Data Mining

Mohammed Almulla

College of I.T

Basic techniques for cluster detection

After learning to preprocess our data, let us learn some of the methods that can be applied to our data set. The technique or method that I will talk about is clustering or cluster detection. Clustering is an unsupervised learning method that take no labels. But, what a cluster mean? It is a group of objects known as members. The center of a cluster is known as the centroid. And members of a cluster are similar to each other, and of different clusters are different. Clustering is a process of discovering clusters by gathering data samples and group them into similar records using predefined distance measures. Figure 39 shows clustering.

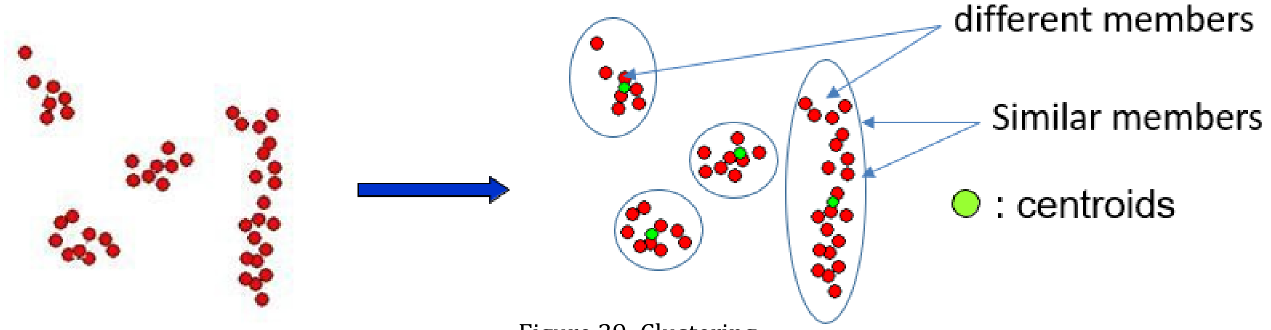
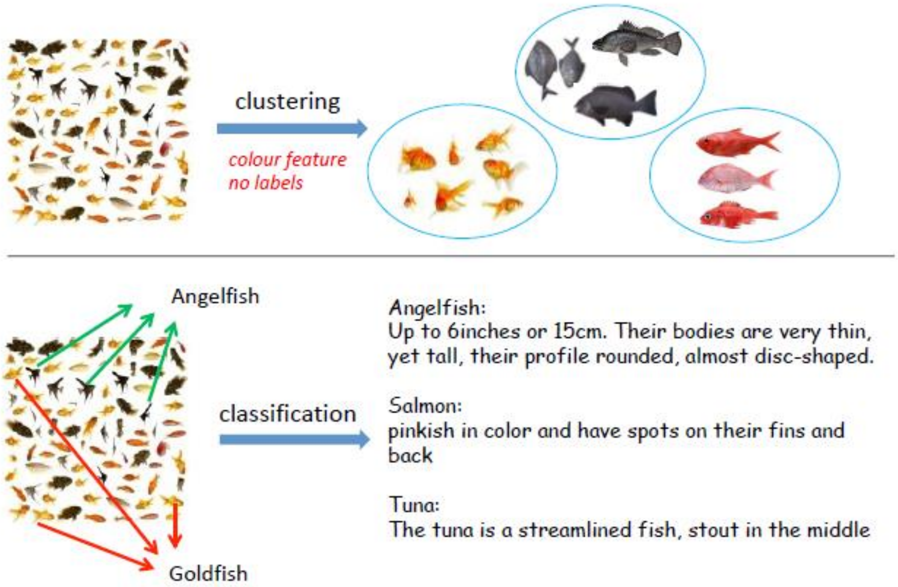


Figure 39: Clustering

Clustering is used in various applications like image processing, economics, recommendation systems. Also, search engines is use clustering for like group related documents for browsing. Clustering can also be used in summarizing data to reduce the size of large data sets such as clustering precipitation in Australia as shown in figure 40.

Figure 40: Clustering precipitation in Australia

There are differences between Clustering and Classification. First, clustering is unsupervised while classification is supervised. Furthermore, classification assign objects based on predefined classes or labels while clustering identify similarities between objects and assign them into groups known as clusters. Figure 41 shows an example on classification vs. clustering.



Angelfish:
Up to 6inches or 15cm. Their bodies are very thin, yet tall, their profile rounded, almost disc-shaped.

Salmon:
pinkish in color and have spots on their fins and back

Tuna:
The tuna is a streamlined fish, stout in the middle

Figure 41: Example on classification vs. clustering

There are many types of clustering categorize based into the following:

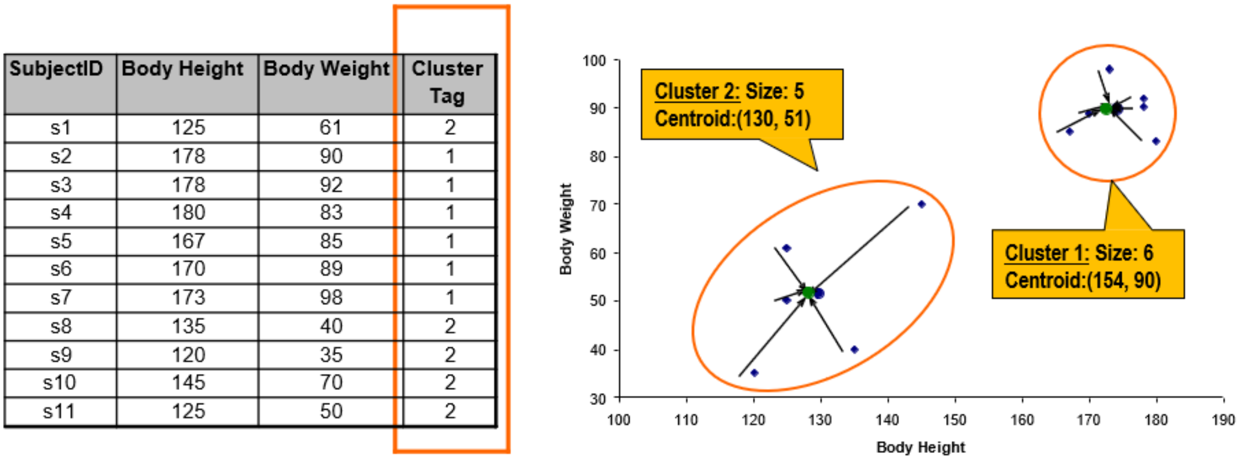
- Goal:
  - Monothetic: cluster members have some common property, e.g., all are males aged 20-35, or all have X% response to test B.
  - Polythetic: cluster members are similar to each other, e.g., distance between elements defines membership.
- Overlap:
  - Hard clustering: clusters do not overlap, element either belongs to a cluster or not.
  - Soft clustering: clusters may overlap, “strength of association” between element and cluster.
- Flat or Hierarchical
  - Set of groups vs. taxonomy

Moreover, to have an ideal clustering algorithm, we need to have a certain of requirements as follows:

1. Scalability: means the speed of execution, that it needs to be able to deal with different types of attributes, and to discover clusters of arbitrary shapes.
2. Minimal requirements for domain knowledge: It need to determine input parameters, and to be able to deal with noise and outliers.

3. Insensitive to order of input data records: it needs to be able to deal with high dimensionality, and incorporation of user-specified constraints.
4. Interpretability and usability: that good algorithm produces good, informative, reasonable small clusters.

The outputs of cluster detection process are assigned cluster tag for members of a cluster, and it give cluster summary e.g., size, centroid, variations, etc. Figure 42 shows a data set with cluster tag.



SubjectID	Body Height	Body Weight	Cluster Tag
s1	125	61	2
s2	178	90	1
s3	178	92	1
s4	180	83	1
s5	167	85	1
s6	170	89	1
s7	173	98	1
s8	135	40	2
s9	120	35	2
s10	145	70	2
s11	125	50	2

Figure 42: Clustering output

The basic elements of a clustering solution are as follows:

1. A sensible measure for similarity, e.g., Euclidean distance.
2. An effective and efficient clustering algorithm, e.g., K-means.
3. A goodness-of-fit function for evaluating the quality of resulting clusters, e.g., Error Sum of Squares (SSE) where it could be Within-Cluster variation (WC) or inter-cluster variation, and Between Clusters (BC) or intra-cluster variation.

I will explain each one of them below.

Measures of Similarity or Proximity

Let us start with the basics. Proximity between two data objects is represented by either similarity or dissimilarity. Where similarity is a numeric measure of the degree of alikeness, and dissimilarity is a numeric measure of the degree of difference between two objects. A distance dissimilarity measure d is called metric when the following properties satisfy:

1. $d(x,y) \geq 0$
2. $d(x,y) = 0 \leftrightarrow x = y$
3. $d(x,y) = d(y,x)$

4. $d(x,z) \leq d(x,y) + d(y,z)$

Where d is the distance (dissimilarity) between points (data objects), x and y.

Furthermore, the common properties for similarity measure s are:

1. $s(x,y) = 1 \leftrightarrow x = y$
2. $s(x,y) = s(y,x)$

Also, distance metric d can be converted to similarity as follows:

$$sim(x,y) = \frac{1}{1 + d(x,y)'}$$

Where 1 means identical or match.

Each attribute type has its own distance measurement, I will explain below distance measurement for nominal, numeric, binary attributes. Also, I will explain cosine similarity.

Distance Measurement for Nominal Attributes

We will use ratio of mismatched features (RMF) to calculate the distance between two nominal attributes.

$$d(x,y) = \frac{p - m}{p}$$

Where x, y are two data objects, p are the number of the nominal attributes, and m are the number of attributes that have a mismatched value.

Distance Measurement for Numeric Attributes

We will use Minkowski function to measure the distance between two numeric attributes.

$$d(x,y) = \sqrt[q]{|x_1 - y_1|^q + |x_2 - y_2|^q + \cdots + |x_n - y_n|^q}$$

Where q is any whole number range [0, inf).

There are three special cases for Minkowski function as follows:

1. Manhattan distance, where $q = 1$.
2. Euclidean distance, where $q = 2$
3. Chebyshev distance: $d(x,y) = max_i|x_i - y_i|$

Distance Measurement for binary attributes

We will use two distance laws as follows:

1. Simple Mismatch Coefficient (SMC):

$$SMC(i,j) = \frac{f_{01} + f_{10}}{f_{01} + f_{10} + f_{00} + f_{11}} = \frac{\#Mismatch}{\#Attributes}$$

2. Jaccard coefficient (JC):

$$JC(i,j) = \frac{f_{01} + f_{10}}{f_{01} + f_{10} + f_{11}} = \frac{\#Mismatch}{\#Attributes\ with\ 1}$$

Cosine Similarity

It is generally used as a metric for measuring distance when the magnitude of the vectors does not matter. This happens for example when working with text data represented by word counts. We could assume that when a word (e.g., science) occurs more frequent in document 1 than it does in document 2, that document 1 is more related to the topic of science. However, it could also be the case that we are working with documents of uneven lengths (Wikipedia articles for example). Then, science probably occurred more in document 1 just because it was way longer than document 2. Cosine similarity corrects for this.

$$Similarity = \cos\theta = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}}$$

Combining Heterogeneous Attributes

In real-life databases it may contain all attribute types such as nominal, numeric, ordinal, etc. So, we need a formula to combine them. It is based on the principle of ratio of mismatched features (RMF). For the k^{th} attributes, compute the dissimilarity d_k in [0,1].

$$d_k = d(i,j) = \frac{|x_{ik} - x_{jk}|}{max - min}$$

Set the indicator variable δ_k as follows:

- $\delta_k = 0$, if the k^{th} attribute is an asymmetric binary attribute and both objects have value 0 for the attribute.
- $\delta_k = 1$, otherwise.

Then, compute the overall distance between i and j.

$$d(i,j) = \frac{\sum_{k=1}^{n} \delta_k \times d_k}{\sum_{k=1}^{n} \delta_k}$$

Attribute Scaling

It is original attribute values are transformed to new values. Same as feature transformation in the previous section (Data, Preprocessing, and Exploration). It is carried when these two reasons satisfy:

1. same attribute may be measured in different scales when data from different data sources are merged i.e., Fahrenheit-Centigrade conversion.
2. on different attributes with different scales when data is projected into the N-space.

Normalizing variables into comparable ranges divide each value by the mean, divide each value by the range, and z-score.

Furthermore, attribute weighting reflects different degrees of importance on those attributes. And the weighted overall dissimilarity function as follows.

$$d(i,j) = \frac{\sum_{k=1}^{n} w_k \times \delta_k \times d_k}{\sum_{k=1}^{n} \delta_k}$$

K-means

It is a clustering algorithm that partition n objects into k clusters in which each object belongs to the cluster with the nearest mean. The objective of K-means is to minimize total intra-cluster variance or the squared error function. The steps for K-means algorithm as follows:

1. Start
2. Decide number of clusters.
3. Find the centroids.
4. Take each record and calculate the distance e.g., Euclidean distance from all centroids.
5. Select minimum distance.
6. Update the clusters centroid.
7. Update record with cluster number.
8. Stop after executing all records.

The clustering criterion (E) is to minimize the Euclidean sums of squared deviations of objects from the cluster mean, which is defined as follows:

$$E = \sum\nolimits_{i=1}^{k} \sum\nolimits_{p \in C_i} (p - c_i)^2$$

As shown in figure 43, how K-means cluster data objects.

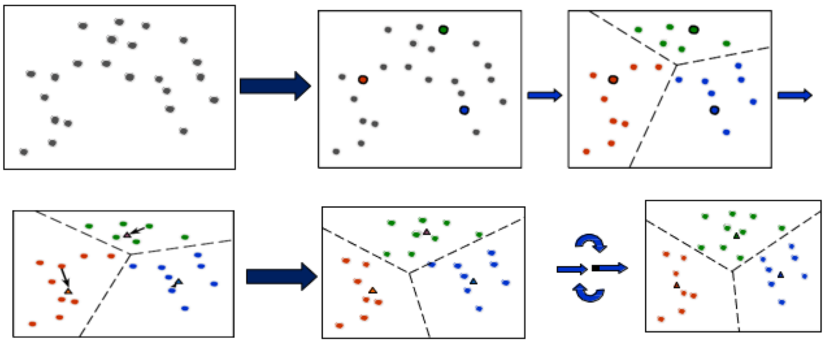


Figure 43: K-means

Let us have an example on K-means.

Example: The table below records the number of house sales and the total revenue generated by branches of a state agency chain. I will perform K-means algorithm (First iteration) on the data.

branch No	Total Sales Quantity	Total Sales Values
b1	29	5500000
b2	10	5000000
b3	29	5000000
b4	12	890000
b5	20	2500000
b6	20	3200000
b7	15	678000
b8	29	5200000
b9	30	5300000
b1	29	5500000
b2	10	5000000

1. First, I will delete the last two records because they are the same as the first two. Then, I will use Excel to normalize the data by making the values of the attributes range [0, 1]. Then, I will multiply it by 100 to be [0, 100].

$$Normalization = \frac{x - min}{max - min}$$

branch No	Total Sales Quantity	Total Sales Values
b1	95	100
b2	0	89.63085856
b3	95	89.63085856
b4	10	4.396515968
b5	50	37.78515139
b6	50	52.3019494
b7	25	0
b8	95	93.77851514
b9	100	95.85234343

2. Now we will start playing in our data by using K-means steps above. So, I will decide the number of clusters k to be 3. And randomly the initial centroids are b4, b6, and b9.

b4	10	4.396515968
b6	50	52.3019494
b9	100	95.85234343

3. The, calculate the distance between the first record with each centroid using Euclidean distance. And choose the minimum distance. Then, update the centroid for the clusters by calculating the mean for the clusters. And for the next record calculate the distance between it and the new centroids.

branch No	Total Sales Quantity	Total Sales Values	Belongs to	Distance C1	Distance C2	Distance C3
b1	95	100	C3	127.925862	65.575178	6.4963878
b2	0	89.63085856	C2	85.8189557	62.397496	100.19335
b3	95	89.63085856	C3	120.3739721	58.467491	7.9816586
b4	10	4.396515968				
b5	50	37.78515139	C2	52.10375203	14.516798	76.627663
b6	50	52.3019494				
b7	25	0	C1	15.63103812	57.969767	121.70732
b8	95	93.77851514	C3	123.3456192	61.198901	5.413018
b9	100	95.85234343				

4. The last clusters will be as follows.

C1	17.5	2.198257984
C2	37.5	54.37577769
C3	95.625	93.77851514

K-means has advantages that it is simple and easy to implement, and quite efficient. While it has many disadvantages that it needs to specify the value of K, but we may not know what the value should be beforehand. Also, sensitive to the choice of initial K centroids that the result can be non-deterministic mean that it will be changeable due to the randomly chosen centroids. It is sensitive to noise, and applicable only when mean is meaningful to the given data set.

To overcome the weaknesses of K-means we need to improve how initial K centroids are chosen by running the clustering a number of times and select the result with highest quality and finding centers that are farther apart. We need also to deal with noise by removing outliers before clustering using K-medoid method, using the nearest data object to the virtual center as the centroid. Also, when mean cannot be defined, we can use K-mode method that is calculating mode instead of mean for the center of the cluster.

Cluster Evaluation and Interpretation

The principle of cluster quality is high-level similarity and low-level variation within a cluster, and high-level dissimilarity between clusters. The measures we will use for cluster quality are:

- Cohesion: sum of squared errors (SSE), and sum of SSEs for all clusters within cluster (WC).
- Separation: sum of distances between clusters (BC).

Combining the cohesion and separation, the ratio BC/WC is a good indicator of overall quality. The greater the ratio is, the more quality the clusters are. As shown in figure 44 cohesion and separation.

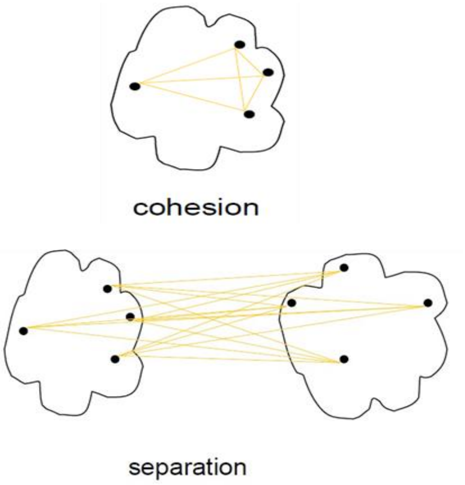


Figure 44: cohesion and separation

$$wc(C_k) = \frac{1}{|C_k|} \sum_{x \in C_k} d(x, r_k)^2$$

$$WC = \sum_{k=1}^{K} wc(C_k)$$

Where x is a member and r_k is the centroid of the cluster, and C_k is the number of clusters.

$$BC = \sum_{1 \le j \le k \le K} d\left(r_j, r_k\right)^2$$

The cluster interpretation requires domain expert, within-cluster interpretation that is summarizing the characteristics of the members of a cluster, outside-cluster interpretation that discover abnormality of small number of data points (outliers) and sometimes, outliers is the focus to be discovered (fraud detection, IDS), and it requires between-cluster interpretation that is comparing clusters and comparing values of important attributes, can help why clusters are formed. Figure 45 shows visual comparison of attribute value distributions.

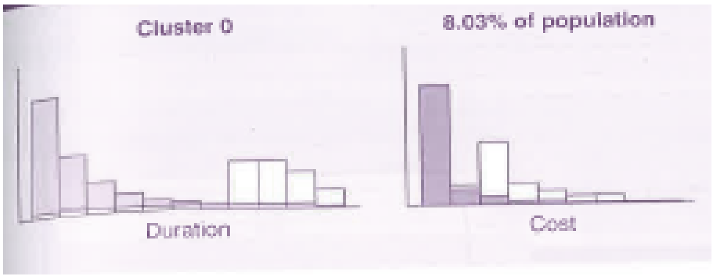
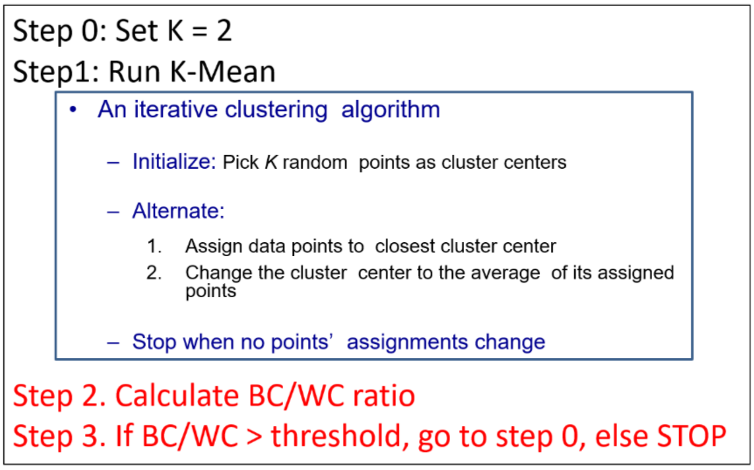


Figure 45: visual comparison of attribute value distributions

To use cluster quality for clustering follow the algorithm below in figure 46.



Step 0: Set K = 2

Step1: Run K-Mean

- An iterative clustering algorithm
  - Initialize: Pick *K* random points as cluster centers
  - Alternate:
    1. Assign data points to closest cluster center
    2. Change the cluster center to the average of its assigned points
  - Stop when no points' assignments change

Step 2. Calculate BC/WC ratio

Step 3. If BC/WC > threshold, go to step 0, else STOP

Figure 46: Using cluster quality for clustering

We have three variations of K-means:

1. K-Modes that uses the mode instead of the mean for each attribute. And it is suitable for categorical attributes.
2. K-Medoids that instead of virtual centroid (mean of data objects), the nearest data object to the virtual mean point is taken as in the centroid.
3. Bisecting K-Means that apply K-Means with K=2 repeatedly on the data set or clusters. And clusters with high WC (SSE) is further clustered (using 2-Means). It stops when reaching number of clusters.

K-Medoids algorithm is as follows (Figure 47 shows K-Medoids with K = 3):

1. Initialization: randomly select k of the m data points as the medoids
2. Assignment: associate each data point to the closest medoid based on some distance measurement (e.g., Minkowski).
3. Update:
    a. for each medoid j and each data point i associated with j, swap j and i and compute the total cost of the configuration (which is, the average dissimilarity of i to all the data points associated to j).
    b. Select the medoid j with the lowest cost of the configuration.
    c. Iterate between steps 2 and 3 until there is no change in the assignments.

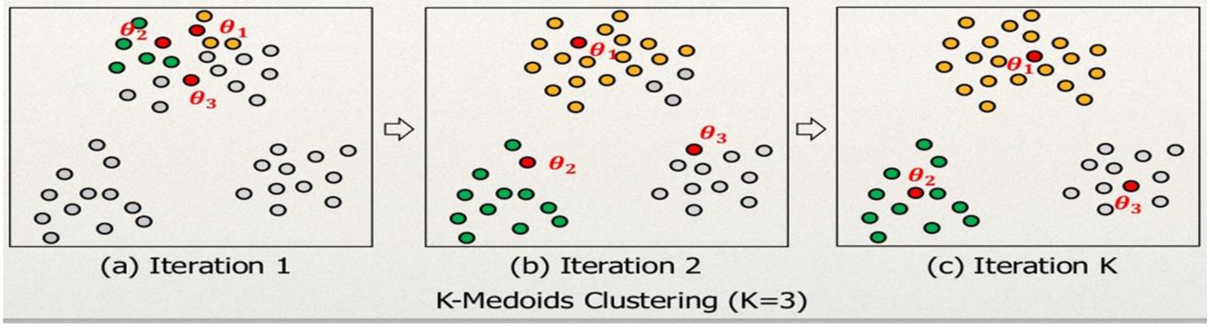


Figure 47: K-Medoids with K = 3

Bisecting K-Means algorithm as follows (Figure 48 shows Bisecting K-Means):

1. Initialize the list of clusters to accommodate the cluster consisting of all points.
2. **Repeat**
3. Discard a cluster from the list of clusters.
4. {Perform several "trial" bisections of the selected cluster.}
5. **for** *i* = 1 to *number of trials* do
6. Bisect the selected clusters using basic K-means.
7. **end for**
8. Select the 2 clusters from the bisection with the least total SSE.
9. **until** Until the list of clusters contain '*K*' clusters

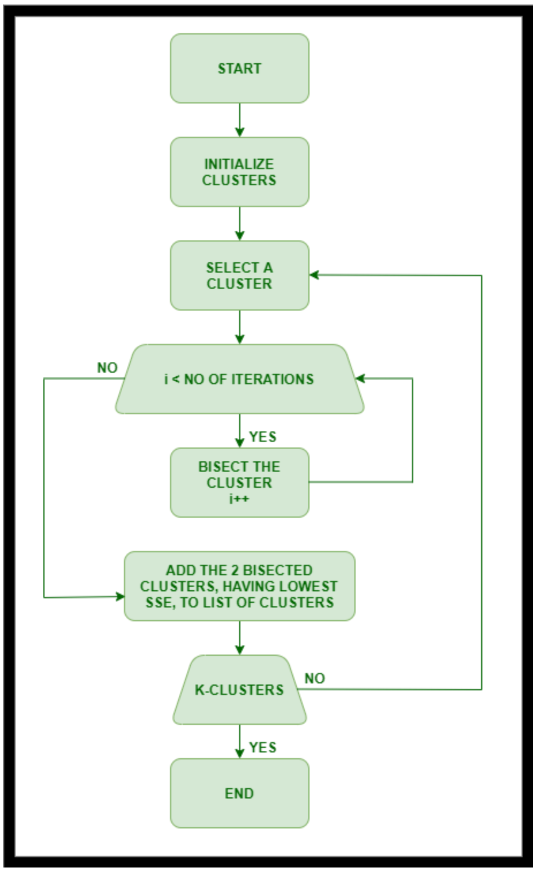


Figure 48: Bisecting K-Means Algorithm

Example on Bisecting K-means: Consider a cluster 'G' = (P1, P2, P3, P4, P5, P6, P7, P8, P9, P10), consisting of 10 points. K=3 (Figure 49 shows example on bisecting K-means).

1. Applying the Bisecting K-Means Algorithm, the cluster 'G', as shown in [A]th step is split into two clusters – 'G1' and 'G2', as shown in [B]th step. The total number of clusters required at the final stage i.e., 'K'=3 (given).
2. Since the required clusters are not obtained yet, we should split one of the two clusters obtained.
3. The cluster with the higher SSE is selected (since the cluster with lower SSE is less erroneous). Here, the cluster with higher SSE is the cluster 'G1'. It is split into (G1)` and (G1)" respectively.
4. Thus, the required number of clusters is obtained.

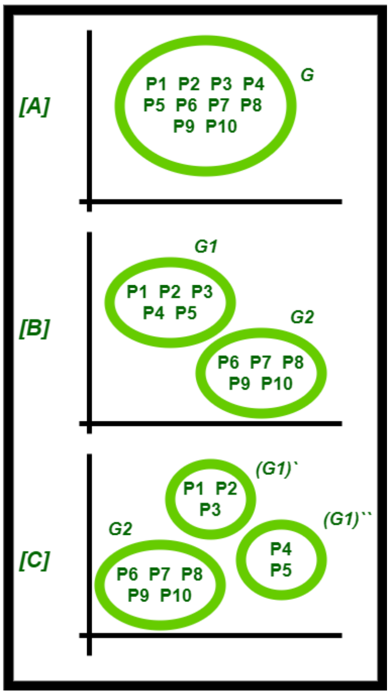


Figure 49: Example on Bisecting K-Means

The K-means variations that I explain it above are flat clustering because each point can belong to one cluster. Also, we have hierarchical clustering that produces hierarchy of nodes. Figure 50 shows flat vs. hierarchical clustering.

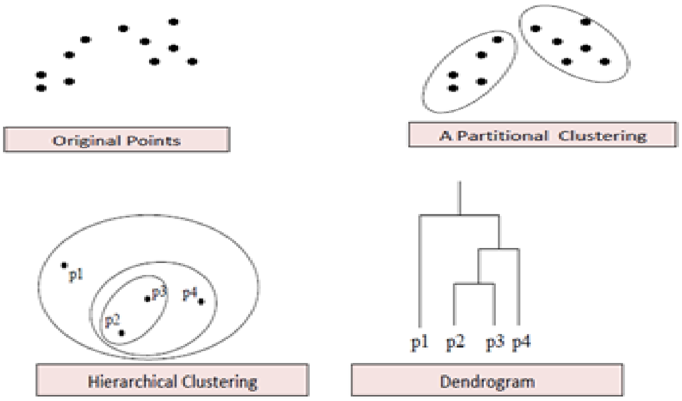


Figure 50: flat vs. hierarchical clustering

Hierarchical clustering is a Sequence of possible grouping rather than single partitioning of the data population space. It produces a nested sequence of clusters, a tree, also called dendrogram. The Proximity matrix must be maintained. Different levels of clusters grouping cannot be undone. The algorithm of hierarchical clustering as follows:

1. Start with the assumption that each data point is a cluster.
2. Merge clusters.

3. Repeat until you reach all group of all clusters.

The drawback of hierarchical clustering is that it requires memory and computational resources. Figure 51 shows hierarchical clustering algorithm. And figure 52 shows cluster dendrogram

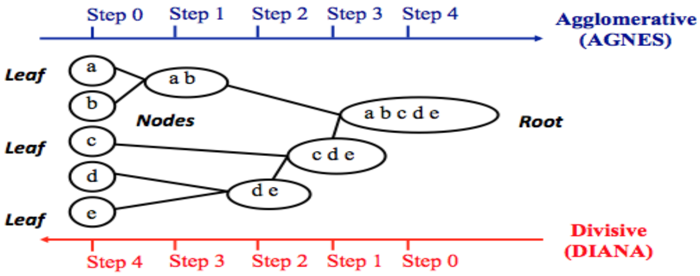


Figure 51: Hierarchical clustering algorithm

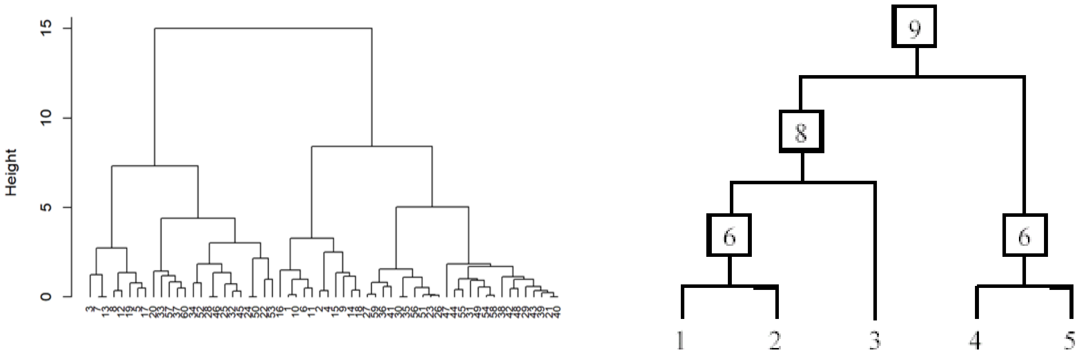


Figure 52: cluster dendrogram

One of the methods uses hierarchical clustering is agglomerative nesting clustering (AGNES). Agglomerative clustering or also known as bottom-up clustering it builds the dendrogram (tree) from the bottom level and merges the most similar (or nearest) pair of clusters. It stops when all the data points are merged into a single cluster (i.e., the root cluster). Since there is a merging cluster step, we need a metric to merge clusters that we will use linkage metrics. The approaches in linkage metrics are as follows (Figure 53 shows linkage metrics):

- Single Link: distance between the closest pair of points in the two clusters (minimum distance).
- Complete Link: distance between the farthest pair of points in the two clusters (maximum distance).
- Group Average: the average distances of all distances between pairs from the two clusters.

- Centroid Method: distance between two centroids.

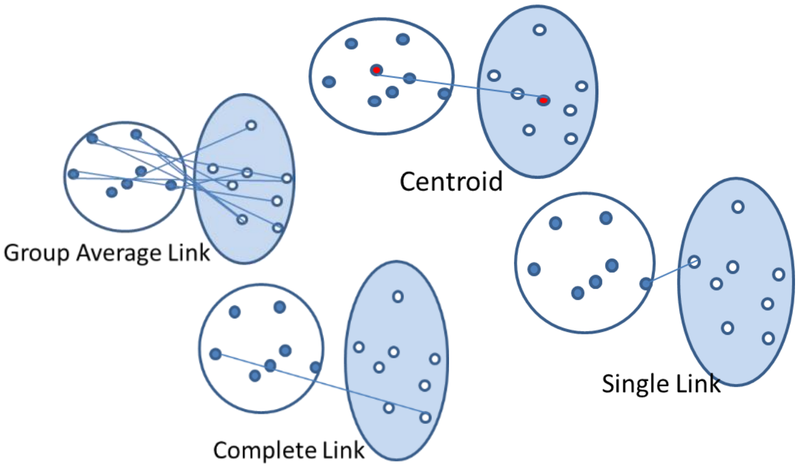


Figure 53: Linkage Metrics

The agglomerative clustering algorithm is as follows:

- Input: set of data points
- Steps:
  1. Calculate Distance Matrix: distance between each two points in the data set.
  2. Select 2 closest or nearest points and merge them in one cluster.
  3. Update the distance matrix according to the merge in step 2.
  4. Repeat step 2 and 3 until all points belongs to one cluster.

Another method that uses hierarchical clustering is divisive analysis clustering (DIANA). It is the invert of agglomerative nesting clustering (AGNES) that we said above it is a bottom-up clustering. DIANA is top-down approach that starts with all data points in one cluster, the root. Then, it splits the root into a set of child clusters. Each child cluster is recursively divided further. Finally, it stops when only singleton clusters of individual data points remain, i.e., each cluster with only a single point. The difference between AGNES and DIANA is that DIANA is more complex and more efficient from AGNES. Also, divisive clustering is more accurate because it takes into consideration the global distribution of data when making top-level partitioning decisions, while agglomerative clustering makes decisions by considering the local patterns or neighbor points without initially considering the global distribution of data.

References and useful resources:

- CSBP 320: Data Mining, UAE University
- Textbook: Hongbo Du.*Data Mining Techniques and Applications: an introduction*, 1st Edition
- https://blog.bismart.com/en/classification-vs.-clustering-a-practical-explanation
- https://link.springer.com/content/pdf/10.1007/s40745-015-0040-1.pdf
- https://towardsdatascience.com/17-clustering-algorithms-used-in-data-science-mining-49dbfa5bf69a
- https://www.kdnuggets.com/2018/06/5-clustering-algorithms-data-scientists-need-know.html#.W2fx7DTtf00.reddit
- https://www.upgrad.com/blog/cluster-analysis-data-mining/
- https://www.geeksforgeeks.org/bisecting-k-means-algorithm-introduction/
- https://www.geeksforgeeks.org/ml-hierarchical-clustering-agglomerative-and-divisive-clustering/