Unit

Clustering

→ Intra Cluster → minimized (btw objects)
→ Inter cluster → maximized (btw clusters)

Cluster Analyze:-

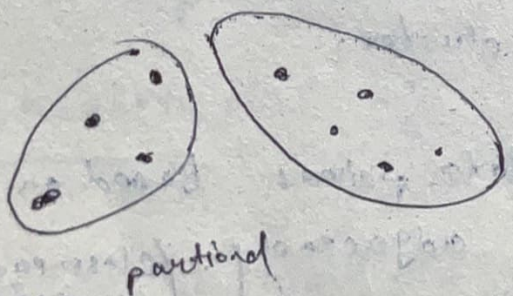
Summarization: reduce the size of large dataset.

Requirements of Clustering —

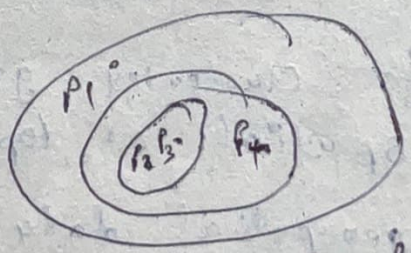
→ Scalability
→ Ability to deal with diff type of attributes
→ Able to deal with noise and outliers
→ High dimensionality
→ Incorporation of user-specified constraints
→ Insensitive to order of input record.

Type of Clustering —

→ Partional clustering → A division of data objects into non-overlaping subsets (clusters)
→ Hierarchical clustering → A set of nested clusters organized as a hierchical tree



partional



Hierachical

- **Exclusive v/s non-exclusive**
  - → In non-exclusive clustering point may belong to multiple clusters.
  - → Fuzzy clustering (non exclusive) → weight between 0 & 1, sum = 1
- **Partial v/s complete** → could be 'border' points

Types Clusters

- → well separated → K-means
- → prototype based
- → Contiguity-based
- → Density based
- → Described by an objective function

Prototype:- cluster refers to a set of representative points that summarize a cluster of data points in clustering algorithms.

- → in K-means prototype are the centroids of each cluster.
- → in K-medoid prototypes are actual data points that best represent the cluster.

Contiguity:- clustering groups data points based on their spatial or topological adjacency. (closeness or connection)

→ It is used in geospatial data to identify regions that

- → clusters are formed based on proximity with adjacent data points sharing similar characteristics or closeness.

Density Based :—

ip a dense region of points which is separated by low-density regions, from other regions of high density.

→ density based clustering groups data points that are closely packed together in high-density regions and separate them from sparse regions.

→ here we use K-means, DBSCAN, OPTICS

Clusters defined by an objective functions

finds clusters that minimize or maximize an objective function.

Enumerate all possible ways of dividing the points into clusters and evaluate the goodness of each potential set of clusters by using the given objective function. (NP Hard)

Can have global or local objectives.

→ Hierarchical clustering algorithms typically have local objectives

→ partitional algo typically have global objective

Characteristics of the Input Data are Imp :—

Good Clustering :-

→ high intra-class similarity
→ low inter-class similarity

Quality → similarity
implementation

Partion clustering :-

$$E = \sum_{k=1}^{K} \sum_{x \in C_k} \text{Distance}(x, m_k)$$

total error

total no. of clusters

$$\text{Distance} = \sum_{d=1}^{D} (x_d - m_{kd})^2$$

Solve K means examples

Strengths

→ Simple : easy to understand
→ $O(tkn)$
t iteration, k cluster num, n

Weakness

Algo is only applicable if the mean is defined
→ user need to specify k
→ outlies : sensitive

preventions :-

→ remove outliers first.
→ take some subsets of point then the chance of selection of outlier is very low.

Weakness :- algo is not suitable for discovering clusters that are not hyper-ellipsoido.

K medoids :-

PAM algo represent some data point.

Solve K mediods examples →

properties :-

$$O(k(n-k)^2)$$

for large val of n & K computation become very costly.

Advantage :- K mediod is more robust than K-mean in the presence of noise and outliers.

Disadvantage :-
→ K mediod is more costly that K mean
→ K means, K mediod requires their user to specify K,
→ it does not work well for large dataset,

PAM (Partitioning Around Medoids) :-

finds mediod that are centraly located in clusters.
→ The goal of the algo is to minimize the avg. dissimilarity of obj. to their closest selected obj.

→ PAM works effectively for small datasets, but does not scale well for large data set.

K-mediods properties :-

$$O(k(n-k)^2)$$

Advantage :- K mediods method is more better than k-means in the presence of noise and outliers

Disadvantage :-
→ k-mediods is more costly than k-mean.
→ k-mediods requires the user to specify k
→ does not scale well for large datasets

CLARA (Clustering Large Application) :- uses a sampling based method to deal with large datasets

→ Choosen mediods will likely be similar to what would have been chosen from the whole dataset.
→ draw multiple samples of the dataset
→ Apply PAM (k-mediod) to each sample
→ return the best clustering

Properties :-

$$O(*K(S-K)^2 + K(n-K))$$

* → max iteration required for PAM
S → same size
K → no. of Clusters
n → no. of obj.

Problem :-

→ The best K medoids may not be selected during the sampling process ; in this case, CLARA will never find the best clustering.

→ If the sampling is biased we cannot have a good clustering.

CLARANS ("Randomized" CLARA)

(A Clustering large application based on randomized search)

→ The clustering process can be presented as searching a graph where every node is a potential solution, that is a set of K mediods

→ Two nodes are neighbours if their sets differ by only one medoid

→ Each node can be assigned a cost that is defined to be the total dissimilarity between every object and the medoid of its cluster.

→ The problem corresponds to search for a minimum on the graph

→ At each step, all neighbours of current node are searched; the neighbours ~~of current node node as~~ which corresponds to the deepest descent in cost is chosen as the next solution.

Graph Abstraction —

→ Every node has $K(n-K)$ adjacent nodes

→ At each step, CLARANS draws sample of neighbours to examine.

→ Note that CLARA draws a sample of nodes at the beginning of search;

therefore, CLARANS had the benefit of not confining the search to a restricted area.

→ If the local optimum is found, CLARANS starts with a new randomly selected node in search for a new local optimum. The number of local optimums to search for is a parameter.

→ It is more efficient and scalable than both PAM and CLARA; returns higher quality clusters.

Disadvantage :- for large value of n and k, examining $K(n-K)$ neighbours is time consuming.

~~Exam~~

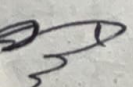
Hierarchical Clustering :-

⇒ Agglomerative vs Divisive :-

⇒ two sequential clustering strategies for constructing a tree of clusters

→ Agglomerative : a bottom-up strategy
→ Initially each data object is in its own (atomic) cluster
→ Then merge these atomic clusters into larger and larger clusters.

Divisive : a top-down strategy :-
→ Initially all objects are in one single cluster
→ Then the cluster is sub divided into smaller and smaller clusters.



single Link :- ⟶ min { }
complete link ⟶ max { }
Avg link ⟶ Avg = $\frac{\{\ldots + \ldots + \ldots\}}{n}$

Single-Link

→ Can find irregular-shaped clusters
→ Sensitive to outliers, suffers the so-called chaining effects
→ In order to merge two groups, only need one pair of points to be close, irrespective of all others.

Average-link and Centroid distance

→ Robust to outliers
→ Tend to break large clusters

AGNES (Agglomerative Nesting) :→ Single link
→ least dissimilarity

UPGMA (un-weighted pair Group Method Average) :-
→ Average link approach

DIANA :- (Divisive Analysis)

BIRCH :-

mahesh huddar solved example youtube

CURE (Clustering Using Representative) :-

Drawbacks of Traditional Clustering Algorithms :-

→ Centroid-based approach (using ~~d~~mean) considers only one point as representative of a cluster - the cluster centroid.

→ All-points approach (based on dmin) makes the clustering algorithm extremely sensitive to outliers.

→ Both of them can't work well for non-spherical or arbitrary shaped clusters.

CURE : ~~Approach~~ Approach

→ CURE is positioned between centroid based and all point extremes.

→ ~~A constant number of well scattered points is~~

~~DBSCAN~~ Density-Based Clustering :-

Basic idea :-

→ Clusters are dense regions in the data space, separated by regions of lower object density

→ A cluster is defined as a maximal set of density connected points.

Each cluster has a considerable higher density of points than outside of the cluster.

DBSCAN :-

Advantages :-

→ Clusters can have ~~arbitrary~~ arbitrary shape and size.

→ No. of clusters is determined automatically.

→ Can separate clusters from surrounding noise.

Disadvantages:-

→ Input parameters may be difficult to determine.

→ In some situations very sensitive to input parameter setting.

Association Rule Mining

Study of "what goes with what"

→ "Customers who bought X also bought Y"

→ what symptoms go with what diagnosis

Transaction-based or event-based

Also called "Market Basket Analysis" and "affinity analysis".

Support:- the support s of an itemset X is the percentage of transactions in the transaction database T that contain X.

→ support of the rule $X \Rightarrow Y$ in the transaction database T is the support of the items set $X \Rightarrow Y$ in T

→ probability that transaction contains $X \cup Y$.

Confidence:- of the rule $X \Rightarrow Y$ in the transaction database T is the ratio of

the no. of transactions in T that contain $X \Rightarrow Y$ to the no. of transactions that contain X in T.

Conditional probability that a transaction having X also contains Y.

Support :- $\frac{\text{kitni transaction (me ek sath dono aa rhe hain)}}{\text{total transaction}} \times 100\%$.

Confidence :- $\frac{\text{(dono (left + right) kitni bar aa rhe hai)}}{\text{(left side wale kacount)}} \times 100\%$.

The Apriori Algorithm :-

Maximum Frequent Set :-

maximum length wale combination jiska support, minimum support (threshhold) se jyada(≥) ho to vo maximum frequent set hoga.

~~Example~~ :-

Border Frequent Set :-

maximum length wale combination jiska support minimum support (threshold) se kam hoga lekin uske sare proper subset ka support min support se jyada hoga to vo border frq. set hoga.

Ex,

T_id	item
T1	{A, B, C}
T2	{A, B, D}
T3	{A, C, D}
T4	{B, C, D}
T5	{A, B, C}

Given → min support = 2

	Support	
{A}	4	✓
{B}	4	✓
{C}	4	✓
{D}	3	✓
{A, B}	3	✓
{B, C}	3	✓
{A, C}	3	✓
{A, D}	2	✓
{B, D}	2	✓
{C, D}	2	✓
{A, B, C}	2	✓
{A, B, D}	1	×
{A, C, D}	1	×
{B, C, D}	1	×
{A, B, C, D}	0	×

Maximum Frequent Set :—

{A, B, C} because its all subsets are frequent itemset (support ≥ 2).

because its length is maximum among all frequent itemset.

Border Frequent Set :—

Maximum length of itemset that is not frequent (support < 2) and all subset are frequent set. so here we check 4 combinations

{A, B, C, D}, {B, C, D}, {A, C, D}, {A, B, D}.

So our border sets are {A, B, D}, {A, C, D}, {B, C, D}.

Challenges of Frequent Pattern Mining

Challenges :—

- → Multiple scans of transaction database
- → Huge no. of candidates
- → Tedious workload of support counting for candidates.

Improving Apriori :

→ Reduce passes of transaction database scans
→ Shrink no. of candidates
→ Facilitate support counting of candidates

Partion Algorithm :-

The partion Algo. is based on the observation that the frequent sets are normally very few in no. compared to the set of all itemsets.

As a result, if we partition the set of transactions to smaller segments such that each segment can be accomodated in the main memory, then we can compute the set of frequent sets of each of these partition.

Since each partition can fit in the main memory there will be no additional disk I/O for each partition after loading the partion into the main memory.

FP-tree Growth Algorithm :-

why FP-tree and not Apriori ?

→ Lots of frequent patterns :
- Big set of items
- low minimum support threshold

→ Long patterns

FP-Tree Construction :-

→ FP tree is constructed using 2 passes over the dataset

→ **Pass 1 :**

→ Scan data and find support for each item.
→ Discard infrequent items.
→ Sort frequent items in decreasing order based on their support

→ Use this order when building the FP-tree, so common prefixes can be shared.

→ In FP-tree nodes correspond to items and have a counter.

Pass 2 :

1. FP Growth reads 1 transaction at a time and maps it to a path.
2. Fixed order is used, so paths can overlap when transactions share items (when they have the same prefix).
   - In this case, counters are incremented.
3. Pointers are maintained between nodes containing the same item, creating singly linked lists (dotted lines)
   - The more paths that overlap, the higher the compression. FP-tree may fit in memory.
4. Frequent itemsets extracted from the FP-tree

Best Case :- Same set of items

Worst Case :- every transaction has unique set of items

Measure for Association Rules :-

- Confidence $(X \rightarrow Y)$ should be sufficiently high
  To ensure that people who buy X will more likly buy Y than not buy Y
- Confidence $(X \rightarrow Y)$ > support (Y)

  Otherwise rule will be misleading bcoz having item X actually reduces the chance of having item Y in the same transaction.

Statistical Relationship btw X & Y :-

Confidence $(X \rightarrow Y)$ = support (Y)

is equivalent to :

- $P(Y/X) = P(Y)$
- $P(X, Y) = P(X) \times P(Y)$ { X and Y are independent

If $P(X, Y) > P(X) \times P(Y)$: X & Y are positively correlated

If $P(X, Y) < P(X) \times P(Y)$: X & Y are negatively correlated

---

$$\text{Lift} = \frac{P(Y/X)}{P(Y)}$$

$$\text{Interest} = \frac{P(X, Y)}{P(X)\,P(Y)}$$

$$PS = P(X, Y) - P(X)\,P(Y)$$

{ PS: pairwise similarity
=0 no association
>0 +vely
<0 -vely

ϕ – coefficient $= \dfrac{P(X,Y) - P(X)P(Y)}{\sqrt{P(X)[1-P(X)]P(Y)[1-P(Y)]}}$

Ex. :-

Association Rule:

Tea ⟶ Coffee

	Coffee	$\overline{\text{Coffee}}$	
Tea	150	50	200
$\overline{\text{Tea}}$	650	150	800
	800	200	1000

Confidence

Interest $= \dfrac{P(X,Y)}{P(X)\cdot P(Y)}$ $\begin{cases} X = \text{Tea} \\ Y = \text{Coffee} \end{cases}$

$= \dfrac{150/1000}{\frac{200}{1000} \times \frac{800}{1000}}$

$= \dfrac{.15}{.2 \times .8} = .9375 < 1$ { therefore negatively associated

Lift $= \dfrac{P(Y/X)}{P(Y)} = \dfrac{\frac{P(Y \cap X)}{P(X)}}{\frac{P(Y)}{1}} = \dfrac{P(Y \cap X)}{P(X)\cdot P(Y)}$

$PS = P(X,Y) - P(X)\cdot P(Y)$

$= .15 - (.2 \times .8)$

$= .15 - .16 = -0.01 < 0$ { negatively associated

Text Mining

Data Mining

- Identify data sets
- Select features
- Prepare data
- Analyze distribution

Text Mining

- Identify documents
- Extract features
- Select features by algo
- Prepare data
- Analyze distribution

Text Mining Applications :-

→ email, newspaper, patent database, company's reports.

Why dealing with Text is Tough? :-

→ Abstract concepts are difficult to represent
→ "Countless" combinations of subtle abstract relationships among concepts
→ many ways to represent similar concepts
Eg. space ship, flying saucer, UFO
→ Concept are difficult to visualize
→ High dimensionality
→ Tens or hundreds of thousands of features

Why dealing with Text is Easy :-

→ High redundant data
- most of the methods count on this property
→ Just about any simple algo can get "good" results for simple tasks:
- pull out "important" pharases
- Find "meaningfully" related words
- Create some sort of summary from documents

Word properties :-

Homonomy :- same form, but diff. meaning.
Polysemy :- same form, related meaning.
Synonymy :- different form, same meaning.
Hyponymy :- one word denotes a subclass of an another.

power distribution :-

→ small no. of very frequent word.
→ big no. of low frequent word.

Stop-words :- are words that from non-linguistic view do not carry information.

ex- A, About, Above, Along, Already ——

Stemming :-

different forms of the same word are usually problematic for text data analysis, because they have different spelling and similar meaning (eg. learns, learned, learning ——)

Phrase level :-

The main effect of using phrases is to more precisely identify sence.

Taxonomies / Thesaurus level :-

Thesaurus has a main function to connect different surface word forms with the same meaning into one sense (synonyms).

Ex. EuroWordNet.

WordNet - database of lexical relations

WordNet is the most well developed and widely used lexical database for english

→ It consist from 4 databases (nouns, verbs, adjectives, and adverbs)

Category	Unique form	Number of senses
Noun	94474	116317
Verb	10319	22066
Adjective	20170	29381
Adverb	4546	5677

WordNet relations :- Each wordnet entry is connected with other entries in the graph through relations.

Vector-Space Model Level :-

The most common way to deal with documents is first to transform them into sparse numeric vectors and then deal with them with linear algebra operations.

this way of forgetting about the structure doesn't harm efficieng of solving many relevant problem.

Typical tasks on vector-space-model are classification, clustering, visualization etc.

Word weighting

$$tfidf(w) = tf \cdot \log\left(\frac{N}{df(w)}\right)$$

The word is more important if it appears several times in a target document.

The word is more important if it appears in less documents

Full-parsing :-

parsing provides maximum structural information per sentence.

- On the input we get a sentence, on the output we generate a parse tree.