> **Subject**: ML slides pack - clustering with and without kmeans by Claudio Sartori
>
> **Course:** Artificial Intelligence - LM
>
> **Department**: DISI (Department of Computer Science and Engineering) - University of Bologna, Italy
>
> **Author**: Roberto Zanolli
>
> **TIPS**
>
> - w.r.t means "with respect to"

Clustering

What is clustering

Given: A set of N objects x_i, each described by D values x_{id}
Task: Find a natural partitioning into K clusters and possibly identify noise objects

Result: A clustering scheme - a function mapping each data object to $\{1..K\}$ (or to noise)

Desired cluster property:

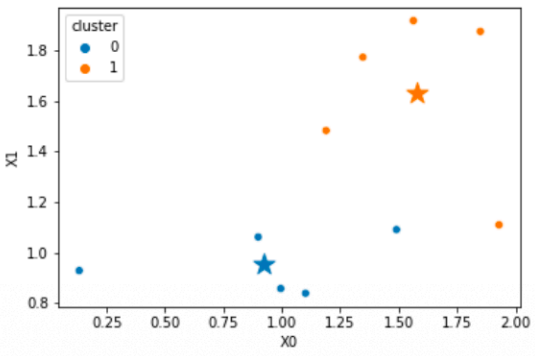
- Objects in same cluster are similar
- Look for clustering scheme that **maximizes intra-cluster similarity**

Formal definition for clustering function

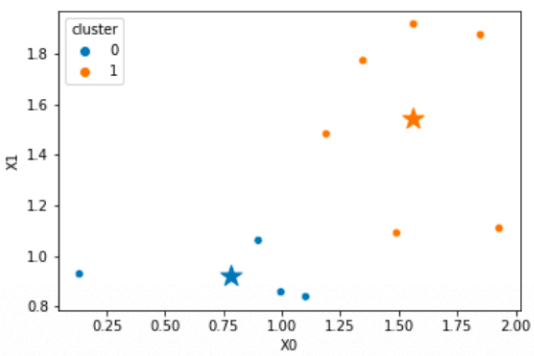
> Find a function $clust()$ from X to $\{1..K\}$ such that:
>
> 1. $\forall x_1, x_2 \in X$, $clust(x_1) = clust(x_2)$ when x_1 and x_2 are similar
> 2. $\forall x_1, x_2 \in X$, $clust(x_1) \neq clust(x_2)$ when x_1 and x_2 are not similar

How do we choose a measure to determine clusters

$clust^a$



$clust^b$



Centroid

Definition: A point whose coordinates are the average coordinates of all points in the cluster

Calculation: For each cluster k and dimension d, the d-th coordinate of the centroid is:

> $$centroid_d^k = \frac{1}{|x_i : clust(x_i) = k|} \sum_{x_i : clust(x_i) = k} x_{id}$$

Taxonomy of clustering method

The ones with means that have been seen during lab*

Partitioning:

- **K-means (MacQueen 67)** *
- Expectation maximization (Lauritzen 95)
- CLARANS (Ng and Han 94)

Hierarchic:

- **Agglomerative/divisive*** *
- BIRCH (Zhang et al 96)
- CURE (Guha et al 98)
- Based on linkage

Based on density:

- **DBSCAN (Ester et al 96)** *
- DENCLUE (Hinnenburg and Keim 98)

Statistics:

- IBM-IM demographic clustering
- COBWEB (Fisher 87)
- Autoclass (Cheeseman 96)

K-MEANS CLUSTERING

The challenge: Given data that appears to form distinct groups (like a five-component gaussian mixture), how do we automatically discover these groups in high-dimensional space?

Core idea: View clustering as a lossy compression problem. If we must transmit point coordinates using very few bits, we need to group similar points and represent them by a common "code" (centroid).

Loss function: We measure quality by the sum of squared errors (SSE) between actual points and their representative centroids.

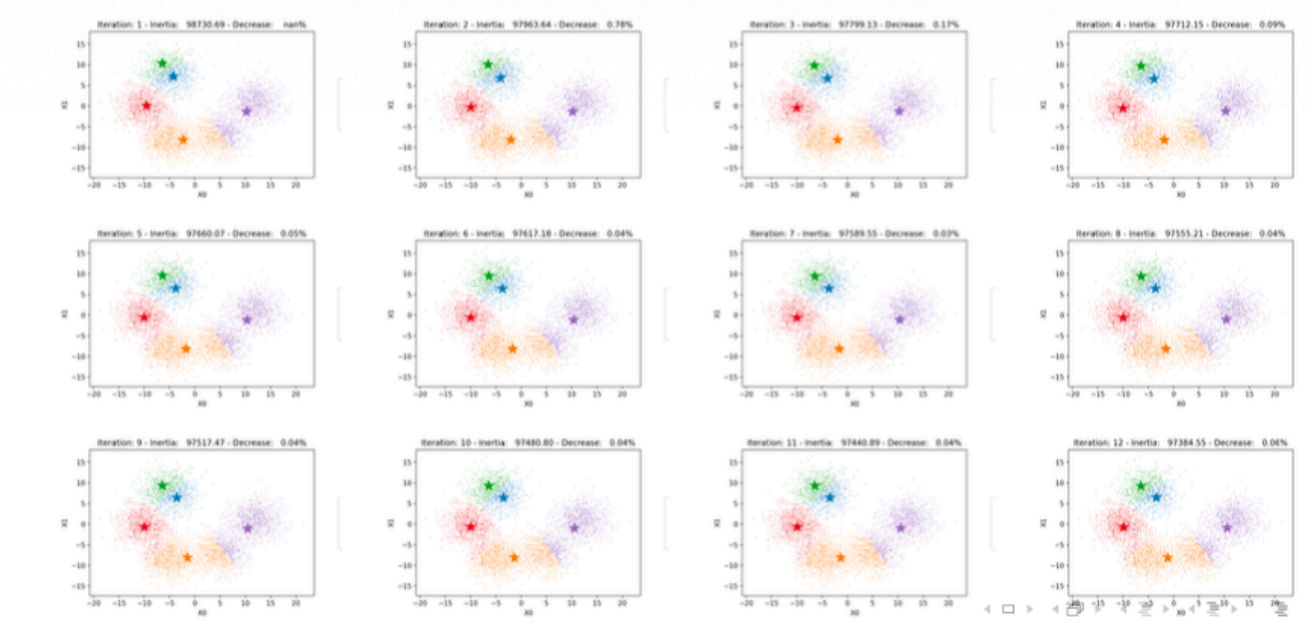
> **Algorithm intuition:** Instead of a fixed grid, let the data itself determine optimal group representatives:
> 1. Start with random guesses for group centers
> 2. Assign each point to its nearest center

> 3. Update centers to be the centroid (mean) of their assigned points
> 4. Repeat until assignments stabilize

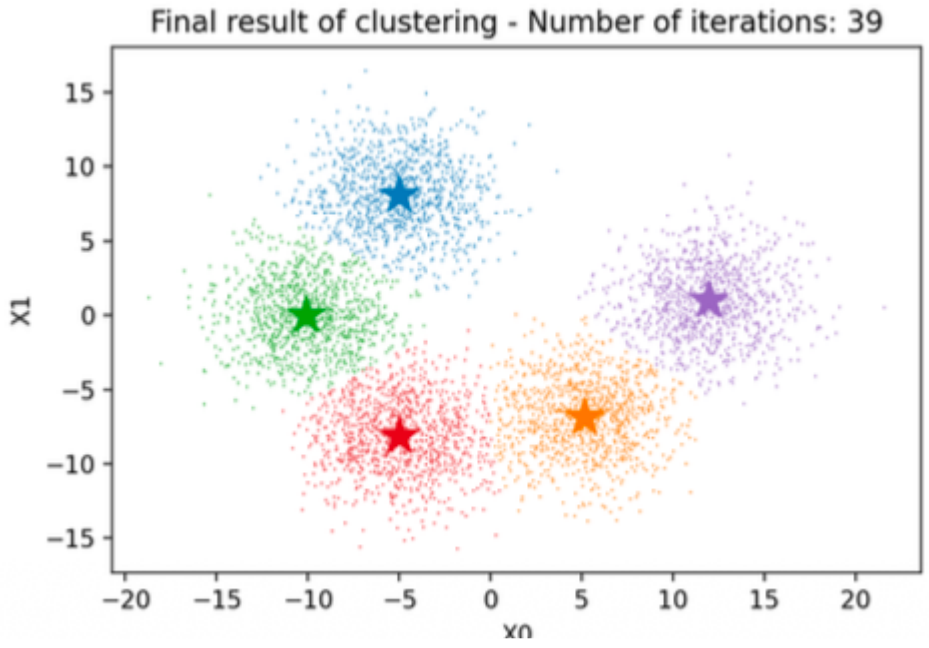
Key questions that remain:

1. What are we trying to optimize?
2. Is termination guaranteed?
3. Are we sure that the best clustering scheme is found? (And what defines "best"?)
4. How should we initialize the centers?
5. How can we determine the optimal number of clusters?

Applying the algorithm seen above we observe the example:



From numerous iterations we get here:



Question 1: What are we trying to optimize?

Answer: Distortion (Inertia)

Definition: Given:

- Dataset $\{x_i, i = 1..N\}$
- Encoding function $Encode : \mathbb{R}^D \to \{1..K\}$
- Decoding function $Decode : \{1..K\} \to \mathbb{R}^D$

$$Distortion = \sum_{i=1}^{N} (x_i - Decode(Encode(x_i)))^2$$

Let $Decode(k) = c_k$, then:

$$Distortion = \sum_{i=1}^{N} (x_i - c_{Encode(x_i)})^2$$

Minimal distortion

Question: What properties do $c_1, \ldots, c_K$ need for minimal distortion?

Property 1: x_i must be ***encoded with the nearest center***

- Otherwise distortion could be reduced by choosing the nearest center $c_{Encode(x_i)} = \arg\min_{c_j \in \{c_1,\ldots,c_K\}} (x_i - c_j)^2$

Property 2: Partial derivative of distortion w.r.t. each center must be zero

- Indicates maximum or minimum of function

Mathematical derivation

$$Distortion = \sum_{i=1}^{N} (x_i - c_{Encode(x_i)})^2 = \sum_{j=1}^{K} \sum_{i \in OwnedBy(c_j)} (x_i - c_j)^2$$

$$\frac{\partial Distortion}{\partial c_j} = -2 \sum_{i \in OwnedBy(c_j)} (x_i - c_j) = 0 \text{ at minimum}$$

Solution:

$$c_j = \frac{1}{|OwnedBy(c_j)|} \sum_{i \in OwnedBy(c_j)} x_i$$

Question 2: Is termination guaranteed?

Answer: YES
Proof: Finite number of states ensures termination

- Finite ways to partition N objects into K groups
- Finite configurations where centers are centroids of their points
- Each iteration reduces distortion → new state never visited before
- Eventually no new states reachable → algorithm stops

Question 3: Are we sure that the best clustering scheme is found? (And what defines "best"?)

Answer: model the problem like a local vs global minimum problem

Problem: Ending state not necessarily global optimum

- Multiple local minima possible
- Final clustering depends on initialization

Question 4: How should we initialize the centers?

Answer: we try several times and pick the best
Strategies:

1. Random first point, subsequent points far from previous ones
2. Run algorithm multiple times with different random starts
3. Choose best result based on distortion

Question 5: How can we determine the optimal number of clusters?

Answer: we try several times and pick the best
Challenge: No easy automatic method

Approach:

- Try various K values
- Use quantitative evaluation metrics
- Balance intra-cluster compactness vs inter-cluster separation

Summary of application

- Try with different starting points with same K and choose the best starting point to ensure you get to global minimum (at least we try)
- After you choose

Proximity function

Default: Euclidean distance (works well for vector spaces)
Alternatives: Various distance metrics for specific data types

Sum of Squared Errors (SSE)

Formula:

$$SSE = \sum_{j=1}^{K} \sum_{i \in OwnedBy(c_j)} (x_i - c_j)^2$$

Properties:

- $SSE_j = 0$ only if all points coincide with centroid
- SSE decreases with increasing K, reaches 0 when $K = N$
- Cannot minimize SSE to choose K (always prefers more clusters)

Outliers

Impact: High distance from centroid → high contribution to SSE

- Bad influence on clustering results
- Sometimes beneficial to remove (domain-dependent)

Common uses of K-means

1. **Data exploration:** Quick initial clustering
2. **1D discretization:** Create non-uniform bins
3. **Vector quantization:** Signal processing, compression
4. **Color quantization:** GIF compression, color palette selection

Time complexity

The time complexity is described as it follows.

> **Time:** $O(TKND)$
>
> *Where*:
>
> - T = number of iterations
> - K = number of clusters
> - N = number of data points
> - D = number of dimensions

Pros and cons

PROS:

- Fairly efficient (nearly linear in N)
- $T, K, D \ll N$ in practice

CONS:

- Requires space where centroid computable (works with Euclidean, some other distances)
- Cannot handle nominal data
- Requires K parameter (though can be searched)
- Sensitive to outliers
- No noise handling
- Poor with non-convex clusters

Evaluation of clustering schemes

Clustering evaluation focuses solely on the results, not the technique used. Since clustering is an unsupervised method with little a priori information (like class labels), evaluation becomes critical. We need score functions to measure various properties of clusters and the overall clustering scheme. The terms "score" and "index" are synonymous in this context.

When supervised data is available, it can aid evaluation. In 2D, clusters can be visually inspected, while in higher-dimensional spaces, 2D projections help but formal methods are preferred.

Key evaluation challenges

- Distinguishing genuine patterns from random apparent regularities
- Determining the optimal number of clusters
- Conducting non-supervised evaluation
- Performing supervised evaluation when possible
- Comparing clustering schemes relatively

Proximity measures

Similarity/Proximity: A two-variable function measuring how similar two objects are based on their property values.

Dissimilarity: A two-variable function measuring how different two objects are based on their property values (e.g., Euclidean distance).

Cohesion and separation metrics

Global separation (SSB): Sum of Squares Between clusters measures separation. Let c be the global centroid:

$$SSB = \sum_{i=1}^{K} N_i \cdot Dist(c_i, c)^2$$

Total Sum of Squares relationship: The total variance can be decomposed into within-cluster and between-cluster components:

$$TSS = SSE + SSB$$

(The total sum of squares is a global property of the dataset, independent from the clustering scheme).
This shows that minimizing within-cluster variance (SSE) is equivalent to maximizing between-cluster variance (SSB) for a fixed dataset.

Mathematical derivation

The total sum of squares expands as:

$$\begin{aligned} TSS &= \sum_{i=1}^{K} \sum_{x \in k_i} (x - c)^2 \\ &= \sum_{i=1}^{K} \sum_{x \in k_i} [(x - c_i) - (c - c_i)]^2 \\ &= \sum_{i=1}^{K} \sum_{x \in k_i} (x - c_i)^2 - 2 \sum_{i=1}^{K} \sum_{x \in k_i} (x - c_i)(c - c_i) + \sum_{i=1}^{K} \sum_{x \in k_i} (c - c_i)^2 \\ &= SSE + SSB \end{aligned}$$

The cross-term vanishes because $\sum_{x \in k_i} (x - c_i) = 0$ by definition of the cluster centroid c_i.

Cluster-specific evaluation

Individual clusters can be evaluated separately:

- Poorly performing clusters may benefit from splitting
- Weakly separated clusters might be merged
- Border objects can negatively impact cohesion or separation

Silhouette score

The silhouette score provides a standardized **measure of clustering quality** with values in $[-1, 1]$. It increases with better separation and decreases with poor cohesion (high sparsity).

The within-cluster sum of squares decomposes as:

$$SSE = \sum_{j=1}^{K} \sum_{i \in \text{OwnedBy}(c_j)} (x_i - c_j)^2 = \sum_{j=1}^{K} SSE_j$$

Individual object contribution

For each object x_i, we compute:

Cohesion measure (a_i): Average distance to other objects in the same cluster:

$$a_i = \text{average}_{j, y(x_j)=y(x_i)} \text{dist}(x_i, x_j)$$

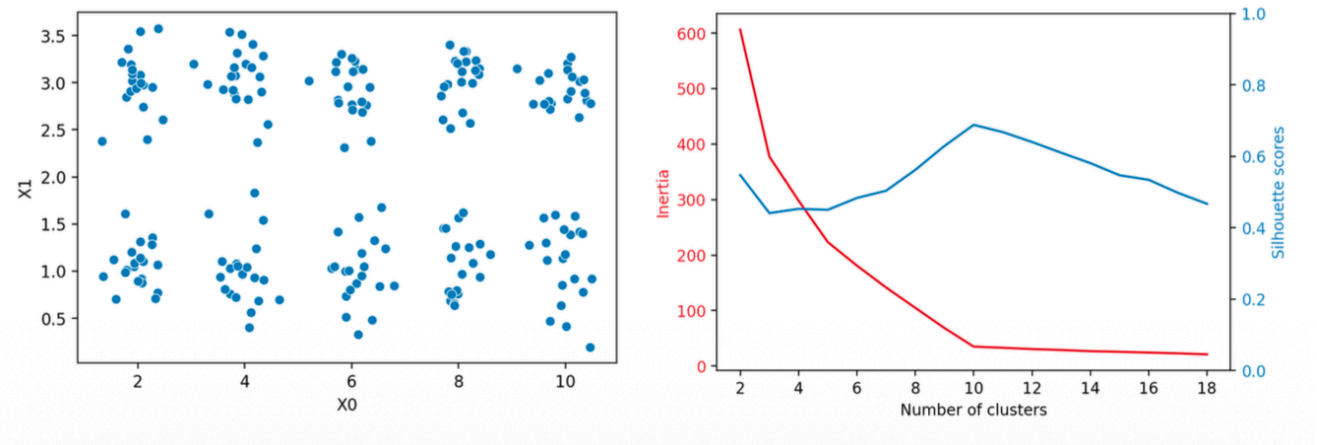
Separation measure (b_i): Minimum average distance to objects in other clusters:

$$b_i = \min_{k \in Y, k \neq y(x_i)} (\text{average}_{j, y(x_j)=k} \text{dist}(x_i, x_j))$$

Silhouette calculation

> The silhouette score for object x_i is:
>
> $$s_i = \frac{b_i - a_i}{\max(a_i, b_i)} \in [-1, 1]$$
>
> Global scores are obtained by averaging over clusters or the entire dataset.

Interpretation: A negative score indicates an object is closer to other clusters than to its own, suggesting poor cluster assignment.



Looking for the best number of clusters

Some clustering algorithms, such as K-means, require specifying the number of clusters K as a parameter beforehand. Since evaluation measures like SSE and Silhouette score are influenced by the choice of K, they can be used to optimize this parameter.

Behavior of evaluation metrics

SSE (within-cluster variance) exhibits predictable behavior:

- Decreases monotonically as K increases
- Equals TSS (total variance) when $K = 1$ (all points in one cluster)
- Approaches zero when $K = N$ (each point is its own cluster)

Silhouette score provides a more nuanced evaluation but is computationally expensive to compute.

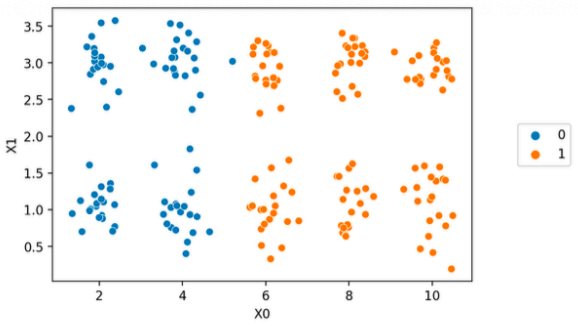
Finding the optimal number of clusters

> **Elbow method**
>
> - Plot inertia (SSE) against different values of K
> - Look for points where the slope decreases significantly
> - These inflection points often represent plausible values for K
> - The method identifies where adding more clusters provides diminishing returns

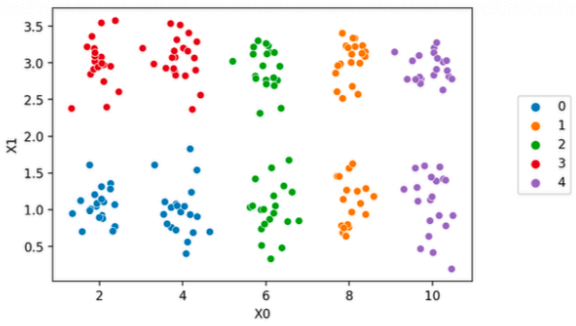
> **Silhouette score method**
>
> - Calculate silhouette score for different values of K
> - The score typically reaches a maximum at one specific K value
> - This maximum point indicates the optimal number of clusters
> - Provides a clear, point-like optimum for cluster selection

Example of a dataset

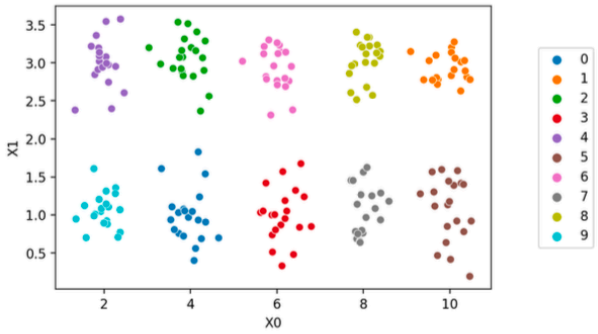
Using this example dataset we can observe the clustering on different K values.
We obtain the best separation on $K = 10$

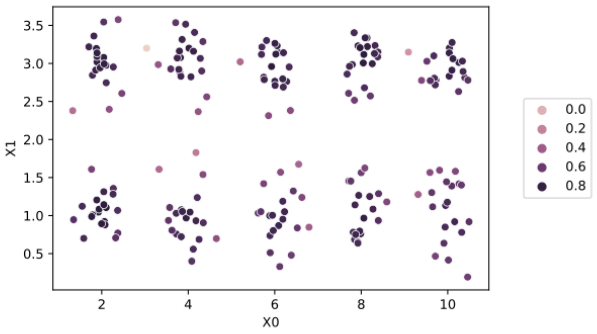


$K = 2$



$K = 5$





Supervised evaluation measures

Gold standard comparison

When a reference partition (gold standard) is available with labeling scheme $y_g(\cdot)$, similar to supervised data for classifier training, we can compare it with our clustering scheme $y_k(\cdot)$. Note that:

- The number of distinct labels V_g and V_k may differ
- Even with identical groupings, label permutations may be needed for comparison

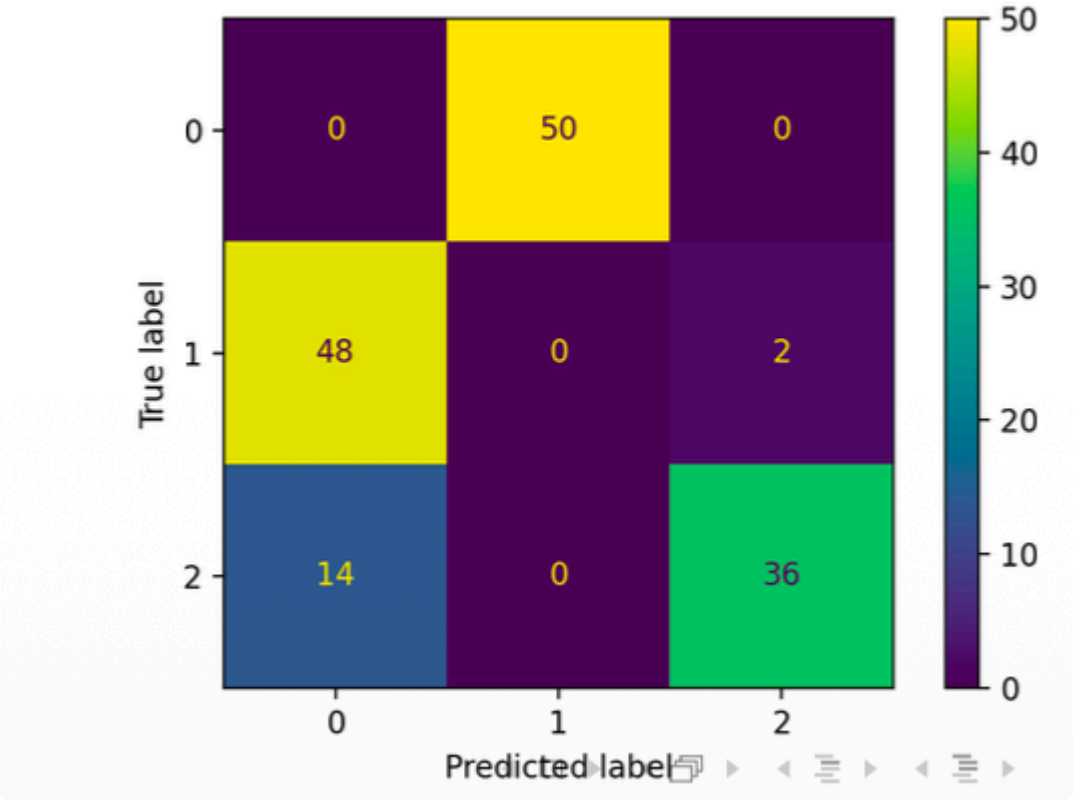
Purpose of gold standard comparison

- Validate clustering techniques for application to new, unlabeled data
- Similar to classifier testing, but focused on grouping rather than labeling
- Ensures the clustering method produces meaningful groupings

Classification-oriented measures

Using confusion matrix analysis:

```
X, y = load_iris(return_X_y=True)
estimator = KMeans(n_clusters=3, random_state=363)
y_km = estimator.fit_predict(X)
disp = ConfusionMatrixDisplay(confusion_matrix(y,y_km))
disp.plot()
```



Measures how gold standard classes distribute among clusters using:

- Confusion matrix
- Precision, recall, F-measure
- Best label permutation matching

Similarity-oriented measures - I

Compare pairs of objects based on their grouping in both schemes:

- **SGSK**: Same group in both y_g and y_k
- **SGDK**: Same group in y_g, different in y_k
- **DGSK**: Different groups in y_g, same in y_k
- **DGDK**: Different groups in both schemes

Similarity-oriented measures - II

Results given by `pair_confusion_matrix(y_g,y_k)`

	SK	DK
SG	13512	1488
DG	1200	6150

Rand Score:

$$\frac{SGSK + DGDK}{SGSK + DGDK + SGDK + DGSK} = 0.88$$

Adjusted Rand Score: Excludes matches expected by chance = 0.73

Jaccard Coefficient for label c:

$$\frac{SG_cSK_c}{SG_cSK_c + SG_cDK_c + DG_cSK_c} = (1, 0.75, 0.69)$$

Requires remapping of $y_g(\cdot)$ to obtain best match.