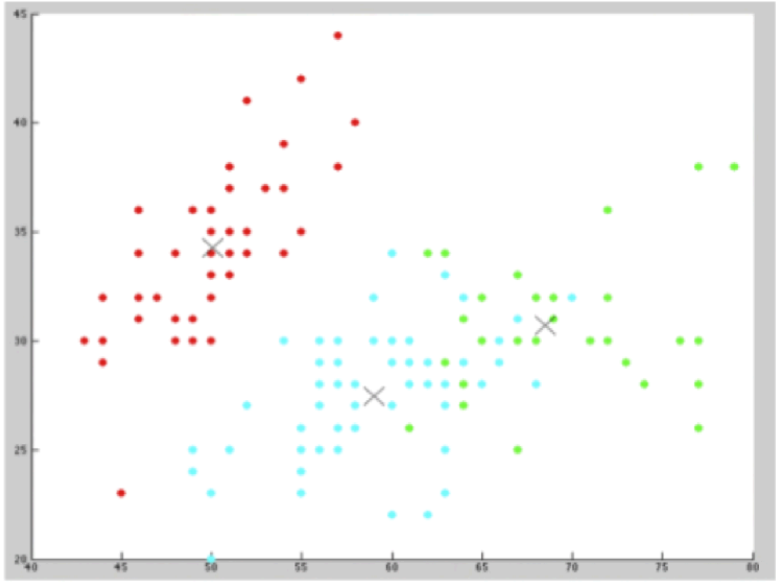
Ciro Donalek (Caltech)

Clustering: k-means

Sponsored by the Jet Propulsion Laboratory and by the Keck Institute for Space Studies, California Institute of Technology

K-Means

- Unsupervised clustering method
  - partitioning algorithm
- Partition the data into k clusters, based on their features and distance from the centroids.
- Each cluster is defined by its centroid
- Goal: minimize the sum of the squared errors (SSE).



Centroid and Medoids

- Centroid: the "middle" of a cluster

$$C_m = \frac{\sum_{i=1}^{N} (t_{ip})}{N}$$

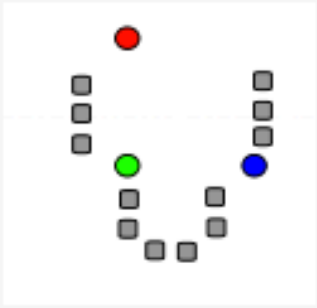
- Medoid: one chosen, centrally located object in the cluster
- Distance between clusters based on centroids: dis(Ki, Kj) = dis(Ci, Cj)
- Distance between clusters based on medoids dis(Ki, Kj) = dis(Mi, Mj)

SSE

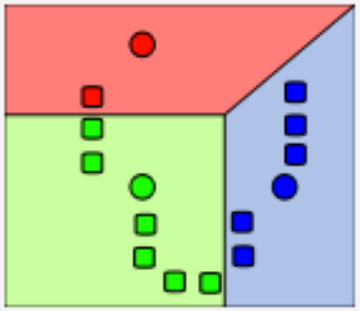
$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

- where **x** is a data point in cluster C_i, $\mathbf{m}_i$ is the centroid or medoid;
- given two partitions choose the one with the smallest error.

How it works

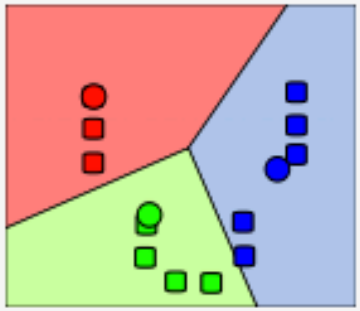


1) *k* initial "means" (in this case *k*=3) are randomly generated within the data domain (shown in color).



2) *k* clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.

3) The centroid of each of the *k* clusters becomes the new mean.

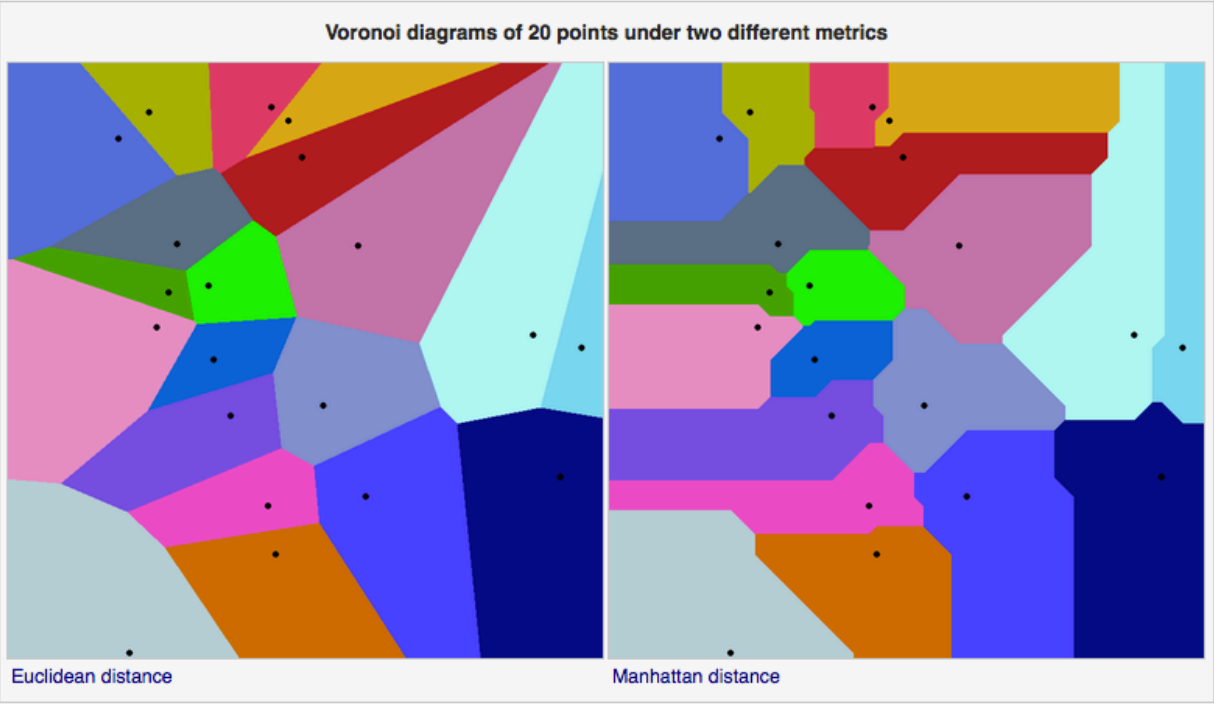


4) Steps 2 and 3 are repeated until convergence has been reached.

Image credits: wikipedia

Quick recap: Voronoi diagram

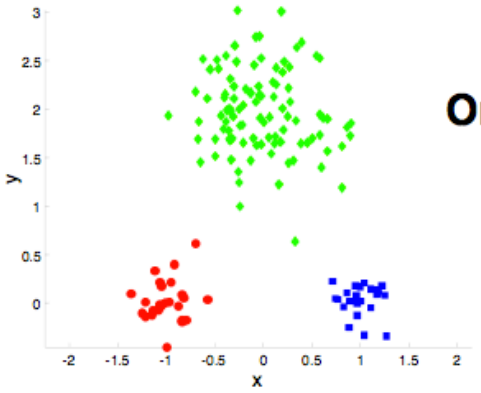
- A Voronoi tessellation is a way of dividing space into a number of regions.
- Choose the seeds.
- Create region consisting of all point closer to that seed that any other.

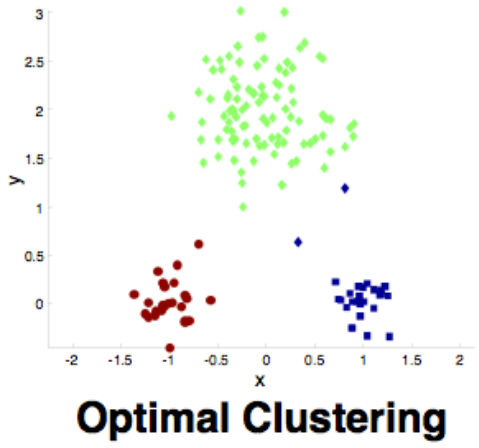


Credit: Wikipedia

Different k-means runs

- Results may vary depending on the initialization.





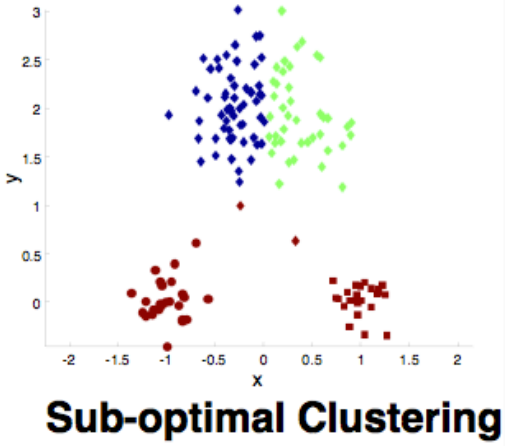


Image credit: Matteo Pardo

Convergence

- There is no guarantee that it will converge to the global optimum.
- Early termination criteria:
  - fixed number of iterations;
  - centroids do not change;
  - assignment of samples to the cluster doesn't change.
- Result may depend on the initial clusters.

Initialization

- Initialization:
  - place the first centroid on a data point, the second centroid on a data point that is far away as possible from the first one and so on...
  - select more than k centroids and then select the k most widely separated.
- Replace means with modes for categorical variables

Variations: distances

- E.g., distances implemented in Matlab for k-Means

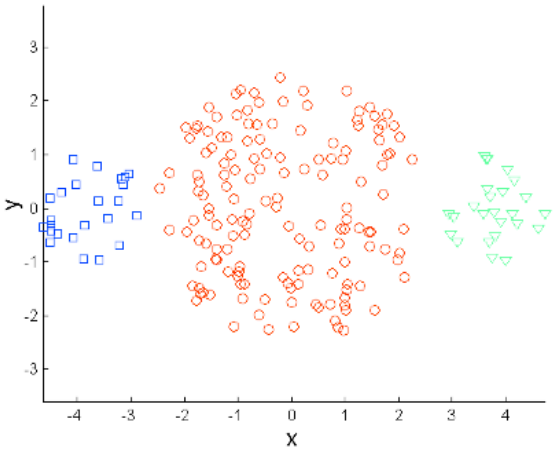
Distance measure, in p-dimensional space. `kmeans` minimizes with respect to this parameter. `kmeans` computes centroid clusters differently for the different supported distance measures.	
`'sqEuclidean'`	Squared Euclidean distance (default). Each centroid is the mean of the points in that cluster.
`'cityblock'`	Sum of absolute differences, i.e., the L1 distance. Each centroid is the component-wise median of the points in that cluster.
`'cosine'`	One minus the cosine of the included angle between points (treated as vectors). Each centroid is the mean of the points in that cluster, after normalizing those points to unit Euclidean length.
`'correlation'`	One minus the sample correlation between points (treated as sequences of values). Each centroid is the component-wise mean of the points in that cluster, after centering and normalizing those points to zero mean and unit standard deviation.
`'Hamming'`	Percentage of bits that differ (only suitable for binary data). Each centroid is the component-wise median of points in that cluster.

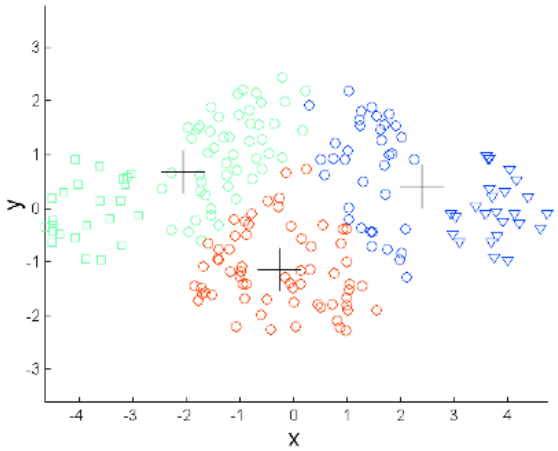
Other variations

'emptyaction'	Action to take if a cluster loses all its member observations.	
	'error'	Treat an empty cluster as an error (default).
	'drop'	Remove any clusters that become empty. kmeans sets the corresponding return values in C and D to NaN.
	'singleton'	Create a new cluster consisting of the one point furthest from its centroid.
'onlinephase'	Flag indicating whether kmeans should perform an online update phase in addition to a batch update phase. The online phase can be time consuming for large data sets, but guarantees a solution that is a local minimum of the distance criterion, that is, a partition of the data where moving any single point to a different cluster increases the total sum of distances.	
	'on'	Perform online update (default).
	'off'	Do not perform online update.

K-Means Pro and Cons

- Pro
  - Simple, use as benchmark
  - Relatively efficient: $O(t*k*n)$, where n is the number of objects, k is the number of clusters, and t is the number of iterations.
  - easy to run multiple time with different k
- Cons
  - unable to handle noisy data
  - need to specify k
  - problems with clusters of different sizes, non-globular shapes, clusters differing in density





Clusters

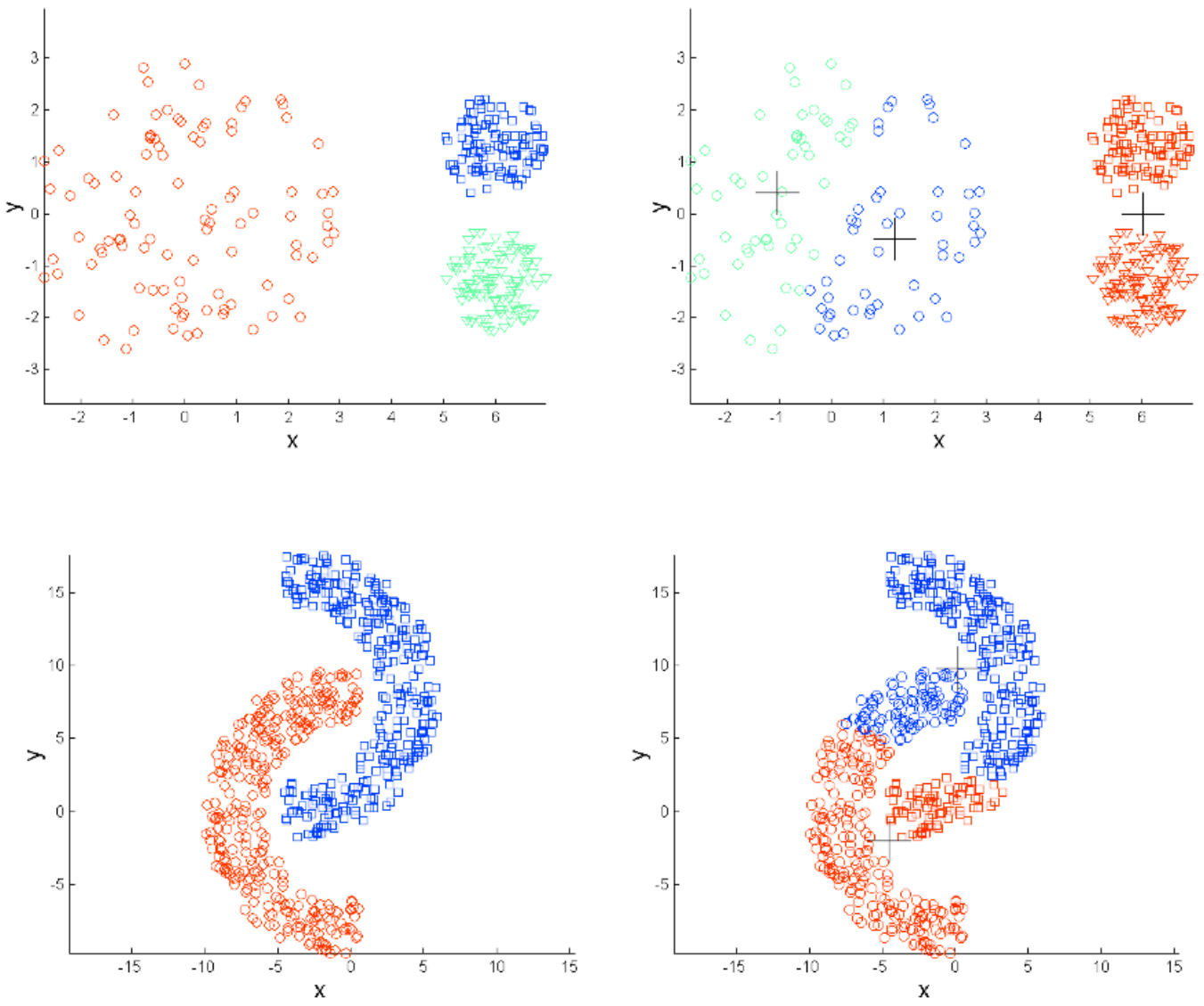


Image credit: Matteo Pardo

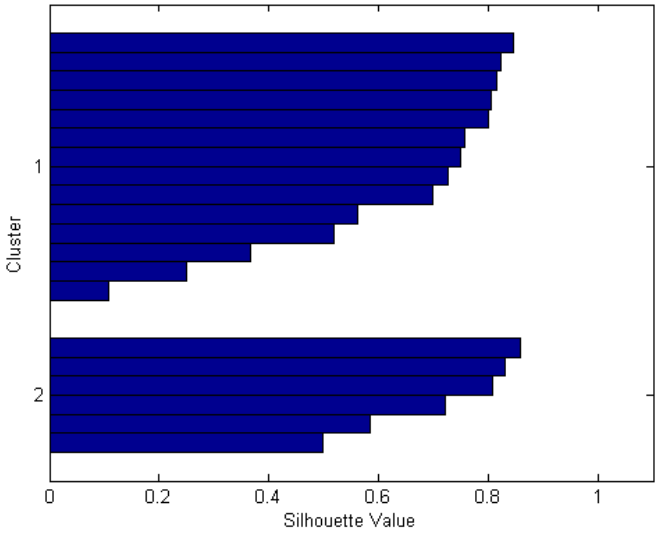
Pre and Post-processing

- Pre-processing:
  - feature selection;
  - normalize data;
  - remove outliers.
- Post-processing:
  - remove small clusters;
  - split clusters with high SSE;
  - merge clusters with low SSE.

Silhouette

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

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



where:

a(i) be the average dissimilarity of i with all other data within the same clusters;

b(i) be the lowest average dissimilarity of i to any other cluster which i is not member.

High silhouette value:
- i is well-matched to its own cluster
- i is poorly-matched to neighboring clusters.

If most points have a high silhouette value, then the clustering solution is appropriate.

If many points have a low or negative silhouette value, we may have either too many or too few clusters.

Learning K

- Find a balance between two variables: the number of clusters (**K**) and the average variance of the clusters.
- Minimize both values
- As the number of clusters increases, the average variance decreases (up to the trivial case of k=n and variance=0).

Criteria

- Information criteria approach
  - BIC (Bayesian Information Criteria)
  - AIC (Akaike Information Criteria)
  - DIC (Deviance Information Criteria)
- Calinski-Harabasz
- Davies-Bouldin Criterion

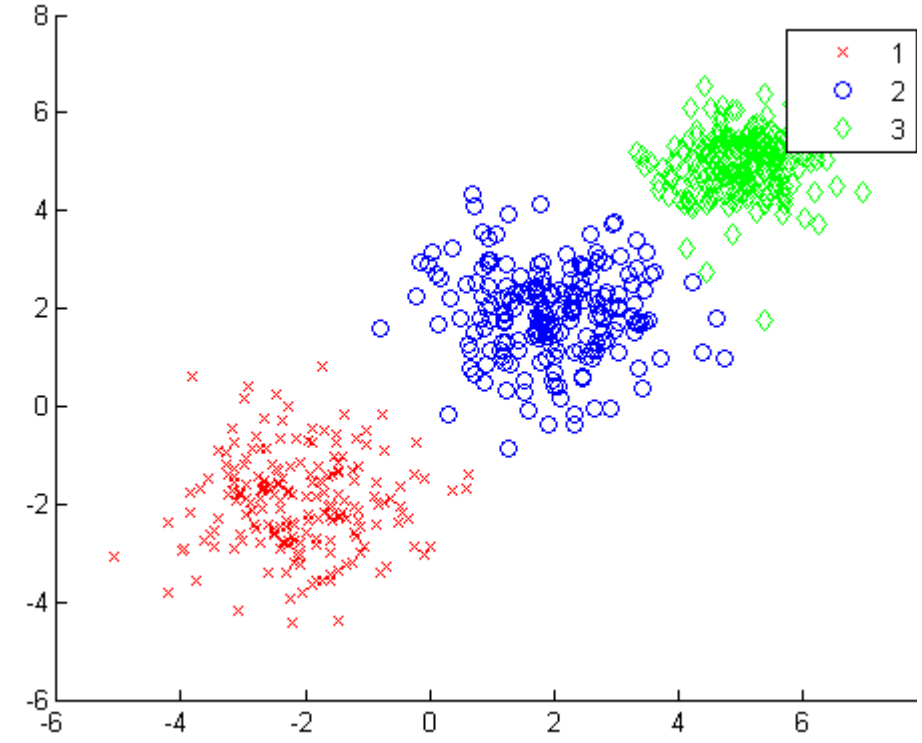
Davies-Bouldin Criterion

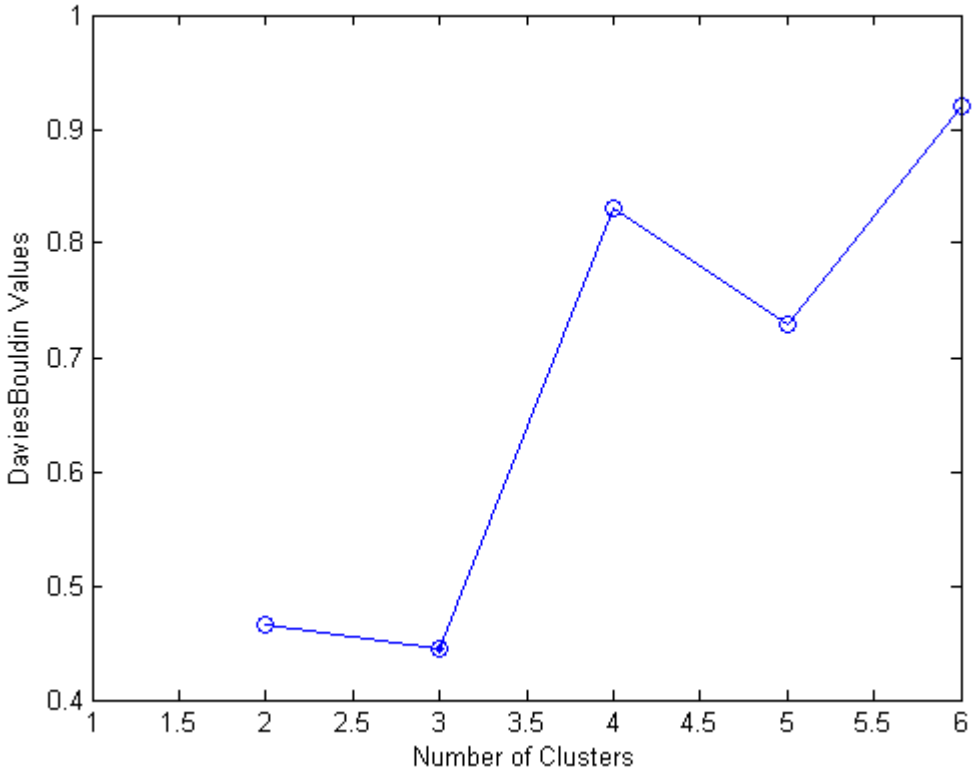
- Based on a ratio of inter-cluster and intra-cluster distances.
- The maximum value of $D_{i,j}$ represents the worst-case within-to-between cluster ratio for cluster *i*. The optimal clustering solution has the smallest Davies-Bouldin index value.

$$DB = \frac{1}{k}\sum_{i=1}^{k} \max_{j \neq i}\left\{D_{i,j}\right\}, \qquad D_{i,j} = \frac{\left(\bar{d}_i + \bar{d}_j\right)}{d_{i,j}}.$$

d_i is the average distance between each point in the i-th cluster and the centroid of the i-th cluster. d_j is the average distance between each point in the ith cluster and the centroid of the j-th cluster. d_{ij} is the Euclidean distance between the centroids of the i*th* and j*th* clusters.

Davies-Bouldin Criterion





Other alternatives

- K-Means is an example of Expectation-Maximization (EM) class of algorithms
- K-Medoids or PAM (Partitioning Around Models)
  - use medoids instead of centroids
- CLARA
  - draws multiple samples of the data set, applies PAM on each sample, and gives the best clustering as the output
- Fuzzy C-Means
  - every point has a degree of belonging to clusters

Summary

- K-Means: simple but effective.
- Pay attention to the shapes of the clusters.
- Choose
  - initialization;
  - empty sets;
  - distance metric;
  - k;
  - …
- Alternatives