# Clustering.
# Unsupervised Learning

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?

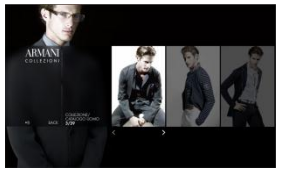
**Useful for:**

- Automatically organizing data.
- 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…)

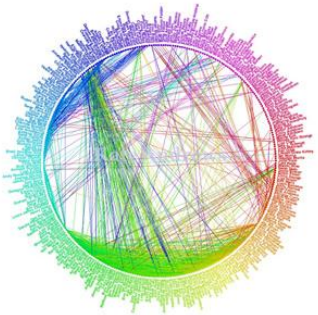
- Cluster news articles or web pages or search results by topic.



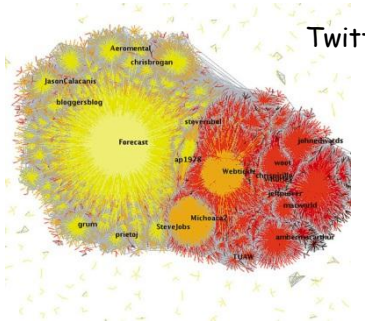



- Cluster protein sequences by function or genes according to expression profile.


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



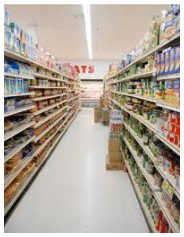
Facebook network




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**:

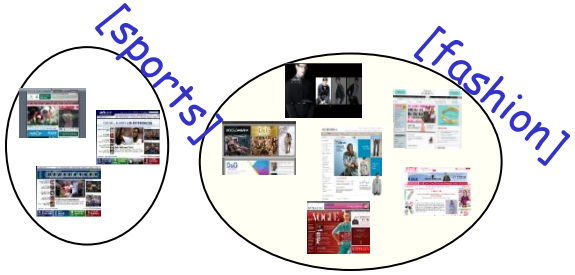
- Objective based clustering
- Hierarchical clustering
- Mention overlapping clusters

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

# Objective Based Clustering

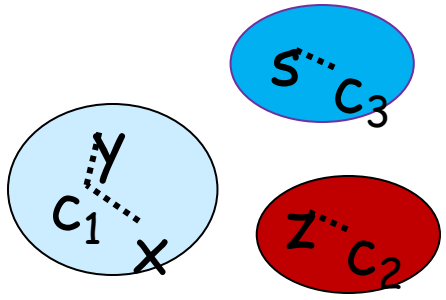
**Input**: A set S of n points, also a distance/dissimilarity measure specifying the distance d(x,y) between pairs (x,y).

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

K-means dricks.

# Euclidean k-means Clustering

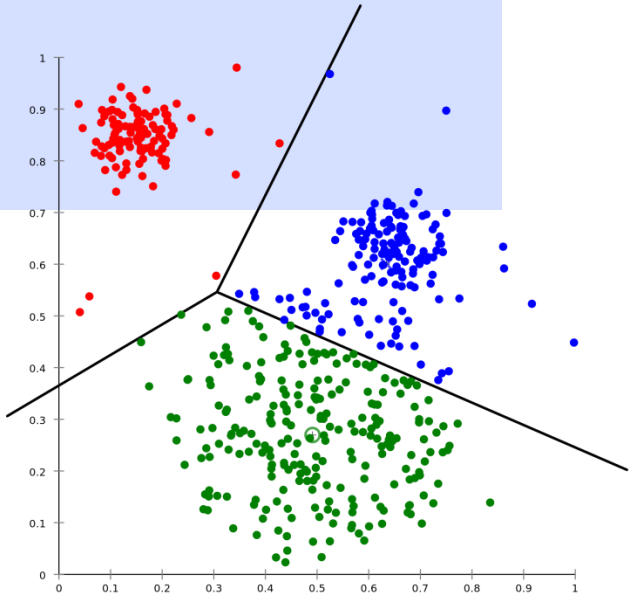
**Input**: A set of $n$ datapoints $\mathbf{x^1}, \mathbf{x^2}, \ldots, \mathbf{x^n}$ in $\mathrm{R^d}$

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

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



# Euclidean k-means Clustering

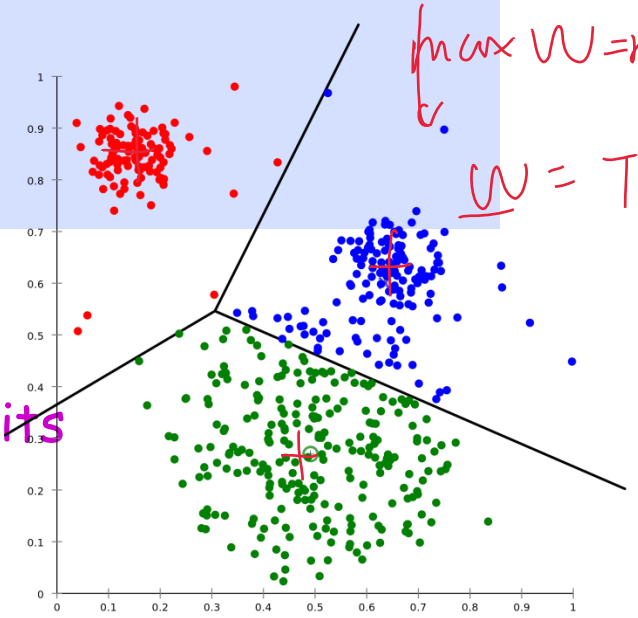
**Input**: A set of $n$ datapoints $\mathbf{x^1}, \mathbf{x^2}, \ldots, \mathbf{x^n}$ in $\mathrm{R^d}$
target #clusters $k$

**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.




With-cluster sim. W ↑ (av.)

Between-cluster sim. B ↓

T = W + B

max W = min B

W = T - B

# Euclidean k-means Clustering

$$\min_{r,C} \sum_{i=1}^{n}\sum_{j=1}^{k} r_{ij}\,\|x_i - c_j\|_2^2 = Q(r,C), \qquad x_i \in \mathbb{R}^d$$

s.t. $r_{ij} \in \{0,1\}, \ \forall i,j$

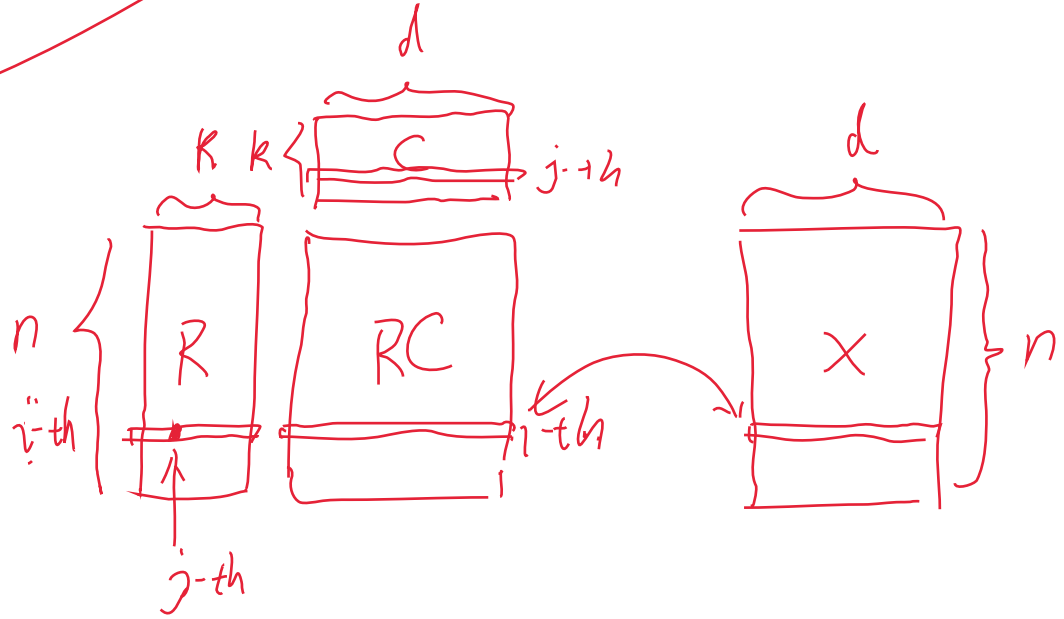
$$Q(R,C) = \|X - RC\|_F^2, \qquad R \in \{0,1\}^{n\times k}, \qquad C \in \mathbb{R}^{k\times d}$$

$$\min_{R,C} \|X - RC\|_F^2$$

s.t.

$$C = DR^TX \in \mathbb{R}^{k\times d}$$

$$D \in \mathbb{R}^{k\times k} \qquad (D)_{kk} = \frac{1}{\sum_{i=1}^{n} r_{ik}}$$



# 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 $\mu$

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 $\boldsymbol{c_1}, \mathbf{c_2}, \ldots, \boldsymbol{c_k} \in R^d$ 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 $C_j$

# 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 $\boldsymbol{c_1}, \mathbf{c_2}, \ldots, \boldsymbol{c_k}$

# 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)

# Common Heuristic: The Lloyd's method

$Q(R,C) = \|X - RC\|_F^2$

$t$-th round: $R^{(t)}$, $C^{(t)}$

Step 1: $\min_R Q(R, C^{(t)}) = \|X - RC^{(t)}\|_F^2$

$R^{(t+1)} = \arg\min_R Q(R, C^{(t)})$

$Q(R^{(t+1)}, C^{(t)}) \le Q(R^{(t)}, C^{(t)})$

Step 2: Fix $R^{(t+1)}$, solve $C$

$C^{(t+1)} = \arg\min_C Q(R^{(t+1)}, C)$

$Q(R^{(t+1)}, C^{(t+1)}) \le Q(R^{(t+1)}, C^{(t)})$

$\Rightarrow Q(R^{(t+1)}, C^{(t+1)}) \le Q(R^{(t)}, C^{(t)})$

# 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.

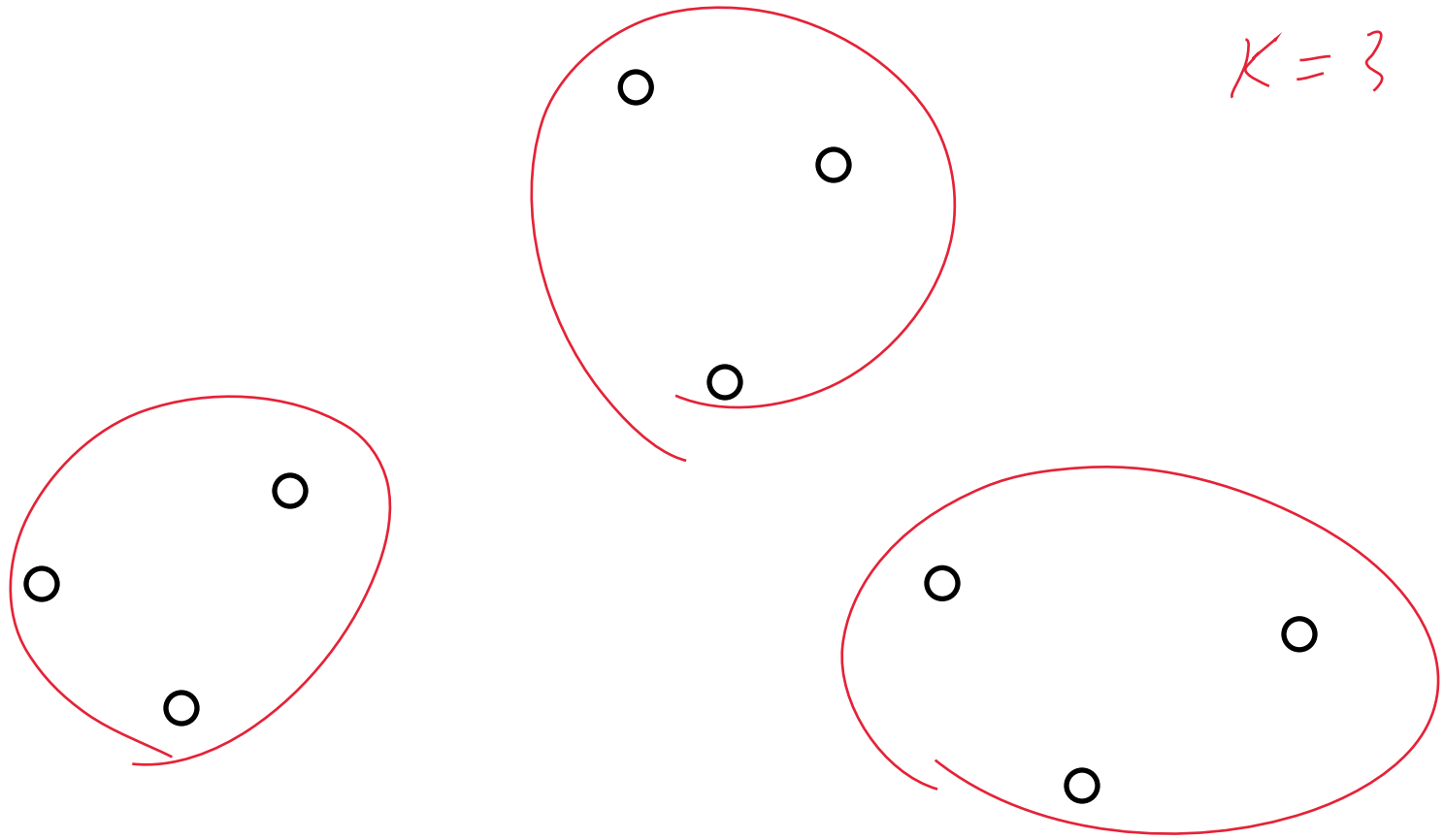
- For each $j$: $C_j \leftarrow \{x \in S$ whose closest center is $\mathbf{c_j}\}$
- For each $j$: $\mathbf{c_j} \leftarrow$ mean of $C_j$

- 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)
  - Furthest traversal
  - K-means ++ (works well and has provable guarantees)

# Lloyd's method: Random Initialization

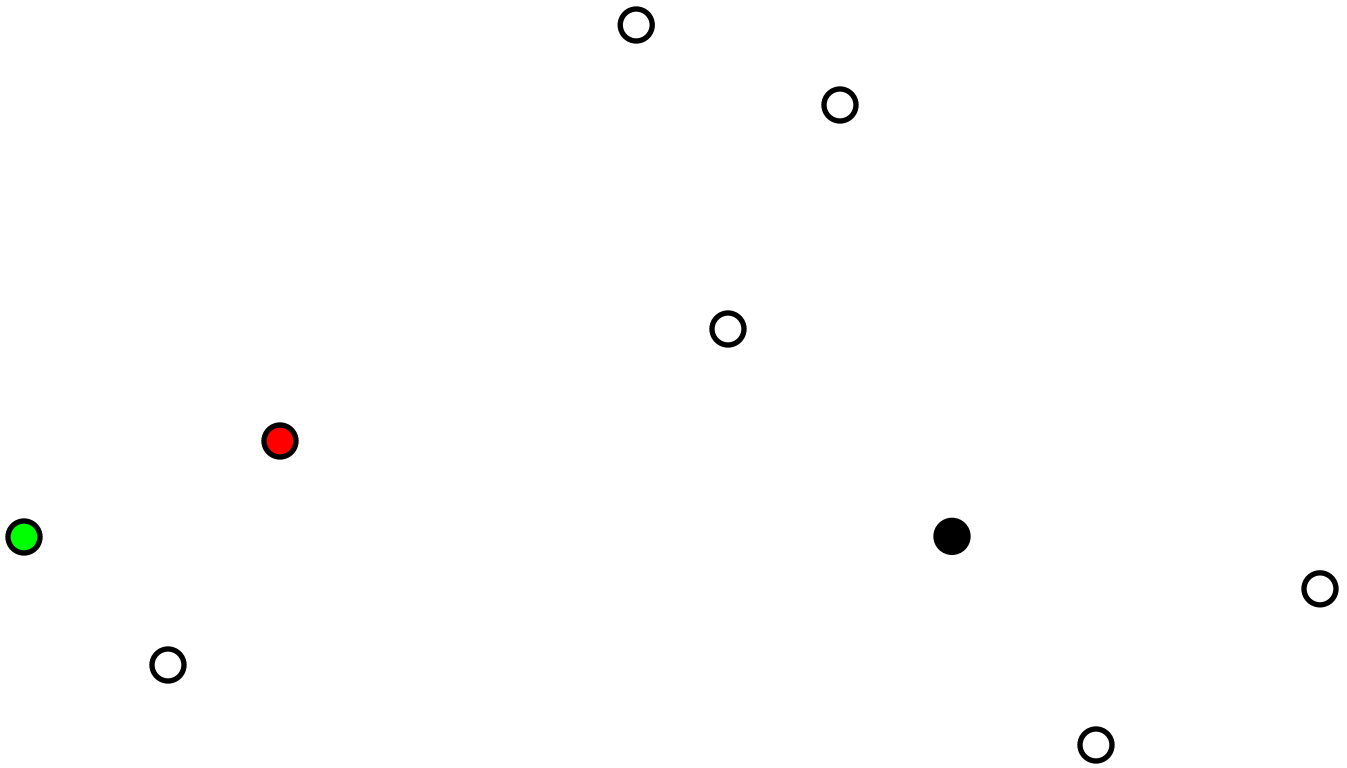
# Lloyd's method: Random Initialization

Example: Given a set of datapoints



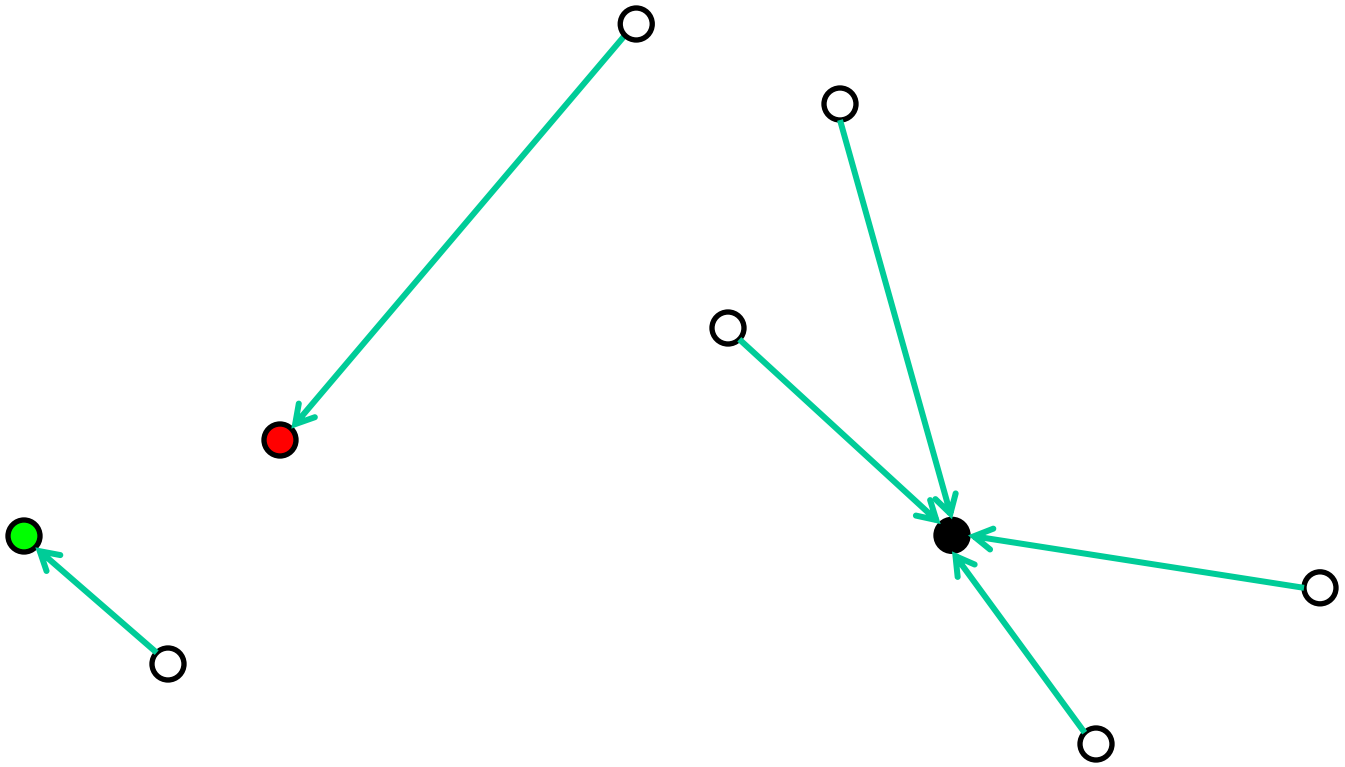
# Lloyd's method: Random Initialization

Select initial centers at random



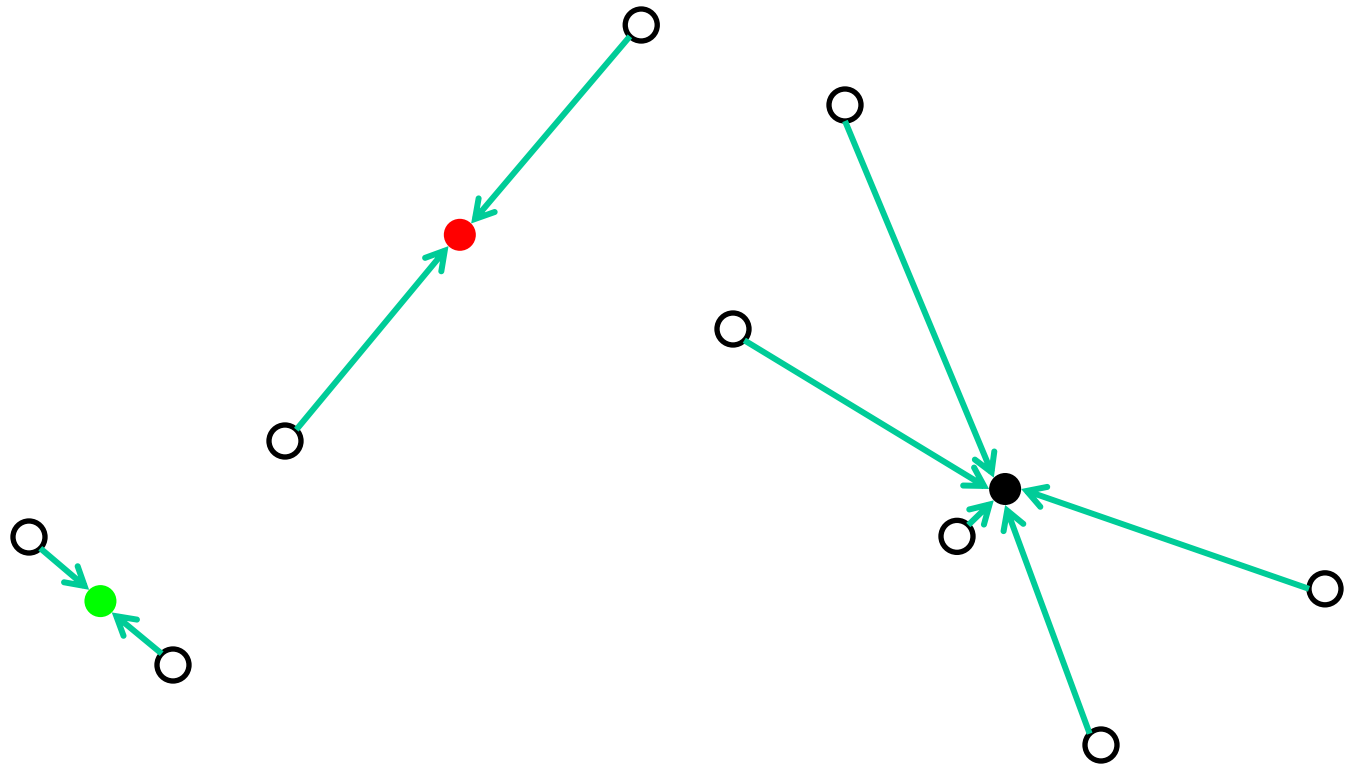
# Lloyd's method: Random Initialization

Assign each point to its nearest center



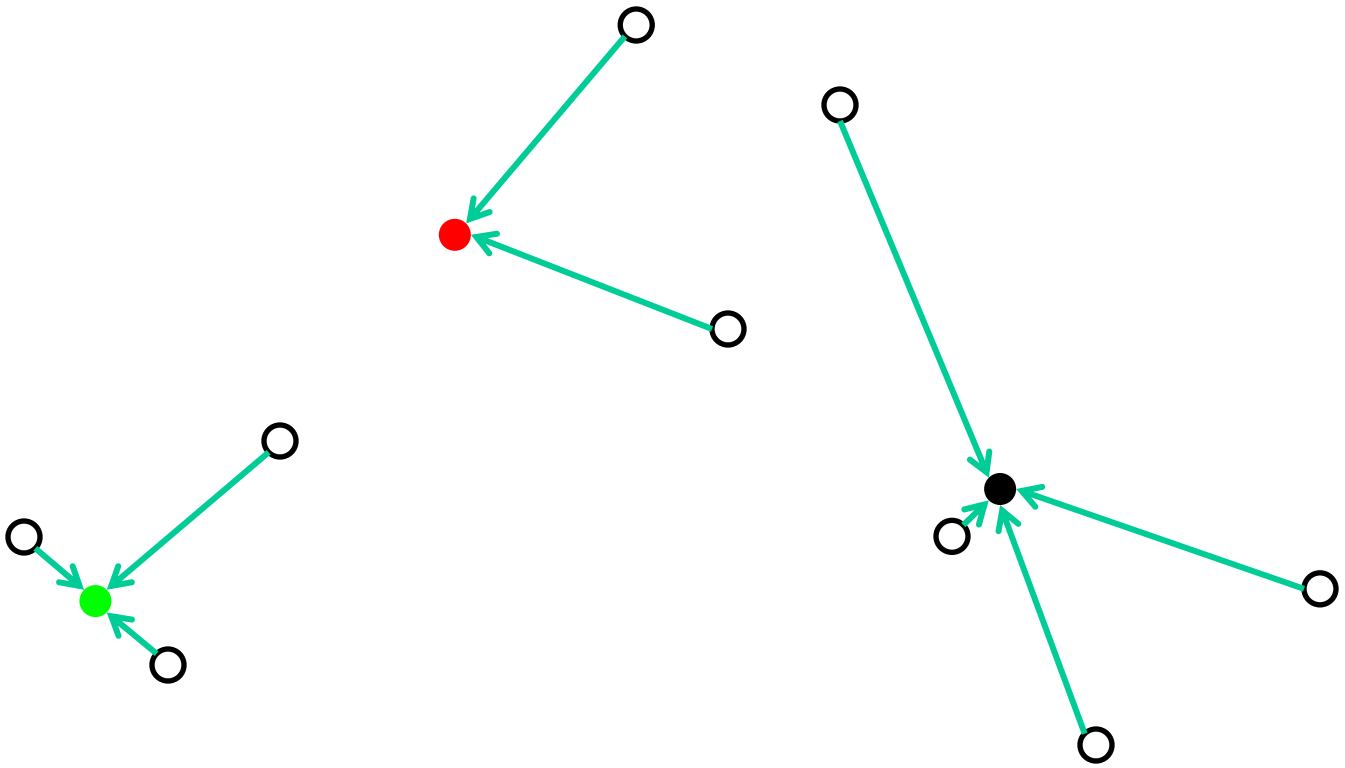
# Lloyd's method: Random Initialization

Recompute optimal centers given a fixed clustering



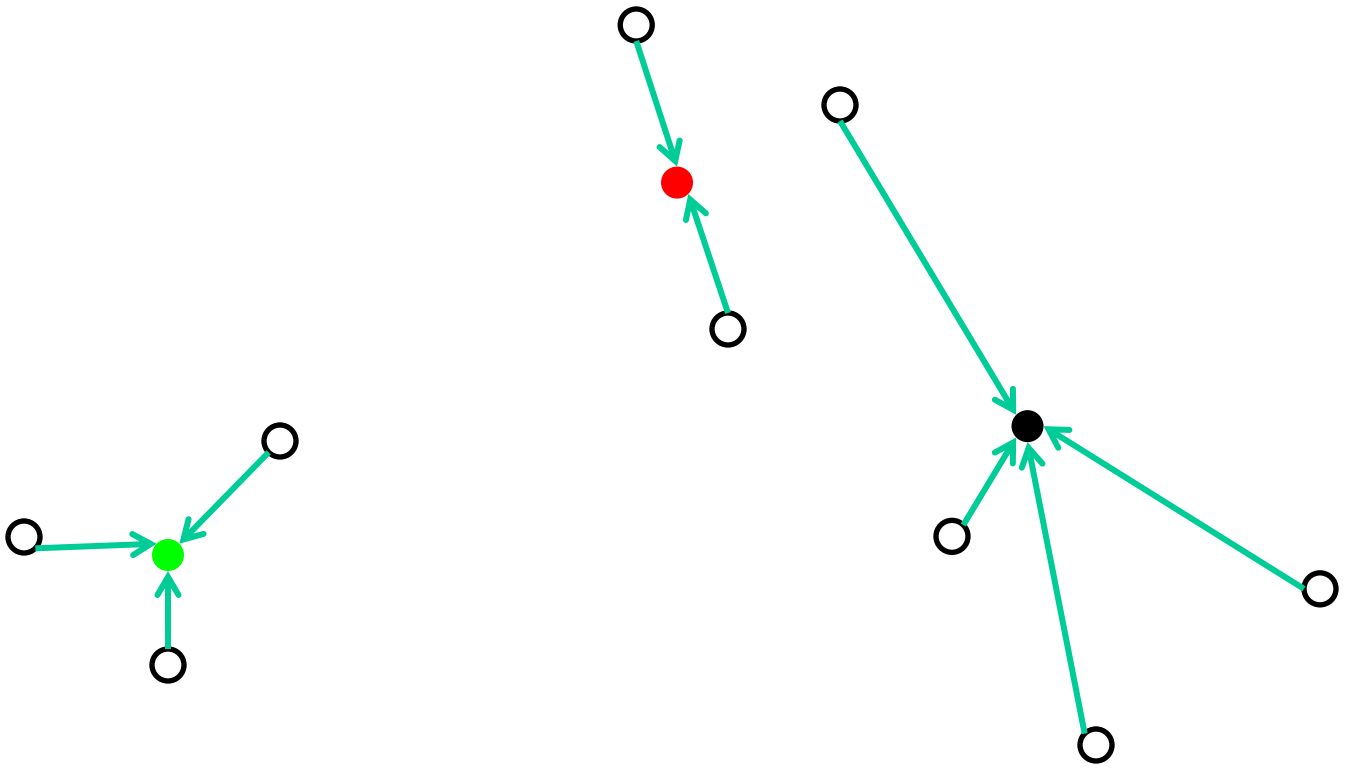
# Lloyd's method: Random Initialization

Assign each point to its nearest center



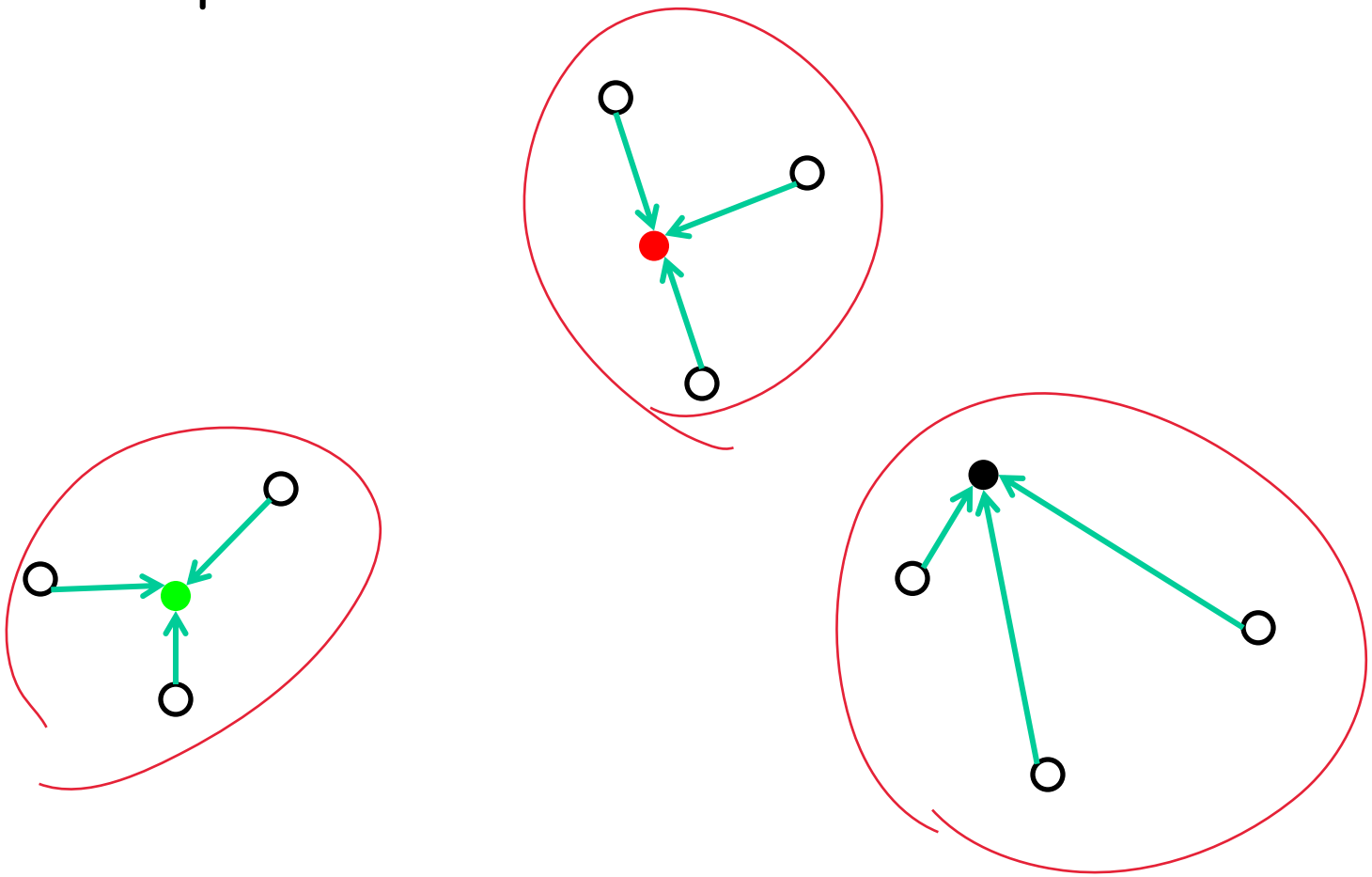
# Lloyd's method: Random Initialization

Recompute optimal centers given a fixed clustering



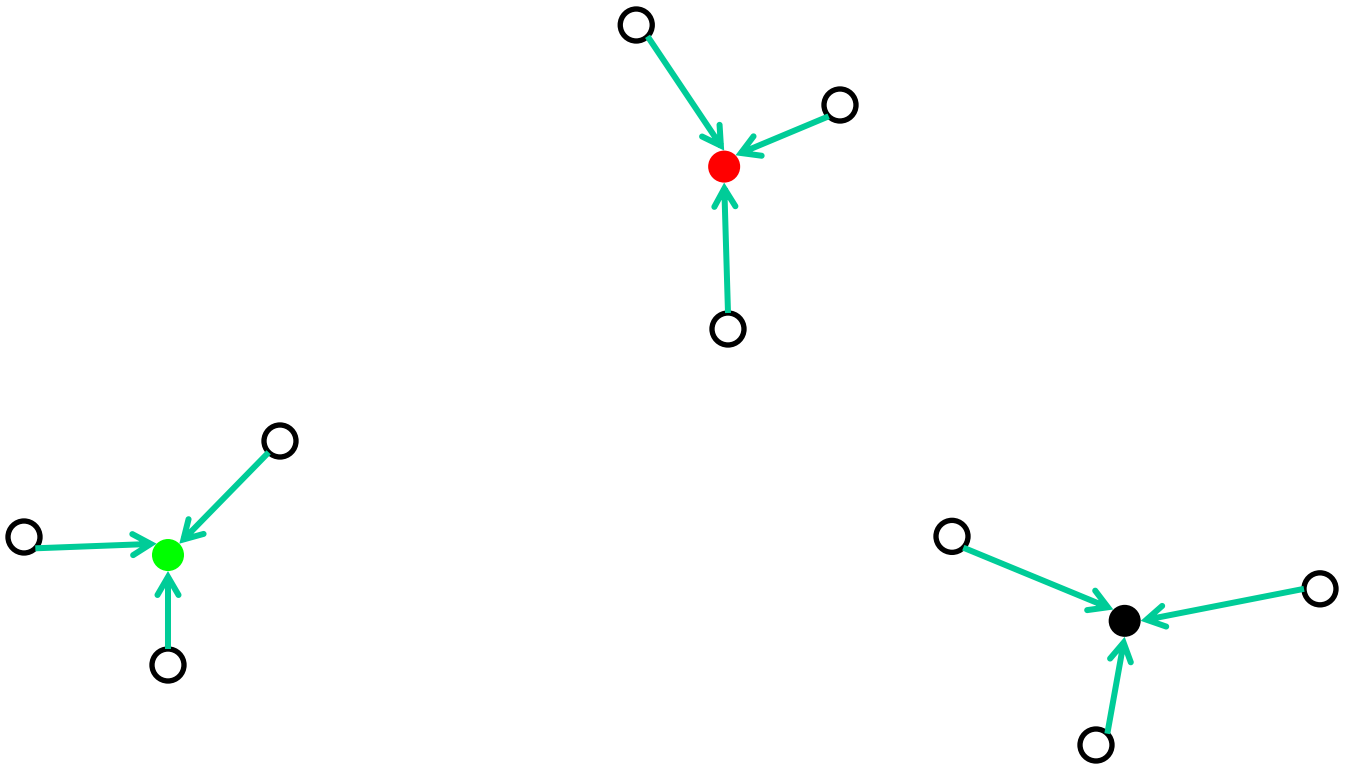
# Lloyd's method: Random Initialization

Assign each point to its nearest center



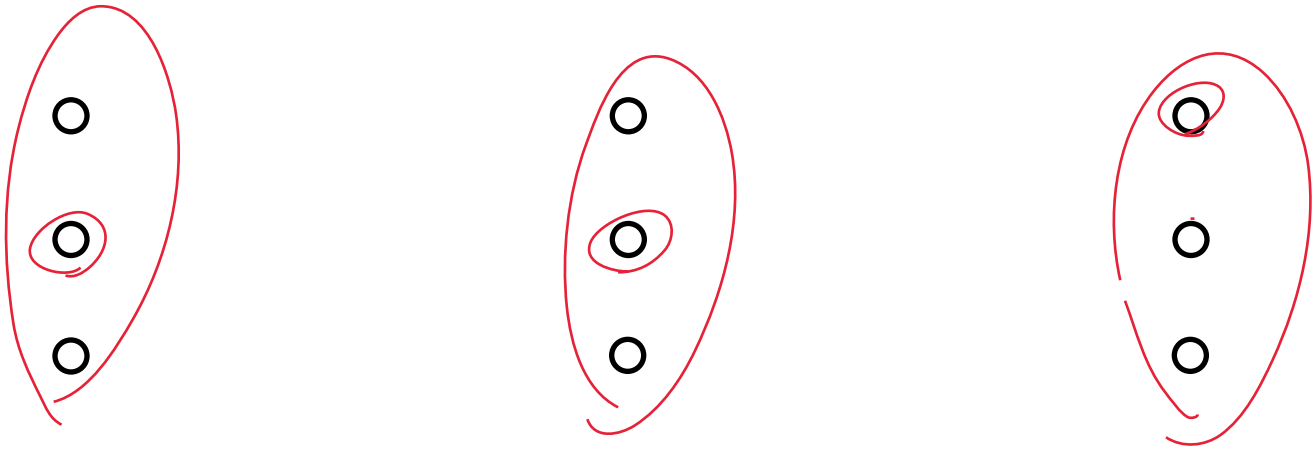
# Lloyd's method: Random Initialization

Recompute optimal centers given a fixed clustering



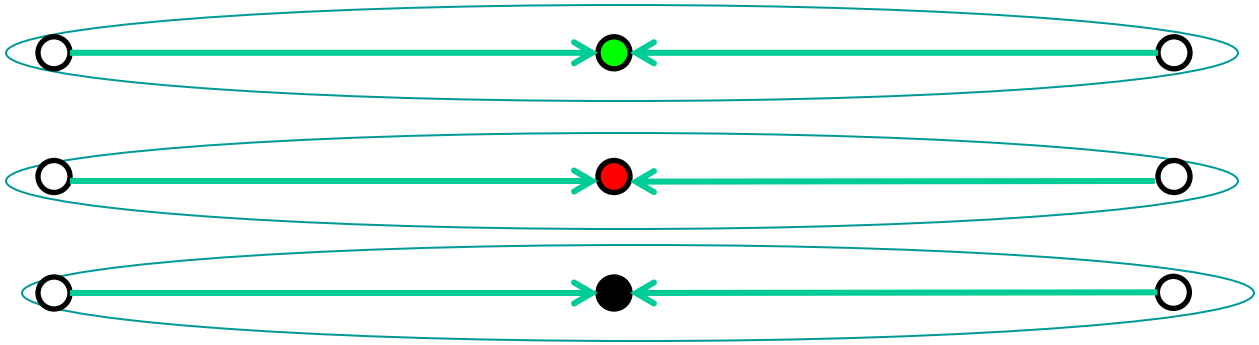
Get a good quality solution in this example.

# Lloyd's method: Performance



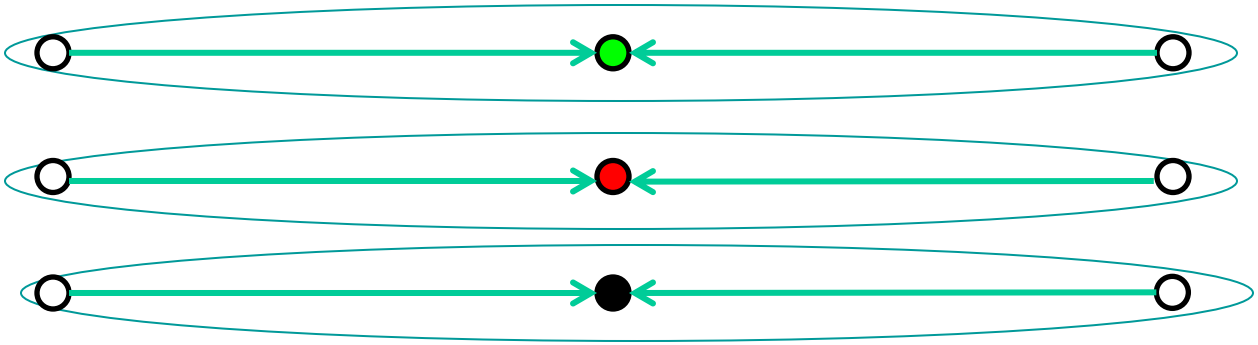
It always converges, but it may converge at a local optimum that is different from the global optimum, and in fact could be arbitrarily worse in terms of its score.

# Lloyd's method: Performance

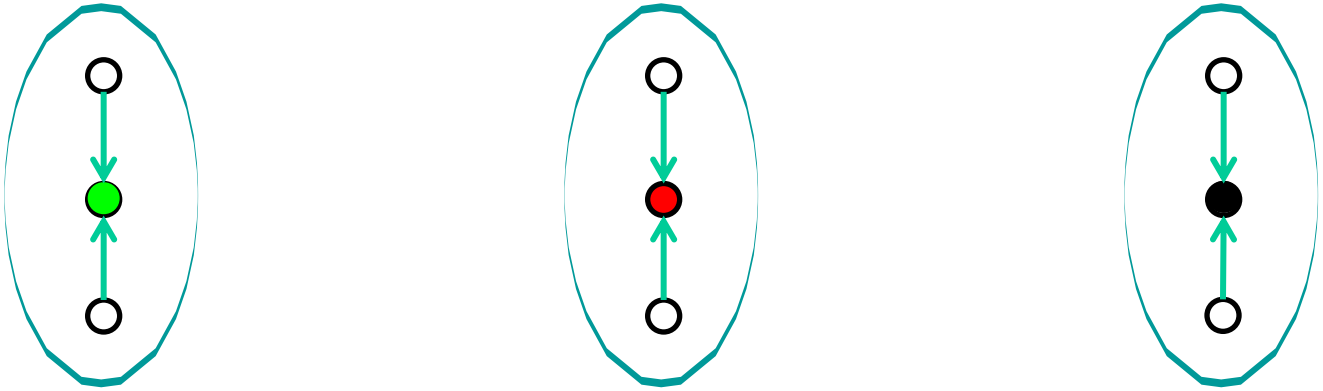


Local optimum: every point is assigned to its nearest center and every center is the mean value of its points.

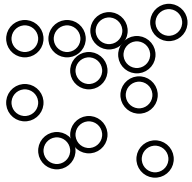
# Lloyd's method: Performance

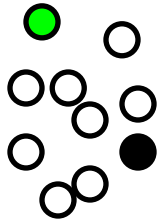


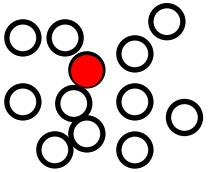
.It is arbitrarily worse than optimum solution….



# Lloyd's method: Performance

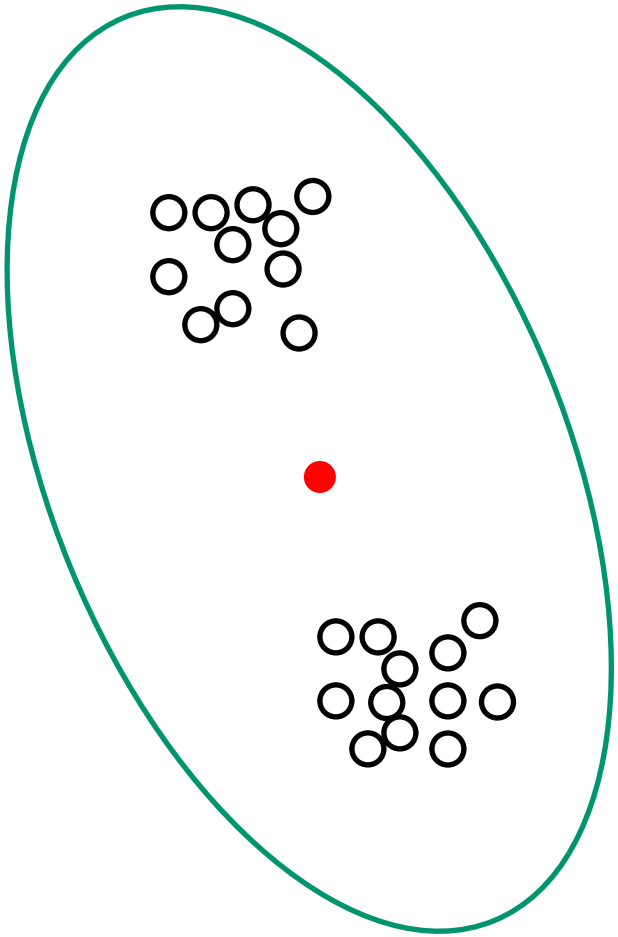


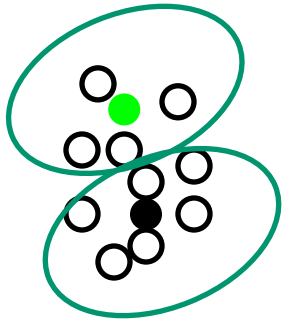




This bad performance, can happen even with well separated Gaussian clusters.

# Lloyd's method: Performance





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).

$$\binom{k+k-1}{k-1} = \Omega(k^k)$$

  - For k equal-sized Gaussians, Pr[each initial center is in a different Gaussian] $\approx \frac{k!}{k^k} \approx \frac{1}{e^k}$
  - Becomes unlikely as k gets large.

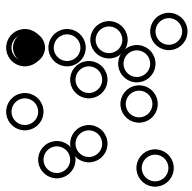
# Another Initialization Idea: Furthest Point Heuristic

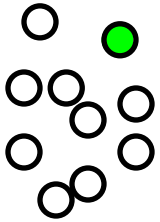
Choose $\mathbf{c_1}$ arbitrarily (or at random).

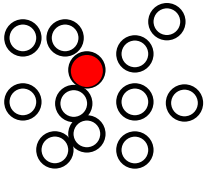
- For $\mathrm{j} = 2, \ldots, \mathrm{k}$
  - Pick $\mathbf{c_j}$ among datapoints $\mathbf{x^1}, \mathbf{x^2}, \ldots, \mathbf{x^d}$ that is farthest from previously chosen $\mathbf{c_1}, \mathbf{c_2}, \ldots, \mathbf{c_{j-1}}$

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

# Furthest point heuristic does well on previous example





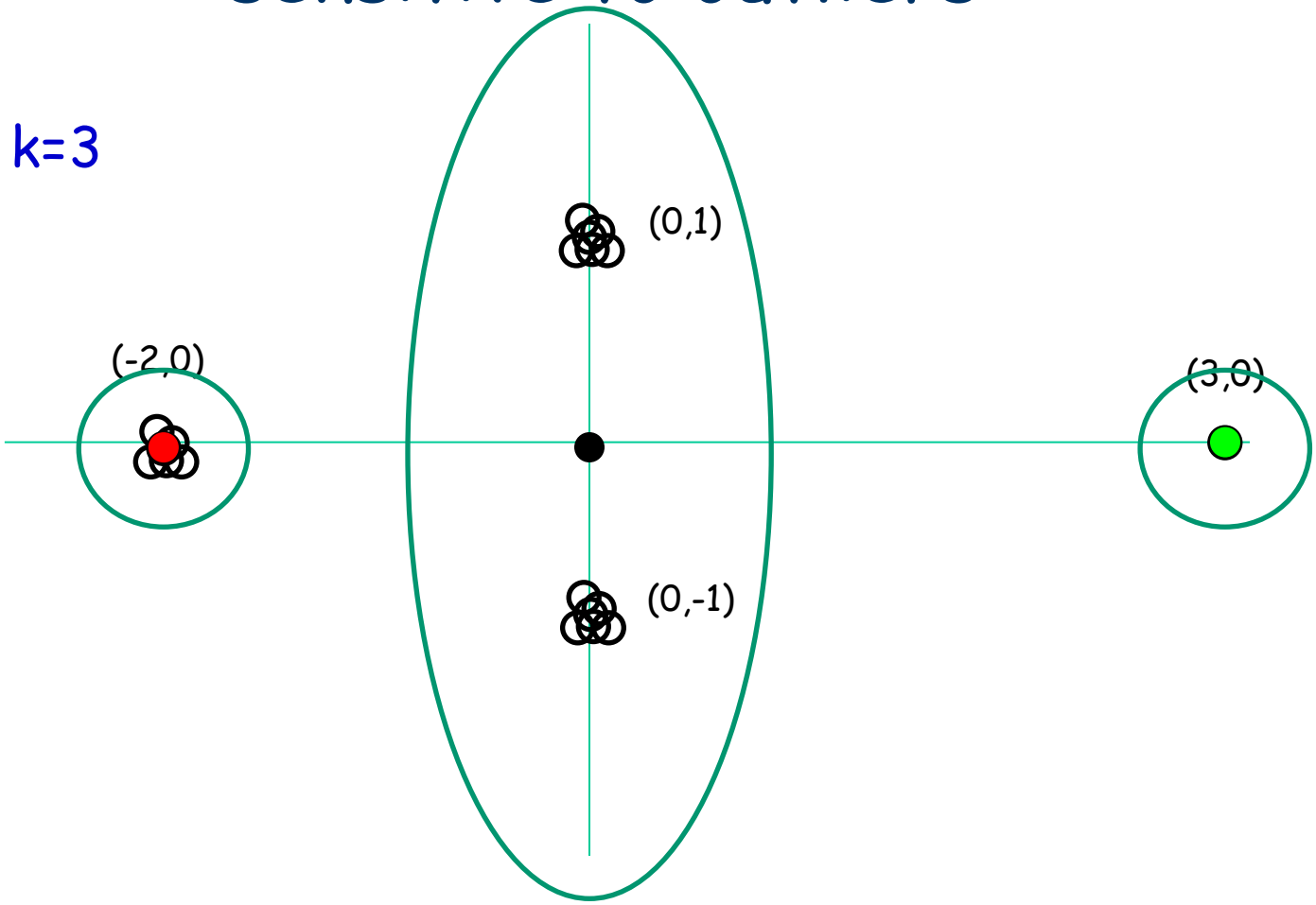


# Furthest point initialization heuristic sensitive to outliers

Assume k=3

# Furthest point initialization heuristic sensitive to outliers

Assume k=3



# 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

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

$C_1, C_2, \ldots C_{j-1}$

**Theorem:** 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.

# 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 $D^{\alpha}(x)$.

  - $\alpha = 0$, random sampling
  - $\alpha = \infty$, furthest point (Side note: it actually works well for k-center)
  - $\alpha = 2$, k-means++

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

$\alpha \in R^d$

$\|a\|_p = \left(\sum_{j=1}^{d} |a_j|^p\right)^{\frac{1}{p}}$

$\|a\|_1 = \sum_{j=1}^{d} |a_j|$

$\|a\|_2 = \sqrt{\sum_{j=1}^{d} |a_j|^2}$

$\|a\|_0 = \sum_{j=1}^{d} |a_j|^0 = d$

$\|a\|_\infty = \max_j |a_j|$

# 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

**Repeat** until there is no change in the cost.

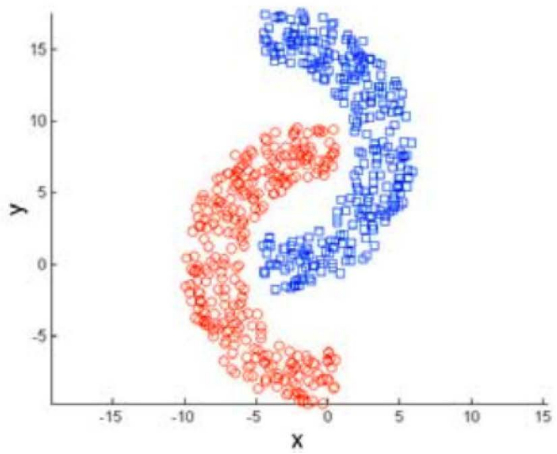
- For each $j$: $C_j \leftarrow \{x \in S$ whose closest center is $\mathbf{c}_j\}$
  - For each $j$: $\mathbf{c}_j \leftarrow$ mean of $C_j$

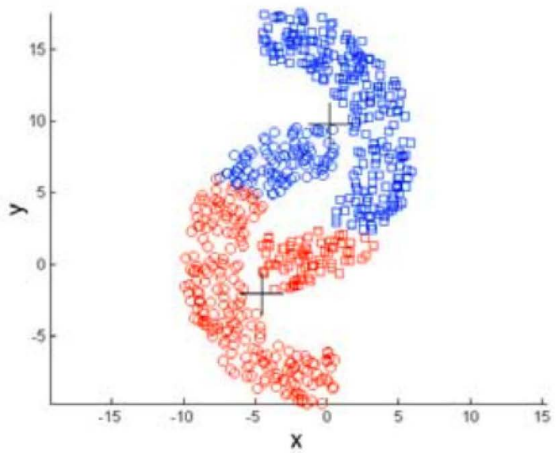
Each round takes time $O(nkd)$.

- 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

$A^TA = (RD^{\frac{1}{2}})^T (RD^{\frac{1}{2}})$

$= D^{\frac{1}{2}} (R^TR) D^{\frac{1}{2}} = I$

$k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle = \phi(x_i) \cdot \phi(x_j)$

$= \phi^T(x_i) \cdot \phi(x_j)$

$K_{:,} \; k_{ij} = \langle \phi(x_i), \phi(x_j) \rangle$

$A \in \mathbb{R}^{m \times n}$

$\|A\|_F^2 = Tr(A^TA)$

$= Tr(AA^T)$

$= \langle A, A \rangle$

$\min_{R,C} \; Q(R,C) = \|X - RC\|_F^2$

$C = DR^TX$

$Q(R) = \|X - RDR^TX\|_F^2$

$= \langle X - RDR^TX, \; X - RDR^TX \rangle$

$= \langle X, X \rangle - 2\langle X, RDR^TX \rangle + \langle RDR^TX, \; RDR^TX \rangle$

$= \langle X, X \rangle - \langle X, RDR^TX \rangle$

$= Tr(RD^{\frac{1}{2}} I^T X X^T I D^{\frac{1}{2}} R^T)$

$\max_A \; Tr(A^T XX^T A)$

s.t. $A^TA = I$

$\|a - b\|_2^2 = \langle a-b, a-b \rangle$

$Tr(ABC^T) = \langle AB, C \rangle$

$= Tr(BC^TA) = \langle B, A^TC \rangle$

$\langle RDR^TX, \; RDR^TX \rangle$

$= \langle X, \; (RDR^T)^T RDR^TX \rangle$

$RDR^TRDR^T$

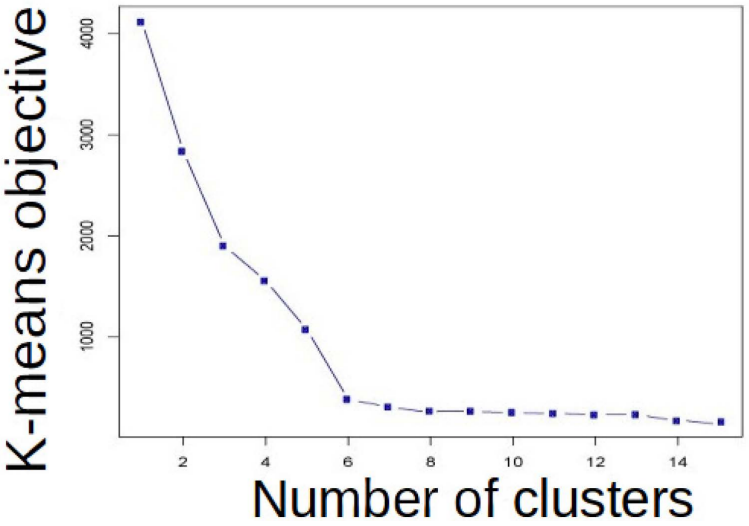
$A_1 = RDR^T$

$D = (R^TR)^{-1}$

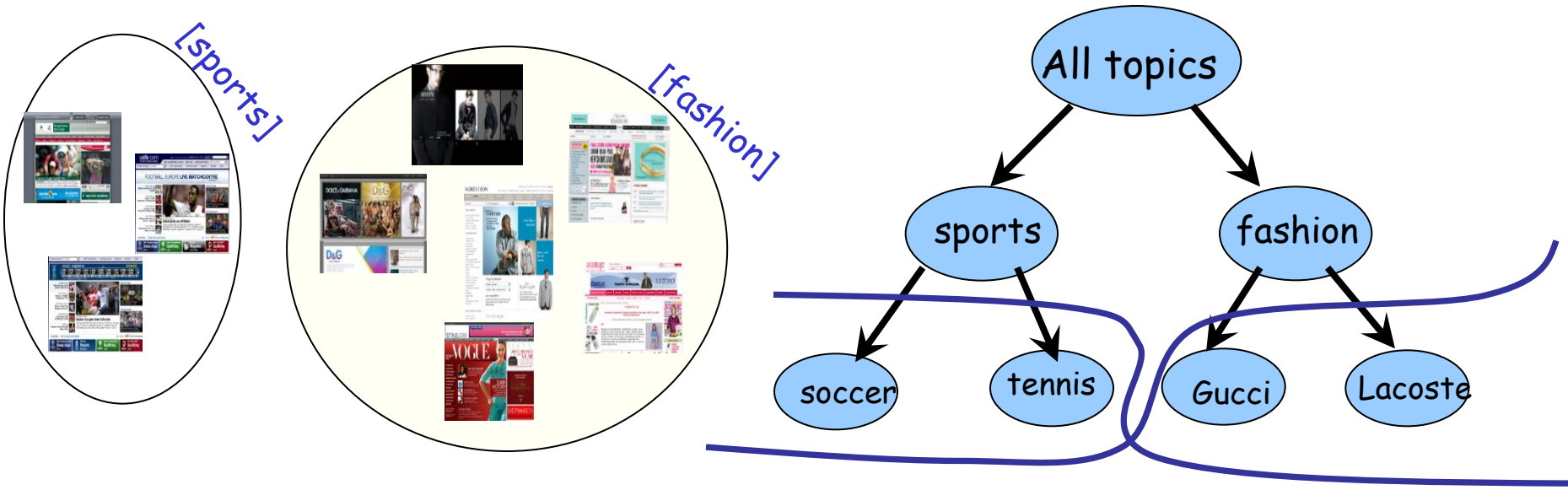
$RI_k = I_n$

# 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




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

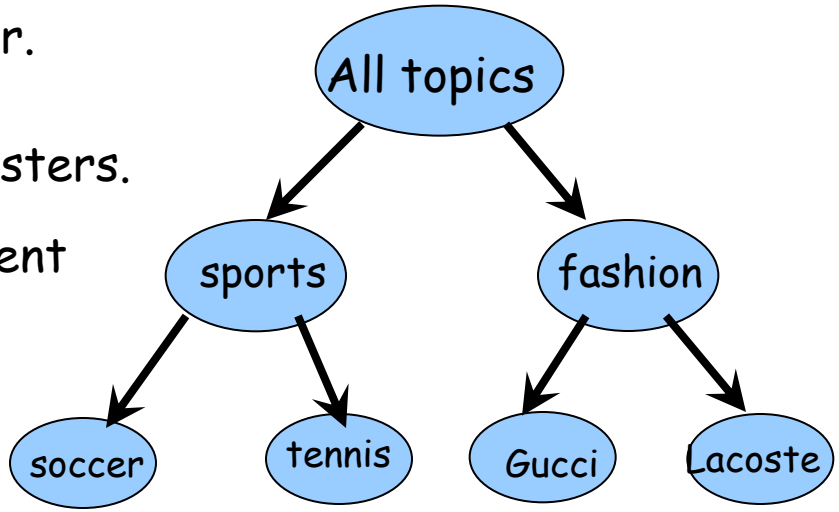
# Hierarchical Clustering

## 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.

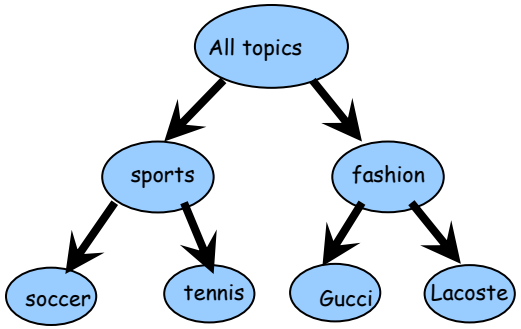


# Bottom-Up (agglomerative)

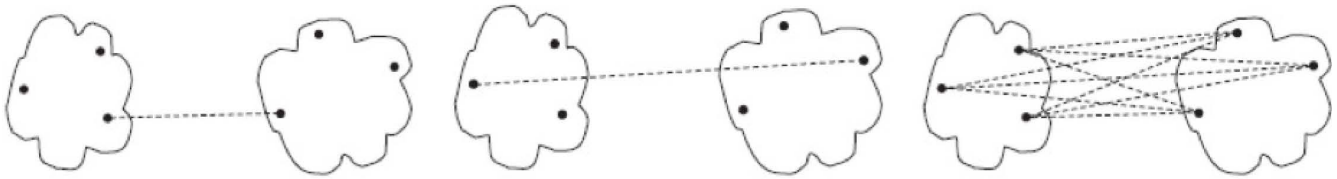
Have a distance measure on pairs of objects.

d(x,y) – distance between x and y

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




- Single linkage: $\text{dist}(A, B) = \min_{x \in A, x' \in B'} \text{dist}(x, x')$
- Complete linkage: $\text{dist}(A, B) = \max_{x \in A, x' \in B'} \text{dist}(x, x')$
- Average linkage: $\text{dist}(A, B) = \text{avg}_{x \in A, x' \in B'} \text{dist}(x, x')$
- Ward's method



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

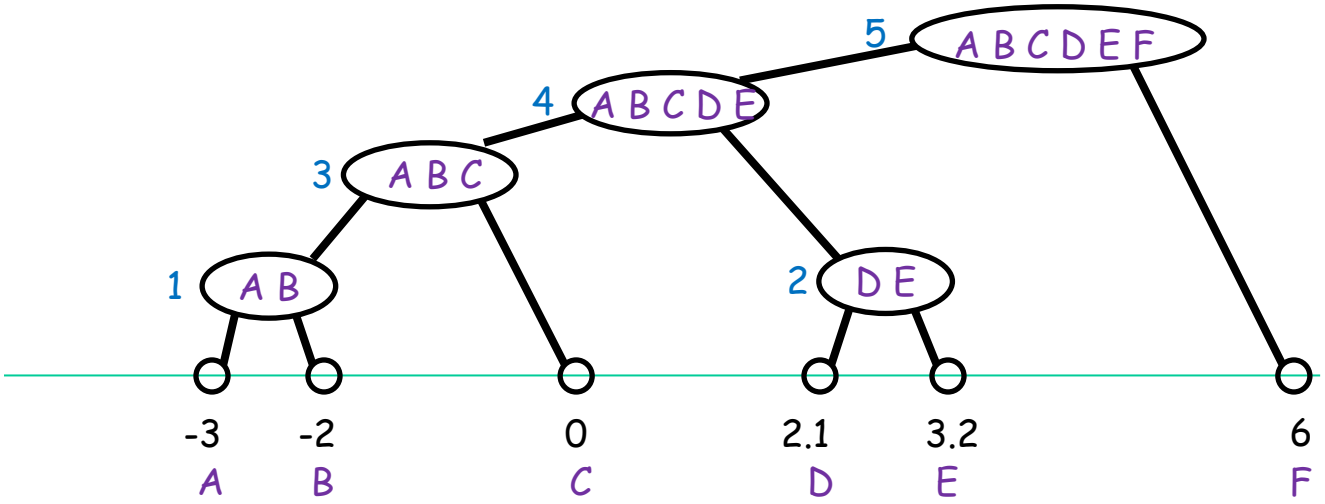
# Single Linkage

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

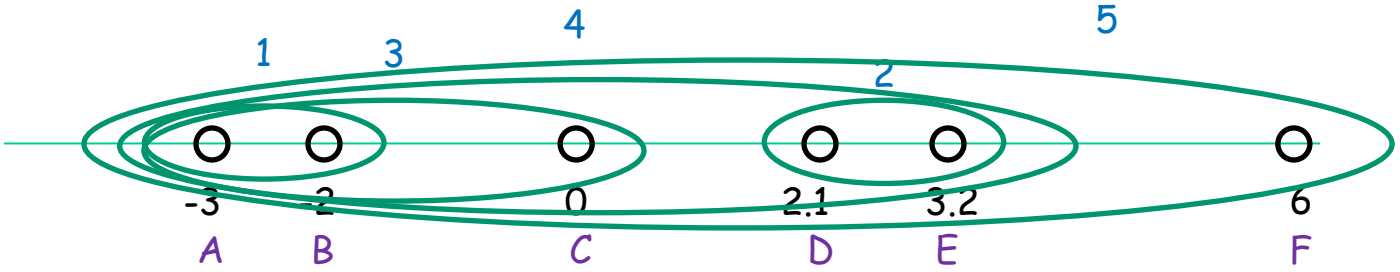
Bottom-up (agglomerative)

- 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).



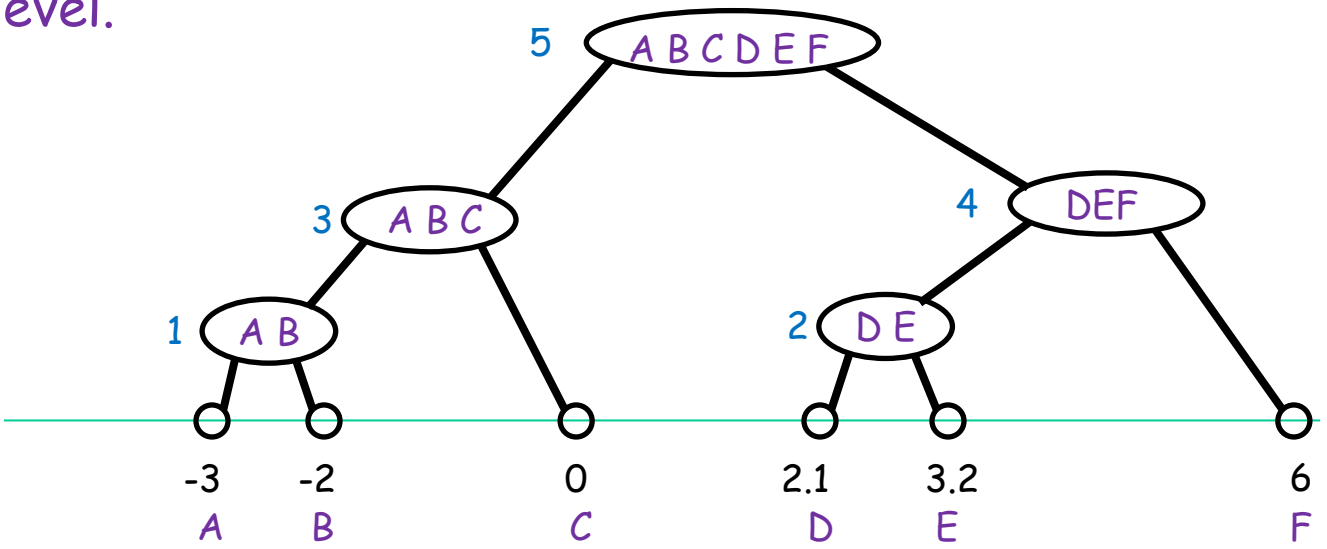
# Complete Linkage

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.



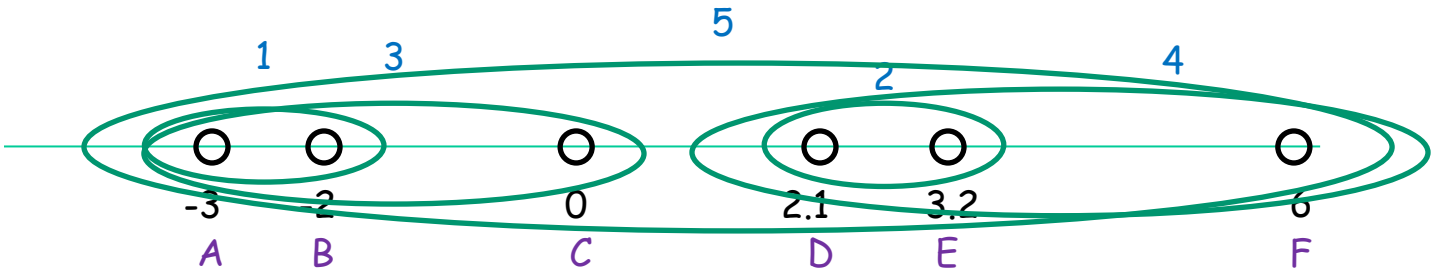
# Complete Linkage

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

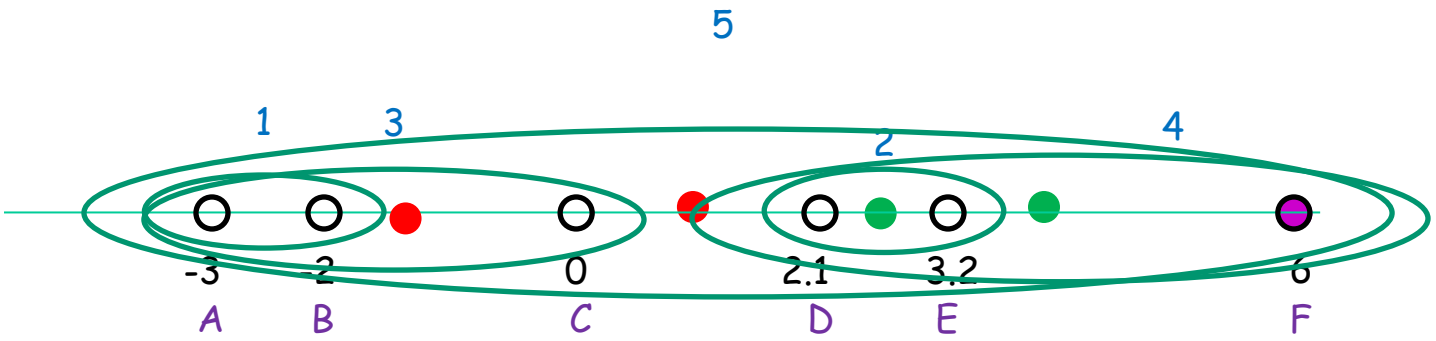
Bottom-up (agglomerative)

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

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

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

Works well in practice.



# Running time

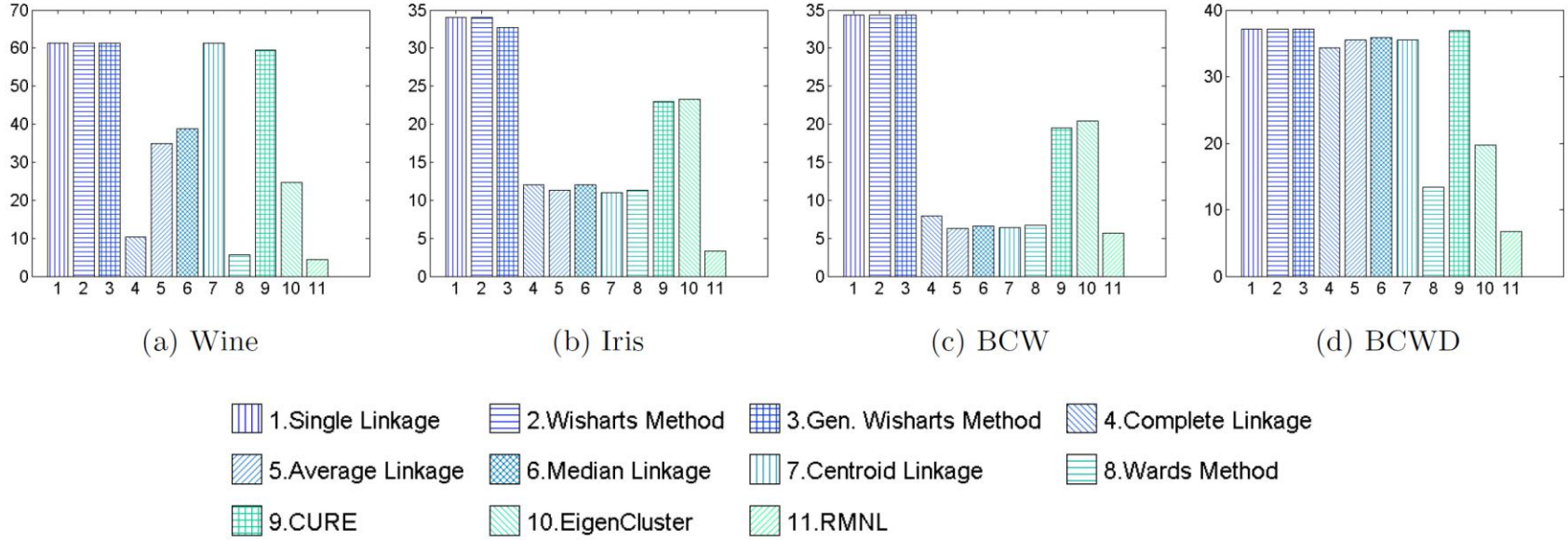
- Each algorithm starts with N clusters, and performs N-1 merges.
- For each algorithm, computing $dist(C, C')$ can be done in time $O(|C| \cdot |C'|)$. (e.g., examining $dist(x, x')$ for all $x \in C, x' \in C'$)
- Time to compute all pairwise distances and take smallest is $O(N^2)$.
- Overall time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

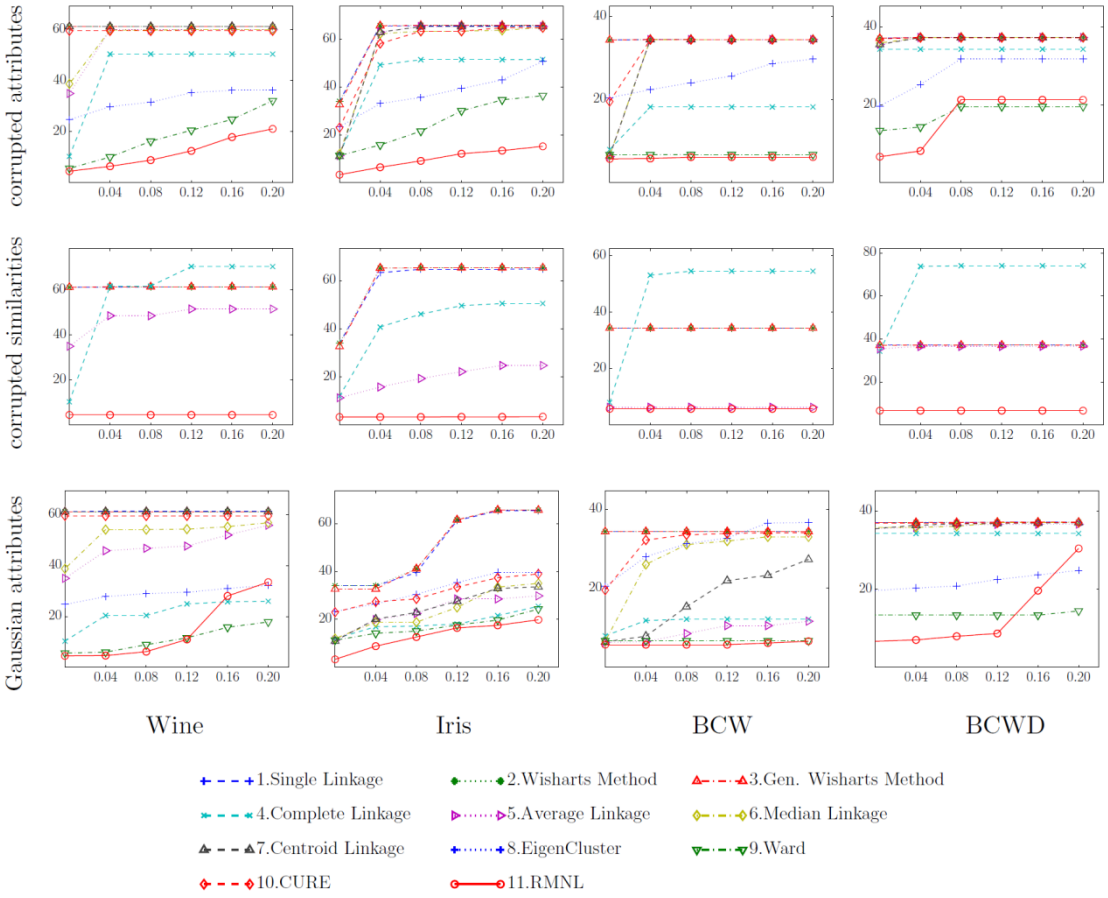
[BLG, JMLR'15]




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

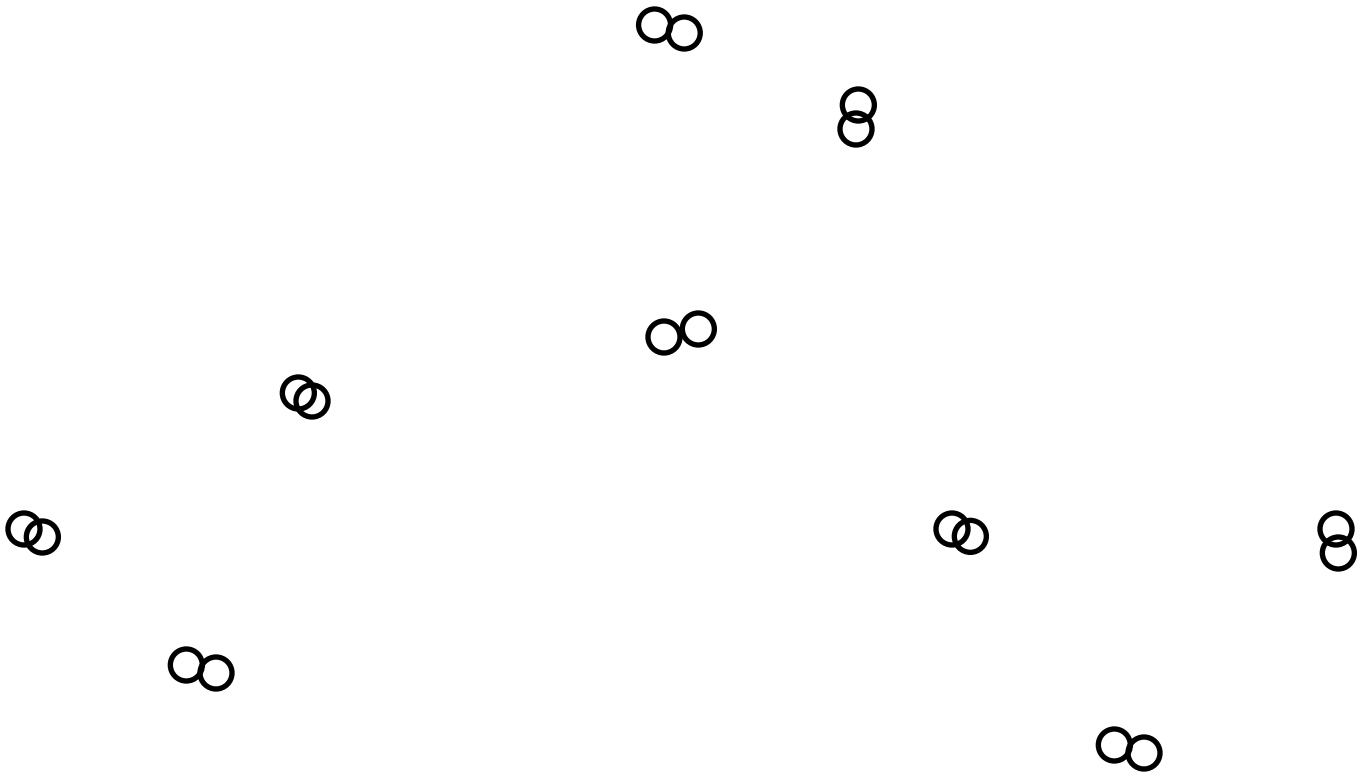
- Partitional Clustering. k-means and k-means ++
  - Lloyd's method
  - Initialization techniques (random, furthest traversal, k-means++)

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

# Additional Slides

# Smoothed analysis model

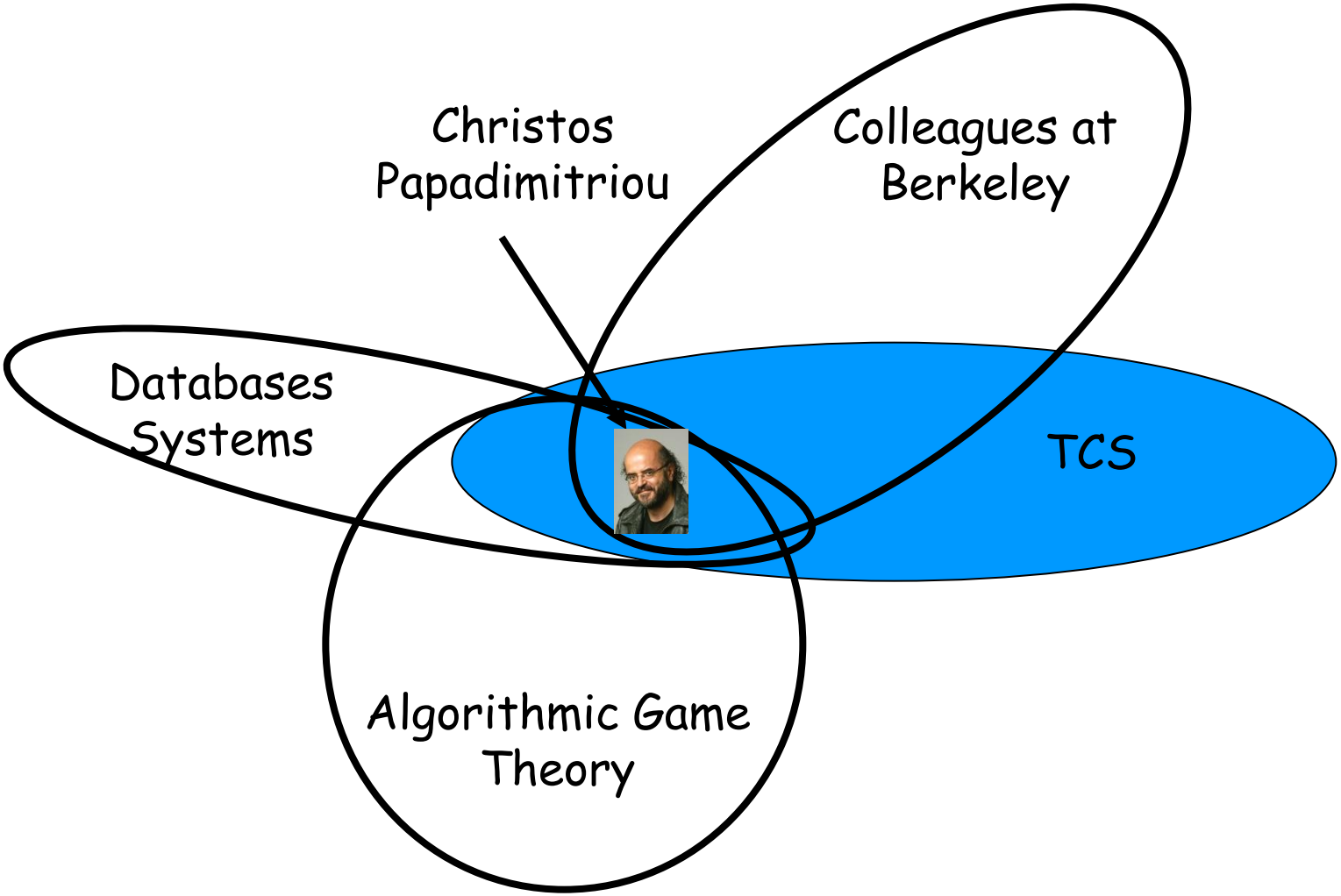
- Imagine a worst-case input.
- But then add small Gaussian perturbation to each data point.



# Smoothed analysis model

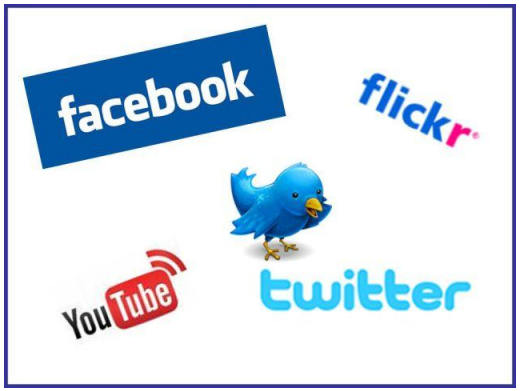
- Imagine a worst-case input.
- But then add small Gaussian perturbation to each data point.
- Theorem [Arthur-Manthey-Roglin 2009]:
  - E[number of rounds until Lloyd's converges] if add Gaussian perturbation with variance $\sigma^2$ is polynomial in $n$, $1/\sigma$.
  - The actual bound is : $O\left(\frac{n^{34}k^{34}d^8}{\sigma^6}\right)$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