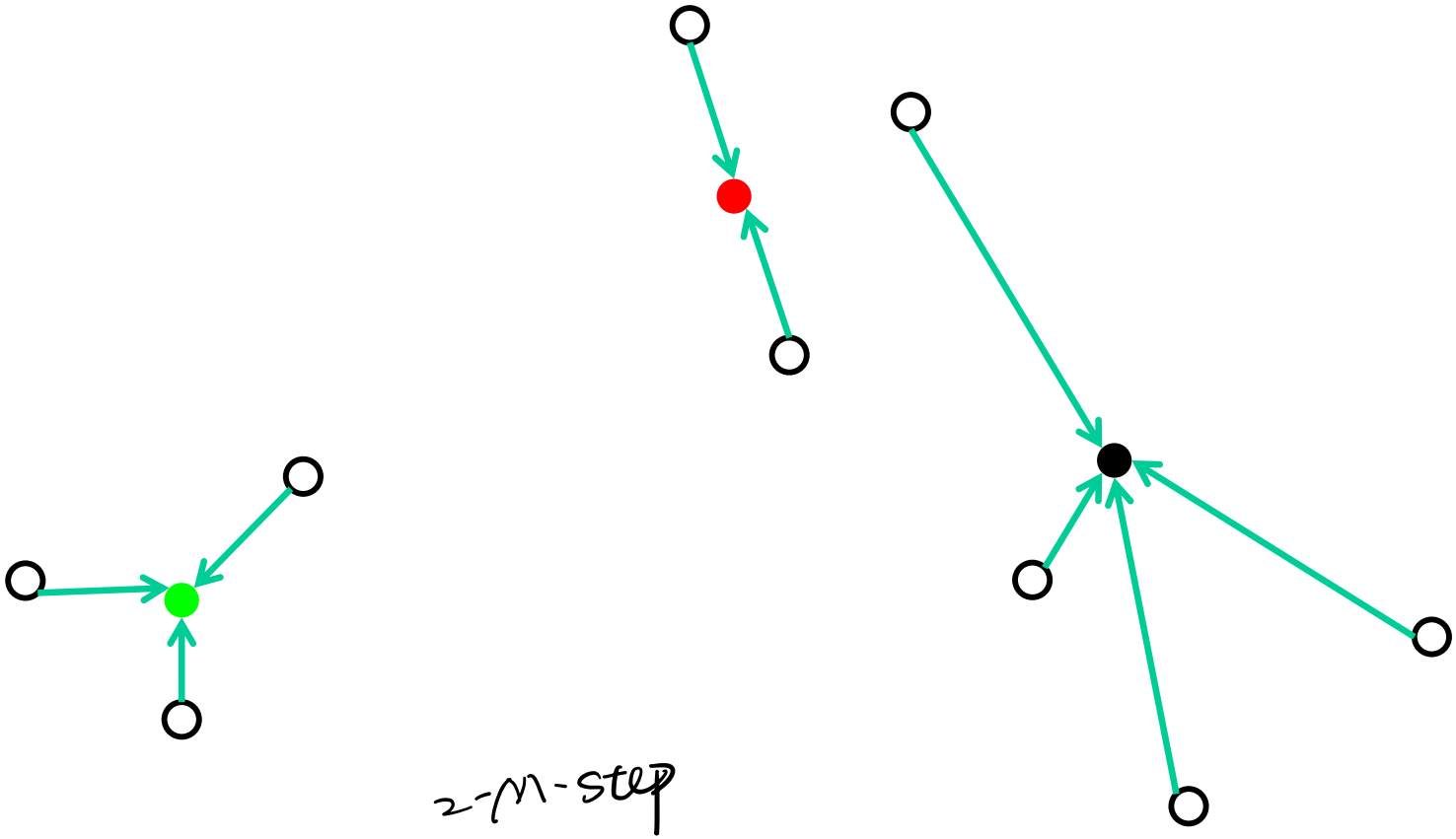
Assign each point to its nearest center



2-E-step

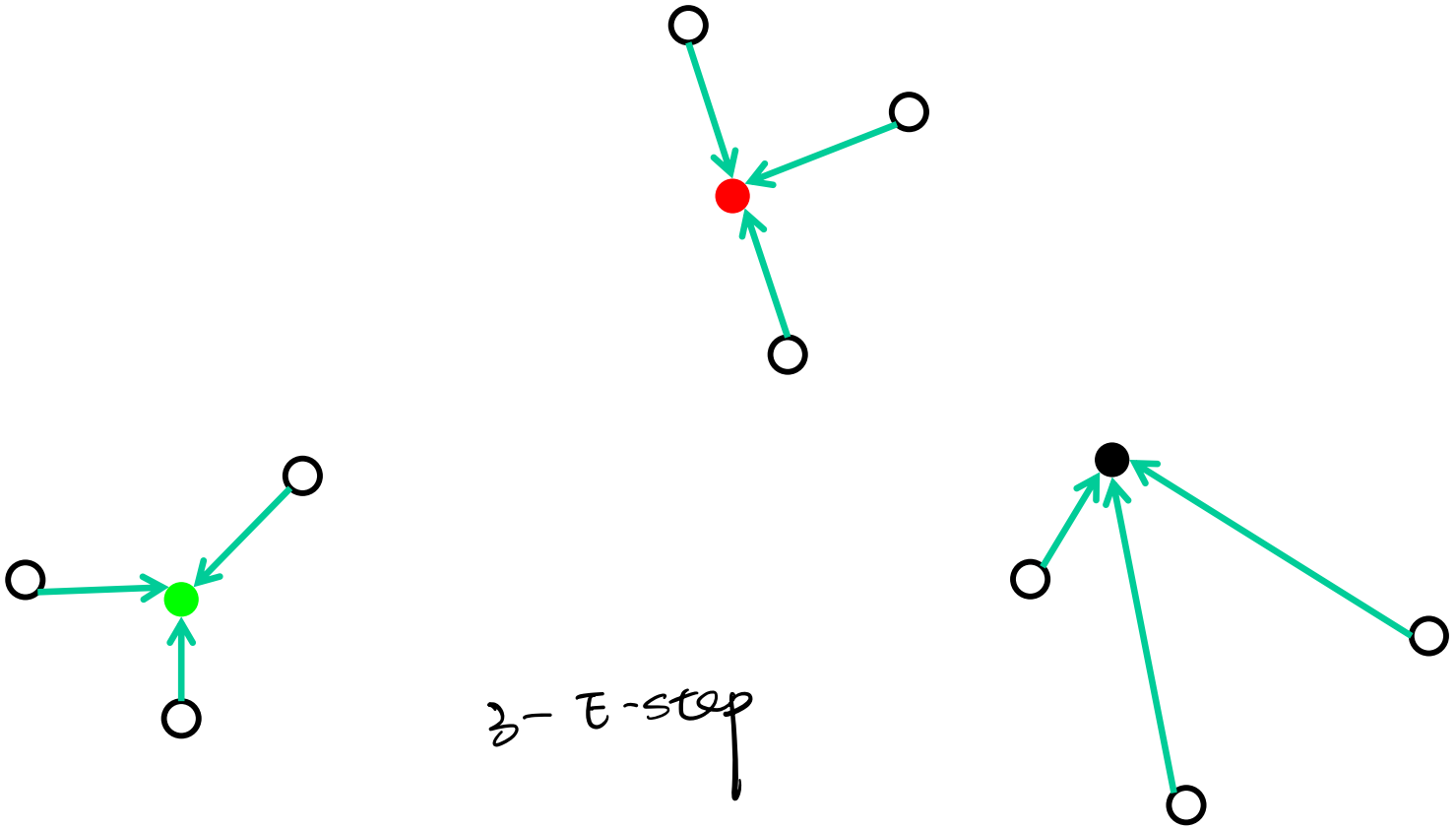
Lloyd's method: Random Initialization

Recompute optimal centers given a fixed clustering



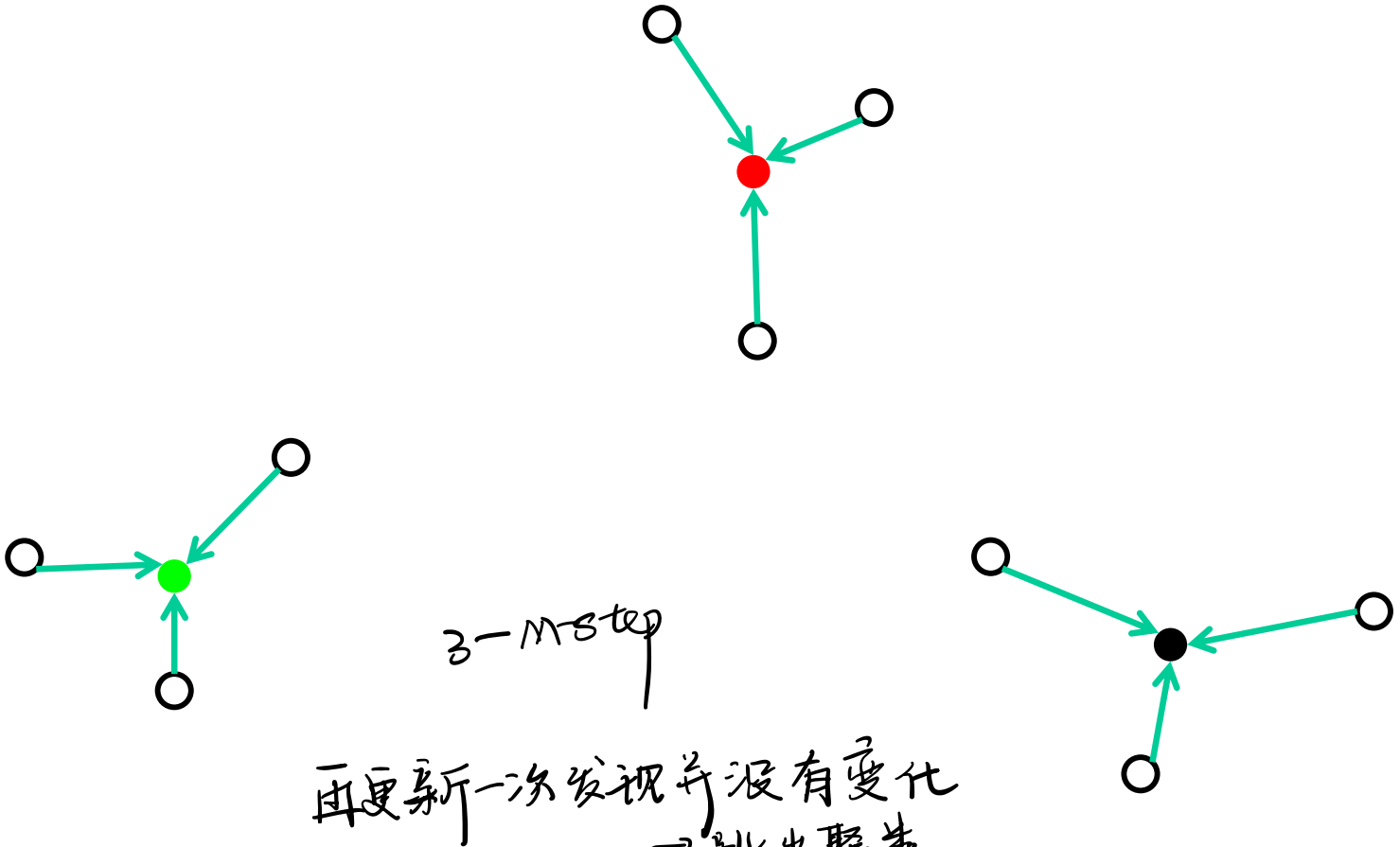
Lloyd's method: Random Initialization

Assign each point to its nearest center



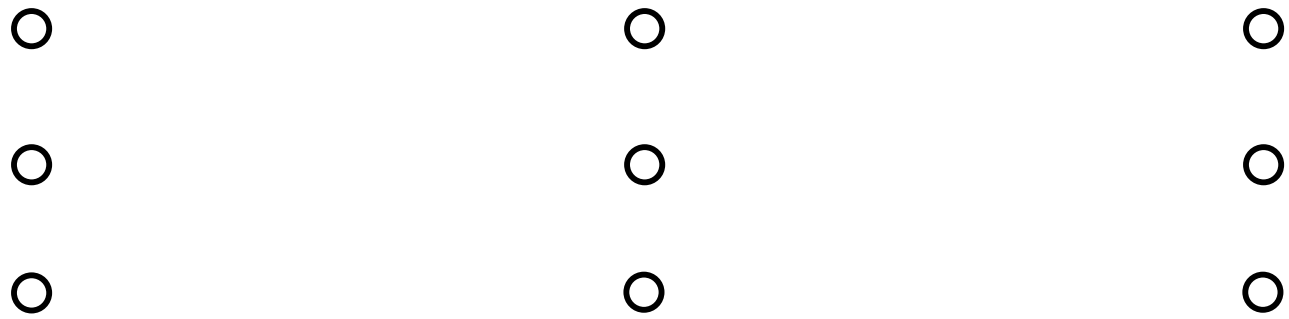
Lloyd's method: Random Initialization

Recompute optimal centers given a fixed clustering



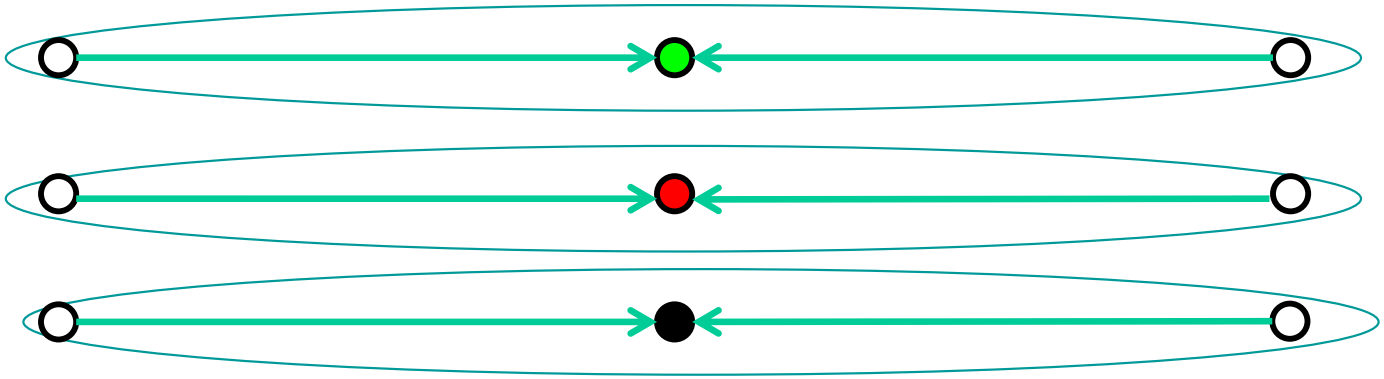
Get a good quality solution in this example.

Lloyd's method: Performance



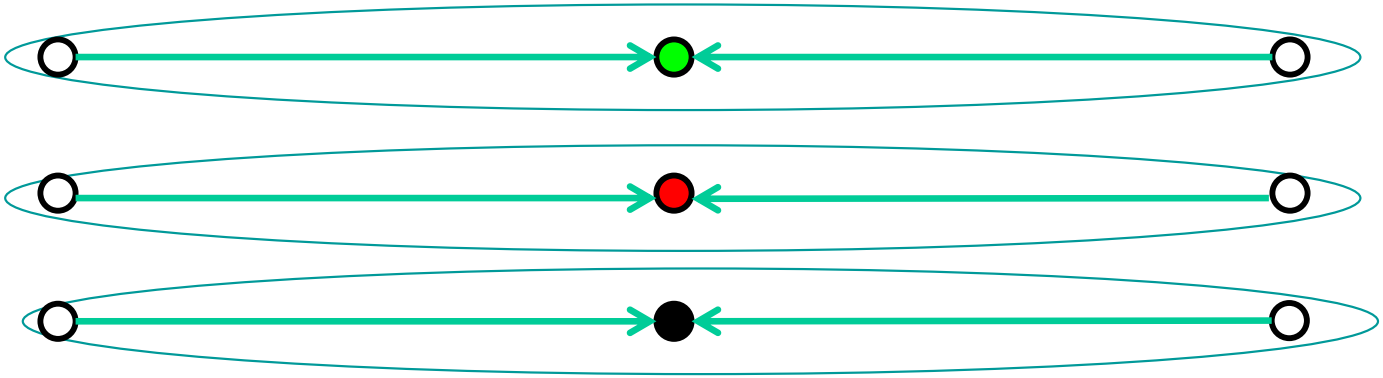
It always converges, but it may converge at a local optimum that is different from the global optimum, and in fact could be arbitrarily worse in terms of its score.

Lloyd's method: Performance

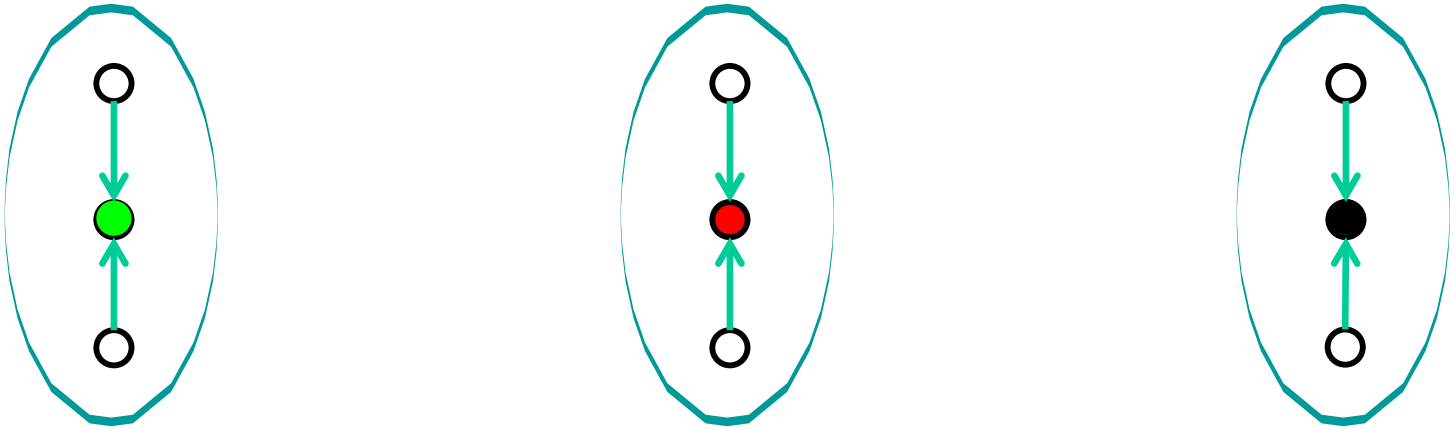


Local optimum: every point is assigned to its nearest center and every center is the mean value of its points.

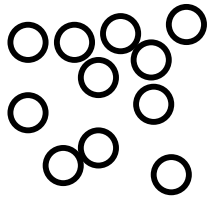
Lloyd's method: Performance

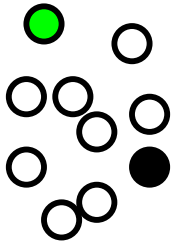


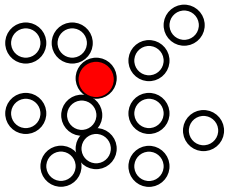
.It is arbitrarily worse than optimum solution….



Lloyd's method: Performance

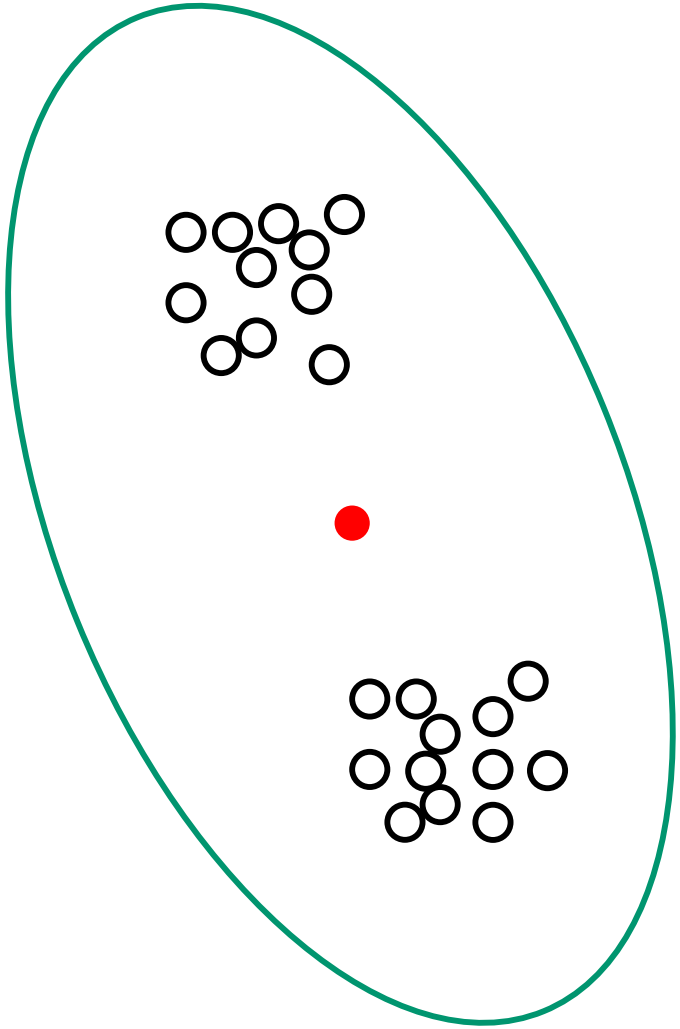






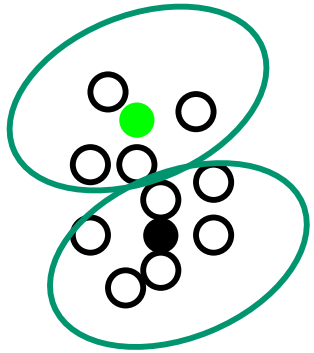
This bad performance, can happen even with well separated Gaussian clusters.

Lloyd's method: Performance



收敛于局部最优解.

但不是全局最优解.



This bad performance, can happen even with well separated Gaussian clusters.

Some Gaussian are combined.....

Lloyd's method: Performance

- If we do random initialization, as k increases, it becomes more likely we won't have perfectly picked one center per Gaussian in our initialization (so Lloyd's method will output a bad solution).

  认为k个 Gaussian Distribution 相同

  - For k equal-sized Gaussians, Pr[each initial center is in a different Gaussian] $\approx \frac{k!}{k^k} \approx \frac{1}{e^k}$ → 好的初始化的概率.
  - Becomes unlikely as k gets large. 随机初始化概率太小，故不用.

Another Initialization Idea: Furthest Point Heuristic

最远点遍历启发式.

第一次是Random. 然后每次，确定 C_1 后 C_2 是距 C_1 最远的点，C_3 是距 C_1 C_2 最远的点，...

Choose $\mathbf{c_1}$ arbitrarily (or at random).

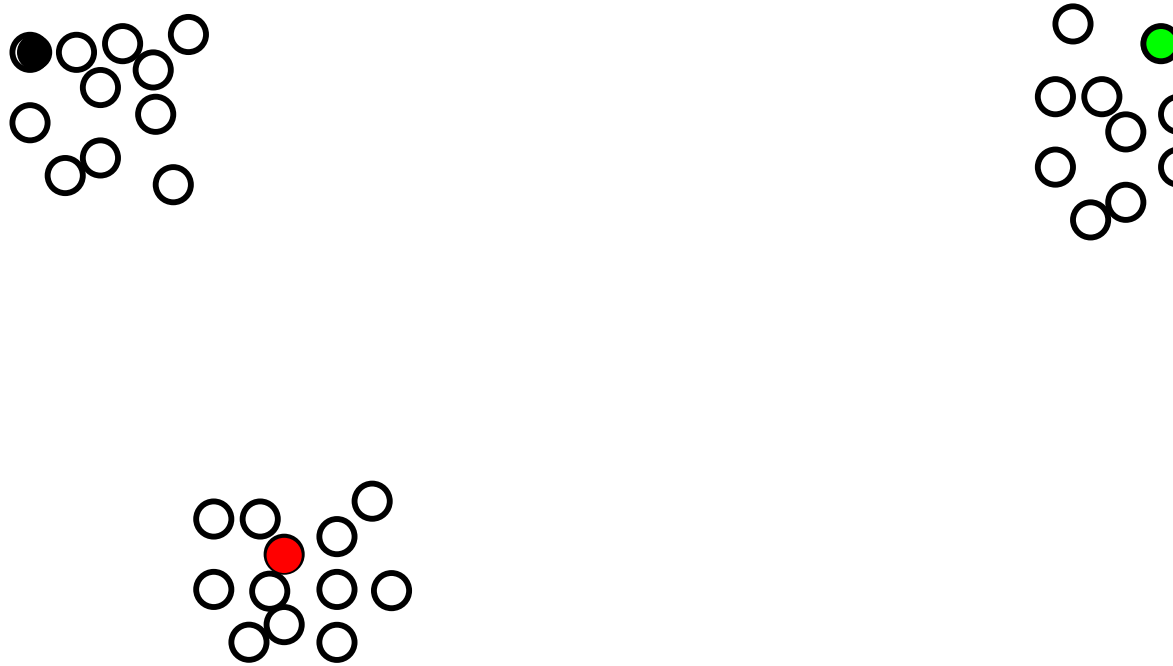
- For $j = 2, \ldots, k$
  - Pick $\mathbf{c_j}$ among datapoints $\mathbf{x^1}, \mathbf{x^2}, \ldots, \mathbf{x^d}$ that is farthest from previously chosen $\mathbf{c_1}, \mathbf{c_2}, \ldots, \mathbf{c_{j-1}}$

缺点：有异常值受影响大

优：理想情况下优秀.

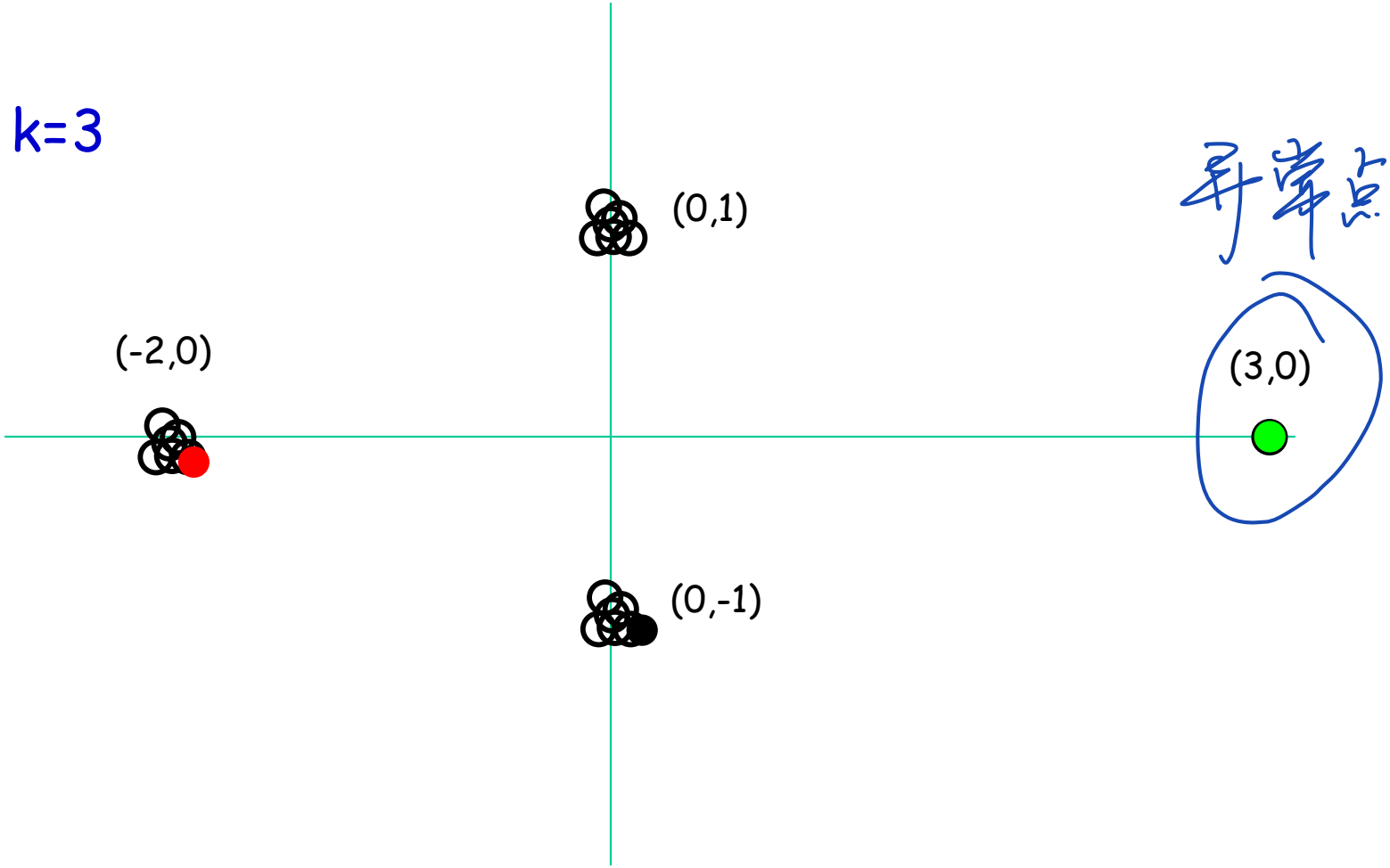
Fixes the Gaussian problem. But it can be thrown off by outliers….

Furthest point heuristic does well on previous example



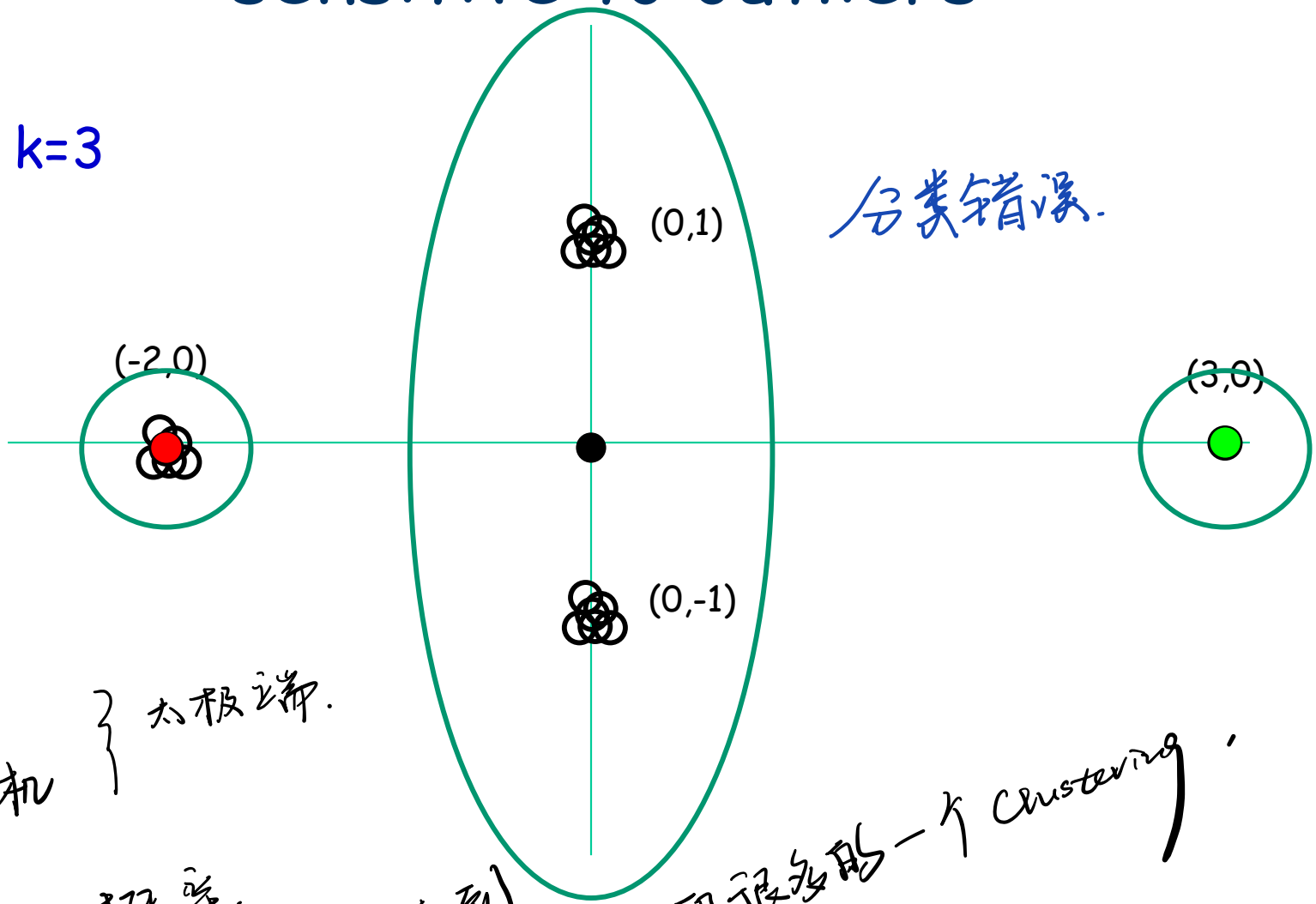
Furthest point initialization heuristic sensitive to outliers

Assume k=3



Furthest point initialization heuristic sensitive to outliers

Assume k=3



分类错误.

最远点, 和 随机 } 太极端.

优化: 按距离给概率.

⇒ 最远点依然很可能选到, 但是也可能选中其它次远且很多的一个 Clustering.

K-means++ Initialization: D^2 sampling [AV07]

- Interpolate between random and furthest point initialization
- Let D(x) be the distance between a point x and its nearest center. Chose the next center proportional to $D^2(x)$.

- Choose c_1 at random.
- For $j = 2, \ldots, k$
  - Pick c_j among $x^1, x^2, \ldots, x^n$ according to the distribution

$$\Pr(c_j = x^i) \propto \min_{j'<j} \left\|x^i - c_{j'}\right\|^2 \quad D^2(x^i)$$

概率 ∝ 距离

在该分布下抽取 c_j

Theorem: K-means++ always attains an O(log k) approximation to optimal k-means solution in expectation.

以 O(log k) 的速率逼近

Running Lloyd's can only further improve the cost.

K-means++ Idea: D^2 sampling

- Interpolate between random and furthest point initialization
- Let D(x) be the distance between a point x and its nearest center. Chose the next center proportional to $D^{\alpha}(\mathbf{x})$.

$\|x^i - c_j\|_\alpha^\alpha = (\sum |x^i - c_j|^\alpha)^{\frac{1}{\alpha}}$

  - $\alpha = 0$, random sampling
  - $\alpha = \infty$, furthest point 无穷范数：选择最大值.
    (Side note: it actually works well for k-center)
  - $\alpha = 2$, k-means++ 欧氏距离. → 是随机与最远点的权衡.

Side note: $\alpha = 1$, works well for k-median

K-means ++ Fix

K-means++/ Lloyd's Running Time

- K-means ++ initialization: O(nd) and one pass over data to select next center. So O(nkd) time in total.
- Lloyd's method 迭代T次：$O(nkd + Tnkd)$

Repeat until there is no change in the cost.

- For each j: $C_j \leftarrow \{x \in S$ whose closest center is $\mathbf{c_j}\}$
- For each j: $\mathbf{c_j} \leftarrow$ mean of C_j

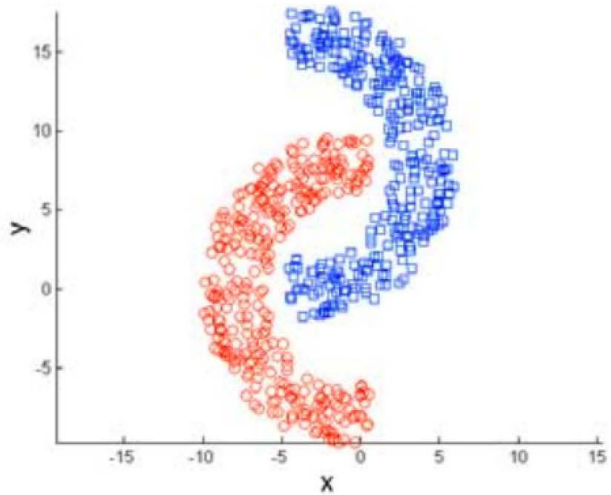
Each round takes time O(nkd).

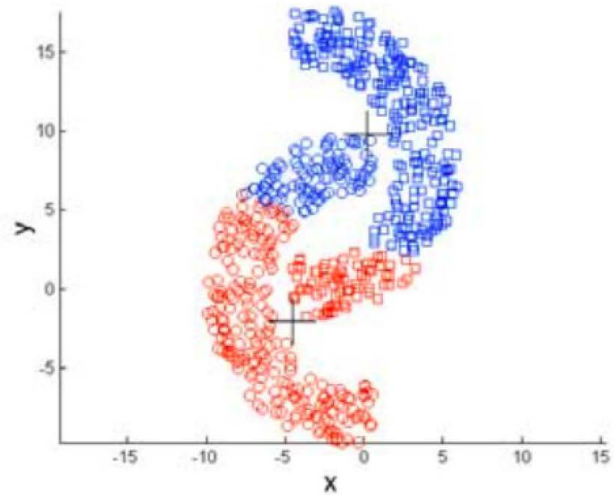
最差时T为指数次数（迭代次数）

- Exponential # of rounds in the worst case [AV07].
- Expected polynomial time in the smoothed analysis model!

K-means++/ Lloyd's Summary

- K-means++ always attains an $O(\log k)$ approximation to optimal k-means solution in expectation.
- Running Lloyd's can only further improve the cost.
- Exponential # of rounds in the worst case [AV07].
- Expected polynomial time in the smoothed analysis model!
- Does well in practice.

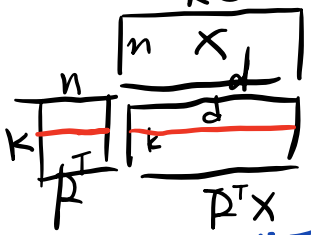




Kernel K-means

矩阵化目标函数：$\min_{R,C} \|X - RC\|_F^2$ s.t. $R \in \{0,1\}^{n\times k}$, $R \times \mathbb{1}_k = \mathbb{1}_n$

$R^T X$：



假设 R_i^T 中有 m 个 1 => 从 X 中找到 m 个样本点.

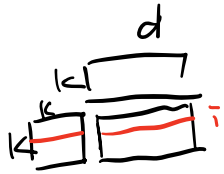
$(R^TR)^{-1}R^TX$

$k\times n$ $n\times k$ $k\times n$ $n\times d$

$k\times k$ $k\times d$

$k\times d$

=>



对角矩阵，描述样本个数.

就是C矩阵.

$$Q(R) = \|X - R(R^TR)^{-1}R^TX\|_F^2$$

$$= \underbrace{\langle X, X\rangle}_{\text{常数}} - 2\langle X, R(R^TR)^{-1}R^TX\rangle + \langle R(R^TR)^{-1}R^TX, R(R^TR)^{-1}R^TX\rangle$$

$$= -\mathrm{Tr}(X^TR(R^TR)^{-1}R^TX)$$

$$\min_R Q(R) = \max_R \mathrm{Tr}(X^TR(R^TR)^{-1}R^TX) = \max_R \mathrm{Tr}(R(R^TR)^{-1}R^T\underbrace{XX^T}_{\text{内积}})$$

可在这套 kernel

$$R(R^TR)^{-1}R^T = \underbrace{R(R^TR)^{-\frac{1}{2}}}_{\text{定义为 F}}\underbrace{(R^TR)^{-\frac{1}{2}}R^T}_{F^T} = FF^T$$

$$\Rightarrow \max_F \mathrm{Tr}(FXX^TF^T) \quad F^TF = I$$

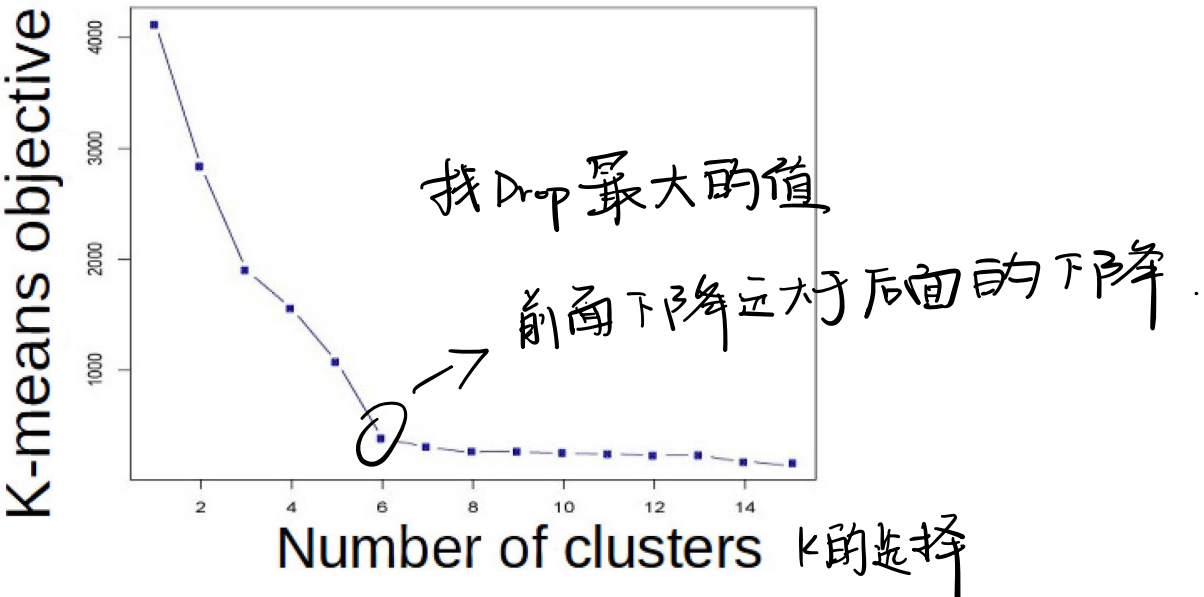
(特征值分解，可给出闭式解)

扔掉了 $\{0,1\}$ 的约束

$$F^TF = (R^TR)^{-\frac{1}{2}}R^TR(R^TR)^{-\frac{1}{2}} = (R^TR)^{\frac{1}{2}}(R^TR)^{-\frac{1}{2}} = I.$$

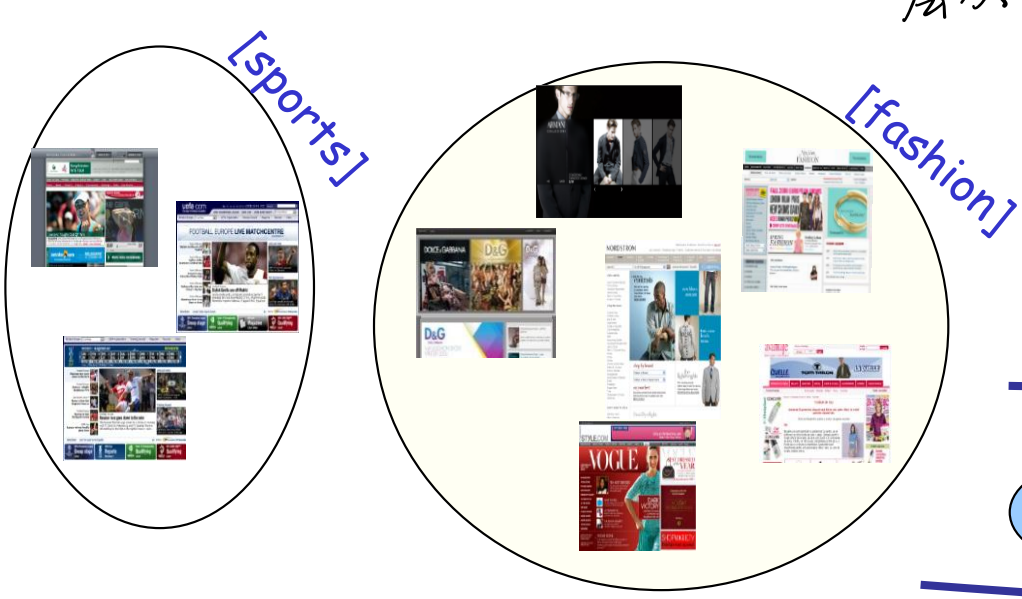
What value of k???

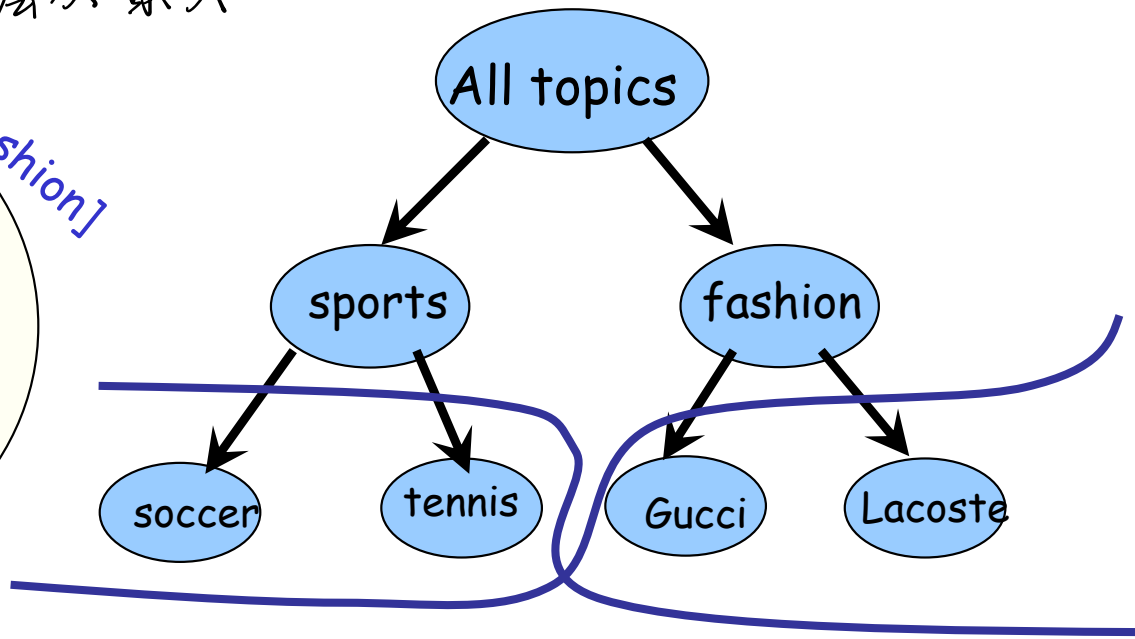
- Heuristic: Find large gap between k -1-means cost and k-means cost.
- According to information criteria (AIC, BIC, etc.)
- Try hierarchical clustering.



Hierarchical Clustering

层次聚类



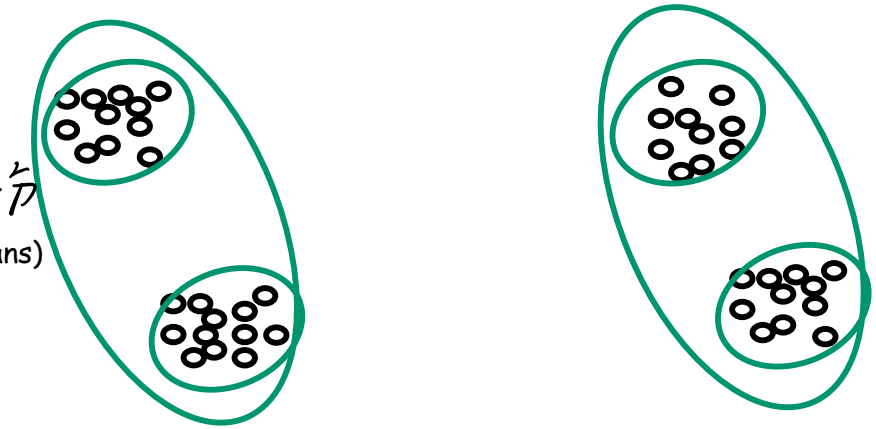


- A hierarchy might be more natural.
- Different users might care about different levels of granularity or even prunings.

Hierarchical Clustering

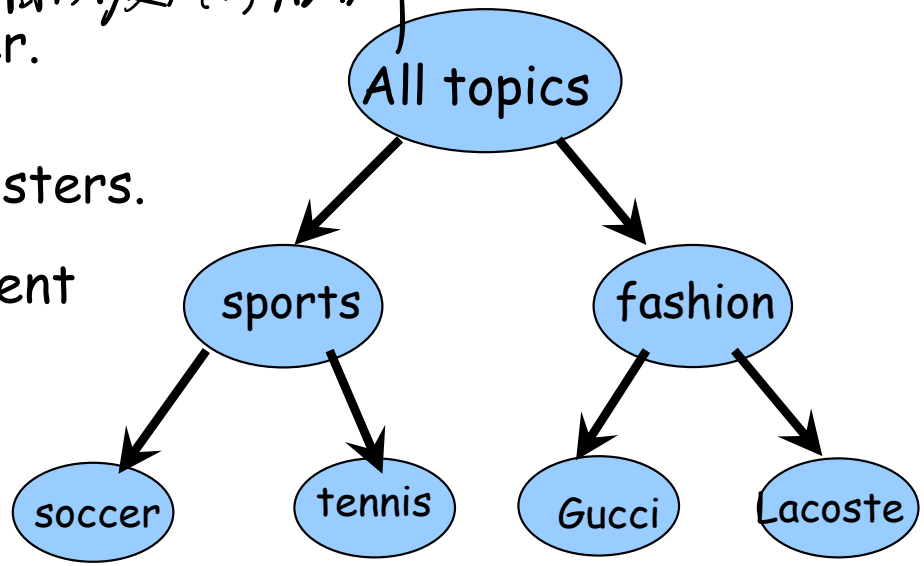
Top-down (divisive)

- Partition data into 2-groups (e.g., 2-means)
- Recursively cluster each group.



Bottom-Up (agglomerative)

- Start with every point in its own cluster.
- Repeatedly merge the "closest" two clusters.
- Different defs of "closest" give different algorithms.

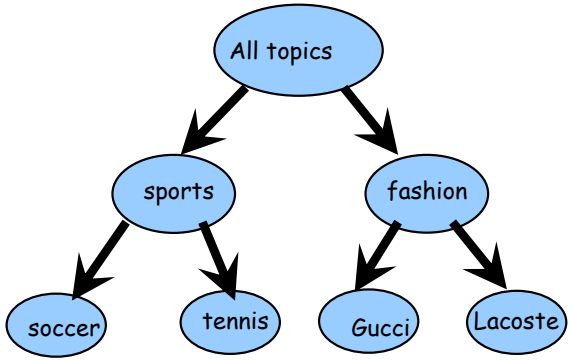


Bottom-Up (agglomerative)

Have a distance measure on pairs of objects.

d(x,y) – distance between x and y

E.g., # keywords in common, edit distance, etc



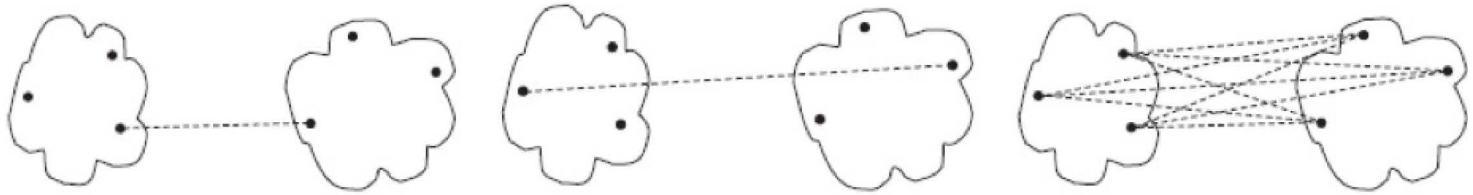
作为两集合距离.

- Single linkage: $\text{dist}(A, B) = \min_{x \in A, x' \in B'} \text{dist}(x, x')$ 两集合间最小值

- Complete linkage: $\text{dist}(A, B) = \max_{x \in A, x' \in B'} \text{dist}(x, x')$ 最大值

- Average linkage: $\text{dist}(A, B) = \text{avg}_{x \in A, x' \in B'} \text{dist}(x, x')$ 均值

- Ward's method



(a) MIN (single link.) (b) MAX (complete link.) (c) Group average.

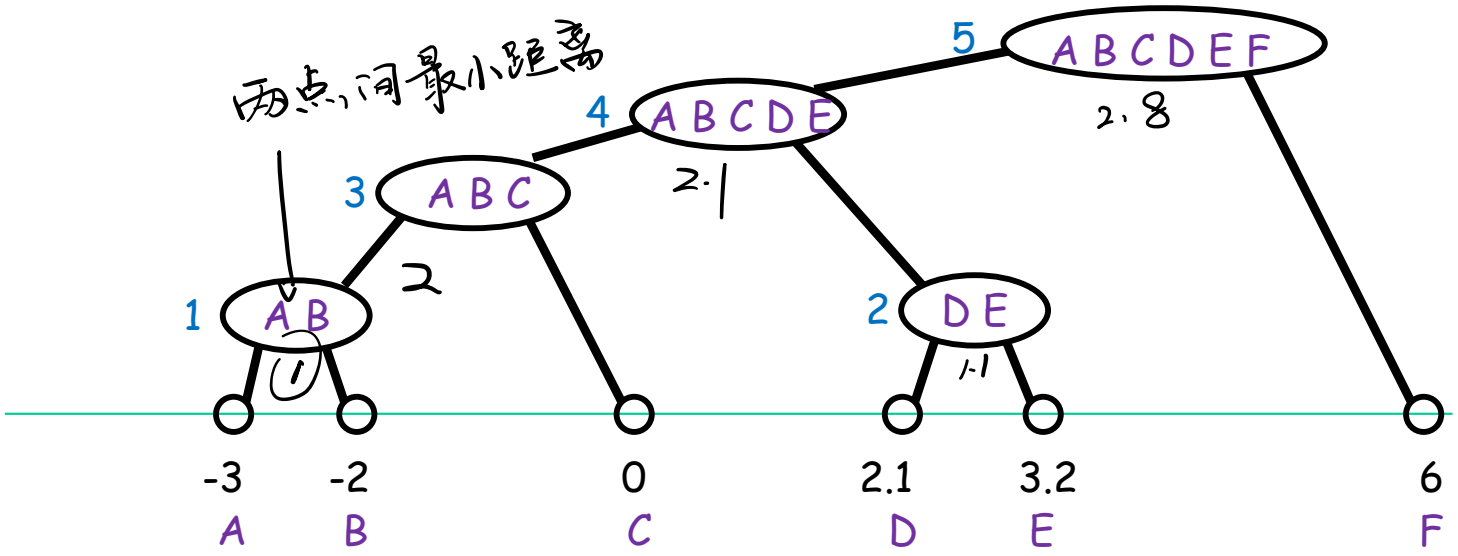
Single Linkage

Bottom-up (agglomerative)

- Start with every point in its own cluster.
- Repeatedly merge the "closest" two clusters.

Single linkage: $\text{dist}(A, B) = \min_{x \in A, x' \in B} \text{dist}(x, x')$

Dendogram



Single Linkage

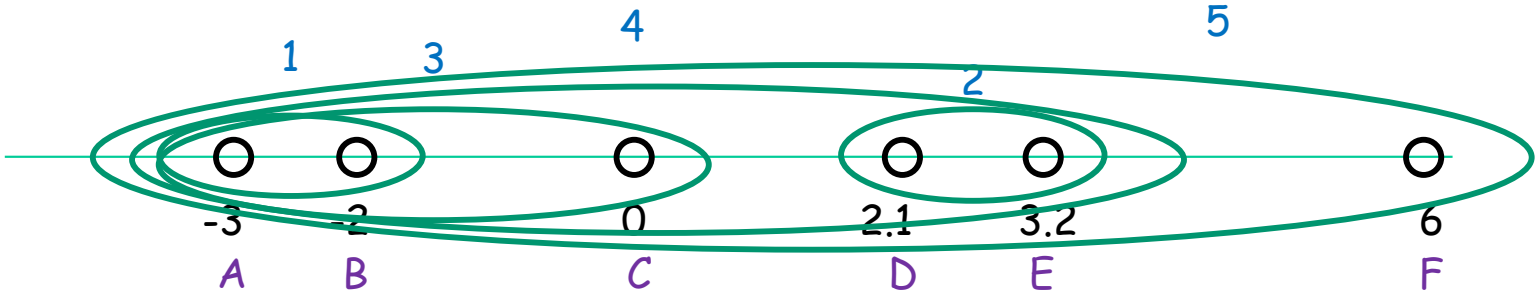
Bottom-up (agglomerative)

- Start with every point in its own cluster.
- Repeatedly merge the "closest" two clusters.

Single linkage: $\text{dist}(A, B) = \min_{x \in A, x' \in B} \text{dist}(x, x')$

One way to think of it: at any moment, we see connected components of the graph where connect any two pts of distance < r.

Watch as r grows (only n-1 relevant values because we only we merge at value of r corresponding to values of r in different clusters).



Complete Linkage

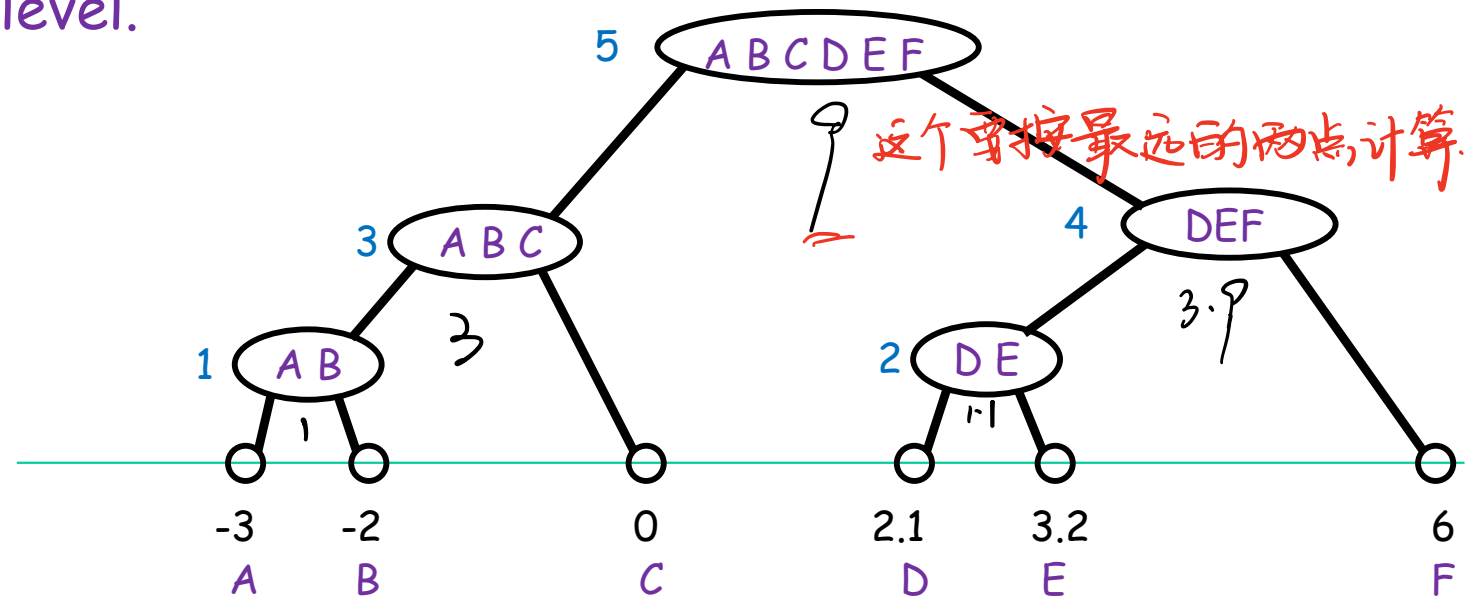
计算最大距离.

Bottom-up (agglomerative)

- Start with every point in its own cluster.
- Repeatedly merge the "closest" two clusters.

Complete linkage: $\text{dist}(A, B) = \max_{x \in A, x' \in B} \text{dist}(x, x')$

One way to think of it: keep max diameter as small as possible at any level.



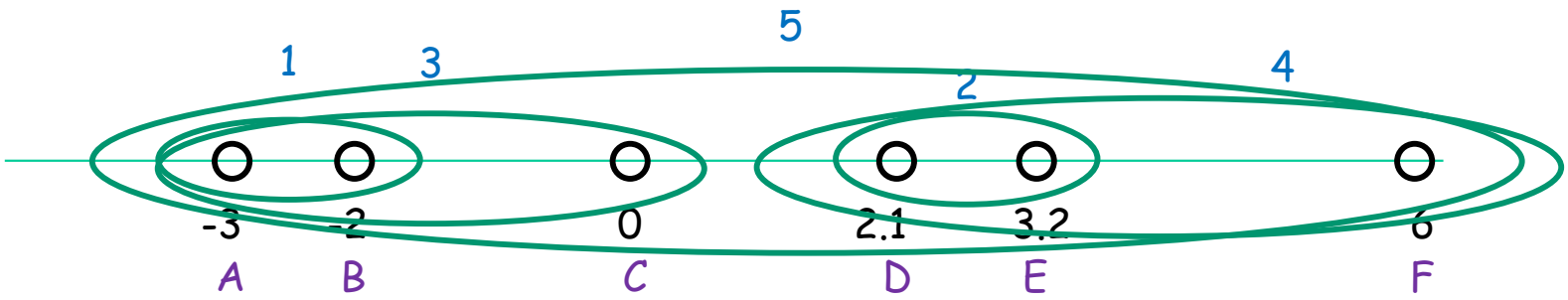
Complete Linkage

Bottom-up (agglomerative)

- Start with every point in its own cluster.
- Repeatedly merge the "closest" two clusters.

Complete linkage: $\text{dist}(A, B) = \max_{x \in A, x' \in B} \text{dist}(x, x')$

One way to think of it: keep max diameter as small as possible.



Ward's Method

Bottom-up (agglomerative)

- Start with every point in its own cluster.
- Repeatedly merge the "closest" two clusters.

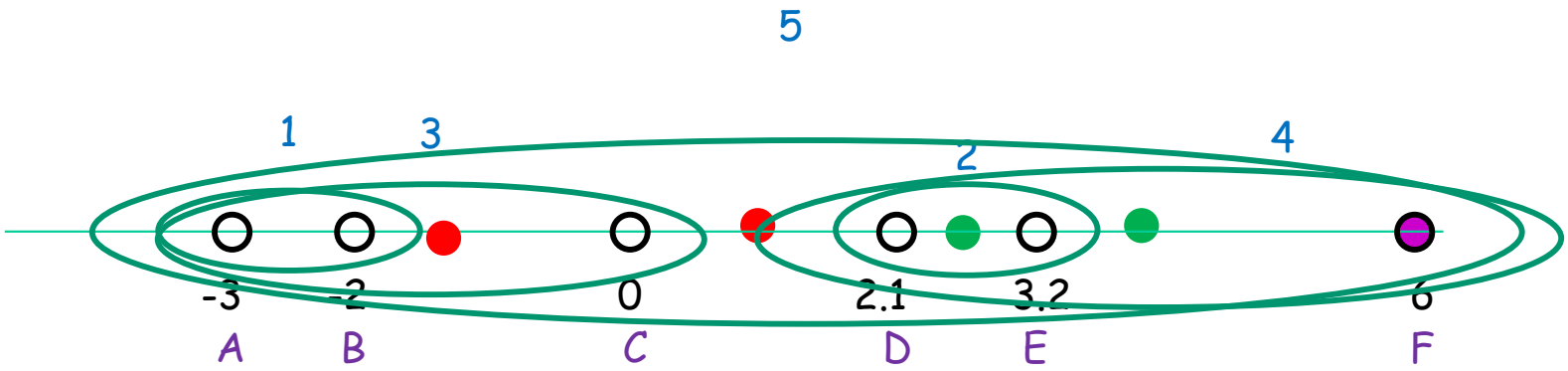
Ward's method: $\mathrm{dist}(C, C') = \frac{|C| \cdot |C'|}{|C| + |C'|} \|\mathrm{mean}(C) - \mathrm{mean}(C')\|^2$

样本个数作为权重

均值距离.

Merge the two clusters such that the increase in k-means cost is as small as possible.

Works well in practice.



Running time

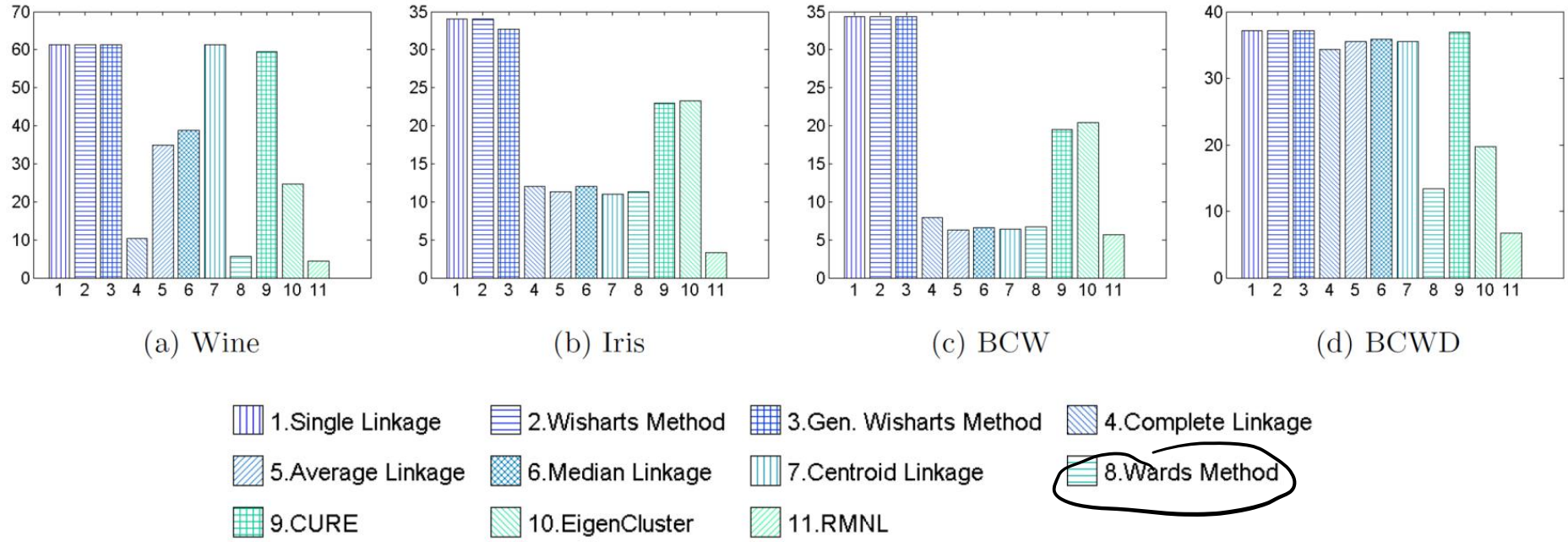
- Each algorithm starts with N clusters, and performs N-1 merges.
- For each algorithm, computing $dist(C, C')$ can be done in time $O(|C| \cdot |C'|)$. (e.g., examining $dist(x, x')$ for all $x \in C, x' \in C'$)
- Time to compute all pairwise distances and take smallest is $O(N^2)$.
- Overall time is $O(N^3)$.

In fact, can run all these algorithms in time $O(N^2 \log N)$.

See: Christopher D. Manning, Prabhakar Raghavan and Hinrich Schütze, Introduction to Information Retrieval, Cambridge University Press. 2008. http://www-nlp.stanford.edu/IR-book/

Hierarchical Clustering Experiments

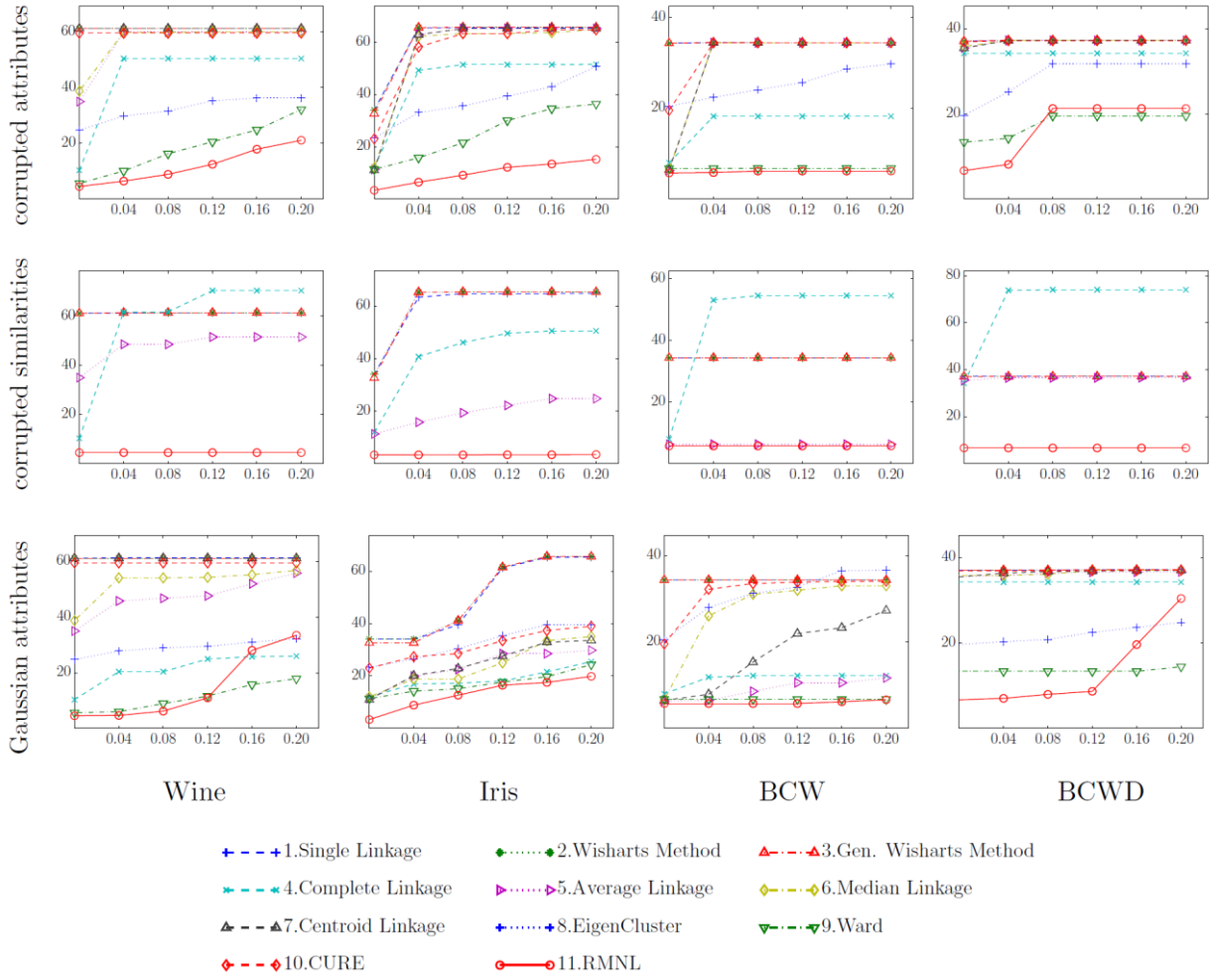
[BLG, JMLR'15]



Ward's method does the best among classic techniques.

Hierarchical Clustering Experiments

[BLG, JMLR'15]



Ward's method does the best among classic techniques.

What You Should Know

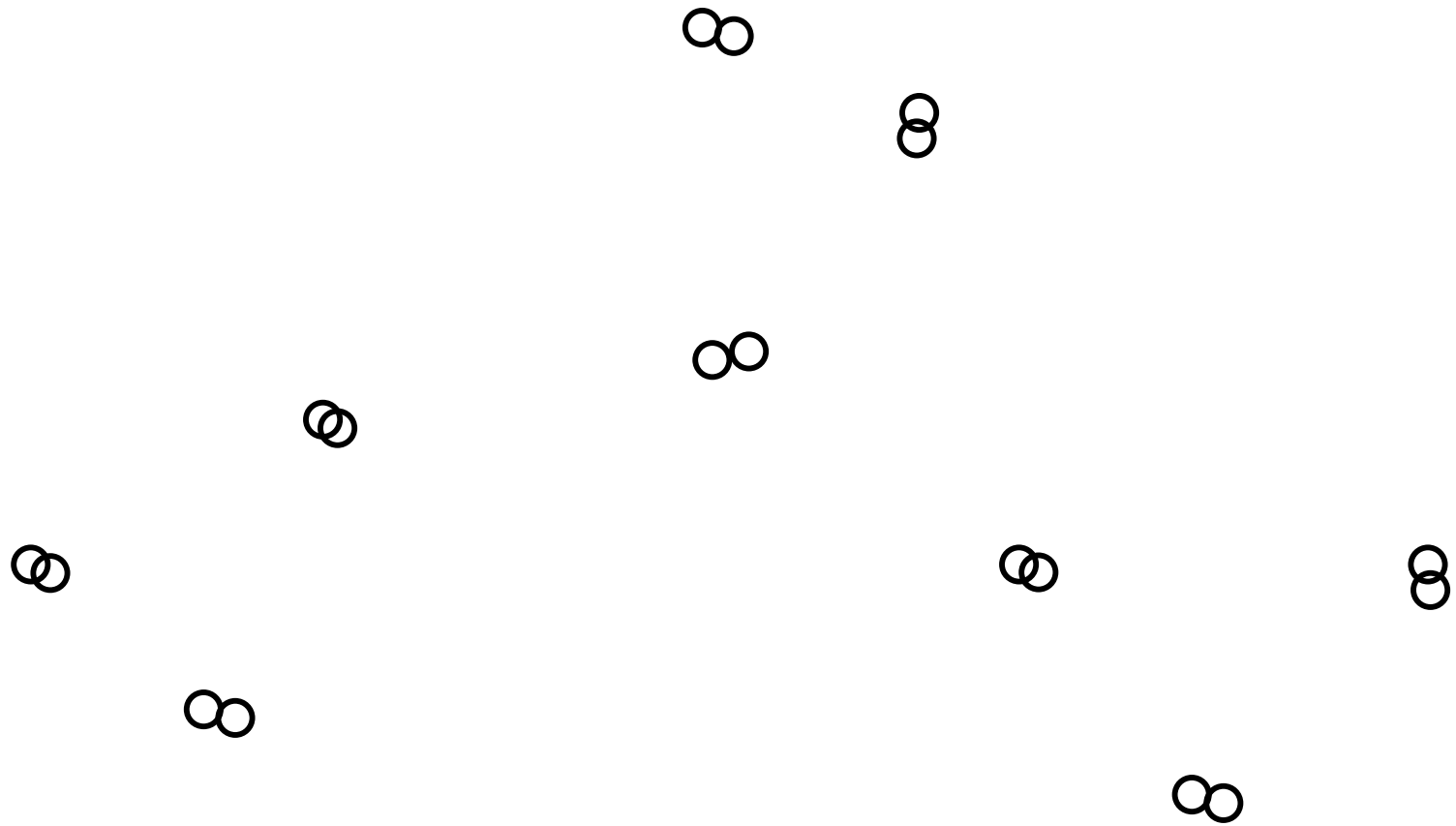
- Partitional Clustering. k-means and k-means ++
  - Lloyd's method
  - Initialization techniques (random, furthest traversal, k-means++)

- Hierarchical Clustering.
  - Single linkage, Complete Linkge, Ward's method

Additional Slides

Smoothed analysis model

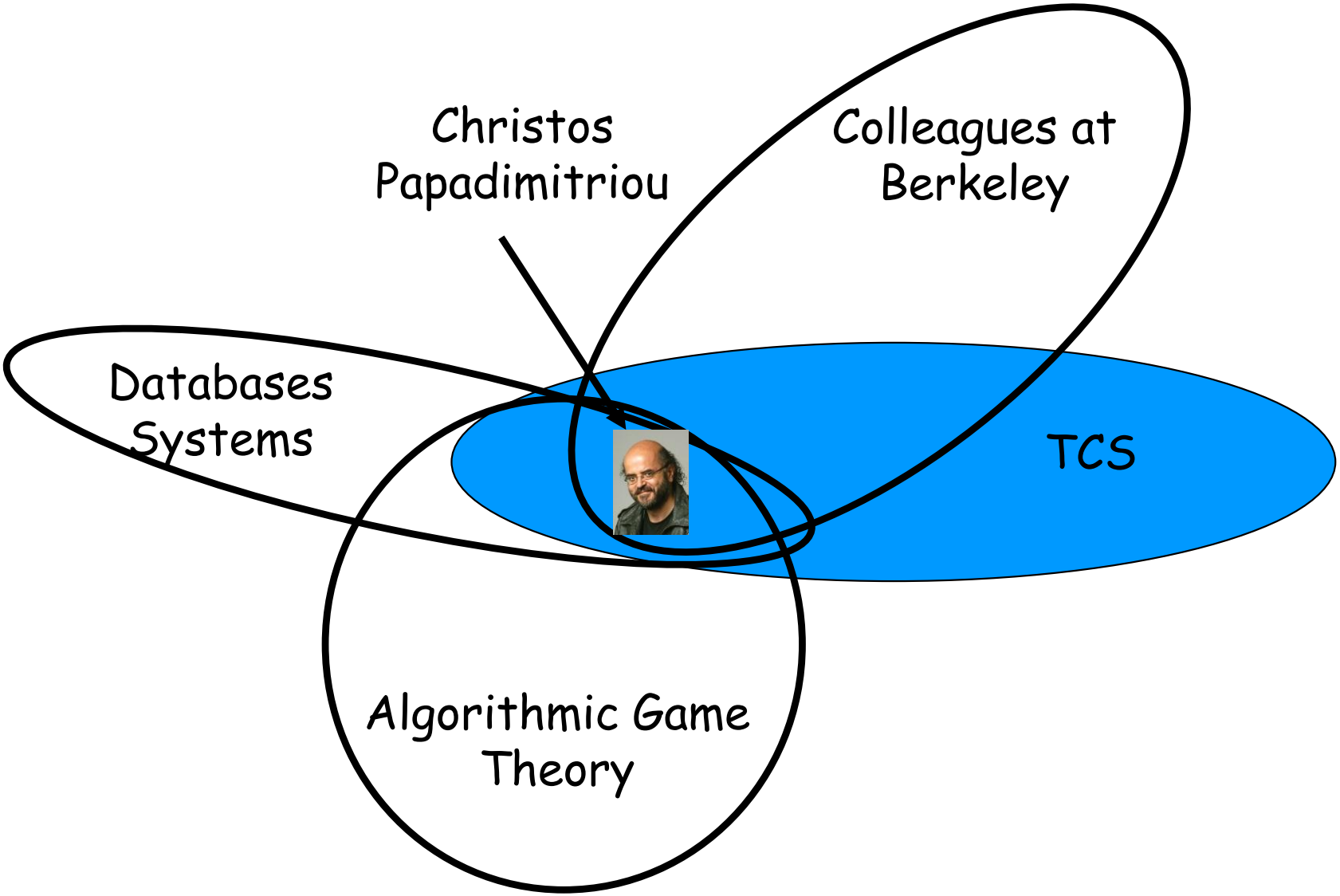
- Imagine a worst-case input.
- But then add small Gaussian perturbation to each data point.



Smoothed analysis model

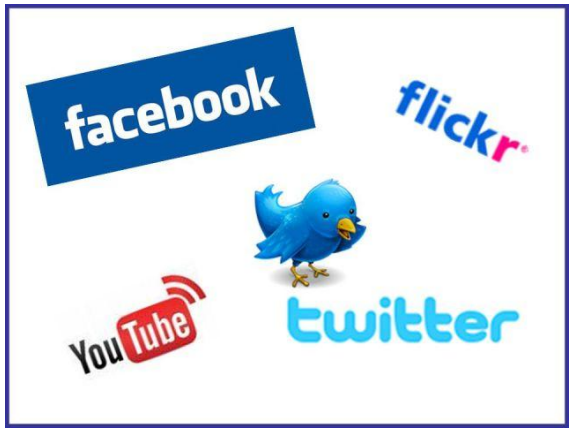
- Imagine a worst-case input.
- But then add small Gaussian perturbation to each data point.
- Theorem [Arthur-Manthey-Roglin 2009]:
  - E[number of rounds until Lloyd's converges] if add Gaussian perturbation with variance σ^2 is polynomial in n, $1/\sigma$.
  - The actual bound is : $O\left(\frac{n^{34}k^{34}d^8}{\sigma^6}\right)$
- Might still find local opt that is far from global opt.

Overlapping Clusters: Communities



Overlapping Clusters: Communities

- Social networks



- Professional networks

- Product Purchasing Networks, Citation Networks, Biological Networks, etc

Overlapping Clusters: Communities