

# Lecture outline

- Density-based clustering (DB-Scan)
  - Reference: Martin Ester, Hans-Peter Kriegel, Jorg Sander, Xiaowei Xu: A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. KDD 2006
- Co-clustering (or bi-clustering)
- References:
  - A. Anagnostopoulos, A. Dasgupta and R. Kumar: Approximation Algorithms for co-clustering, PODS 2008.
  - K. Puolamaki. S. Hanhijarvi and G. Garriga: An approximation ratio for biclustering, Information Processing Letters 2008.

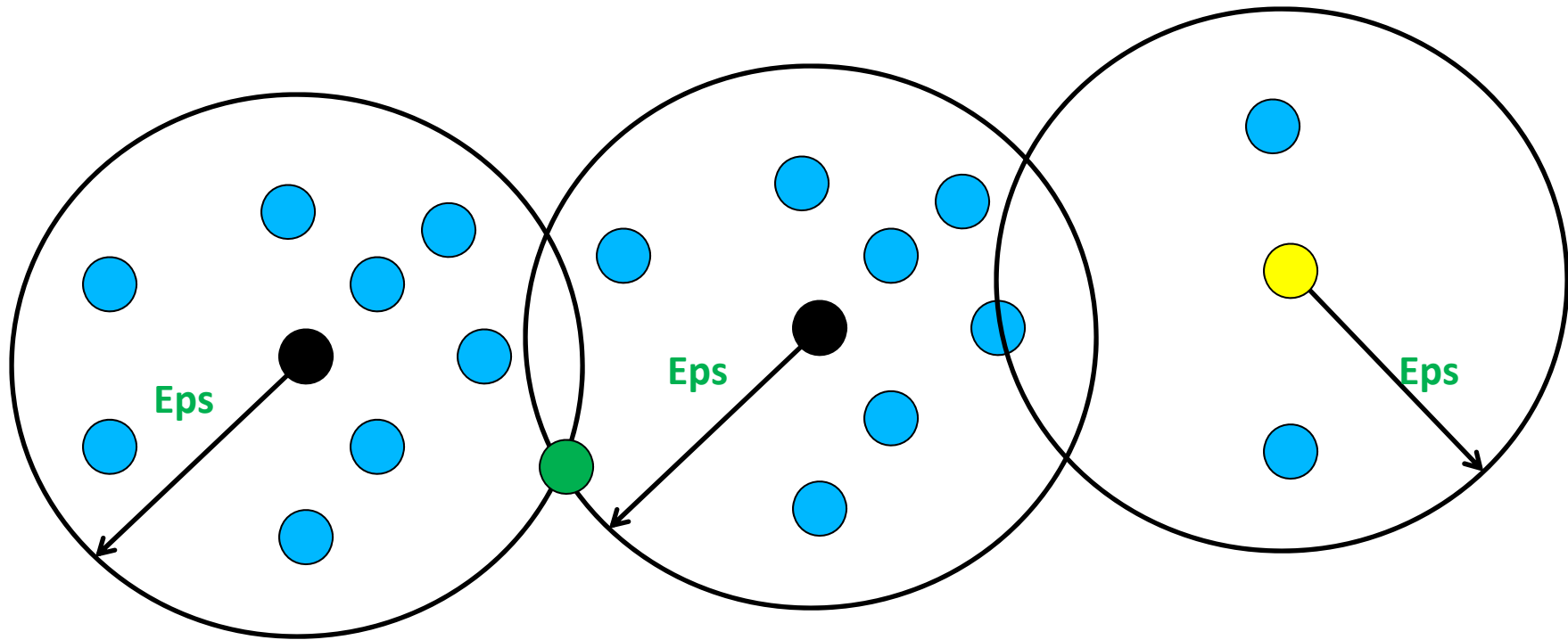
# Density-Based Clustering Methods

- Clustering based on density (local cluster criterion), such as density-connected points
- Major features:
  - Discover clusters of arbitrary shape
  - Handle noise

# Classification of points in density-based clustering

- **Core points:** Interior points of a density-based cluster. A point  $p$  is a core point if for distance **Eps** :
  - $|N_{\text{Eps}}(p) = \{q \mid \text{dist}(p,q) \leq \varepsilon\}| \geq \text{MinPts}$
- **Border points:** Not a core point but within the neighborhood of a core point (it can be in the neighborhoods of many core points)
- **Noise points:** Not a core or a border point

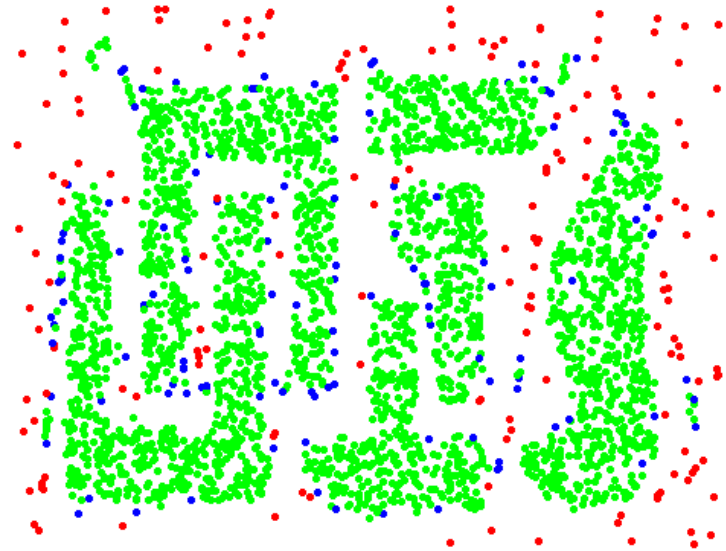
# Core, border and noise points



# Core, Border and Noise points

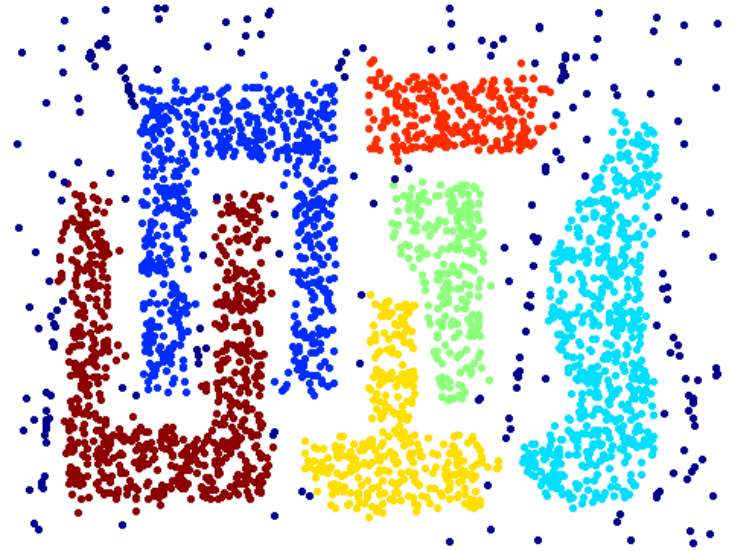


Original Points



Point types: core border  
and noise

# Clusters output by DBScan



- Resistant to Noise
- Can handle clusters of different shapes and sizes

# Classification of points in density-based clustering

- **Core points:** Interior points of a density-based cluster. A point  $p$  is a core point if for distance **Eps** :
  - $|N_{\text{Eps}}(p) = \{q \mid \text{dist}(p,q) \leq \varepsilon\}| \geq \text{MinPts}$
- **Border points:** Not a core point but within the neighborhood of a core point (it can be in the neighborhoods of many core points)
- **Noise points:** Not a core or a border point

# DBSCAN: The Algorithm

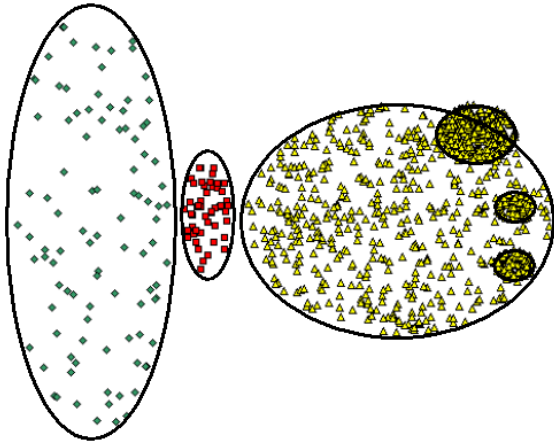
- Label all points as ***core***, ***border***, or ***noise*** points
- Eliminate noise points
- Put an edge between all core points that are within ***Eps*** of each other
- Make each group of connected core points into a separate cluster
- Assign each border point to one of the cluster of its associated core points



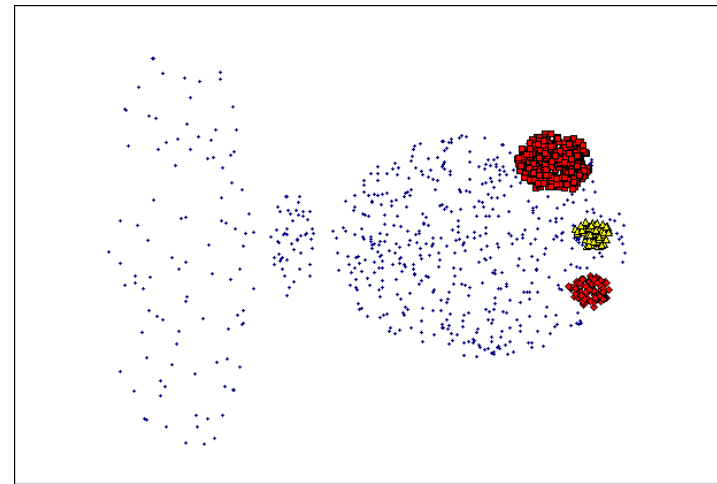
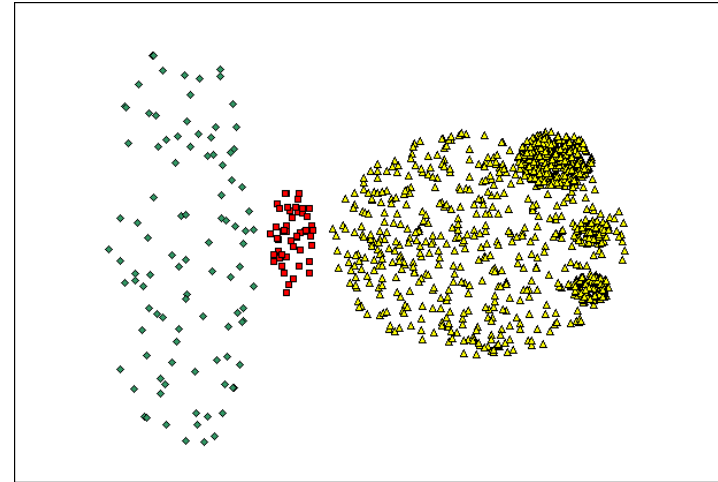
# Time and space complexity of DBSCAN

- For a dataset  $X$  consisting of  $n$  points, the time complexity of DBSCAN is  $O(n \times \text{time to find points in the Eps-neighborhood})$
- Worst case  $O(n^2)$
- In low-dimensional spaces  $O(n \log n)$ ; efficient data structures (e.g., *kd-trees*) allow for efficient retrieval of all points within a given distance of a specified point

# When DBSCAN Does NOT Work Well

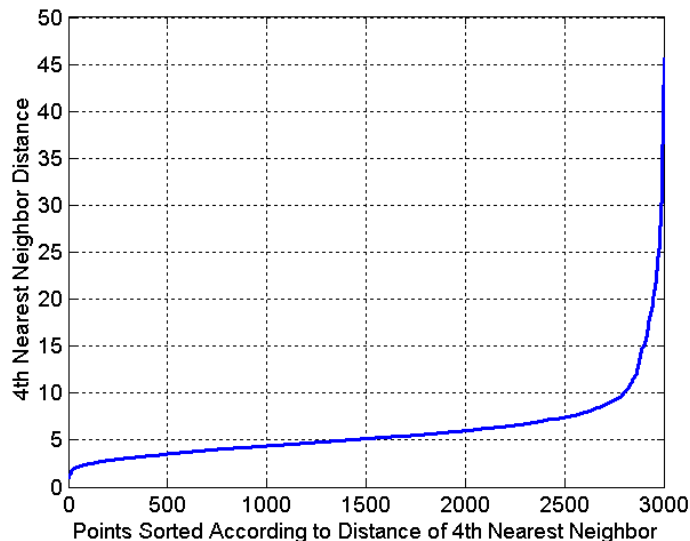


DBScan can fail to identify clusters of varying densities

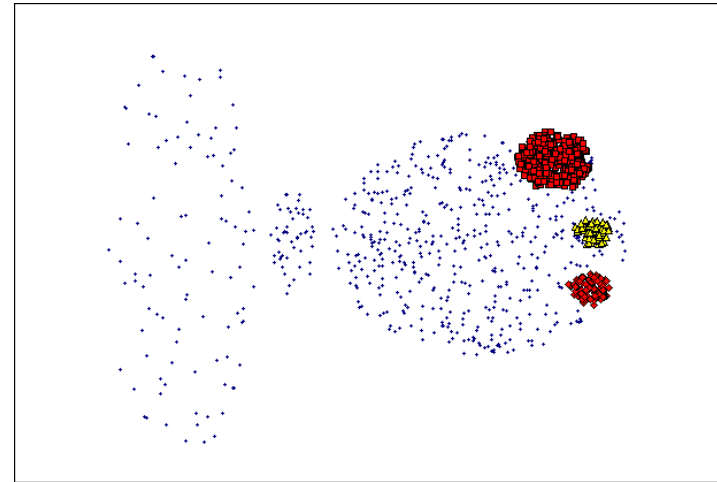
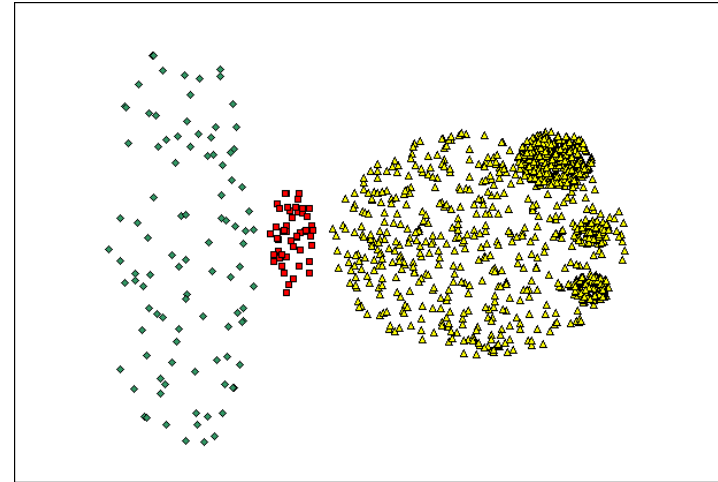
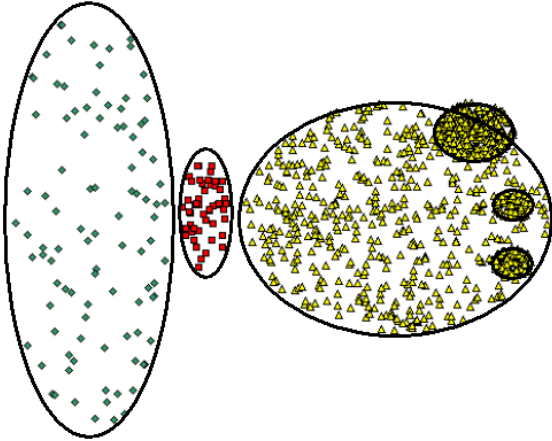


# Determining EPS and MinPts

- Idea is that for points in a cluster, their  $k^{\text{th}}$  nearest neighbors are at roughly the same distance
- Noise points have the  $k^{\text{th}}$  nearest neighbor at farther distance
- So, plot sorted distance of every point to its  $k^{\text{th}}$  nearest neighbor

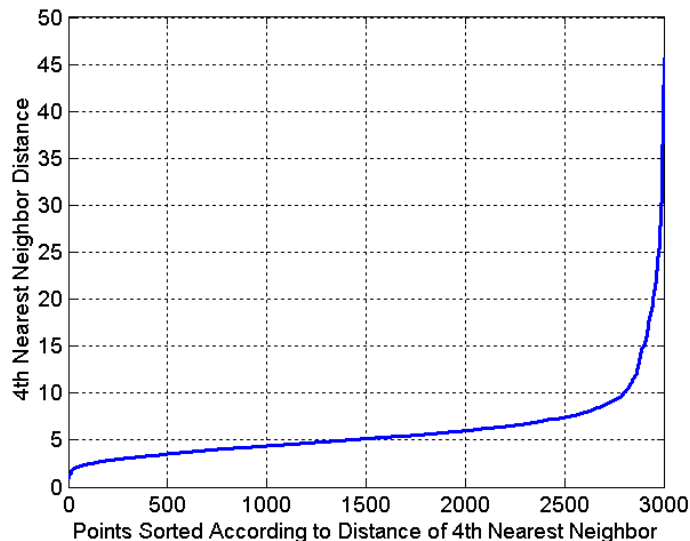


# When DBSCAN Does NOT Work Well



# Determining EPS and MinPts

- Idea is that for points in a cluster, their  $k^{\text{th}}$  nearest neighbors are at roughly the same distance
- Noise points have the  $k^{\text{th}}$  nearest neighbor at farther distance
- So, plot sorted distance of every point to its  $k^{\text{th}}$  nearest neighbor



# Strengths and weaknesses of DBSCAN

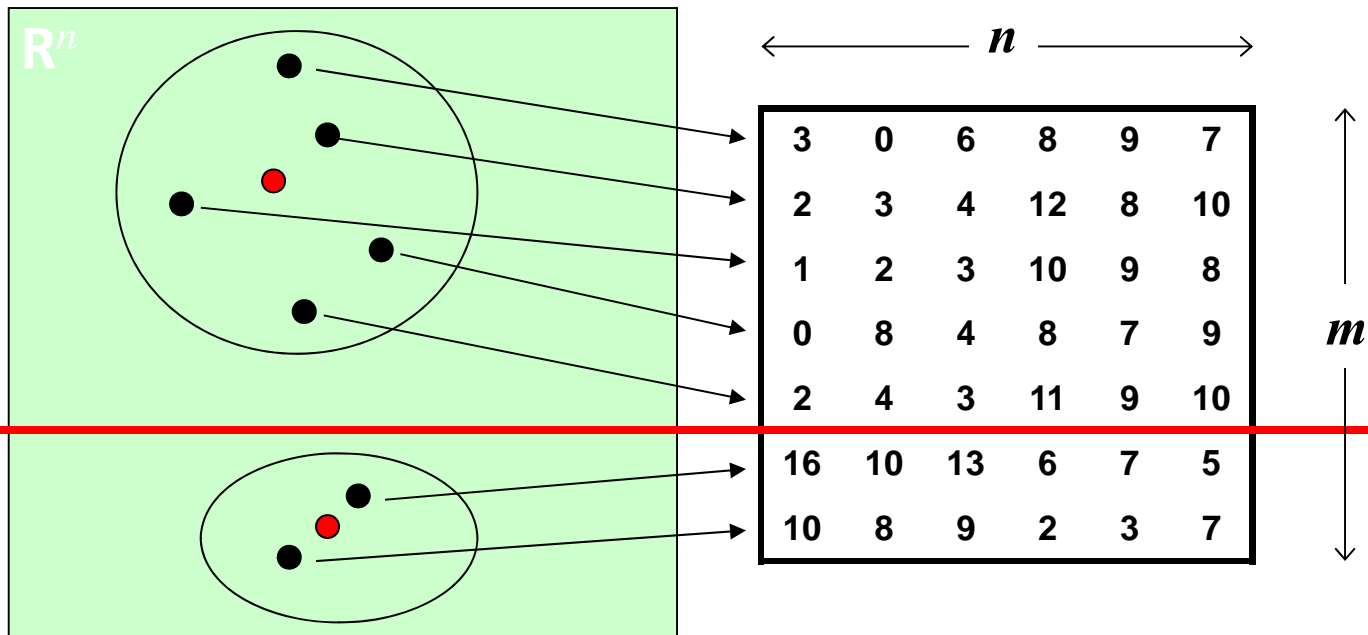
- Resistant to noise
- Finds clusters of arbitrary shapes and sizes
- Difficulty in identifying clusters with varying densities
- Problems in high-dimensional spaces; notion of density unclear
- Can be computationally expensive when the computation of nearest neighbors is expensive

# Lecture outline

- Density-based clustering
- Co-clustering (or bi-clustering)
- References:
  - A. Anagnostopoulos, A. Dasgupta and R. Kumar: Approximation Algorithms for co-clustering, PODS 2008.
  - K. Puolamaki, S. Hanhijarvi and G. Garriga: An approximation ratio for biclustering, Information Processing Letters 2008.

# Clustering

- $m$  points in  $\mathbf{R}^n$
- Group them to  $k$  clusters
- Represent them by a matrix  $A \in \mathbf{R}^{m \times n}$ 
  - A point corresponds to a row of  $A$
- **Cluster:** Partition the rows to  $k$





# Co-Clustering

- **Co-Clustering:** Cluster rows and columns of  $A$  simultaneously:

$\ell = 2$

$k = 2$

3	0	6	8	9	7
2	3	4	12	8	10
1	2	3	10	9	8
0	8	4	8	9	7
2	4	3	11	9	10
16	10	13	6	7	5
10	8	9	2	3	7

$A$

Co-cluster

# Motivation: Sponsored Search

The screenshot shows a Yahoo! search results page for the query "car insurance". The search bar at the top contains "car insurance" and the Yahoo! logo is in the top right. Below the search bar, there are navigation links for "Web", "Images", "Video", "Local", "Shopping", and "more". The search results are displayed in two columns. The left column shows organic search results, including "Allstate - Auto Insurance Quote, Anonymous Online Car Insurance ..." and "Esurance.com - Online Auto Quotes, Comparisons and Resources". The right column shows sponsored results, which are highlighted with a red box and labeled "Ads" with red arrows. The sponsored results include "AIG Auto Insurance - Instant Quotes", "California Insurance Quotes Online", "California Car Insurance", "Auto Insurance Quotes", and "USAA Auto Insurance".

Web | Images | Video | Local | Shopping | more

car insurance Search Options

YAHOO!

1-10 of 279,000,000 for car insurance (About) - 0.39 sec

Also try: [car insurance quotes](#), [cheap car insurance](#), [geico car insurance](#), [More...](#)

**SPONSORED RESULTS**

**GEICO Car Insurance**  
www.GEICO.com - GEICO could save you over \$500. Get an instant insurance quote.

**Progressive Car Insurance: Official Site**  
www.progressive.com - Get our rates and our top competitors'. You could save hundreds.

**Esurance - Online Auto Insurance**  
www.esurance.com - Get a quote, compare quotes and buy your policy instantly online.

**AAA Insurance**  
www.aaa.com/insurance - Get 10% off your auto policy when you insure your auto & home with us.

1. **Allstate - Auto Insurance Quote, Anonymous Online Car Insurance ...**  
Save on Car Insurance with Your Choice Auto Insurance: Accident Forgiveness, Deductible Rewards, Safe Driver Bonus. & New Car Replacement.  
[Allstate Auto Insurance near you](#)  
auto-insurance.allstate.com - 53k - Cached

2. **Esurance.com - Online Auto Quotes, Comparisons and Resources**  
At Esurance, save hundreds on your auto insurance today by comparing quotes online.  
Quick Links: [Get A Quote](#)  
www.esurance.com

**SPONSOR RESULTS**

**AIG Auto Insurance - Instant Quotes**  
Instant, online, accurate car insurance quotes direct from AIG Auto.  
www.aigauto.com

**California Insurance Quotes Online**  
Compare auto insurance quotes from top companies online.  
www.Insurance.com

**California Car Insurance**  
Buy, print car insurance in 10 minutes- with accidents, violations.  
www.TheGeneral.com

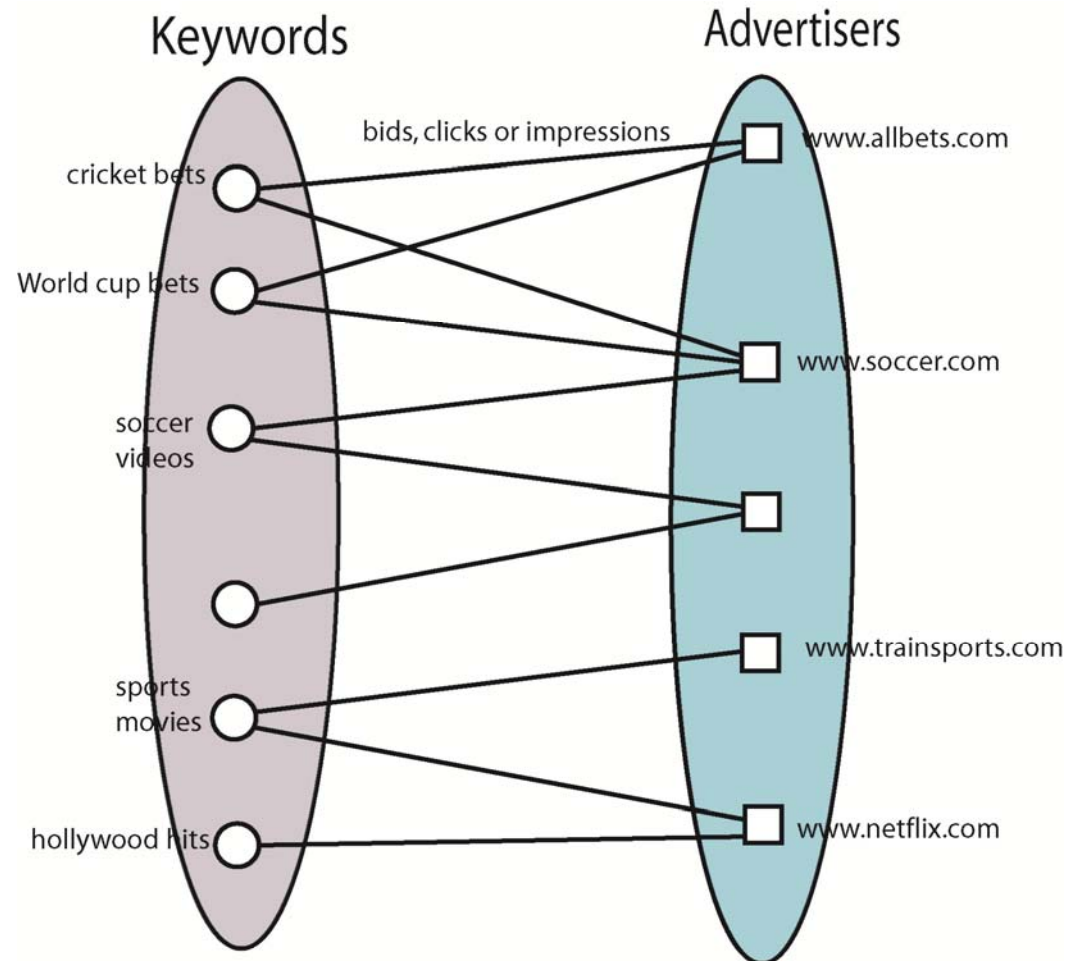
**Auto Insurance Quotes**  
Get Free Quote from Liberty Mutual. No Obligation. Apply in Minutes.  
www.LibertyMutual.com

**USAA Auto Insurance**  
Switch And You Could Save More.

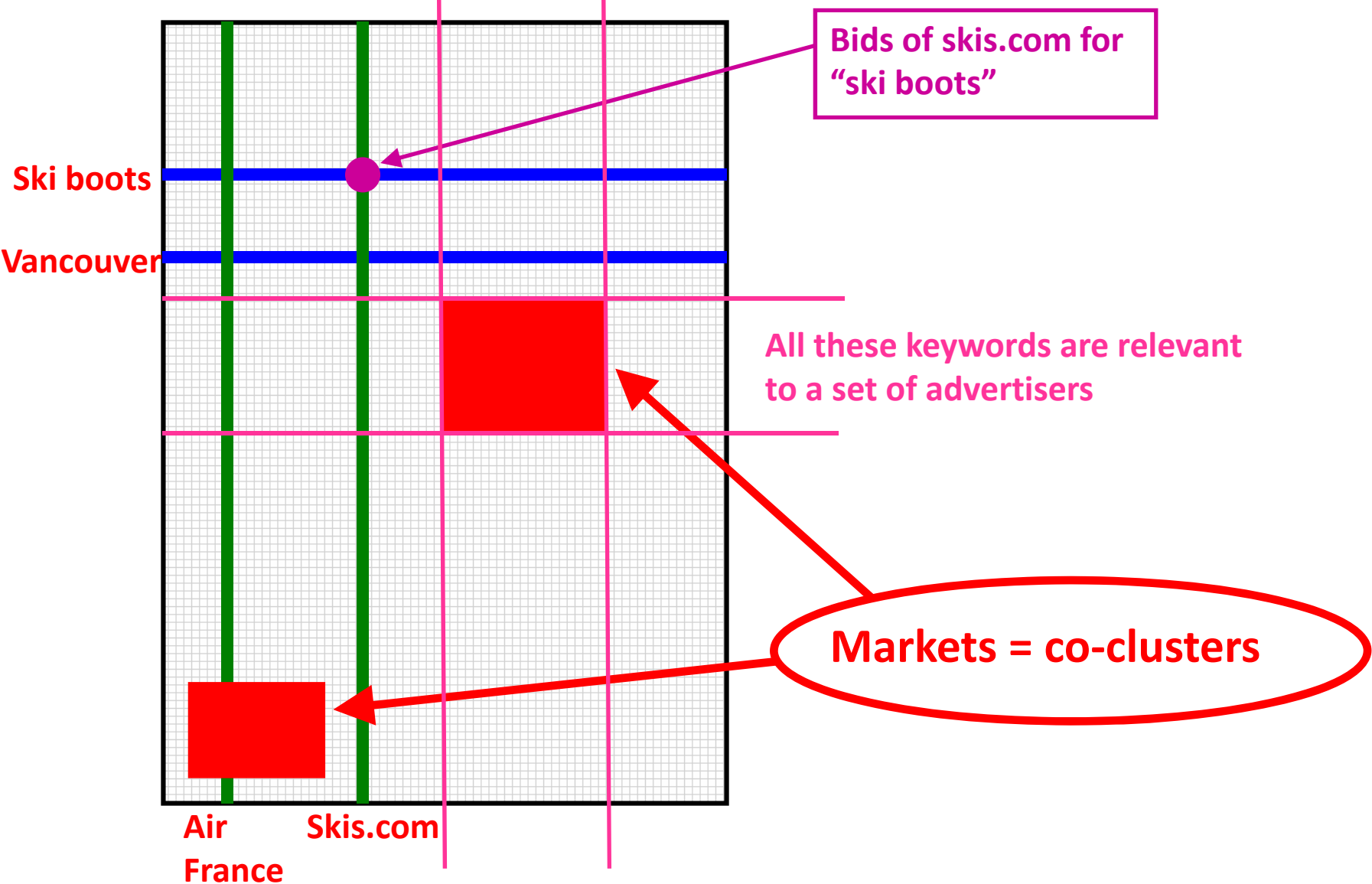
- Advertisers bid on keywords
- A user makes a query
- Show ads of advertisers that are relevant and have high bids
- User clicks or not an ad

# Motivation: Sponsored Search

- For every  $(\text{advertiser}, \text{keyword})$  pair we have:
  - Bid amount
  - Impressions
  - # clicks
- Mine information at query time
  - Maximize # clicks / revenue



# Co-Clusters in Sponsored Search



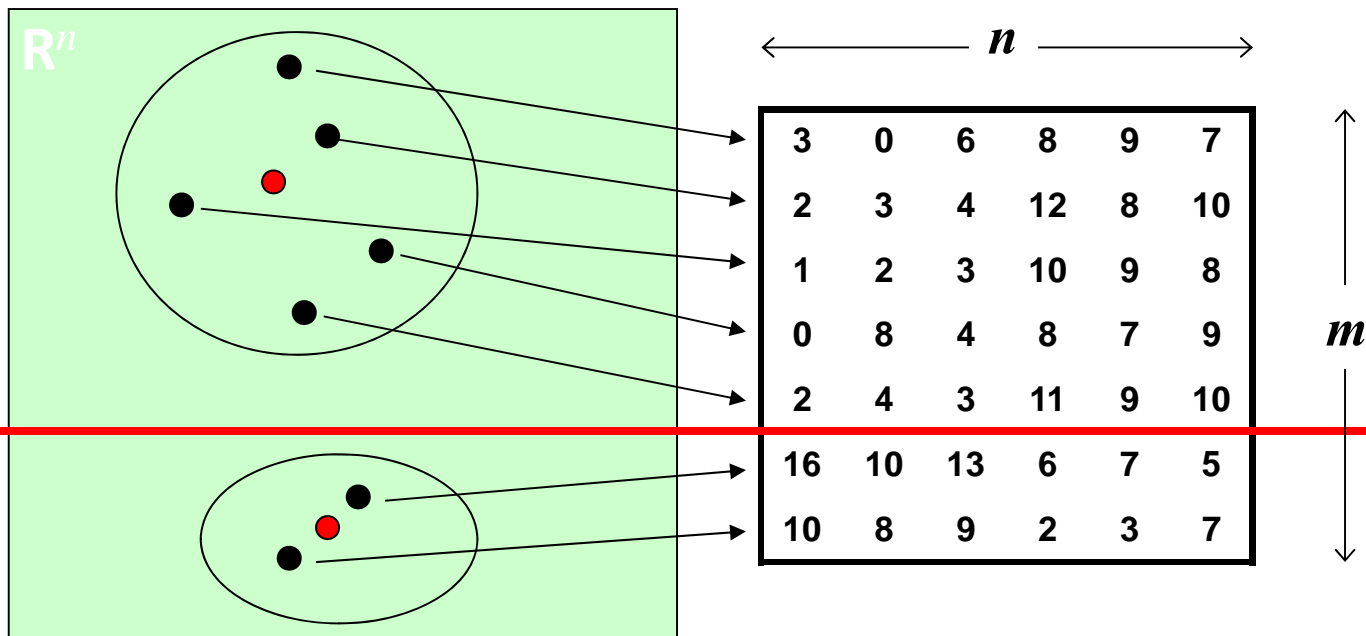
# Co-Clustering in Sponsored Search

## Applications:

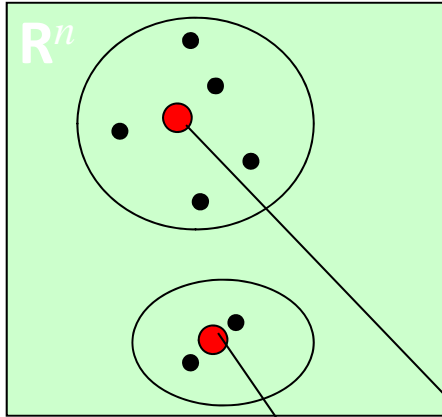
- Keyword suggestion
  - Recommend to advertisers other relevant keywords
- Broad matching / market expansion
  - Include more advertisers to a query
- Isolate submarkets
  - Important for economists
  - Apply different advertising approaches
- Build taxonomies of advertisers / keywords

# Clustering of the rows

- $m$  points in  $\mathbb{R}^n$
- Group them to  $k$  clusters
- Represent them by a matrix  $A \in \mathbb{R}^{m \times n}$ 
  - A point corresponds to a row of  $A$
- **Clustering:** Partitioning of the rows into  $k$  groups



# Clustering of the columns



- $n$  points in  $\mathbb{R}^m$
- Group them to  $k$  clusters
- Represent them by a matrix  $A \in \mathbb{R}^{m \times n}$ 
  - A point corresponds to a column of  $A$
- **Clustering:** Partitioning of the columns into  $k$  groups

3	0	6	8	9	7
2	3	4	12	8	10
1	2	3	10	9	8
0	8	4	8	7	9
2	4	3	11	9	10
16	10	13	6	7	5
10	8	9	2	3	7

3	3	3	9	9	9
3	3	3	9	9	9
3	3	3	9	9	9
3	3	3	9	9	9
3	3	3	9	9	9
11	11	11	5	5	5
11	11	11	5	5	5

# Cost of clustering

3	0	6	8	9	7
2	3	4	12	8	10
1	2	3	10	9	8
0	8	4	8	7	9
2	4	3	11	9	10
16	10	13	6	7	5
10	8	9	2	3	7

Original data points **A**

1.6	3.4	4	9.8	8.4	8.8
1.6	3.4	4	9.8	8.4	8.8
1.6	3.4	4	9.8	8.4	8.8
1.6	3.4	4	9.8	8.4	8.8
1.6	3.4	4	9.8	8.4	8.8
13	9	11	4	5	6
13	9	11	4	5	6

Data representation **A'**

- In **A'** every point in **A** (row or column) is replaced by the corresponding representative (row or column)
- The quality of the clustering is measured by computing distances between the data in the cells of **A** and **A'**.

• **k-means clustering:**  $\text{cost} = \sum_{i=1 \dots n} \sum_{j=1 \dots m} (A(i,j) - A'(i,j))^2$

• **k-median clustering:**  $\text{cost} = \sum_{i=1 \dots n} \sum_{j=1 \dots m} |A(i,j) - A'(i,j)|$



# Co-Clustering

- **Co-Clustering:** Cluster rows and columns of  $A \in \mathbf{R}^{m \times n}$  simultaneously
- $k$  row clusters,  $\ell$  column clusters
- Every cell in  $A$  is represented by a cell in  $A'$
- All cells in the same co-cluster are represented by the same value in the cells of  $A'$

3	0	6	8	9	7
2	3	4	12	8	10
1	2	3	10	9	8
0	8	4	8	9	7
2	4	3	11	9	10
16	10	13	6	7	5
10	8	9	2	3	7

Original data  $A$

3	3	3	9	9	9
3	3	3	9	9	9
3	3	3	9	9	9
3	3	3	9	9	9
3	3	3	9	9	9
11	11	11	5	5	5
11	11	11	5	5	5

Co-cluster representation  $A'$

# Co-Clustering Objective Function

3	0	6	8	9	7
2	3	4	12	8	10
1	2	3	10	9	8
0	8	4	8	7	9
2	4	3	11	9	10
16	10	13	6	7	5
10	8	9	2	3	7

3	3	3	9	9	9
3	3	3	9	9	9
3	3	3	9	9	9
3	3	3	9	9	9
3	3	3	9	9	9
11	11	11	5	5	5
11	11	11	5	5	5

- In  $A'$  every point in  $A$  (row or column) is replaced by the corresponding representative (row or column)
- The quality of the clustering is measured by computing distances between the data in the cells of  $A$  and  $A'$ .

• **k-means Co-clustering:**  $\text{cost} = \sum_{i=1\dots n} \sum_{j=1\dots m} (A(i,j)-A'(i,j))^2$

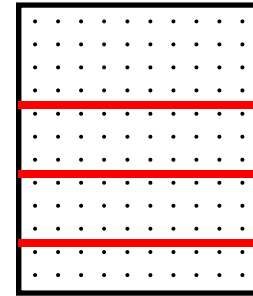
• **k-median Co-clustering:**  $\text{cost} = \sum_{i=1\dots n} \sum_{j=1\dots m} |A(i,j)-A'(i,j)|$

# Some Background

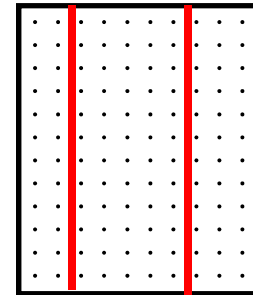
- A.k.a.: biclustering, block clustering, ...
- Many objective functions in co-clustering
  - This is one of the easier
  - Others factor out row-column average (priors)
  - Others based on information theoretic ideas (e.g. KL divergence)
- A lot of existing work, but mostly heuristic
  - $k$ -means style, alternate between rows/columns
  - Spectral techniques

# Algorithm

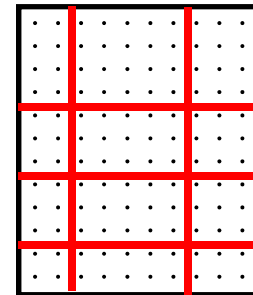
1. Cluster rows of  $A$



2. Cluster columns of  $A$



3. Combine



# Properties of the algorithm

**Theorem 1.** Algorithm with optimal row/column clusterings is 3-approximation to co-clustering optimum.

**Theorem 2.** For  $L_2$  distance function, the algorithm with optimal row/column clusterings is a 2-approximation.

# Algorithm--details

- Clustering of the  $n$  rows of  $A$  assigns every row to a cluster with cluster name  $\{1, \dots, k\}$ 
  - $R(i) = r_i$  with  $1 \leq r_i \leq k$
- Clustering of the  $m$  columns of  $A$  assigns every column to a cluster with cluster name  $\{1, \dots, \ell\}$ 
  - $C(j) = c_j$  with  $1 \leq c_j \leq \ell$
- $A'(i, j) = \{r_i, c_j\}$
- $(i, j)$  is in the same co-cluster as  $(i', j')$  if  $A'(i, j) = A'(i', j')$

# From distance to points: Multi-dimensional scaling

# Multi-Dimensional Scaling (MDS)

- So far we assumed that we know both data points  $\mathbf{X}$  and distance matrix  $\mathbf{D}$  between these points
- What if the original points  $\mathbf{X}$  are not known but only distance matrix  $\mathbf{D}$  is known?
- Can we reconstruct  $\mathbf{X}$  or some approximation of  $\mathbf{X}$ ?



# Problem

- Given distance matrix  $\mathbf{D}$  between  $n$  points
- Find a  $k$ -dimensional representation of every  $\mathbf{x}_i$  point  $i$
- So that  $d(\mathbf{x}_i, \mathbf{x}_j)$  is as close as possible to  $\mathbf{D}(i,j)$

**Why do we want to do that?**

How can we do that? (Algorithm)

# High-level view of the MDS algorithm

- Randomly initialize the positions of  $n$  points in a  $k$ -dimensional space
- Compute pairwise distances  $D'$  for this placement
- Compare  $D'$  to  $D$
- Move points to better adjust their pairwise distances (make  $D'$  closer to  $D$ )
- Repeat until  $D'$  is close to  $D$

# The MDS algorithm

- **Input:**  $n \times n$  distance matrix  $D$
- Random  $n$  points in the  $k$ -dimensional space  $(x_1, \dots, x_n)$
- **stop = false**
- **while not stop**
  - **totalerror = 0.0**
  - For every  $i, j$  compute
    - $D'(i, j) = d(x_i, x_j)$
    - $\text{error} = (D(i, j) - D'(i, j)) / D(i, j)$
    - **totalerror += error**
    - For every dimension  $m$ :  $\text{grad}_{im} = (x_{im} - x_{jm}) / D'(i, j) * \text{error}$
  - If **totalerror** small enough, **stop = true**
  - **If(!stop)**
    - For every point  $i$  and every dimension  $m$ :  $x_{im} = x_{im} - \text{rate} * \text{grad}_{im}$

# Questions about MDS

- Running time of the MDS algorithm
  - $O(n^2I)$ , where  $I$  is the number of iterations of the algorithm
- MDS does not guarantee that metric property is maintained in  $D'$