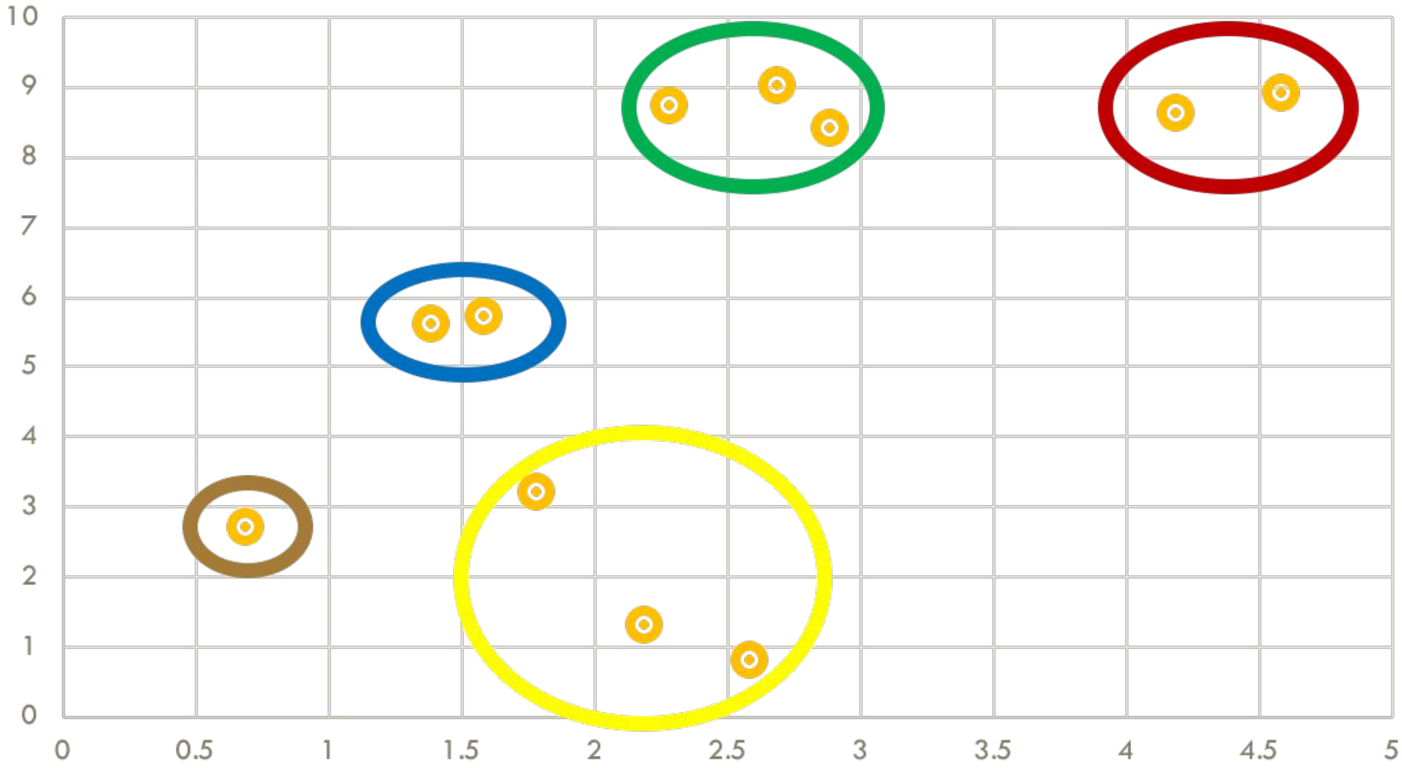
CSCI-620

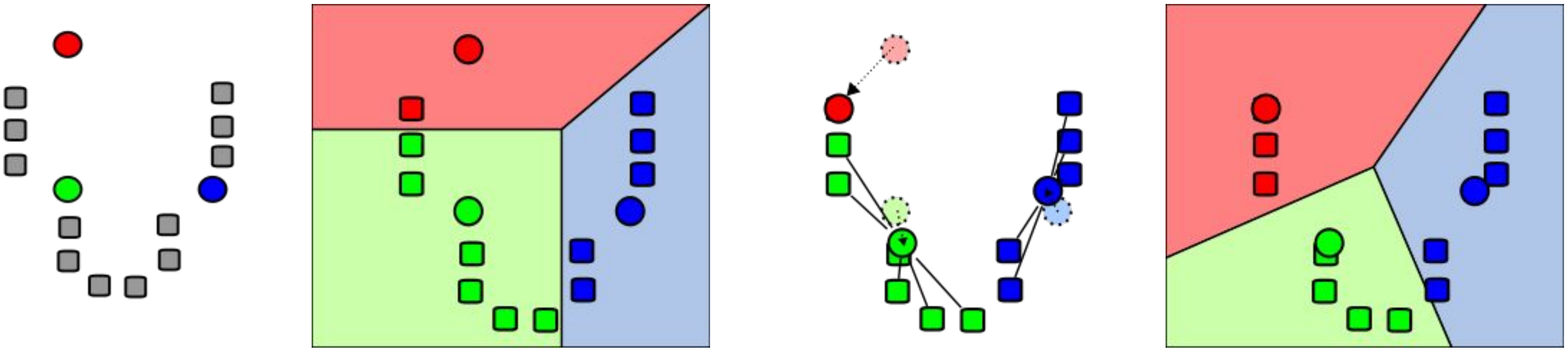
Clustering



Clustering

Clustering models

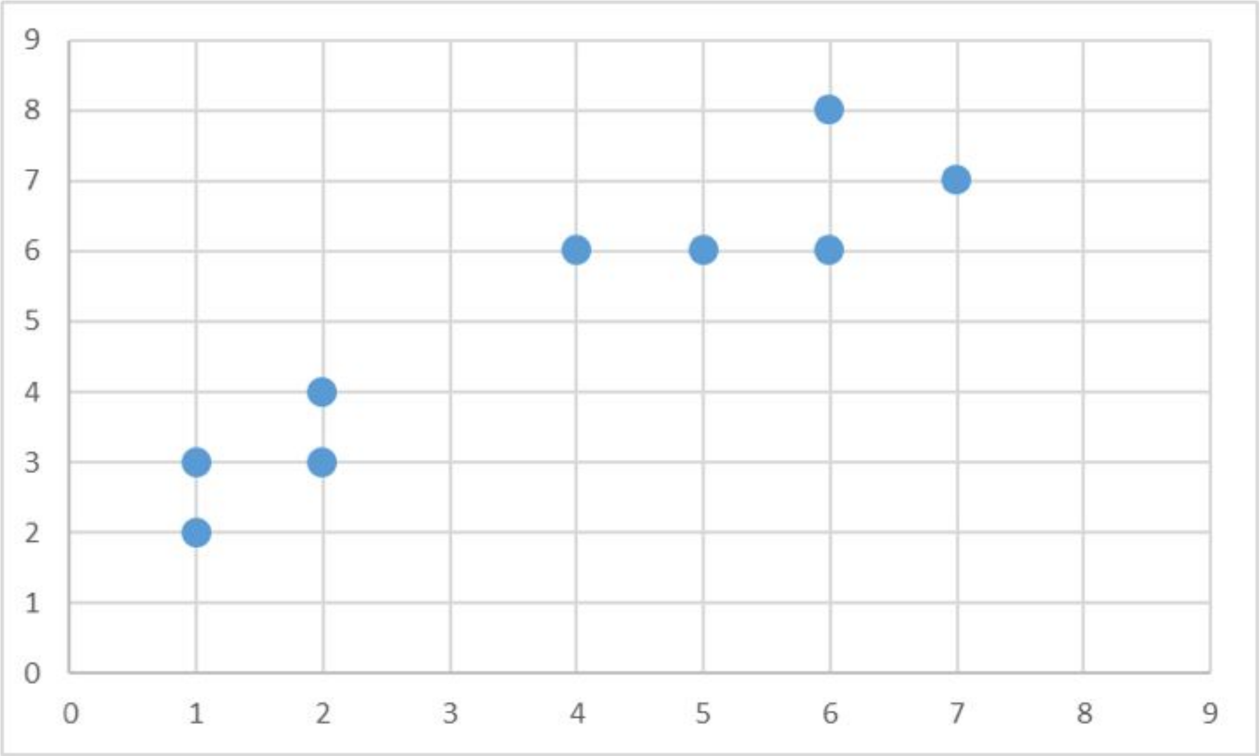
- Connectivity models based on connectivity distance
- Centroid models based on central individuals and distance
- Density models based on connected and dense regions in a space
- Graph-based models based on cliques and their relaxations



K-means clustering

```
Initially choose k points that are likely to be in
    different clusters;
Make these points the centroids of their clusters;
FOR each remaining point p DO
    find the centroid to which p is closest;
    Add p to the cluster of that centroid;
    Adjust the centroid of that cluster to account for p;
END;
```

K-means clustering



Example

Points:

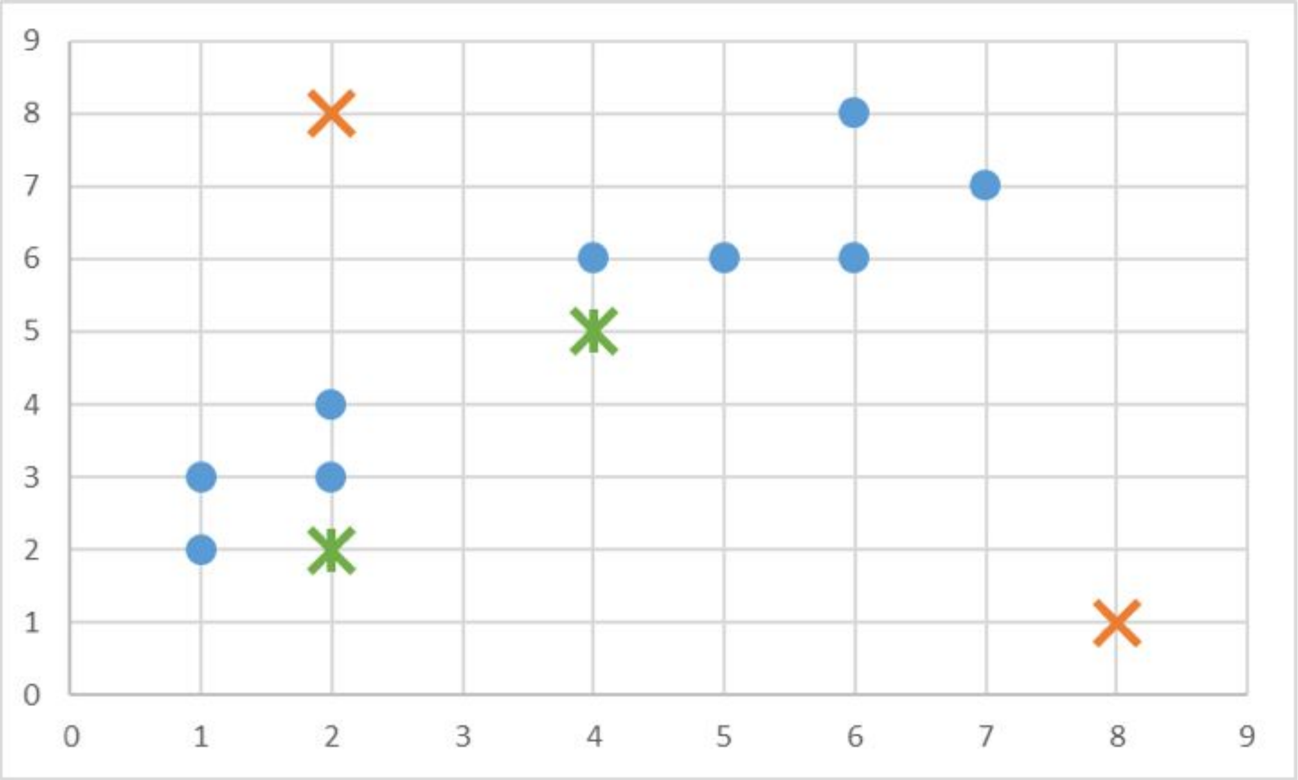
(1, 2), (1, 3), (2, 3), (2, 4), (4, 6),
(5, 6), (6, 6), (6, 8), (7, 7)

Manhattan distance: $d((a, b), (x, y)) = |a - x| + |b - y|$

$k = 2$

Random centroids: 1 = (2, 8), 2 = (8, 1)

Example



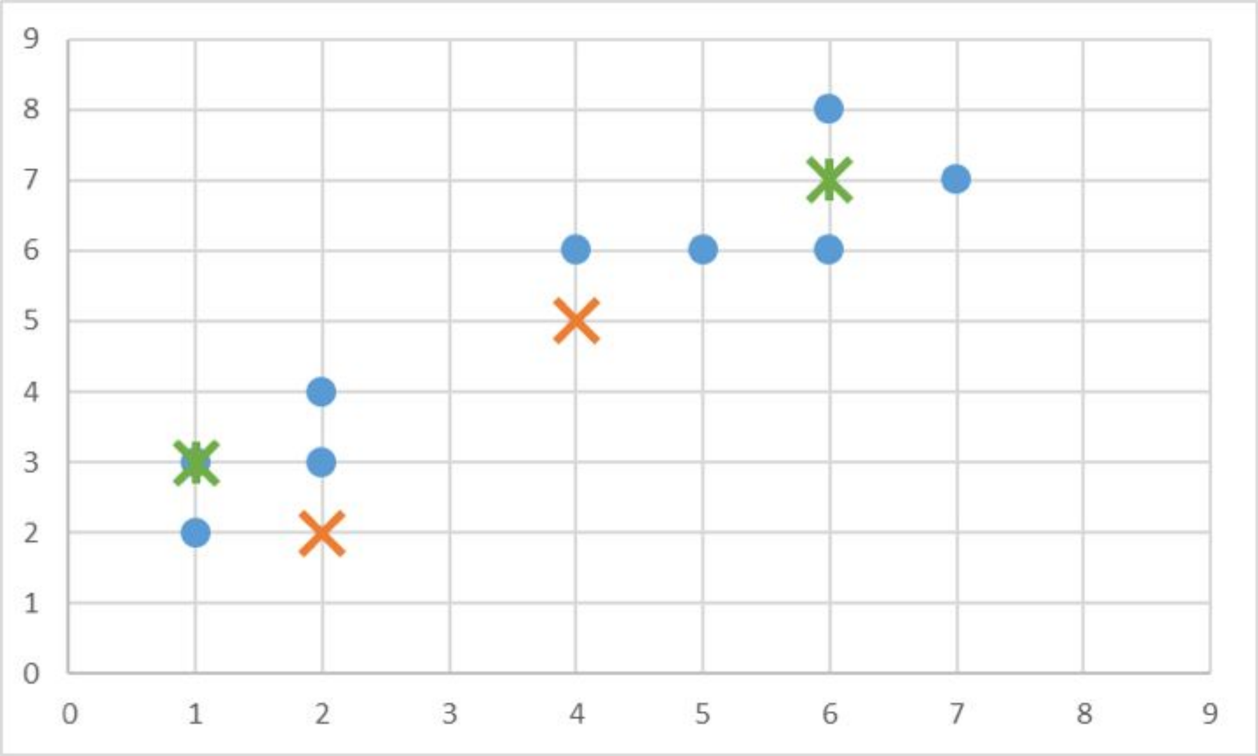
Example

Example

Centroid distances $\mu_1|\mu_2$

<table>
<tr><th></th><th>1</th><th>2</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td>2</td><td>7|8</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>3</td><td>6|9</td><td>5|8</td><td></td><td></td><td></td><td></td></tr>
<tr><td>4</td><td></td><td>4|9</td><td></td><td></td><td></td><td></td></tr>
<tr><td>6</td><td></td><td></td><td>4|11</td><td>5|8</td><td>6|7</td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td>6|7</td></tr>
<tr><td>8</td><td></td><td></td><td></td><td></td><td>4|9</td><td></td></tr>
</table>

Old centroids (2,8) (8,1)
New centroids (4,5) (2,2)

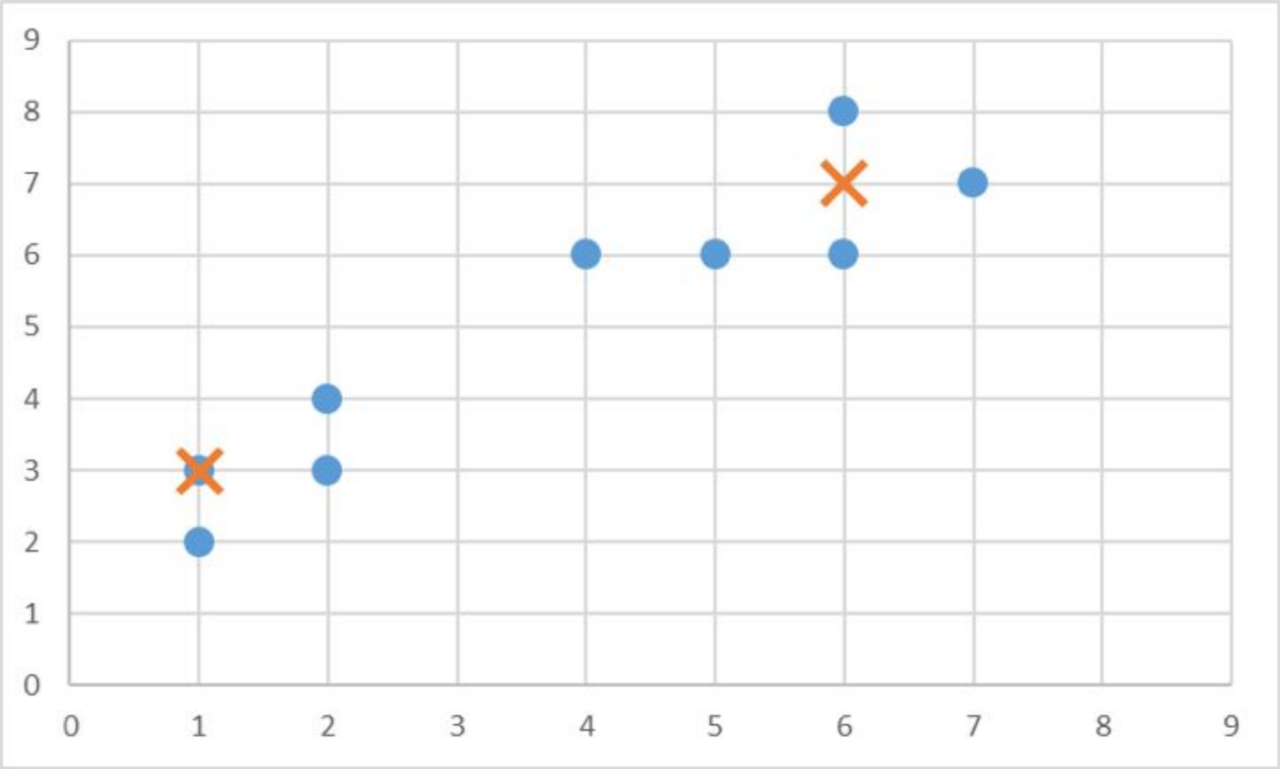


Example

<table>
<tr><th></th><th>1</th><th>2</th><th>4</th><th>5</th><th>6</th><th>7</th></tr>
<tr><td>2</td><td>10|1</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td>3</td><td>8|0</td><td>8|1</td><td></td><td></td><td></td><td></td></tr>
<tr><td>4</td><td></td><td>8|2</td><td></td><td></td><td></td><td></td></tr>
<tr><td>6</td><td></td><td></td><td>3|6</td><td>2|7</td><td>1|8</td><td></td></tr>
<tr><td>7</td><td></td><td></td><td></td><td></td><td></td><td>1|10</td></tr>
<tr><td>8</td><td></td><td></td><td></td><td></td><td>1|10</td><td></td></tr>
</table>

Old centroids (4,5) (2,2)
New centroids (6,7) (1,3)

Example



Example

	1	2	4	5	6	7
2	6|1					
3	5|2	4|1				
4		3|2				
6			1|6	2|5	3|8	
7						5|10
8					5|10	

Old centroids (6,7) (1,3)
New centroids (6,7) (1,3)

Example

Algorithm for KNN

- For each training example <x,f(x)>, add the example to the list of training_examples.

- Given a query instance x_q to be classified,
  - Let $x_1, x_2 \ldots . x_k$ denote the k instances from training_examples that are nearest to x_q .
  - Return the class that represents the maximum of the k instances.

$$SSE = \sum_{i=1}^{k} \sum_{x_j \in C_i} \left(x_j - \mu_i\right)^2$$

Sum of squared errors

Example

Calculate distance to centroids

$\mu_1 = (6, 7)$

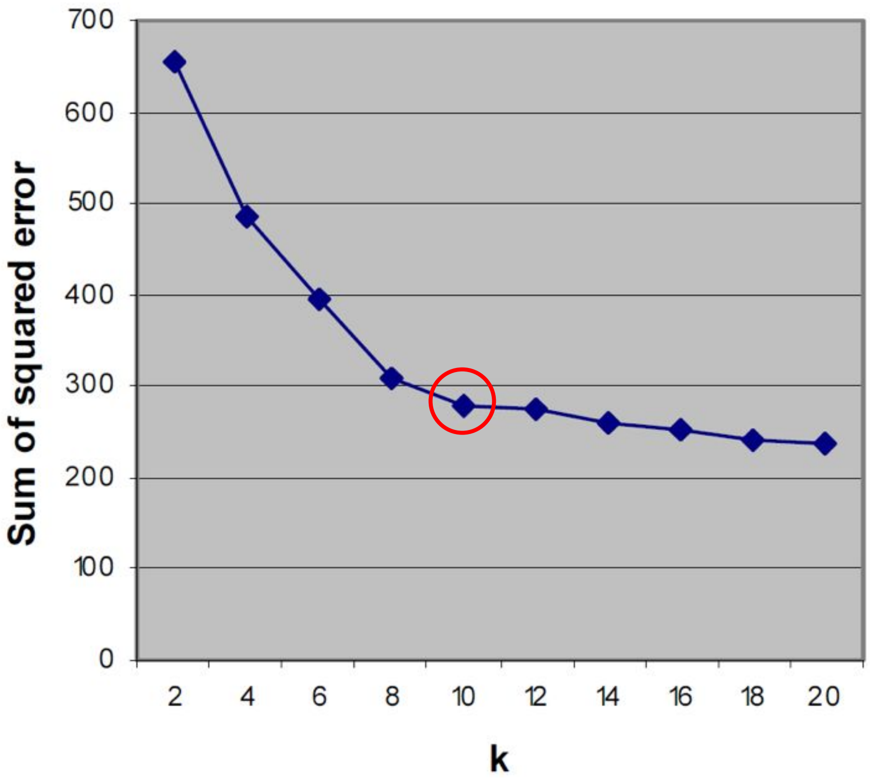
$\mu_2 = (1, 3)$

	1	2	4	5	6	7
2	1					
3	0	1				
4		2				
6			3	2	1	
7						1
8					1	

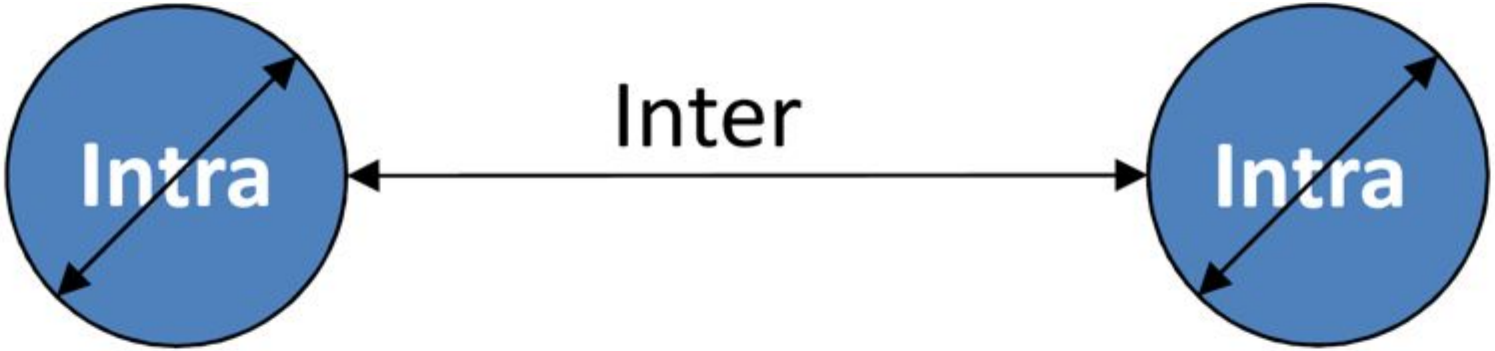
$$SSE = 1^2 + 0^2 + 1^2 + 2^2 + 3^2 + 2^2 + 1^2 + 1^2 + 1^2$$
$$= 22$$

Overfitting

- With any predicting algorithm we need to be careful to avoid *overfitting*
- Overfitting occurs when our model is too closely tied to our training data
- Usually a simpler model is better to avoid overfitting



Choosing cluster count



Inter/intra-cluster distance

For each x_i, $a(x_i)$ is the average distance between x_i and other points in C_k (the same cluster as x_i)

For each x_i and cluster C_j ($j \neq k$), let $d(x_i, C_j)$ be the average distance to other points in C_j

Let $b(x_i) = \min_{j \neq k} d(x_i, C_j)$ (the minimum average distance to any cluster)

$$S(x_i) = [b(x_i) - a(x_i)] / \max(a(x_i), b(x_i))$$

$S = \sum_i S(x_i)/m$ **closer to 1 is better!**

Silhouette coefficient

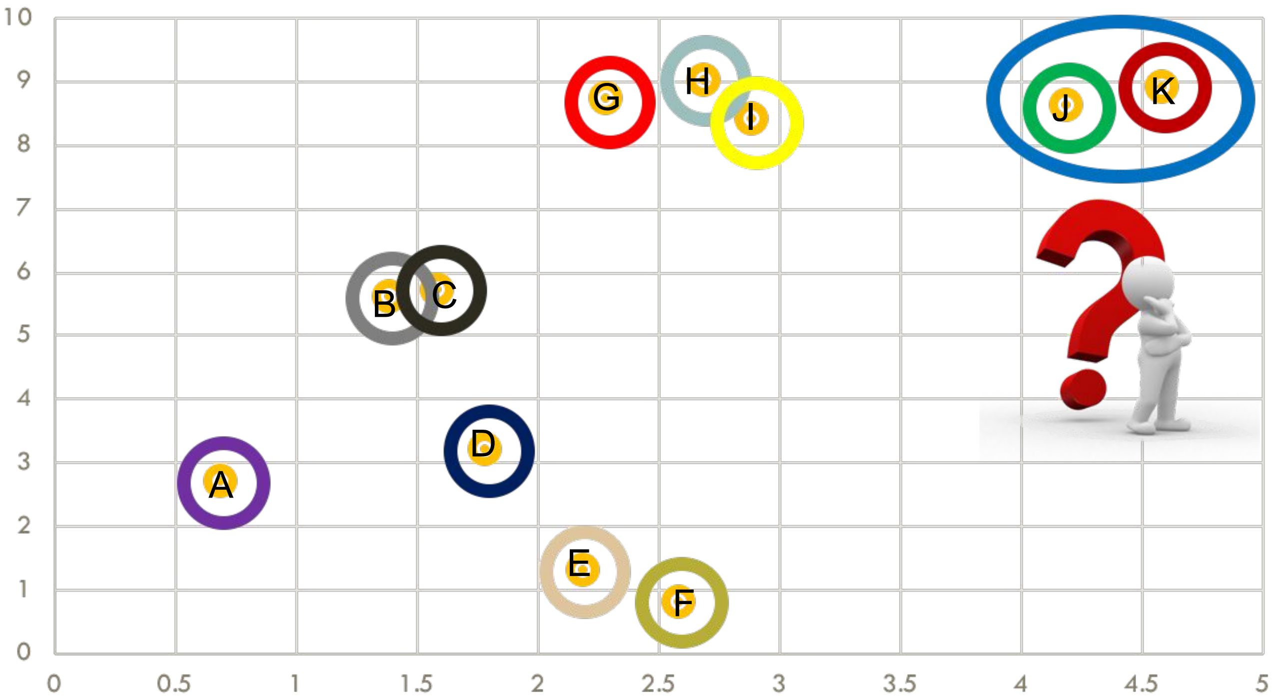
C_1: (1, 2), (1, 3) C_2: (3, 4), (4, 5) C_3: (7, 7), (8, 7)

S((1, 2)) = (5 – 1) / 5
S((3, 4)) = (3.5 – 2) / 3.5
S((7, 7)) = (6 – 1) / 6

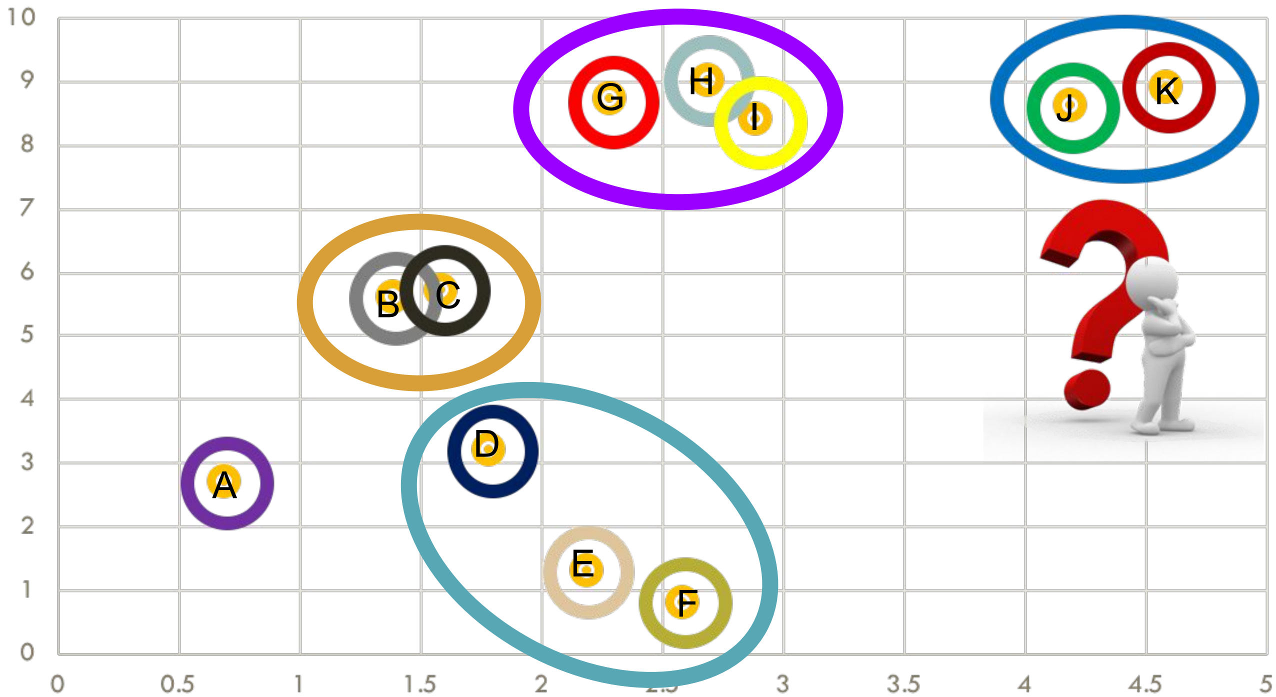
S((1, 3)) = (4 – 1) / 4
S((4, 5)) = (5.5 – 2) / 5.5
S((8, 7)) = (7 – 1) / 7

S = (4/5 + 3/4 + 1.5/3.5 + 3.5/5.5 + 5/6 + 6/7) / 6
= (0.8 + 0.75 + 0.43 + 0.64 + 0.83 + 0.86) / 6
= 0.72

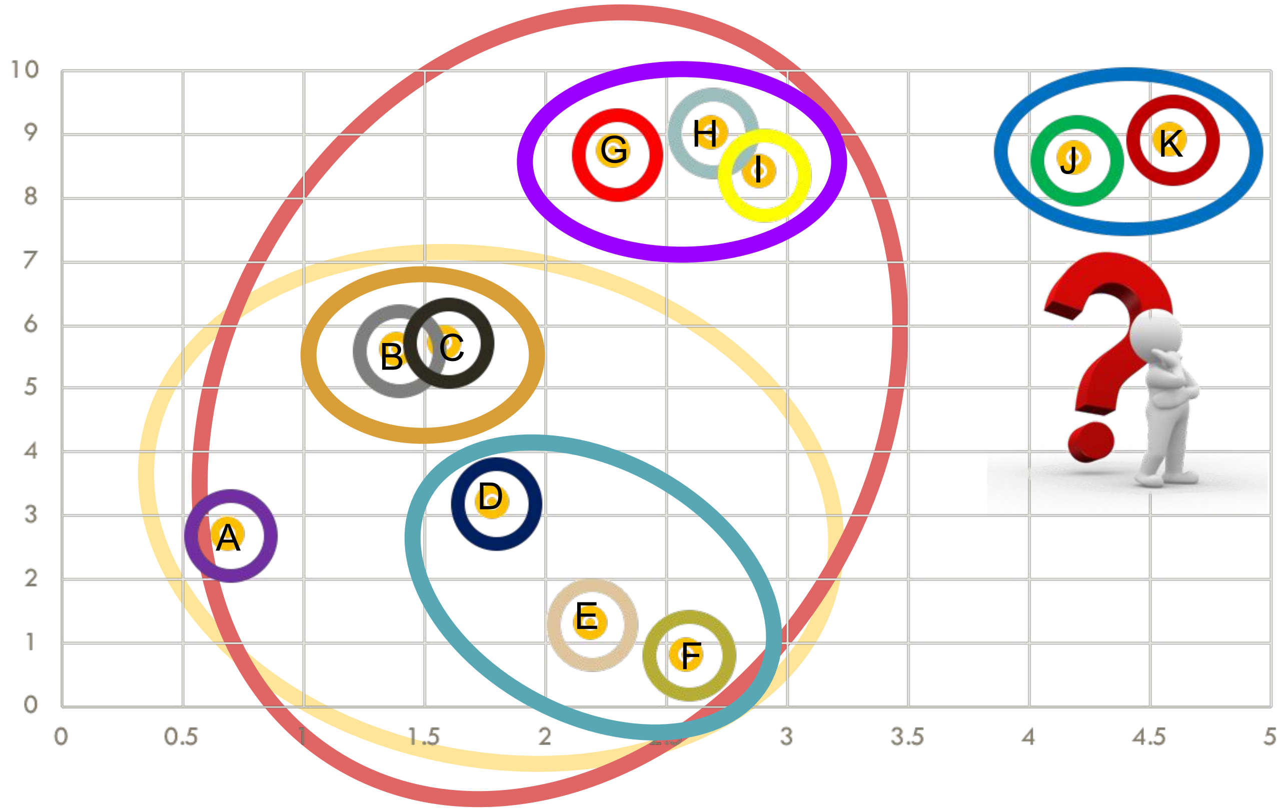
Example



Hierarchical clustering



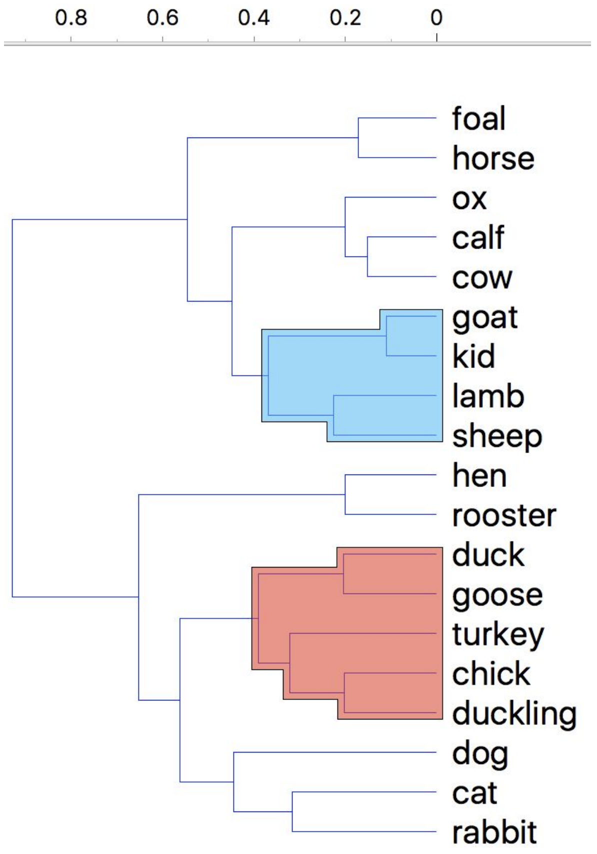
Hierarchical clustering



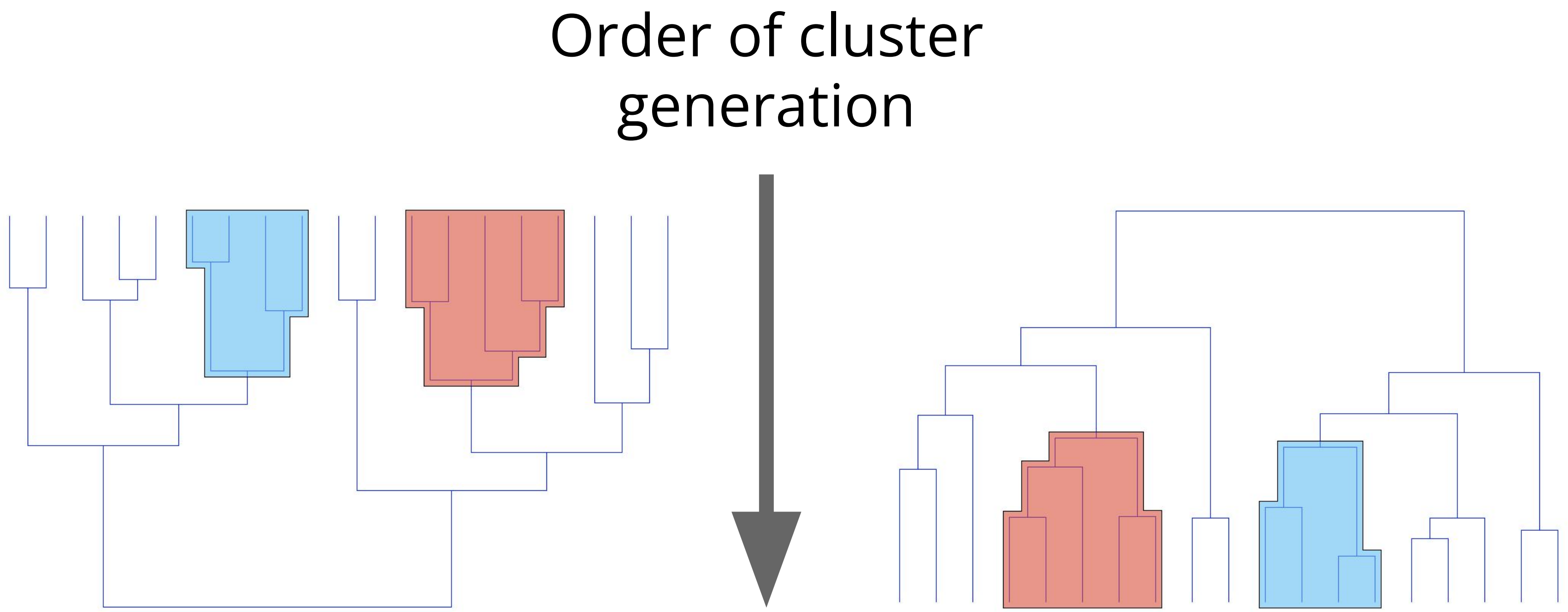
Hierarchical clustering

```
WHILE it is not time to stop DO
    pick the best two clusters to merge;
    combine those two clusters into one cluster;
END;
```

Hierarchical clustering


0.8
0.6
0.4
0.2
0
foal
horse
ox
calf
cow
goat
kid
lamb
sheep
hen
rooster
duck
goose
turkey
chick
duckling
dog
cat
rabbit

Dendograms



Agglomerative vs Divisive

Single-linkage clustering

- Agglomerative clustering algorithm
- Distance between clusters is based on the closest point in each cluster
- Continue clustering until all points are a single cluster

	BOS	NY	DC	MIA	CHI	SEA	SF	LA	DEN
BOS	0	206	429	1504	963	2976	3095	2979	1949
NY	206	0	233	1308	802	2815	2934	2786	1771
DC	429	233	0	1075	671	2684	2799	2631	1616
MIA	1504	1308	1075	0	1329	3273	3053	2687	2037
CHI	963	802	671	1329	0	2013	2142	2054	996
SEA	2976	2815	2684	3273	2013	0	808	1131	1307
SF	3095	2934	2799	3053	2142	808	0	379	1235
LA	2979	2786	2631	2687	2054	1131	379	0	1059
DEN	1949	1771	1616	2037	996	1307	1235	1059	0

Example

	BOS/NY	DC	MIA	CHI	SEA	SF	LA	DEN
BOS/NY	0	223	1308	802	2815	2934	2786	1771
DC	223	0	1075	671	2684	2799	2631	1616
MIA	1308	1075	0	1329	3273	3053	2687	2037
CHI	802	671	1329	0	2013	2142	2054	996
SEA	2815	2684	3273	2013	0	808	1131	1307
SF	2934	2799	3053	2142	808	0	379	1235
LA	2786	2631	2687	2054	1131	379	0	1059
DEN	1771	1616	2037	996	1307	1235	1059	0

Example

	BOS/NY/DC	MIA	CHI	SEA	SF	LA	DEN
BOS/NY/DC	0	1075	671	2684	2799	2631	1616
MIA	1075	0	1329	3273	3053	2687	2037
CHI	671	1329	0	2013	2142	2054	996
SEA	2684	3273	2013	0	808	1131	1307
SF	2799	3053	2142	808	0	379	1235
LA	2631	2687	2054	1131	379	0	1059
DEN	1616	2037	996	1307	1235	1059	0

Example

	BOS/ NY/DC	MIA	CHI	SEA	SF/LA	DEN
BOS/NY/DC	0	1075	671	2684	2631	1616
MIA	1075	0	1329	3273	2687	2037
CHI	671	1329	0	2013	2054	996
SEA	2684	3273	2013	0	808	1307
SF/LA	2631	2687	2054	808	0	1059
DEN	1616	2037	996	1307	1059	0

Example

	BOS/NY/DC/ CHI	MIA	SEA	SF/LA	DEN
BOS/NY/DC/CHI	0	1075	2013	2054	996
MIA	1075	0	3273	2687	2037
SEA	2013	3273	0	808	1307
SF/LA	2054	2687	808	0	1059
DEN	996	2037	1307	1059	0

Example

	BOS/NY/DC/CHI	MIA	SF/LA/SEA	DEN
BOS/NY/DC/CHI	0	1075	2013	996
MIA	1075	0	2687	2037
SF/LA/SEA	2054	2687	0	1059
DEN	996	2037	1059	0

Example

	BOS/NY /DC/CHI/DEN	MIA	SF/LA/SEA
BOS/NY/DC/CHI/DEN	0	1075	1059
MIA	1075	0	2687
SF/LA/SEA	1059	2687	0

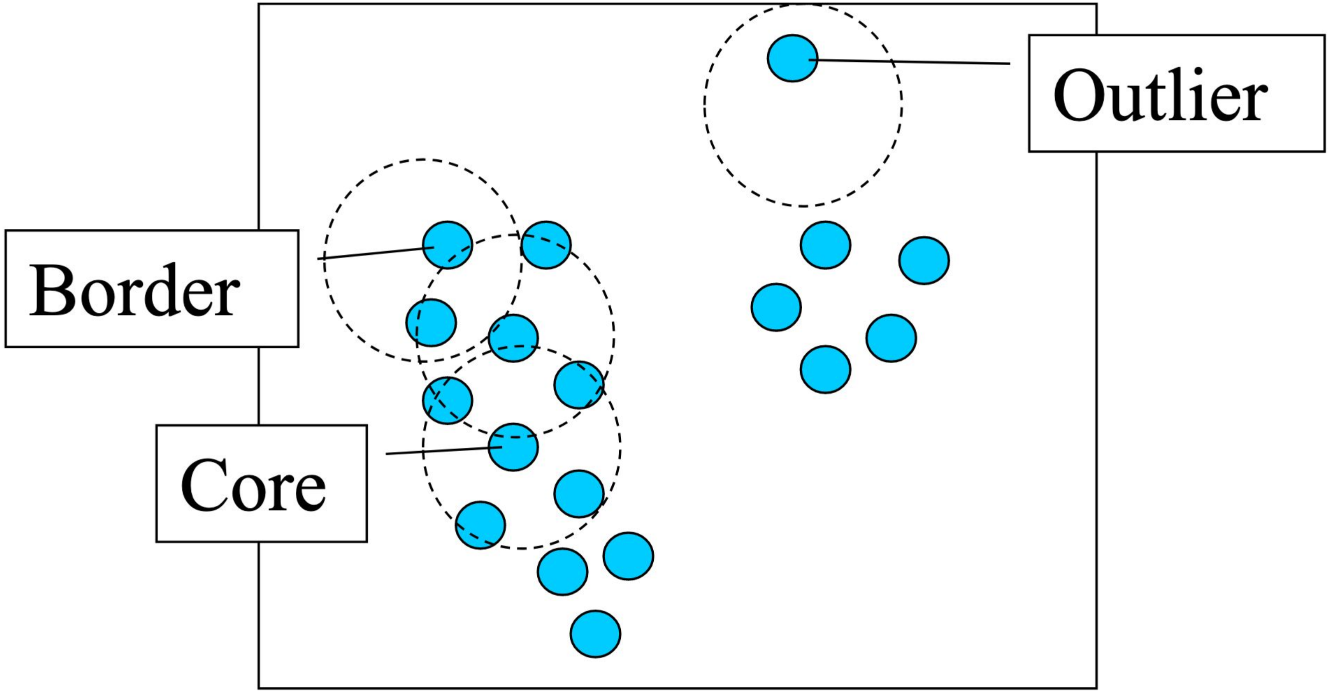
Example

	BOS/NY /DC/CHI /DEN/SF /LA/SEA	MIA
BOS/NY/DC/CHI/DEN/SF/LA/SEA	0	1075
MIA	1075	0

Example

Density-based clustering

- Group points with similar density
- Clusters should be separated by areas with low density
- Allows for easier generation of clusters with different sizes



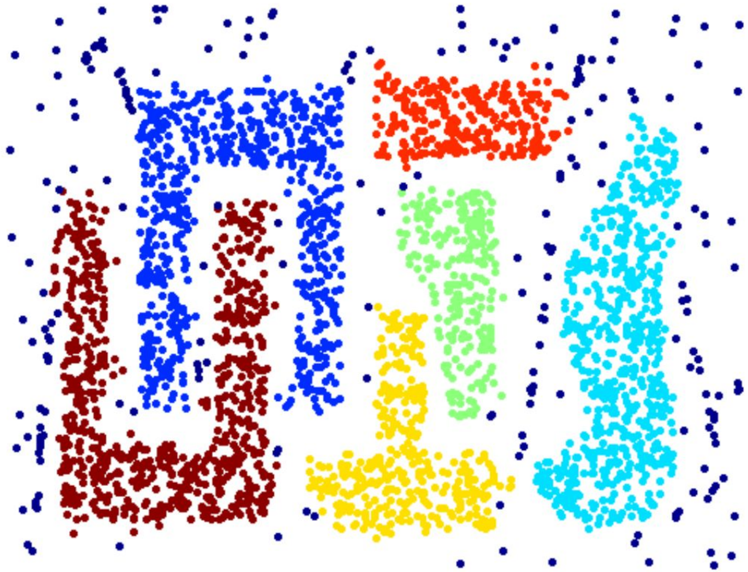
ε = 1unit, MinPts = 5

DBSCAN

For each point *p*
    If *p* is not classified
        If *p* is a core object
            Create a new cluster
            Assign density-reachable points to the cluster
        Else
            Classify *p* as noise

DBSCAN

Original Points

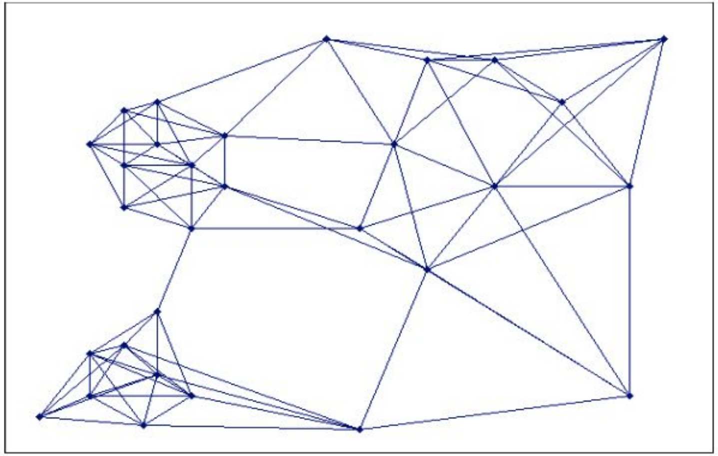


Clusters

ε = 10, MinPts = 4

DBSCAN

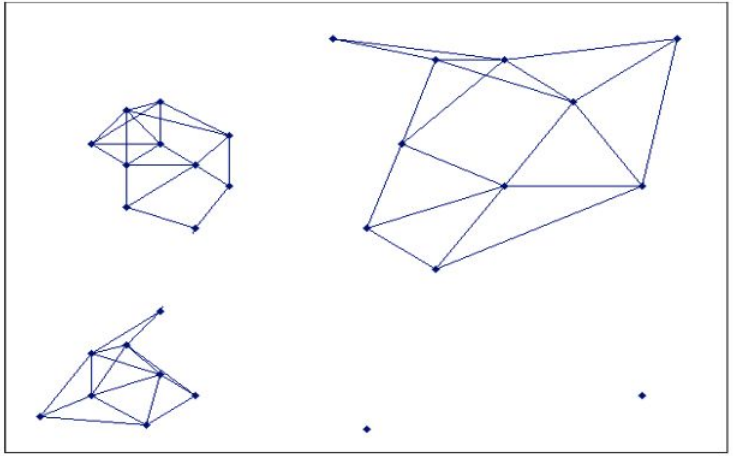
Graph clustering



Graph partitioning

⇒

Each connected component is a cluster



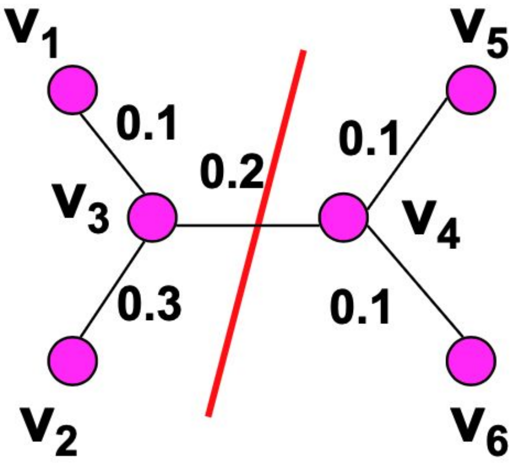
Graph clustering

We need two things for graph clustering:

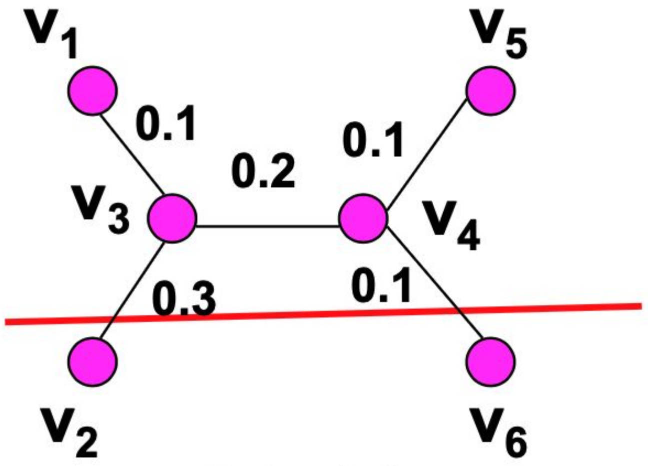
1. An objective function to determine the best way to cut the graph
2. An algorithm to find the optimal partitioning of the graph

$$\mathrm{Cut}(V_1, V_2) = \sum_{\substack{i \in V_1, \\ j \in V_2}} w_{ij}$$

w_{ij} is weight of the edge between nodes i and j

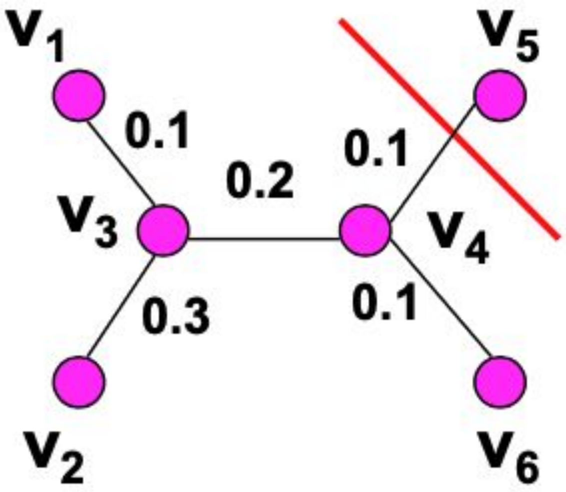


Cut = 0.2



Cut = 0.4

Graph cut



Cut = 0.1

Min cut

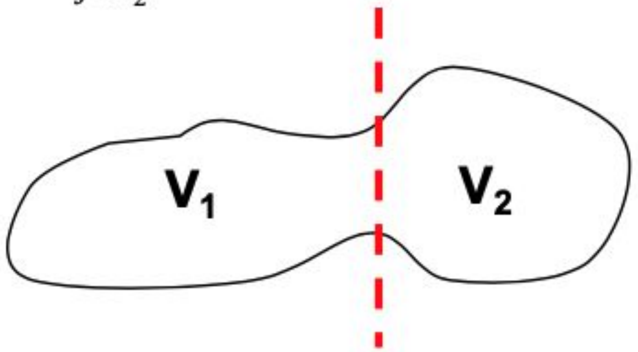
$$\text{Ratio cut}(V_1,V_2) = \frac{\text{Cut}(V_1,V_2)}{|V_1|} + \frac{\text{Cut}(V_1,V_2)}{|V_2|}$$

$$\text{Normalized cut}(V_1,V_2) = \frac{\text{Cut}(V_1,V_2)}{\sum_{i \in V_1} d_i} + \frac{\text{Cut}(V_1,V_2)}{\sum_{j \in V_2} d_j}$$

$$\text{where } d_i = \sum_j w_{ij}$$

V_1 and V_2 are the set of nodes in partitions 1 and 2

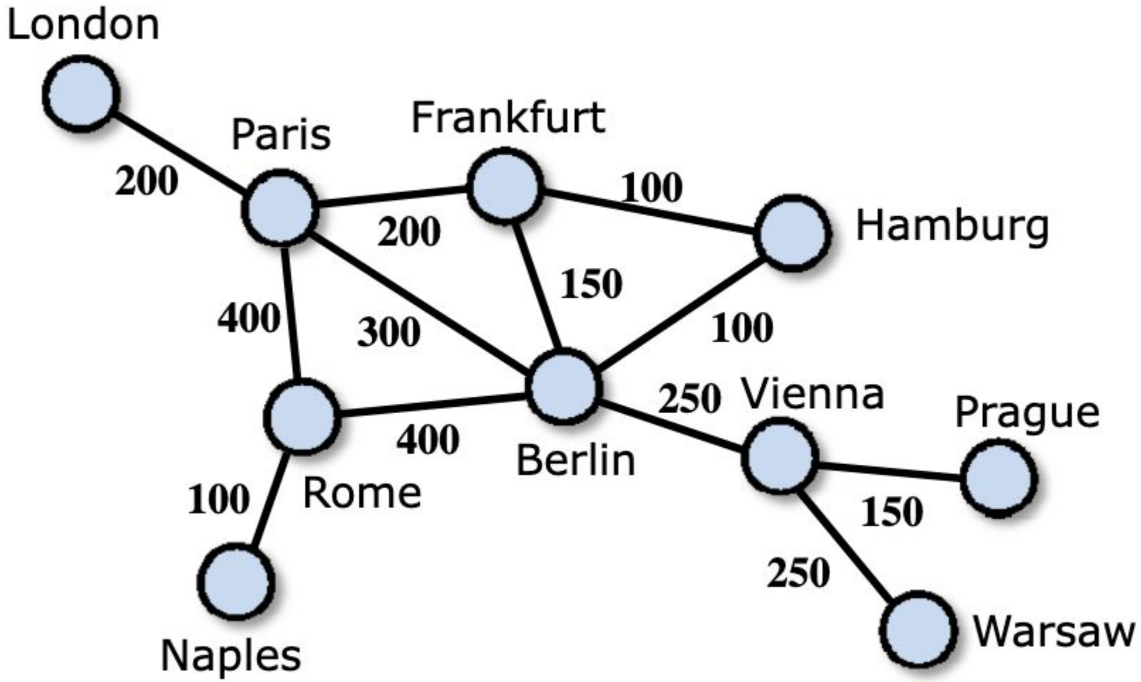
$|V_i|$ is the number of nodes in partition V_i



Balanced cut

Greedy partitioning

As a simple clustering algorithm, we can just pick the smallest edges and stop when we have *k* clusters



Greedy partitioning