COMP6237 Data Mining
Lecture 2: Discovering Groups

Zhiwu Huang

Zhiwu.Huang@soton.ac.uk

Lecturer (Assistant Professor) @ VLC of ECS

University of Southampton

Lecture slides available here:

http://comp6237.ecs.soton.ac.uk/jon.html

(Thanks to Prof. Jonathon Hare and Dr. Jo Grundy for providing the lecture materials used to develop the slides.)

Recap – Recommendation System

Content-based

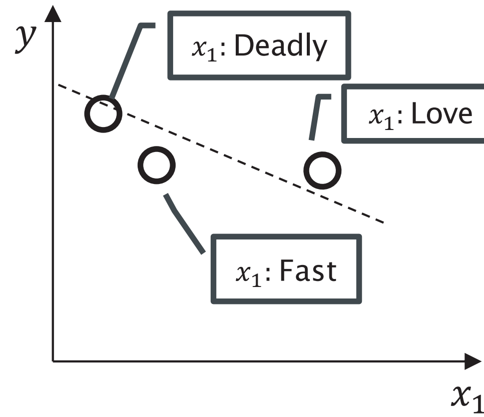
$$\min_{\theta} \frac{1}{m} \sum_{i=1}^{m} (\theta^T X_i - y)^2$$

x(Love)=(x_1, x_2) = (1,0.1)

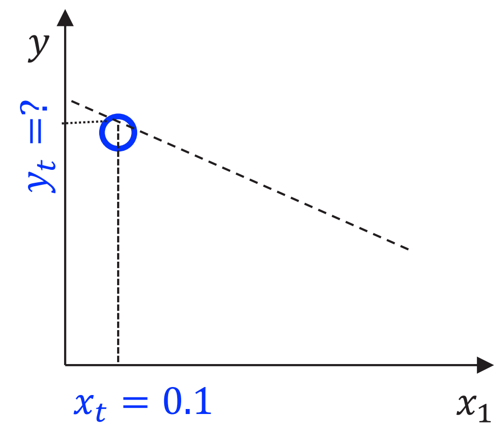
Film	Dave (y)	x_1 romance	x_2 action
Love Really	4	1	0.1
Deadly Weapon	5	0.1	1
Fast and Cross	4	0.2	0.9
Star Battles	?	0.1	1

y_t =?

1. Fitting User Behavior



2. Prediction



User-based Collaborative

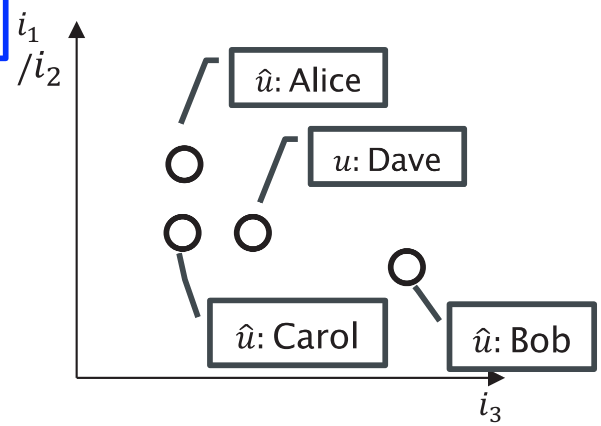
$$r_{u,i} = \frac{\sum_{\hat{u} \in U} sim(u, \hat{u}) r_{\hat{u},i}}{\sum_{\hat{u} \in U} |sim(u, \hat{u})|}$$

$\hat{u}$(Alice) = (i_1, i_3) =(4,5)

	Film	Alice	Bob	Carol	Dave
i_1	Love Really	4	1		4
i_2	Deadly Weapon		1	4	5
i_3	Fast and Cross	5		5	4
	Star Battles	1	5	2	?

$r_{u,t}$ =?

1. Finding Similar Users



2. Prediction

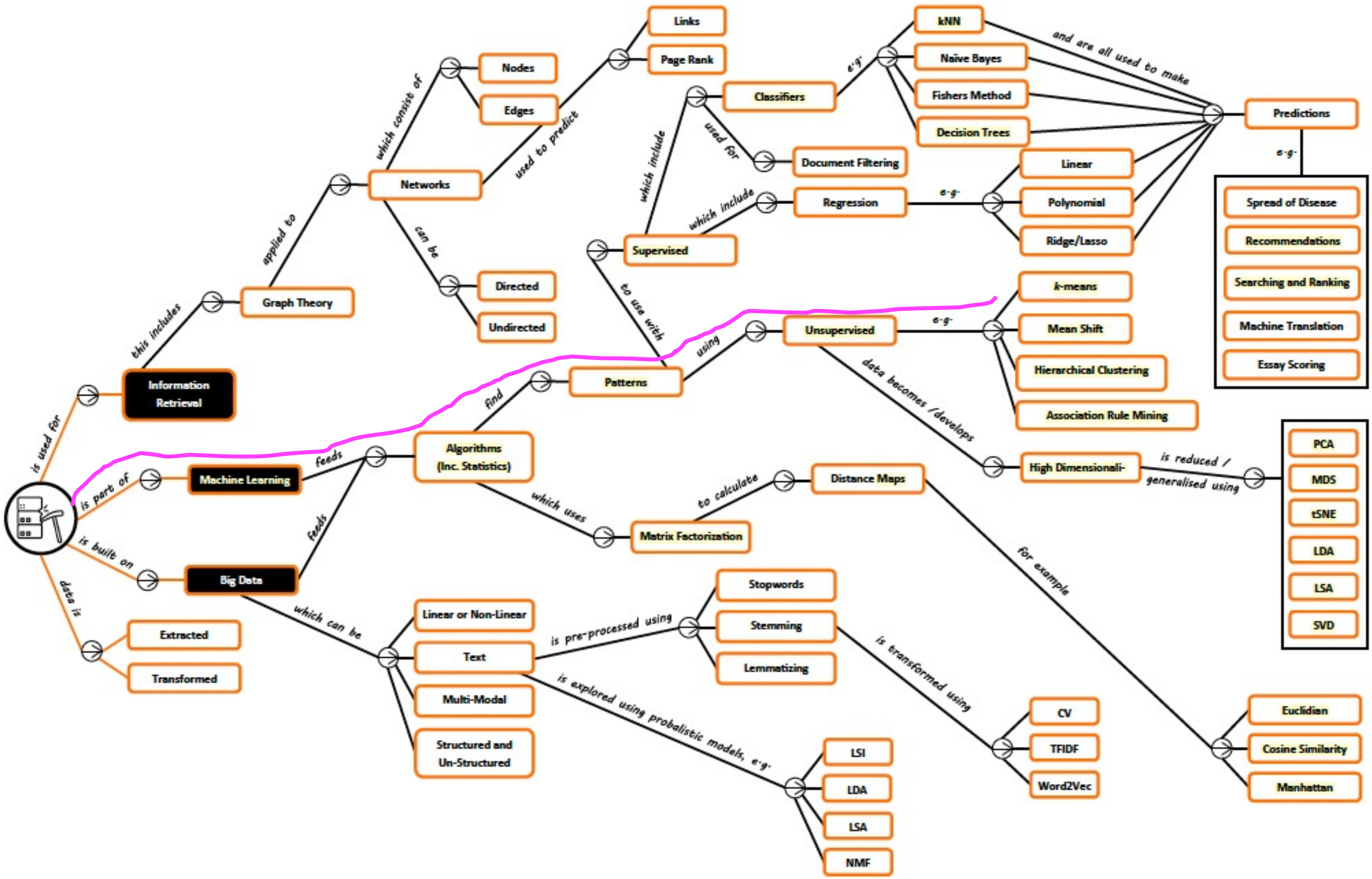
Sim(Dave, Alice) = 0.8
Sim(Dave, Carol) = 0.6

$$r_{u,t} = \frac{0.8 \times 1 + 0.6 \times 2}{0.8 + 0.6}$$

Cosine Similarity:

$$\cos(\theta) = \frac{\mathbf{p}.\mathbf{q}}{|\mathbf{p}||\mathbf{q}|} = \frac{\sum_{i=1}^{N} p_i q_i}{\sqrt{\sum_{i=1}^{N} p_i^2}\sqrt{\sum_{i=1}^{N} q_i^2}}$$

Discovering Groups – **Roadmap**



Discovering Groups – **Textbook**

CHAPTER 3

Discovering Groups

- Programming Collective Intelligence: Building Smart Web 2.0 Applications *T. Segaran*.

Chapter 7

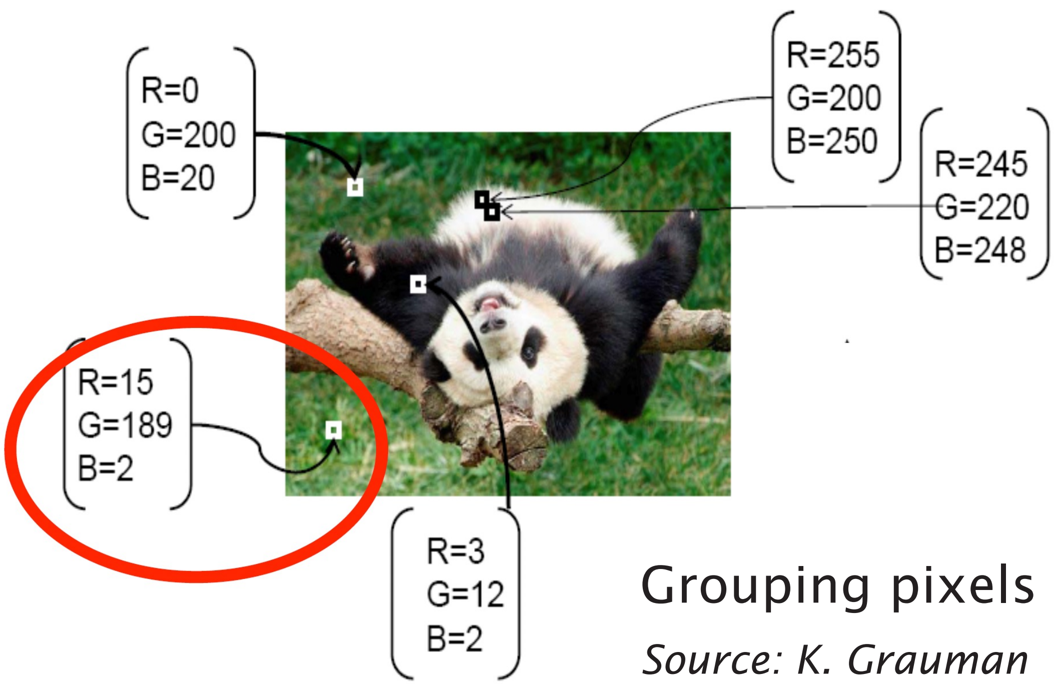
Clustering

Clustering is the process of examining a collection of "points," and grouping the points into "clusters" according to some distance measure. The goal is that points in the same cluster have a small distance from one another, while points in different clusters are at a large distance from one another. A suggestion of

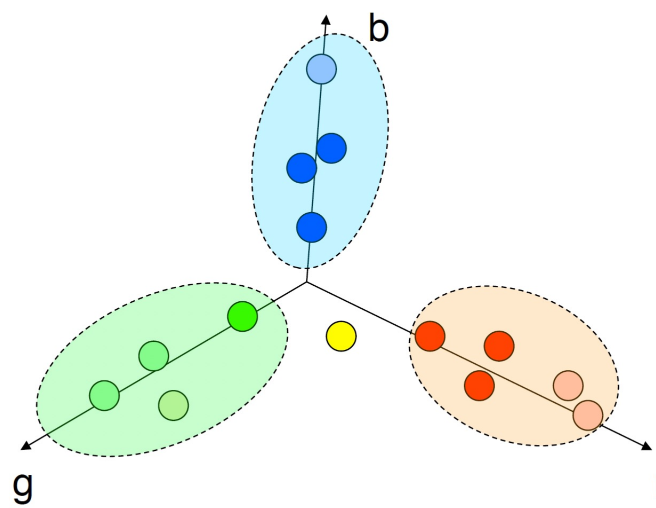
- Mining of Massive Datasets J. Leskovec *et al*

Discovering Groups – **Overview (1/5)**

- Grouping data, just using the feature vectors
  - Unsupervised: no group labels for training
  - Key idea: data with similar features grouped together
  - Can be
    - Soft (allow overlapping groups)
    - Hard (each item assigned to one group)



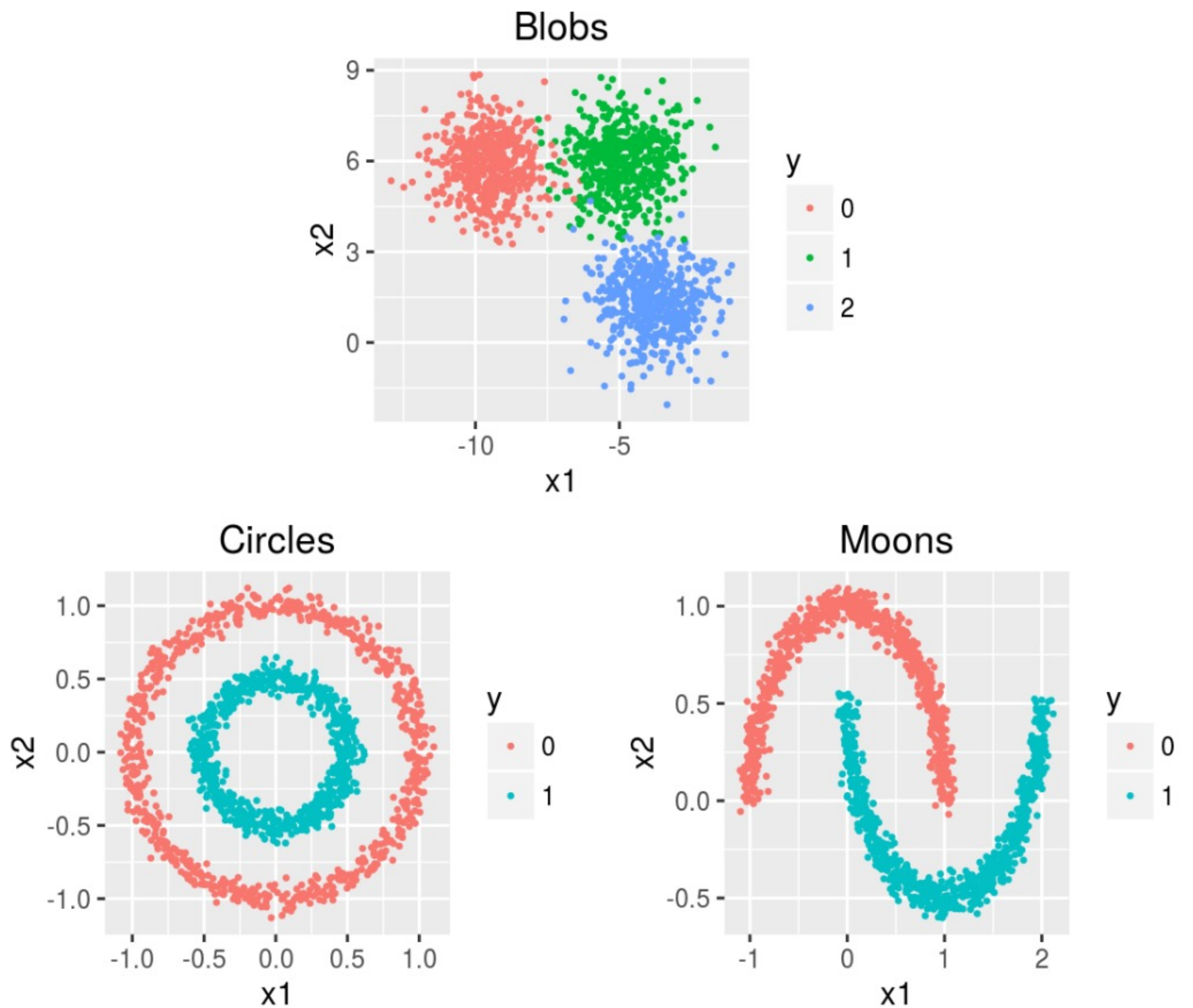
Feature space:
color value (3D)



Grouping pixels based on color similarity

Source: K. Grauman

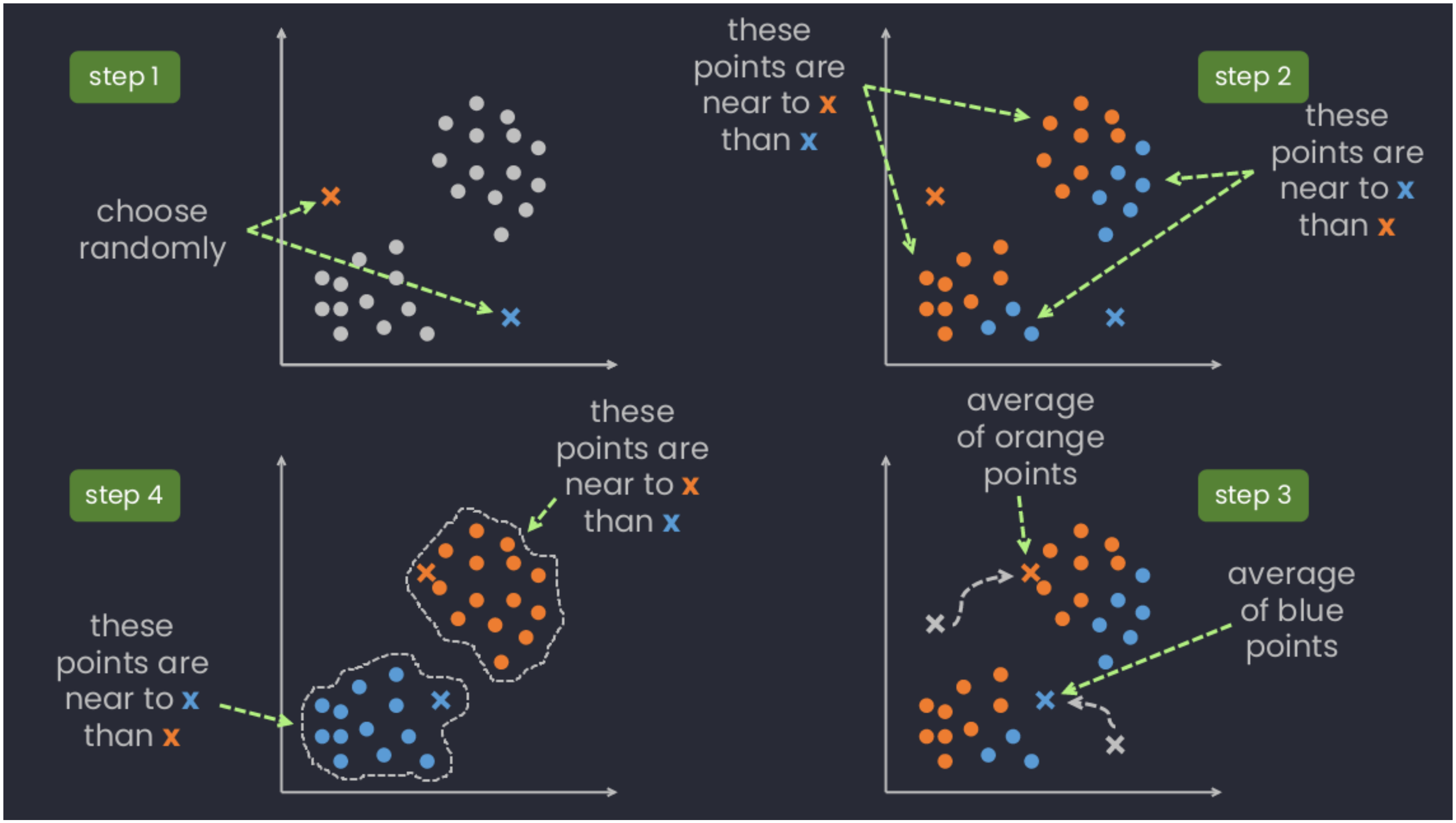
Discovering Groups – **Overview (2/5)**



https://marlabskochi.github.io

Discovering Groups – **Overview (3/5)**

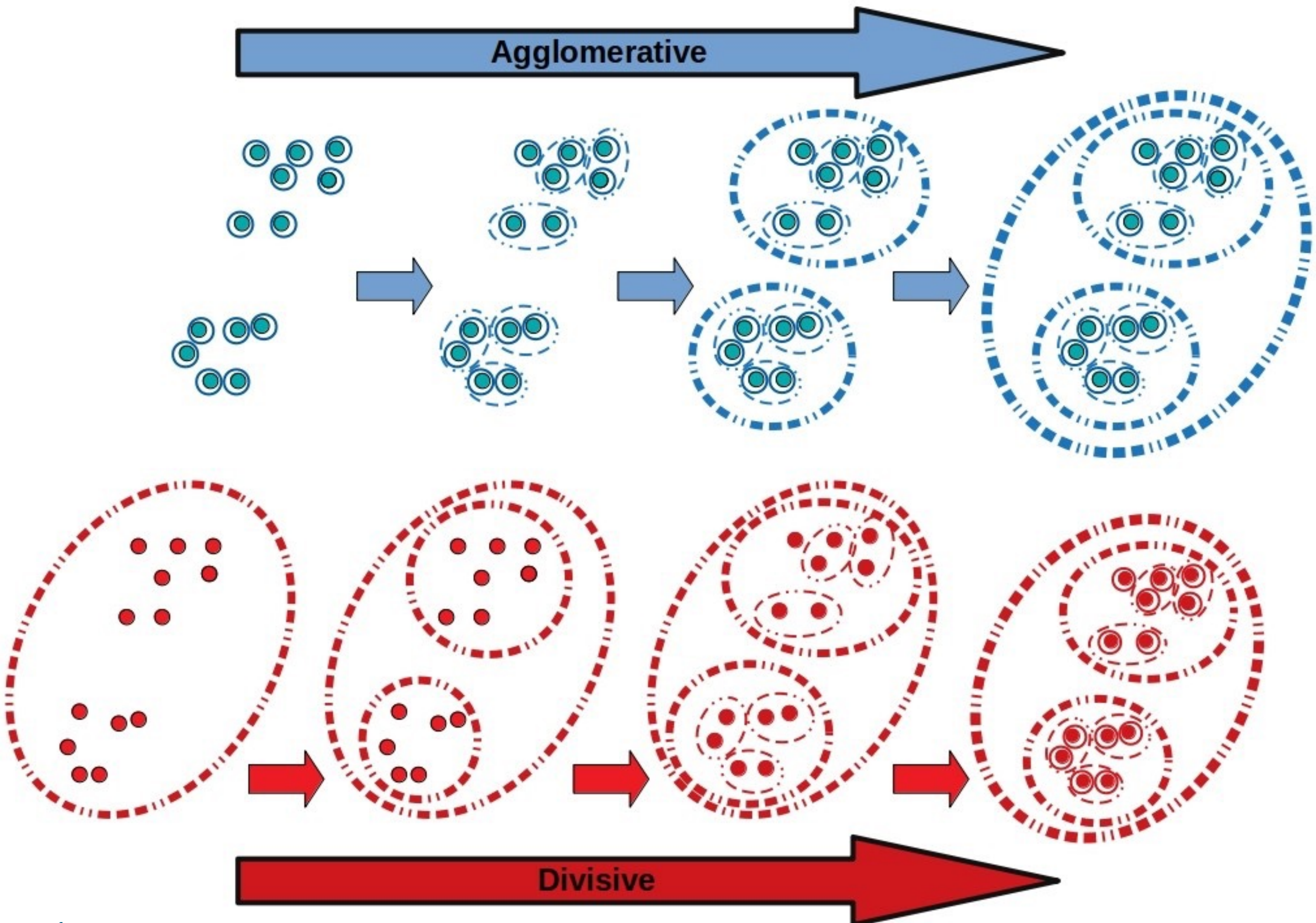
Clustering Algorithm (1/3) – K-means



Credict: Pratik Thorat

Discovering Groups – **Overview (4/5)**

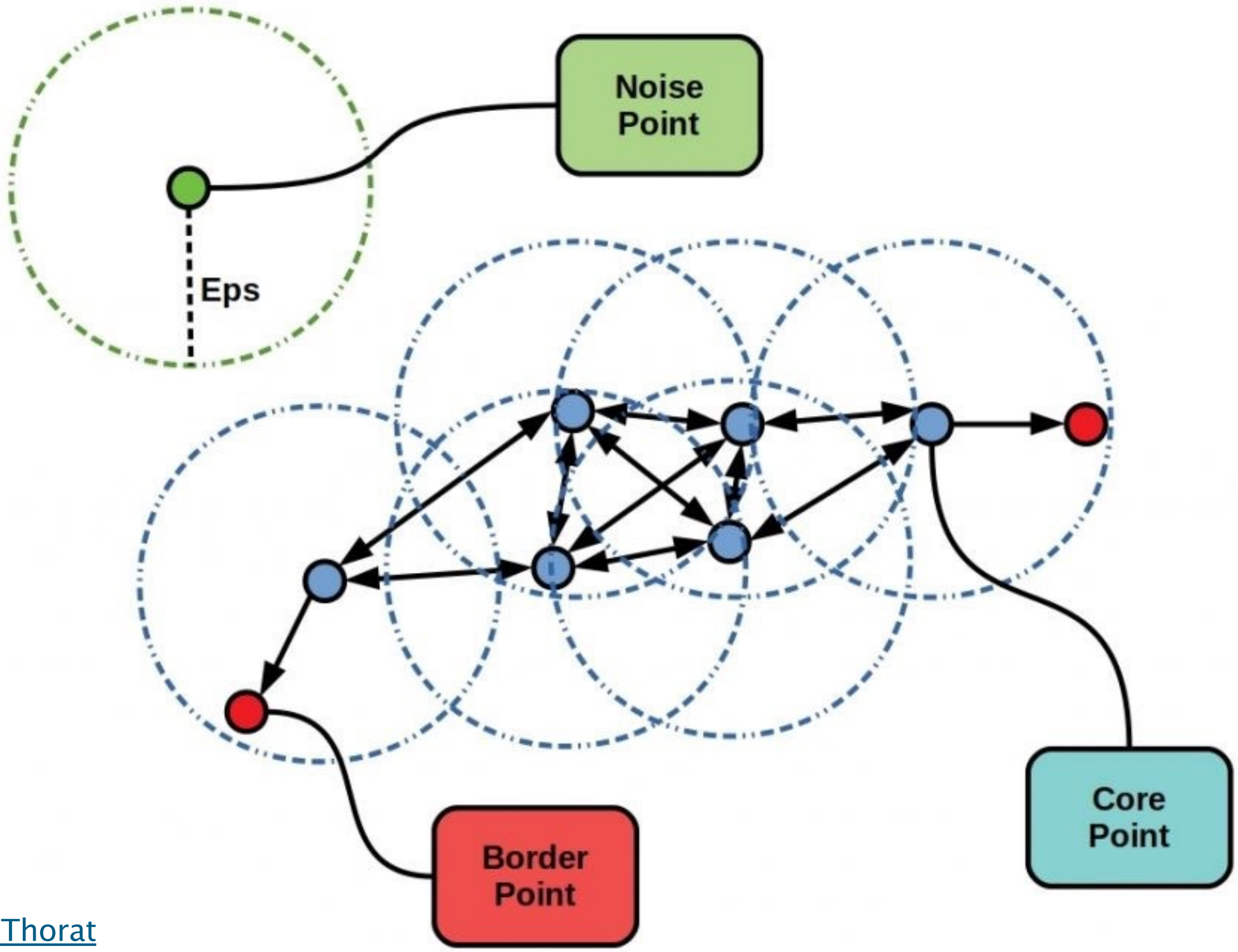
Clustering Algorithm (2/3) – Agglomerative (Divisive not learned in this lecture)



Credit: Pratik Thorat

Discovering Groups – **Overview (5/5)**

Clustering Algorithm (2/3) – DBSCAN



Credit: Pratik Thorat

Discovering Groups – **Learning Outcomes**

- **LO1**: Comprehend the key ideas and the essential mathematical formulations employed in clustering methods (exam).
  - E.g., how is sum of squared error (SSE) defined?
  - E.g., understand the pros and cons of the learned algorithms
- **LO2**: Compute the fundamental stages of learned clustering approaches (exam).
  - E.g., given a dataset and a distance metric, be prepared to follow the selected clustering algorithm to cluster the instances in the dataset
- **LO3**: Implement and evaluate the learned clustering algorithms using Python (course work)

Assessment hints: Multi-choice Questions (single answer: concepts, calculation etc)

- *Textbook Exercises: textbooks (Programming + Mining)*
- *Other Exercises: https://www-users.cse.umn.edu/~kumar001/dmbook/sol.pdf*
- *ChatGPT or other AI-based techs*

Discovering Groups – K-means

- Given: data set $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, number of clusters K
- Goal: find cluster centers $\{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K\}$ so that

$$J = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\mathbf{x}_n - \boldsymbol{\mu}_k\|^2$$

  is minimal, where $r_{nk} = 1$ if $\mathbf{x}_n$ is assigned to $\boldsymbol{\mu}_k$
- Idea: compute r_{nk} and $\boldsymbol{\mu}_k$ iteratively

Otherwise, $r_{n,k} = 0$

The used objective function is **Sum of Squared Error (SSE)**

Source: D. Cremers

Discovering Groups – K-means

University of Southampton

Algorithm 1: K Means clustering

Data: X, K

initialise K centroids;

while *positions of centroids change* **do**

  for *each data point* **do**

    assign to nearest centroid

  end

  for *each centroid* **do**

    move to average of assigned data points

  end

end

return centroids, assignments;

A special case of Expectation Maximisation - why?

Credit: Jo Grundy

Discovering Groups – K-means

Algorithm 2: K Means clustering

Data: X, K

initialise K centroids;

Lecture: Maximum Likelihood Estimation

```
while positions of centroids change do
    for each data point do
        assign to nearest centroid ;          // Expectation of
          associations
    end
    for each centroid do
        move to average of assigned data points ;
          // Maximisation of likelihood
    end
end
return centroids, assignments;
```

E-step: Estimate the posterior probabilities…

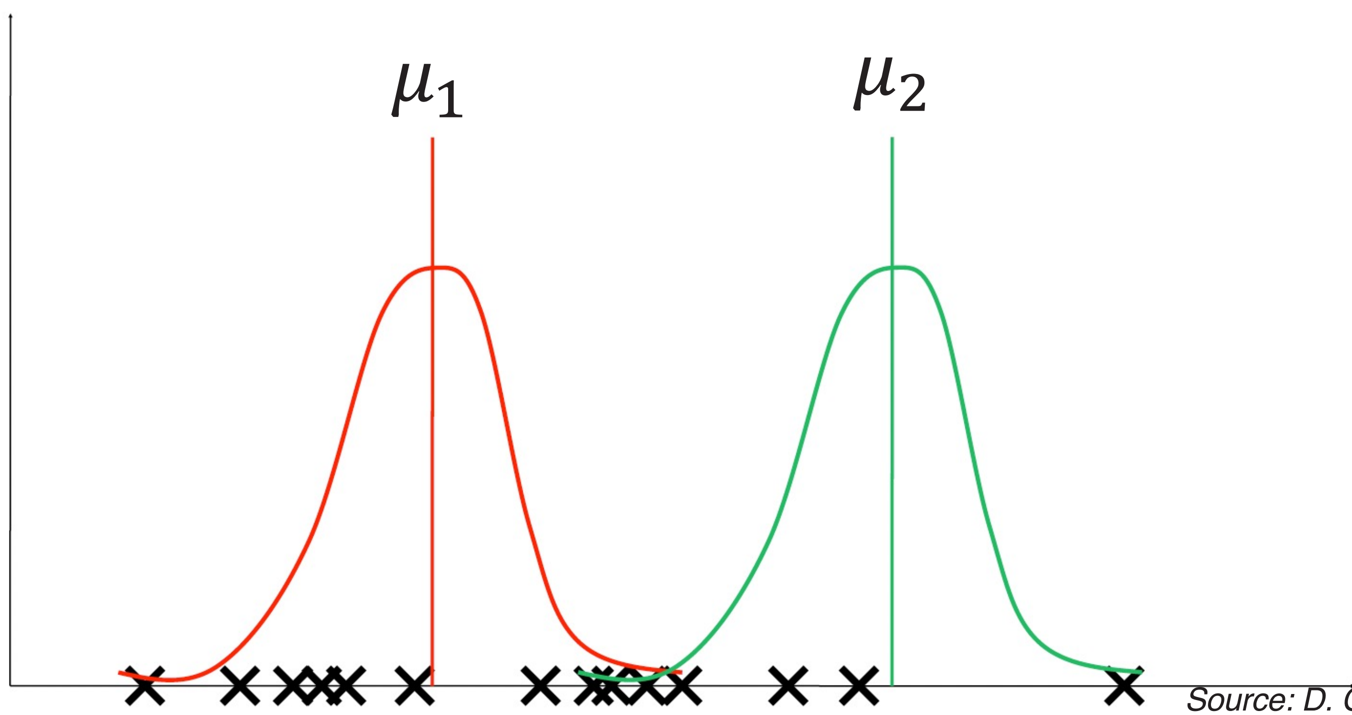
M-step: Estimate new parameters

Assumes spherical clusters

Credit: Jo Grundy

Discovering Groups – K-means
1D Example

Initialize cluster means: $\{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_K\}$

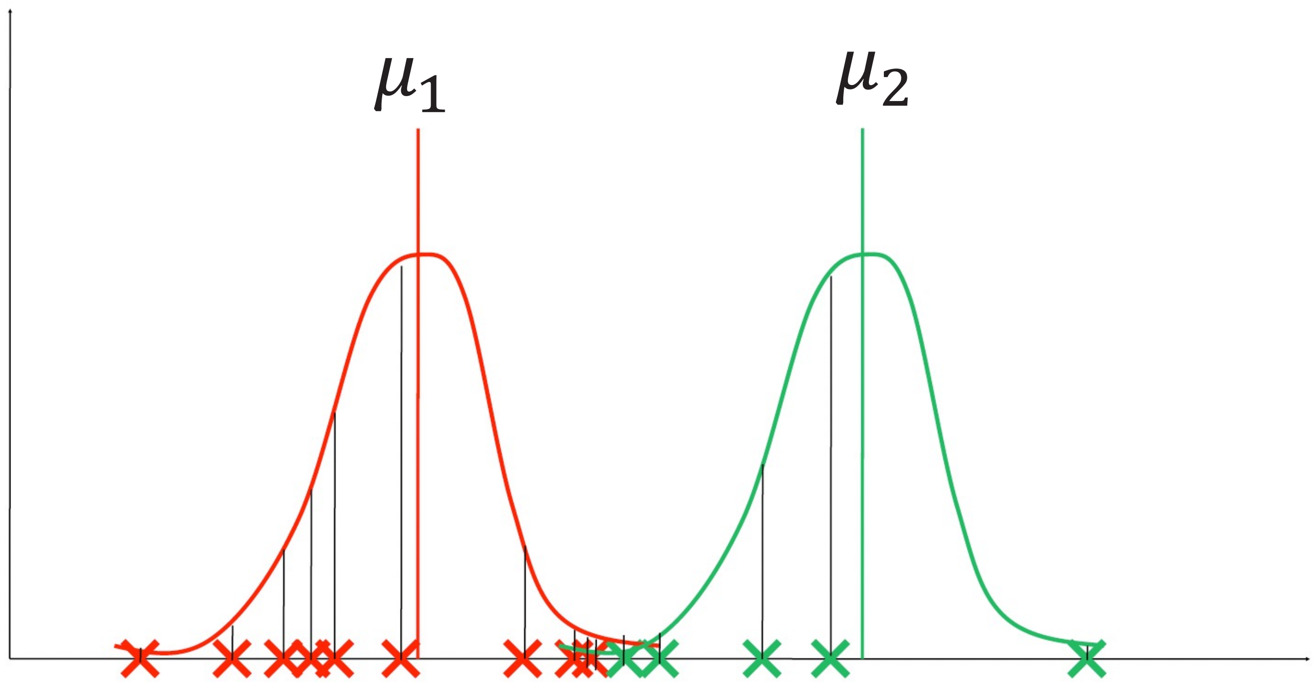


Source: D. Cremers

Discovering Groups – K-means
1D Example

Find optimal assignments:

$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_j \|\mathbf{x}_n - \boldsymbol{\mu}_j\| \\ 0 & \text{otherwise} \end{cases}$$



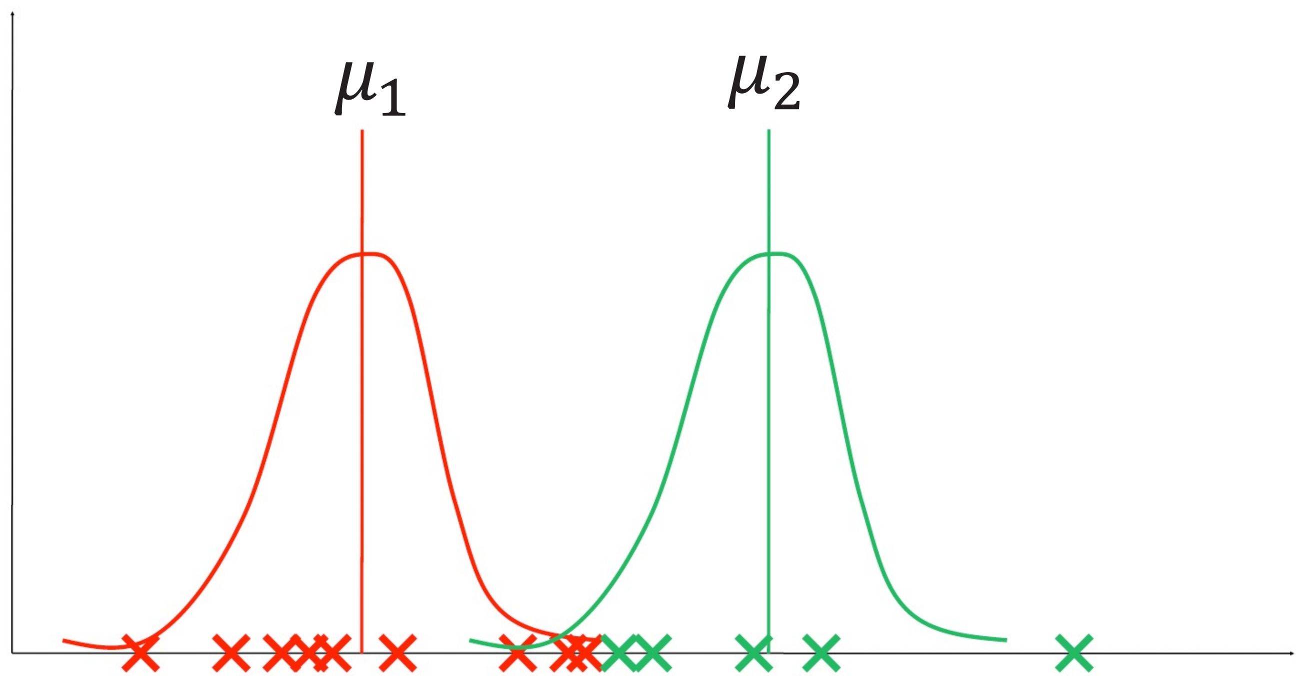
Source: D. Cremers

Discovering Groups – K-means
1D Example

Find new optimal means:

$$\frac{\partial J}{\partial \boldsymbol{\mu}_k} = 2\sum_{n=1}^{N} r_{nk}(\mathbf{x}_n - \boldsymbol{\mu}_k) \stackrel{!}{=} 0$$

$$\Rightarrow \boldsymbol{\mu}_k = \frac{\sum_{n=1}^{N} r_{nk}\mathbf{x}_n}{\sum_{n=1}^{N} r_{nk}}$$

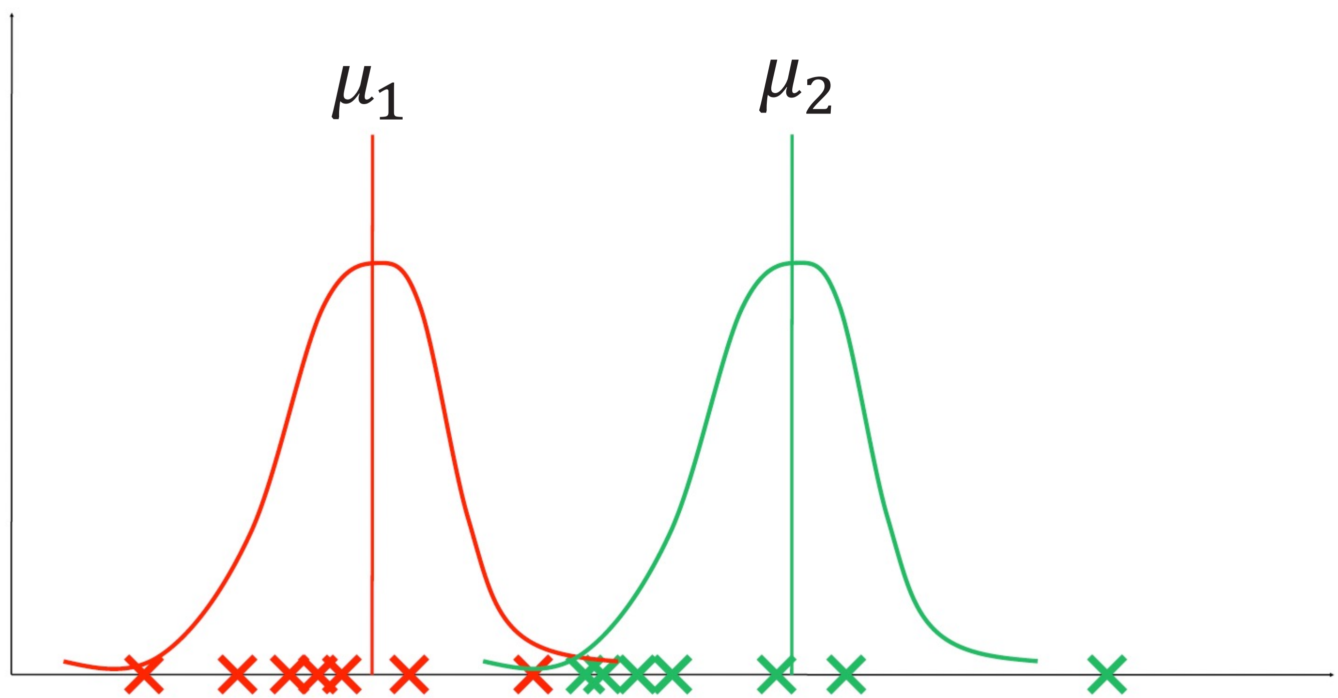


Source: D. Cremers

Discovering Groups – K-means
1D Example

Find new optimal assignments:

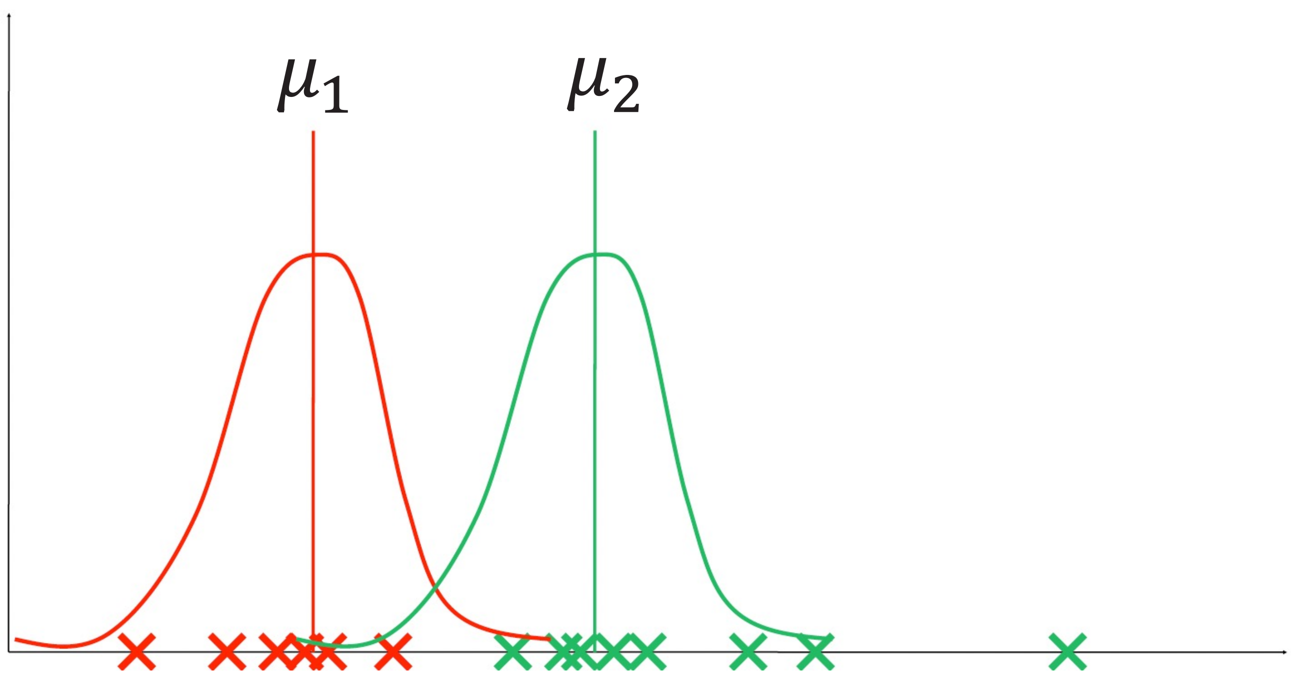
$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_j \|\mathbf{x}_n - \boldsymbol{\mu}_j\| \\ 0 & \text{otherwise} \end{cases}$$



Source: D. Cremers

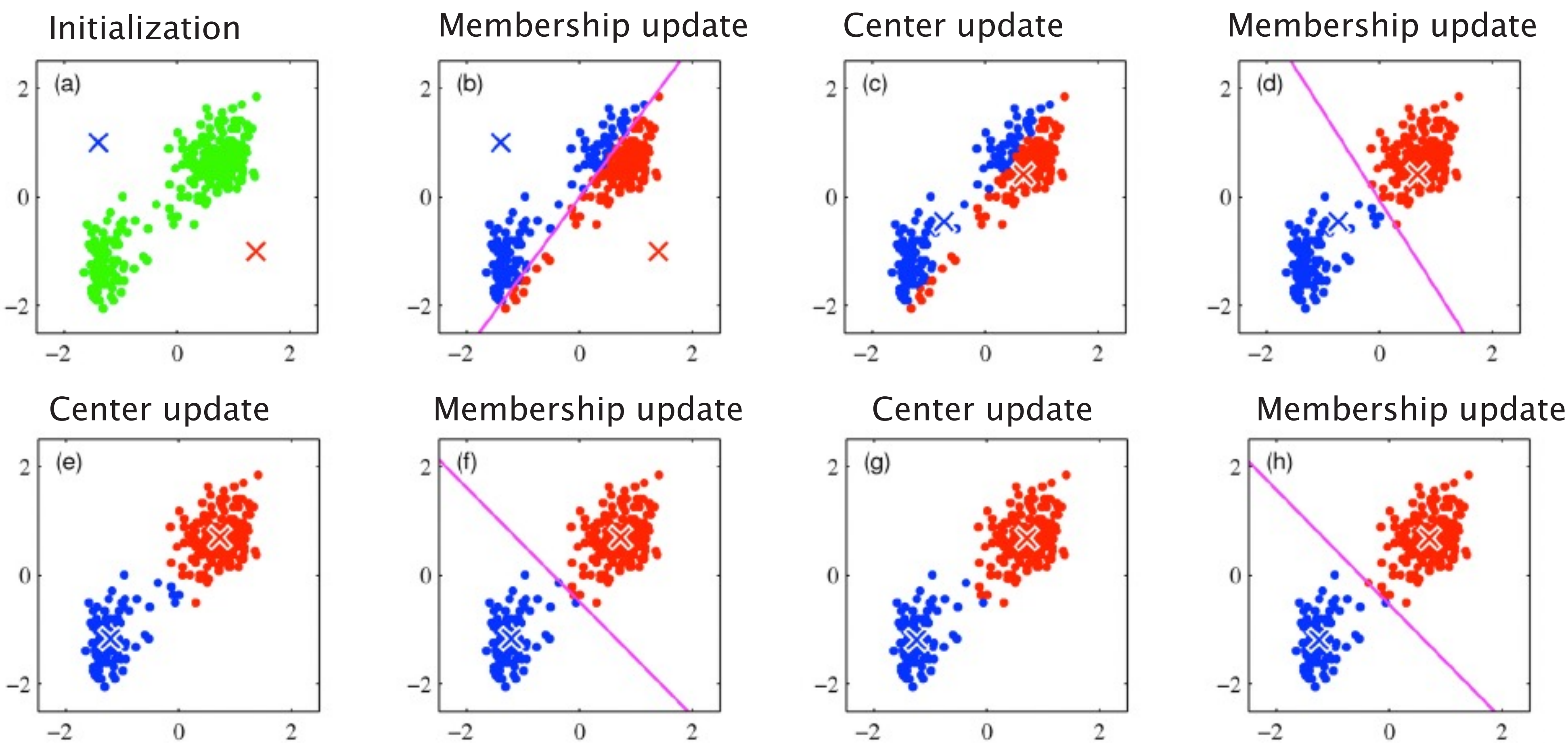
Discovering Groups – K-means
1D Example

Iterate these steps until means and assignments do not change any more



Source: D. Cremers

Discovering Groups – K-means
2D Example



Source: D. Cremers

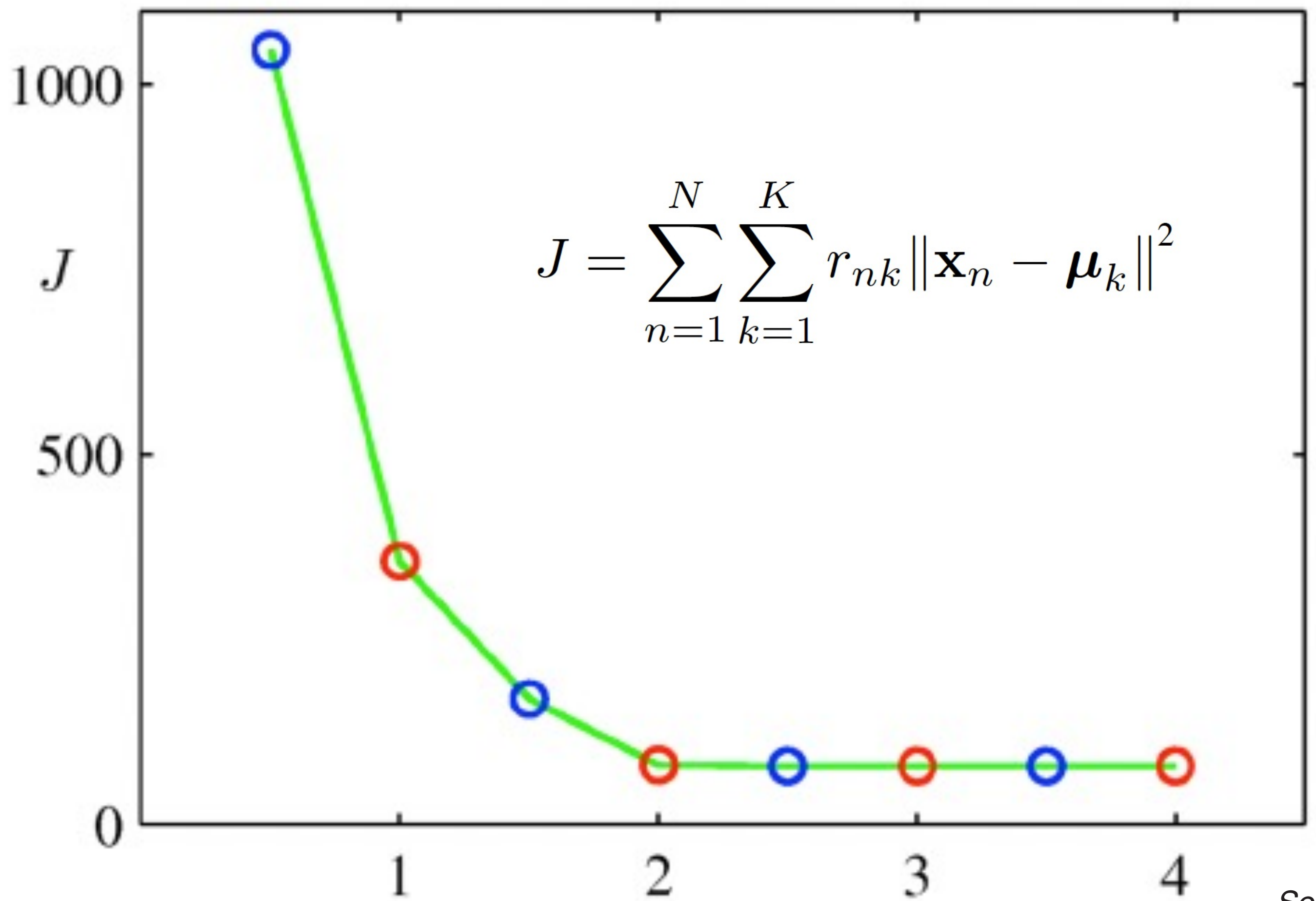
- Real data set
- Radom initialization

- Magenta line is ‘decision boundary’

Source: D. Cremers

Discovering Groups – K-means

Sum of Squared Error (SSE) Curve



Source: D. Cremers

- After every step the cost function J is minimized
- Blue steps: update assignments
- Red steps: update means
- Convergence after 4 rounds

Discovering Groups – K-means

K-means can quickly and cheaply cluster data.

Problems?

- need to specify the cluster number k
- depends on good initial centroid guesses
- may converge on local minimum
- assumes spherical data (or ellipsoid-shaped clusters, or at best convex clusters)

Gaussian Mixture models (GMM) can work better, using a generalization of K-means (assuming each cluster is Gaussian), not discussed in this lecture.

Discovering Groups – DBSCAN

- **Idea**: uses the **local density** of points to determine the clusters, rather than using only the distance between points

$$N_\epsilon(\mathbf{x}) = B_d(\mathbf{x}, \epsilon) = \{\mathbf{y} \mid \delta(\mathbf{x}, \mathbf{y}) \leq \epsilon\}$$

where $\delta(\mathbf{x}, \mathbf{y})$ represents the distance between **x** and **y, ε** indicates **Max radius,** and **x** is a **core point** if $|N_\epsilon(\mathbf{x})| \geq minpts$, where *minpts* is a **Min number** that is user-defined local density or frequency threshold

- **x** belongs to a density-based cluster when

$$|N_\epsilon(\mathbf{x})| \geq minpts \text{ or } \mathbf{x} \in N_\epsilon(\mathbf{z})$$

where **z** is another data point, *minpts* is a **Min number** that is user-defined local density or frequency threshold

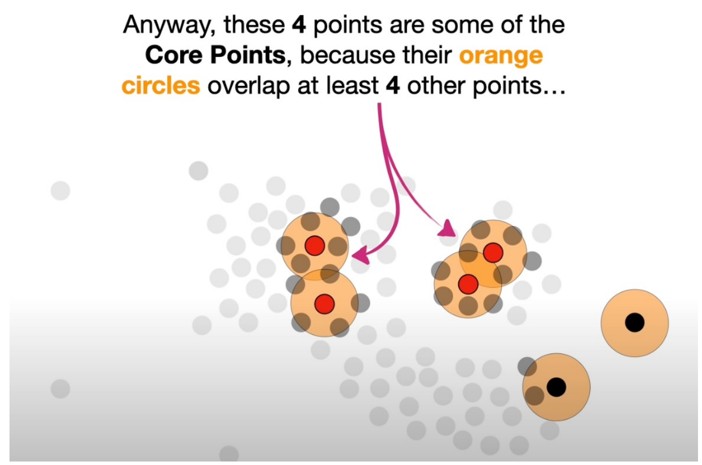
Max radius is the limit on which to look for neighbours
Min number is the lower limit on what can be in a cluster

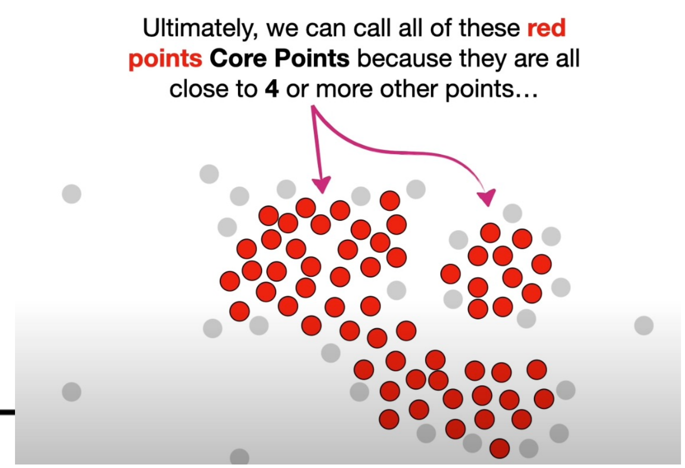
Discovering Groups – DBSCAN

Algorithm 3: DBSCAN

```
Data: X, eps, min_pts
initialse labels list as zeros, count list, core list;
Find neighbours for each point, Find core points;
class = 1;
for each core point p do
    add neighbours(p) to queue;
    while queue not empty do
        neighbours = next(queue);
        for q in neighbours do
            set label(q = class;
            if label(q) is 'core' then
                add neighbours(q) to queue
            end
        end
    end
    class = class + 1
end
return labels;
```

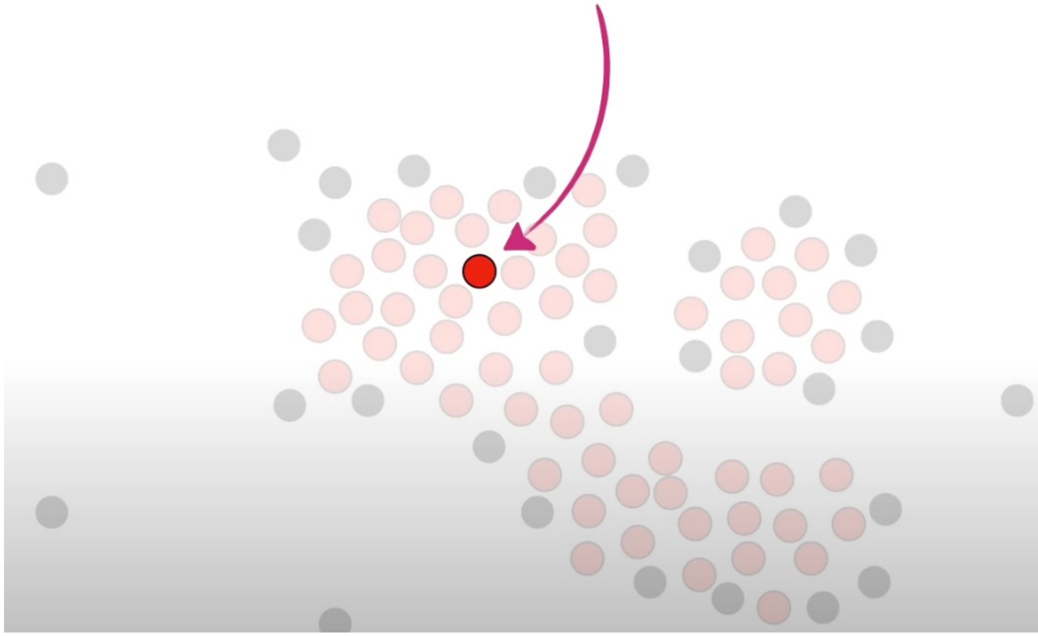
$$|N_\epsilon(\mathbf{x})| \geq minpts$$



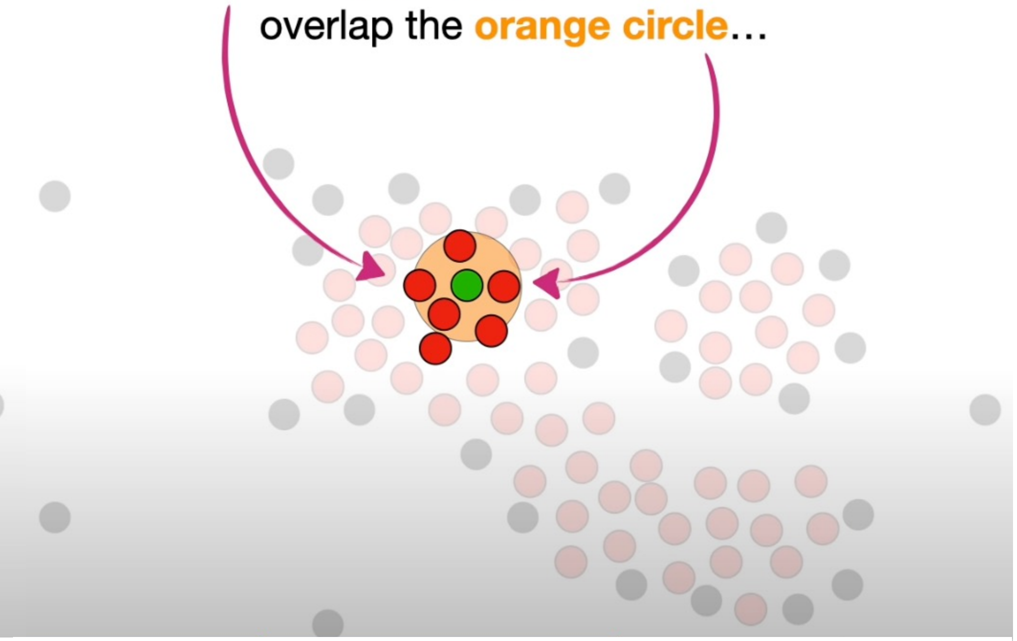


Credit: Jo Grundy

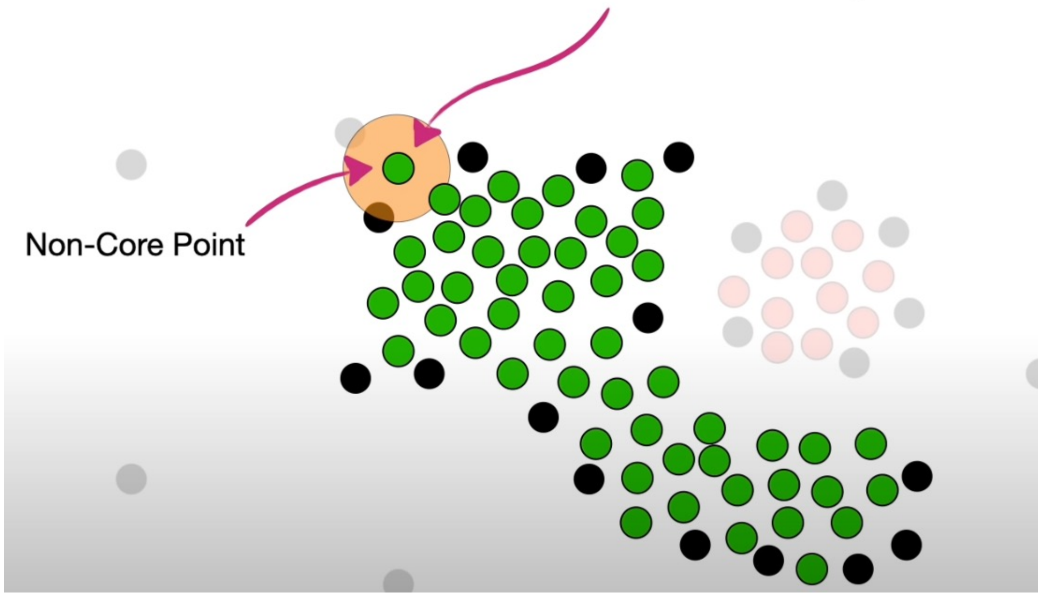
Now we randomly pick a **Core Point**…



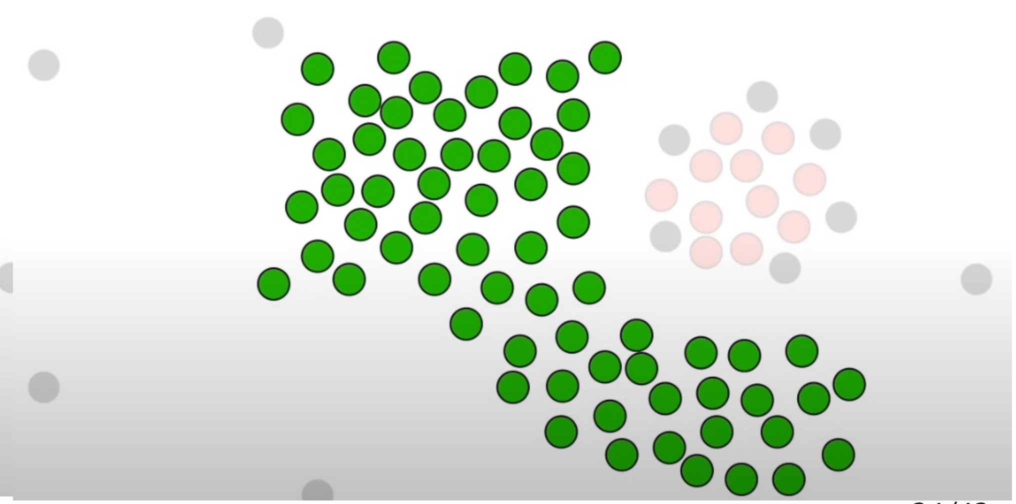
Next, the **Core Points** that are close to the **first cluster**, meaning they overlap the **orange circle**…



However, because this is not a **Core Point**, we do not use it to extend the **first cluster** any further.



And now we are done creating the **first cluster**.



https://www.youtube.com/watch?v=RDZUdRSDOok

Discovering Groups – DBSCAN

DBSCAN works well on any shape of data and is robust to outliers.

Problems?

- can struggle in high dimensions
- needs a distance parameter
- same parameter may not work for different cluster density
- need also minimum number specified

Credit: Jo Grundy

Discovering Groups – Hierarchical Clustering

Creates a **binary tree** that **recursively groups** pairs of similar items or clusters

Can be:

- Agglomerative (bottom up)
- Divisive (top down)

We will look at Agglomerative clustering. Needs a distance measure.

Credit: Jo Grundy

Discovering Groups – Hierarchical Clustering

Algorithm 4: Hierarchical Agglomerative Clustering

Data: N data points with feature vectors X_i $i = 1 \ldots N$

$numClusters = N$;

while $numClusters > 1$ **do**

  cluster1, cluster2 = FindClosestClusters();

  merge(cluster1, cluster2);

end

The distance between the clusters is evaluated using a linkage criterion.

If each merge is recorded, a binary tree structure linking the clusters can be formed.

This gives a **dendrogram**

Credit: Jo Grundy

Discovering Groups – Hierarchical Clustering

Linkage criterion: A measure of dissimilarity between clusters

Centroid Based:

- Dissimilarity is equal to distance between centroids
- Needs numeric feature vectors

Distance-Based:

- Dissimilarity is a function of distance between items in clusters
- Only needs precomputed measure of similarity between items

We could compute a distance matrix between points

Credit: Jo Grundy

Discovering Groups – Hierarchical Clustering

Centroid based linkage:

- WPGMC: Weighted Pair Group Method with Centroids. When two clusters are combined into a new cluster, the average of the two centroids is the new centroid

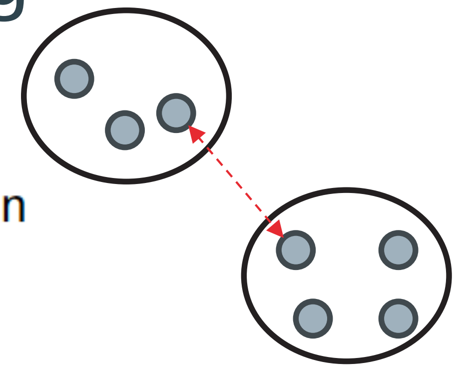
- UPGMC: Unweighted Pair Group Method with Centroids. When two clusters are combined into a new cluster, the new centroid is recalculated based on the positions of the items

Credit: Jo Grundy

Discovering Groups – Hierarchical Clustering

Distance based linkage:

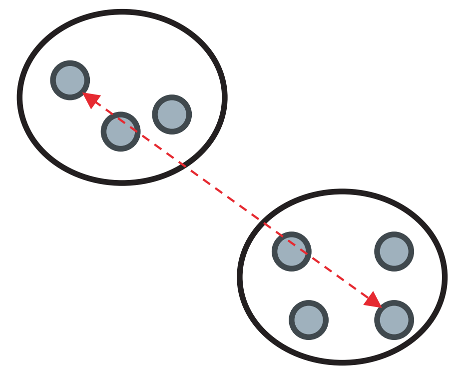
- **Minimum**, or **single-linkage clustering** Distance between two closest members



$$\min d(a,b) : a \in A, b \in B$$

Produces long, thin clusters

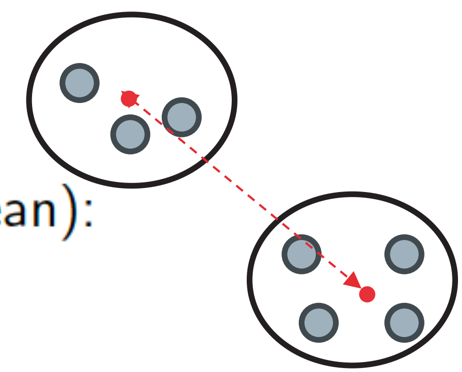
- **Maximum**, or **complete-linkage clustering** Distance between two most distant members



$$\max d(a,b) : a \in A, b \in B$$

Finds compact clusters, approximately equal diameter

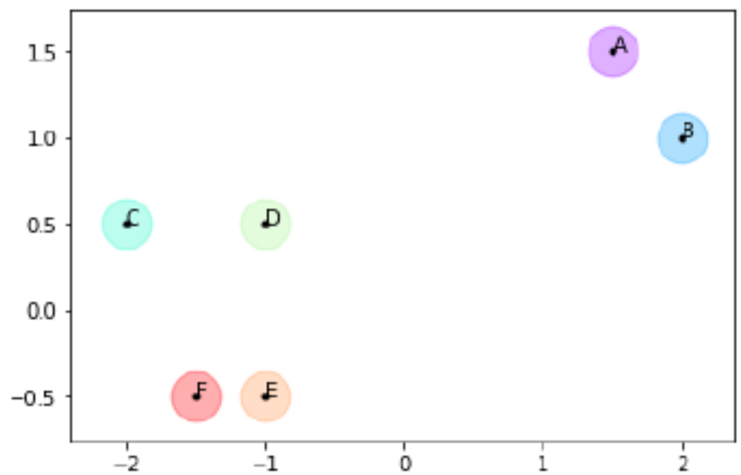
- **Mean** or **Average Linkage Clustering** (**UPGMA**: Unweighted Pairwise Group Method with Arithmetic Mean):



$$\frac{1}{|A||B|}\sum_{a\in A}\sum_{b\in B} d(a,b)$$

Credit: Jo Grundy

Discovering Groups – Agglomerative Clustering



With sample data:

$$X = \begin{bmatrix} 1.5 & 1.5 \\ 2.0 & 1.0 \\ 2.0 & 0.5 \\ -1.0 & 0.5 \\ -1.5 & -0.5 \\ -1 & 0.5 \end{bmatrix}$$

Distance matrix: *demo distances, not ground truth*

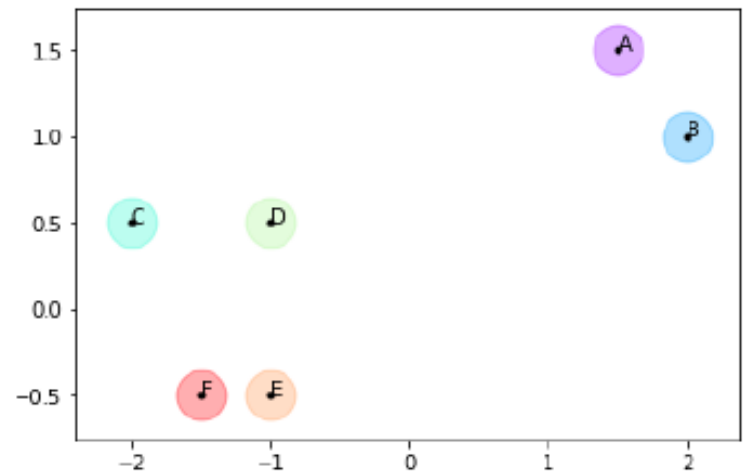
	A	B	C	D	E	F
A	0	0.7	2.7	1.8	...	
B	0.7	0	...			
C	2.7		0	...		
D	1.8			0	...	
E	$\vdots$				0	

Credit: Jo Grundy

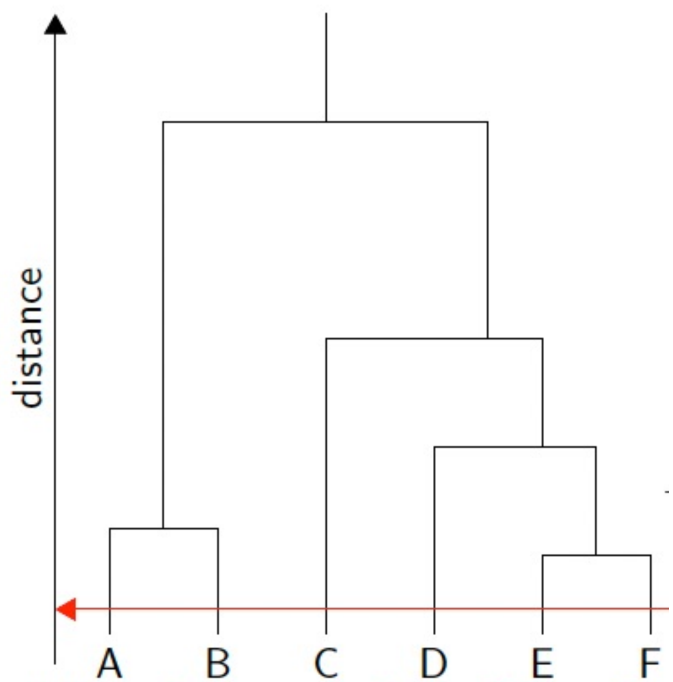
Discovering Groups – Agglomerative Clustering

Algorithm 4: Hierarchical Agglomerative Clustering

Data: N data points with feature vectors X_i $i = 1 \dots N$
numClusters $= N$;
while *numClusters* > 1 **do**
  cluster1, cluster2 = FindClosestClusters();
  merge(cluster1, cluster2);
end

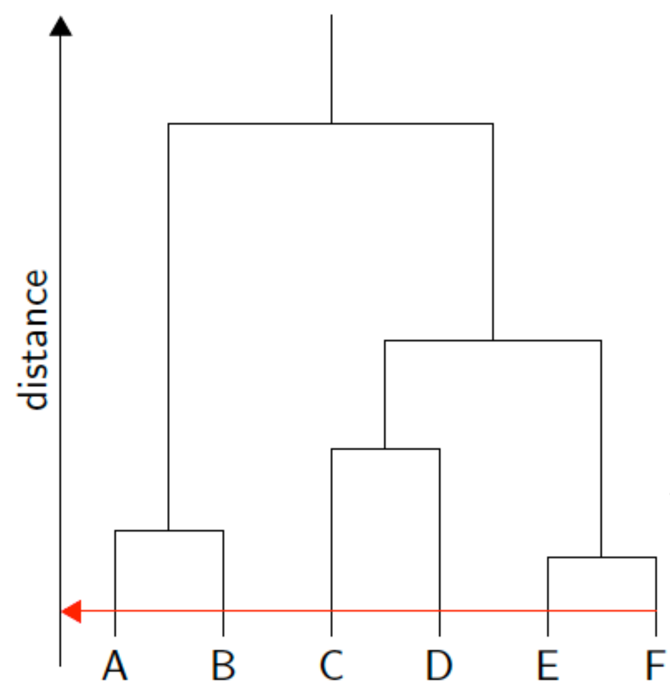


Dendrograms



Using minimum or single-linkage clus

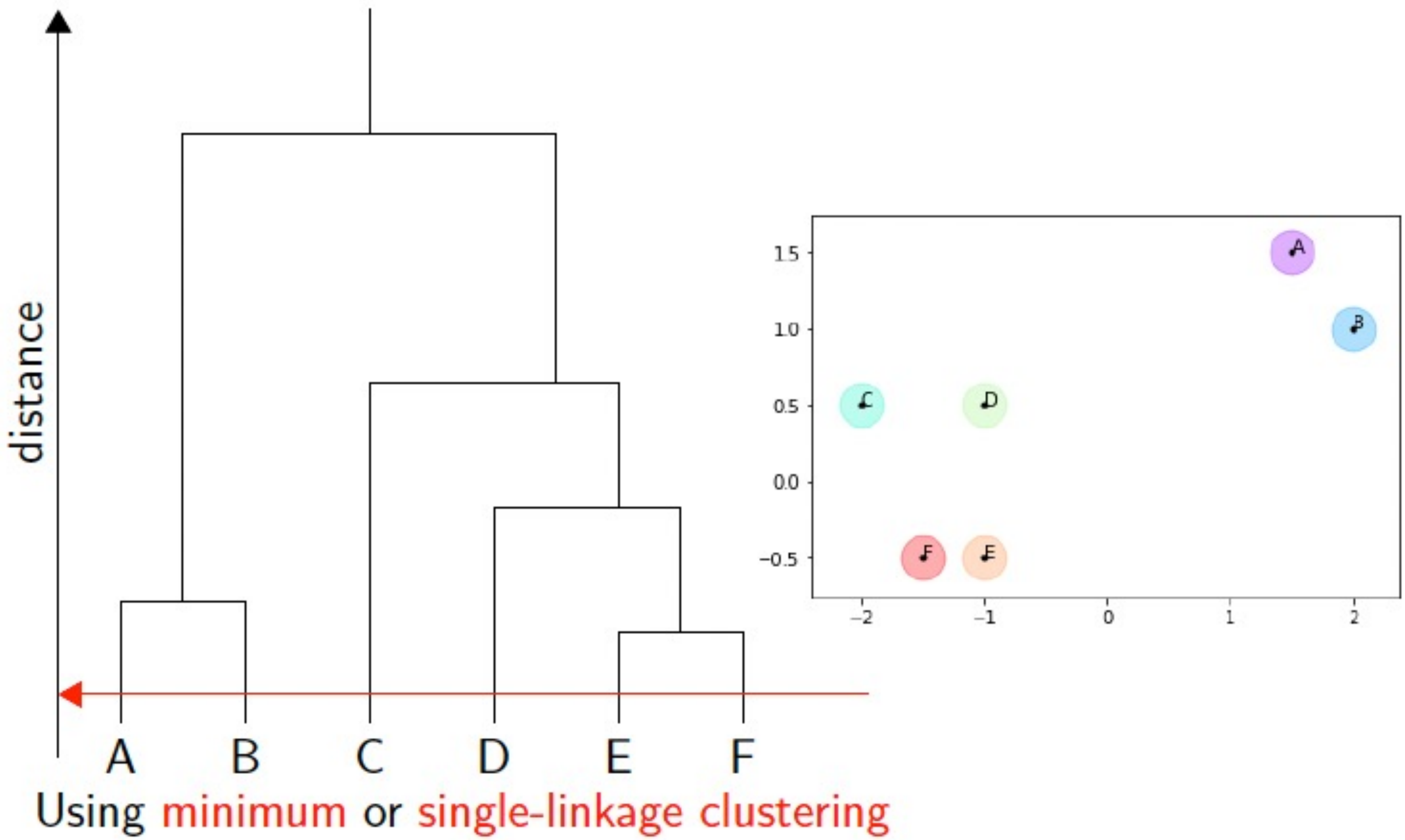
d_{min}



Using maximum or complete-linkage

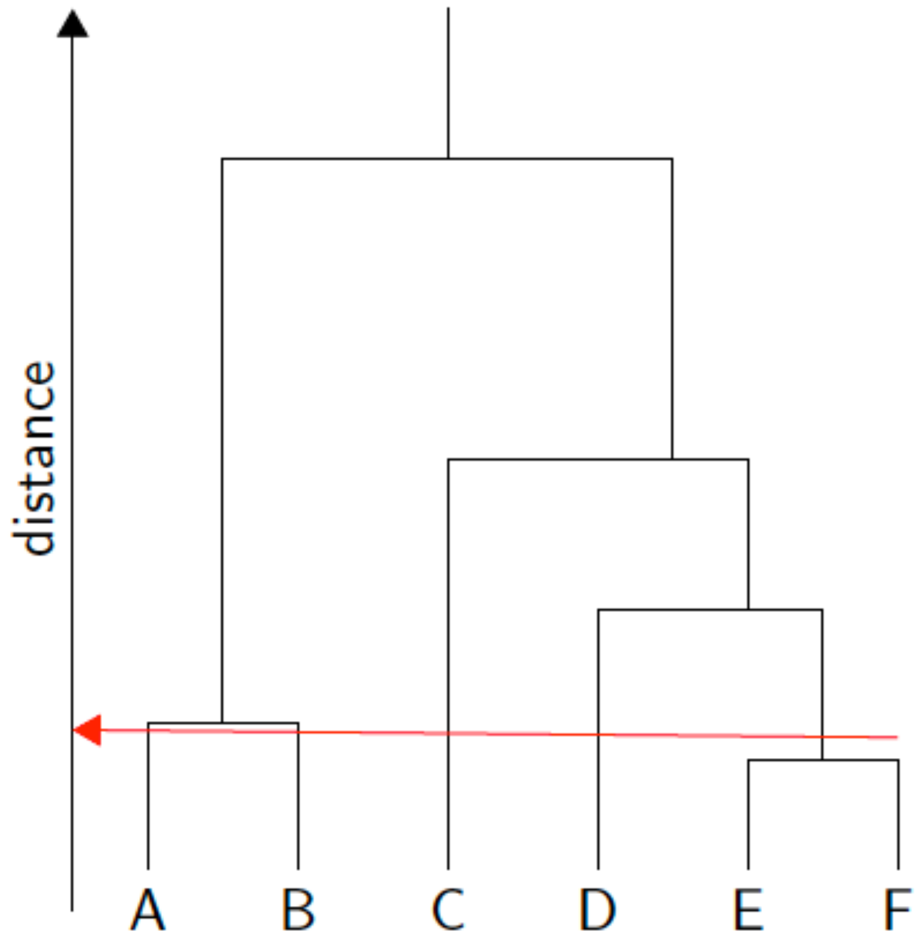
d_{max}

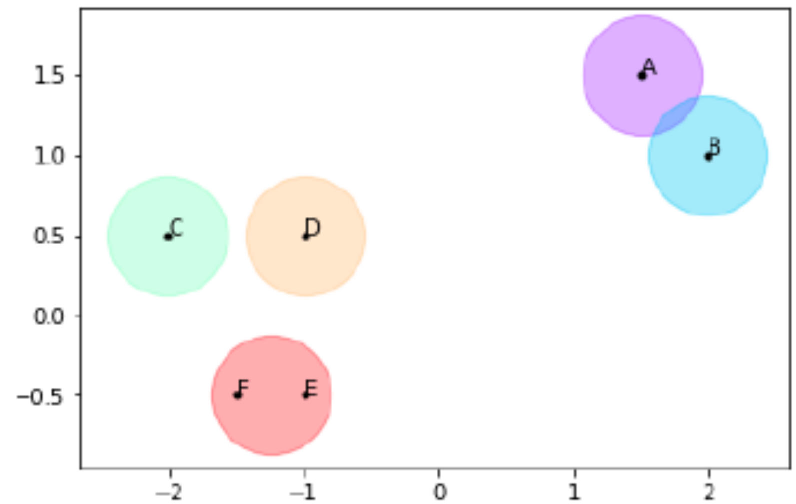
Discovering Groups – Agglomerative Clustering



Credit: Jo Grundy

Discovering Groups – Agglomerative Clustering

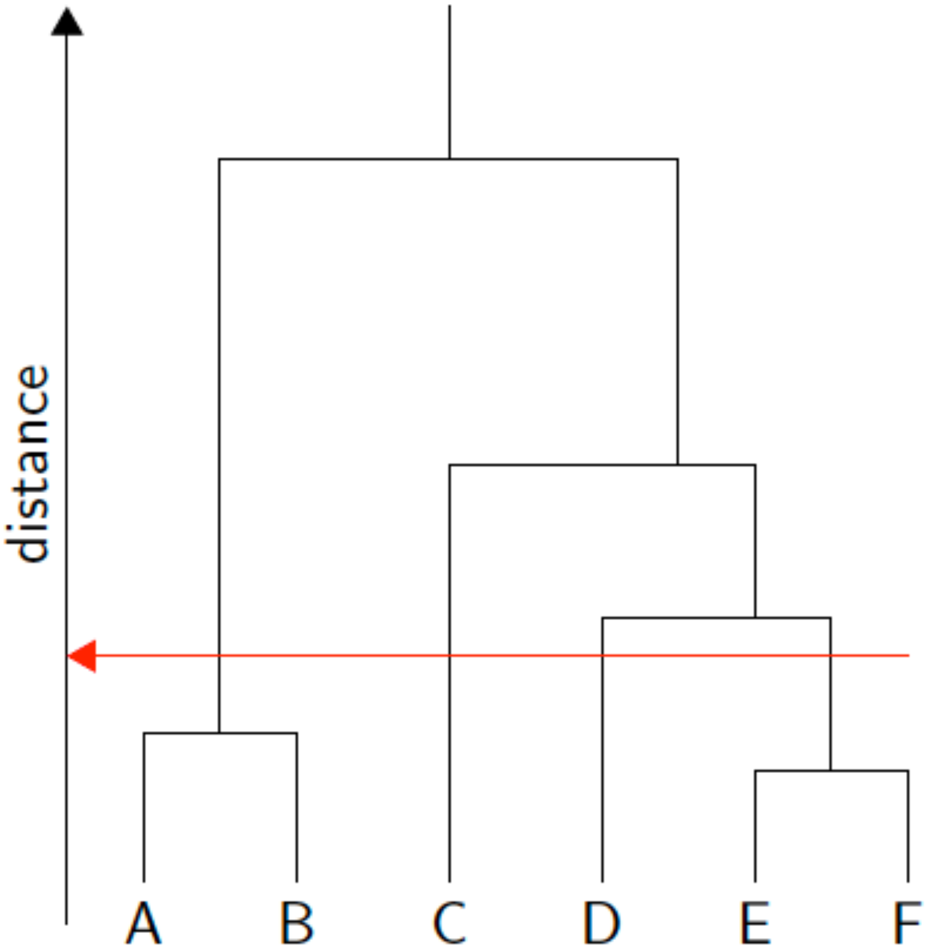


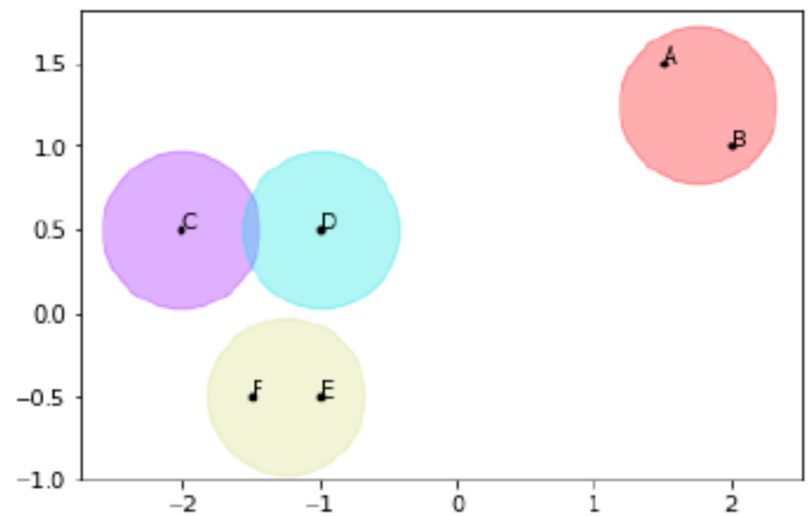


Using minimum or single-linkage clustering

Credit: Jo Grundy

Discovering Groups – Agglomerative Clustering



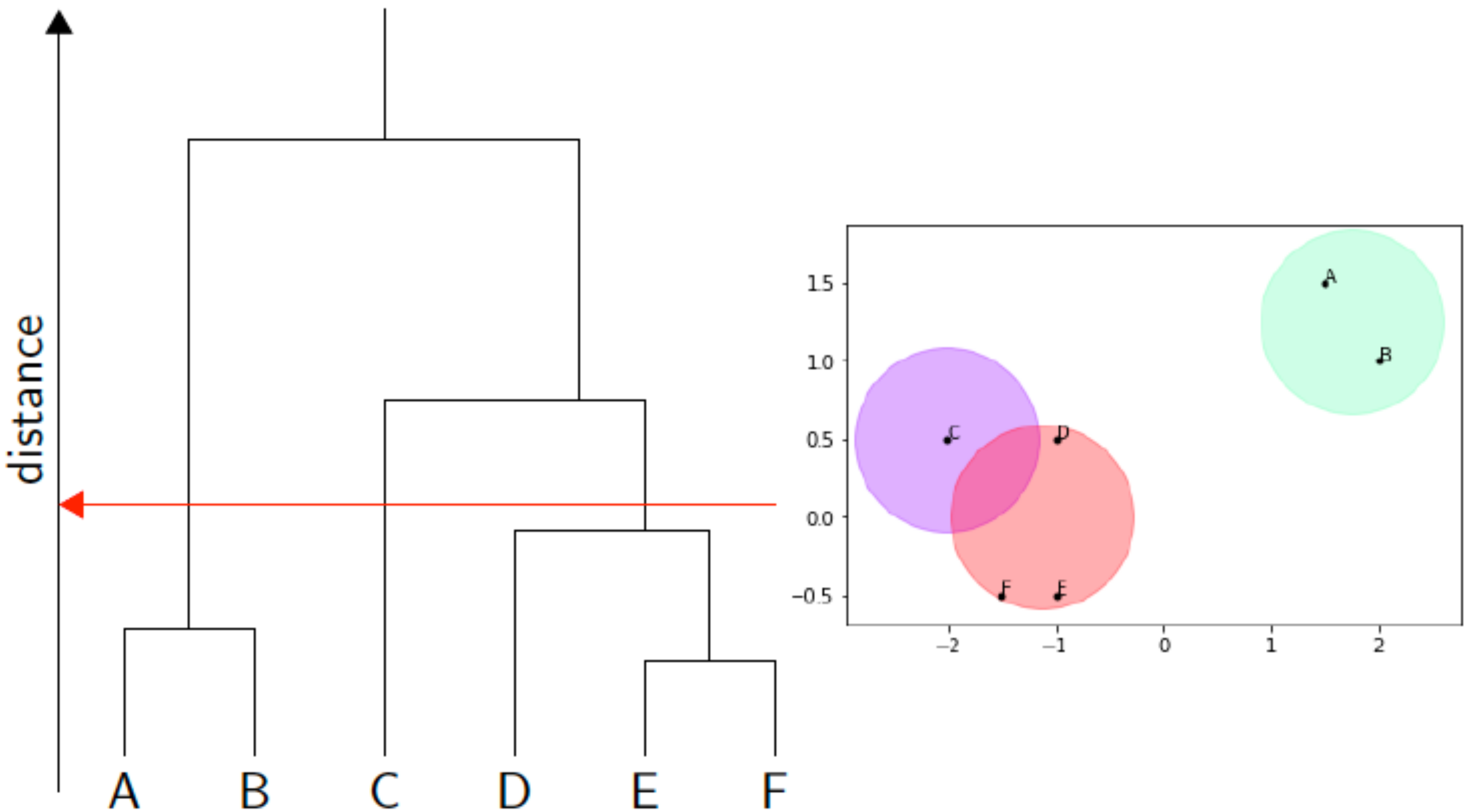


Using minimum or single-linkage clustering

Credit: Jo Grundy

University of Southampton

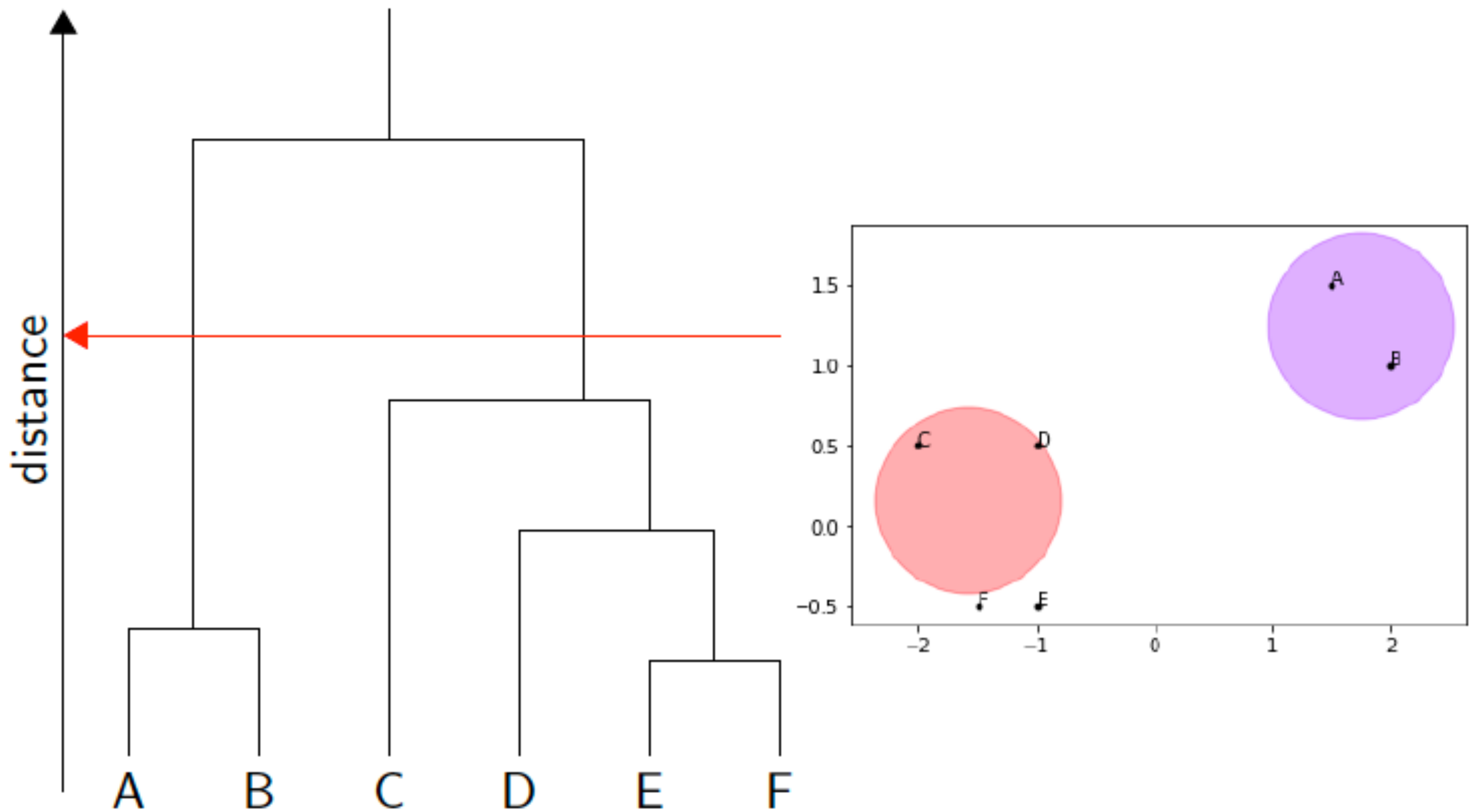
Discovering Groups – Agglomerative Clustering



Using minimum or single-linkage clustering

Credit: Jo Grundy

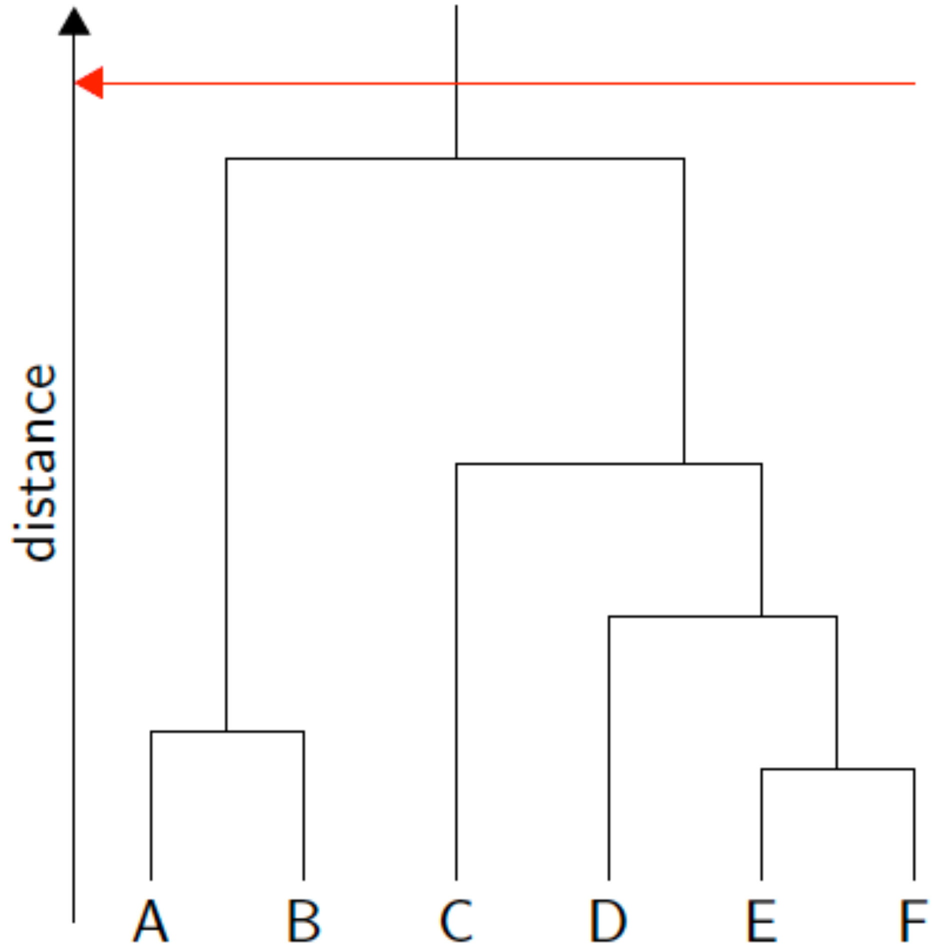
Discovering Groups – Agglomerative Clustering

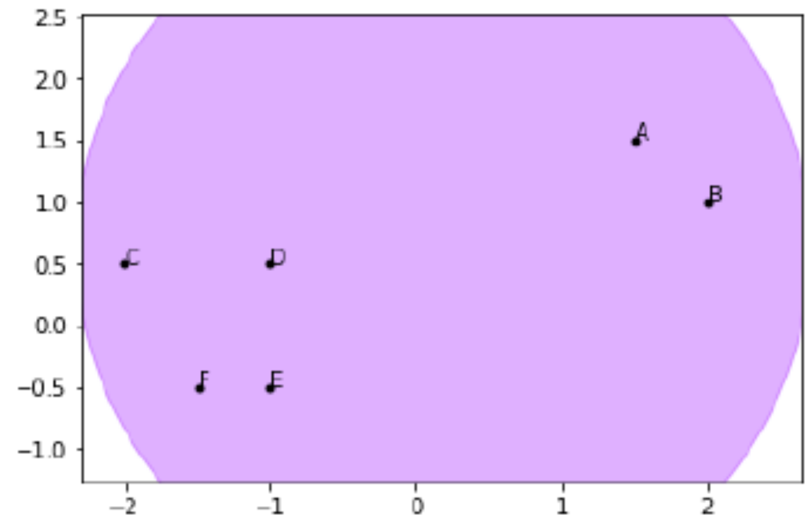


Using minimum or single-linkage clustering

Credit: Jo Grundy

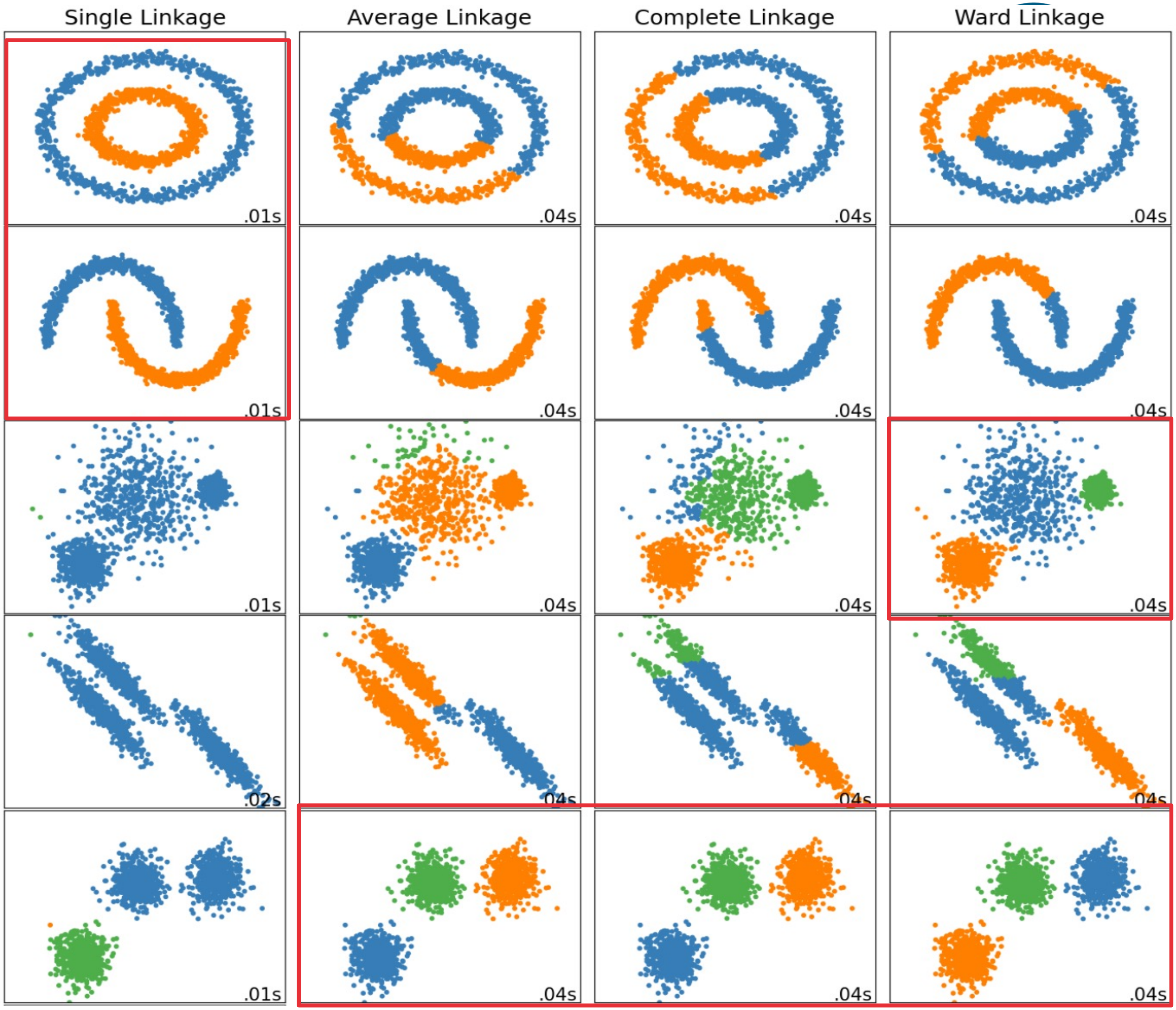
Discovering Groups – Agglomerative Clustering





Using minimum or single-linkage clustering

Credit: Jo Grundy



Source: https://scikit-learn.org/stable/auto_examples/cluster/plot_linkage_comparison.html#sphx-glr-auto-examples-cluster-plot-linkage-comparison-py

Discovering Groups – Hierarchical Clustering

Pros:

- No need to pre-specify cluster numbers; cut the dendrogram at the desired level for the clusters.
- Dendrograms easily summarize data into a hierarchy, facilitating cluster examination and interpretation.

Cons:

- Needs a threshold to determine the number of clusters
- Non-trivial to select the best linkage method

Discovering Groups – Summary

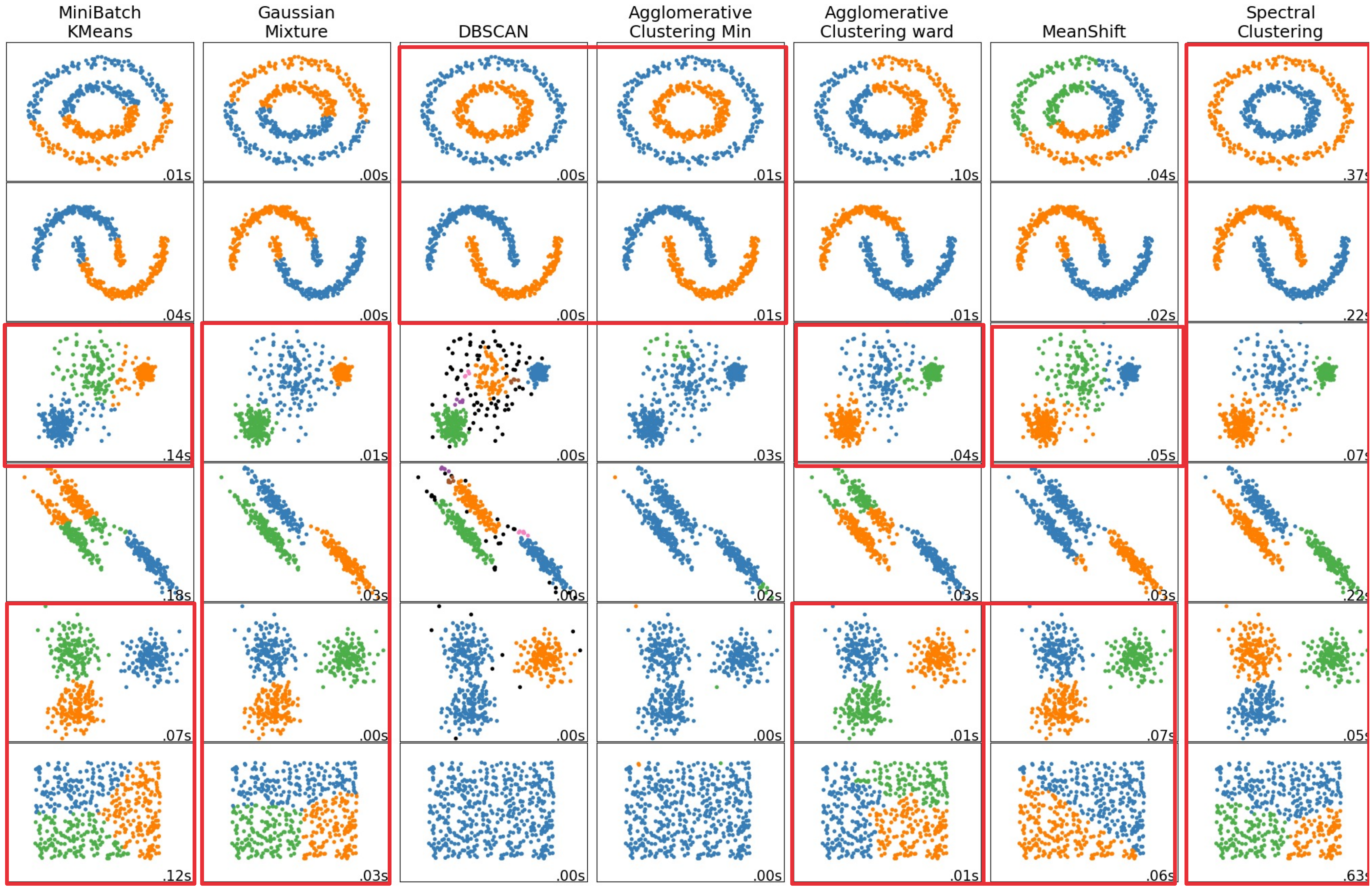
Clustering is a key way to understand your data.

There are many different approaches

- K Means - Need to chose K
- DBSCAN - need to choose min points and radius
- Hierarchical Agglomerative Clustering - needs a threshold or number of clusters

They are a very good way to start exploring a dataset

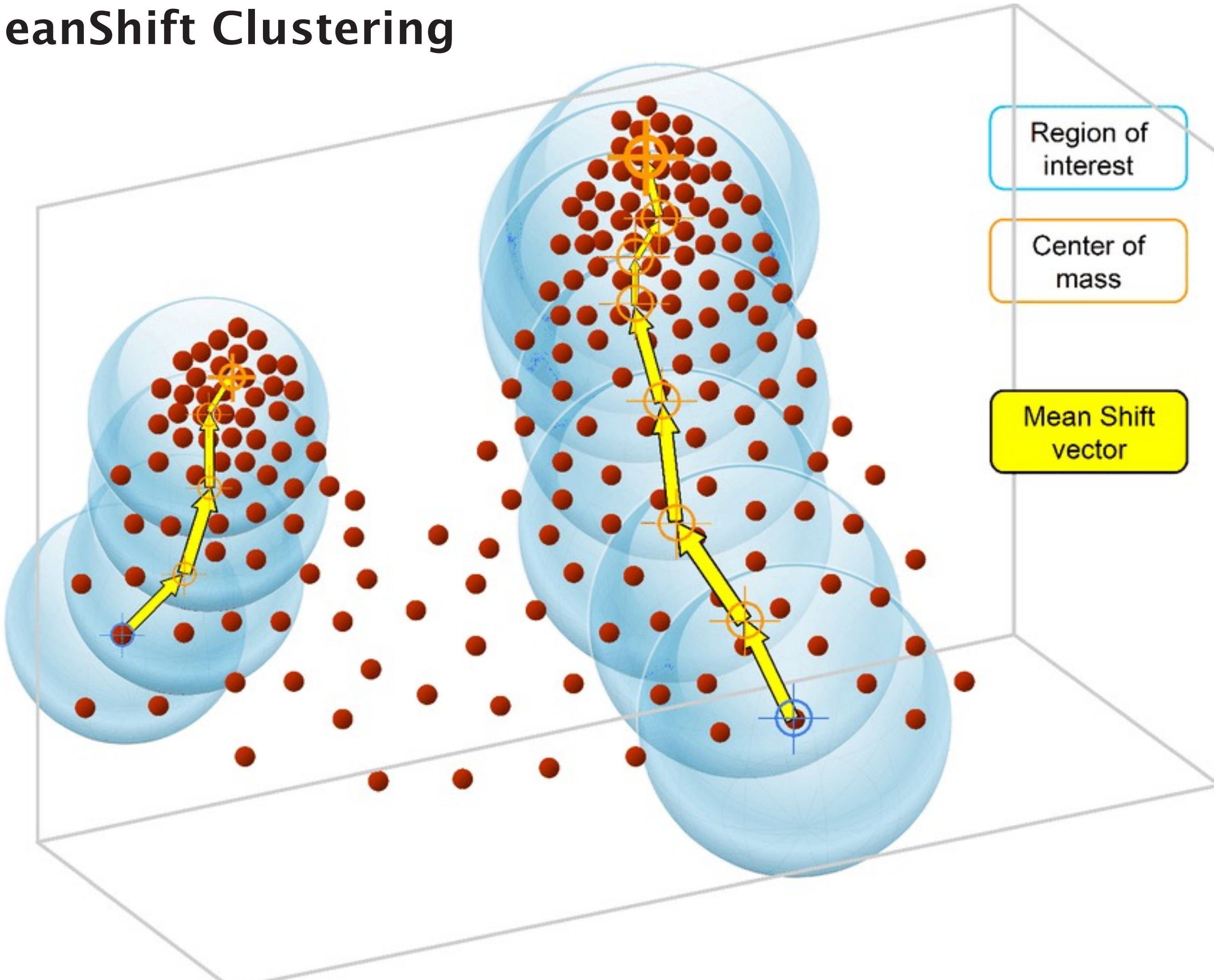
Credit: Jo Grundy



Source: https://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html#sphx-glr-auto-examples-cluster-plot-cluster-comparison-py

Discovering Groups – **Appendix (1/2)**

MeanShift Clustering



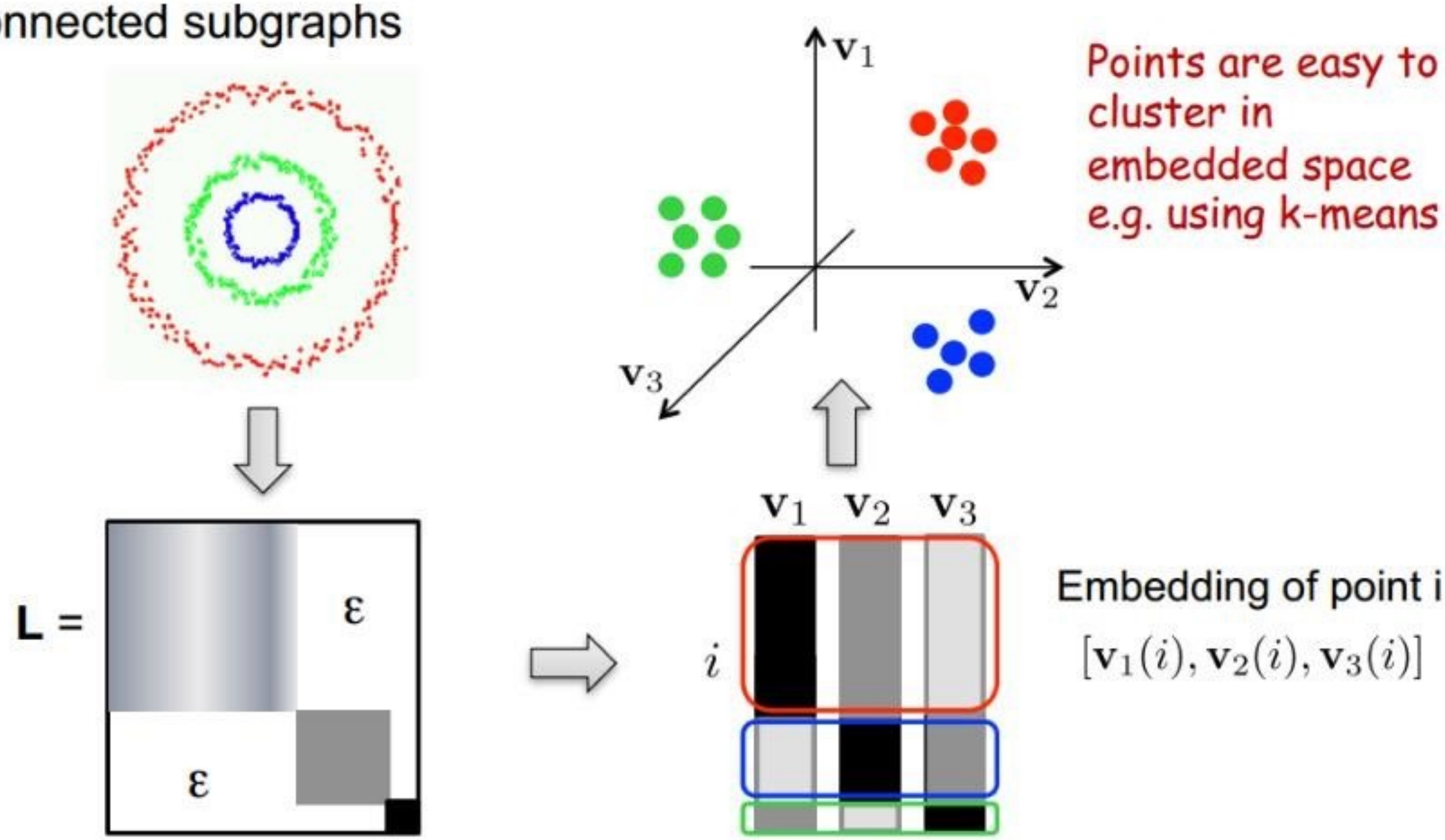
https://ailephant.com/how-to-program-mean-shift/

Discovering Groups – **Appendix (2/2)**

Spectral Clustering

Eigenvectors of the Laplacian matrix provide an embedding of the data based on similarity.

Disconnected subgraphs



Slides Courtesy: Eric Xing, M. Hein & U.V. Luxburg