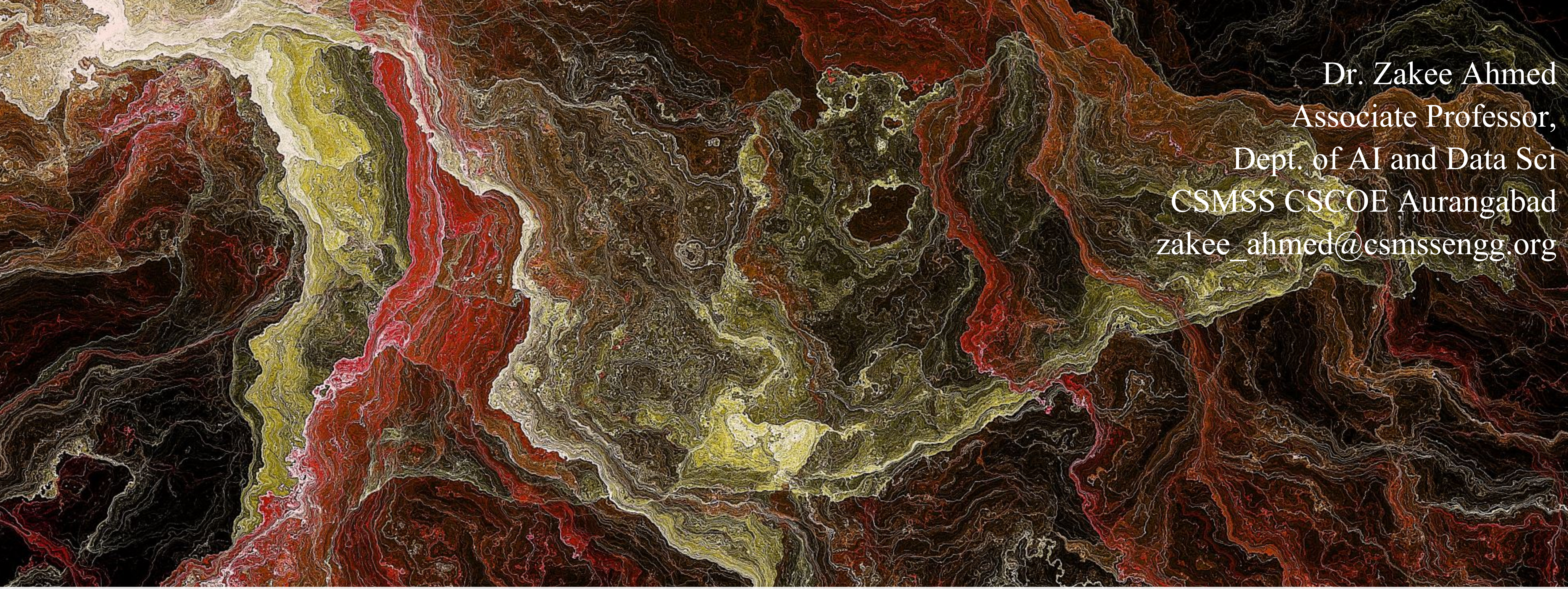


Dr. Zakee Ahmed
Associate Professor,
Dept. of AI and Data Sci
CSMSS CSCOE Aurangabad
zakee_ahmed@csmssengg.org

ADVANCED MACHINE LEARNING BTAIC602

Unit 1: Unsupervised Learning

SYLLABUS
UNIT NO 1: UNSUPERVISED LEARNING

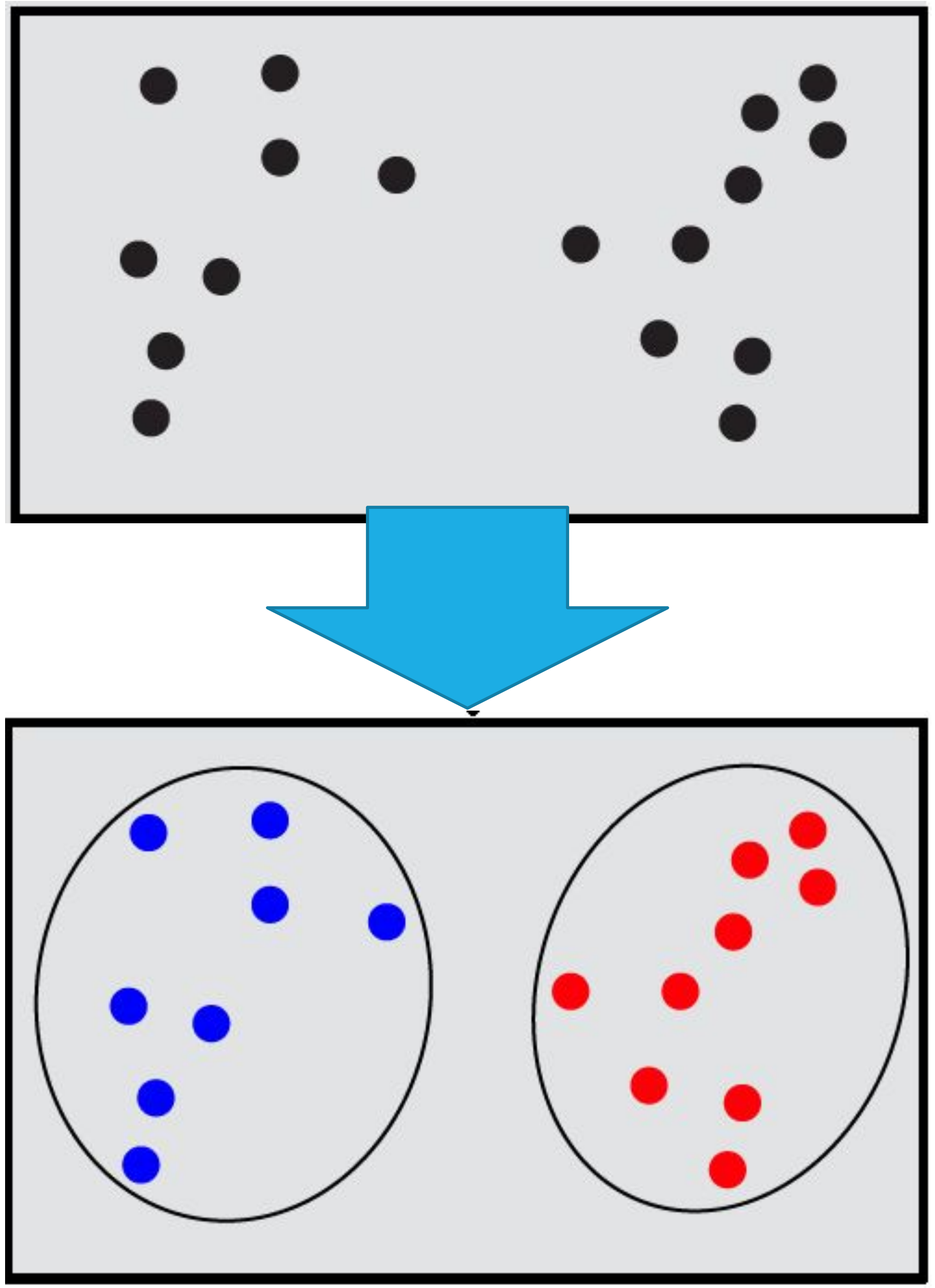
1. **Unsupervised Learning – 1:** Introduction to Unsupervised Learning, Introduction to Clustering, Using K-means for Flat Clustering, KMeans Algorithm, Using KMeans from Sklearn, Implementing Fit & Predict Functions, Implementing K-Means Class
2. **Unsupervised Learning – 2:** How to choose Optimal K, Silhouette algorithm to choose K, Introduction to K Medoids, K Medoids Algorithm, Introduction to Hierarchical Clustering, Top down/Divisive Approach, Bottom up/Divisive Approach Principal Component Analysis
3. **PCA – 1:** Intuition behind PCA, Applying PCA to 2D data, Applying PCA on 3D data, Math behind, PCA, Finding Optimal Number of Features, Magic behind PCA, Dimensionality reduction
4. **PCA – 2:** PCA on Images, PCA on Olevitti Images, Reproducing Images, Eigenfaces, Classification of LFW Images

INTRODUCTION TO UNSUPERVISED LEAR

- ***Unsupervised Machine Learning*** uses machine learning algorithms to analyze and cluster unlabeled datasets.
- We do not provide the machine with ***labeled data***, and the machine is expected to derive structure from the data all on its own.
- Generally, it is used as a process to find meaningful structure, explanatory underlying processes, generative features, and groupings inherent in a set of examples
- There are many forms of ***Unsupervised Machine Learning***, though the main form of unsupervised machine learning is '***CLUSTERING***'.

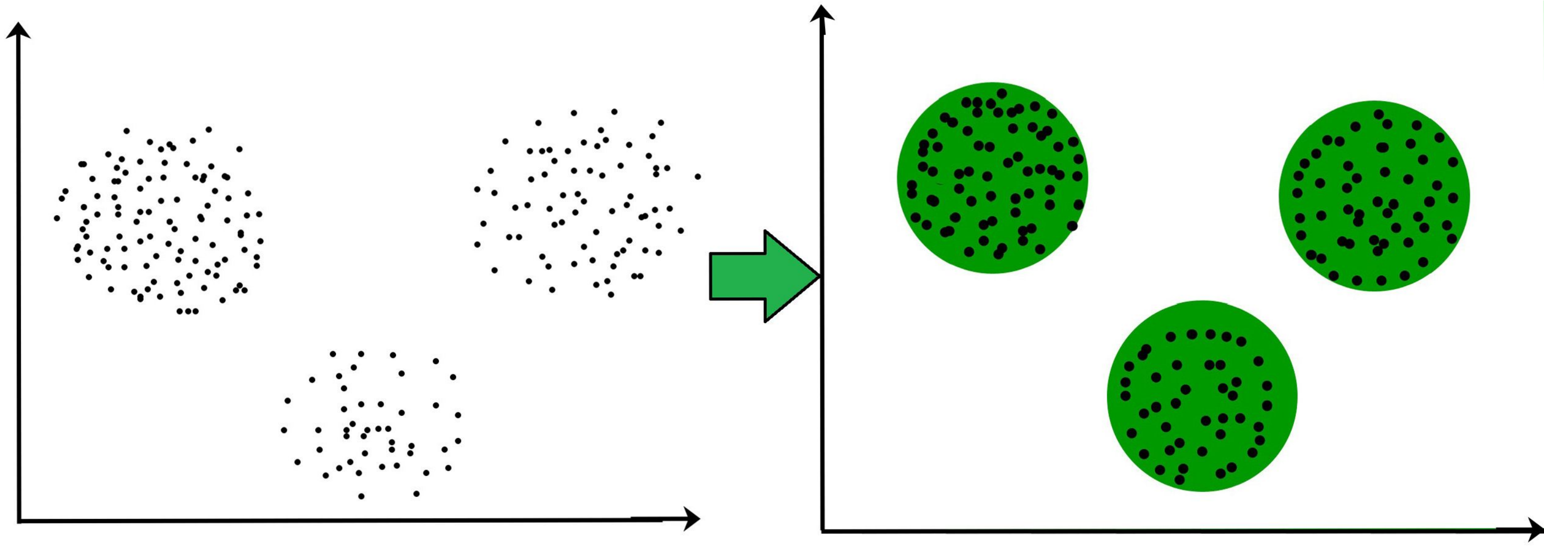
INTRODUCTION TO CLUSTERING

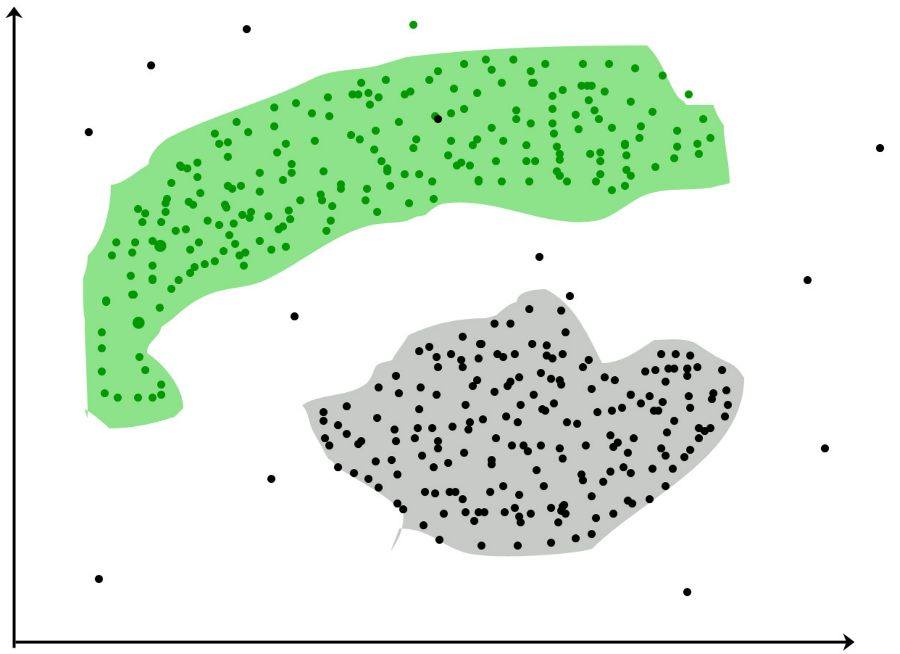
- Clustering or Cluster analysis is the method of grouping the entities based on similarities
- A cluster is a collection of objects which are "similar" amongst themselves and are "dissimilar" to the objects belonging to a different cluster
- It is defined as an unsupervised learning problem that aims to make training data with a given set of inputs but without any target values, or labels
- In clustering, the machine learns the attributes and trends by itself without any provided input-output mapping.
- The clustering algorithms extract patterns and inferences from the type of data objects and then make discrete classes of clustering them suitably.



EXAMPLE

- The data points in the graph, clustered together can be classified into one single group.
- We can distinguish the clusters, and we can identify that there are 3 clusters as shown in the picture
- It is not necessary for clusters to be spherical
- There are no criteria for good clustering, it depends on the user, what is the criteria they may use which satisfy their need.





CLUSTERING VS CLASSIFICATION

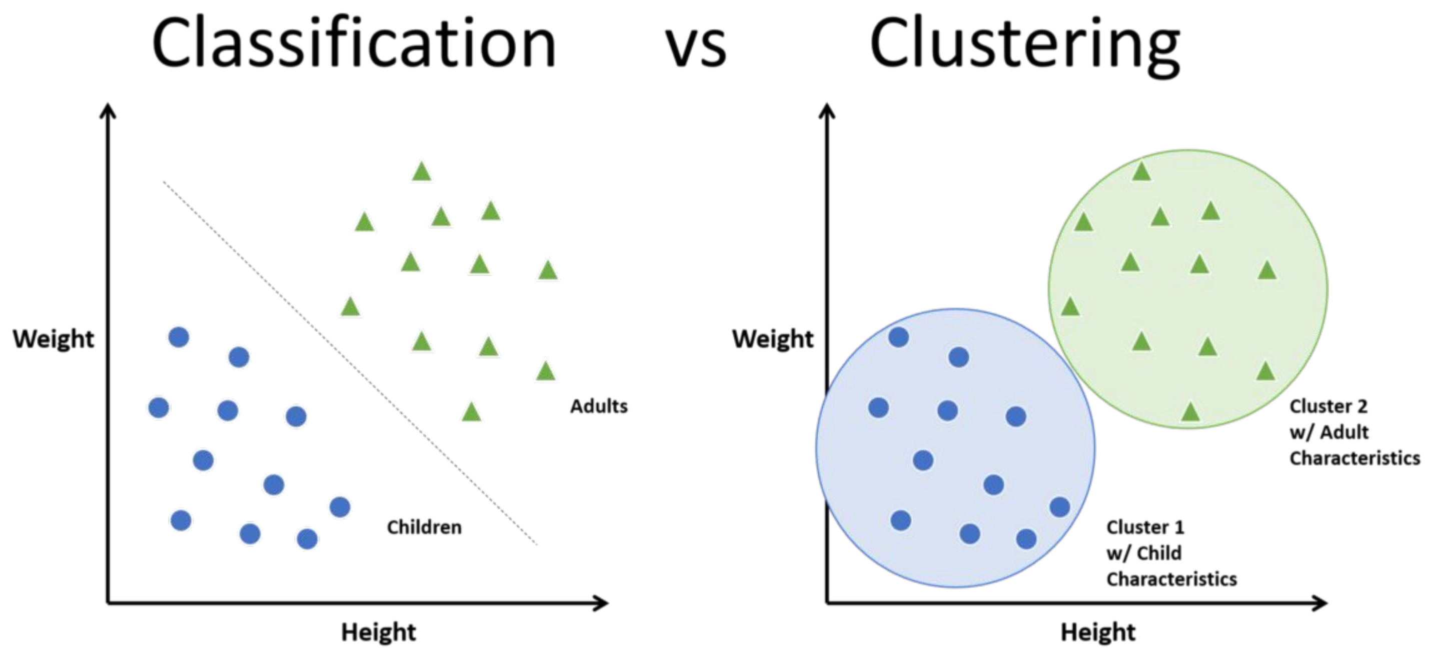
The difference between clustering and classification may not seem great at first.

As, in both cases we have a partition of a set of documents into groups.

But as we will see the two problems are fundamentally different.

Classification is a form of supervised learning our goal is to replicate a categorical distinction that a human supervisor imposes on the data.

In unsupervised learning, of which clustering is the most important example, we have no such teacher to guide us.


Classification
vs
Clustering
Weight
Height
Adults
Children
Weight
Height
Cluster 2 w/ Adult Characteristics
Cluster 1 w/ Child Characteristics

CLUSTERING VS CLASSIFICATION

Parameter	CLASSIFICATION	CLUSTERING
Type	used for supervised learning	used for unsupervised learning
Basic	process of classifying the input instances based on their corresponding class labels	grouping the instances based on their similarity without the help of class labels
Need	it has labels so there is need of training and testing dataset for verifying the model created	there is no need of training and testing dataset
Complexity	more complex as compared to clustering	less complex as compared to classification
Example Algorithms	Logistic regression, Naive Bayes classifier, Support vector machines, etc.	k-means clustering algorithm, Fuzzy c-means clustering algorithm, Gaussian (EM) clustering algorithm, etc.

TYPES OF CLUSTERING METHODS

The major fundamental clustering methods are:

1. Partitioning Methods
2. Hierarchical Methods
3. Density-based Methods
4. Grid Based Methods

1. PARTITIONING METHODS

Given a set of n objects, a partitioning method constructs k partitionof the data

Where each partition represents a cluster and k <= n

That is, it divides the data into k groups such that each group must contain atleast one object

The basic partitioning methods typically adopt exclusive cluster separation

That is the object must belongs to exactly one group

Most popular partitioning methods are:

1. K-Means Clustering

2. K-Medoids algorithm

2. HIERARCHICAL METHODS

A hierarchical method creates a hierarchical decomposition of the given set of data objects

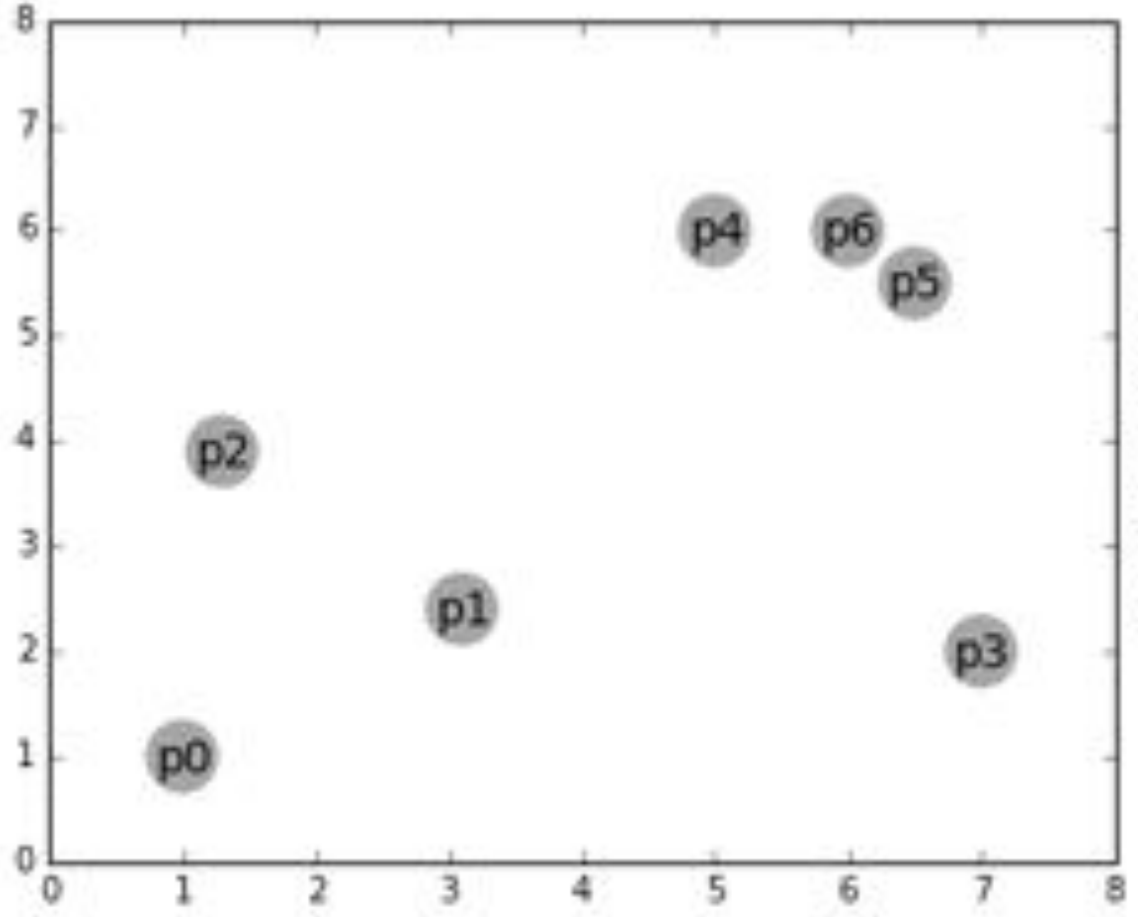
A hierarchical method can be classified into ***agglomerative*** or ***divisive*** approach

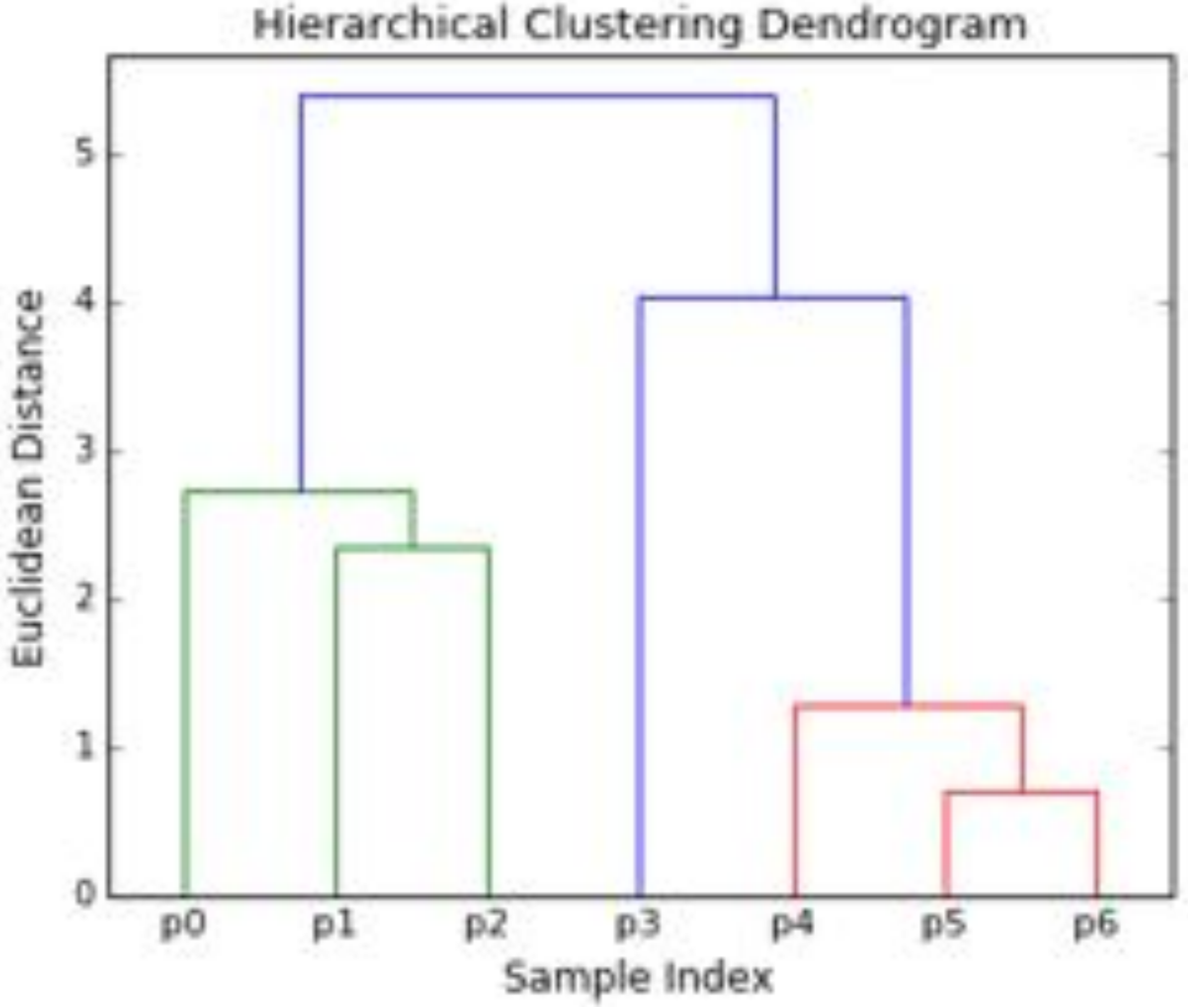
The ***agglomerative*** approach also called ***bottom up*** approach, starts with each object forming a separate group

It then successively merges the objects or groups close to one another, until all groups are merged into one .

The ***divisive*** approach, also called top-down approach, starts with all the objects in the same cluster

It then split the cluster into smaller clusters, until each object is one cluster


p0
p1
p2
p3
p4
p5
p6


Hierarchical Clustering Dendrogram
Euclidean Distance
Sample Index
p0
p1
p2
p3
p4
p5
p6

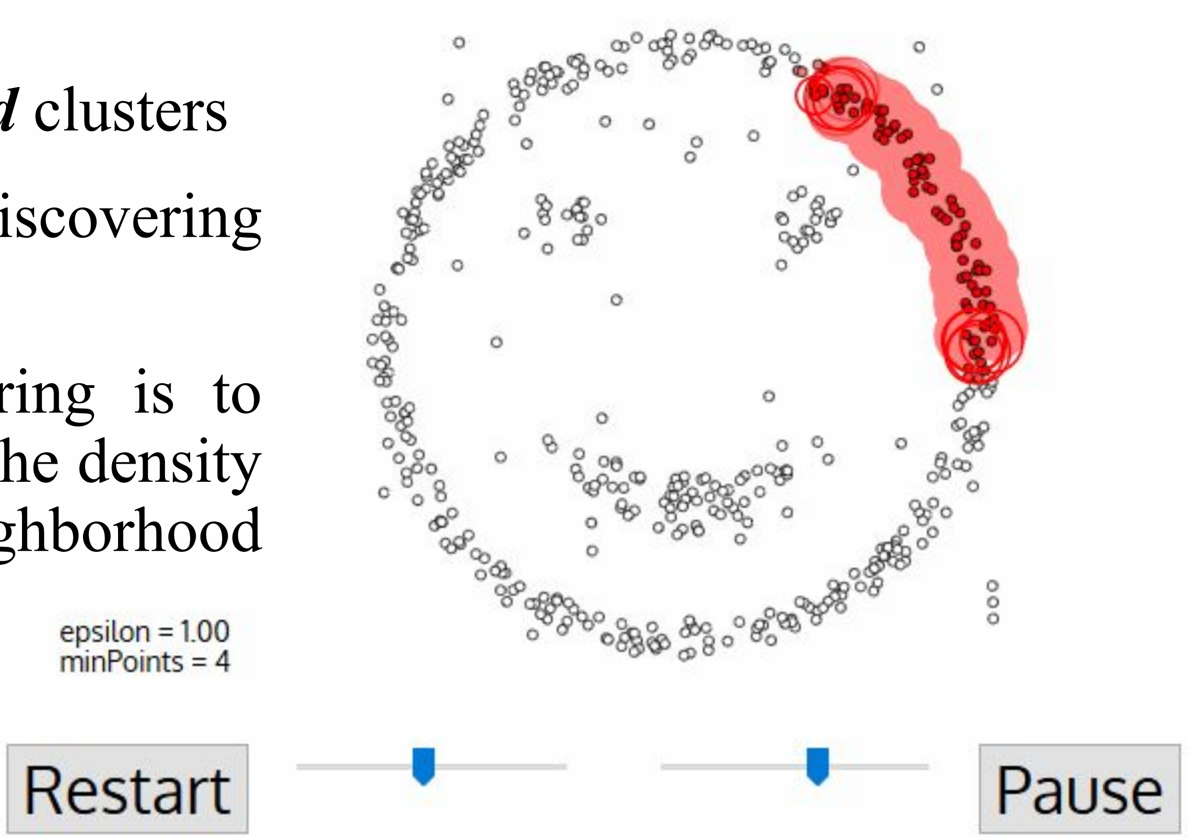
3. DENSITY-BASED METHODS

Most partitioning methods cluster objects based on distance between objects

Such methods can find only ***spherical shaped*** clusters

Such methods encounter difficulty in discovering clusters of ***arbitrary shapes***

The general idea of density based clustering is to continue growing a given cluster as long as the density (number of objects or data points) in the neighborhood exceeds some threshold

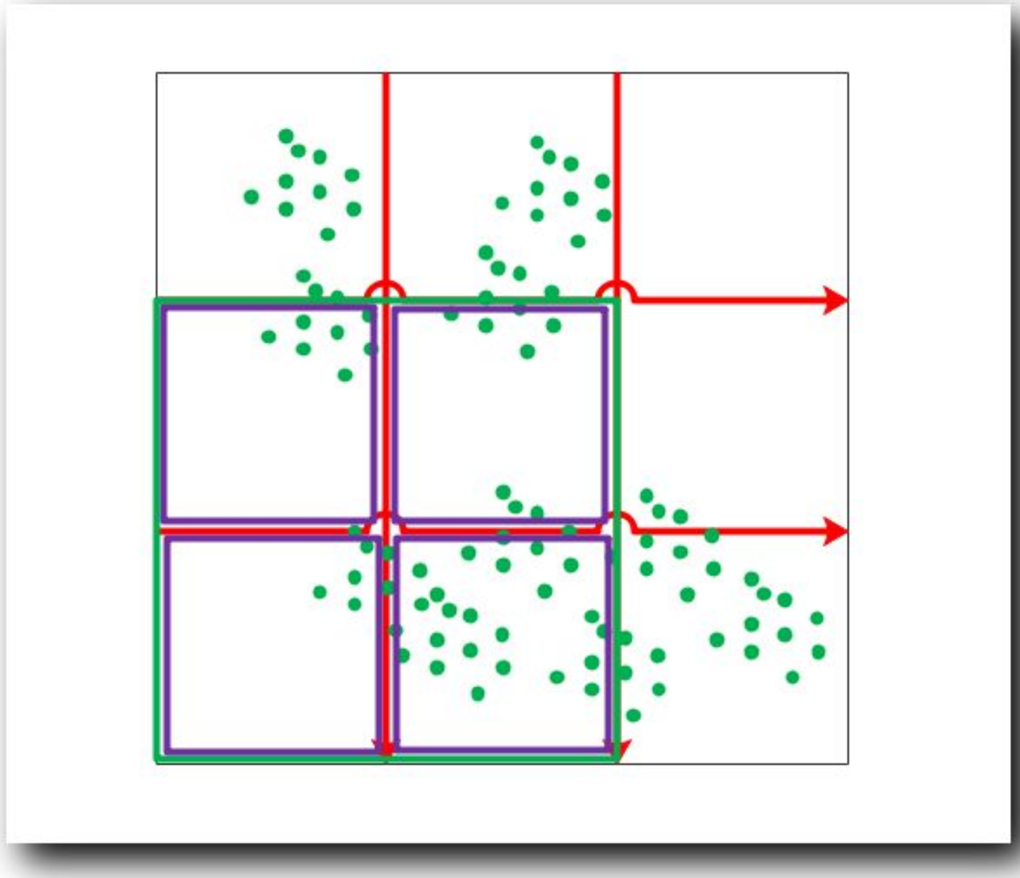

epsilon = 1.00
minPoints = 4

4. GRID BASED METHODS

Grid based methods quantize the object space into a finite number of cells that form a grid structure

All the clustering operations are performed on the grid structure

These are seen as Faster Clustering Methods



CLUSTERING METHODS- COMPARISON

Method	General Characteristics
Partitioning Methods	- Find Mutually Exclusive clusters of spherical shape - Distance based - May use Mean or Medoid to represent cluster center - Effective for small or medium sized data sets
Hierarchical Methods	- Clustering is a hierarchical decomposition - Cannot correct erroneous merges or splits
Density-based Methods	- Can find arbitrarily shaped clusters - Clusters are dense regions of objects in space that are separated by low density regions - Cluster Density: each point must have a minimum number of points within its neighborhood - May filter out outliers
Grid- based Methods	- Use multiresolution grid data structure - Fast processing time

PARTITIONING METHODS: USING K-MEANS FOR FLAT CLUSTERING

K-MEANS CLUSTERING

Suppose a Data set D, contains n objects in Euclidean space

Partitioning methods distribute the objects in D into k clusters, $C_1, \ldots C_k$, that is $C_i \subset D$, and $C_i \cap C_j = \emptyset$ for $(1 \leq i, j \leq k)$

An objective function is used to assess the partitioning quality so that objects within a cluster are similar to one another but dissimilar to objects in other clusters

That is objective function aims for high intra-cluster similarity and low inter-cluster similarity

K-MEANS CLUSTERING

A centroid based partitioning technique uses the centroid of a cluster C_i to represent that cluster

Conceptually the centroid of a cluster is its center point

The centroid can be defined in various ways such as by the ***mean*** or ***medoid*** of the objects assigned to the cluster

The difference between an object $p \epsilon C_i$ and c_i the representation of cluster is measured by $dist(p,c_i)$,

Where $dist(x,y)$ is the Euclidean distance between two points x and y

K-MEANS CLUSTERING - *WITHIN-CLUSTER VARIATION*

The quality of cluster C_i can be measured by the *within-cluster variation*:

$$E = \sum_{i=1}^{k} \sum_{p \epsilon C_i} dist(p, c_i)^2$$

For Each object in each cluster, the distance from the object to its cluster centre is squared, and distances are summed up

This objective function tries to make the resulting k clusters as compact and as separate as possible

Where, E is sum of all squared error for all objects in the data set

p is the point in space representing a given object

c_i is the centroid of cluster $\boldsymbol{C}_i$

K-MEANS ALGORITHM

WORKING

The k means algorithm defines the centroid of a cluster as the mean value of the points within the cluster

1. it randomly selects k of the objects in D, each of which initially represents a cluster mean or center
2. For each remaining objects, an object is assigned to the cluster to which it is most similar, based on Euclidean Distance between object and cluster mean
3. The k-mean algorithm then iteratively improves the within cluster variation
4. For each cluster, it computes the new mean the objects assigned to the cluster in the previous iteration
5. All the objects are then reassigned using the updated means as the new cluster centers
6. The iterations continue until the assignment is stable, that is, the cluster formed in the current round are the same as those formed in the previous round

K-MEANS ALGORITHM

Algorithm

- The K-means algorithm for partitioning, where each cluster’s center is represented by the mean value of the objects in the cluster

Input

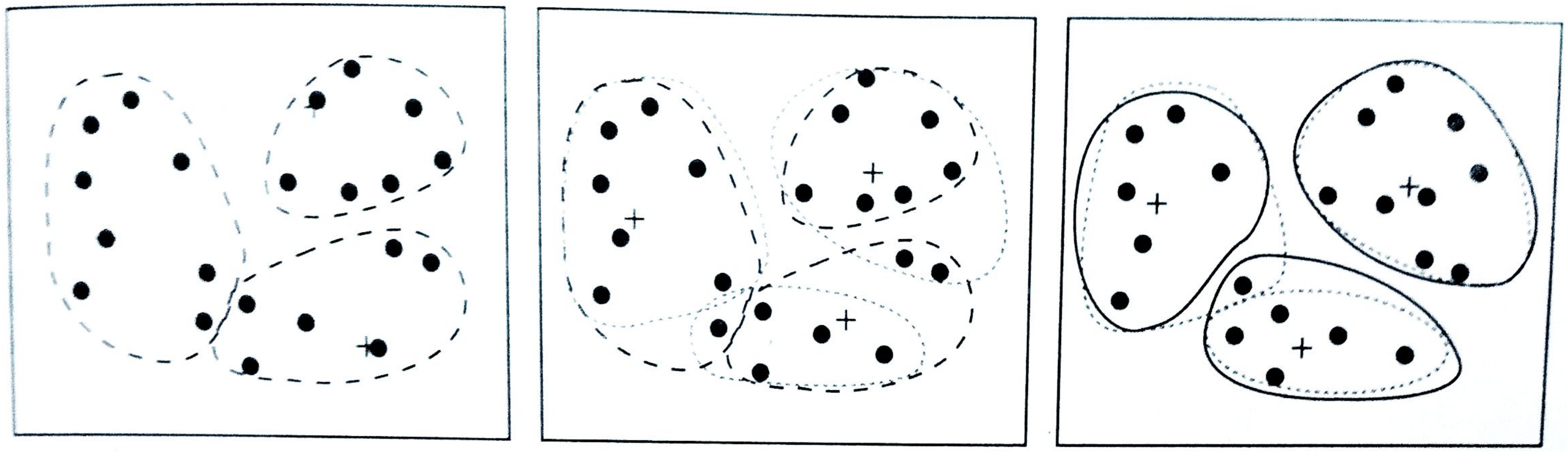
- K: the number of clusters
- D: a Data set consisting of n Objects

Output:

- A set of k Clusters

Method:

```
1.  Arbitrarily choose k objects from D as the initial cluster centers
2.  Repeat:
3.      Calculate the distance from each data point to each centroid.
4.      Assign the data points to the closest centroid.
5.      Recalculate the centroids as the mean of the points in their cluster.
6.  Until no change
```



igure 10.3 Clustering of a set of objects using the *k*-means method; for (b) update cluster centers and reassign objects accordingly (the mean of each cluster is marked by a +).

EXAMPLE

Suppose that we have following items to cluster: {2,4,10,12,3,20,30,11,25}

And suppose that k = 2

We Assign means to the first two values m1=2, m2=4

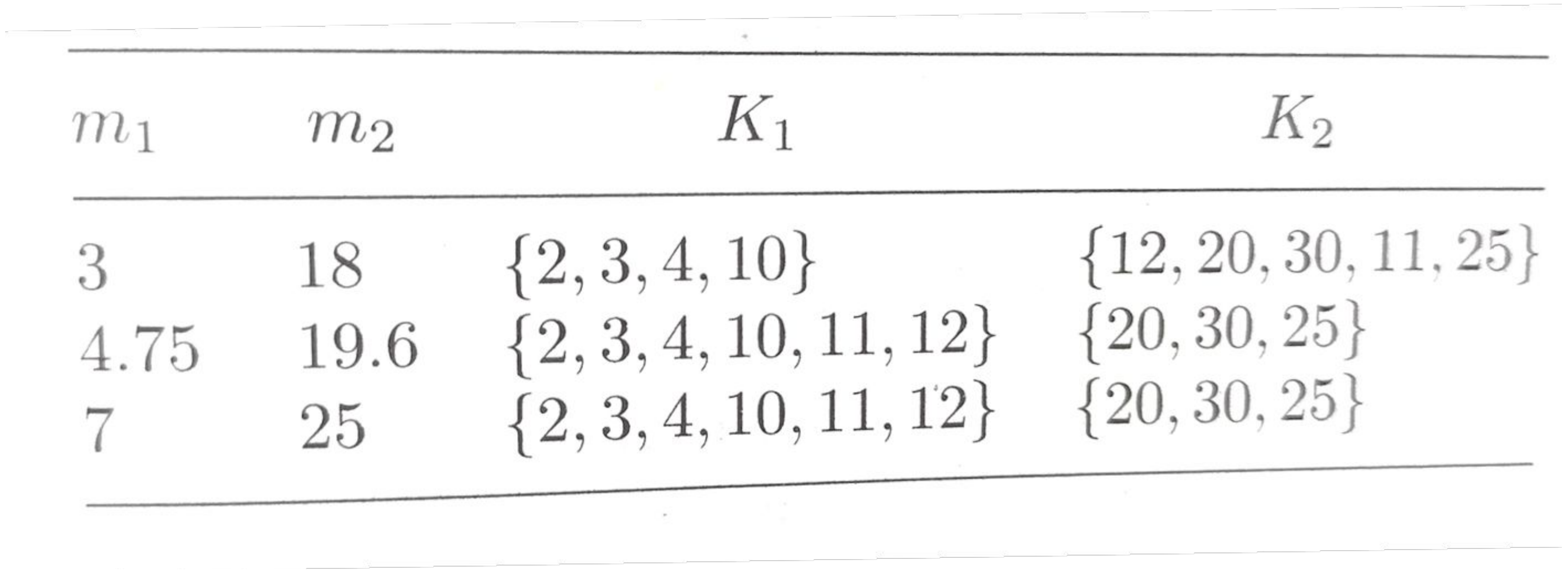
Using Euclidean distance we find that initially K1={2,3}, K2={4,10,12,20,30,11,25}

The value 3 is equally closed to both means, so arbitrarily choose K1

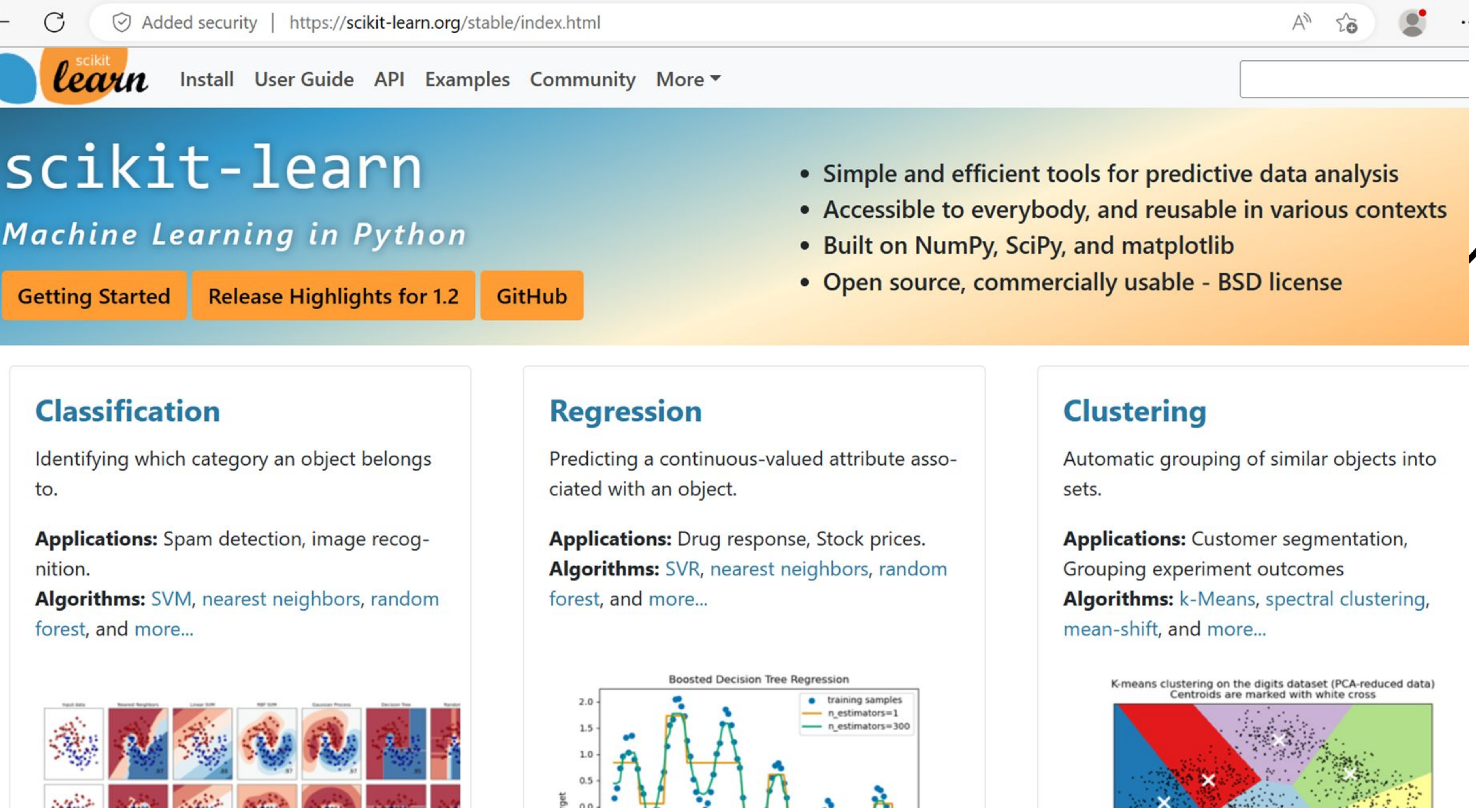
We then re-calculate the means to get m1=2.5 and m2=16

We again make assignments to cluster to get K1={2,3,4} and K2={10,12,20,30,11,25}

EXAMPLE



m_1	m_2	K_1	K_2
3	18	$\{2,3,4,10\}$	$\{12,20,30,11,25\}$
4.75	19.6	$\{2,3,4,10,11,12\}$	$\{20,30,25\}$
7	25	$\{2,3,4,10,11,12\}$	$\{20,30,25\}$


Added security
https://scikit-learn.org/stable/index.html
scikit learn
Install
User Guide
API
Examples
Community
More
Getting Started
Release Highlights for 1.2
GitHub
Boosted Decision Tree Regression
training samples
n_estimators=1
n_estimators=300
K-means clustering on the digits dataset (PCA-reduced data)
Centroids are marked with white cross

scikit-learn

Machine Learning in Python

- Simple and efficient tools for predictive data analysis
- Accessible to everybody, and reusable in various contexts
- Built on NumPy, SciPy, and matplotlib
- Open source, commercially usable - BSD license

Classification

Identifying which category an object belongs to.

Applications: Spam detection, image recognition.
Algorithms: SVM, nearest neighbors, random forest, and more...

Regression

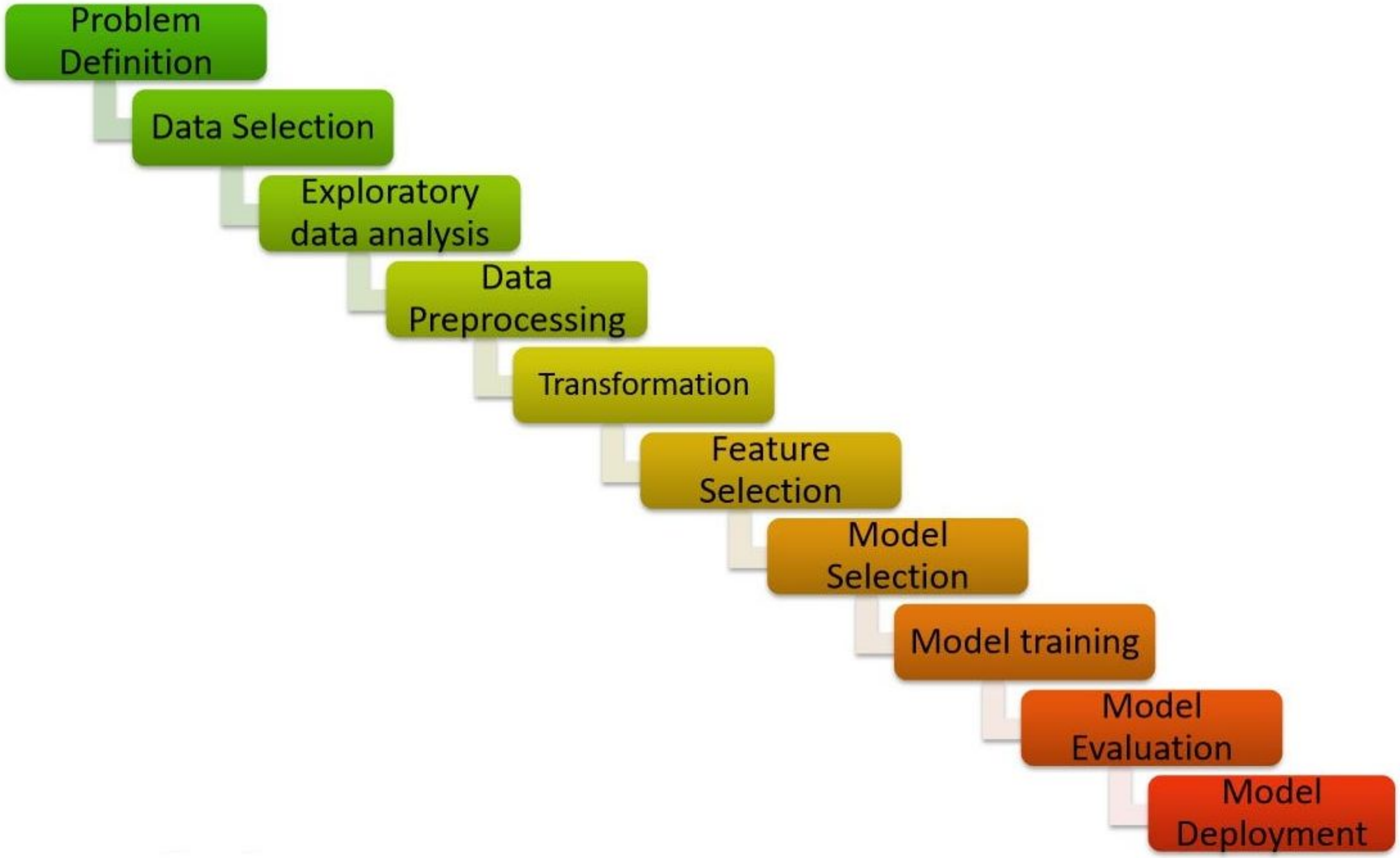
Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.
Algorithms: SVR, nearest neighbors, random forest, and more...

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, and more...


Problem Definition
Data Selection
Exploratory data analysis
Data Preprocessing
Transformation
Feature Selection
Model Selection
Model training
Model Evaluation
Model Deployment

USING KMEANS FROM SKLEARN

Clustering of unlabeled data can be performed with the module sklearn.cluster

K-Means algorithm can be implemented using the class ***KMeans***

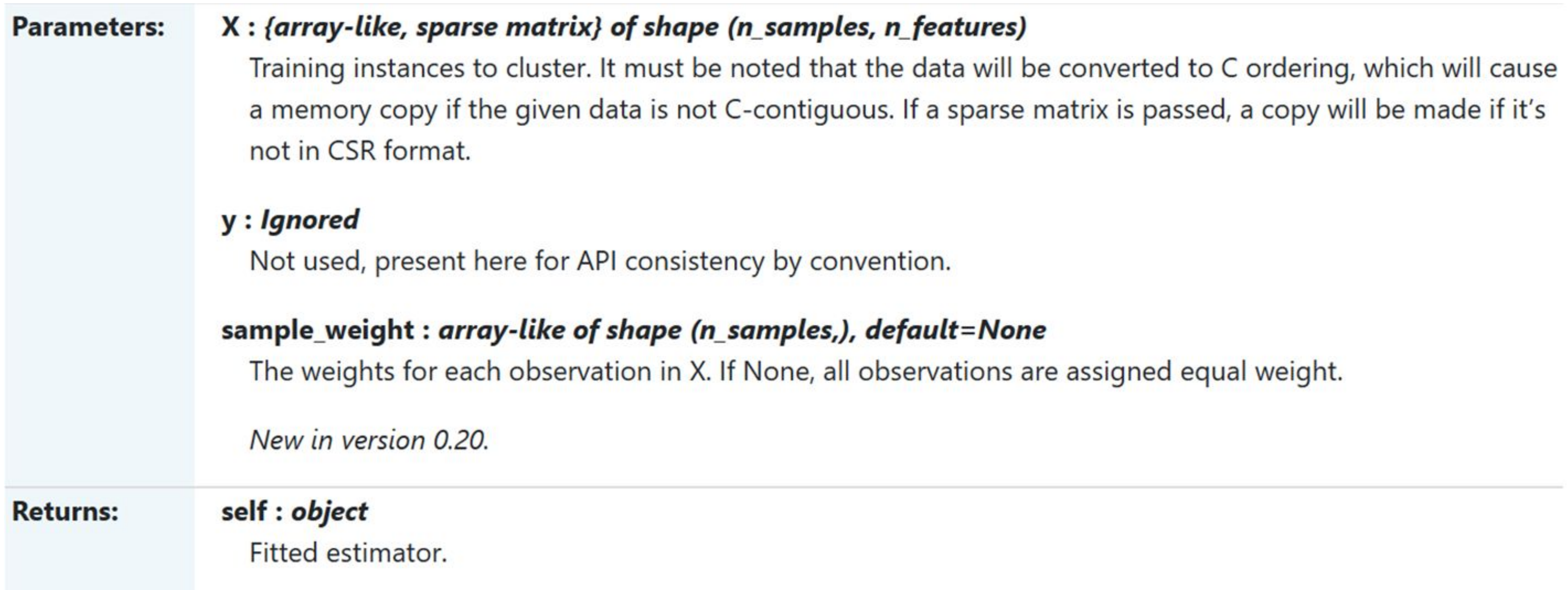
The Kmeans class has many important methods, constructors and attributes, which are used to implement K-Means Algorithm

Method	Description
fit(**X[, y, sample_weight]**)	Compute k-means clustering.
fit_predict(**X[, y, sample_weight]**)	Compute cluster centers and predict cluster index for each sample.
fit_transform(**X[, y, sample_weight]**)	Compute clustering and transform X to cluster-distance space.
get_feature_names_out(**[input_features]**)	Get output feature names for transformation.
get_params(**[deep]**)	Get parameters for this estimator.
predict(**X[, sample_weight]**)	Predict the closest cluster each sample in X belongs to.
score(**X[, y, sample_weight]**)	Opposite of the value of X on the K-means objective.
set_output(***[, transform]**)	Set output container.
set_params(****params**)	Set the parameters of this estimator.
transform(**X**)	Transform X to a cluster-distance space

IMPLEMENTING FIT & PREDICT FUNCTIONS

`fit`(*X*, *y=None*, *sample_weight=None*)

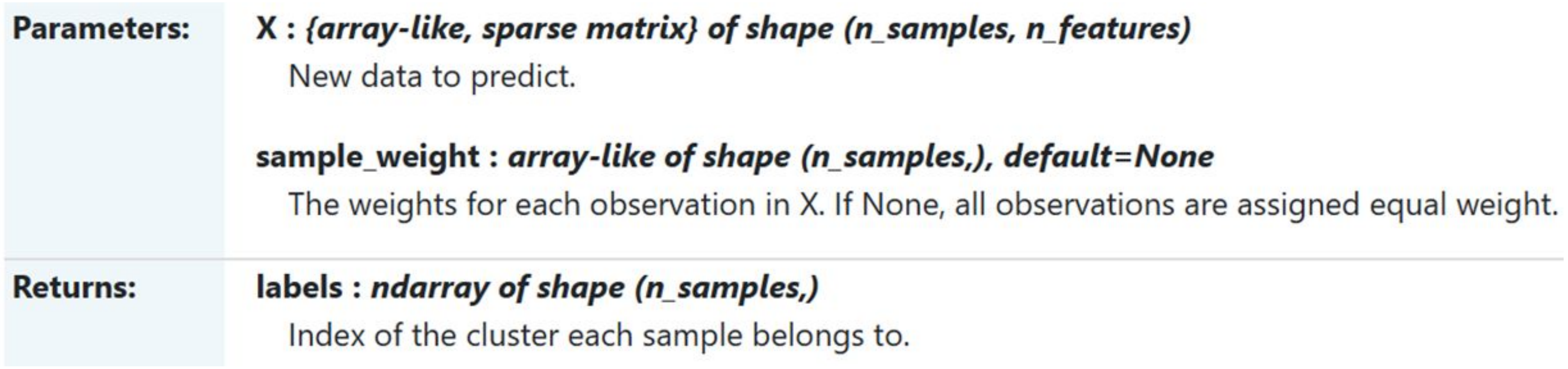
Compute k-means clustering



Parameters:	**X : *{array-like, sparse matrix} of shape (n_samples, n_features)*** Training instances to cluster. It must be noted that the data will be converted to C ordering, which will cause a memory copy if the given data is not C-contiguous. If a sparse matrix is passed, a copy will be made if it's not in CSR format. **y : *Ignored*** Not used, present here for API consistency by convention. **sample_weight : *array-like of shape (n_samples,), default=None*** The weights for each observation in X. If None, all observations are assigned equal weight. *New in version 0.20.*
Returns:	**self : *object*** Fitted estimator.

IMPLEMENTING FIT & PREDICT FUNCTIONS

predict(*X*, *sample_weight=None*) (Predict the closest cluster each sample in X belongs to)

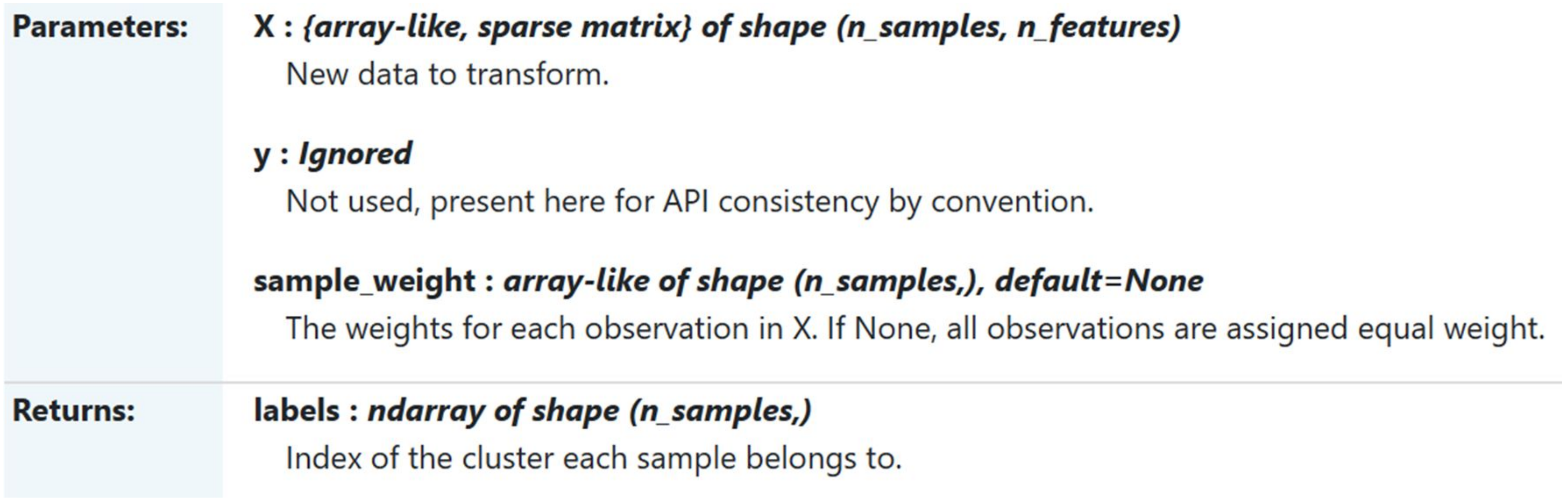


Parameters:	**X : *{array-like, sparse matrix} of shape (n_samples, n_features)*** New data to predict. **sample_weight : *array-like of shape (n_samples,), default=None*** The weights for each observation in X. If None, all observations are assigned equal weight.
Returns:	**labels : *ndarray of shape (n_samples,)*** Index of the cluster each sample belongs to.

IMPLEMENTING FIT & PREDICT FUNCTIONS

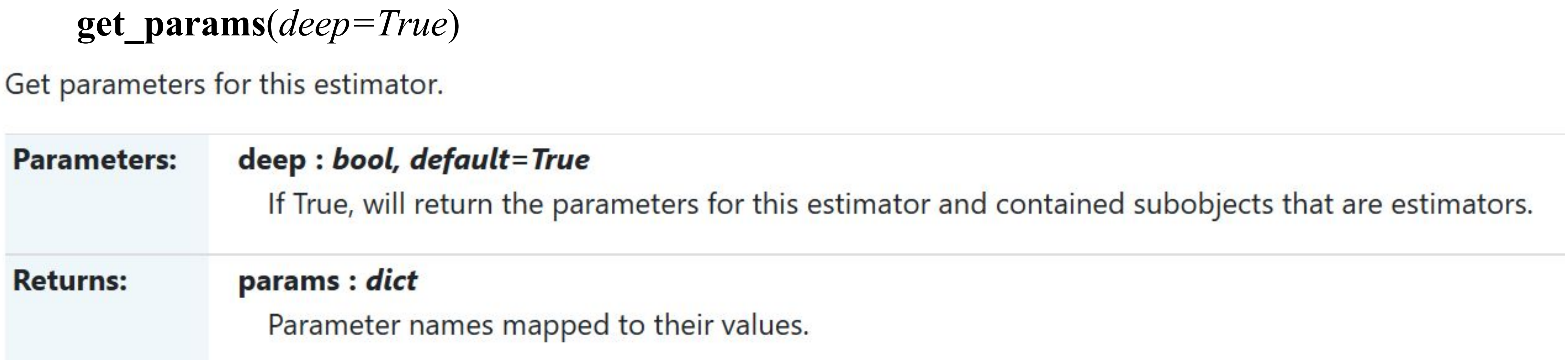
`fit_predict`(*X*, *y=None*, *sample_weight=None*)

Compute cluster centers and predict cluster index for each sample.



Parameters:	**X : *{array-like, sparse matrix} of shape (n_samples, n_features)*** New data to transform. **y : *Ignored*** Not used, present here for API consistency by convention. **sample_weight : *array-like of shape (n_samples,), default=None*** The weights for each observation in X. If None, all observations are assigned equal weight.
Returns:	**labels : *ndarray of shape (n_samples,)*** Index of the cluster each sample belongs to.

IMPLEMENTING FIT & PREDICT FUNCTIONS



get_params(*deep=True*)

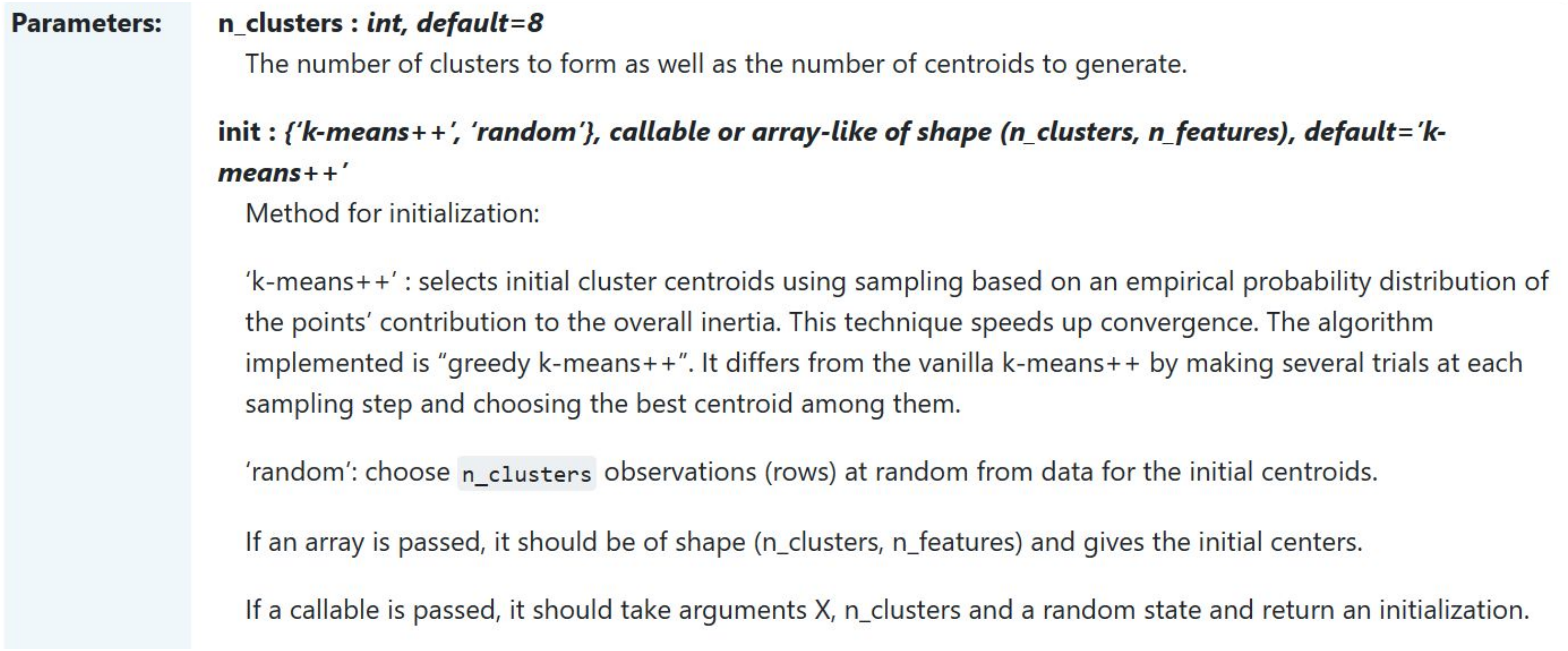
Get parameters for this estimator.

Parameters:	**deep : *bool, default=True*** If True, will return the parameters for this estimator and contained subobjects that are estimators.
Returns:	**params : *dict*** Parameter names mapped to their values.

IMPLEMENTING K-MEANS CLASS

class `sklearn.cluster.`**`KMeans`**(*n_clusters=8,*, init='k-means++', n_init='warn', max_iter=300,*

tol=0.0



Parameters:

n_clusters : *int, default=8*

The number of clusters to form as well as the number of centroids to generate.

init : *{'k-means++', 'random'}, callable or array-like of shape (n_clusters, n_features), default='k-means++'*

Method for initialization:

'k-means++' : selects initial cluster centroids using sampling based on an empirical probability distribution of the points' contribution to the overall inertia. This technique speeds up convergence. The algorithm implemented is "greedy k-means++". It differs from the vanilla k-means++ by making several trials at each sampling step and choosing the best centroid among them.

'random': choose `n_clusters` observations (rows) at random from data for the initial centroids.

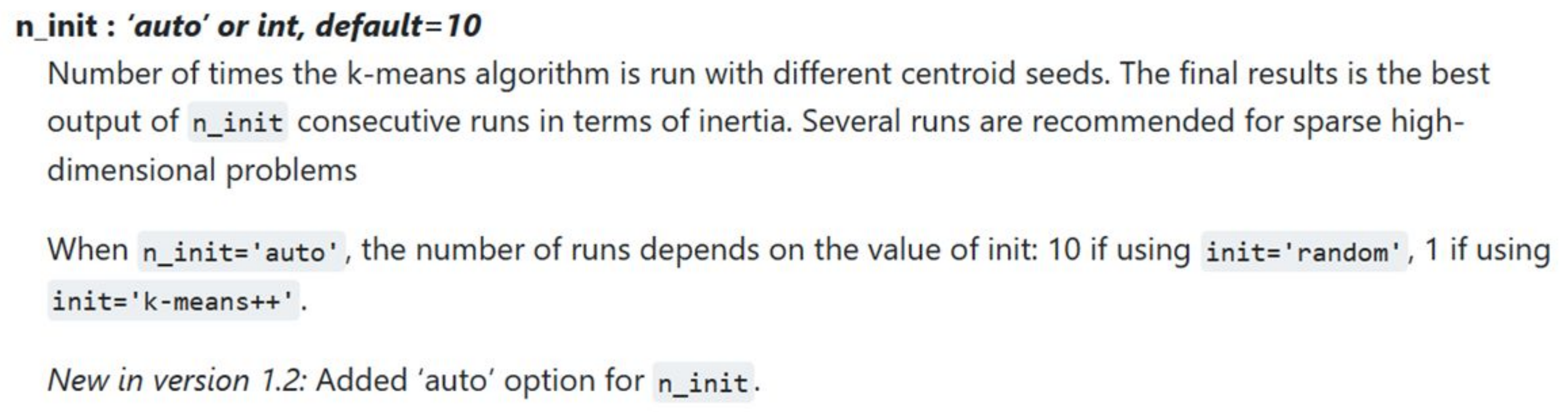
If an array is passed, it should be of shape (n_clusters, n_features) and gives the initial centers.

If a callable is passed, it should take arguments X, n_clusters and a random state and return an initialization.

IMPLEMENTING K-MEANS CLASS

class `sklearn.cluster.`**`KMeans`**(*n_clusters=8,*, init=‘k-means++', n_init='warn', max_iter=300,*

tol=0.0001, verbose=0, random_state=None, copy_x=True, algorithm='lloyd')



n_init : *‘auto’ or int, default=10*

Number of times the k-means algorithm is run with different centroid seeds. The final results is the best output of `n_init` consecutive runs in terms of inertia. Several runs are recommended for sparse high-dimensional problems

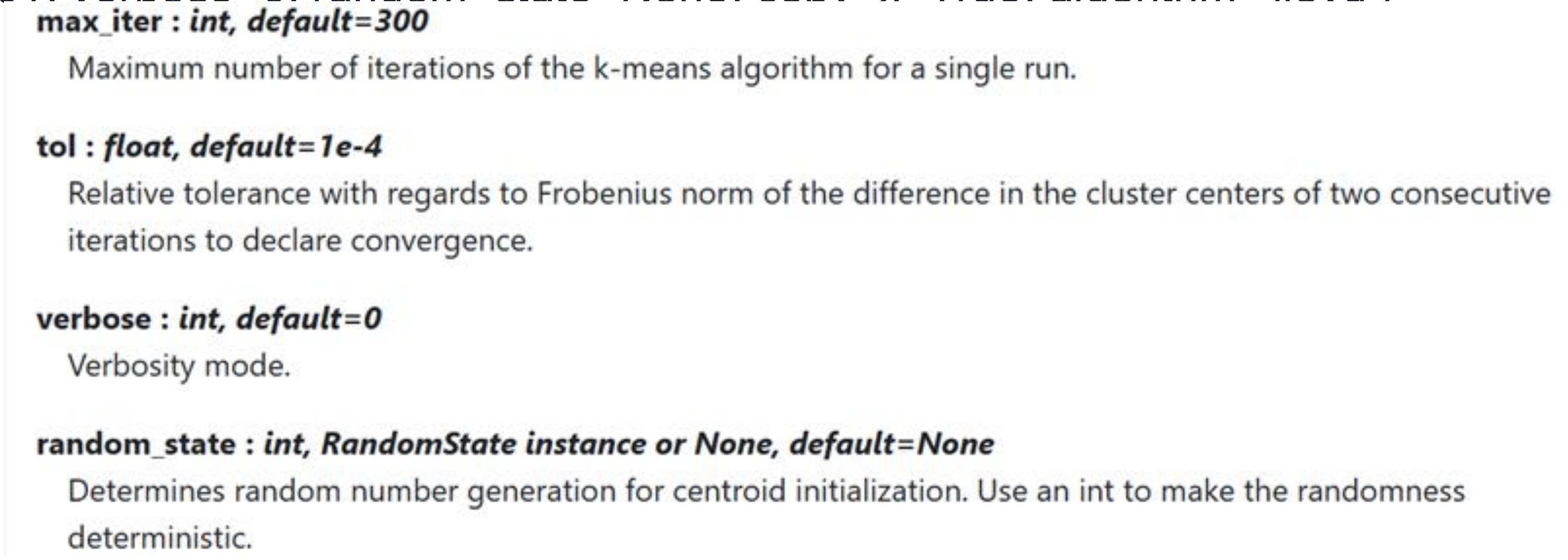
When `n_init='auto'`, the number of runs depends on the value of init: 10 if using `init='random'`, 1 if using `init='k-means++'`.

New in version 1.2: Added ‘auto’ option for `n_init`.

IMPLEMENTING K-MEANS CLASS

class `sklearn.cluster.`**`KMeans`**(*n_clusters=8*,*, *init='k-means++'*, *n_init='warn'*, *max_iter=300*,

tol=0.0001, *verbose=0*, *random_state=None*, *copy_x=True*, *algorithm='lloyd'*)



max_iter : *int, default=300*

Maximum number of iterations of the k-means algorithm for a single run.

tol : *float, default=1e-4*

Relative tolerance with regards to Frobenius norm of the difference in the cluster centers of two consecutive iterations to declare convergence.

verbose : *int, default=0*

Verbosity mode.

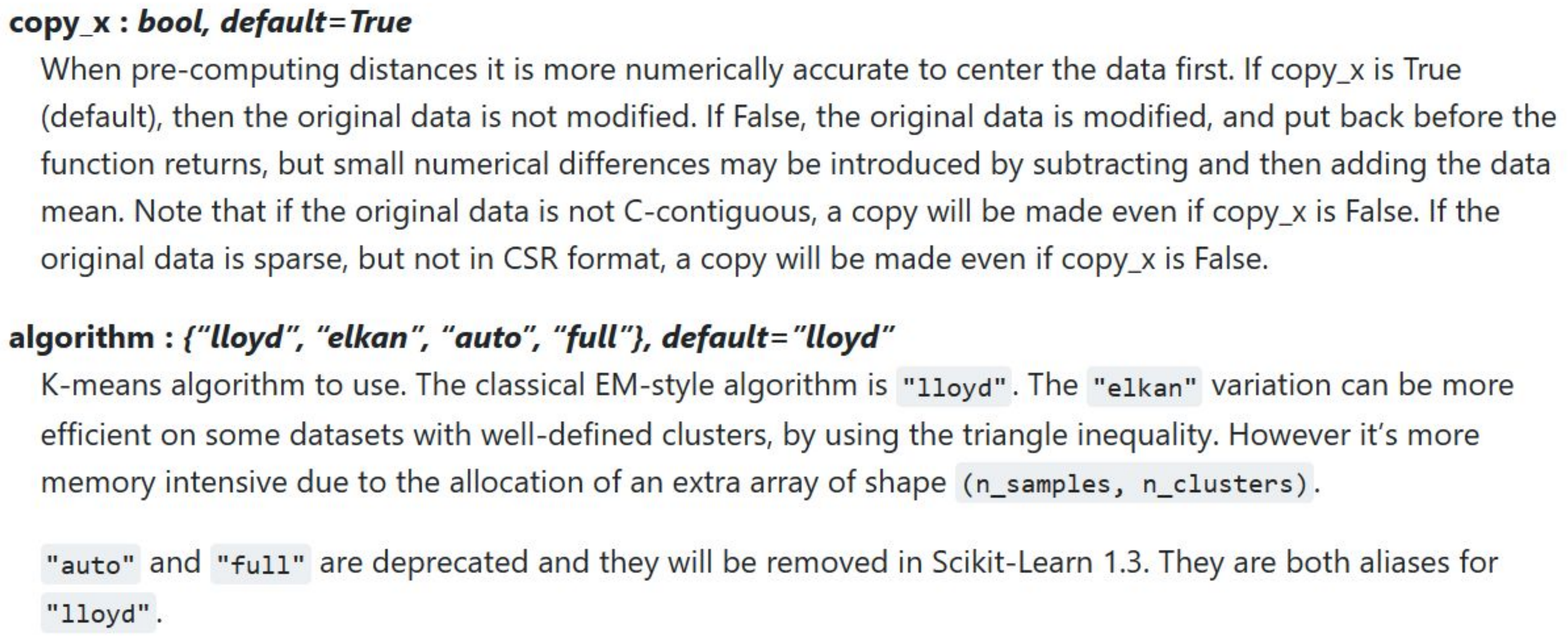
random_state : *int, RandomState instance or None, default=None*

Determines random number generation for centroid initialization. Use an int to make the randomness deterministic.

IMPLEMENTING K-MEANS CLASS

class `sklearn.cluster.`**`KMeans`**(*n_clusters=8*,***, *init='k-means++'*, *n_init='warn'*, *max_iter=300*,

tol=0.0001, verbose=0, random_state=None, copy_x=True, algorithm='lloyd')



copy_x : *bool, default=True*

When pre-computing distances it is more numerically accurate to center the data first. If copy_x is True (default), then the original data is not modified. If False, the original data is modified, and put back before the function returns, but small numerical differences may be introduced by subtracting and then adding the data mean. Note that if the original data is not C-contiguous, a copy will be made even if copy_x is False. If the original data is sparse, but not in CSR format, a copy will be made even if copy_x is False.

algorithm : *{"lloyd", "elkan", "auto", "full"}, default="lloyd"*

K-means algorithm to use. The classical EM-style algorithm is `"lloyd"`. The `"elkan"` variation can be more efficient on some datasets with well-defined clusters, by using the triangle inequality. However it's more memory intensive due to the allocation of an extra array of shape `(n_samples, n_clusters)`.

`"auto"` and `"full"` are deprecated and they will be removed in Scikit-Learn 1.3. They are both aliases for `"lloyd"`.

USING KMEANS FROM SKLEARN

SKLearn

We usually follow following 4 steps:

1. import the library:
```
from sklearn.cluster import KMeans
```

2. initialize the model
```
model = KMeans(2)
```

3. Train the model with model.fit(...)
```
model = model.fit(df[ [“Label1“, Label2”, “Label3”] ] )
```

4. predict the outcome
```
df["cluster"] = model.predict(df[[“Label1“, Label2”, “Label3”] ] )
```

EXAMPLE

https://www.kaggle.com/code/funxexcel/p1-sklearn-k-means-example/notebook

QUESTIONS?

Thank you!

REFERENCES

[1] Han Kamber Pie, “Data Mining Concepts and Techniques”, Morgon Kaufman, Elsvier 2012

[2] https://www.analytixlabs.co.in/blog/types-of-clustering-algorithms/

DISCLAIMER

The material for the presentation has been compiled from various sources such as books, tutorials (offline and online), lecture notes, several resources available on Internet. The information contained in this lecture/presentation is for general information and education purpose only. While we endeavor to keep the information up to date and correct, we make no representation of any kind about the completeness and accuracy of the material. The information shared through this presentation material should be used for educational purpose only.