Machine Learning I

Lecture 16: Cluster Analysis Part II

Nathaniel Bade

Northeastern University Department of Mathematics

Table of contents

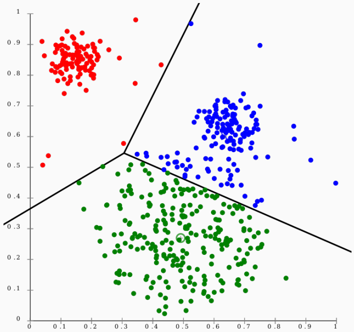
Cluster Analysis Review

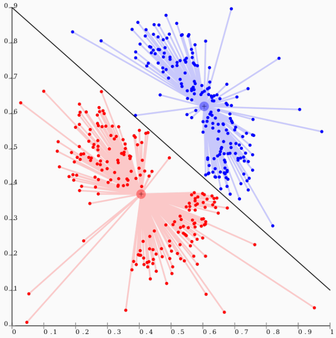
Clustering Paradigms

There are five basic kinds of clustering paradigms:

- **Centroid** based clustering is governed by overall distance to closest centroid.
- **Hierarchical** clustering build nested clusters by merging or splitting, forming a cluster **dendrogram**.
- **Density** based methods are determined by contiguous regions with density above some threshold.
- **Spectral** clustering decomposes the similarity graph $d(x_i, x_j)$ as in factor analysis and PCA.
- **Distribution models/mixing models** assume clusters provide a underlying distribution on the domain.

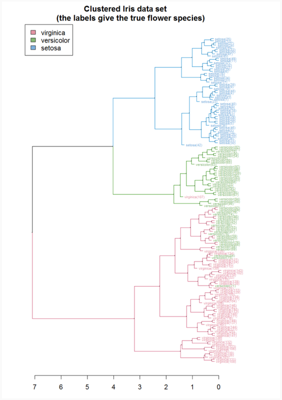
Centroid Based Clustering

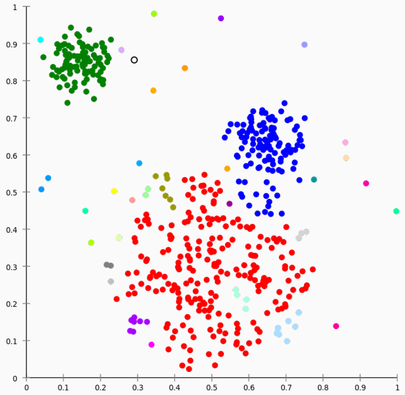




In centroid based clustering, clusters are determined by distance from a centroid μ_i, under a metric $d(x, y)$ of similarity matrix D. Examples include k-`means`, k-`mediods`, and the `meanshift` algorithm.

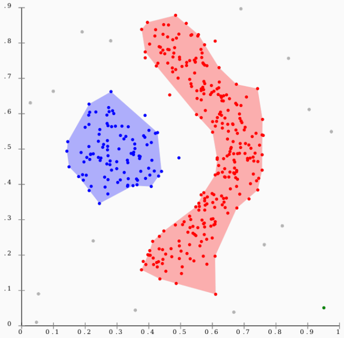
Hierarchical Clustering

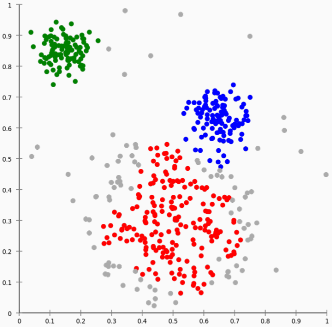




In hierarchical (connectivity, linkage) clustering, nested clusters are formed using greedy algorithms but either merging or splitting sets. Algorithms include `Ward` and `AgglomerativeClustering`.

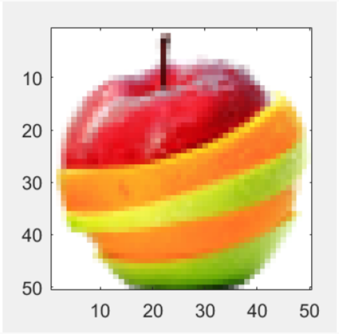
Density Based Clustering

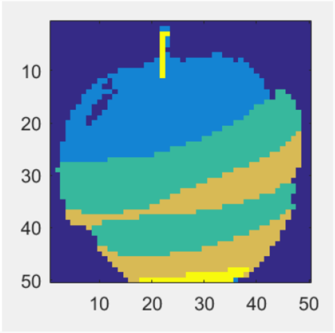




Density based clustering algorithms determine regions by applying a local density threshold (with a dynamically determined or fixed density) and then extracting the connected regions. Algorithms like DBSCAN on the right assume clusters are of similar density and have trouble separating nearby clusters. Algorithms like OPTICS improve upon this but are not currently implemented.

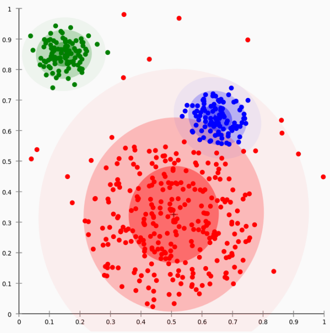
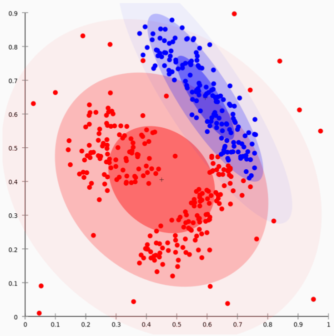
Spectral Clustering



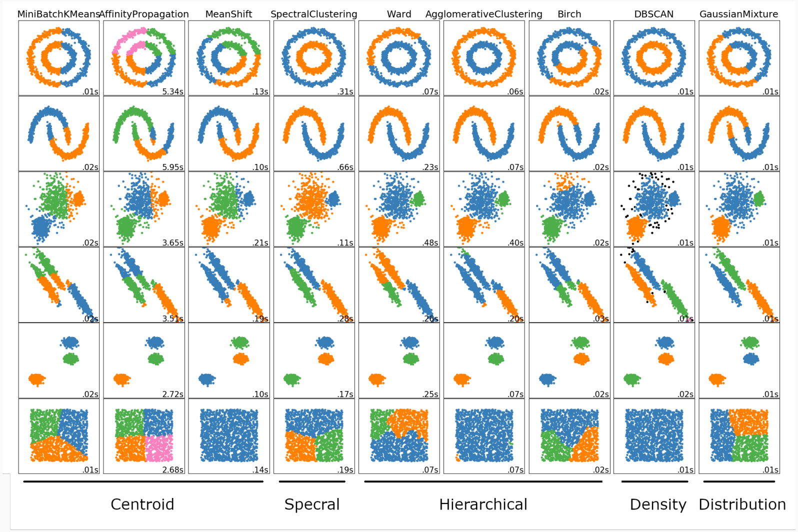


Spectral clustering forms a similarity matrix S_{ij} where each entry encode the similarity between datapoints i and j. We then project onto the first k eigenvectors of S_{ij} and perform the clustering there. Equivalently, we can construct a similarity graph and thing of pruning away low relevance connections until the graph disconnects. Sci-kit learn implements this as SpectralClustering.

Distribution Based Clustering

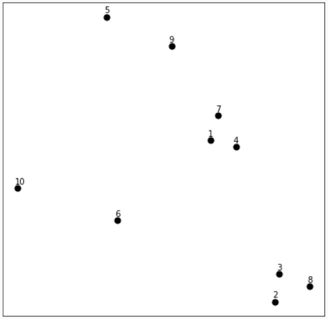
Distribution based, or mixing models, try to fit an underlying distribution to the data, often using maximum likelihood estimation. This is implemented in sci-kit learn as GaussianMixture.

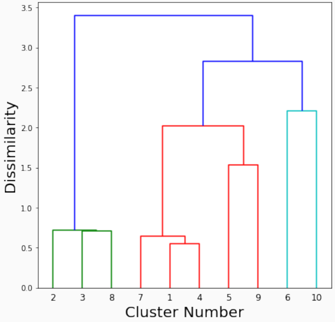


Sci-kit learn's clustering algorithms compared: `https://scikit-learn.org/stable/modules/clustering.html`

Hierarchical Clustering

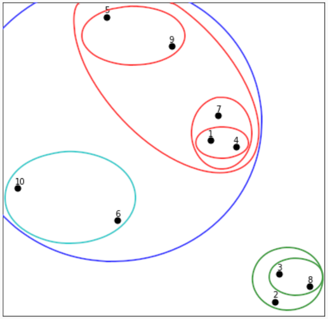
Hierarchical Clustering

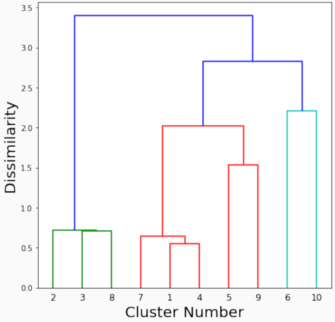




Hierarchical clustering algorithms work by building a dendrogram, where the clusters at each level are created by merging the clusters at the lower levels.

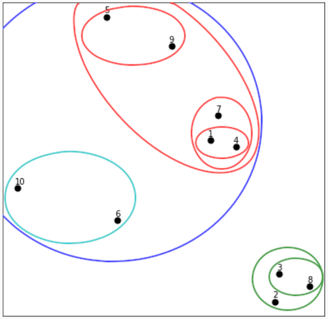
Hierarchical Clustering

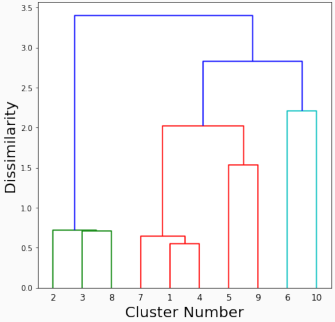




Hierarchical clustering algorithms work by building a dendrogram, where the clusters at each level are created by merging the clusters at the lower levels. This creates a hierarchy of clusters, starting from the discrete clustering and ending with a single cluster containing all the data. Choosing a clustering is then given by choosing a cut depth.

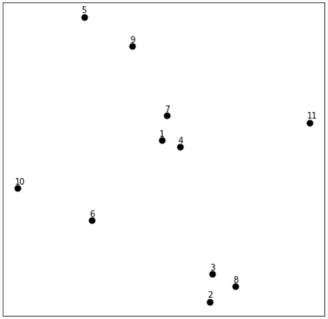
Hierarchical Clustering

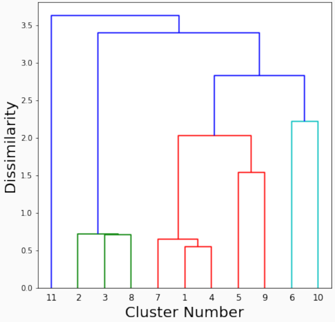




The height on a dendrogram is meaningful: it designates the total dissimilarity between clusters. We will discuss in moment how this can be calculated for clusters with multiple elements.

Hierarchical Clustering

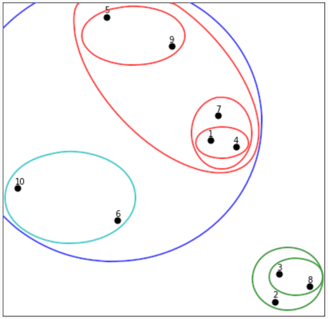


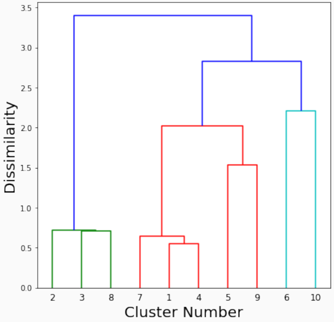


The height on a dendrogram is meaningful: it designates the total dissimilarity between clusters. We will discuss in moment how this can be calculated for clusters with multiple elements.

It is also possible to use the linkage diagram to detect outliers by looking for singleton clusters that are only merged in at a high level.

Hierarchical Clustering





Given a dissimilarity matrix $D_{i,i'}$, how do we construct the dendrogram? For dendrograms with n leaves, there are

$$\frac{(2n-3)!}{2^{(n-2)}(n-2)!} \geq (n-1)!$$

possible dendrograms, so an exhaustive search is out of the question.

Hierarchical Clustering

There are two main methods:

Bottom up/Agglomerative: Starting with all data points in their own discrete clusters, combine the closest cluster at each step until only one cluster remains.

Top down/Divisive: Starting with all data points in one cluster, consider every way of dividing the clusters (pairwise) and pick the best. Repeat until all clusters are singletons.

Agglomerative Clustering

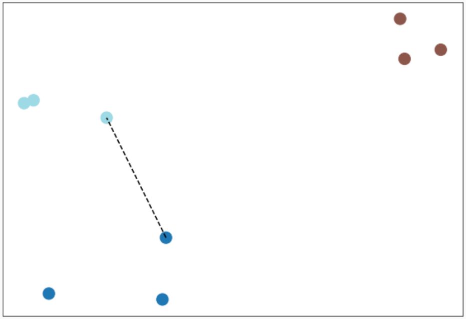
Starting with a data set $\mathcal{T}$ and a dissimilarity matrix $D_{i,i'}$ giving the total dissimilarity between each element of $\mathcal{T}$, we have to determine a dissimilarity between clusters. The most popular distance updates are given by specializations of the following formula:

If $d_{i,j} \in D_{i,i'}$ is the smallest element of D, then we merge clusters i and j into a cluster k. The dissimilarity between m and the new k are calculated by

$$d_{km} = \alpha_i d_{i,m} + \alpha_j d_{j,m} + \beta d_{i,j} + \gamma |d_{i,m} - d_{j,m}| \, ,$$

where α_i, α_j, β and γ are yet to be determined. Expressing a cluster distance this way is called the Lance-Williams Method.

Single Link Clustering

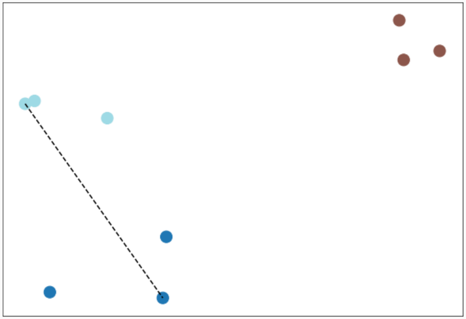


For **single link**, the distance between clusters is the distance between the closest points. The dissimilarity equation has $\alpha_i = \alpha_j = .5$, $\beta = 0$ and $\gamma = -.5$. The cluster dissimilarity is

$$d_{km} = \frac{1}{2}(d_{i,m} + d_{j,m} - |d_{i,m} - d_{j,m}|) = \min_{x \in C_i, x \in C_j} D(x, x')\,,$$

which can be shown to be the shortest distance by induction.

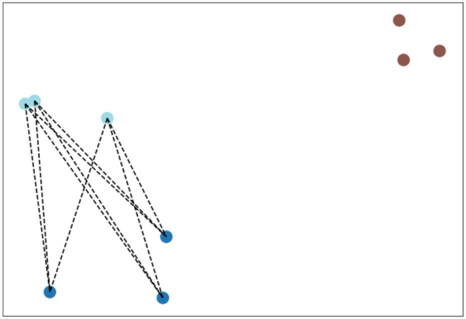
Complete Link Clustering



Complete link: We take the dissimilarity to be the maximum dissimilarity between points in the clusters. In this case, the new dissimilarity can be computed by

$$d_{km} = \frac{1}{2}(d_{i,m} + d_{j,m} + |d_{i,m} - d_{j,m}|) = \max_{x \in C_i, x' \in C_j} D(x, x')\,.$$

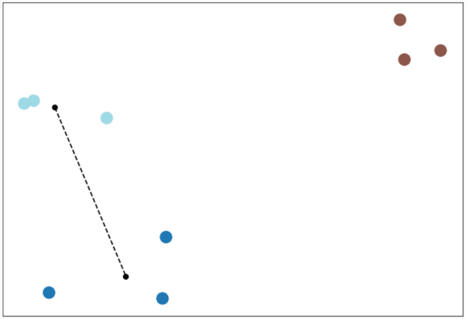
Group Average Clustering



Group Average (unweighted average): We take the average dissimilarity between all members of both clusters. The new dissimilarity can be computed by

$$d_{km} = \frac{1}{|C_i||C_j|}(|C_i|d_{i,m} + |C_j|d_{j,m}) = \frac{1}{|C_i||C_j|}\sum_{x \in C_i, x' \in C_j} D(x, x')\,.$$

Centroid Link Clustering

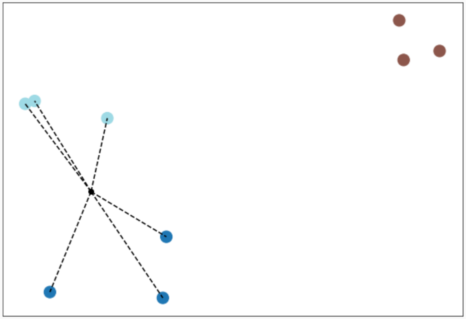


Centroid Average (unweighted pair-group centroid): We find the centroid (vector average) of each cluster and take the average dissimilarity between centroids. The new dissimilarity can be computed by

$$d_{km} = \frac{1}{|C_i| + |C_j|}(|C_i| d_{i,m} + |C_j| d_{j,m}) - \frac{|C_i||C_j|}{(|C_i| + |C_j|)^2} d_{i,j} = D\left(\mu_i, \mu_j\right) ,$$

where $\mu_i = \frac{1}{|C_i|} \sum_{x \in C_i} x$.

Ward Clustering



Finally, the **Ward Minimum Variance** criteria forms pairs so as to minimize the pooled within group sum of squares. The new dissimilarity can be computed by

$$d_{km} = \sum_{x \in C_i \cup C_j} D\left(x, \mu_{C_i \cup C_j}\right)^2 ,$$

where $\mu_{C_i \cup C_j}$ is the centroid of $C_i \cup C_j$. This is $\alpha_i = (|C_i| + |C_m|)/(|C_k| + |C_m|)$, $\beta = -|C_m|/(|C_k| + |C_m|)$, $\gamma = 0$ in the schema from before.

Clustering Methods Compared

Single link: Tend to form long chains in which nearby elements are close, but elements at opposite end could be far away. For similar reasons it is sensitive to noise and outliers.

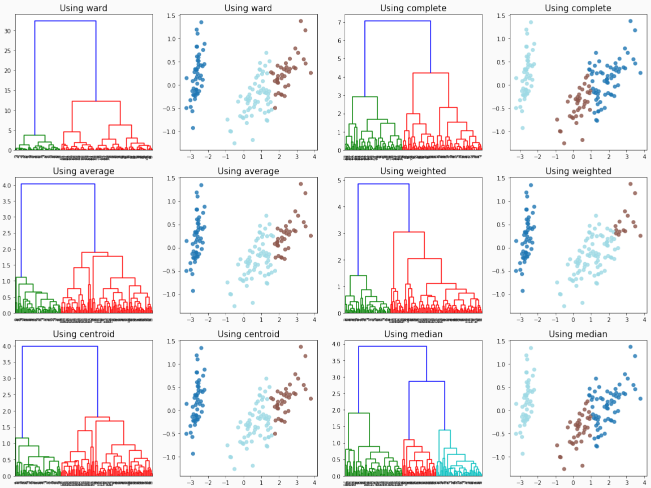
Complete Link: Tends to find compact clusters of approximately equal diameters, avoids the chaining of single link and is less susceptible to noise. But it can break large clusters and merge small clusters into large ones.

Group Average: Acts as a compromise between single link and complete link clustering. While it is less susceptible to noise it is preferenced towards globular clusters.

Centroid Average: Acts as a compromise between averaging and Wards method.

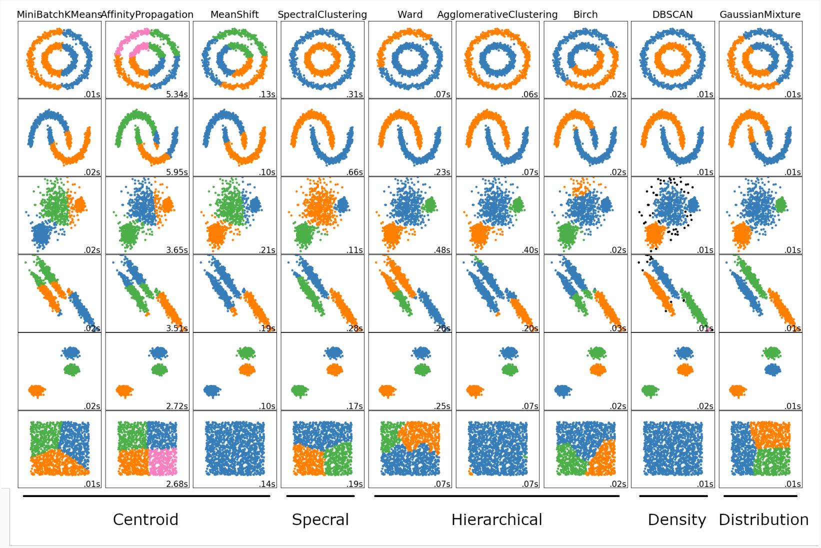
Wards Minimum Variance: The hierarchical version of K-means clustering.

Clustering Methods Compared



Here, we compare the distance criteria on the first two principle components of the IRS dataset. Note that for hierarchical clustering, R and scipy have much more robust libraries than scikit learn.

Clustering Paradigms Compared



We see that Ward strikes a medium between K-means clustering and hierarchical. Birch is a method adapted to large datasets.

Divisive Clustering

Divisive Clustering

While Agglomerative clustering is almost exclusively used today, divisive clustering tracks our cognitive processes of information organization more closely. The algorithm for divisive clustering is

```
    Choose cluster to split.
    Replace chosen cluster with subclusters.
```

This requires a criteria for each step. We will discuss one set of criteria here called cut based optimization using Inter/Intra cluster costs.

Intercost and intracost

Given a training set $\mathcal{T}$, and clusters C_i, define the **intercost** to be

$$\text{Inter}(C_i) = \sum_{x \in C_i} \sum_{x' \in \mathcal{T} - C_i} D(x, x')\,,$$

to be the total dissimilarity between C_i and all other datapoints. The intracost is define as the total internal dissimilarity

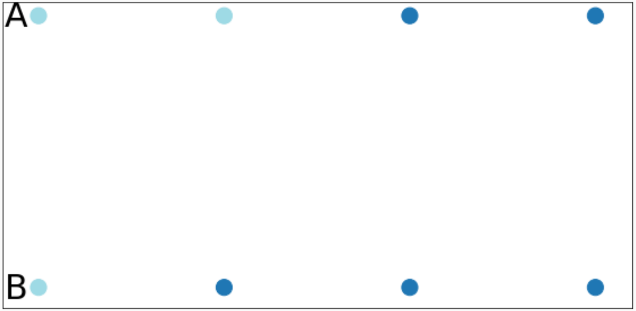
$$\text{Intra}(C_i) = \sum_{x \in C_i} \sum_{x' \in C_i} D(x, x')\,.$$

The cost of a clustering is then

$$\text{Cost}(C) = \sum_{i} \frac{\text{Intra}(C_i)}{\text{Cut}(C_i)}\,,$$

and our goal is to minimize this cost.

Intercost and intracost: Example



Lets compute the cost for the clusters above, where the metric is just the euclidean distance:

$$\text{Cost}(A) : \frac{1+1}{5+3} + \frac{1+1}{5+3} = \frac{1}{2}, \qquad \text{Cost}(B) : \frac{0}{6} + \frac{3+2+3}{1+2+3} = \frac{7}{6}.$$

Notice that the intercost distance tends upward as the intracost distance tends downward while equalizing the partition.

Divisive Clustering

The following algorithm, after a cluster is selected for splitting, the following algorithm is non-increasing on cost: For a fixed K, pick a random clustering $C_1, \ldots, C_K$. Then

Add all datapoints to a lock set L

> Pick C_i at random and pick a $x_j \in C_i$ such that $x_j \notin L$.
>
> Move x_j to the cluster that minimized the cost.
>
> Remove x_j from the lock set L.
>
> Repeat until L is empty.

The algorithm is repeated until convergence

Divisive Clustering

Advantages of divisive clustering:

- Vertex locking ensures all points are considered.
- Works to minimize cost based metric that depends on both internal and external information.
- Much more natural for our conception of many problems as classified from rough splitting to fine.

There are also some significant disadvantages:

- During splitting, needs to find best of $O(2^n)$ splits as opposed to $O(n^2)$ merges.
- If not splitting K must be specified in advance.
- Doesn't seem to be a consensus on a schema for splitting.

In modern times divisive clustering is simply not used for the reasons above. But it may be ready for another now that the computational power is here.

Evaluating Clustering

Clustering Criteria

Evaluating clustering falls into two rough categories:

Criteria internal to a dataset:

Sum of squared error, both intracluster and intercluster.

Scattering Criteria: Minimize the intracluster variance and maximize the intercluster variance.

External criteria:

Precision and recall for true labeling.

Uses mutual information.

Internal Measures: Scattering Criteria

The total cluster variance can be decomposed into the inter-(between)-scatter variance and the intra-(within)-scatter variance

$$\mathrm{Cov} = \sum_{i=1}^{N} (x_n - \mu)(x_n - \mu)^T = S_W + S_B \,,$$

where

Total Intra $$S_W = \sum_{j=1}^{K} \sum_{x \in C_k} (x - \mu_k)(x - \mu_k)^T \,,$$

Total Inter $$S_B = \sum_{j=1}^{K} C_k|(\mu_j - \mu)(\mu_j - \mu)^T \,.$$

Internal Measures: Calinski-Harabaz Index

The Calinski-Harabaz (CH) index is defined as

$$CH = \frac{\mathrm{Tr}(S_B)}{\mathrm{Tr}(S_W)}\,.$$

The trace begin the sum over eigenvalues, the index will be large when the between group variance is large and the within group variance is small.

The Calinski-Harabaz (CH) index is a good measure of relative clustering on a specific dataset, but should not be treated as a measure to compare clustering between datasets.

Internal Measures: Silhouette Coefficient

The silhouette coefficient measures how similar each data point is to it's own cluster (cohesion) compared to other clusters (separation). Let a_i be the average distance from x_i to the other elements in the cluster

$$a_i = \frac{1}{|C_k| - 1} \sum_{x_j \in C_k} D(x_i, x_j)\,,$$

and let b_i be the smallest average distance of x_i to all points in another cluster

$$b_i = \min_{\ell \neq k} \frac{1}{|C_\ell|} \sum_{x_j \in C_\ell} D(x_i, x_j)\,.$$

Then the silhouette coefficient is

$$SC = \frac{1}{N} \sum_{i=1}^{N} \frac{b_i - a_i}{\max(a_i, b_i)}\,.$$

Internal Measures: Silhouette Coefficient

The silhouette coefficient

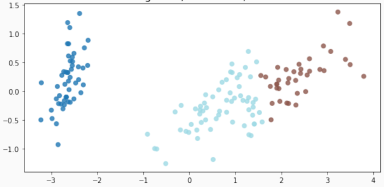
$$SC = \frac{1}{N} \sum_{i=1}^{N} \frac{b_i - a_i}{\max(a_i, b_i)} .$$

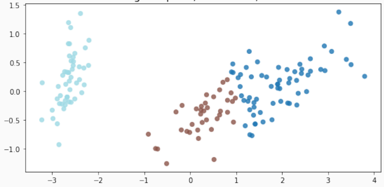
Ranges from -1 to $+1$, where a high value indicates that the object is well matched to its own cluster and poorly matched to neighboring clusters.

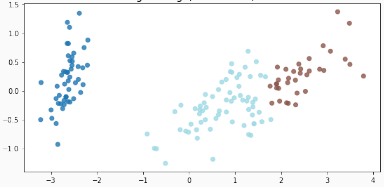
The silhouette for each datapoint can also be measured individually. If most points have a silhouette near 1, then the clustering configuration is cohesive and separated. Otherwise, the clustering configuration may have an inappropriate number of clusters.

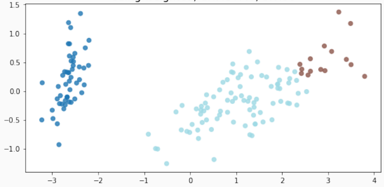
The individual coefficient $s_i = \frac{b_i - a_i}{\max(a_i, b_i)}$ is also a good test for outliers or misclassified points with zero values corresponding to outliers and negative values to misclassifications.

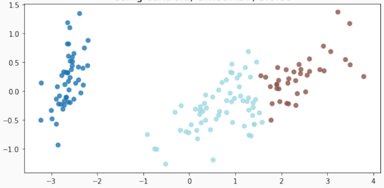
Clustering Metrics Compared

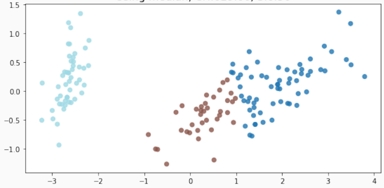

Using ward, CH:687.37, S:0.60


Using complete, CH:629.66, S:0.56


Using average, CH:675.14, S:0.60


Using weighted, CH:497.49, S:0.56


Using centroid, CH:687.37, S:0.60


Using median, CH:629.66, S:0.56

External Measures: Mutual Information

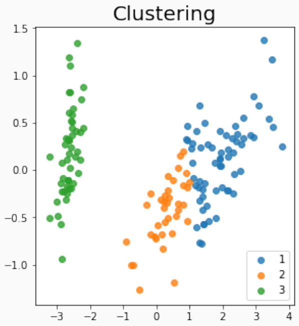
Assume now that there is a true categorical labeling that is known an let $y \in \mathcal{Y}$ be the classes. Let k parameterize the K classes picked by clustering. The **mutual information** is

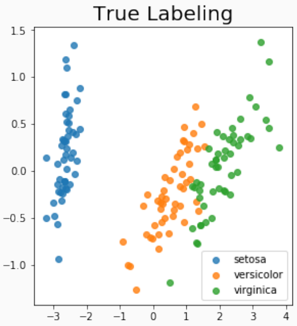
$$I(\mathcal{Y}, K) = \sum_{y \in \mathcal{Y}, k \in K} p(y, k) \log \frac{p(y, k)}{p(y)p(k)} ,$$

where $p(y, k)$ is the total chance of a datapoint x_i with true label y being in cluster k, $p(y)$ is the proportion of data points labeled y, and $p(k)$ is the proportion of data points in cluster k.

Mutual information measures how much knowing one of these variables reduces uncertainty about the other. High mutual information indicates a large reduction in uncertainty; low mutual information indicates a small reduction; and zero mutual information between two random variables means the variables are independent.

External Measures: Mutual Information





$p(y,k)$	1	2	3	$p(k)$
setosa	0	0	.333	.333
versicolor	.093	.24	0	.333
virginica	.327	.007	0	.333
$p(y)$	.42	.247	.333	

$I(\mathcal{Y}, K) = 0.845$ in this example,

External Measures: Mutual Information

Without getting too much into the details, mutual information is used here instead of correlation because it can account for categorical variables. Comparing the two,

$$\mathrm{Cov}(X, Y) = \sum_{x,y} [p(x, y) - p(x)p(y)]xy \,,$$

and

$$I(X, Y) = \sum_{x,y} [log(p(x, y)) - log(p(x)p(y))]p(x, y) \,,$$

we see that covariance looks at how non-independence effects the product xy, while mutual information measure the log of the effect of nonindependence on the mutual probability distribution. In some sense here, mutual information is the covariance of the distributions. It is subtly but directly related to the χ^2 statistic.

The Theoretical Problem

Axiomatizing Clustering

What distinguishes a clustering algorithm from any arbitrary function from a training set to partition space? One may try to use an axiomatic approach, for example a 2003 attempt by Kleinberg:

Consider a clustering function $F(D)$ from a domain $\mathcal{X}$ to its partitions that depends on a dissimilarity matrix D. Consider the following properties:

Scale Invariance: For any $\alpha > 0$ the following should hold: $F(D) = F(\alpha D)$
Richness: For every finite set $\mathcal{X}$ and every partition $C = \{C_1, \ldots, C_K\}$, there exists some dissimilarity function D over $\mathcal{X}$ such that $F(\mathcal{X}, D) = C$.
Constancy: If D, D' are dissimilarity functions such that for every $x, y \in \mathcal{X}$, (if x, y belong to the **same** cluster in $F(D)$ then $D'(x, y) \leq D(x, y)$) **and** (if x, y belong to **different** clusters in $F(D)$ then $D'(x, y) \geq D(x, y)$), **then** $F(D) = F(D')$.

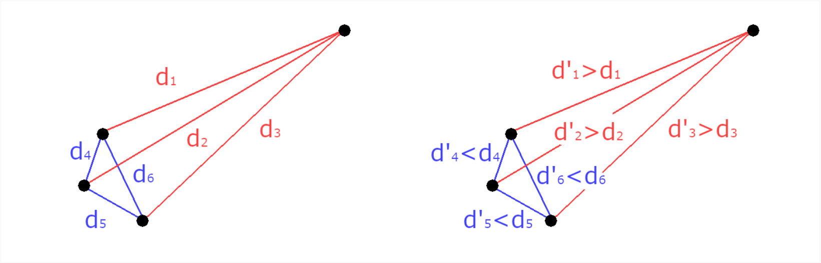
Axiomatizing Clustering

Scale invariance is an obvious requirement. Richness might seem a little strange at first but it enforces the fact that the distance function should contain all of the information about the clustering, none comes from the background space itself.

Constancy: If D, D' are dissimilarity functions such that for $x, y \in \mathcal{X}$, if x, y belong to the **same** cluster in $F(D)$ then $D'(x, y) \leq D(x, y)$ **and** if x, y belong to **different** clusters in $F(D)$ then $D'(x, y) \geq D(x, y)$, **then** $F(D) = F(D')$.

Consistency is the condition that enforces the definition of clustering. Similar points should be clustered together and different points should be clustered differently, and if points that share a cluster get **more** similar and those don't get less similar the clustering function should be even stronger.

Axiomatizing Clustering



Constancy: If D, D' are dissimilarity functions such that for $x, y \in \mathcal{X}$, if x, y belong to the **same** cluster in $F(D)$ then $D'(x, y) \leq D(x, y)$ **and** if x, y belong to **different** clusters in $F(D)$ then $D'(x, y) \geq D(x, y)$, **then** $F(D) = F(D')$.
Consistency states that the clusters $F(D)$ should be the same as for $F(D')$.

Axiomatizing Clustering

You can see the problem a mile away: **Theorem** (Kleinber, 2003) There exists no function F that satisfies all three properties.

Proof: Let $\mathcal{X} = \{a, b, c\}$. By richness there must be a D such that $F(D) = \{\{a\}, \{b\}, \{c\}\}$ and another D' such that $F(D') \neq F(D)$.

For some value of α, $\alpha D'(x, y) \geq D(x, y)$ for all $x, y \in \mathcal{X}$. By scale invariance, $F(\alpha D') = F(D')$.

Finally, since all distinct x lie in their own cluster wrt $F(D)$ and $\alpha D'(x, y) \geq D(x, y)$, consistency implies that $F(\alpha D') = F(D) = F(D')$, a contradiction. $\square$

Note, that an two of these axioms are alright, it takes all three to result in a problem.

Axiomatizing Clustering

Kleinbergs impossibility result can easily be circumvented by varying the properties. For example, if we fix the number of clusters we can replace richness with K richness so every partition into K-subsets is realizable by some D. Scaling, Consistency, and K richness all hold for K-means.

One can also relax the consistency property, making the clustering with more extreme D a refinement of those with less extreme. .

The way Ben-David understands this result is as a No-Free-Lunch kind of theorem for clustering, stating the every clustering problem will have some undesirable properties. What we see is the same trade-off as before, a tradeoff between fixing the number of clusters K, alowing the clusters to solidify based on internal and external distance, but not allowing the algorithm to start splitting legitimate clusters into subclusters.

References

A lecture on divisive clustering: http://www.cs.princeton.edu/courses/archive/spr08/cos435/Class_notes/clustering4.pdf

Mutual information vs Covariance: https://stats.stackexchange.com/questions/81659/mutual-information-versus-correlation