

# Unsupervised Machine Learning with Python

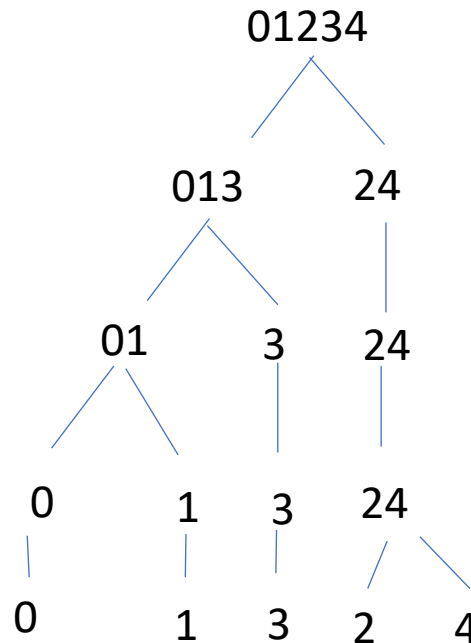
# Section 4.1: Hierarchical Clustering Algorithm

# Hierarchical Clustering

- Hierarchical clustering is a connectivity-based approach
- Bottom up approach called Agglomerative
- Top down approach called Divisive
- Results depend on the distance measure used (use L2 in course)
- See [UnsupervisedML\\_Resources.pdf](#) for links to additional resources

# Hierarchical (Agglomerative) Clustering

- Bottom up approach: each data point starts as its own cluster
- Nearby clusters are repeatedly combined until all points in single cluster
- Creates clusters “at all levels”
- Can be represented in a tree structure (dendrogram)

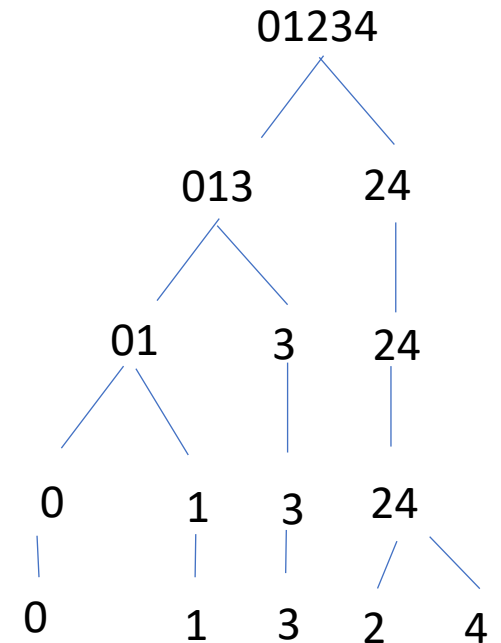
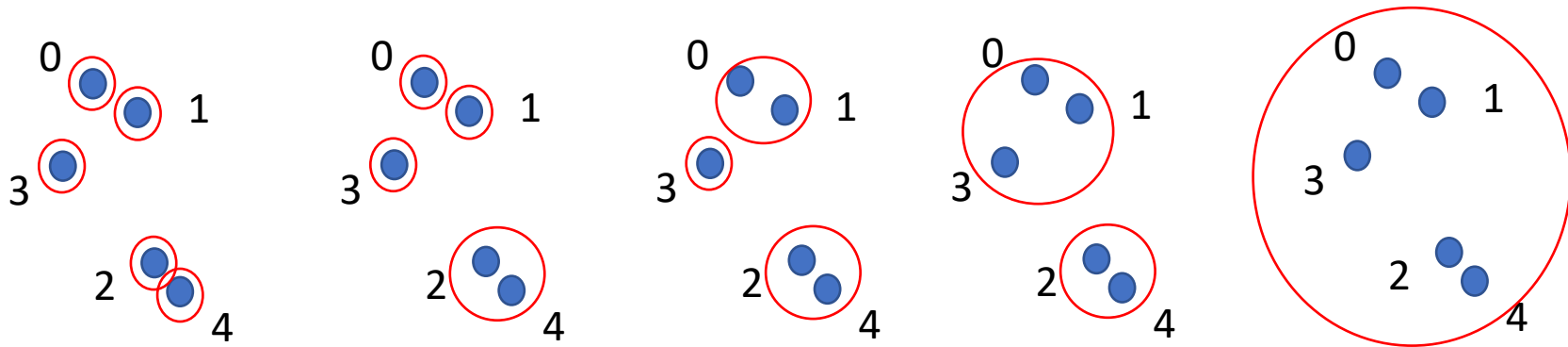


# Hierarchical Clustering - Algorithm

- Assume M data points and define each as a cluster

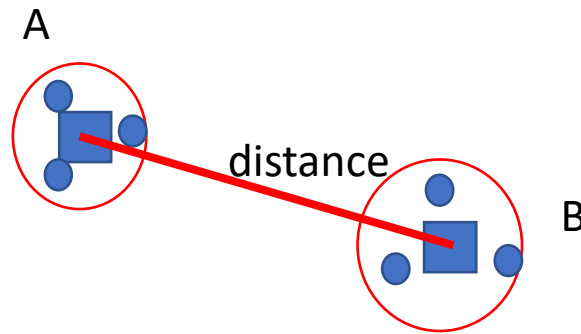
(1) Loop until there is a single cluster

- Compute pairwise distances between each of the clusters
- Combine clusters with the shortest pairwise distance into one cluster



# Distance between Clusters

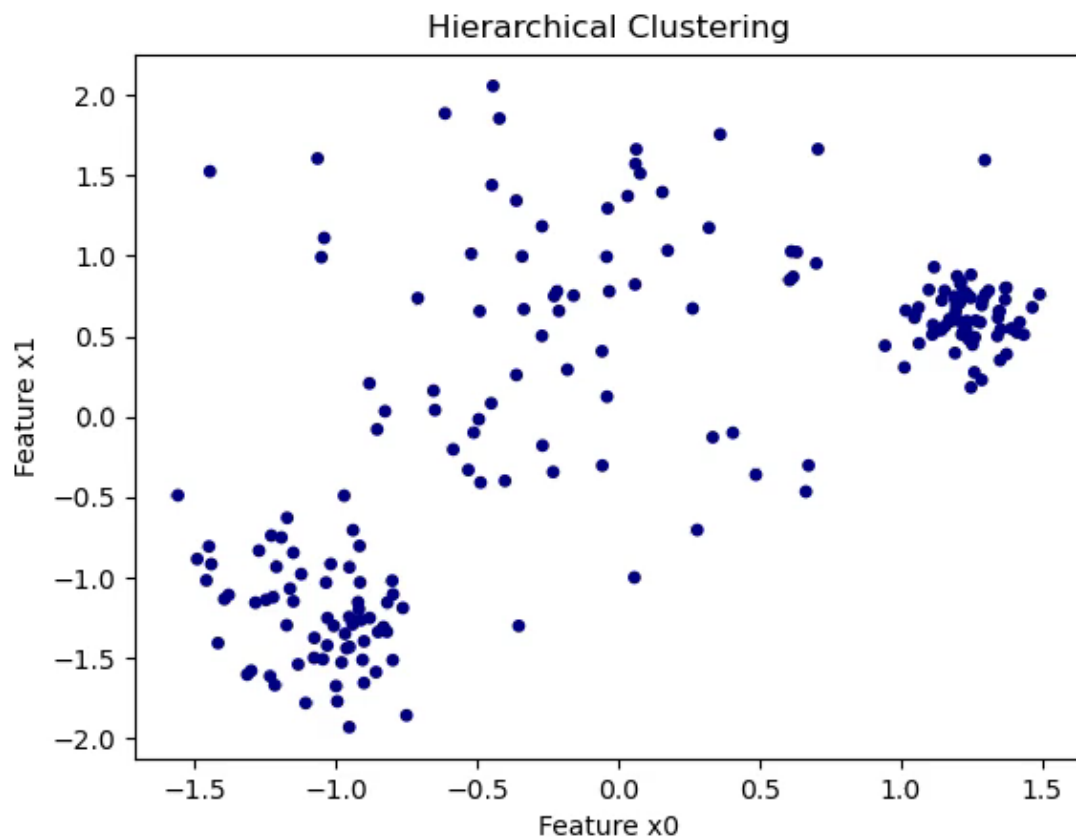
- Define distance between clusters as distance between cluster means



- Alternative:
  - Distance between clusters is min distance between points in cluster A and cluster B

# Hierarchical Clustering - Example

- Dataset: varied\_blobs1 dataset with 200 data points in (2 dimensions)
- Animation shows clustering at all levels to 3 clusters



- Clusters of 1 data point are dark blue
- Change in colour when clusters of 2 points created
- When clusters are combined, colour is that of the larger cluster

# Hierarchical Clustering: Complexity

Assume  $M$  data points and  $d$  dimensions

- Number of operations for clustering for all levels is  $O(M^3)$  as  $M \rightarrow \infty$  (derive this in exercises)
- Can limit memory used to  $O(M)$  as  $M \rightarrow \infty$  (to store feature matrix), as one can compute pairwise distances as needed and then discard
- Number of operations and memory required are both proportional to dimension  $d$



# Hierarchical Clustering: Notes

- Creates cluster of arbitrary shapes
- Clustering at all levels is unique if distances between clusters are unique
  - If  $\text{dist}(\text{clusterA}, \text{clusterB})$  and  $\text{dist}(\text{clusterC}, \text{clusterD})$  are smallest and same, then course code will combine first pair of clusters encountered with smallest distance
  - Can update code to combine clusters A and B and clusters C and D at same level
- Principal limitation: not feasible for large number of data points as number of operations is  $O(M^3)$  as  $M \rightarrow \infty$

# Unsupervised Machine Learning with Python

# Section 4.2: Hierarchical Clustering Code Design

# General Clustering Code Design

- Create a base class that can be used for:
  - Hierarchical Clustering
  - DBSCAN
  - K Means
  - Gaussian Mixture Model
- Start with:
  - Principal Variables
  - Key Methods
- If you would like to design code yourself, then stop video

# General Clustering Code Design

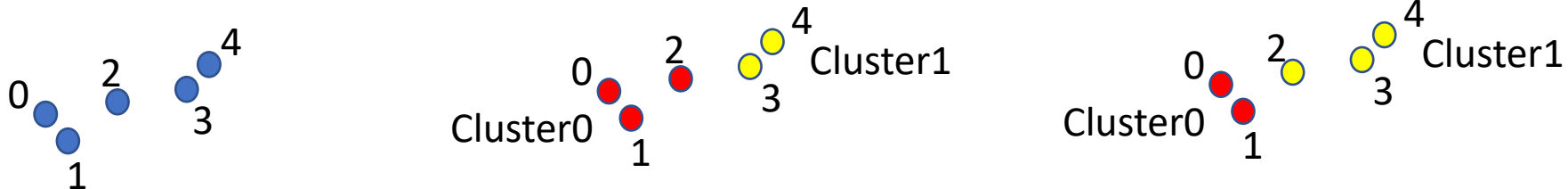
Clustering Codes have the following structure:

## (1) Initialization

- Make initial cluster assignments for each data point
- Initialize other relevant variables

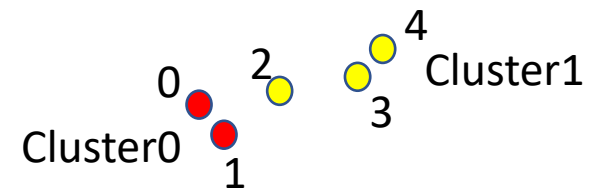
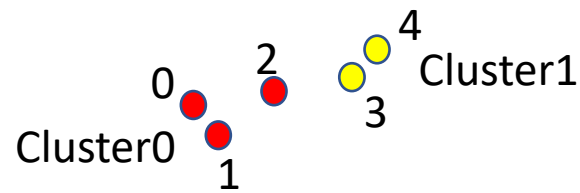
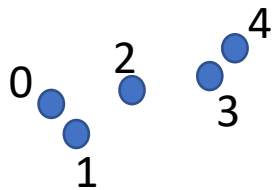
## (2) Determine cluster assignments for each data point

- Use iterative approach (which involves looping) to make better guesses for cluster assignments
- Stop when cluster assignments converge



# clustering\_base class: Principal Variables

Variable	Type	Description
self.time_fit	float	Time for clustering
self.X	2d numpy array	Contains the dataset Number of rows = number of dimensions for data Number of cols = number of data points Example: 2 dimensions and 5 data points $\begin{bmatrix} 1 & 1.1 & 0.8 & 0.6 & 0.6 \\ 0.9 & 1.0 & 0.7 & 0.5 & 0.5 \end{bmatrix}$
self.clustersave	list of 1d numpy arrays	self.clustersave[i][j] is cluster assignment for iteration i, data point j Example: for 3 iterations: $\begin{bmatrix} -1 & -1 & -1 & -1 & -1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \end{bmatrix}$



# clustering\_base class – Key Methods

Method	Description
<code>__init__</code>	Initialize class and input relevant details for algorithm
<code>initialize_algorithm</code>	Initialize variables for the algorithm
<code>fit</code>	Determines cluster assignments at each iteration
<code>get_index</code>	Input: <code>nlevel</code> (integer), <code>cluster_number</code> Return: indices of data points belonging to cluster = <code>cluster_number</code> at iteration = <code>nlevel</code>
<code>plot_cluster</code>	Input: <code>nlevel</code> (integer), <code>title</code> (string), <code>xlabel</code> (string), <code>ylabel</code> (string) Plot data points showing cluster assignments at a single iteration ( <code>nlevel</code> ) (clusters distinguished by color) See <a href="#">UnsupervisedML/Examples/Section02/MatplotlibAdvanced.ipynb</a>
<code>plot_cluster_animation</code>	Input: <code>nlevel</code> (integer), <code>interval</code> (integer), <code>title</code> (string), <code>xlabel</code> (string), <code>ylabel</code> (string) Create animation showing data points and evolution of cluster assignments See <a href="#">UnsupervisedML/Examples/Section02/MatplotlibAdvanced.ipynb</a>

# Hierarchical Clustering Code Design

Additional design considerations:

- (1) Design obviously accounts for Hierarchical Clustering Algorithm
- (2) Design takes into account visual representation of clustering algorithm

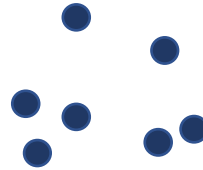
Create a hierarchical class derived from `clustering_base`



# Hierarchical Clustering Code Design

Cluster colouring for visualization purposes

- Initially, all points are in their own clusters – assign label -1, so all data points coloured the same



- Cluster assignment changes from -1 when a cluster of 2 data points is created. Leads to change in colour of data points in visual representation

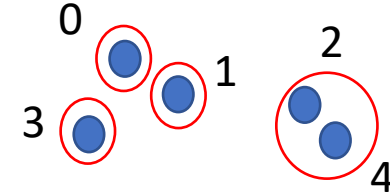


- When combining clusters, smaller cluster is “added” to larger cluster

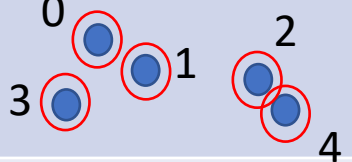
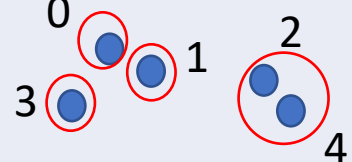
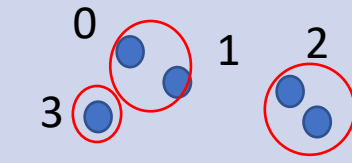
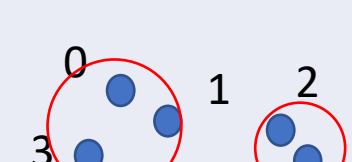



# Hierarchical Clustering: Additional Variables

- `list_cluster` is a list of indices for each cluster
  - Example `list_cluster = [[0], [1], [2,4], [3], []]`
  - Data point 0 is a cluster
  - Data point 1 is a cluster
  - Data points 2,4 are in a cluster
  - Data point 3 is in a cluster
- `list_clustermean` is a list of current cluster means
  - `list_clustermean[i]` is mean of data points in `list_cluster[i]`



# Hierarchical Clustering: Example

Clusters	Description	
	Level 0: list_cluster: each data point is its own cluster clustersave[0]: set all labels to -1	$\text{list\_cluster} = [[0], [1], [2], [3], [4]]$ $\text{clustersave}[0] = [-1, -1, -1, -1, -1]$
	Level 1: Clusters [2] and [4] are closest so combine Assign label 2 to points 2 and 4 (smallest index value)	$\text{list\_cluster} = [[0], [1], [2,4], [3], []]$ $\text{clustersave}[1] = [-1, -1, 2, -1, 2]$
	Level 2: Clusters [0] and [1] are closest so combine Assign label 0 to points 0 and 1 (smallest index value)	$\text{list\_cluster} = [[0,1], [], [2,4], [3], []]$ $\text{clustersave}[2] = [0, 0, 2, -1, 2]$
	Level 3: Clusters [0,1], [3] are closest so append [3] to [0,1] (append cluster with smaller number of points to one with larger number of points) Assign label of larger cluster to new point(s)	$\text{list\_cluster} = [[0,1,3], [], [2,4], [], []]$ $\text{clustersave}[3] = [0, 0, 2, 0, 2]$
	Level 4: Combine final 2 clusters into single cluster (append cluster with smaller number of points to one with larger number of points) Assign label of larger cluster to new point(s)	$\text{list\_cluster} = [[0,1,3,2,4], [], [], [], []]$ $\text{clustersave}[4] = [0, 0, 0, 0, 0]$

# hierarchical class – Key Methods

Method	Input	Description
initialize_algorithm		Initialize variables self.clustersave, self.list_cluster, self.list_clustermean Return: nothing
fit	X (2d numpy array)	Performs hierarchical clustering algorithm Return: nothing
combine_closest_clusters		Combine closest clusters and update self.list_cluster Return: nothing
update_cluster_assignment	idx1 (list)	Update cluster assignments in self.clustersave to account for creation of new cluster with points in idx1 Return: nothing

# Additional Functions

Method	Input	Description
create_dataset	n_sample (integer) case (string)	Use sklearn functionality to create 2d dataset for clustering with n_sample data points and for case equals one of “blobs”, “varied_blobs1”, “varied_blobs2”, “aniso”, “noisy_moons”, “noisy_circles” Return: dataset X See UnsupervisedML/Examples/Section02/SklearnDatasets.ipynb

# Unsupervised Machine Learning with Python

# Section 4.3: Hierarchical Clustering Code Walkthrough

# 4.3 Hierarchical Clustering Code Walkthrough

Code located at:

- UnsupervisedML/Code/Programs

Files to Review	Description
create_dataset_sklearn.py	Create test 2d datasets for clustering
clustering_base.py	Base class for clustering codes
hierarchical.py	Class for hierarchical clustering derived from clustering_base
driver_hierarchical.py	Driver for hierarchical clustering

Course Resources at:

- <https://github.com/satishchandrareddy/UnsupervisedML/>
- Stop video if you want to do coding yourself first