4 UNIT

Frequent Itemsets and Clustering

CONTENTS

PART-1

Frequent Itemsets and Clustering : Mining Frequent Itemsets, Markets Based Modelling, Apriori Algorithm.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.1. Write short notes on frequent patterns in data mining.

Answer

1. Frequent patterns are patterns (such as itemsets, subsequences, or substructures) that appear frequently in a dataset.
2. A substructure can refer to different structural forms, such as sub-graphs, sub-trees, or sub-lattices, which may be combined with itemsets or subsequences.
3. If a substructure occurs frequently, it is called a (frequent) structured pattern.
4. Finding frequent patterns plays an essential role in mining associations, correlations, and many other interesting relationships among data.
5. It helps in data classification, clustering, and other data mining tasks.
6. Frequent pattern mining searches for recurring relationships in a given dataset.
7. For example, a set of items, such as milk and bread that appear frequently together in a grocery transaction dataset is a frequent itemset.
8. A subsequence, such as buying first a PC, then a digital camera, and then a memory card, if it occurs frequently in a shopping history database, is a (frequent) sequential pattern.

Que 4.2. Explain frequent itemset mining.

Answer

1. Frequent itemset mining leads to the discovery of associations and correlations among items in large transactional or relational datasets.
2. With massive amounts of data continuously being collected and stored, many industries are becoming interested in mining such patterns from their databases.
3. The discovery of interesting correlation relationships among huge amounts of business transaction records can help in many business

decision-making processes such as catalogue design, cross-marketing, and customer shopping behaviour analysis.

4. A typical example of frequent itemset mining is market basket analysis.
5. This process analyzes customer buying habits by finding associations between the different items that customers place in their "shopping baskets".
6. The discovery of these associations can help retailers to develop marketing strategies by gaining insight into which items are frequently purchased together by customers.
7. For instance, if customers are buying some product, how likely are they to also other products at the same time. This information can lead to increased sales by helping retailers do selective marketing.

Que 4.3. Write short notes on market based modelling.

Answer

1. The market-basket model of data is used to describe a common form of many to many relationship between two kinds of objects.
2. On the one hand, we have items, and on the other we have baskets, sometimes called "transactions."
3. Each basket consists of a set of items (an itemset), and usually we assume that the number of items in a basket is small or much smaller than the total number of items.
4. The number of baskets is usually assumed to be very large, bigger than what can fit in main memory.
5. The data is assumed to be represented in a file consisting of a sequence of baskets.

Que 4.4. Write short notes on algorithm for finding frequent itemsets.

Answer

1. The Apriori algorithm takes a bottom-up iterative approach to find the frequent itemsets by first determining all the possible items and then identifying which among them are frequent.
2. Let variable C_k be the set of candidate k-itemsets and variable L_k be the set of k-itemsets that satisfy the minimum support.
3. Given a transaction database D, a minimum support threshold δ, and an optional parameter N indicating the maximum length an itemset could reach, Apriori iteratively computes frequent itemsets L_{k-1} based on L_k.

Apriori algorithm :

Apriori (D, δ, N) :

1. $k \leftarrow 1$
2. $L_k \leftarrow$ {1-itemsets that satisfy minimum support δ}
3. while $L_k \neq \emptyset$
4. if $\exists N \vee (\exists N \wedge k < N)$
5. $C_{k+1} \leftarrow$ candidate itemsets generated from L_k
6. for each transaction t in database D do
7. increment the counts of C_{k+1} contained in t
8. L_{k+1} candidates in C_{k+1} that satisfy minimum support δ
9. $k \leftarrow k+1$
10. return $\bigcup_k L_k$

4. At each iteration, the algorithm checks whether the support criterion can be met; if it can, the algorithm grows the item set, repeating the process until it runs out of support or until the item sets reach a predefined length.
5. The first step of the Apriori algorithm is to identify the frequent item sets by starting with each item in the transactions that meets the predefined minimum support threshold δ.
6. These itemsets are 1-itemsets denoted as L_1, as each 1-itemset contains only one item. Next, the algorithm grows the item set s by joining L_1 onto itself to form new, grown 2-itemsets denoted as L_2 and determines the support of each 2-itemset in L_2. Those itemsets that do not meet the minimum support threshold δ are pruned away.
7. The growing and pruning process is repeated until no itemsets meet the minimum support threshold.
8. A threshold N can be set up to specify the maximum number of items the item set can reach or the maximum number of iterations of the algorithm. Once completed, output of the Apriori algorithm is the collection of all the frequent k-itemsets.

Que 4.5. How is the Apriori property used in the algorithm ?

Answer

A two-step process is followed, consisting of join and prune actions.

1. **The join step :**
   a. To find L_k, a set of candidate k-itemsets is generated by joining L_{k-1} with itself. This set of candidates is denoted C_k.
   b. Let l_1 and l_2 be itemsets in L_{k-1}. The notation $l_i[j]$ refers to the j^{th} item in l_i (e.g., $l_1[k-2]$ refers to the second to the last item in l_1).
   c. For efficient implementation, Apriori assumes that items within a transaction or itemset are sorted in lexicographic order. For the

$(k-1)$-itemset, l_i, this means that the items are sorted such that $l_i[1] < l_i[2] < \cdots < l_i[k-1]$.

d. The join, $L_{k-1} \bowtie L_{k-1}$, is performed, where members of L_{k-1} are joinable if their first $(k-2)$-items are in common. That is, members l_1 and l_2 of L_{k-1} are joined if $(l_1[1] = l_2[1]) \wedge (l_1[2] = l_2[2]) \wedge \cdots \wedge (l_1[k-2] = l_2[k-2]) \wedge (l_1[k-1] < l_2[k-1])$.

e. The condition $l_1[k-1] < l_2[k-1]$ simply ensures that no duplicates are generated. The resulting itemset formed by joining l_1 and l_2 is $\{l_1[1], l_1[2], \ldots, l_1[k-2], l_1[k-1], l_2[k-1]\}$. }

2. The prune step :

a. C_k is a superset of L_k, that is, its members may or may not be frequent, but all of the frequent k-itemsets are included in C_k.

b. A database scan to determine the count of each candidate in C_k would result in the determination of L_k (*i.e.*, all candidates having a count less than the minimum support count are frequent by definition, and therefore belong to L_k).

c. C_k, however, can be huge, and so this could involve heavy computation. To reduce the size of C_k, the Apriori property is used.

d. According to Apriori property, any $(k-1)$-itemset that is not frequent cannot be a subset of a frequent k-itemsets. Hence if any $(k-1)$-subset of a candidate k-itemset is not in L_{k-1}, then the candidate cannot be frequent and can be removed from C_k.

Que 4.6. Write short note on generating association rules from frequent itemsets.

Answer

1. Once the frequent itemsets from transactions in a database D have been found, it is straightforward to generate strong association rules from them (where strong association rules satisfy both minimum support and minimum confidence).
2. This can be done using equation (4.6.1) for confidence, which is shown here for completeness :

$$\text{confidence}(A \Rightarrow B) = P(B|A) = \frac{\text{support\_count}(A \cup B)}{\text{support\_count}(A)} \quad \ldots(4.6.1)$$

3. The conditional probability is expressed in terms of itemset support count, where support_count $(A \cup B)$ is the number of transactions containing the itemsets $A \cup B$, and support_count (A) is the number of transactions containing the itemset A.
4. Based on equation (4.6.1), association rules can be generated as follows :

a. For each frequent itemset l, generate all non-empty subsets of l.

b. For every non-empty subset s of l, output the rule $s \Rightarrow (l - s)$ if (support_count(l)/ support_count(s)) $\geq$ min_conf, where min_conf is the minimum confidence threshold.

5. Because the rules are generated from frequent itemsets, each one automatically satisfies the minimum support.

6. Frequent itemsets can be stored ahead of time in hash tables along with their counts so that they can be accessed quickly.

Que 4.7. **How can we improve the efficiency of Apriori-based mining ?**

Answer

Many variations of the Apriori algorithm have been proposed that focus on improving the efficiency of the original algorithm. Several of these variations are as follows :

1. **Hash-based technique (hashing itemsets into corresponding buckets) :**

   a. A hash-based technique can be used to reduce the size of the candidate k-itemsets, C_k, for $k > 1$.

   b. For example, when scanning each transaction in the database to generate the frequent 1-itemsets, L_1, we can generate all the 2-itemsets for each transaction, hash (*i.e.*, map) them into the different buckets of a hash table structure, and increase the corresponding bucket counts (Fig. 4.7.1).

Table 4.7.1 : Transactional data for an all electronics branch

TID	List of item_IDs
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3

Create hash table H_2 using hash function $h(x, y) = ((\text{order of } x) \times 10 + (\text{order of } y)) \bmod 7$

H_2

bucket address	0	1	2	3	4	5	6
bucket count	2	2	4	2	2	4	4
bucket countents	(I1, I4) (I3, I5)	(I1, I5) (I1, I5)	(I2, I3) (I2, I3) (I2, I3) (I2, I3)	(I2, I4) (I2, I4)	(I2, I5) (I2, I5)	(I1, I2) (I1, I2) (I1, I2) (I1, I2)	(I1, I3) (I1, I3) (I1, I3) (I1, I3)

Fig. 4.7.1. Hash table, H_2, for candidate 2-itemsets. This has table was generated by scanning Table 4.7.1 transactions while determining L_1. If the minimum support count is, say, 3, then the itemsets in buckets 0, 1, 3, and 4 cannot be frequent and so they should not be included in C_2.

c. A 2-itemset with a corresponding bucket count in the hash table that is below the support threshold cannot be frequent and thus should be removed from the candidate set.

d. Such a hash-based technique may substantially reduce the number of candidate k-itemsets examined (especially when $k = 2$).

2. Transaction reduction (reducing the number of transactions scanned in future iterations) :

a. A transaction that does not contain any frequent k-itemsets cannot contain any frequent $(k + 1)$-itemsets.

b. Therefore, such a transaction can be marked or removed from further consideration because subsequent database scans for j-itemsets, where $j > k$, will not need to consider such a transaction.

3. Partitioning (partitioning the data to find candidate itemsets) :

a. A partitioning technique can be used that requires just two database scans to mine the frequent itemsets as shown in Fig. 4.7.2.

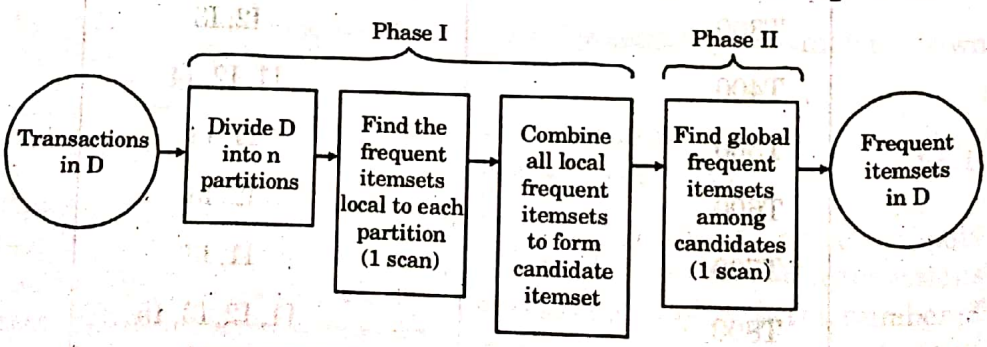

Fig. 4.7.2. Mining by partitioning the data.

b. **It consists of two phases :**

Phase I :

i. In phase I, the algorithm divides the transactions of D into n non-overlapping partitions. If the minimum relative support threshold for transactions in D is min_sup, then

Minimum support count for a partition = min_sup × the number of transactions in that partition

ii. For each partition, all the local frequent itemsets are found.

iii. A local frequent itemset may or may not be frequent with respect to the entire database, D. However, any itemset that is potentially frequent with respect to D must occur as a frequent itemset in at least one of the partitions.

iv. Therefore, all local frequent itemsets are candidate itemsets with respect to D. The collection of frequent itemsets from all partitions forms the global candidate itemsets with respect to D.

Phase II :

i. In phase 2, a second scan of D is conducted in which the actual support of each candidate is assessed to determine the global frequent itemsets.

ii. Partition size and the number of partitions are set so that each partition can fit into main memory and therefore be read only once in each phase.

4. **Sampling (mining on a subset of the given data) :**

a. The basic idea of the sampling approach is to pick a random sample S of the given data D, and then search for frequent itemsets in S instead of D.

b. In this way, we trade off some degree of accuracy against efficiency.

c. The S sample size is such that the search for frequent itemsets in S can be done in main memory, and so only one scan of the transactions in S is required overall.

d. In this technique, it is possible that we will miss some of the global frequent itemsets.

5. **Dynamic itemset counting (adding candidate itemsets at different points during a scan) :**

a. A dynamic itemset counting technique was proposed in which the database is partitioned into blocks marked by start points.

b. In this variation, new candidate itemsets can be added at any start point, which determines new candidate itemsets only immediately before each complete database scan.

Que 4.8. What are the applications of frequent itemset analysis ?

Answer

Applications of frequent itemset analysis :

a. Related concepts :

1. Let items be words, and let baskets be documents (e.g., Web pages, blogs, tweets).
2. A basket/document contains those items/words that are present in the document.
3. If we look for sets of words that appear together in many documents, the sets will be dominated by the most common words (stop words).
4. If the document contain many the stop words such as "and" and "a" then it will consider as more frequent itemsets.
5. However, if we ignore all the most common words, then we would hope to find among the frequent pairs some pairs of words that represent a joint concept.

b. Plagiarism :

1. Let the items be documents and the baskets be sentences.
2. An item is in a basket if the sentence is in the document.
3. This arrangement appears backwards, and we should remember that the relationship between items and baskets is an arbitrary many-many relationship.
4. In this application, we look for pairs of items that appear together in several baskets.
5. If we find such a pair, then we have two documents that share several sentences in common.

c. Biomarkers :

1. Let the items be of two types such as genes or blood proteins, and diseases.
2. Each basket is the set of data about a patient: their genome and blood-chemistry analysis, as well as their medical history of disease.
3. A frequent itemset that consists of one disease and one or more biomarkers suggest a test for the disease.

PART-2

Handling Large Data Sets in Main Memory, Limited Pass Algorithm, Counting Frequent Itemsets in a Stream.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.9. What are the different methods for storing itemset count in main memory ?

Answer

Different method for storing itemset count in main memory :

1. **The triangular-matrix method :**
   a. Even after coding items as integers, we still have the problem that we must count a pair $\{i, j\}$ in only one place.
   b. For example, we could order the pair so that $i < j$, and only use the entry $a[i, j]$ in a two-dimensional array a. That strategy would make half the array useless.
   c. A more space-efficient way is to use a one-dimensional triangular array.
   d. We store in $a[k]$ the count for the pair $\{i, j\}$, with $1 \leq i < j \leq n$, where
   $$k = (i - 1)(n - i/2) + j - i .$$
   e. The result of this layout is that the pairs are stored in lexicographic order, that is first $\{1, 2\}$, $\{1, 3\}, \ldots, \{1, n\}$, then $\{2, 3\}$, $\{2, 4\}, \ldots, \{2, n\}$, and so on, down to $\{n-2, n-1\}$, $\{n-2, n\}$, and finally $\{n-1, n\}$.
2. **The triples method :**
   a. This is more appropriate approach to store counts that depend on the fraction of the possible pairs of items that actually appear in some basket.
   b. We can store counts as triples $[i, j, c]$, meaning that the count of pair $\{i, j\}$, with $i < j$, is c. A data structure, such as a hash table with i and j as the search key, is used so we can tell if there is a triple for a given i and j and, if so, to find it quickly.
   c. We call this approach the triples method of storing counts.
   d. The triples method does not require us to store anything if the count for a pair is 0.
   e. On the other hand, the triples method requires us to store three integers, rather than one, for every pair that does appear in some basket.

Que 4.10. Explain PCY algorithm for handling large dataset in main memory.

Answer

1. In first pass of Apriori algorithm, there may be much unused space in main memory.
2. The PCY Algorithm uses the unused space for an array of integers that generalizes the idea of a Bloom filter. The idea is shown schematically in Fig. 4.10.1.

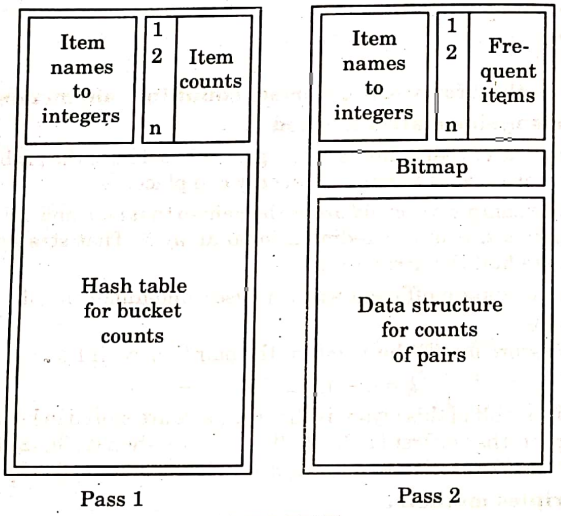

Fig. 4.10.1.

3. Array is considered as a hash table, whose buckets hold integers rather than sets of keys or bits. Pairs of items are hashed to buckets of this hash table. As we examine a basket during the first pass, we not only add 1 to the count for each item in the basket, but we generate all the pairs, using a double loop.
4. We hash each pair, and we add 1 to the bucket into which that pair hashes.
5. At the end of the first pass, each bucket has a count, which is the sum of the counts of all the pairs that hash to that bucket.
6. If the count of a bucket is at least as great as the support threshold s, it is called a frequent bucket. We can say nothing about the pairs that hash to a frequent bucket; they could all be frequent pairs from the information available to us.
7. But if the count of the bucket is less than s (an infrequent bucket), we know no pair that hashes to this bucket can be frequent, even if the pair consists of two frequent items.

8. We can define the set of candidate pairs C_2 to be those pairs $\{i, j\}$ such that:
   a. i and j are frequent items.
   b. $\{i, j\}$ hashes to a frequent bucket.

Que 4.11. Explain simple and randomized algorithm to find most frequent itemsets using at most two passes.

Answer

Simple and randomized algorithm :

1. In simple and randorimized algorithm, we pick a random subset of the baskets and pretend it is the entire dataset instead of using the entire file of baskets.
2. We must adjust the support threshold to reflect the smaller number of baskets.
3. For instance, if the support threshold for the full dataset is s, and we choose a sample of 1% of the baskets, then we should examine the sample for itemsets that appear in at least s/100 of the baskets.
4. The best way to pick the sample is to read the entire dataset, and for each basket, select that basket for the sample with some fixed probability p.
5. Suppose there are m baskets in the entire file. At the end, we shall have a sample whose size is very close to pm baskets.
6. However, if the baskets appear in random order in the file already, then we do not even have to read the entire file.
7. We can select the first pm baskets for our sample. Or, if the file is part of a distributed file system, we can pick some chunks at random to serve as the sample.
8. Having selected our sample of the baskets, we use part of main memory to store these baskets.
9. Remaining main memory is used to execute one of the algorithms such as A-Priori or PCY. However, the algorithm must run passes over the main-memory sample for each itemset size, until we find a size with no frequent items.

Que 4.12. Explain SON algorithm to find all or most frequent itemsets using at most two passes.

Answer

SON Algorithm :

1. The idea is to divide the input file into chunks.
2. Treat each chunk as a sample, and run the simple and randomized

algorithm on that chunk.

3. We use ps as the threshold, if each chunk is fraction p of the whole file, and s is the support threshold.
4. Store on disk all the frequent itemsets found for each chunk.
5. Once all the chunks have been processed in that way, take the union of all the itemsets that have been found frequent for one or more chunks. These are the candidate itemsets.
6. If an itemset is not frequent in any chunk, then its support is less than ps in each chunk. Since the number of chunks is $1/p$, we conclude that the total support for that itemset is less than $(1/p)ps = s$.
7. Thus, every itemset that is frequent in the whole is frequent in at least one chunk, and we can be sure that all the truly frequent itemsets are among the candidates; *i.e.*, there are no false negatives. We have made a total of one pass through the data as we read each chunk and processed it.
8. In a second pass, we count all the candidate itemsets and select those that have support at least s as the frequent itemsets.

Que 4.13. Explain SON algorithm usng MapReduce.

Answer

1. The SON algorithm work well in a parallel-computing environment.
2. Each of the chunks can be processed in parallel, and the frequent itemsets from each chunk combined to form the candidates.
3. We can distribute the candidates to many processors, have each processor count the support for each candidate in a subset of the baskets, and finally sum those supports to get the support for each candidate itemset in the whole dataset.
4. There is a natural way of expressing each of the two passes as a MapReduce operation.

MapReduce-MapReduce sequence :

First Map function :

a. Take the assigned subset of the baskets and find the itemsets frequent in the subset using the simple and randomized algorithm.

b. Lower the support threshold from s to ps if each Map task gets fraction p of the total input file.

c. The output is a set of key-value pairs $(F, 1)$, where F is a frequent itemset from the sample.

First Reduce Function :

a. Each Reduce task is assigned a set of keys, which are itemsets.

b. The value is ignored, and the Reduce task simply produces those keys (itemsets) that appear one or more times. Thus, the output of the first Reduce function is the candidate itemsets.

Second Map function :

a. The Map tasks for the second Map function take all the output from the first Reduce Function (the candidate itemsets) and a portion of the input data file.

b. Each Map task counts the number of occurrences of each of the candidate itemsets among the baskets in the portion of the dataset that it was assigned.

c. The output is a set of key-value pairs (C, v), where C is one of the candidate sets and v is the support for that itemset among the baskets that were input to this Map task.

Second Reduce function :

a. The Reduce tasks take the itemsets they are given as keys and sum the associated values.

b. The result is the total support for each of the itemsets that the Reduce task was assigned to handle.

c. Those itemsets whose sum of values is at least s are frequent in the whole dataset, so the Reduce task outputs these itemsets with their counts.

d. Itemsets that do not have total support at least s are not transmitted to the output of the Reduce task.

Que 4.14. Explain Toivonen's algorithm.

Answer

1. Toivonen's algorithm is a heuristic algorithm for finding frequent itemsets from a given set of data.
2. For many frequent itemset algorithms, main memory is considered a critical resource.
3. This is typically because itemset counting over large data sets results in very large data structures that quickly begin to strain the limits of main memory.
4. Toivonen's algorithm presents an interesting approach to discovering frequent itemsets in large data sets. The algorithm's deceptive simplicity allows us to discover all frequent itemsets through a sampling process.
5. **Negative border :** An itemset is in the negative border if it is not frequent in the sample, but all its immediate subsets are frequent in the sample.

6. Passes of Toivonen's algorithm :

Pass 1 :

a. Start with the random sample, but lower the threshold slightly for the subset.

b. Add to the itemsets that are frequent in the sample the negative border of these itemsets.

Pass 2 :

a. Count all candidate frequent itemsets from the first pass, and also count sets in their negative border.

b. If no itemset from the negative border turns out to be frequent, then we found all the frequent itemsets.

Que 4.15. Discuss sampling techniques to extract frequent itemsets from a stream.

Answer

1. We assume that stream elements are baskets of items.
2. The simplest approach to maintaining a current estimate of the frequent itemsets in a stream is to collect some number of baskets and store it as a file.
3. Run one of the frequent-itemset algorithms, meanwhile ignoring the stream elements that arrive, or storing them as another file to be analyzed later.
4. When the frequent-itemsets algorithm finishes, we have an estimate of the frequent itemsets in the stream.
5. We can use this collection of frequent itemsets for the application, but start running another iteration of the chosen frequent-itemset algorithm immediately. This algorithm can either :
   a. Use the file that was collected while the first iteration of the algorithm was running. At the same time, collect yet another file to be used at another iteration of the algorithm, when this current iteration finishes.
   b. Start collecting another file of baskets, and run the algorithm until an adequate number of baskets has been collected.

PART-3

Clustering Techniques: Hierarchical, k-means.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.16. Write short notes on clustering.

Answer

1. Clustering is the process of grouping a set of data objects into multiple groups or clusters so that objects within a cluster have high similarity, but are very dissimilar to objects in other clusters.
2. Dissimilarities and similarities are assessed based on the attribute values describing the objects and often involve distance measures.
3. The "quality" of a cluster may be represented by its diameter, the maximum distance between any two objects in the cluster.
4. Centroid distance is an alternative measure of cluster quality and is defined as the average distance of each cluster object from the cluster centroid.
5. Cluster analysis or simply clustering is the process of partitioning a set of data objects (or observations) into subsets.
6. The set of clusters resulting from a cluster analysis can be referred to as a clustering.
7. Clustering can lead to the discovery of previously unknown groups within the data.
8. Cluster analysis has been widely used in many applications such as business intelligence, image pattern recognition, Web search, biology, and security.

Que 4.17. What are the requirements for clustering in data mining ?

Answer

Following are requirements of clustering in data mining :

1. **Scalability :**
   a. Many clustering algorithms work well on small data sets containing fewer than several hundred data objects.
   b. Clustering on only a sample of a given large data set may lead to biased results. Therefore, highly scalable clustering algorithms are needed.

2. **Ability to deal with different types of attributes :** Many algorithms are designed to cluster numeric (interval-based) data. However, applications may require clustering other data types, such as binary, nominal (categorical), and ordinal data, or mixtures of these data types.
3. **Discovery of clusters with arbitrary shape :**
   a. Many clustering algorithms determine clusters based on Euclidean distance measures.
   b. Algorithms based on such distance measures tend to find spherical clusters with similar size and density. However, a cluster could be of any shape.
   c. It is important to develop algorithms that can detect clusters of arbitrary shape.
4. **Requirements for domain knowledge to determine input parameters :**
   a. Many clustering algorithms require users to provide domain knowledge in the form of input parameters such as the desired number of clusters.
   b. The clustering results may be sensitive to such parameters.
5. **Ability to deal with noisy data :**
   a. Most real-world data sets contain outliers and/or missing, unknown, or erroneous data.
   b. Clustering algorithms can be sensitive to noise and may produce poor-quality clusters. Therefore, we need clustering methods that are robust to noise.
6. **Capability of clustering high-dimensionality data :**
   a. A data set can contain numerous dimensions or attributes.
   b. Most clustering algorithms are good at handling low-dimensional data such as data sets involving only two or three dimensions.
   c. Finding clusters of data objects in a high dimensional space is challenging, especially considering that such data can be very sparse and highly skewed.
7. **Constraint-based clustering :**
   a. Real-world applications may need to perform clustering under various kinds of constraints.
   b. A challenging task is to find data groups with good clustering behaviour that satisfy specified constraints.

Que 4.18. **Write short notes on hierarchical method of clustering.**

Answer

1. A hierarchical method creates a hierarchical decomposition of the given set of data objects.
2. A hierarchical method can be classified as :
   a. **Agglomerative approach :**
      i. The agglomerative approach, also called the bottom-up approach, starts with each object forming a separate group.
      ii. It successively merges the objects or groups close to one another, until all the groups are merged into one (the topmost level of the hierarchy), or a termination condition holds.
   b. **Divisive approach :**
      i. The divisive approach, also called the top-down approach, starts with all the objects in the same cluster.
      ii. In each successive iteration, a cluster is split into smaller clusters, until eventually each object is in one cluster, or a termination condition holds.
3. Hierarchical clustering methods can be distance-based or density- and continuity-based.
4. Various extensions of hierarchical methods consider clustering in subspaces.
5. Hierarchical methods suffer from the fact that once a step (merge or split) is done, it can never be undone. This rigidity is useful in that it leads to smaller computation costs by not having to worry about a combinatorial number of different choices.

Que 4.19. Write short notes on partitioning method of clustering.

Answer

1. Given a set of n objects, a partitioning method constructs k partitions of the data, where each partition represents a cluster and $k \leq n$. That is, it divides the data into k groups such that each group must contain at least one object.
2. In other words, partitioning methods conduct one-level partitioning on data sets. The basic partitioning methods typically adopt exclusive cluster separation *i.e.*, each object must belong to exactly one group.
3. Most partitioning methods are distance-based. Given k, the number of partitions to construct, a partitioning method creates an initial partitioning.
4. It then uses an iterative relocation technique that attempts to improve the partitioning by moving objects from one group to another.

5. The general criterion of a good partitioning is that objects in the same cluster are "close" or related to each other, whereas objects in different clusters are "far apart" or very different. There are various kinds of other criteria for judging the quality of partitions.
6. Achieving global optimality in partitioning-based clustering is often computationally prohibitive, potentially requiring an exhaustive enumeration of all the possible partitions.

Que 4.20. Explain *k*-means (or centroid-based partitioning technique) clustering method.

Answer

1. Suppose a data set, D, contains n objects in Euclidean space. Partitioning methods distribute the objects in D into k clusters, $C_1,\ldots,C_k$, that is, $C_i \subset D$ and $C_i \cap C_j = \phi$ for $(1 \le i, j \le k)$.
2. An objective function is used to assess the partitioning quality so that objects within a cluster are similar to one another but dissimilar to objects in other clusters.
3. A centroid-based partitioning technique uses the centroid of a cluster, C_i, to represent that cluster. Conceptually, the centroid of a cluster is its center point. The centroid can be defined in various ways such as by the mean of the objects (or points) assigned to the cluster.
4. The difference between an object $p \in C_i$ and c_i, the representative of the cluster, is measured by dist(p, c_i), where dist(x, y) is the Euclidean distance between two points x and y.
5. The quality of cluster C_i can be measured by the within-cluster variation, which is the sum of squared error between all objects in C_i and the centroid c_i, defined as :

$$E = \sum_{i=1}^{k} \sum_{p \in c_i} \text{dist}\,(p, c_i)^2$$

Where, E is the sum of the squared error for all objects in the data set;

p is the point in space representing a given object

c_i is the centroid of cluster C_i (both p and c_i are multi-dimensional)

6. In other words, for each object in each cluster, the distance from the object to its cluster center is squared, and the distances are summed. This objective function tries to make the resulting k clusters as compact and as separate as possible.

Que 4.21. How does the *k*-means algorithm work? Write *k*-means algorithm for partitioning.

Answer

1. First, it randomly selects k of the objects in D, each of which initially represents a cluster mean or center.
2. For each of the remaining objects, an object is assigned to the cluster to which it is the most similar, based on the Euclidean distance between the object and the cluster mean.
3. The k-means algorithm then iteratively improves the within-cluster variation.
4. For each cluster, it computes the new mean using the objects assigned to the cluster in the previous iteration.
5. All the objects are then reassigned using the updated means as the new cluster centers.
6. The iterations continue until the assignment is stable, that is, the clusters formed in the current round are the same as those formed in the previous round.

Algorithm :

Input :

k: the number of clusters,

D: a data set containing n objects.

Output : A set of k clusters.

Method :

1. Arbitrarily choose k objects from D as the initial cluster centers;
2. repeat
3. (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;
4. update the cluster means, that is, calculate the mean value of the objects for each cluster;
5. until no change;

Que 4.22. What are the characteristics of different clustering techniques/methods ?

Answer

Characteristics of different clustering techniques/methods are :

Characteristics of partitioning methods :

1. Find mutually exclusive clusters of spherical shape
2. Distance-based
3. May use mean or medoid to represent cluster center
4. Effective for small to medium size data sets

Characteristics of hierarchical methods :

1. Clustering is a hierarchical decomposition (*i.e.*, multiple levels)
2. Cannot correct erroneous merges or splits
3. May incorporate other techniques like micro clustering or consider object "linkages"

Characteristics of density-based methods :

1. Can find arbitrarily shaped clusters
2. Clusters are dense regions of objects in space that are separated by low-density regions
3. May filter out outliers

Characteristics of grid-based methods :

1. Use a multi resolution grid data structure
2. Fast processing time

PART-4

Clustering High Dimensional Data, CLIQUE and ProCLUS, Frequent Pattern Based Clustering Methods.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.23. What are the approaches for high dimensional data clustering ?

Answer

Approaches for high dimensional data clustering are :

1. **Subspace clustering :**
   a. Subspace clustering subspace clustering algorithms localize the search for relevant dimensions allowing them to find clusters that exist in multiple, and possibly overlapping subspaces.
   b. This technique is an extension of feature selection that attempts to find clusters in different subspaces of the same dataset.
   c. Subspace clustering requires a search method and evaluation criteria.
   d. It limits the scope of the evaluation criteria so as to consider different subspaces for each different cluster.

2. **Projected clustering :**
   a. In high-dimensional spaces, even though a good partition cannot be defined on all the dimensions because of the sparsity of the data, some subset of the dimensions can always be obtained on which some subsets of data form high quality and significant clusters.
   b. Projected clustering methods are aimed to find clusters specific to a particular group of dimensions. Each cluster may refer to different subsets of dimensions.
   c. The output of a typical projected clustering algorithm, searching for k clusters in subspaces of dimension l, is twofold :
      i. A partition of data of $k + 1$ different clusters, where the first k clusters are well shaped, while the $(k + 1)^{th}$ cluster elements are outliers, which by definition do not cluster well.
      ii. A possibly different set of l dimensions for each of the first k clusters, such that the points in each of those clusters are well clustered in the subspaces defined by these vectors.

3. **Biclustering :**
   a. Biclustering (or two-way clustering) is a methodology allowing for feature set and data points clustering simultaneously, *i.e.*, to find clusters of samples possessing similar characteristics together with features creating these similarities.
   b. The output of biclustering is not a partition or hierarchy of partitions of either rows or columns, but a partition of the whole matrix into sub-matrices or patches.
   c. The goal of biclustering is to find as many patches as possible, and to have them as large as possible, while maintaining strong homogeneity within patches.

Que 4.24. Write short note on CLIQUE.

Answer

1. CLIQUE is a subspace clustering method.
2. CLIQUE (CLustering In QUEst) is a simple grid-based method for finding density based clusters in subspaces.
3. CLIQUE partitions each dimension into non-overlapping intervals, thereby partitioning the entire embedding space of the data objects into cells. It uses a density threshold to identify dense cells and sparse ones.
4. A cell is dense if the number of objects mapped to it exceeds the density threshold.
5. The main strategy behind CLIQUE for identifying a candidate search

space uses the monotonicity of dense cells with respect to dimensionality. This is based on the Apriori property used in frequent pattern and association rule mining.

6. In the context of clusters in subspaces, the monotonicity says the following. A k-dimensional cell c $(k > 1)$ can have at least l points only if every $(k-1)$-dimensional projection of c, which is a cell in a $(k-1)$-dimensional subspace, has at least l points.

7. CLIQUE performs clustering in following two steps :

a. First step :

i. In the first step, CLIQUE partitions the d-dimensional data space into non-overlapping rectangular units, identifying the dense units among these.

ii. CLIQUE finds dense cells in all of the subspaces.

iii. To do so, CLIQUE partitions every dimension into intervals, and identifies intervals containing at least l points, where l is the density threshold.

iv. CLIQUE then iteratively joins two k-dimensional dense cells, c_1 and c_2, in subspaces $(D_{i_1},...,D_{i_k})$ and $(D_{j_1},...,D_{j_k})$, respectively, if $D_{i_1} = D_{j_1}, ..., D_{i_{k-1}} = D_{j_{k-1}}$, and c_1 and c_2 share the same intervals in those dimensions. The join operation generates a new $(k+1)$-dimensional candidate cell c in space $(D_{i_1},...,D_{i_{k-1}}, D_{i_k}, D_{j_k})$.

v. CLIQUE checks whether the number of points in c passes the density threshold. The iteration terminates when no candidates can be generated or no candidate cells are dense.

b. Second step :

i. In the second step, CLIQUE uses the dense cells in each subspace to assemble clusters, which can be of arbitrary shape.

ii. The idea is to apply the Minimum Description Length (MDL) principle to use the maximal regions to cover connected dense cells, where a maximal region is a hyper rectangle where every cell falling into this region is dense, and the region cannot be extended further in any dimension in the subspace.

Que 4.25. Write short notes on PROCLUS.

Answer

1. Projected clustering (PROCLUS) is a top-down subspace clustering algorithm.
2. PROCLUS samples the data and then selects a set of k-medoids and iteratively improves the clustering.
3. PROCLUS is actually faster than CLIQUE due to the sampling of large data sets.

4. The three phases of PROCLUS are as follows :
   a. **Initialization phase :** Select a set of potential medoids that are far apart using a greedy algorithm.
   b. **Iteration phase :**
      i. Select a random set of k-medoids from this reduced data set to determine if clustering quality improves by replacing current medoids with randomly chosen new medoids.
      ii. Cluster quality is based on the average distance between instances and the nearest medoid.
      iii. For each medoid, a set of dimensions is chosen whose average distances are small compared to statistical expectation.
      iv. Once the subspaces have been selected for each medoid, average Manhattan segmental distance is used to assign points to medoids, forming dusters.
   c. **Refinement phase :**
      i. Compute a new list of relevant dimensions for each medoid based on the clusters formed and reassign points to medoids, removing outliers.
      ii. The distance-based approach of PROCLUS is biased toward clusters that are hype-spherical in shape.

Que 4.26. Discuss the basic subspace clustering approaches.

Answer

Basic subspace clustering approaches are :

1. **Grid-based subspace clustering :**
   a. In this approach, data space is divided into axis-parallel cells. Then the cells containing objects above a predefined threshold value given as a parameter are merged to form subspace clusters. Number of intervals is another input parameter which defines range of values in each grid.
   b. Apriori property is used to prune non-promising cells and to improve efficiency.
   c. If a unit is found to be dense in $k-1$ dimension, then it is considered for finding dense unit in k dimensions.
   d. If grid boundaries are strictly followed to separate objects, accuracy of clustering result is decreased as it may miss neighbouring objects which get separated by string grid boundary. Clustering quality is highly dependent on input parameters.

2. **Window-based subspace clustering :**
   a. Window-based subspace clustering overcomes drawbacks of cell-based subspace clustering that it may omit significant results.
   b. Here a window slides across attribute values and obtains overlapping intervals to be used to form subspace clusters.
   c. The size of the sliding window is one of the parameters. These algorithms generate axis-parallel subspace clusters.
3. **Density- based subspace clustering :**
   a. A density-based subspace clustering overcome drawbacks of grid-based subspace clustering algorithms by not using grids.
   b. A cluster is defined as a collection of objects forming a chain which fall within a given distance and exceed predefined threshold of object count. Then adjacent dense regions are merged to form bigger clusters.
   c. As no grids are used, these algorithms can find arbitrarily shaped subspace clusters.
   d. Clusters are built by joining together the objects from adjacent dense regions.
   e. These approaches are prone to values of distance parameters.
   f. The effect curse of dimensionality is overcome in density-based algorithms by utilizing a density measure which is adaptive to subspace size.

Que 4.27. What are the major tasks of clustering evaluation ?

Answer

The major tasks of clustering evaluation include the following:

1. **Assessing clustering tendency :**
   a. In this task, for a given data set, we assess whether a non-random structure exists in the data.
   b. Blindly applying a clustering method on a data set will return clusters; however, the clusters mined may be misleading.
   c. Clustering analysis on a data set is meaningful only when there is a nonrandom structure in the data.
2. **Determining the number of clusters in a data set :**
   a. A few algorithms, such as k-means, require the number of clusters in a data set as the parameter.
   b. Moreover, the number of clusters can be regarded as an interesting and important summary statistic of a data set.
   c. Therefore, it is desirable to estimate this number even before a clustering algorithm is used to derive detailed clusters.

3. Measuring clustering quality :

a. After applying a clustering method on a data set, we want to assess how good the resulting clusters are.

b. A number of measures can be used.

c. Some methods measure how well the clusters fit the data set, while others measure how well the clusters match the ground truth, if such truth is available.

d. There are also measures that score clustering and thus can compare two sets of clustering results on the same data set.

PART-5

Clustering in Non-Euclidean Space, Clustering For Streams and Parallelism.

Questions-Answers

Long Answer Type and Medium Answer Type Questions

Que 4.28. Explain representation of clusters in GRGPF algorithm.

Answer

1. The representation of a cluster in main memory consists of several features.
2. Before listing these features, if p is any point in a cluster, let ROWSUM(p) be the sum of the squares of the distances from p to each of the other points in the cluster.
3. The following features form the representation of a cluster.
   a. N, the number of points in the cluster.
   b. The clustroid of the cluster, which is defined specifically to be the point in the cluster that minimizes the sum of the squares of the distances to the other points; that is, the clustroid is the point in the cluster with the smallest ROWSUM.
   c. The rowsum of the clustroid of the cluster.
   d. For some chosen constant k, the k points of the cluster that are closest to the clustroid, and their rowsums. These points are part of the representation in case the addition of points to the cluster causes the clustroid to change. The assumption is made that the new clustroid would be one of these k points near the old clustroid.

5. The k points of the cluster that are furthest from the clustroid and their rowsums. These points are part of the representation so that we can consider whether two clusters are close enough to merge. The assumption is made that if two clusters are close, then a pair of points distant from their respective clustroids would be close.

Que 4.29. Explain initialization of cluster tree in GRGPF algorithm.

Answer

1. The clusters are organized into a tree, and the nodes of the tree may be very large, perhaps disk blocks or pages, as in the case of a B-tree, which the cluster-representing tree resembles.
2. Each leaf of the tree holds as many cluster representations as can fit.
3. A cluster representation has a size that does not depend on the number of points in the cluster.
4. An interior node of the cluster tree holds a sample of the clustroids of the clusters represented by each of its subtrees, along with pointers to the roots of those subtrees.
5. The samples are of fixed size, so the number of children that an interior node may have is independent of its level.
6. As we go up the tree, the probability that a given cluster's clustroid is part of the sample diminishes.
7. We initialize the cluster tree by taking a main-memory sample of the dataset and clustering it hierarchically.
8. The result of this clustering is a tree T, but T is not exactly the tree used by the GRGPF Algorithm. Rather, we select from T certain of its nodes that represent clusters of approximately some desired size n.
9. These are the initial clusters for the GRGPF Algorithm, and we place their representations at the leaf of the cluster-representing tree. We then group clusters with a common ancestor in T into interior nodes of the cluster-representing tree. In some cases, rebalancing of the cluster-representing tree will be necessary.

Que 4.30. Write short note on BMDO stream clustering algorithm.

Answer

1. In BMDO algorithm, the points of the stream are partitioned into, by, buckets whose sizes are a power of two. Here, the size of a bucket is the number of points it represents, rather than the number of stream elements that are 1.

2. The sizes of buckets obey the restriction that there is one or two of each size, up to some limit. They are required only to form a sequence where each size is twice the previous size such as 3, 6, 12, 24, . . .
3. The contents of a bucket consist of :
   a. The size of the bucket.
   b. The timestamp of the bucket, that is, the most recent point that contributes to the bucket.
   c. A collection of records that represent the clusters into which the points of that bucket have been partitioned. These records contain:
      i. The number of points in the cluster.
      ii. The centroid or clustroid of the cluster.
      iii. Any other parameters necessary to enable us to merge clusters and maintain approximations to the full set of parameters for the merged cluster.