S21CSE471: Statistical Methods in AI **15/02/21, 01/03/21**

Clustering Algorithms

Prepared by: Nisarg Sheth(2020201049), Rohith R (2020201042), Gorre Shashidhar Reddy(20171113)

1 Introduction

Broadly, Machine Learning tasks are divided mainly into two groups:

- Predictive Tasks
  - Classification
  - Regression
- Descriptive Taks
  - Clustering
  - Dimensionality Reduction
  - Density Estimation

In this document we are going to learn about what clustering is basically and what are some of the famous clustering algorithms, how they work, advantages over each other etc.

So let us start with a real life example of clustering first. You might have noticed that when you open Google News, There are some of the news that are displayed in the same section or sort of "clustered" together.

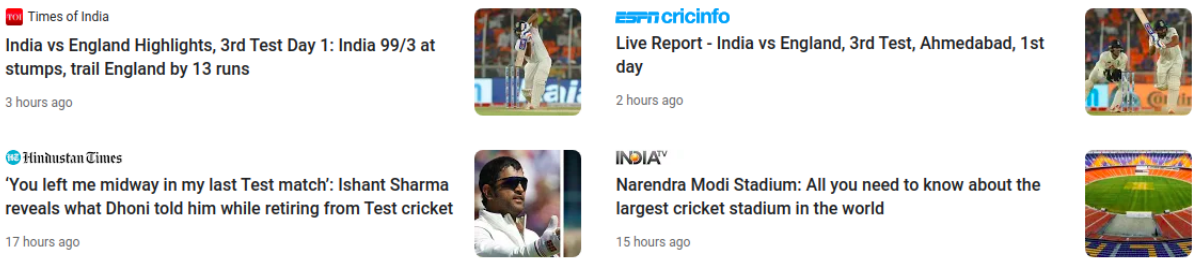


Figure 1: Google News

There's a snapshot of the google news in the above figure that shows the news which are related to the 3rd test match between India and England held at the largest stadium in the world. Note that it is us who gave a sort of label that the news is about the test match but the algorithm knows just that these news are talking about the similar content and nothing else. In other words, clustering is a form of "Unsupervised" learning algorithm which means that the input is not paired with the labelled output.

Another example of clustering problem would be, given a set of images, cluster the similar images together. Or, Given a set of images of people, cluster the images of similar aged people or cluster them on the basis of clothing style, etc.

2 What is Clustering?

Clustering is the task of dividing the data into number of groups such that data points in the same groups are more similar to other data points in the same group than those in other groups.In other words, we want to organise the data in groups such that, there is "high intra group similarity (within a cluster)" and "low inter group similarity(across clusters)".

Clustering can be divided into two parts:

- Hard clustering : In hard clustering, each data point either belongs to a cluster completely or not, there is no in between.
  Example : k-means
- Soft clustering: In soft clustering, nstead of putting each data point into a separate cluster, a probability or likelihood of that data point to be in those clusters is assigned.
  Example : Gaussian Mixture Models (GMMs).

  In this document we are going to look at some of the Hard clustering algorithms.

2.1 Grouping similar images

Grouping can be based on criteria such as similar looking images, size, pixel distribution and similar background etc.

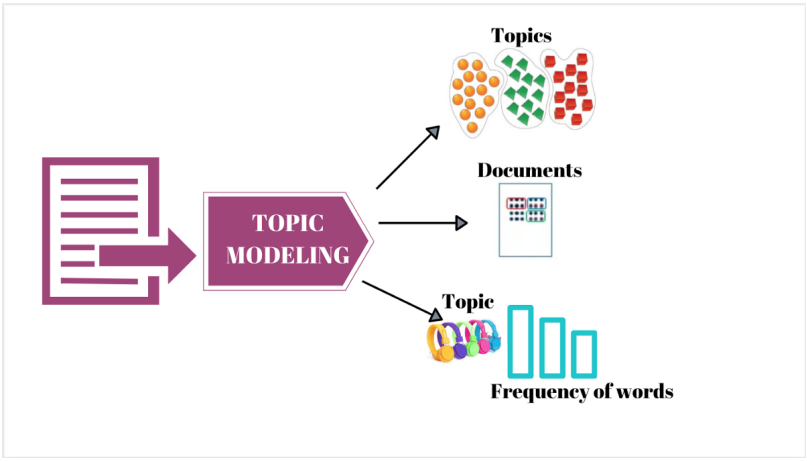
2.2 Grouping types of people

Grouping types of people based on criteria such as clothing styles, gender, age etc.

2.3 Determine moving objects in video

Consider video as a collection of images, and group the pixels that show motion. For example, all the pixels that belong to car move at same speed. We can group them, without being told the category.

2.4 Topic Modelling



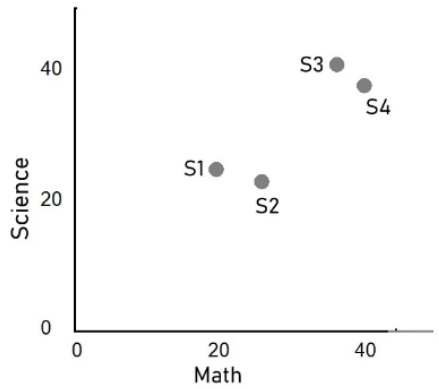
A topic model is a type of statistical model for discovering the abstract "topics" that occur in a collection of documents. Topic modeling is a frequently used text-mining tool for discovery of hidden semantic structures in a text body.

3 Agglomerative Clustering

Agglomerative clustering is a form of heirarchical clustering algorithm which is based on bottom up approach and it is a type of hard clustering algorithm. That means it starts with many small clusters and merges them together to create bigger clusters. Let us now look in detail how agglomerative clustering works.

	MATH	SCIENCE
S1	20	25
S2	25	22
S3	35	40
S4	40	35

(a) Data



(b) Representation

Figure 2: Marks data

The above figure shows the data related to marks scored by 4 students in Math and Science and we need to create clusters of students to draw insights.
As shown in the figure, initially we consider each data point as a seperate cluster in itself.
Next, we compute the distance of each data point with every other data point. For that we will have to first decide upon a distance metric. Let us go with Euclidean distance for the following example.

$$d_{L2}(x, y) = \sqrt{\sum_{i=1}^{i=n}(x_i - y_i)^2}$$

	S1	S2	S3	S4
S1	0	5.83	21.21	22.36
S2	5.83	0	20.59	19.84
S3	21.21	20.59	0	7.07
S4	22.36	19.84	7.07	0

Forming a cluster between S1 and S2.

Figure 3: Iteration 1

As we can see the distance between S1 and S2 is minimum so we now merge them together and create a new cluster. The figure besides the distance metric that connects S1 and S2 is called a "dendogram".

Now the question arises, after merging S1 and S2 how do we represent both as a single cluster? One way is we take the average of the points in the cluster and denote the cluster with that value. Depending on how do we represent the clusters, it is given different names:

Different methods to compute distances between clusters:

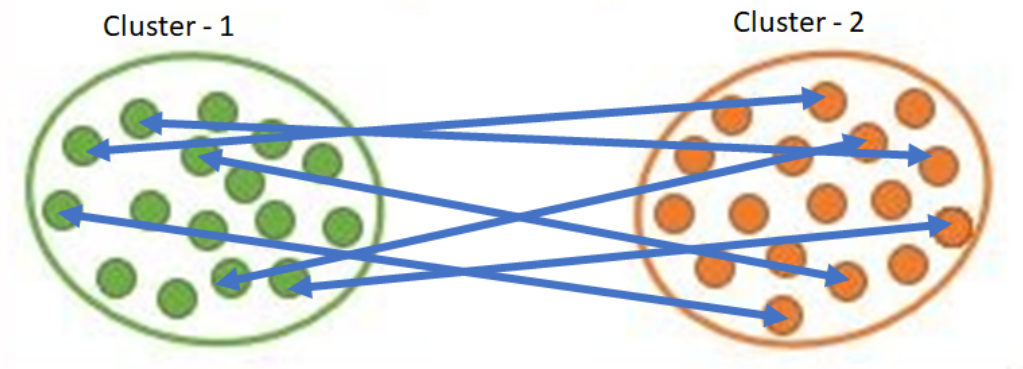
The linkage creation step in Agglomerative clustering is where the distance between clusters is calculated. So basically, a linkage is a measure of dissimilarity between the clusters. There are several methods of linkage creation. Some of them are:

Single linkage: The distance between the two clusters is the minimum distance between clusters'

data points.

Complete linkage: The distance between two clusters is the maximum distance between clusters' data points.

Average linkage: The distance between clusters is the average distance between each data point in one cluster to every data point in the other cluster.



For this example let us use average link clustering. Coming back to example,

	Math	Sci
[S1,S2]	22.5	23.5
S3	35	40
S4	40	35

(a) Data

	[S1,S2]	S3	S4
[S1,S2]	0	20.7	20.9
S3	20.7	0	7.07
S4	20.9	7.07	0

(b) Distance matrix

Figure 4: Data after iteration 1

Again let us find the closest points and create another cluster.

	[S1,S2]	S3	S4
[S1,S2]	0	20.7	20.9
S3	20.7	0	7.07
S4	20.9	7.07	0

Forming Cluster between S3 and S4.

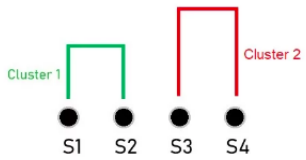


Figure 5: Iteration 2

Finally we are left with just two clusters [S1,S2] and [S3,S4] and so we merge them together and our final dendogram will look something like this:

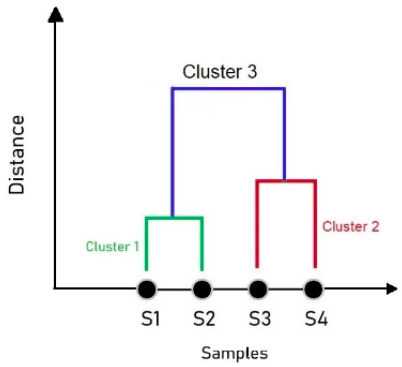


Figure 6: Final Dendogram

Time complexity : In each iteration we find distance of each point with every other point so it is roughly $O(n^2)$ and we have $n-1$ iterations so $O(n^3)$.
With priority queue the complexity can be brought down to $O(n^2 logn)$

4 K-means clustering

K-Means algorithm is a centroid based clustering technique. It is an iterative algorithm that tries to partition the dataset into K pre-defined distinct non-overlapping subgroups (clusters) where each data point belongs to only one group which means that K-means is also an example of hard clustering algorithm.
The centroid point is the point that represents its cluster. Centroid point is the average of all the points in the set and will change in each step and will be computed by:

$$C_i = \frac{1}{||S_i||} \sum_{x_j \in S_i} x_j$$

$C_i = i$ th centroid
$S_i =$ All points belonging to set i with centroid as C_i
$x_j =$ j'th point from the set
$||S_i|| =$ number of points in set i

K-means algorithm works as follows:

1. Specify number of clusters K.
2. Initialize centroids by first shuffling the dataset and then randomly selecting K data points for the centroids.
3. Compute the sum of the squared distance between data points and all centroids.
4. Assign each data point to the closest cluster (centroid).
5. Compute the centroids for the clusters by taking the average of the all data points that belong to each cluster.
6. Keep iterating until convergence i.e. there is no change to the centroids. i.e assignment of data points to clusters isn't changing. i.e. do
   Assignment step: For every set i, set

$$c^{(i)} = \arg\min_j ||x^{(i)} - \mu_j||^2$$

   Refitting step : Move each cluster center to the center of the data assigned to it : For each j, set

$$\mu_j = \frac{\sum_{i=1}^{i=m} 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^{i=m} 1\{c^{(i)} = j\}}$$

   Let us take an example and apply the algorithm for better understanding.

   We are given the following data points we want 3 clusters.
   $A1 = (2, 10), A2 = (2, 5), A3 = (8, 4), A4 = (5, 8), A5 = (7, 5), A6 = (6, 4), A7 = (1, 2), A8 = (4, 9)$.

   Let us assume initial centroids are $A1, A4$ and $A7$ and we are using euclidian distance as the distance metric.

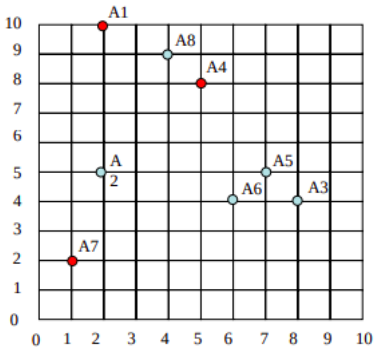


Figure 7: Initialization

Let us compute the distance of every point with these clusters.

	A1	**A4**	**A7**
A1	0	3.605551275	8.062257748
A2	5	4.24	3.16
A3	6	5	7.28
A4	3.605551275	0	7.211102551
A5	7.07	3.6	6.7
A6	7.21	4.12	5.38
A7	8.062257748	7.211102551	0
A8	2.236067977	1.414213562	7.615773106

Figure 8: Assigning clusters

The shaded cell represents the assigned cluster.
Thus we got new clusters as $1 : \{A1\}, 2 : \{A3, A4, A5, A6, A8\}, 3 : \{A2, A7\}$ Let us now find the centroids of each cluster.
$C1 = (2, 10)$
$C2 = ((8 + 5 + 7 + 6 + 4)/5, (4 + 8 + 5 + 4 + 9)/5) = (6, 6),$
$C3 = ((2 + 1)/2,\ (5 + 2)/2) = (1.5, 3.5)$

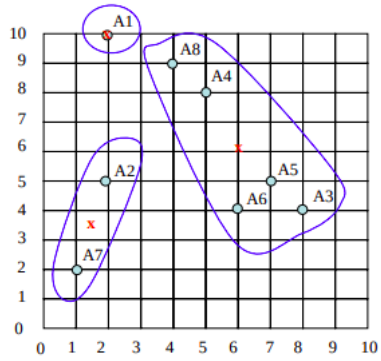


Figure 9: Iteration 1

	C1(2,10)	C2 (6,6)	C3(1.5,3.5)
A1	0	5.65	6.519
A2	5	4.12	1.58
A3	8.48	2.82	6.51
A4	3.6	2.23	5.7
A5	7.07	1.41	5.7
A6	7.21	2	4.52
A7	8.06	6.4	1.58
A8	2.23	3.6	6.04

Figure 10: Assigning clusters

Thus we got new clusters as $1 : \{A1, A8\}, 2 : \{A3, A4, A5, A6\}, 3 : \{A2, A7\}$ Let us now find the centroids of each cluster.
$C1 = ((2 + 4)/2, (10 + 9)/2) = (3, 9.5)$
$C2 = ((8 + 5 + 7 + 6)/4, (4 + 8 + 5 + 4)/4) = (6.5, 5.25),$
$C3 = ((2 + 1)/2,\ (5 + 2)/2) = (1.5, 3.5)$

	C1(3,9.5)	C2 (6.5,5.25)	C3(1.5,3.5)
A1	1.11	6.54	6.519
A2	4.6	4.5	1.58
A3	7.43	1.95	6.51
A4	2.5	3.13	5.7
A5	6.02	0.55	5.7
A6	6.26	1.34	4.52
A7	7.76	6.38	1.58
A8	1.118	4.5	6.04

Figure 11: Assigning clusters

Thus we got new clusters as $1 : \{A1, A4, A8\}, 2 : \{A3, A5, A6\}, 3 : \{A2, A7\}$
After this iteration the algorithm converges and final clusters look like below: Centroids of each cluster.
$C1 = (3.66, 9)$
$C2 = (7, 4.33),$
$C3 = (1.5, 3.5)$

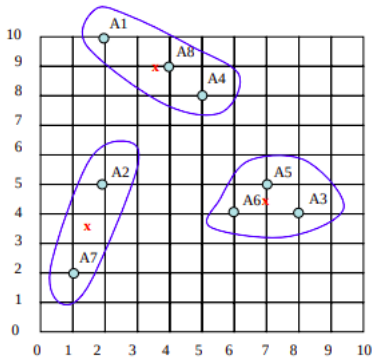


Figure 12: Final Clusters

5 Advantages of K-means Clustering

- Relatively simple to implement.

- Scales to large data sets.
- Guarantees convergence.
- Can warm-start the positions of centroids.
- Easily adapts to new examples.
- Generalizes to clusters of different shapes and sizes, such as elliptical clusters.

6 Drawback of standard K-means Clustering

One disadvantage of the K-means algorithm is that it is sensitive to the initialization of the centroids or the mean points.In K-means clustering to find initial centroid we were using randomization,i.e the initial centroids are picked randomly from data points. This randomization of picking k-centroids points results in the problem of initialization sensitivity.
This problem affects the final formed clusters. The final formed clusters depend on how initial centroids were picked.
So, if a centroid is initialized to be a "far-off" point, it might just end up with no points associated with it, and at the same time, more than one cluster might end up linked with a single centroid.
Similarly, more than one centroids might be initialized into the same cluster resulting in poor clustering.
For example, consider the images shown below. A poor initialization of centroids resulted in poor clustering.

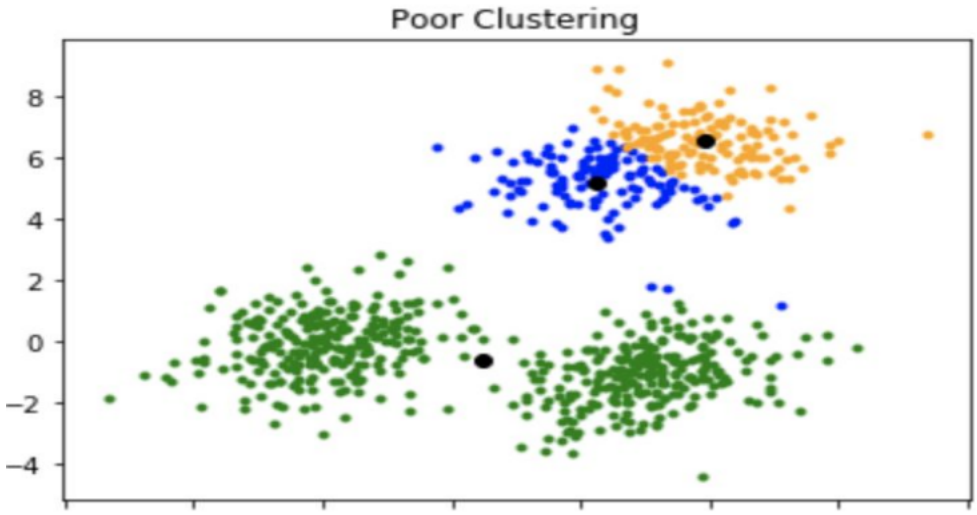


Figure 13: Poor Clustering

This is how the clustering should have been:

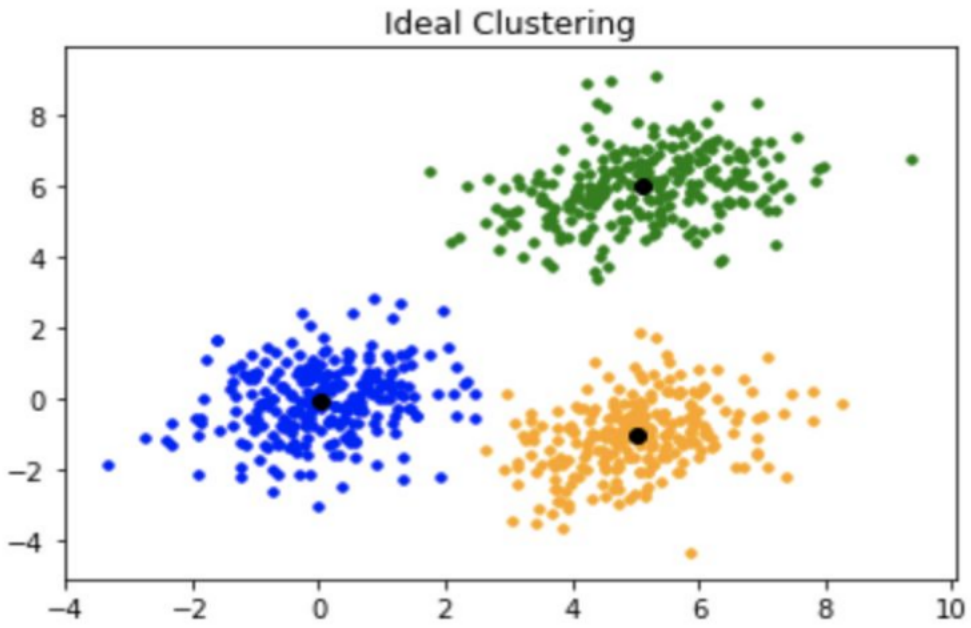


Figure 14: Ideal Clustering

We use K-means++ to solve this problem.

7 K-means++ Clustering

This algorithm ensures a smarter initialization of the centroids and improves the quality of the clustering. Apart from initialization, the rest of the algorithm is the same as the standard K means algorithm. That is K means++ is the standard K means algorithm coupled with a smarter initialization of the centroids. K means++ is an algorithm for choosing the initial values for the k-means clustering algorithm.

8 How K-means++ works?

- Randomly select the first centroid (C_1) from the data points.
- For each data point compute its distance from the nearest,previously chosen centroid.

  $$d_i = max_{j:1 \to m} ||x_i - C_j||^2$$

  (Distance between the data point and the centroid can be calculated by the above formulae where X_i is the data point.)
- Select the next centroid from the data points such that the probability of choosing a point as centroid is directly proportional to its distance from the nearest, previously chosen centroid. (i.e. the point having maximum distance from the nearest centroid is most likely to be selected next as a centroid).
- Repeat steps 2 and 3 until k centroids have been sampled.

By following the above procedure for initialization, we pick up centroids that are far away from one another. This increases the chances of initially picking up centroids that lie in different clusters. Also, since centroids are picked up from the data points, each centroid has some data points associated with it at the end.

Note - Although the initialization in K-means++ is computationally more expensive than the standard K-means algorithm, the run-time for convergence to optimum is drastically reduced for K-means++. This is because the centroids that are initially chosen are likely to lie in different clusters already.

References

[1] https://towardsdatascience.com/understanding-k-means-k-means-and-k-medoids-clustering-algorithms-ad9c9fbf47ca

[2] https://towardsdatascience.com/k-means-clustering-for-beginners-2dc7b2994a4

[3] https://towardsdatascience.com/machine-learning-algorithms-part-12-hierarchical-agglomerative-clustering-example-in-python-1e18e0075019