Machine Learning with Python
Unsupervised Learning - Clustering

Session 4

By Dr. Maryam Rahbaralam

جلسه چهارم

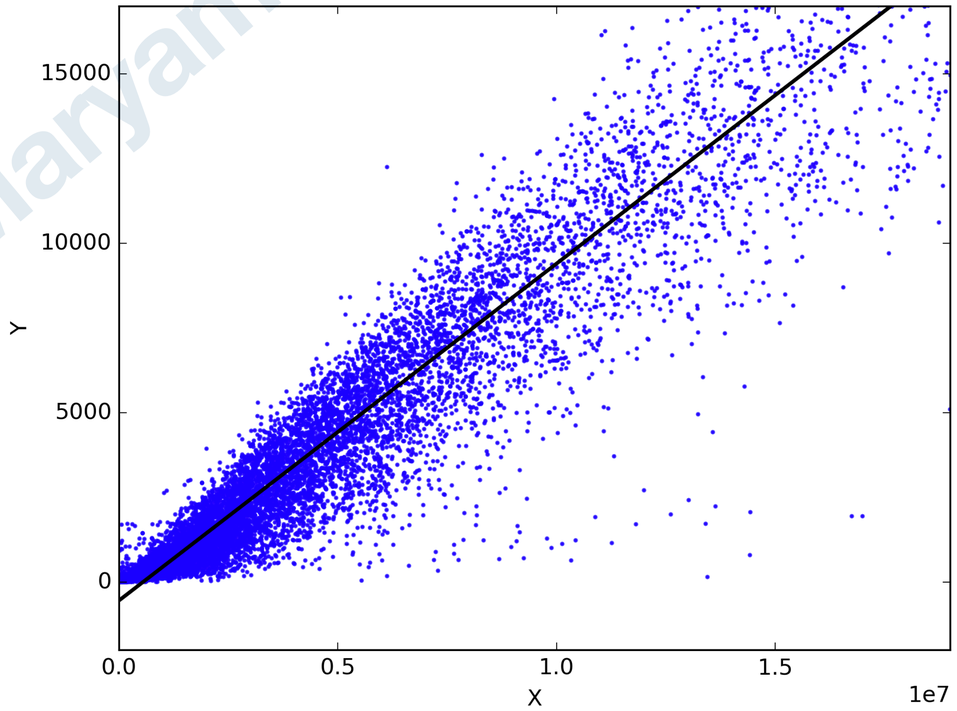
یادگیری بدون نظارت-۱

خوشه‌بندی

دکتر مریم رهبر عالم

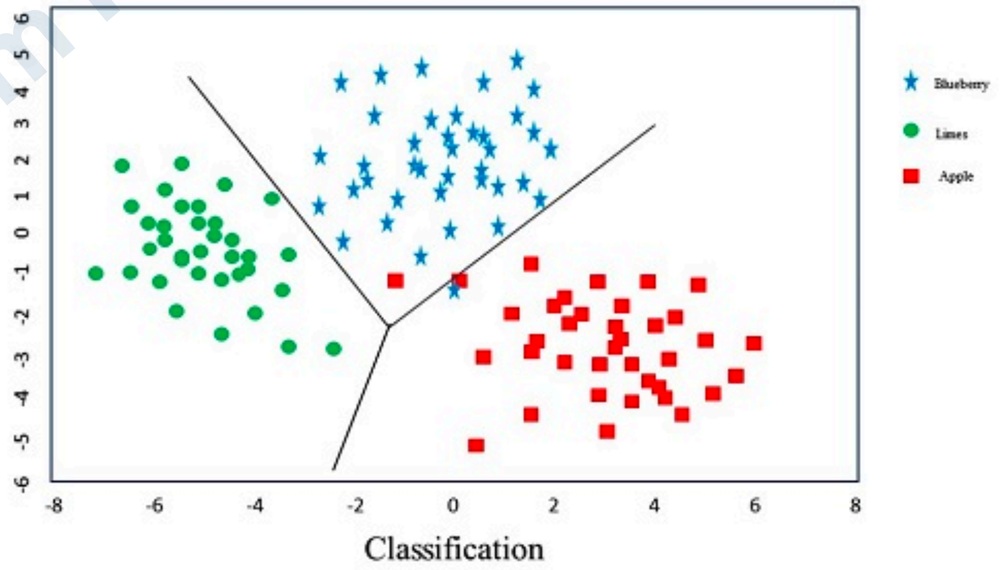
Supervised learning

- **Regression:** the task of approximating a mapping function (f) from input variables (X) to **a continuous** output variable (y).



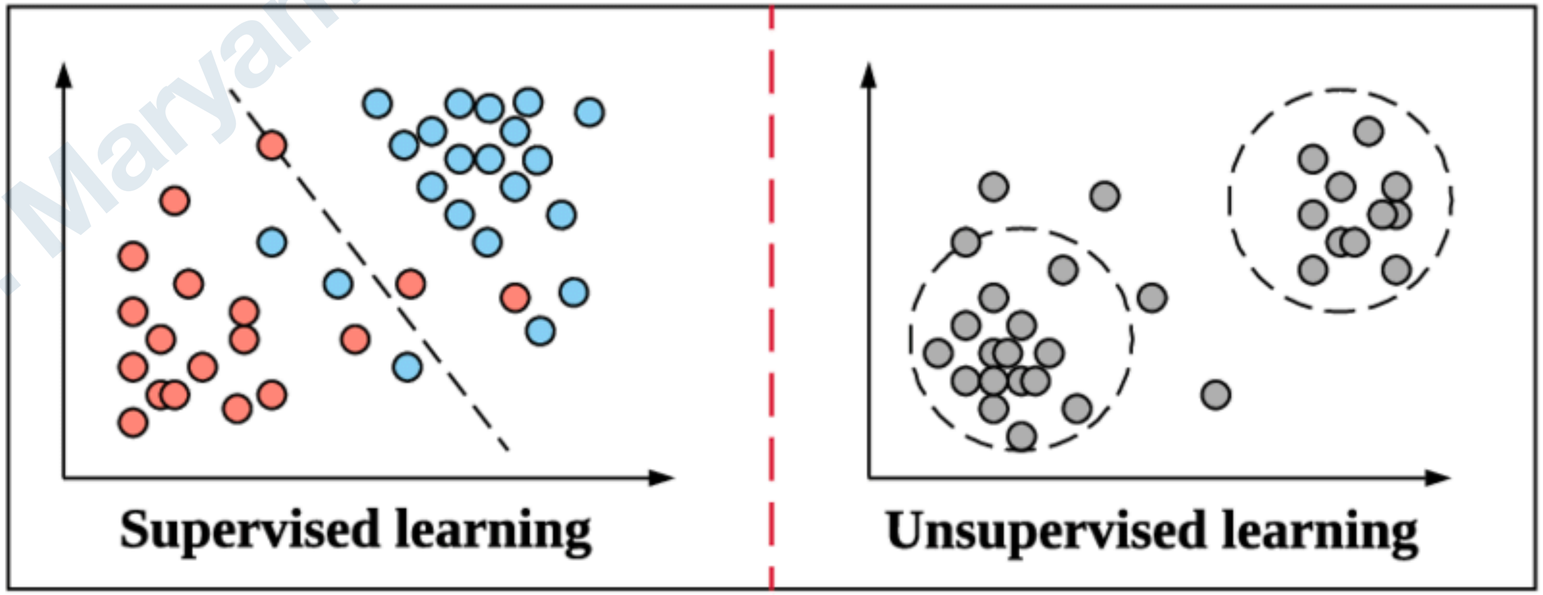
Supervised learning

- **Classification:** the task of approximating a mapping function (f) from input variables (X) to **discrete** output variables (y).



Unsupervised learning

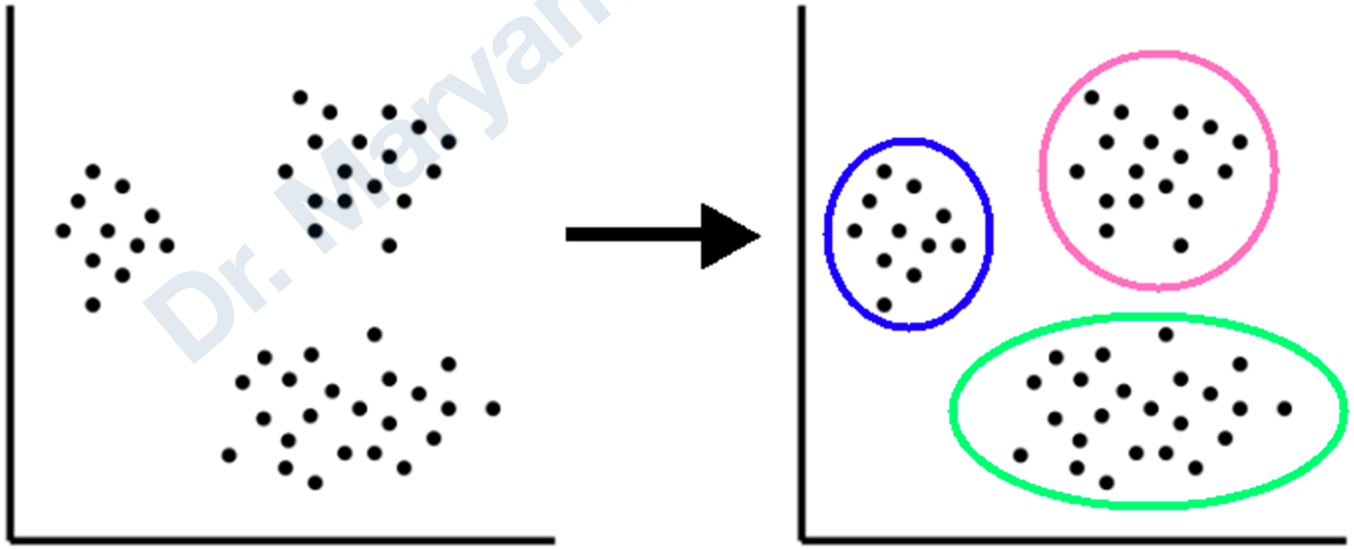
- A type of machine learning algorithm used to draw inferences from datasets consisting of input data **without labeled** responses.



Unsupervised learning

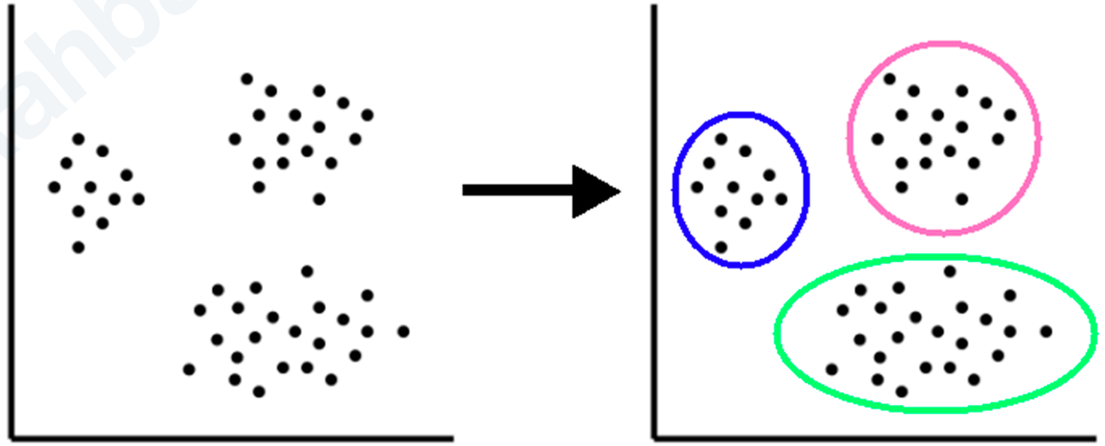
- Goal: finds patterns in data

... but without a specific prediction task in mind



Applications of Clustering in Real-World Scenarios

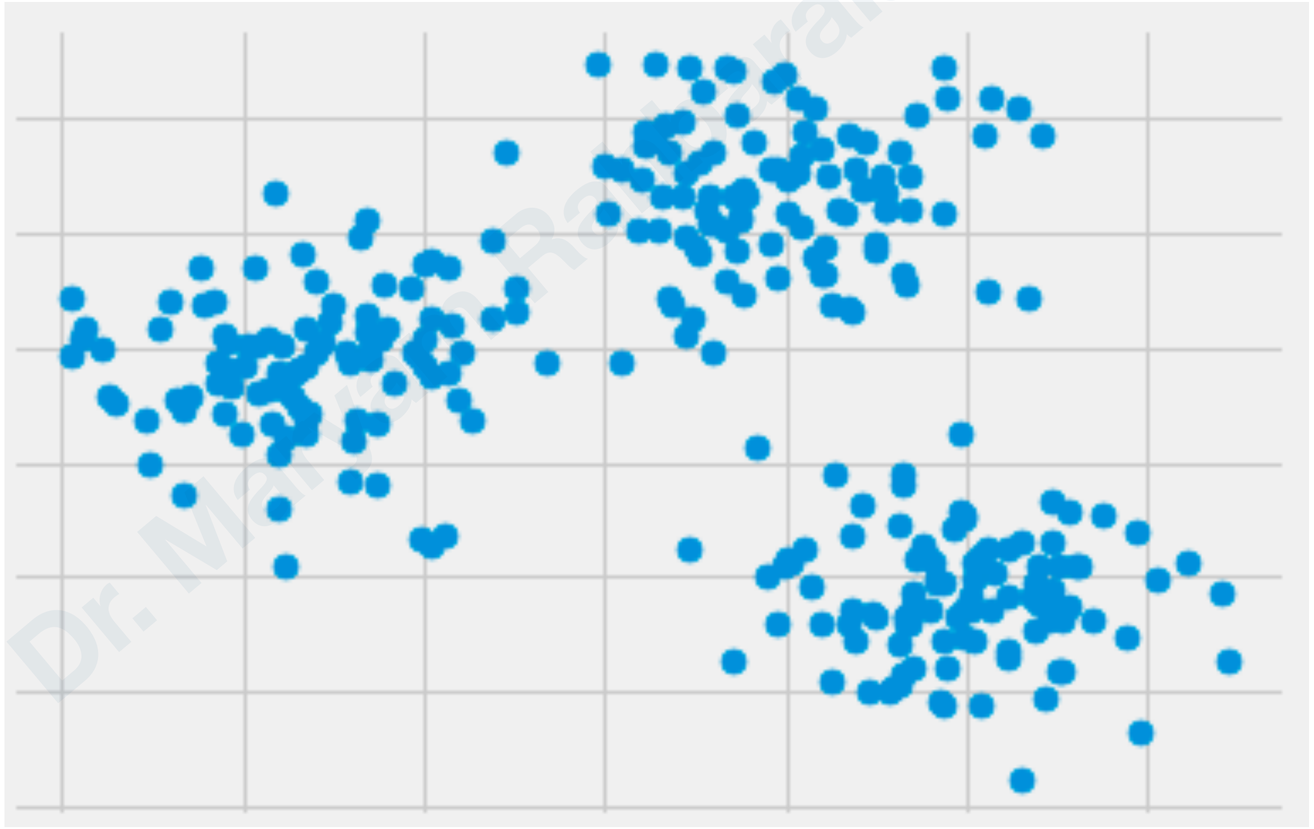
- Customer Segmentation
- Document Clustering
- Image Segmentation
- Recommendation Engines



Unsupervised learning

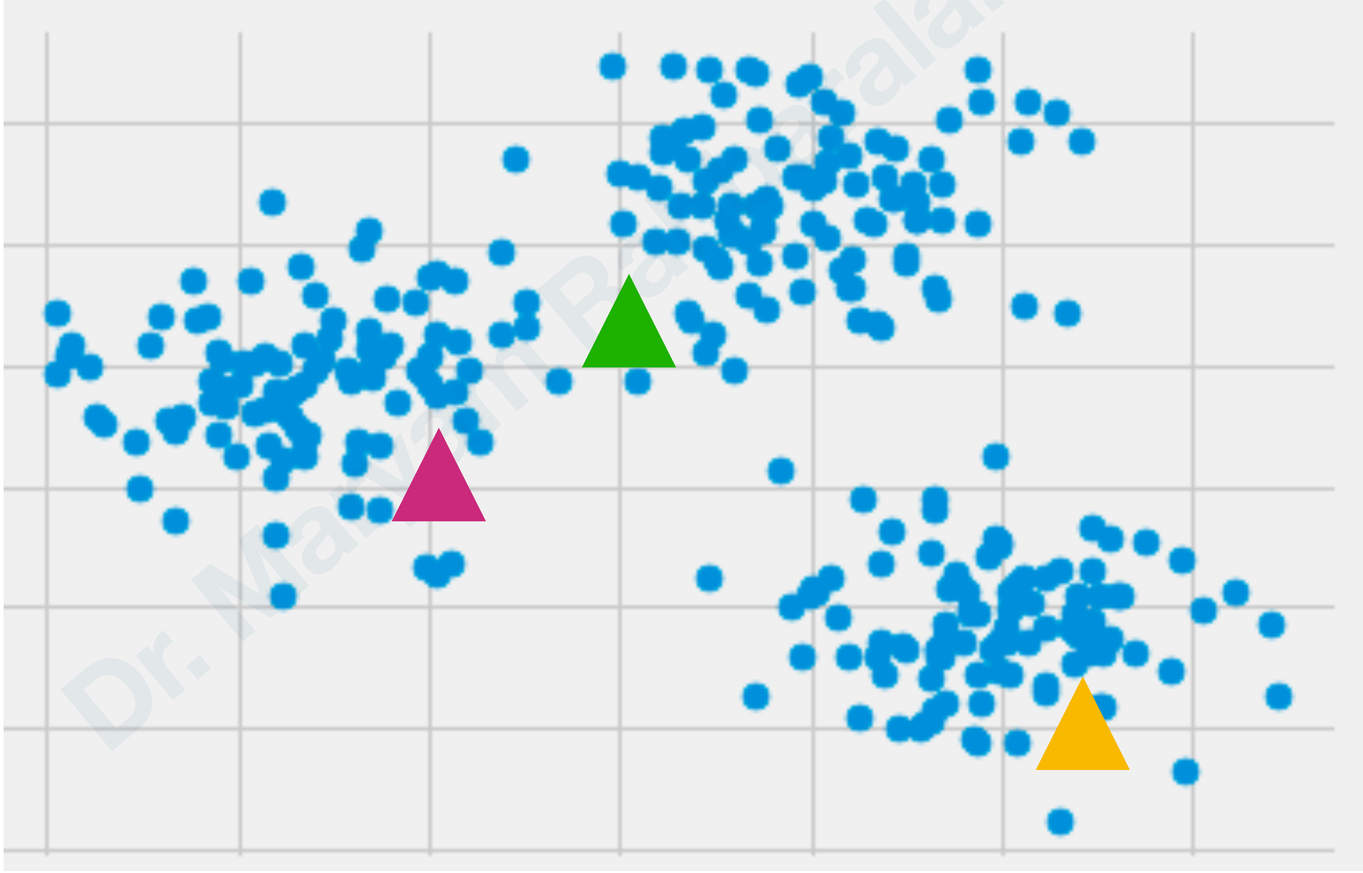
K-Means Clustering

K=3



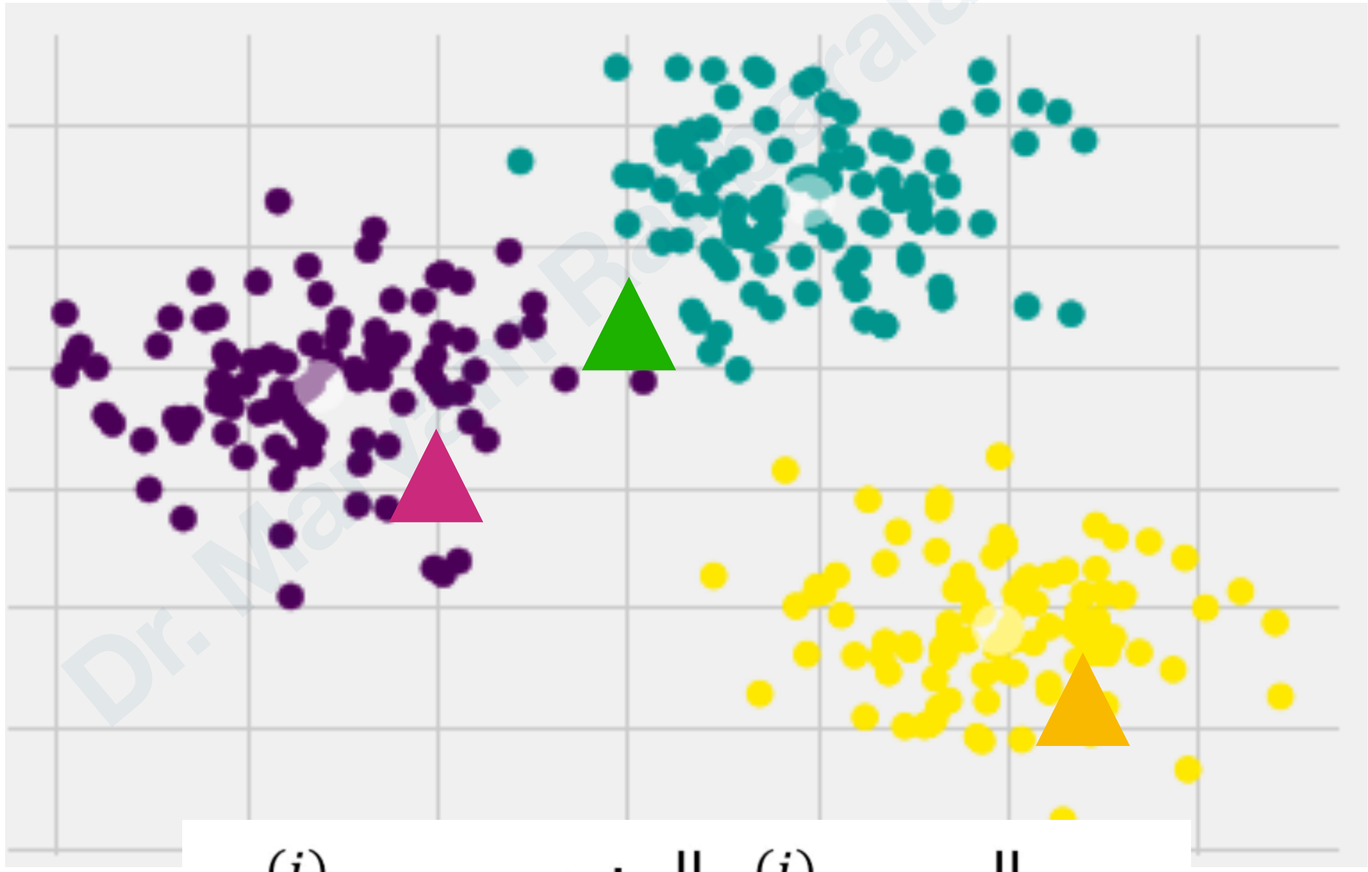
Unsupervised learning

K-Means Clustering



Unsupervised learning

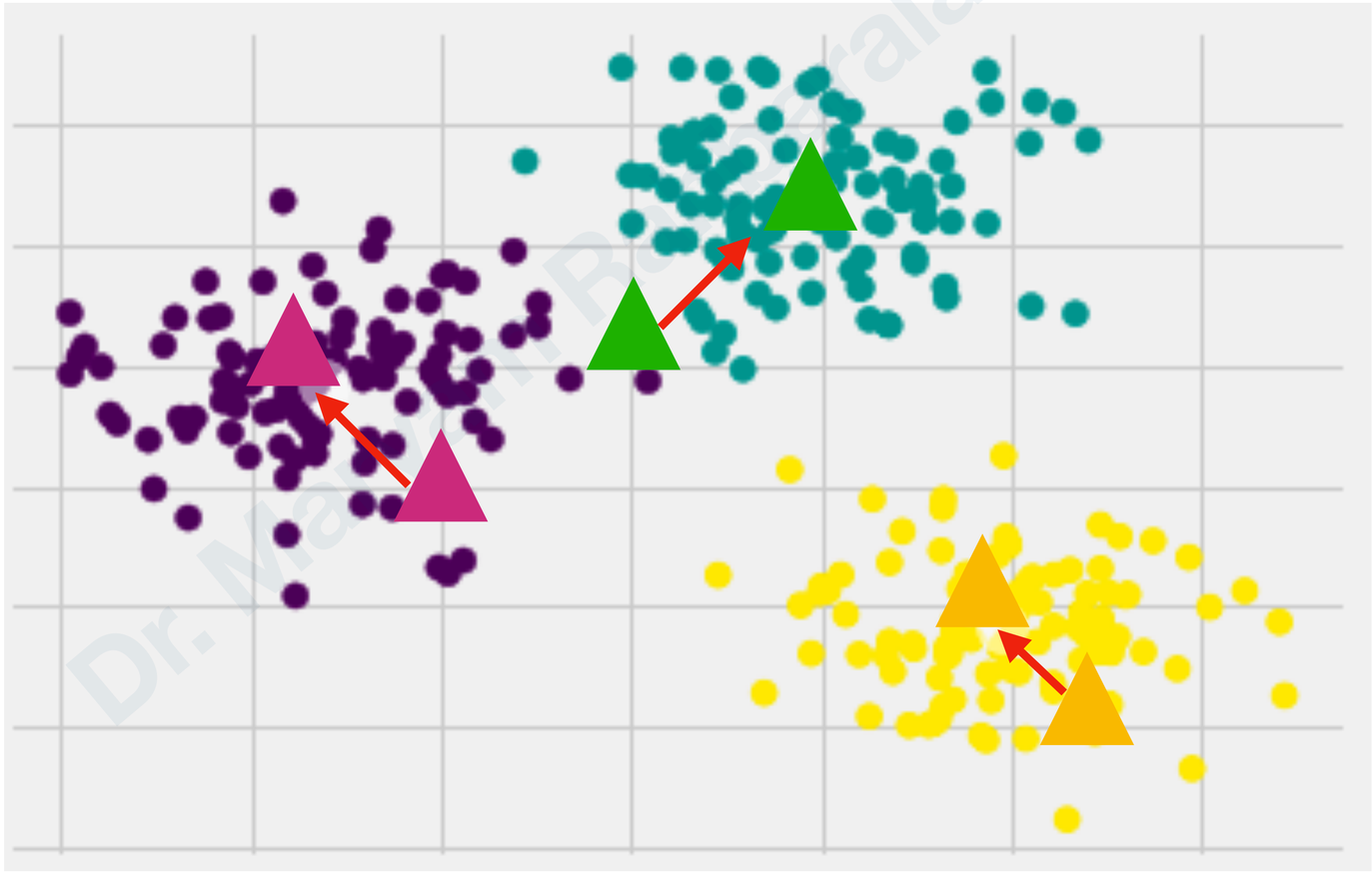
K-Means Clustering



$$c^{(i)} = \arg\min_k \left\| x^{(i)} - \mu_k \right\|$$

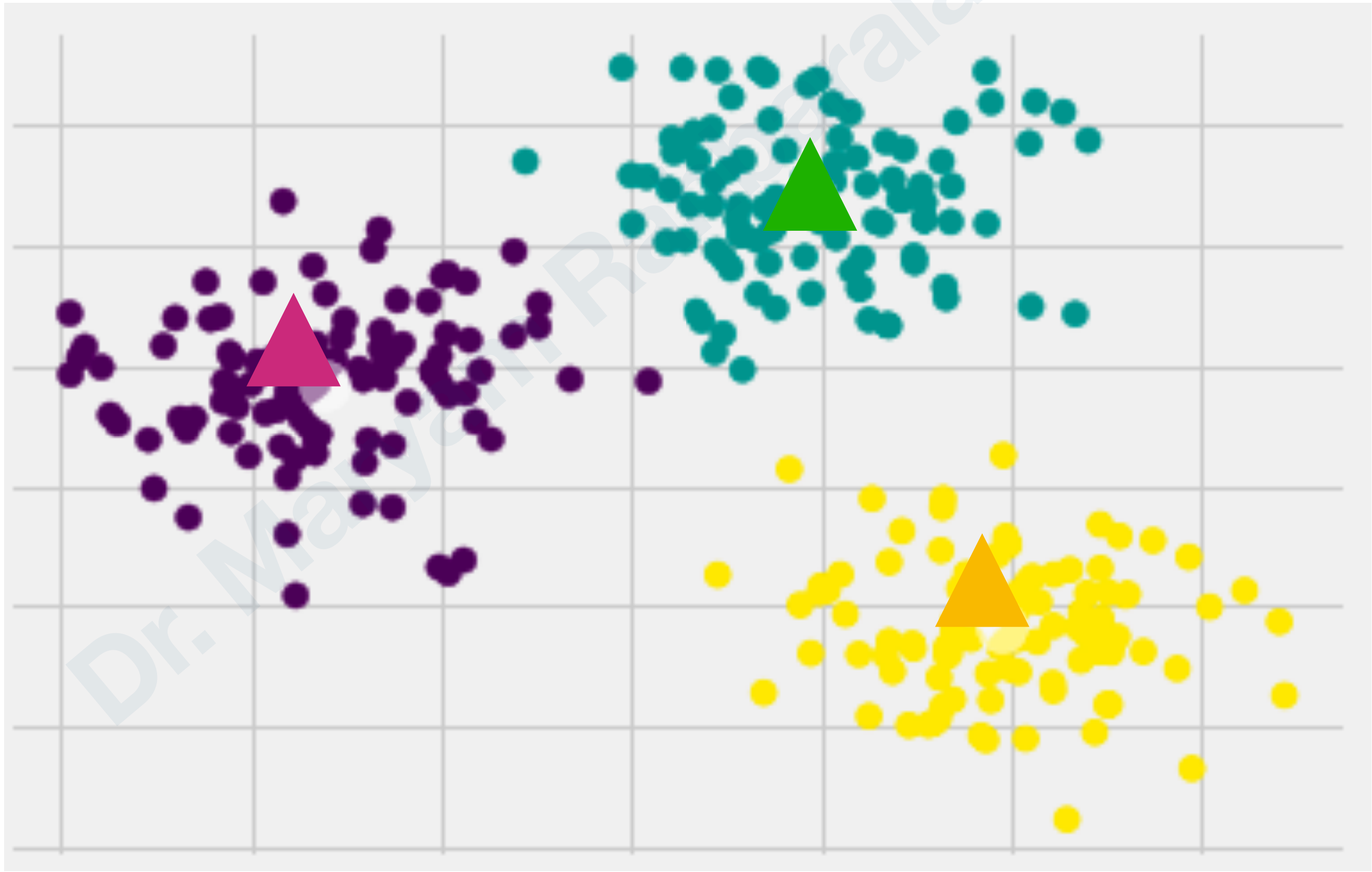
Unsupervised learning

K-Means Clustering



Unsupervised learning

K-Means Clustering



Unsupervised learning

K-Means Clustering

Here, we are stopping the training when the centroids are not changing after two iterations.

Understanding the K-Means Algorithm

Two-step process called **expectation-maximization**:

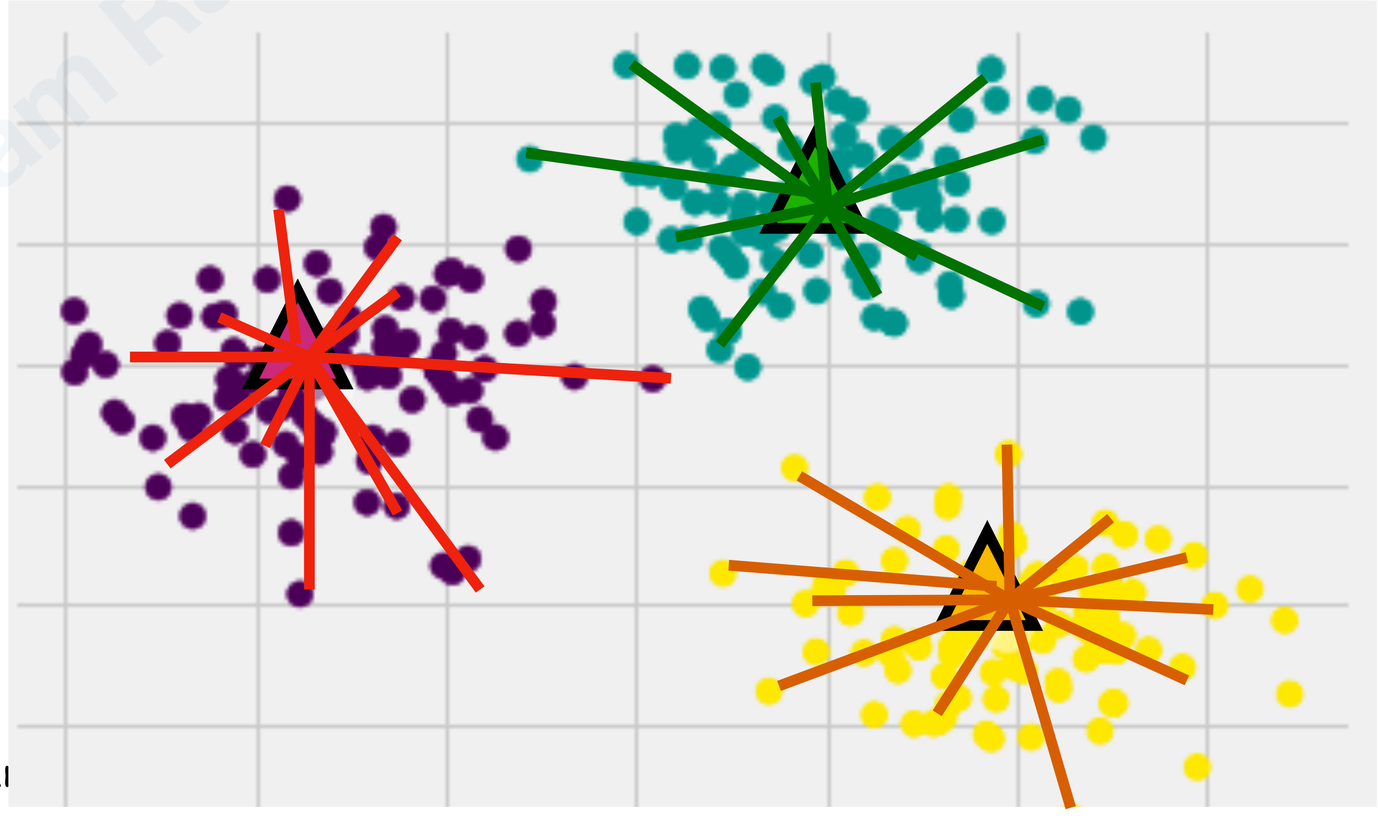
Algorithm 1 k-means algorithm

1: Specify the number k of clusters to assign.
2: Randomly initialize k centroids.
3: **repeat**
4: **expectation:** Assign each point to its closest centroid.
5: **maximization:** Compute the new centroid (mean) of each cluster.
6: **until** The centroid positions do not change.

K-means

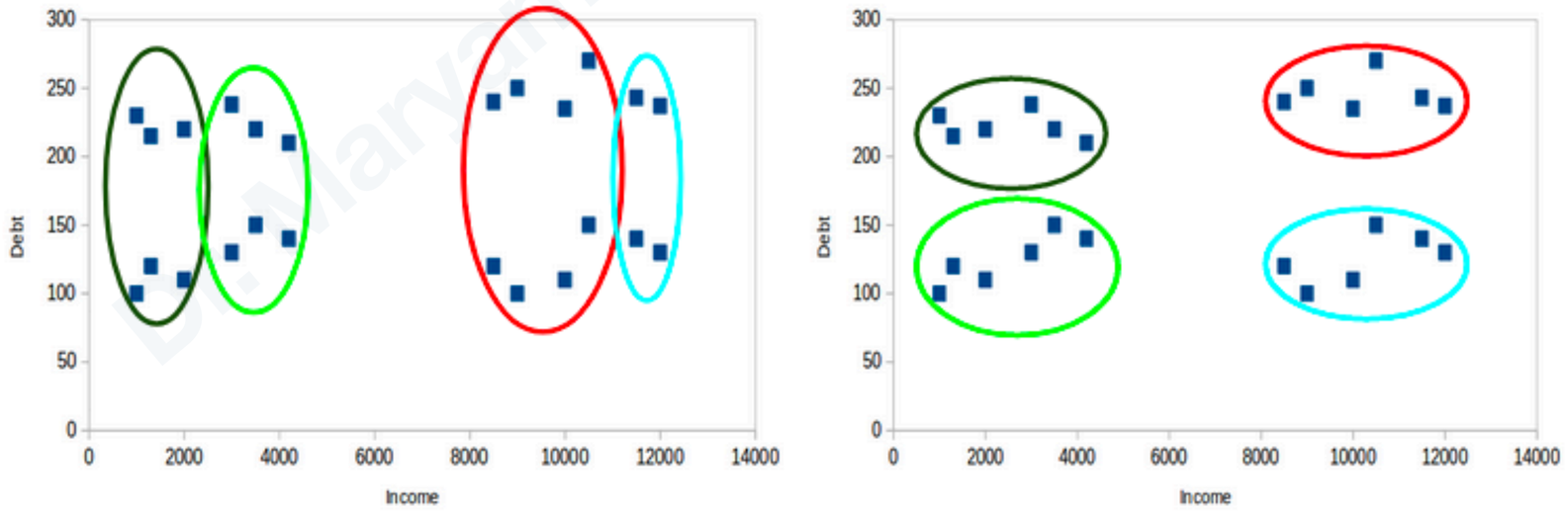
The quality of the cluster assignments is determined by computing the **sum of the squared error (SSE)** after the centroids **converge**, or match the previous iteration's assignment.

$$SSE = \frac{1}{m_k} \sum_{i=1}^{m_k} \|x^i - \mu_{c^k}\|^2$$



K-means

$$SSE = \frac{1}{m_k} \sum_{i=1}^{m_k} \|x^i - \mu_{c^k}\|^2$$

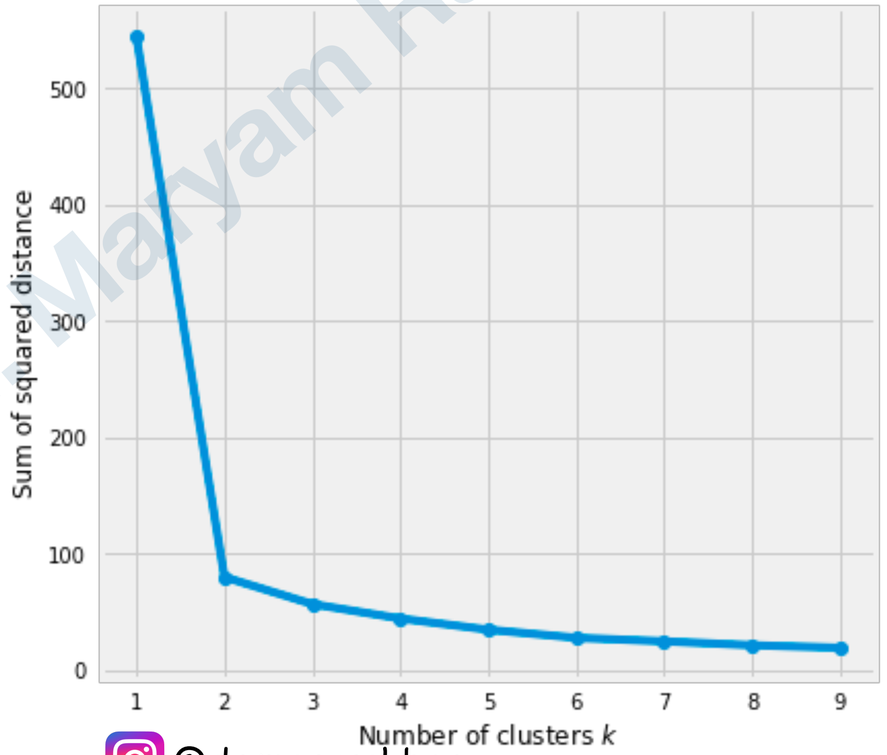


Case - I

Case - II

K-means
Elbow Method

Elbow method gives us an idea on what a good k number of clusters would be based on the sum of squared distance (SSE) between data points and their assigned clusters' centroids.



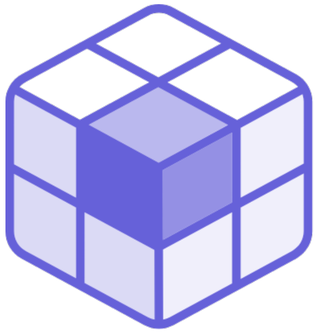
K-means in Python

- from **sklearn.cluster** import **KMeans**
- To create a KMeans instance with 3 clusters, use **KMeans()** along with the keyword argument **n_clusters=**3.
- Use **model.fit()** on points to fit the model to the data.
- You can calculate the coordinates of the cluster centers using **model.cluster_centers_.**
- The lowest SSE value **kmeans.inertia_**
- Final locations of the centroid **kmeans.cluster_centers_**
- The number of iterations required to converge **kmeans.n_iter_**

k-means is limited to linear cluster

- The fundamental model assumptions of *k*-means (points will be closer to their own cluster center than to others) means that the algorithm will often be ineffective if the clusters have **complicated geometries**.
- In particular, the boundaries between *k*-means clusters will always be **linear**, which means that it will fail for more complicated boundaries.

k-means is limited to linear cluster



how we **solve**

- One version of kernelized k-means is implemented in Scikit-Learn within the **SpectralClustering** estimator.
- It uses the graph of nearest neighbors to compute a higher-dimensional representation of the data, and then assigns labels using a k-means algorithm

k-means can be slow for large numbers of samples¶

Because each iteration of k-means must access every point in the dataset, the algorithm can be relatively slow as the number of samples grows. You might wonder if this requirement to use all data at each iteration can be relaxed; for example, you might just use **a subset of the data** to update the cluster centers at each step.

This is the idea behind batch-based k-means algorithms, one form of which is implemented in **sklearn.cluster.MiniBatchKMeans.** The interface for this is the same as for standard **KMeans**; we will see an example of its use as we continue our discussion.

k-means for color compression¶

One interesting application of clustering is in **color compression** within images. For example, imagine you have an image with millions of colors. In most images, a large number of the colors will be unused, and many of the pixels in the image will have similar or even identical colors.