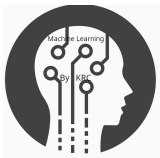
Machine Learning (Clustering Algorithms)

Prof K R Chowdhary

CSE Dept., MBM University

December 29, 2024

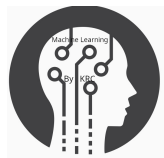


Pattern representation and feature extraction

$\Rightarrow$ A better quality pattern representation results to simple and easily understood clustering. E.g., In Cartesian coordinates, many clustering algorithms may fragment data into two or more clusters; In polar coordinates, radius coordinate causes tight clustering and a one-cluster solution can be easily obtained.

$\Rightarrow$ A pattern can be for a physical object or an abstract notion. Physical: a chair, table, book, house, abstract: a style of writing, attitude, belief. Both can be represented as multidimensional vectors. Features of pattern can be quantitative /qualitative: weight, color, $(black, 5)$ is black object with 5 units of weight, or degree of blackness.

$\Rightarrow$ Other representations are: tree structures, a parent node represents a generalization of its child nodes. E.g., a parent node "4-wheeler": *generalization* of "cars," "jeep," "tractor,". The node "cars" could be a generalization of car make, "Hundai," "Tata,", etc.

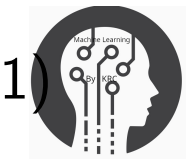


$\Rightarrow$ In clustering, where it lacks class labels, feature selection is an ad hoc, but a necessity. As it lacks class labels, there can only be a trial-and-error process for selection of features. The resultant patterns are clustered, and output is evaluated using a *validity index*. Popular feature extraction processes: principal components analysis (PCA), it does not depend on labeled data. Patterns having smaller number of features are beneficial [1].

$\Rightarrow$ For clustering: first requirement is to find out similarities, and more similar patterns are clubbed together.

$\Rightarrow$ Dissimilarity between two patterns is the feature space using the distance measure. The popular metric for continuous features is *Euclidean distance*:

$$d_2(\mathbf{x}_i, \mathbf{x}_j) = \Big(\sum_{k=1}^{d}(x_{i,k} - x_{j,k})^2\Big)^{1/2}$$
$$= \| \mathbf{x}_i - \mathbf{x}_j \|_2 . \qquad (1)$$



$\Rightarrow$ The equation (1) is a special case of the *Minkowski's metric*, where p was taken as 2, expressed by,

$$d_p(\mathbf{x}_i, \mathbf{x}_j) = (\sum_{k=1}^{d}(x_{i,k} - x_{j,k})^p)^{1/p}$$
$$= \| \mathbf{x}_i - \mathbf{x}_j \|_p . \qquad (2)$$

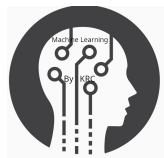
where d_2 Stands for two dimensions, d is number of dimensions (=no. of attributes).

$\Rightarrow$ Approach based on Euclidean distance, the method is used to evaluate proximity of objects in 2D/3D spaces.

$\Rightarrow$ Set of 2D data points (Table 1) and a data point, $x = (2.5, 2.9)$ as a query, rank these database points based on similarity with query.

Table 1: 2D data

	A_1	A_2
x_1	1.9	1.7
x_2	2.1	2.1
x_3	2.6	3.0
x_4	2.2	2.5
x_5	1.8	2.0



$\Rightarrow$ Using equation (1), we compute the Euclidean distance for the two dimensional data points $x_1, \ldots, x_5$ with respect to the query $x = (2.5, 2.9)$. The result are shown in Table 2.

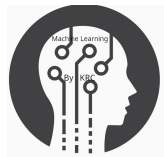
Table 2: Euclidean Distances

Data pt.	Euclid. dist. with x
x_1	1.341
x_2	0.894
x_3	0.141
x_4	0.500
x_5	1.140

The distance matrix shows that query $(2.5, 2.9)$ is nearest to x_3, having distance 0.141.

$\Rightarrow$ **Nearest Neighbor Clustering (NN)**: An iterative algorithm assigns each unlabeled pattern to the cluster of its nearest labeled neighbor pattern. Condition is: distance to that nearest pattern is below threshold.

$\Rightarrow$ This process continues until all the input patterns are labeled.



Clustering Algorithms: Nearest Neighbor Clustering (NN)

$\Rightarrow$ To grow the clusters from NN, a concept: *mutual neighbor distance*, (MN_d), is used.

$$MN_d(\mathbf{x}_i, \mathbf{x}_j) = C_n(\mathbf{x}_j, \mathbf{x}_i) + C_n(\mathbf{x}_i, \mathbf{x}_j).$$

where, $\mathbf{x}_i, \mathbf{x}_j$, are patterns, $C_n(\mathbf{x}_i, \mathbf{x}_j)$ is count of NN of $\mathbf{x}_j$ w.r.t $\mathbf{x}_i$. Fig. 1(a)NN of pattern P is Q, and Q's NN is P. Also, $C_n(P, Q) = C_n(Q, P) = 1$, So, $MN_d(P, Q) = 2$. If $C_n(Q, R) = 1$ but $C_n(R, Q) = 2$, then $MN_d(Q, R) = C_n(Q, R) + C_n(R, Q) = 3$.

$\Rightarrow$ Fig. 1(b) we get from figure 1(a) by adding three more patterns S, T, U. Now, $MN_d(Q, R) = 3$, but $MN_d(P, Q)$ =5. Note: $MN_d(P, Q)$ has increased from 2 to 5 due to three more patterns S, T, U, however, P and Q remains at same place.

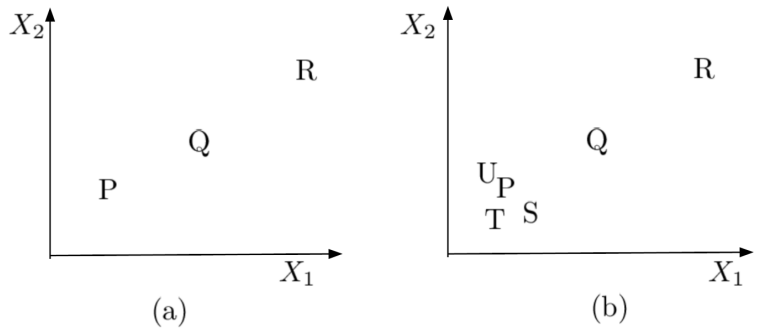


Figure 1: NN clustering: (a) P, Q are more similar than P, R, (b) Q, R are more similar than Q, P

$\Rightarrow$ A general case of NN algorithm is k-nearest neighbor algorithm. Fig. 2 illustrates for $k = 1$, 2, and 3-nearest neighbor graphs.

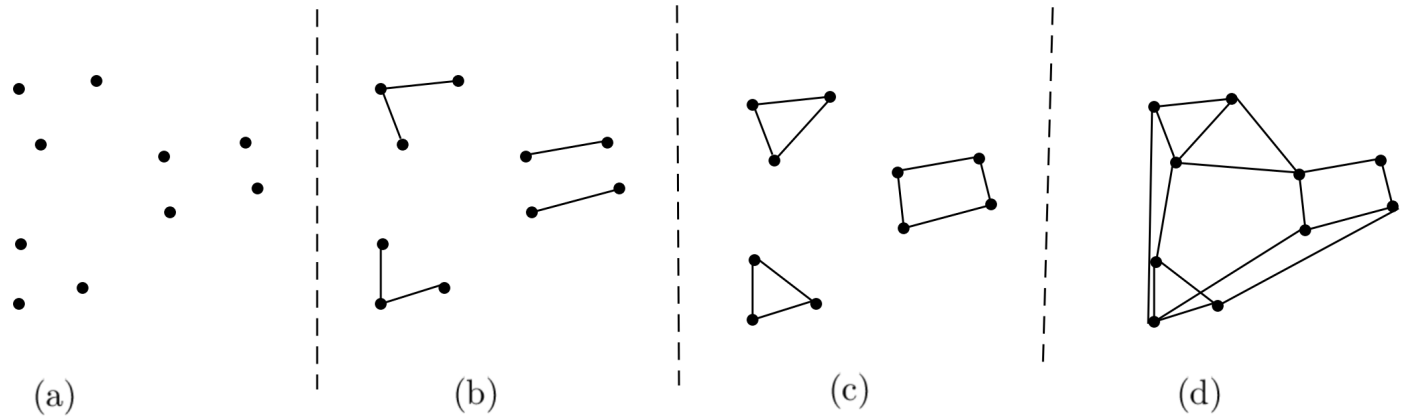
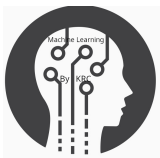


Figure 2: Construction steps of k-nearest neighbour graph using original data: (a) Original data, (b) 1- (c) 2- (d) 3-nearest neighbor graphs

Many advantages of k-NN (clusters). 1. far apart data items are completely disconnected, and 2. since data items are connected with nearer items, weights (on edges) are indicator of population density.



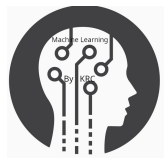
Partitional Algorithms

$\Rightarrow$ It first obtains a single partition of the data, with no structure. In next step, clusters are produced by optimization of a *criterion function* defined locally (over a subset of the patterns).

$\Rightarrow$ "Squared error function" approach is most intuitive concept for partitional clustering, it ideally suited for compact and isolated clusters. For an input set of $\mathcal{X}$ patterns, the "squared error" for clustering $\mathcal{C}$, consisting K clusters $(C_1, ..., C_K)$, expressed as:

$$e^2(\mathcal{X}, \mathcal{C}) = \sum_{j=1}^{K} \sum_{i=1}^{m_j} \| \mathbf{x}_i^{(j)} - \mathbf{c}_j \|^2 . \quad (3)$$

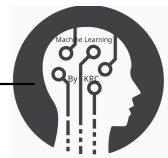
$\Rightarrow$ In the equation (3), $\mathbf{c}_j$ is *centroid* of the j^{th} cluster in total K clusters formed, m_j is number of patterns in j^{th} cluster, and $\mathbf{x}_i^{(j)}$ is the i^{th} pattern in j^{th} cluster.



Squared Error Clustering Algorithm

Algorithm 1 Squared Error Clustering Algorithm

1: Select an initial partition $\mathbf{X}$ of patterns, with a fixed k number of clusters, and cluster centers
2: **repeat**
3: **for** each pattern $\mathbf{x}_i \in \mathbf{X}$ **do**
4: Find centroid $\mathbf{c}_j$ (of cluster C_j) having minimum distance with pattern $\mathbf{x}_i$
5: $C_j = C_j \cup \{\mathbf{x}_i\}$
6: Compute the new centroids (cluster centers) of all the clusters
7: **end for**
8: Merge and split clusters based on some heuristic criterion
9: **until** convergence is achieved
10: **end**

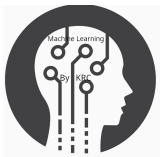


K-Means Clustering

The steps of squared error clustering algorithm are listed in algorithm 1. The repetition in the *repeat* ... *until* loop continues until the convergence is achieved, i.e., the cluster membership is stable.

$\Rightarrow$ The k-means tries to find k number of clusters, the count is specified by the user. These are represented by their centroids. It is simplest and most commonly used algorithm that uses *squared error* criterion.

$\Rightarrow$ The k-means algorithm starts with a random initial partition and keeps reassigning the patterns to clusters based on the similarity between the pattern and the cluster centers (centroid distances) until a convergence condition is reached.



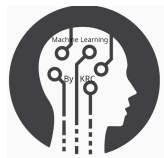
K-Means Clustering..

$\Rightarrow$ The k-means is a partitional clustering technique that tries to find a k number of clusters (count is given by the user). These are represented by their centroids. It is simplest and commonly used algorithm that uses *squared error* criterion.

$\Rightarrow$ k-means algorithm starts with a random initial partition and keeps reassigning the patterns to clusters based on the similarity between the pattern and the cluster centers (centroid distances) until a convergence condition is reached.

$\Rightarrow$ In clustering process, there is no reassignment of any pattern from one cluster to another, this gives gives it a property of *linear time complexity*.

$\Rightarrow$ Advantages of k-means: 1). It is easy to implement, 2. Its time complexity is $O(n)$, where n is total number of patterns. Disadvantage: sensitive to selection of the initial partition – if not properly selected, it may converge to a *local minima* of the criterion function value.



K-Means Clustering: Example

$\Rightarrow$ Using the k-means approach to perform clustering.

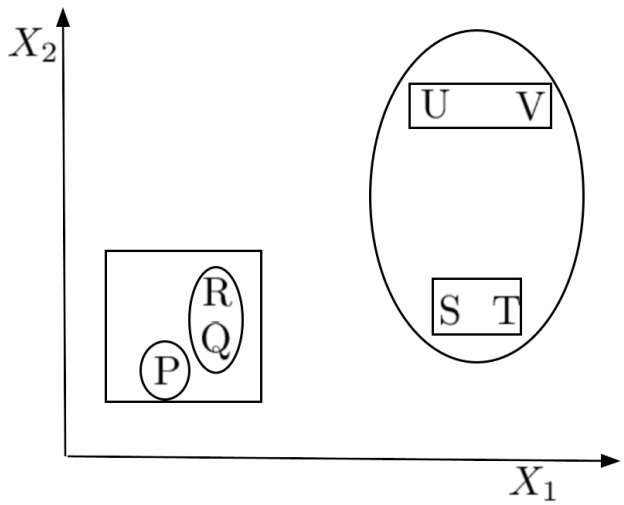
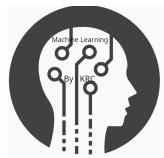


Figure 3: The k-means clustering is sensitive to initial partition

$\Rightarrow$ Fig. shows 2D patterns P, Q, R, S, T, U, V. Process is started with initial patterns P, Q, R. Around these, three (given $k = 3$) clusters are to be constructed. We we end up with partition $\{\{P\}, \{Q, R\}, \{S, T, U, V\}\}$, where three clusters are ellipses.

$\Rightarrow$ The *squared error* criterion value turns out to be very large for this partition (see eqation (3)). This will happen, for the centroid vs. the patterns in the largest ellipse.



K-Means Clustering..

$\Rightarrow$ Hence, we construct a better partition $\{\{P, Q, R\}, \{S, T\}, \{U, V\}\}$, where clusters are shown by rectangles. This grouping results to the global minimum value of the squared error criterion function, for clustering comprising of $k = 3$ clusters.

$\Rightarrow$ The correct three-cluster solution is obtained by choosing, for example, P, S, and U as the initial cluster *means*, which will form the partition as $\{\{P, Q, R\}, \{S, T\}, \{U, V\}\}$.


Chowdhary, K.R. (2020). Data Mining. In: Fundamentals of Artificial Intelligence. Springer, New Delhi. https://doi.org/10.1007/978-81-322-3972-7_17 pp. 519-534.