Artificial Intelligence Lab

AL-2002

Lab 07

Instructor: Muhammad Saood Sarwar

Semester: Spring 2024

Artificial Intelligence Lab 07

Objective

The objective of this lab is to provide students with a comprehensive understanding of K-means clustering, including its algorithm, stopping criteria, and methods for choosing the appropriate number of clusters.

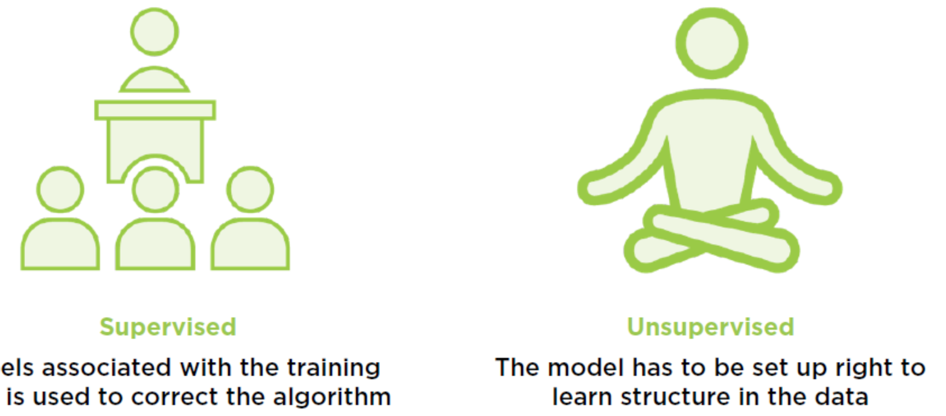
Learning Outcomes

1. A thorough understanding of K-means clustering, including its algorithm, assumptions, and key parameters.
2. The ability to implement K-means clustering using a programming language such as Python.
3. Knowledge of the elbow method for selecting the optimal number of clusters, and the ability to apply this method to real-world datasets.

Table of Contents

K -Mean Clustering

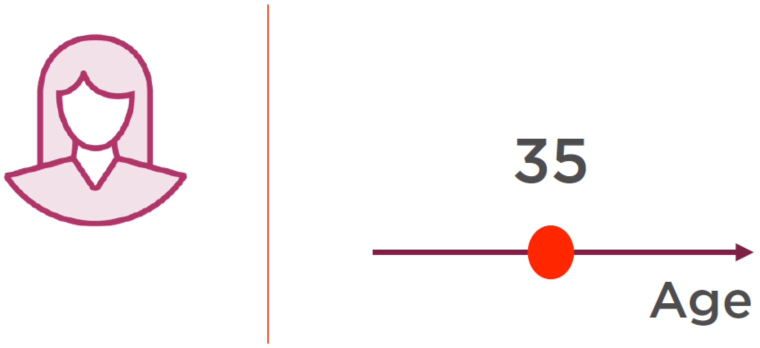
There are two main types of machine learning methods. Supervised learning techniques require labeled training data. Unsupervised learning techniques do not need label instances. Instead, they try to find patterns within the data itself.


Supervised
Labels associated with the training data is used to correct the algorithm
Unsupervised
The model has to be set up right to learn structure in the data

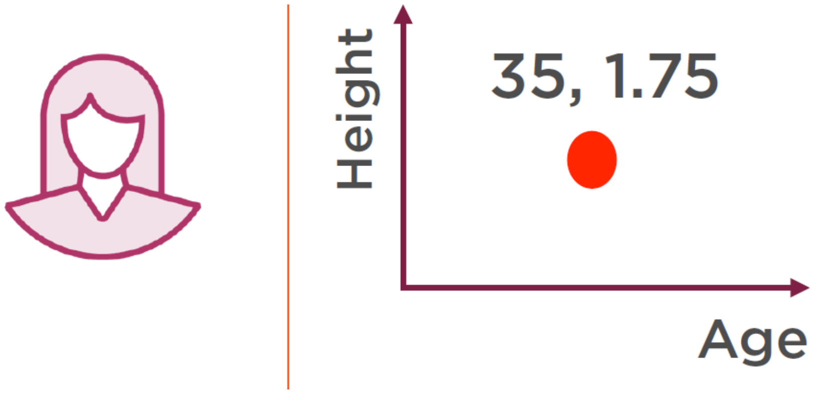
Clustering

Clustering is a popular unsupervised learning technique which helps find patterns in the underlying data. Clustering does not use any Y variables or labels on the data. It looks at the data structure itself. Let's first understand how clustering works and how we can use it with any kind of data.

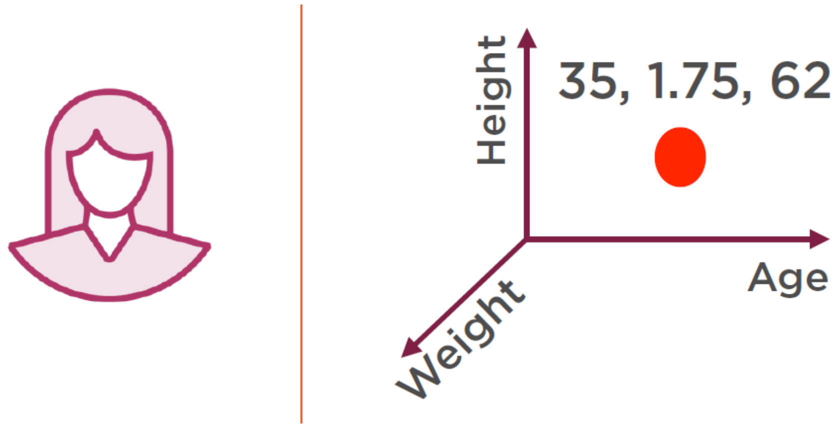
The important principle behind clustering is that anything can be represented by a set of numbers. Whether it's an object, a person, a document, or a webpage, all of these can be represented in some numeric form. Let's consider a person. A person is of a certain age that can be represented on a number line.



A person may be of a certain height. All you need to do then is to represent this information in two dimensions. The person is a point on this plane.



Let's say you were to add a third dimension. A person has a certain weight. Now this individual is represented using three distinct pieces of data.



Now, assuming you have a whole bunch of other information about this person, you could then use an N dimensional Hypercube to represent the set of N numbers. The basic principle is that all the information about a particular person can be represented in **numeric form**.

Now let's take the example of Facebook users. Facebook users have certain characteristics. Different users have different characteristics. Hypothetically, you could have a set of Facebook users where each user is a point in an N dimensional Hypercube.

A set of points, each representing a Facebook user

Clustering involves finding groups of people within this data who have the same characteristics. What those characteristics are can differ. It could be that they like the same music, they went to the same high school, anything. Clustering results in the formation of groups within the data where people within the same group are similar. People who are in different groups are different.

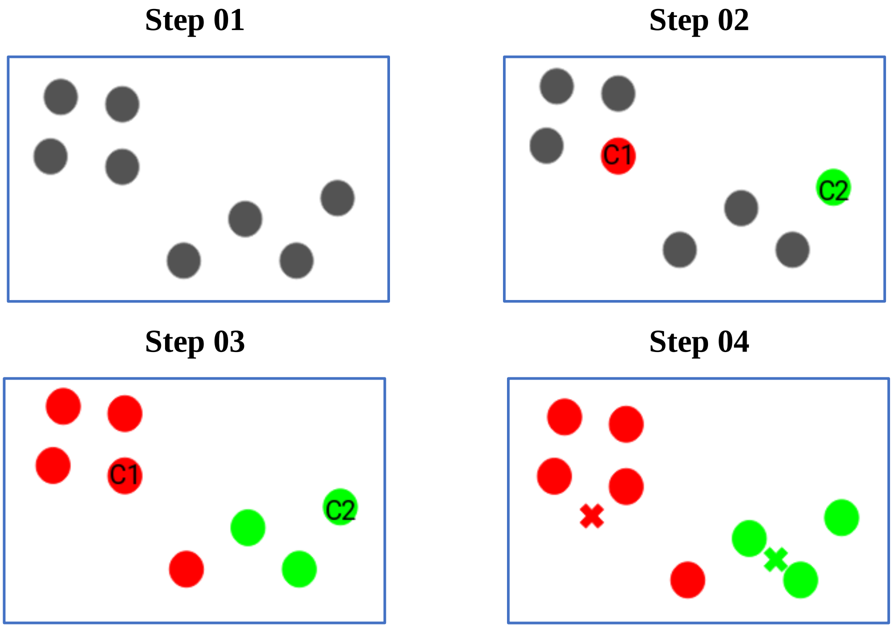
Let's say you were to change the features based on which you performed clustering. You could end up with a completely different set of groups. One of these groups could be parents with children under five. Another group could be parents of teenagers.

If you think about the Facebook example, clustering of users is important because then you can target specific ads to specific groups. **So, how well did your algorithm cluster the underlying data?** This can be measured by considering the distance between individual points in a cluster. The smaller this distance, the better the clustering. The distance between users in a cluster is a measure of how similar the users are, and the goal of clustering is to maximize intra-cluster similarity.

A good clustering algorithm should aim to maximize the similarity between data points within each cluster while also minimizing the similarity between data points in different clusters. This means that we want to achieve high intra-cluster similarity and low inter-cluster similarity. Specifically, we want to ensure that data points within each cluster are as similar as possible while keeping distinct clusters separate from one another. Therefore, minimizing inter-cluster similarity is an important objective for a clustering algorithm, as it helps to ensure that the distance between data points in different clusters is as large as possible.

K-Mean Algorithm

K-means is a centroid-based algorithm, or a distance-based algorithm, where we calculate the distances to assign a point to a cluster. In K-Means, each cluster is associated with a centroid. Recall the first property of clusters – _it states that the points within a cluster should be similar to each other. So, our aim here is to minimize the distance between the points within a cluster.



Step 01

Step 02

Step 03

Step 04

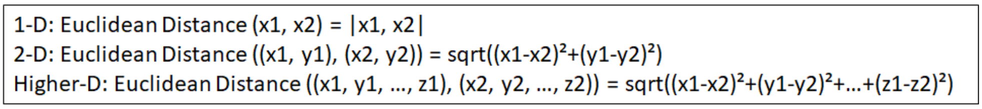
Step 1: Choose the number of clusters k

Step 2: Select k random points from the data as centroids.

Randomly select the centroid for each cluster. Let’s say we want to have 2 clusters, so k is equal to 2 here. We then randomly select the centroid: Here, the red and green circles represent the centroid for these clusters.

Step 3: Assign all the points to the closest cluster centroid.

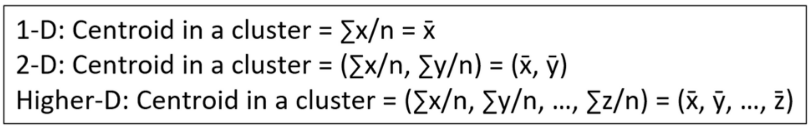
Once we have initialized the centroids, we assign each point to the closest cluster centroid: You can see that the points which are closer to the red point are assigned to the red cluster whereas the points which are closer to the green point are assigned to the green cluster.



1-D: Euclidean Distance (x1, x2) = |x1, x2|
2-D: Euclidean Distance ((x1, y1), (x2, y2)) = sqrt($(x1-x2)^2+(y1-y2)^2$)
Higher-D: Euclidean Distance ((x1, y1, …, z1), (x2, y2, …, z2)) = sqrt($(x1-x2)^2+(y1-y2)^2+\ldots+(z1-z2)^2$)

Step 4: Recompute the centroids of newly formed clusters.

Now, once we have assigned all the points to either cluster, the next step is to compute the centroids of newly formed clusters: Here, the red and green crosses are the new centroids.



1-D: Centroid in a cluster = $\sum x/n = \bar{x}$
2-D: Centroid in a cluster = $(\sum x/n, \sum y/n) = (\bar{x}, \bar{y})$
Higher-D: Centroid in a cluster = $(\sum x/n, \sum y/n, \ldots, \sum z/n) = (\bar{x}, \bar{y}, \ldots, \bar{z})$

Step 5: Repeat steps 3 and 4

The step of computing the centroid and assigning all the points to the cluster based on their distance from the centroid is a single iteration.

when should we stop this process? It can't run till eternity, right?

Stopping Criteria for K-Means Clustering

There are essentially three stopping criteria that can be adopted to stop the K-means algorithm:

- Centroids of newly formed clusters do not change.
- Points remain in the same cluster.
- Maximum number of iterations are reached.

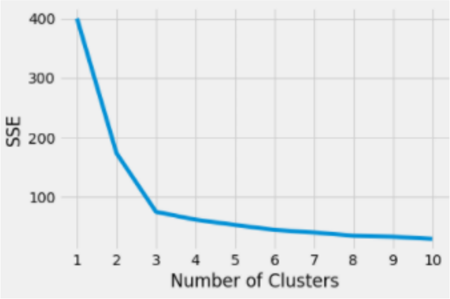
Choosing the Appropriate Number of Clusters

Commonly used to evaluate the appropriate number of clusters are:

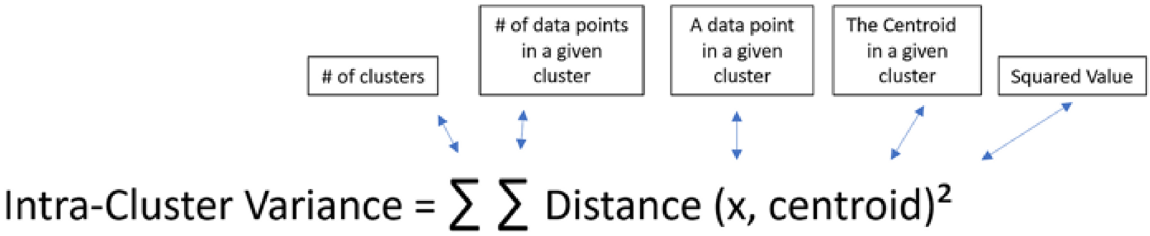
- The elbow method
- The silhouette coefficient

Elbow method.

To perform the elbow method, run several k-means, increment k with each iteration, and record the sum of squared error (SSE). When you plot SSE as a function of the number of clusters, notice that SSE continues to decrease as you increase k. As more centroids are added, the distance from each point to its closest centroid will decrease. There's a sweet spot where the SSE curve starts to bend known as the elbow point. The x-value of this point is thought to be a reasonable trade-off between error and number of clusters. In this example, the elbow is located at x=3:



Intra-cluster variance (a.k.a., the squared error function or sum of squares within (SSW) or sum of squares error (SSE)) is used to quantify internal cohesion. It is defined as the sum of the squared distance between the average point (called **Centroid**) and each point of the cluster. *The smaller the value, the better the clustering is.*


# of clusters
# of data points in a given cluster
A data point in a given cluster
The Centroid in a given cluster
Squared Value
Intra-Cluster Variance = ∑ ∑ Distance (x, centroid)²

Applications

1. Image segmentation: K-means clustering is often used to segment images based on their colors. Each pixel in an image can be considered as a data point, and k-means clustering can group pixels with similar colors together.
2. Customer segmentation: K-means clustering can be used to segment customers based on their behavior or demographic data. This helps businesses target their marketing efforts more effectively by tailoring their messaging to specific customer segments.
3. Anomaly detection: K-means clustering can also be used to detect anomalies or outliers in a dataset. Data points that are significantly different from the rest of the data can be identified as potential anomalies.
4. Natural language processing: K-means clustering is often used in natural language processing to group similar text documents together. This helps with tasks such as document classification and topic modeling.
5. Recommender systems: K-means clustering can be used to group users or items based on their behavior or characteristics. This helps with building recommender systems that

can suggest products or services to users based on their similarity to other users or items in the same cluster.