GENERAL ASSEMBLY

INTRO TO DATA SCIENCE

K-MEANS CLUSTERING

I. CLUSTER ANALYSIS
II. K-MEANS CLUSTERING
III. INTERPRETING RESULTS

EXERCISE:
IV. K-MEANS CLUSTERING

I. CLUSTER ANALYSIS

	continuous	*categorical*
supervised	???	???
unsupervised	???	???

	continuous	*categorical*
supervised	regression	classification
unsupervised	dimension reduction	clustering

Q: What is a cluster?

Q: What is a cluster?

A: A group of **similar** data points.

Q: What is a cluster?

A: A group of **similar** data points.

The concept of similarity is central to the definition of a cluster, and therefore to cluster analysis.

Q: What is a cluster?

A: A group of **similar** data points.

The concept of similarity is central to the definition of a cluster, and therefore to cluster analysis.

In general, greater similarity between points leads to better clustering.

Q: What is the purpose of cluster analysis?

Q: What is the purpose of cluster analysis?

A: To enhance our understanding of a dataset by dividing the data into groups.

Q: What is the purpose of cluster analysis?

A: To enhance our understanding of a dataset by dividing the data into groups.

Clustering provides a *layer of abstraction* from individual data points.

Q: What is the purpose of cluster analysis?

A: To enhance our understanding of a dataset by dividing the data into groups.

Clustering provides a *layer of abstraction* from individual data points.

The goal is to extract and enhance the natural structure of the data (not to impose arbitrary structure!)

Q: How do you solve a clustering problem?

Q: How do you solve a clustering problem?

A: Think of a cluster as a “potential class”; then the solution to a clustering problem is to programatically determine these classes.

Q: How do you solve a clustering problem?

A: Think of a cluster as a “potential class”; then the solution to a clustering problem is to programatically determine these classes.

The real purpose of clustering can be data exploration, so a solution is anything that contributes to your understanding.

II. K-MEANS CLUSTERING

Q: What is k-means clustering?

Q: What is k-means clustering?

A: A **greedy** learner that **partitions** a data set into k clusters.

Q: What is k-means clustering?

A: A **greedy** learner that **partitions** a data set into k clusters.

greedy – captures local structure (depends on initial conditions)

Q: What is k-means clustering?

A: A **greedy** learner that **partitions** a data set into k clusters.

greedy – captures local structure (depends on initial conditions)

partition – performs complete clustering (each point belongs to exactly one cluster)

Q: How are these partitions determined?

Q: How are these partitions determined?

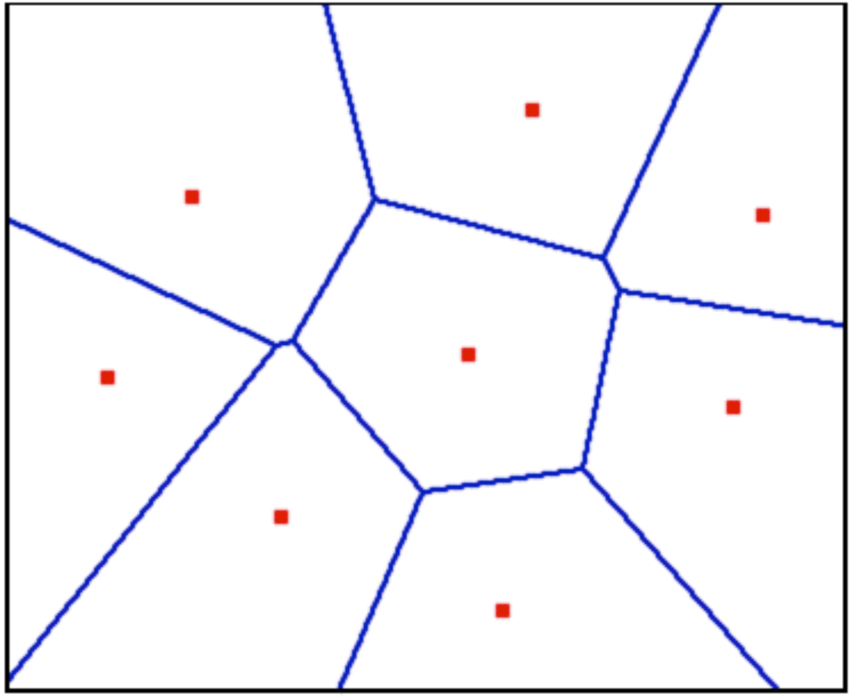
A: Each point is assigned to the cluster with the nearest **centroid**.

Q: How are these partitions determined?

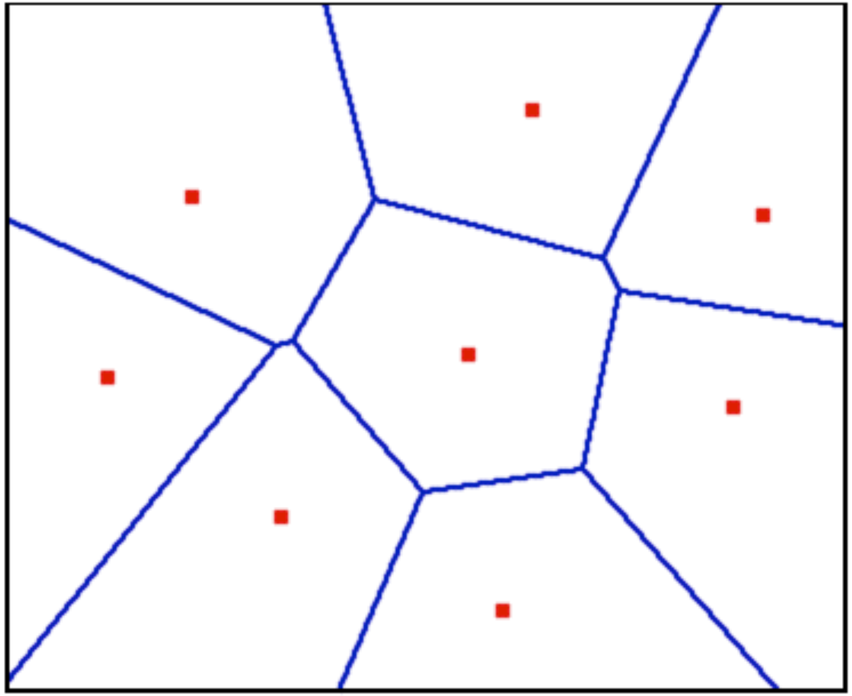
A: Each point is assigned to the cluster with the nearest **centroid**.

centroid – the mean of the data points in a cluster
→ requires continuous (vector-like) features
→ highlights iterative nature of algorithm

Q: What do these partitions look like?



source: http://www.dma.fi.upm.es/mabellanas/tfcs/fvd/images/voronoi.gif



NOTE

These partitions are sometimes called *Voronoi cells*, and these maps *Voronoi diagrams*.

source: http://www.dma.fi.upm.es/mabellanas/tfcs/fvd/images/voronoi.gif

One important point to keep in mind is that partitions are not scale-invariant!

One important point to keep in mind is that partitions are not scale-invariant!

This means that the same data can yield very different clustering results depending on the scale and the units used.

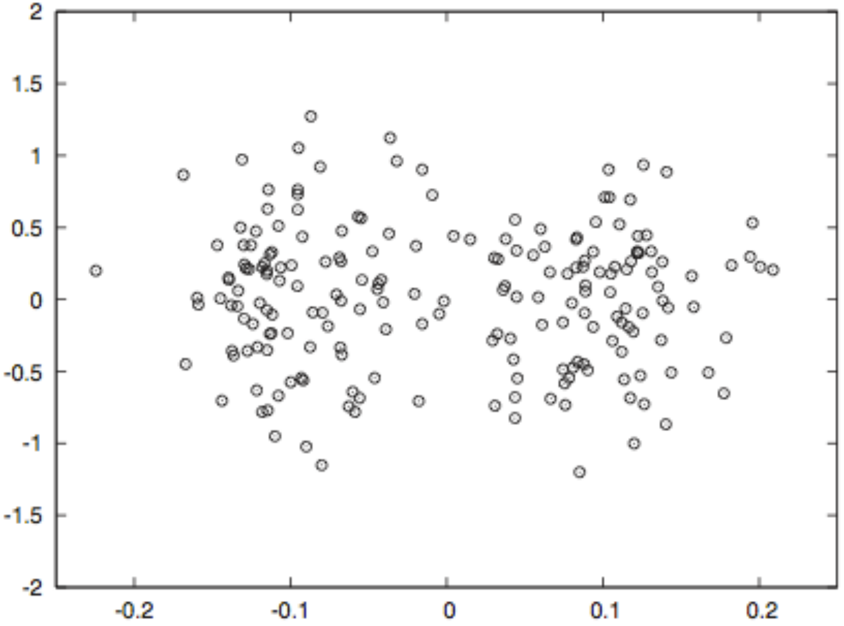
One important point to keep in mind is that partitions are not scale-invariant!

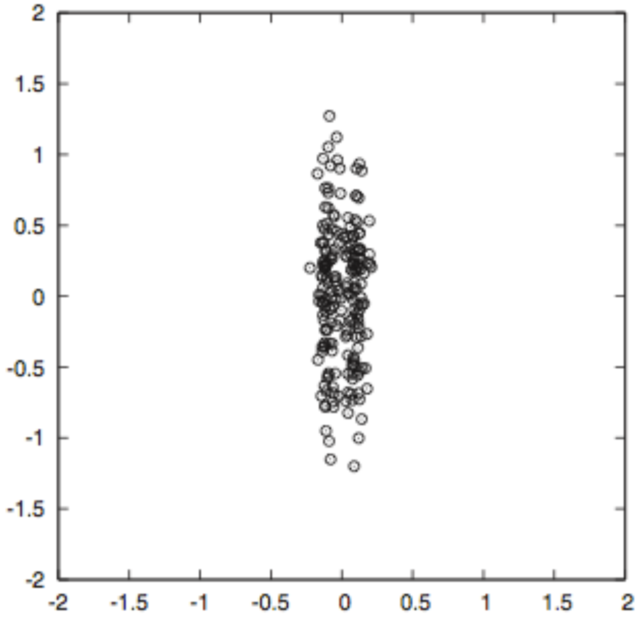
This means that the same data can yield very different clustering results depending on the scale and the units used.

Therefore it’s important to think about your data representation before applying a clustering algorithm.

These graphs show two different representations of the same data:

These graphs show two different representations of the same data:





source: Data Analysis with Open Source Tools, by Philipp K. Janert. O'Reilly Media, 2011.

1) choose k initial centroids (note that k is an input)

2) for each point:
   - find distance to each centroid
   - assign point to nearest centroid

3) recalculate centroid positions
4) repeat steps 2-3 until stopping criteria met

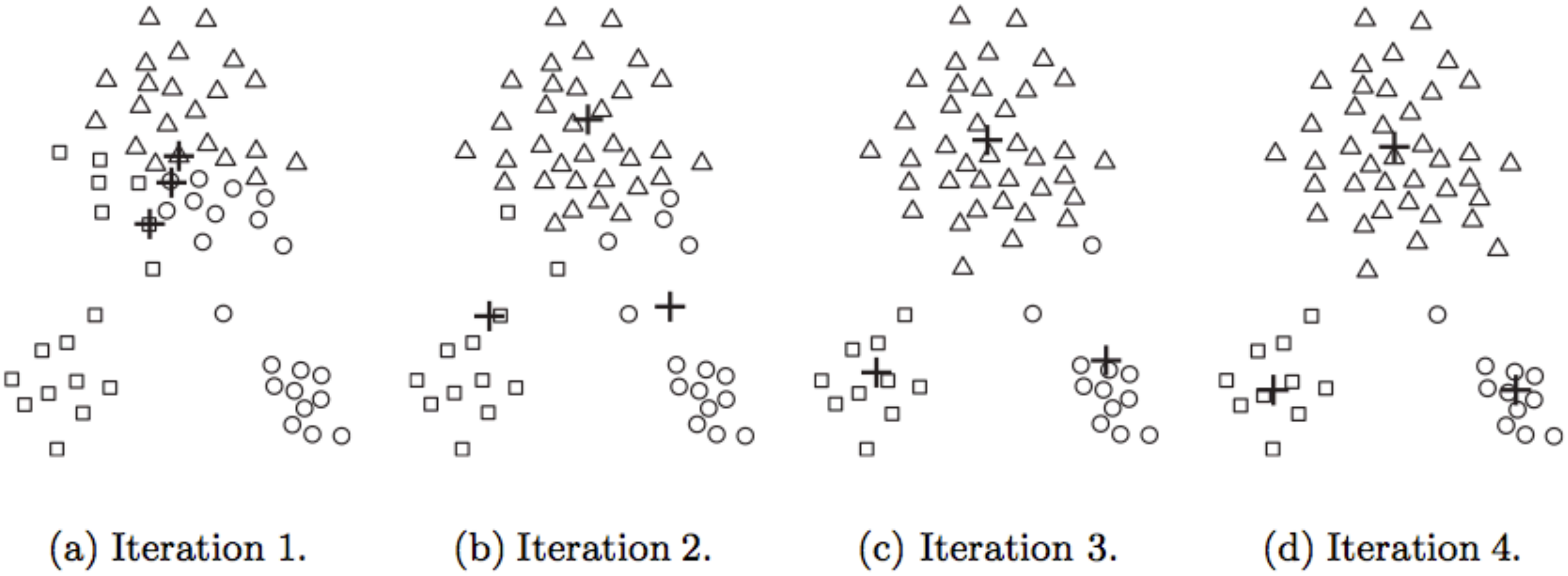


Figure 8.3. Using the K-means algorithm to find three clusters in sample data.

source: http://www-users.cs.umn.edu/~kumar/dmbook/ch8.pdf

K-means is algorithmically pretty elegant, and has performance linear in the number of data points in practice.

K-means is algorithmically pretty elegant, and has performance linear in the number of data points in practice.

It has a hard time dealing with *non-convex* clusters, or data with widely varying shapes and densities.

K-means is algorithmically pretty elegant, and has performance linear in the number of data points in practice.

It has a hard time dealing with *non-convex* clusters, or data with widely varying shapes and densities.

Difficulties can sometimes be overcome by increasing the value of k and combining subclusters in a post-processing step.

Q: How do you choose the initial centroid positions?

Q: How do you choose the initial centroid positions?

A: There are several options:

Q: How do you choose the initial centroid positions?

A: There are several options:
- randomly (but may yield divergent behavior)

Q: How do you choose the initial centroid positions?

A: There are several options:

- randomly (but may yield divergent behavior)
- perform alternative clustering task, use resulting centroids as initial k-means centroids

Q: How do you choose the initial centroid positions?

A: There are several options:

- randomly (but may yield divergent behavior)
- perform alternative clustering task, use resulting centroids as initial k-means centroids
- start with global centroid, choose point at max distance, repeat (but might select outlier)

Q: How do you determine which centroid is the nearest?

Q: How do you determine which centroid is the nearest?

The “nearness” criterion is determined by the similarity/distance measure we discussed earlier.

Q: How do you determine which centroid is the nearest?

The "nearness" criterion is determined by the similarity/distance measure we discussed earlier.

This measure makes quantitative inference possible.

Q: How do you determine which centroid is the nearest?

The “nearness” criterion is determined by the similarity/distance measure we discussed earlier.

This measure makes quantitative inference possible.

NOTE

Technically, by defining a similarity measure we are mapping our observations into a *metric space*.

A similarity measure must satisfy certain general conditions:

A similarity measure must satisfy certain general conditions:

$$d(x, y) \geq 0$$

$$d(x, y) = 0 \iff x = y$$

$$d(x, y) = d(y, x)$$ (symmetry)

$$d(x, y) + d(y, z) \geq d(x, z)$$ (triangle inequality)

A similarity measure must satisfy certain general conditions:

$$d(x, y) \geq 0$$

$$d(x, y) = 0 \iff x = y$$

$$d(x, y) = d(y, x)$$ (symmetry)

$$d(x, y) + d(y, z) \geq d(x, z)$$ (triangle inequality)

NOTE

Another useful property is *smoothness.*

There are a number of different similarity measures to choose from, and in general the right choice depends on the problem.

There are a number of different similarity measures to choose from, and in general the right choice depends on the problem.

For data that takes values in R^n, the typical choice is the **Euclidean distance**:

$$d(x, y) = \sqrt{\Sigma(x_i - y_i)^2}$$

There are a number of different similarity measures to choose from, and in general the right choice depends on the problem.

For data that takes values in R^n, the typical choice is the **Euclidean distance**:

$$d(x, y) = \sqrt{\Sigma(x_i - y_i)^2}$$

We can express different semantics about our data through the choice of metric.

Ex: One popular metric for text mining problems (or any problem with *sparse binary* data) is the **Jaccard coefficient**,

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

Ex: One popular metric for text mining problems (or any problem with *sparse binary* data) is the **Jaccard coefficient**,

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

Applying this metric to a problem expresses the sparse nature of the data, and makes a variety of text mining techniques accessible.

The matrix whose entries D_{ij} contain the values $d(x,\ y)$ for all x and y is called the **distance matrix**.

The matrix whose entries D_{ij} contain the values $d(x,\ y)$ for all x and y is called the **distance matrix**.

The distance matrix contains *all of the information* we know about the dataset as far as clustering is concerned.

The matrix whose entries D_{ij} contain the values $d(x, y)$ for all x and y is called the **distance matrix**.

The distance matrix contains *all of the information* we know about the dataset as far as clustering is concerned.

For this reason, it's really the choice of metric that determines the definition of a cluster.

Q: How do we recompute the positions of the centroids at each iteration of the algorithm?

Q: How do we recompute the positions of the centroids at each iteration of the algorithm?

A: By optimizing an **objective function** that tells us how “good” the clustering is.

Q: How do we recompute the positions of the centroids at each iteration of the algorithm?

A: By optimizing an **objective function** that tells us how “good” the clustering is.

The iterative part of the algorithm (recomputing centroids and reassigning points to clusters) explicitly tries to minimize this objective function.

Ex: Using the Euclidean distance measure, one typical objective function is the **sum of squared errors** from each point *x* to its centroid c_i:

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} d(x, c_i)^2$$

Ex: Using the Euclidean distance measure, one typical objective function is the **sum of squared errors** from each point *x* to its centroid c_i:

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} d(x, c_i)^2$$

Given two clusterings, we will prefer the one with the lower SSE since this means the centroids have converged to better locations (a better local optimum).

We iterate until some stopping criteria are met; in general, suitable convergence is achieved in a small number of steps.

We iterate until some stopping criteria are met; in general, suitable convergence is achieved in a small number of steps.

Stopping criteria can be based on the centroids (eg, if positions change by no more than ε) or on the points (eg, if no more than $x\%$ change clusters between iterations).

We iterate until some stopping criteria are met; in general, suitable convergence is achieved in a small number of steps.

Stopping criteria can be based on the centroids (eg, if positions change by no more than ε) or on the points (eg, if no more than $x\%$ change clusters between iterations).

Recall that, in general, different runs of the algorithm will converge to different local optima (centroid configurations).

III. CLUSTER VALIDATION

In general, k-means will converge to a solution and return a partition of k clusters, even if no natural clusters exist in the data.

In general, k-means will converge to a solution and return a partition of k clusters, even if no natural clusters exist in the data.

We will look at two validation metrics useful for partitional clustering, **cohesion** and **separation**.

Cohesion measures clustering effectiveness within a cluster.

$$\hat{C}(C_i) = \sum_{x \in C_i} d(x, c_i)$$

Cohesion measures clustering effectiveness within a cluster.

$$\hat{C}(C_i) = \sum_{x \in C_i} d(x, c_i)$$

Separation measures clustering effectiveness between clusters.

$$\hat{S}(C_i, C_j) = d(c_i, c_j)$$

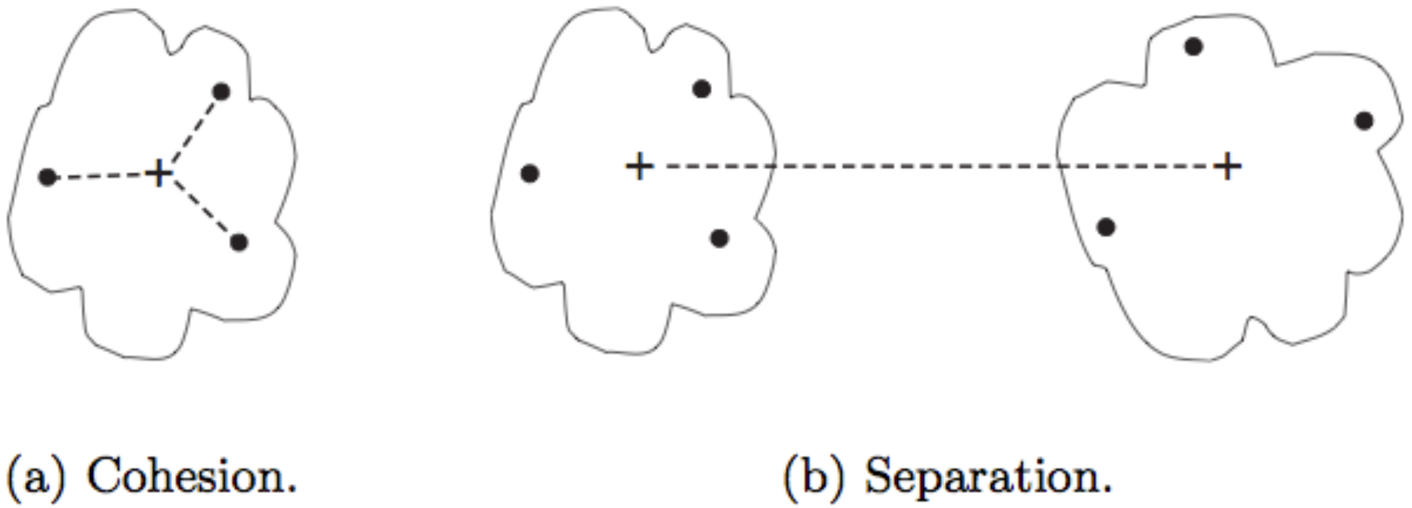


Figure 8.28. Prototype-based view of cluster cohesion and separation.

source: http://www-users.cs.umn.edu/~kumar/dmbook/ch8.pdf

We can turn these values into overall measures of clustering validity by taking a weighted sum over clusters:

$$\hat{V}_{total} = \sum_{1}^{K} w_i \hat{V}(C_i)$$

Here *V* can be cohesion, separation, or some function of both.

We can turn these values into overall measures of clustering validity by taking a weighted sum over clusters:

$$\hat{V}_{total} = \sum_{1}^{K} w_i \hat{V}(C_i)$$

Here *V* can be cohesion, separation, or some function of both.

The weights can all be set to 1 (best for k-means), or proportional to the cluster *masses* (the number of points they contain).

Cluster validation measures can be used to identify clusters that should be split or merged, or to identify individual points with disproportionate effect on the overall clustering.

One useful measure than combines the ideas of cohesion and separation is the **silhouette coefficient**. For point x_i, this is given by:

$$SC_i = \frac{b_i - a_i}{max(a_i, b_i)}$$

such that:

a_i = average in-cluster distance to x_i

b_{ij} = average between-cluster distance to x_i

$b_i = min_j(b_{ij})$

The silhouette coefficient can take values between -1 and 1.

In general, we want separation to be high and cohesion to be low. This corresponds to a value of SC close to +1.

A negative silhouette coefficient means the cluster radius is larger than the space between clusters, and thus clusters overlap.

SILHOUETTE COEFFICIENT

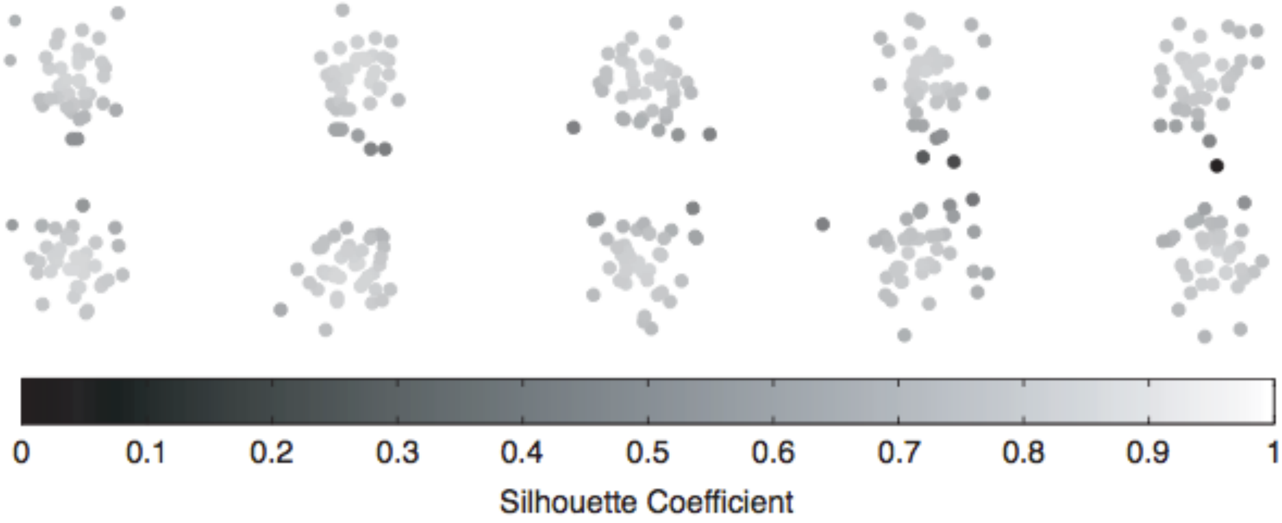


Figure 8.29. Silhouette coefficients for points in ten clusters.

source: http://www-users.cs.umn.edu/~kumar/dmbook/ch8.pdf

The silhouette coefficient for the cluster C_i is given by the average silhouette coefficient across all points in C_i:

$$SC(C_i) = \frac{1}{m_i} \sum_{x \in C_i} SC_i$$

The silhouette coefficient for the cluster C_i is given by the average silhouette coefficient across all points in C_i:

$$SC(C_i) = \frac{1}{m_i} \sum_{x \in C_i} SC_i$$

The overall silhouette coefficient is given by the average silhouette coefficient across all points:

$$SC_{total} = \frac{1}{k} \sum_{1}^{k} SC(C_i)$$

The silhouette coefficient for the cluster C_i is given by the average silhouette coefficient across all points in C_i:

$$SC(C_i) = \frac{1}{m_i} \sum_{x \in C_i} SC_i$$

The overall silhouette coefficient is given by the average silhouette coefficient across all points:

$$SC_{total} = \frac{1}{k} \sum_{1}^{k} SC(C_i)$$

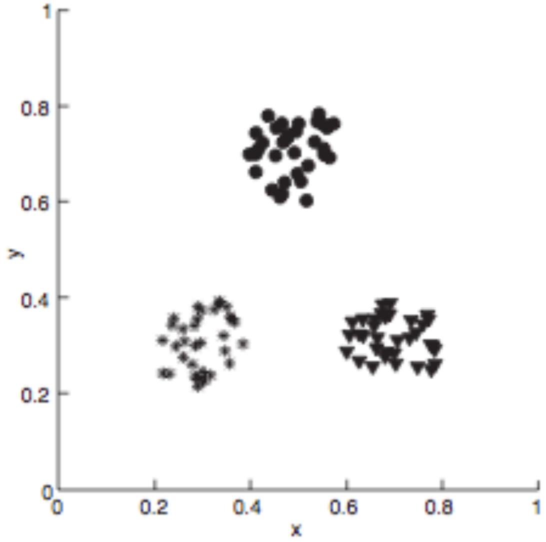
NOTE

This gives a summary measure of the overall clustering quality.

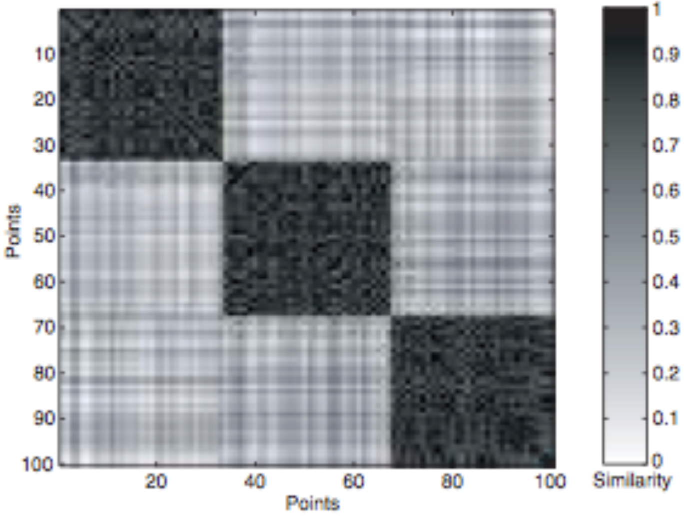
An alternative validation scheme is given by comparing the similarity matrix with an idealized (0/1) similarity matrix that represents the same clustering configuration.

An alternative validation scheme is given by comparing the similarity matrix with an idealized (0/1) similarity matrix that represents the same clustering configuration.

This can be done either graphically or using correlations.



(a) Well-separated clusters.



(b) Similarity matrix sorted by K-means cluster labels.

source: http://www-users.cs.umn.edu/~kumar/dmbook/ch8.pdf

One useful application of cluster validation is to determine the best number of clusters for your dataset.

One useful application of cluster validation is to determine the best number of clusters for your dataset.

Q: How would you do this?

One useful application of cluster validation is to determine the best number of clusters for your dataset.

Q: How would you do this?

A: By computing the overall SSE or SC for different values of k.

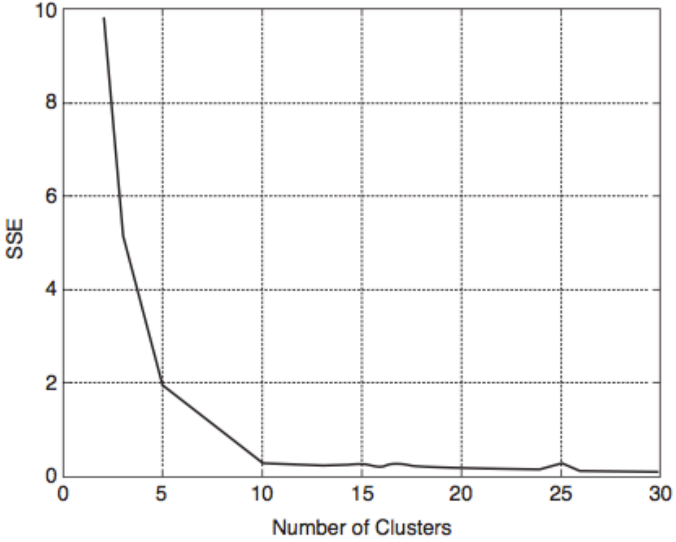


Figure 8.32. SSE versus number of clusters for the data of Figure 8.29.

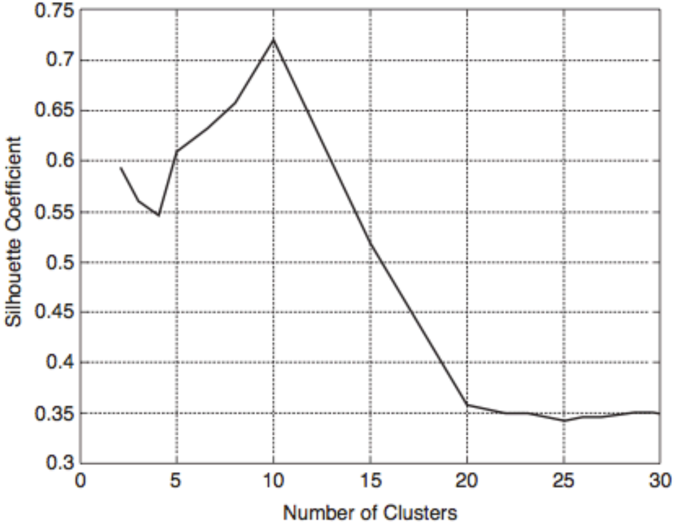


Figure 8.33. Average silhouette coefficient versus number of clusters for the data of Figure 8.29.

source: http://www-users.cs.umn.edu/~kumar/dmbook/ch8.pdf

Q: How can you determine your level of confidence in these validation metrics?

Q: How can you determine your level of confidence in these validation metrics?

A: Statistically; e.g., by computing frequency distributions for these metrics (over several runs of the algorithm) and determining statistical significance.

Ultimately, cluster validation and clustering in general are suggestive techniques that rely on human interpretation to be meaningful.

EX: K-MEANS CLUSTERING