Clustering 2

26 March 2026

Alex Lyman

Clustering Review

Clustering

- In unsupervised learning you are given a data set with no output classifications (labels)
- The goal in clustering is to find "natural" clusters (classes) into which the data can be divided – a particular breakdown into clusters is a clustering
- Generalization with clustering: when given a novel instance, we just assign it to the most similar cluster

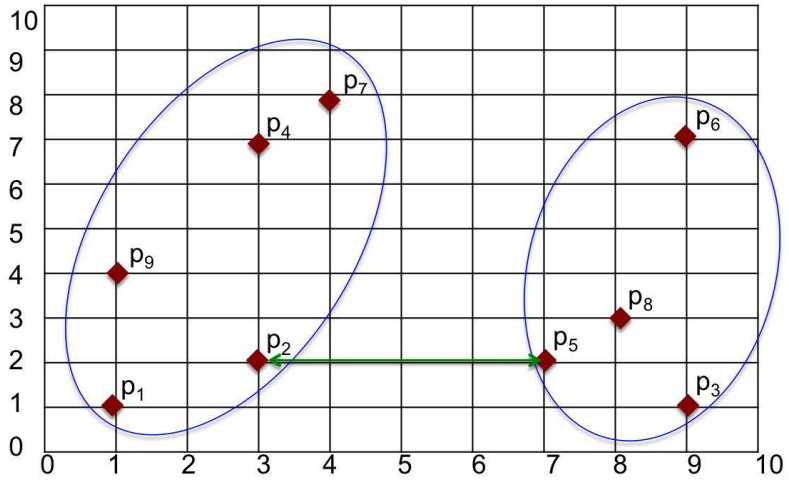
Clustering

- How do we decide which instances should be in which cluster?
- Typically put data which is "similar" into the same cluster
  - Similarity is measured with some distance metric
- Also try to maximize between-class dissimilarity
- Seek balance of within-class similarity and between-class dissimilarity
- Similarity Metrics
  - Euclidean Distance most common for real valued instances
  - Can use (1,0) distance for nominal and unknowns like with *k*-NN
  - Can create arbitrary distance metrics based on the task
  - Important to normalize the features

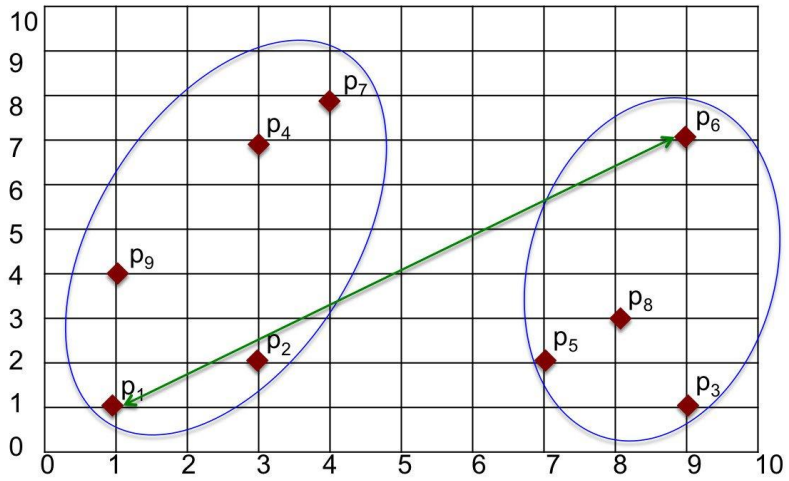
Distances Between Clusters

- Easy to measure distance between instances (elements, points), but how about the distance of an instance to another cluster or the distance between 2 clusters
- Can represent a cluster with
  - Centroid – cluster mean
    - Then just measure distance to the centroid
  - Medoid – an actual instance which is most typical of the cluster (e.g. Medoid is point which would make the average distance from it to the other points the smallest)
- Other common distances between two Clusters *A* and *B*
  - Single link – Smallest distance between any 2 points in *A* and *B*
  - Complete link – Largest distance between any 2 points in *A* and *B*
  - Average link – Average distance between points in *A* and points in *B*

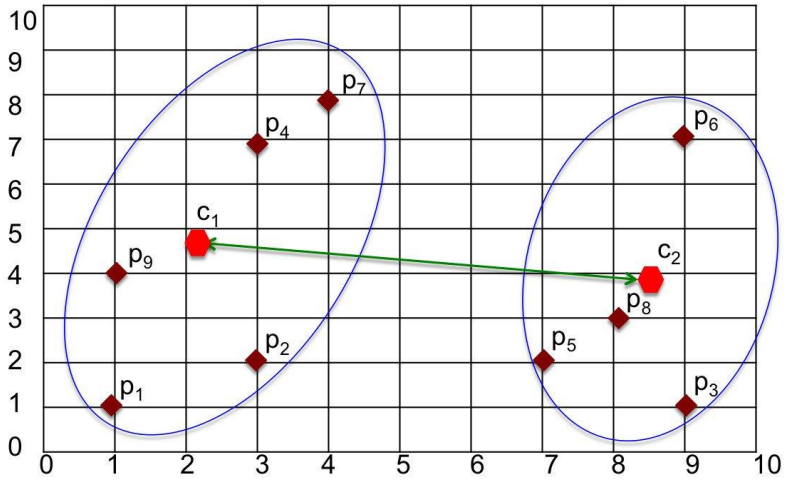
Distances Between Clusters



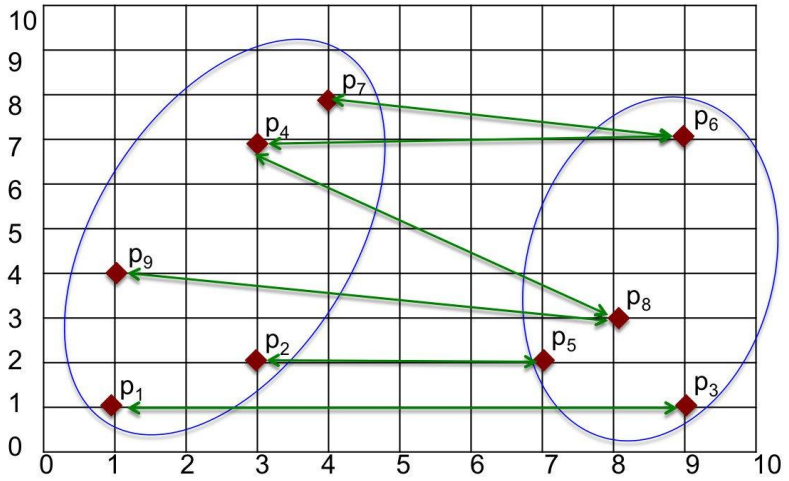
Single Link



Complete(max) Link



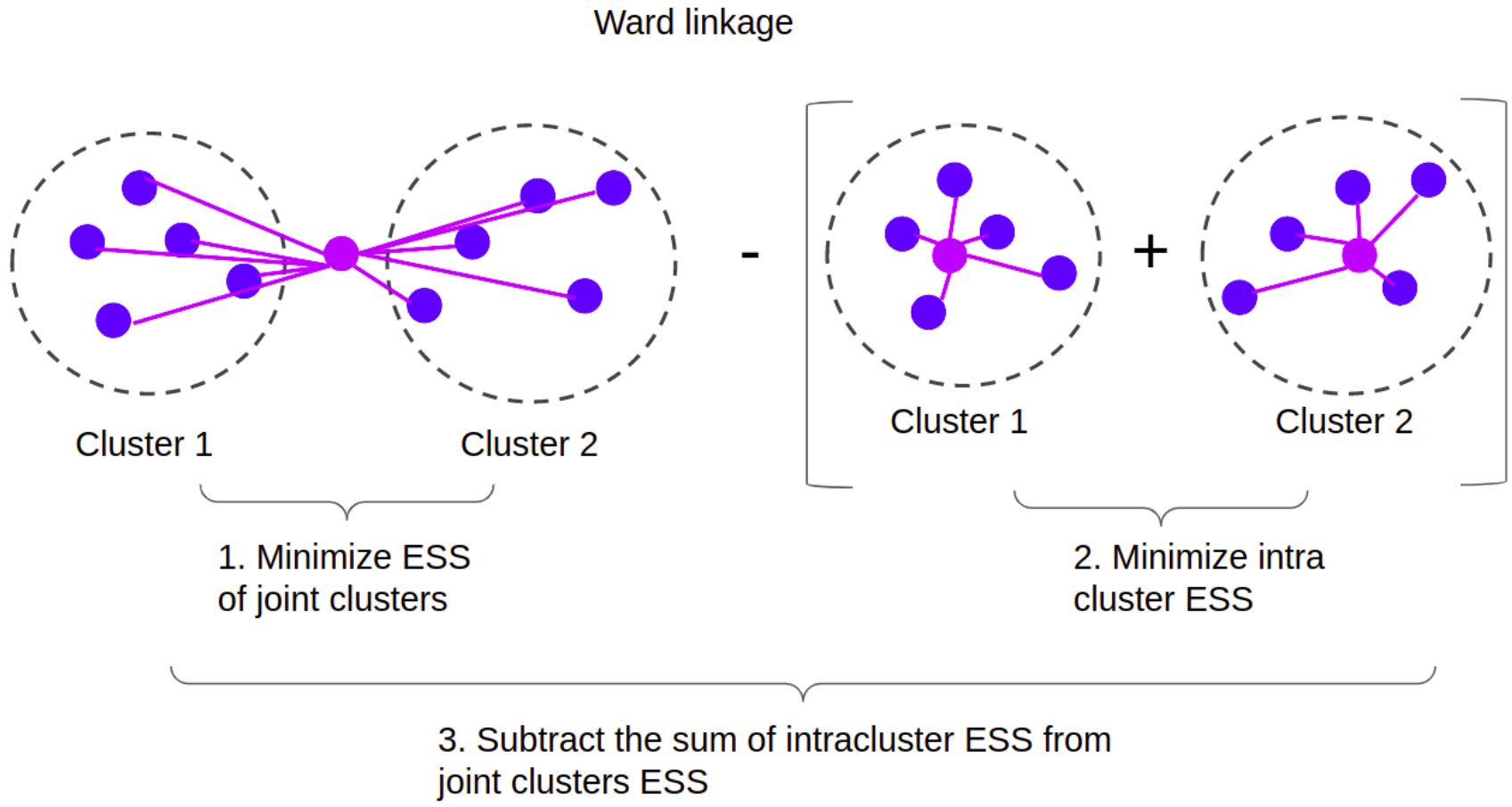
Centroid Link



Average Link

Ward Linkage

Ward linkage measures variance of clusters. The distance between two clusters, A and B, is how much the sum of squares distance from each point to its centroid would increase if we merged them. Merge the two clusters with minimum increase.



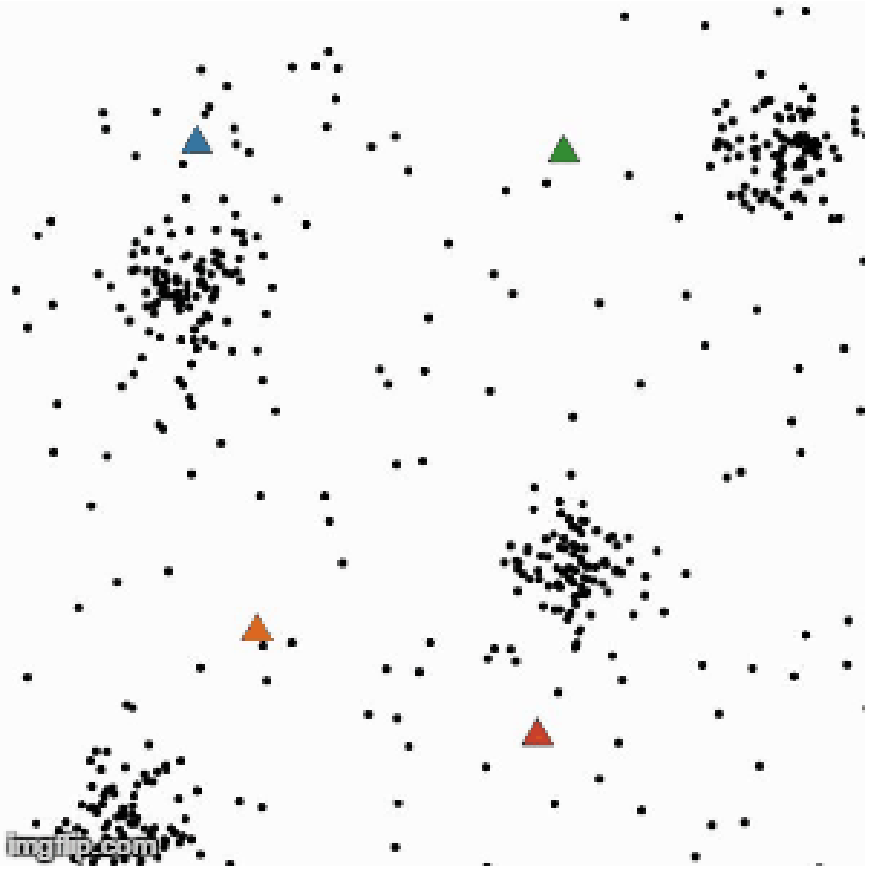
K-Means Clustering

K-Means

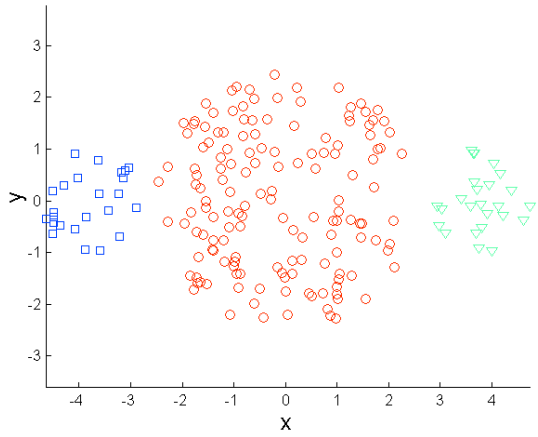
1. You choose *k* beforehand!

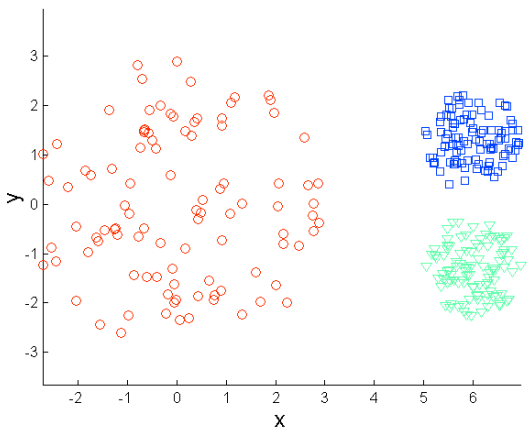
2. Randomly choose *k* instances from the data set to be the initial *k* centroids

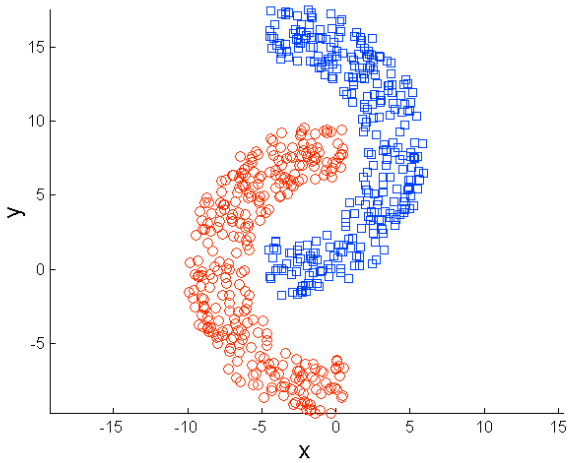
3. Repeat until no (or negligible) changes occur
    a) Group each instance with its closest centroid
    b) Recalculate the centroid based on its new cluster

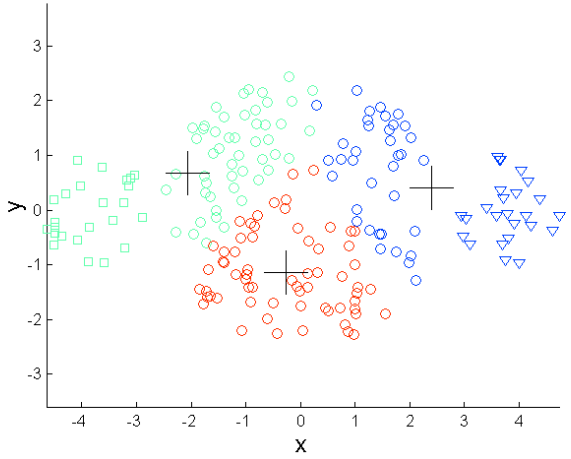


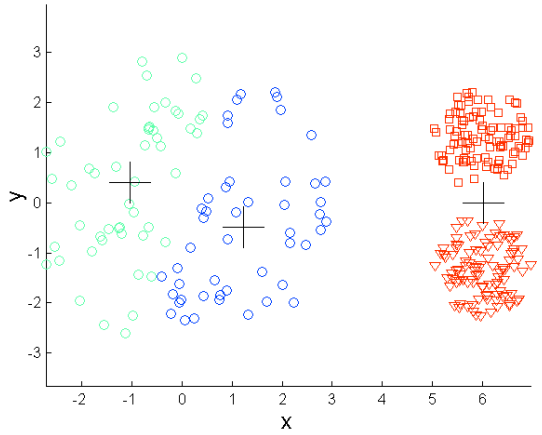
Limitations of K-means

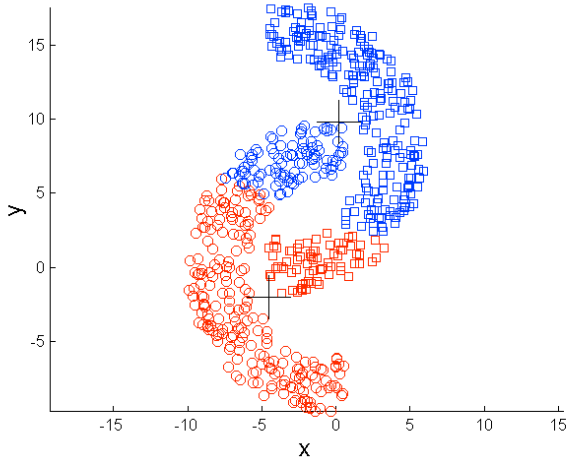








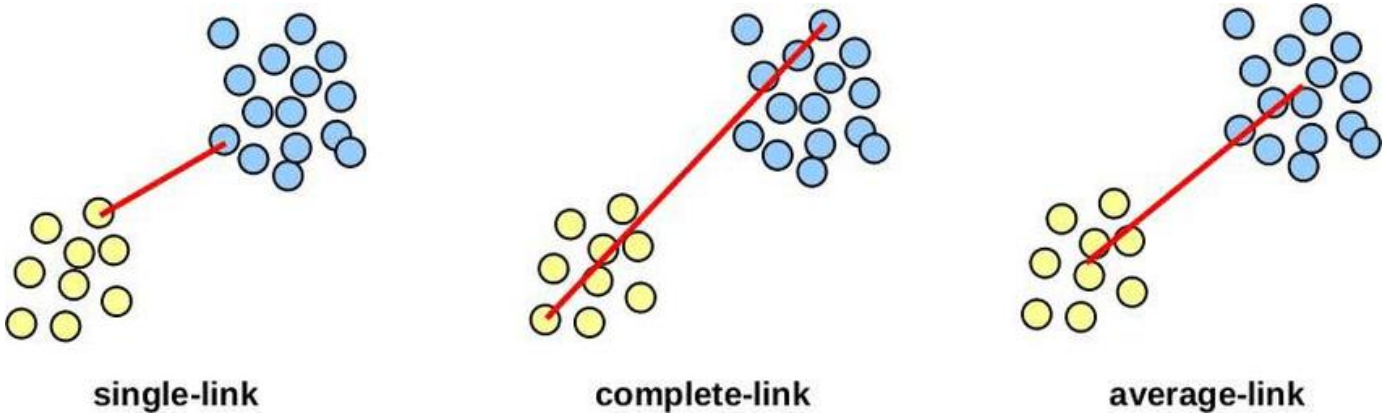


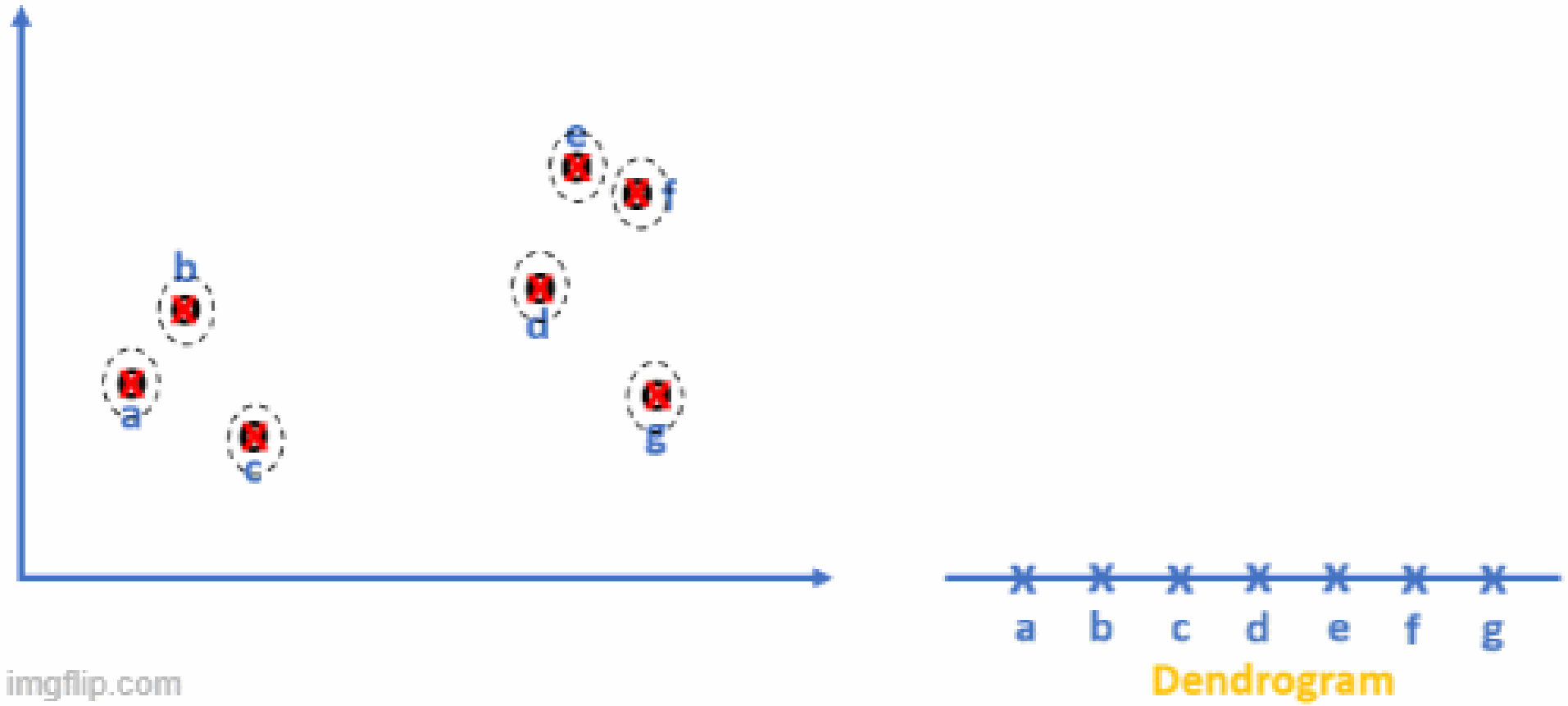


Hierarchical Clustering

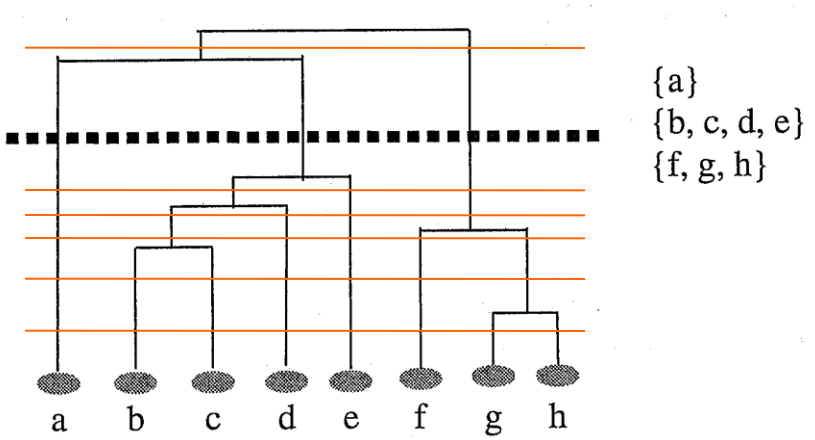
Hierarchical Agglomerative Clustering (HAC)

- Input is an $n \times n$ adjacency matrix giving the distance between each pair of instances
- Initialize each instance to be its own cluster
- Repeat until there is just one cluster containing all instances
  - Merge the two "closest" remaining clusters into one cluster
- HAC varies based on "Closeness definition"
  - single, complete, and average link distances common



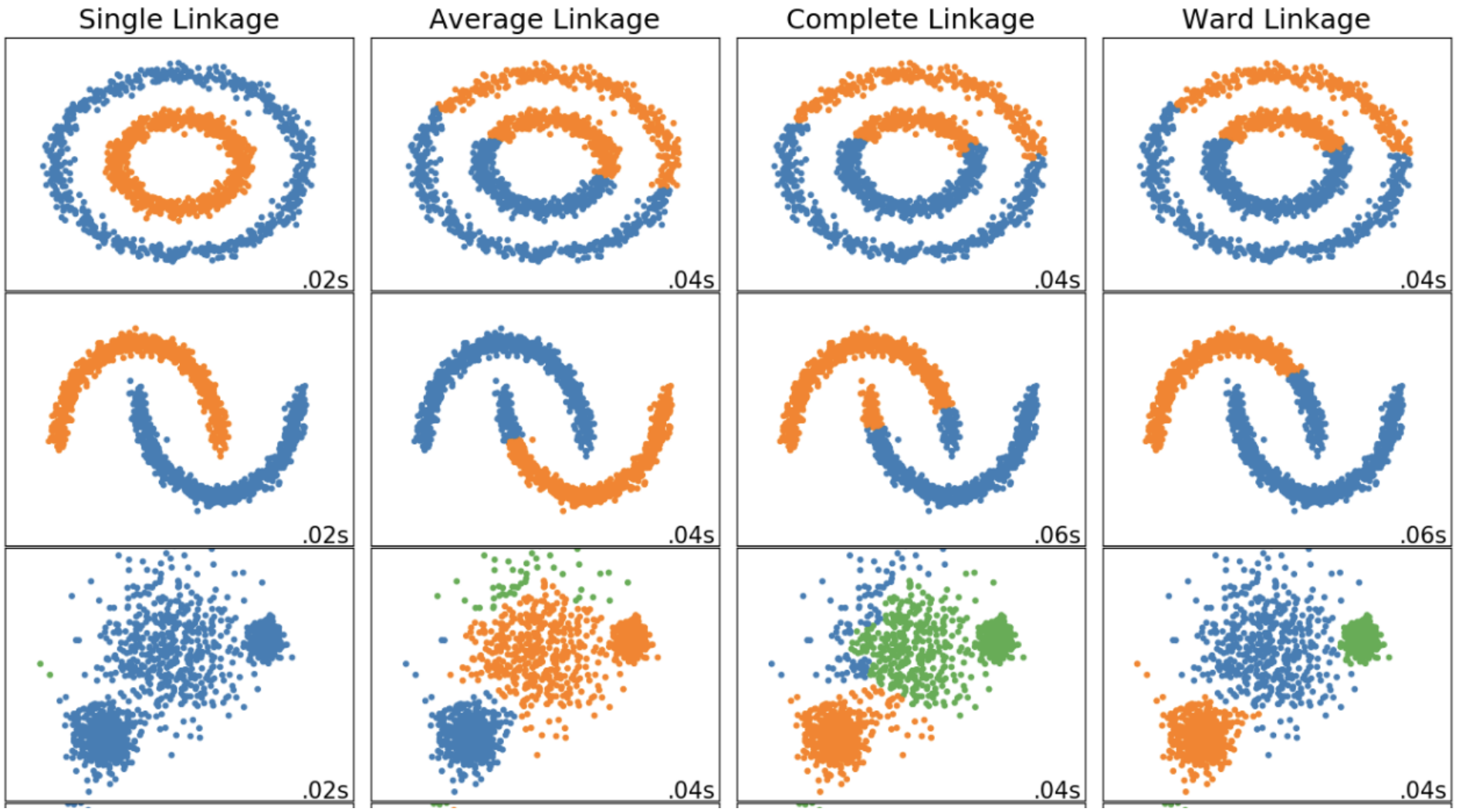

e
f
b
d
a
g
c
a b c d e f g
Dendrogram
imgflip.com

Which cluster level to choose?



- Depends on goals
  - May know beforehand how many clusters you want - or at least a range (e.g. 2-10)
  - Could analyze the dendrogram and data after the full clustering to decide which subclustering level is most appropriate for the task at hand
  - Could use automated *cluster validity* metrics to help
- Could do stopping criteria during clustering

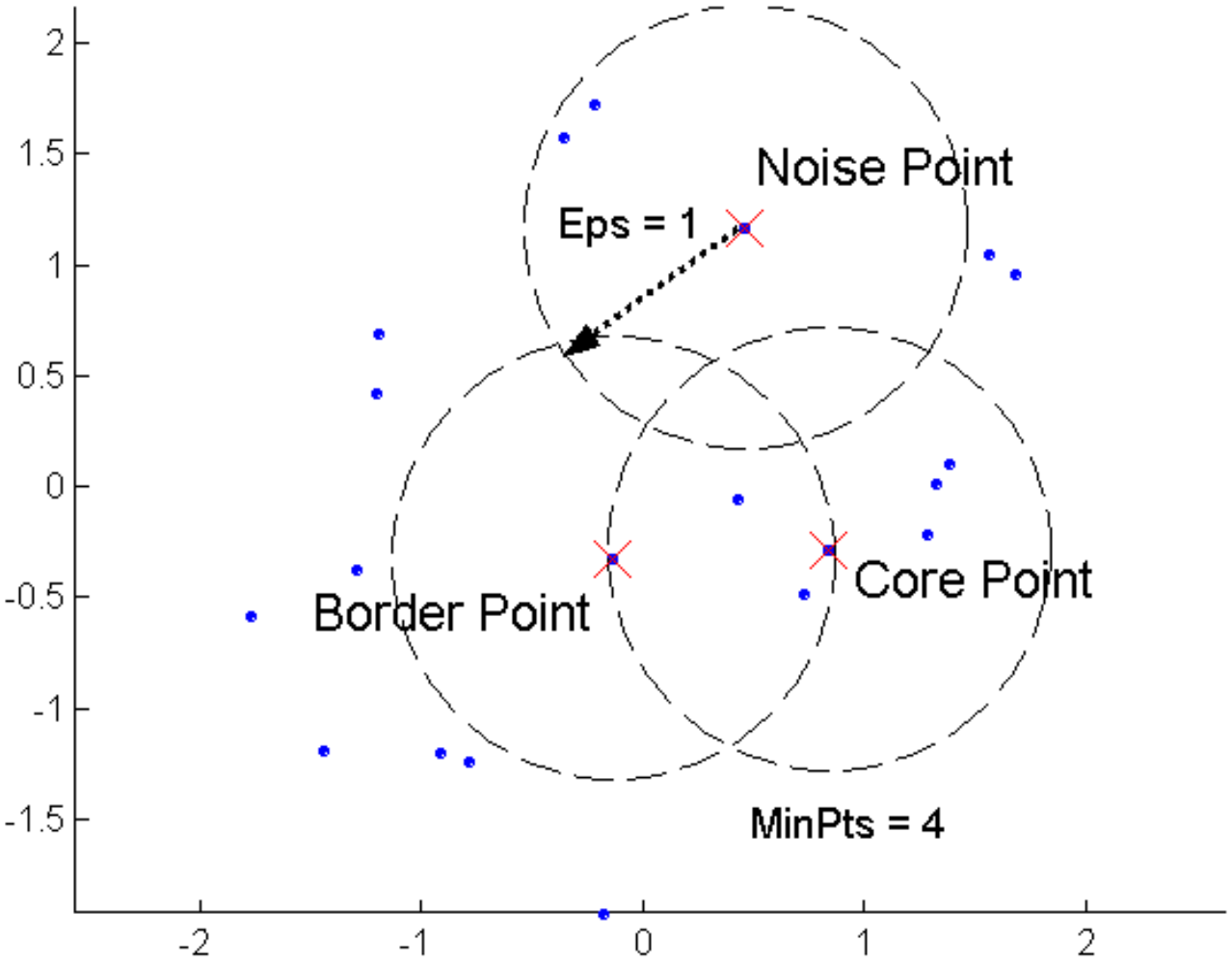
Linkage Methods



DBSCAN

DBSCAN

- DBSCAN is a density-based algorithm.
  - Density = number of points within a specified radius (Eps)
  - A point is a core point if it has more than a specified number of points (MinPts) within Eps
    - These are points that are at the interior of a cluster
  - A border point has fewer than MinPts within Eps, but is in the neighborhood of a core point
  - A noise point is any point that is not a core point or a border point.

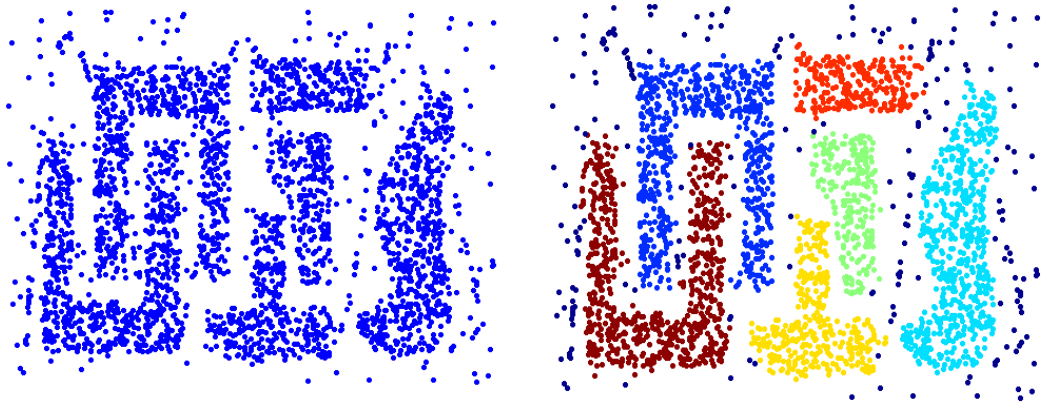


DBSCAN Algorithm

- Eliminate noise points
- Perform clustering on the remaining points
- https://youtu.be/_A9Tq6mGtLI?t=170
- https://www.naftaliharris.com/blog/visualizing-dbscan-clustering/

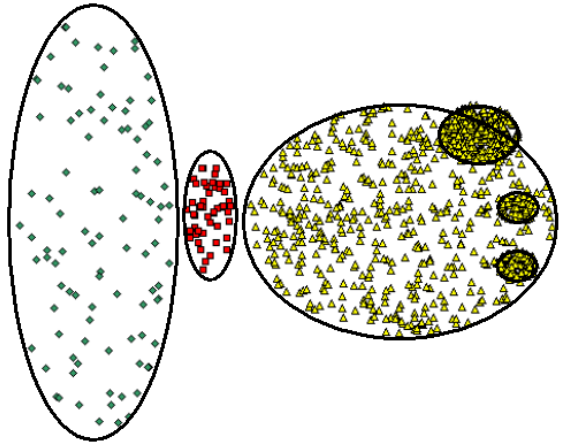
```
For every unvisited point P in the dataset:
    Mark P as visited.
    Find all neighbors within Epsilon (ε).

    If neighbors < Min_Points:
        Label P as NOISE.

    Else:
        Label P as a CORE point and create a NEW CLUSTER.
        Add P to the cluster.

        // The Expansion Queue
        For every neighbor N of P:
            If N was previously labeled NOISE:
                Relabel N as a BORDER point.
                Add N to the cluster.

            If N is unvisited:
                Mark N as visited.
                Find all neighbors of N within Epsilon (ε).

                If N's neighbors >= Min_Points:
                    Label N as a CORE point.
                    Add N's neighbors to our expansion queue!

                Add N to the cluster.
```

When DBSCAN Works Well/Poorly

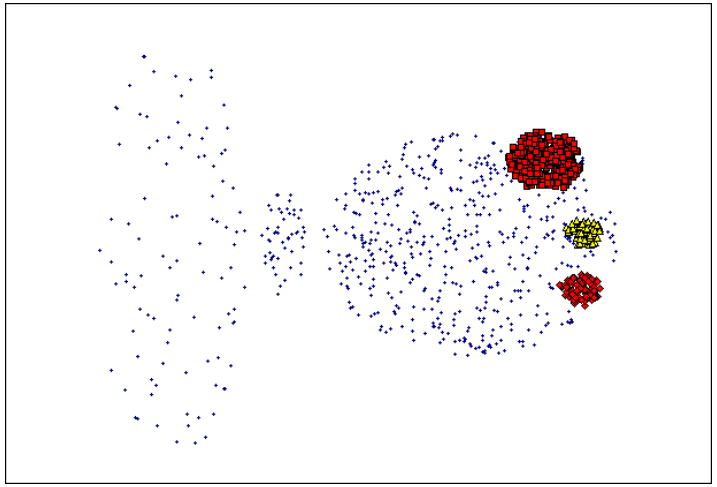


Original Points Clusters

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



Original Points



Silhouette

Silhouette

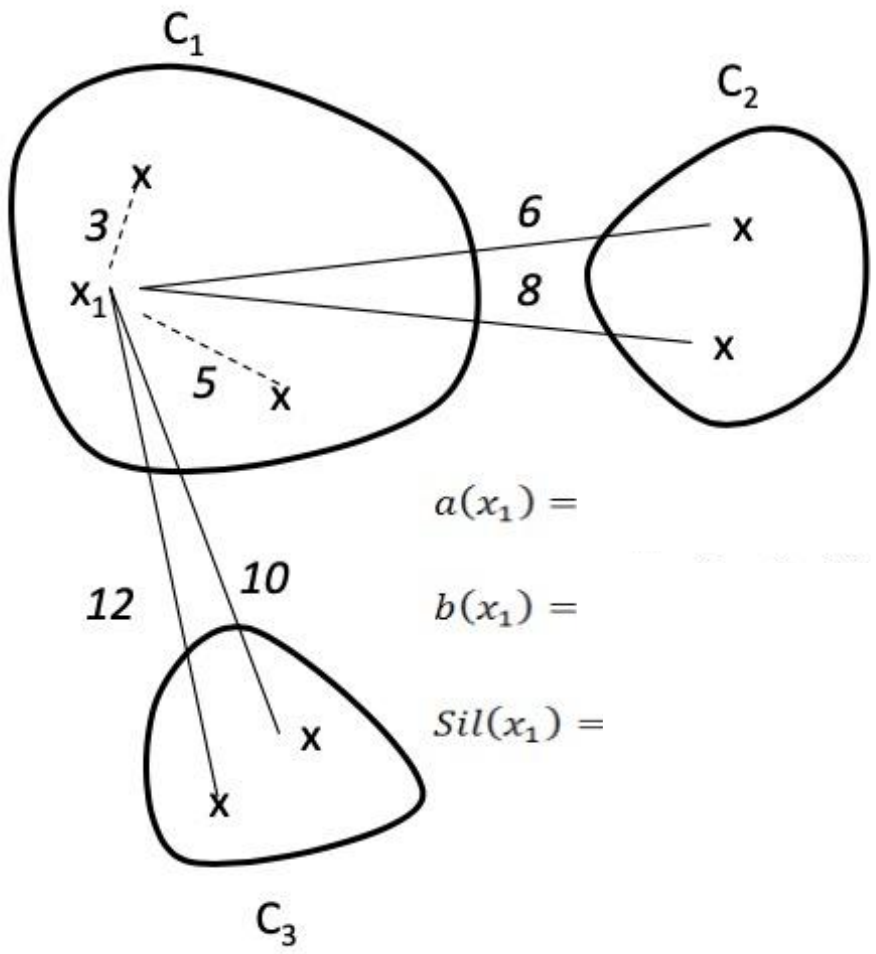
- We want techniques that find a balance between inter-cluster similarity and intra-cluster dissimilarity
- Silhouette is one good popular approach
- Scores any clustering with an arbitrary number of unique clusters. Clustering can come from any clustering algorithm.
- $a(i)$ = average dissimilarity of instance i to all other instances in the cluster to which i is assigned – Want it small
  - Dissimilarity could be Euclidian distance, etc.
- $b(i)$ = the smallest average dissimilarity of instance i to instances in other clusters – Want it large

Silhouette

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

***a*(*i*)** = average dissimilarity of instance *i* to all other instances in the cluster to which *i* is assigned
– Want it small

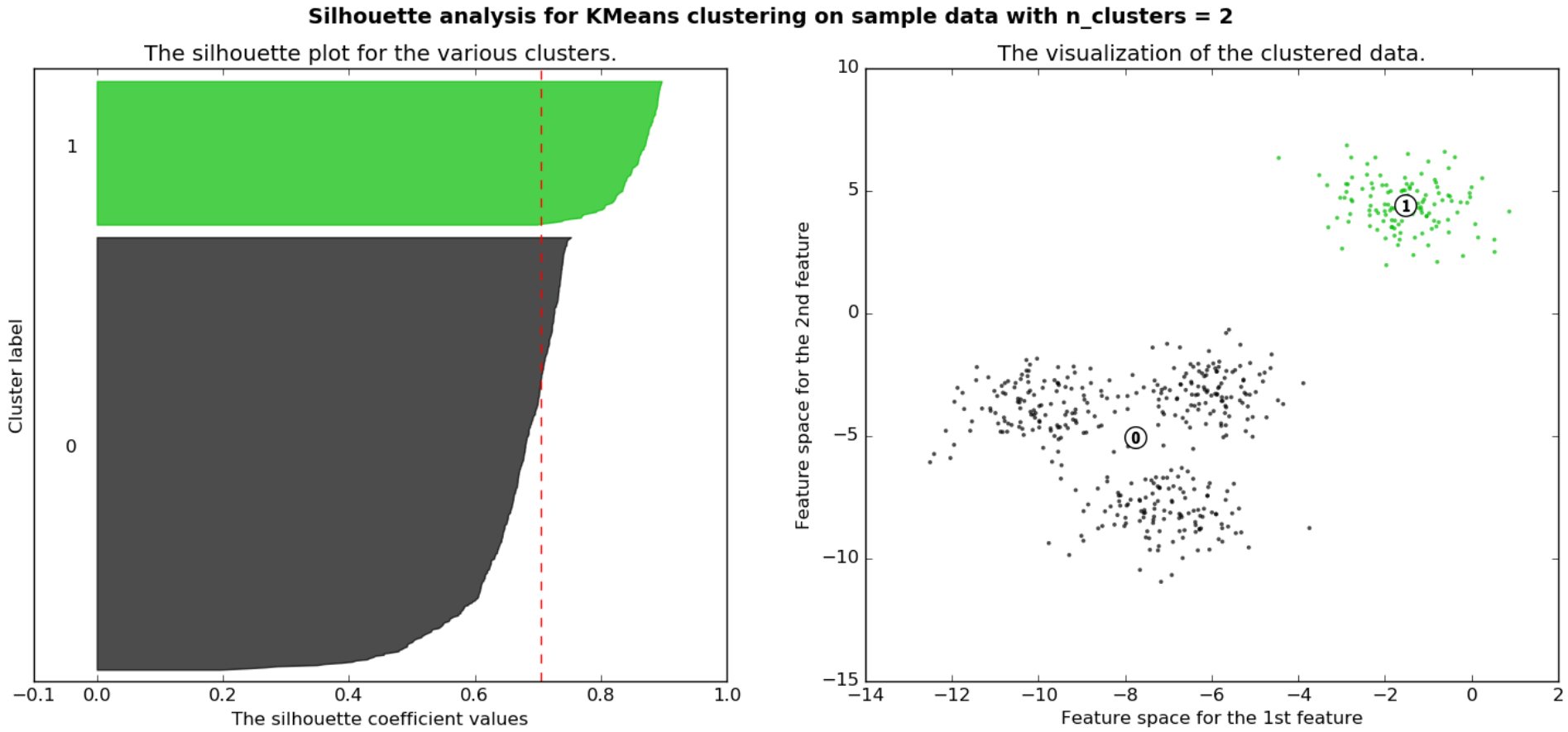
***b*(*i*)** = the smallest average dissimilarity of instance *i* to instances in other clusters
– Want it large



$$s(i) = \begin{cases} 1 - a(i)/b(i) & \text{if } a(i) < b(i) \\ 0 & \text{if } a(i) = b(i) \\ b(i)/a(i) - 1 & \text{if } a(i) > b(i) \end{cases}$$

- Silhouette of a cluster – average of the silhouette of all the points in the cluster
- Silhouette of a clustering - average of the silhouette of all the points in the clustering

Visualizing Silhouette



Width is quality of cluster, height is size of cluster
Dashed line is average silhouette score for clustering
Why does cluster 1 have a higher score?

Silhouette

- Could just use total silhouette average to decide best clustering but best to do silhouette analysis with a visualization tool and use score along with other aspects of the clustering
  - Cluster sizes
  - Number of clusters
  - Shape of clusters
  - Etc.
- Note when task dimensions are > 3 (typical and no longer visualizable for us), silhouette graph still easy to visualize
- O(n^2) complexity due to $b(i)$ computation
- There are other cluster metrics out there
- **These metrics are rough guidelines and should be "taken with a grain of salt"**