## Unsupervised Learning

Unsupervised learning is a branch of [machine learning](https://www.geeksforgeeks.org/machine-learning/) that deals with unlabelled data. Unlike supervised learning, where the data is labelled with a specific category or outcome, unsupervised learning algorithms are tasked with finding patterns and relationships within the data without any prior knowledge of the data’s meaning. Unsupervised machine learning algorithms find hidden patterns and data without any human intervention, i.e., we don’t give output to our model. The training model has only input parameter values and discovers the groups or patterns on its own.

There are mainly 3 types of Algorithms which are used for Unsupervised dataset.

* Clustering
* Association Rule Learning
* Dimensionality Reduction

### **Clustering Algorithms**

[Clustering](https://www.geeksforgeeks.org/clustering-in-machine-learning/) in unsupervised machine learning is the process of grouping unlabelled data into clusters based on their similarities. The goal of clustering is to identify patterns and relationships in the data without any prior knowledge of the data’s meaning. Broadly this technique is applied to group data based on different patterns, such as similarities or differences, our machine model finds. These algorithms are used to process raw, unclassified data objects into groups. Some common clustering algorithms:

* **K-means Clustering**: Groups data into K clusters based on how close the points are to each other.
* **Hierarchical Clustering**: Creates clusters by building a tree step-by-step, either merging or splitting groups.
* **Density-Based Clustering (DBSCAN):** Finds clusters in dense areas and treats scattered points as noise.
* **Mean-Shift Clustering**: Discovers clusters by moving points toward the most crowded areas.
* **Spectral Clustering**: Groups data by analysing connections between points using graphs.

## K-Means Clustering

K-means clustering is a way of grouping data based on how similar or close the data points are to each other. Imagine you have a bunch of points, and you want to group them into clusters. The algorithm works by first randomly picking some central points (called centroids) and then assigning every data point to the nearest centroid. Once that’s done, it recalculates the centroids based on the new groupings and repeats the process until the clusters make sense. It’s a pretty fast and efficient method, but it works best when the clusters are distinct and not too mixed up. One challenge, though, is figuring out the right number of clusters (K) beforehand. Plus, if there’s a lot of noise or overlap in the data, K Means might not perform as well.

**Algorithm:**

1. Let's say we have x1, x2, x3……… x(n) as our inputs, and we want to split this into K clusters. We randomly pick K (centroids). We name them c1, c2, ... ck, and we can say that where C is the list of all centroids.
2. We assign each data point to its nearest centre, which is accomplished by calculating the Euclidean distance.

where dist() is the Euclidean distance. Here, we calculate each x value's distance from each c value, i.e. the distance between x1-c1, x1-c2, x1-c3, and so on. Then we find which is the lowest value and assign x1 to that particular centroid. Similarly, we find the minimum distance for x2, x3, etc.

1. We identify the actual centroid by taking the average of all the points assigned to that cluster.

A mathematical equation with numbers and symbols

AI-generated content may be incorrect.

where Si is the set of all points assigned to the cluster. It means the original point, which we thought was the centroid, will shift to the new position, which is the actual centroid for each of these groups.

1. Keep repeating step 2 and step 3 until convergence is achieved.

**Objectives**:

1. **Grouping Similar Data Points**: K-Means is designed to cluster data points that share common traits, allowing patterns or trends to emerge. Whether analysing customer behaviour or images, the method helps reveal hidden relationships within your dataset.
2. **Minimizing Within-Cluster Distance**: Another objective is to keep data points in each group as close to the cluster's centroid as possible. Reducing this internal distance results in compact, cohesive clusters, enhancing the accuracy of your results.
3. **Maximizing Between-Cluster Distance**: K-Means also aims to maintain clear separation between different clusters. By maximizing the distance between groups, the algorithm ensures that each cluster remains distinct, providing a better understanding of data categories without overlap.

**Properties**:

1. **Similarity Within a Cluster**: One of the main things K Means aims for is that all the data points in a cluster should be pretty similar to each other. Imagine a bank that wants to group its customers based on income and debt. If customers within the same cluster have vastly different financial situations, then a one-size-fits-all approach to offers might not work. For example, a customer with high income and high debt might have different needs compared to someone with low income and low debt. By making sure the customers in each cluster are similar, the bank can create more tailored and effective strategies.
2. **Differences Between Clusters**: Another important aspect is that the clusters themselves should be as distinct from each other as possible. Going back to our bank example, if one cluster consists of high-income, high-debt customers and another cluster has high-income, low-debt customers, the differences between the clusters are clear. This separation helps the bank create different strategies for each group. If the clusters are too similar, it can be challenging to treat them as separate segments, which can make targeted marketing less effective.

**Applications of K-Means Clustering**

* **Distance Measures**: At the heart of K-Means clustering is the concept of distance. Euclidean distance, for example, is a simple straight-line measurement between points and is commonly used in many applications. Manhattan distance, however, follows a grid-like path, much like how you'd navigate city streets. Squared Euclidean distance makes calculations easier by squaring the values, while cosine distance is handy when working with text data because it measures the angle between data vectors. Picking the right distance measure really depends on what kind of problem you’re solving and the nature of your data.
* **Customer Segmentation**: One of the most popular uses of K-means clustering is for customer segmentation. From banks to e-commerce, businesses use K-means clustering customer segmentation to group customers based on their behaviours. For example, in telecom or sports industries, companies can create targeted marketing campaigns by understanding different customer segments better. This allows for personalized offers and communications, boosting customer engagement and satisfaction.
* **Document Clustering**: When dealing with a vast collection of documents, K-Means can be a lifesaver. It groups similar documents together based on their content, which makes it easier to manage and retrieve relevant information. For instance, if you have thousands of research papers, clustering can quickly help you find related studies, improving both organization and efficiency in accessing valuable information.
* **Image Segmentation**: In image processing, K-Means clustering is commonly used to group pixels with similar colours, which divides the image into distinct regions. This is incredibly helpful for tasks like object detection and image enhancement. For instance, clustering can help separate objects within an image, making analysis and processing more accurate. It’s also widely used to extract meaningful features from images in various visual tasks.
* **Recommendation Engines**: K-Means clustering also plays a vital role in recommendation systems. Say you want to suggest new songs to a listener based on their past preferences; clustering can group similar songs together, helping the system provide personalized suggestions. By clustering content that shares similar features, recommendation engines can deliver a more tailored experience, helping users discover new songs that match their taste.
* **K-Means for Image Compression**: K-Means can even help with image compression by reducing the number of colours in an image while keeping the visual quality intact. K-Means reduces the image size without losing much detail by clustering similar colours and replacing the pixels with the average of their cluster. It’s a practical method for compressing images for more accessible storage and transmission, all while maintaining visual clarity.

**Advantages of K-means**

1. Simple and easy to implement: The k-means algorithm is easy to understand and implement, making it a popular choice for clustering tasks.
2. Fast and efficient: K-means is computationally efficient and can handle large datasets with high dimensionality.
3. Scalability: K-means can handle large datasets with many data points and can be easily scaled to handle even larger datasets.
4. Flexibility: K-means can be easily adapted to different applications and can be used with varying metrics of distance and initialization methods.

**Disadvantages of K-Means**

1. Sensitivity to initial centroids: K-means is sensitive to the initial selection of centroids and can converge to a suboptimal solution.
2. Requires specifying the number of clusters: The number of clusters k needs to be specified before running the algorithm, which can be challenging in some applications.
3. Sensitive to outliers: K-means is sensitive to outliers, which can have a significant impact on the resulting clusters.

**Different Evaluation Metrics for Clustering**

When it comes to evaluating how well your clustering algorithm is working, there are a few key metrics that can help you get a clearer picture of your results. Here’s a rundown of the most useful ones:

* **Silhouette Analysis**: Silhouette analysis is like a report card for your clusters. It measures how well each data point fits into its own cluster compared to other clusters. A high silhouette score means that your points are snugly fitting into their clusters and are quite distinct from points in other clusters. Imagine a score close to 1 as a sign that your clusters are well-defined and separated. Conversely, a score close to 0 indicates some overlap, and a negative score suggests that the clustering might need some work.
* **Inertia**: Inertia is a bit like a gauge of how tightly packed your data points are within each cluster. It calculates the sum of squared distances from each point to the cluster's center (or centroid). Think of it as measuring how snugly the points are huddled together. Lower inertia means that points are closer to the centroid and to each other, which generally indicates that your clusters are well-formed. For most numeric data, you'll use Euclidean distance, but if your data includes categorical features, Manhattan distance might be better.
* **Dunn Index**: The Dunn Index takes a broader view by considering both the distance within and between clusters. It’s calculated as the ratio of the smallest distance between any two clusters (inter-cluster distance) to the largest distance within a cluster (intra-cluster distance). A higher Dunn Index means that clusters are not only tight and cohesive internally but also well-separated from each other. In other words, you want your clusters to be as far apart as possible while being as compact as possible.

 Elbow method:

The Elbow method is the best way to find the number of clusters. The elbow method constitutes running K-Means clustering on the dataset.

Next, we use within-sum-of-squares as a measure to find the optimum number of clusters that can be formed for a given data set. Within the sum of squares (WSS) is defined as the sum of the squared distance between each member of the cluster and its centroid.

A black and white math equation

AI-generated content may be incorrect.

Where = data point and = closest point to centroid. The WSS is measured for each value of K. The value of K, which has the least amount of WSS, is taken as the optimum value. Now, we draw a curve between WSS and the number of clusters.

A graph of a function

AI-generated content may be incorrect.

Here, WSS is on the y-axis and number of clusters on the x-axis. You can see that there is a very gradual change in the value of WSS as the K value increases from two. So, you can take the elbow point value as the optimal value of K. It should be either two, three, or at most four. But, beyond that, increasing the number of clusters does not dramatically change the value in WSS, it gets stabilized.