5 Unsupervised Learning

5.1 Supervised vs. Unsupervised Learning

Criteria	Supervised learning	Unsupervised learning
Input Data	Algorithms are trained using labeled data.	Algorithms are used against data that is unlabeled.
Computational Complexity	Simpler method	Computationally complex
Accuracy	Highly accurate	Less accurate
No. of classes	No. of classes is known	No. of classes is not known
Data Analysis	Uses offline analysis	Uses real-time analysis of data
Algorithms used	Linear and Logistics regression, Random forest, Support Vector Machine, Neural Network, etc.	K-Means clustering, Hierarchical clustering, Apriori algorithm, etc.
Output	Desired output is given.	Desired output is not given.
Training data	Use training data to infer model.	No training data is used.
Complex model	It is not possible to learn larger and more complex models than with supervised learning.	It is possible to learn larger and more complex models with unsupervised learning.
Model	We can test our model.	We can not test our model.
Example	Example: Optical character recognition.	Example: Find a face in an image.

Unsupervised Learning:

Unsupervised learning is a type of machine learning algorithm where the input data is not labeled and the algorithm must find patterns or structure within the data on its own.

Unlike supervised learning, there are no target variables to predict, and the algorithm is left to discover patterns and relationships on its own.

5.2 Applications of unsupervised learning

1. Clustering: Unsupervised learning algorithms like k-means and hierarchical clustering are widely used in customer segmentation, image segmentation, document clustering, and social network analysis.

   For example, a clothing retailer may use unsupervised learning to identify three distinct customer segments: young fashion-conscious shoppers, budget-conscious families, and outdoor enthusiasts. They can then tailor their advertising and product offerings to each segment, improving customer engagement and increasing sales.

2. Anomaly detection: Unsupervised learning algorithms like isolation forest and one-class SVM can be used to detect anomalies in data such as fraud detection, intrusion detection, and manufacturing quality control.

   For example, a bank may use unsupervised learning to detect credit card fraud by identifying transactions that are significantly different from a customer's usual spending pattern. They can then flag these transactions for further investigation or automatically block them to prevent fraudulent activity.

3. Dimensionality reduction: Techniques like principal component analysis (PCA), t-SNE, and autoencoders are used for reducing the complexity of high-dimensional data to improve visualization, compression, and feature selection.

4. Association rule learning: Unsupervised learning algorithms like Apriori and FP-Growth can be used for market basket analysis, product recommendation, and customer behavior analysis.

5. Topic modeling: Unsupervised learning algorithms like latent Dirichlet allocation (LDA) and non-negative matrix factorization (NMF) can be used to identify topics in large document collections and social media data.

6. Generative models: Unsupervised learning algorithms like variational autoencoders (VAEs) and generative adversarial networks (GANs) are used for generating synthetic data, image synthesis, and style transfer.

7. Recommendation Systems: Collaborative filtering algorithms like matrix factorization or nearest-neighbor can be used to make personalized recommendations to users based on their past behavior or preferences. This is commonly used in e-commerce, music or video streaming, and social media platforms.

8. Natural Language Processing: Unsupervised learning algorithms like topic modeling or word embeddings can be used to identify patterns and relationships in large text datasets, improving text classification, sentiment analysis, and language translation.

9. Bioinformatics: Unsupervised learning algorithms like clustering or principal component analysis can be used to identify patterns in gene expression data, improve drug discovery, or predict disease risk.

10. Outlier Detection: Clustering or density-based algorithms can be used to identify outliers or anomalies in data that may indicate errors or outliers.

This is used in finance, manufacturing, and healthcare applications.

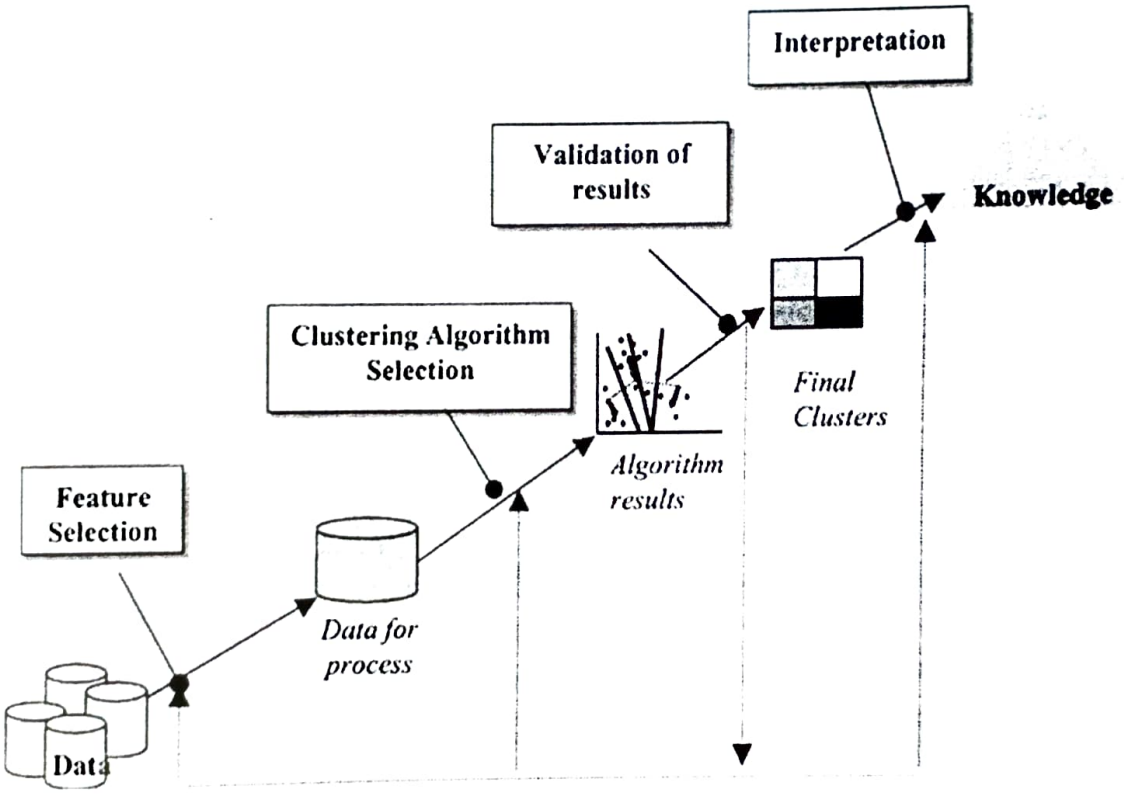
5.3 Clustering

Clustering is a type of unsupervised learning algorithm in machine learning that involves grouping similar data points together into clusters based on their features or attributes. The main objective of clustering is to partition the data in such a way that the points within each cluster are similar to each other and different from points in other clusters.

The choice of clustering algorithm depends on the characteristics of the data and the specific requirements of the application. Clustering algorithms may also require preprocessing steps like normalization or feature scaling to ensure that the features are comparable across different data points.

Overall, clustering is a powerful technique in machine learning that can help uncover hidden patterns in data and facilitate decision-making in various applications.

The following figure shows the steps of clustering process:



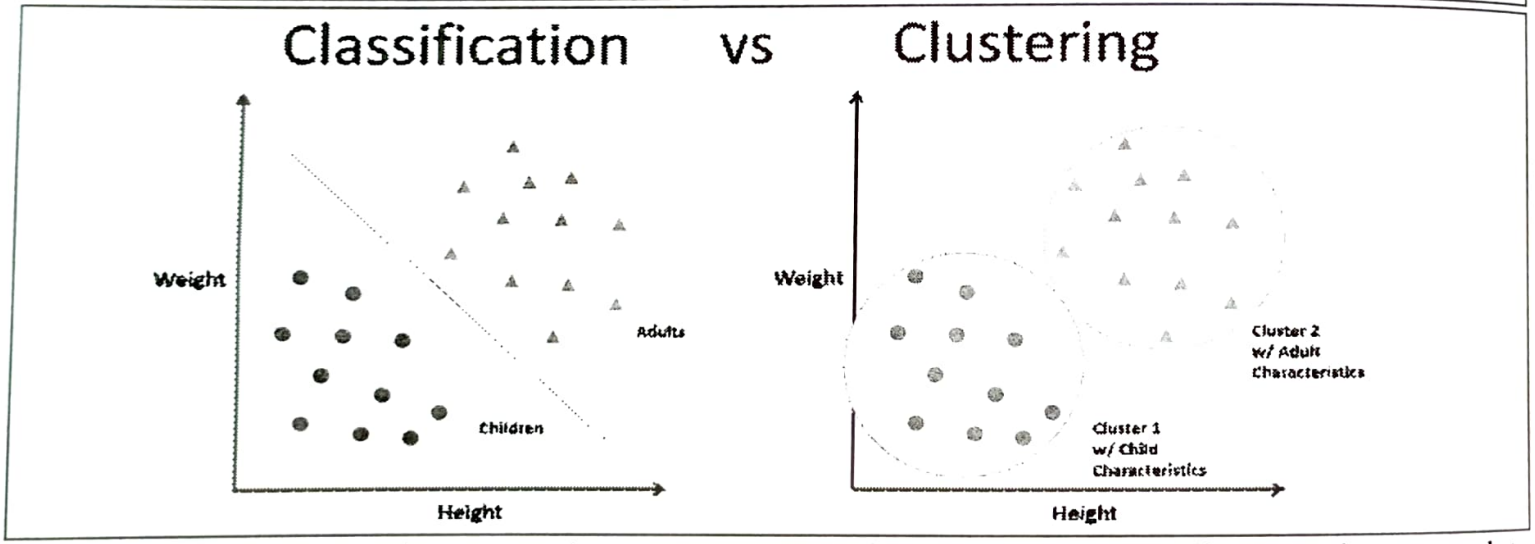
Clustering

Clustering is very much important as it determines the intrinsic grouping among the unlabelled data present. There are no criteria for good clustering. It depends on the user, what is the criteria they may use which satisfy their need.

Let's see how clustering is differ from classification?

Clustering is the method of unsupervised learning which classification is the method of supervised learning.

	Clustering
Users labelled data as the input	Users unlabelled data as the input
The output is known	The output is unknown
Uses supervised machine learning	Uses unsupervised machine learning
A training data set is provided and used to produce classifications	A training data set is provided and used to produce clusters
Examples of algorithms: Decision-trees, Bayesian Classifiers and Support Vector Machine (SVM)	Examples of algorithms: Partition-based clustering (k-means), Hierarchical clustering (agglomerative & divisive) and DBSCAN
Can be more compex than clustering	Can be less compex than classification
Does not specify areas for improvement	specifies areas for improvement
Two-phase	Single-phase
Boundary conditions must be specified	Boundary conditions do not always need to specified



5.2.1 K-means clustering Algorithm

K-means clustering is a popular unsupervised machine learning algorithm used for clustering data points into K clusters. The algorithm starts with randomly selecting K centroids, and then assigns each data point to the nearest centroid. It then calculates the new centroid of each cluster and repeats the process until convergence. The main objective of the algorithm is to minimize the sum of squared distances between data points and their assigned centroids.

The steps of the k-Means clustering algorithm are as follows:

1. Initialize the algorithm by selecting k random points from the dataset as the initial centroids.
2. Assign each data point to the nearest centroid, based on the Euclidean distance.
3. Calculate the new centroid of each

cluster by taking the mean of all data points assigned to that cluster.

4. Repeat steps 2 and 3 until the centroids no longer change or a specified number of iterations is reached.
5. The resulting clusters are the groups of data points that are closest to their respective centroids.

Pseudo code for the k-means clustering algorithm is:

Algorithm 1 *k*-means algorithm

1: Specify the number k of clusters to assign.
2: Randomly initialize k centroids.
3: **repeat**
4: **expectation:** Assign each point to its closest centroid.
5: **maximization:** Compute the new centroid (mean) of each cluster.
6: **until** The centroid positions do not change.

Example: Cluster the following eight points (with (x, y) representing locations) into three 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)

Initial cluster centers are :

A1(2, 10), A4(5, 8) and A7(1, 2).

The distance function between two points a = (x1, y1) and b = (x2, y2) is defined as-

$\tilde{N}(a, b) = |x2 - x1| + |y2 - y1|$

Use K-Means Algorithm to find the three cluster centers after the first iteration.

Iteration-01 :

- We calculate the distance of each point from each of the center of the three clusters.
- The distance is calculated by using the given distance function.

The following illustration shows the calculation of distance between point A1(2, 10) and each of the center of the three clusters-

Calculating Distance Between A1(2, 10) and C1(2, 10)-

$\tilde{N}(A1, C1)$
$= |x2 - x1| + |y2 - y1|$
$= |2 - 2| + |10 - 10|$
$= 0$

Calculating Distance Between A1(2, 10) and C2(5, 8)-

$\tilde{N}(A1, C2)$
$= |x2 - x1| + |y2 - y1|$
$= |5 - 2| + |8 - 10|$
$= 3 + 2$
$= 5$

Calculating Distance Between A1(2, 10) and C3(1, 2)-

P(A1, C3)
$= |x2 - x1| + |y2 - y1|$
$= |1 - 2| + |2 - 10|$
$= 1 + 8$
$= 9$

In the similar manner, we calculate the distance of other points from each of the center of the three clusters.

Next,

We draw a table showing all the results.

Using the table, we decide which point belongs to which cluster.

The given point belongs to that cluster whose center is nearest to it.

Given Points	Distance from center (2, 10) of Cluster-01	Distance from center (5, 8) of Cluster-02	Distance from center (1, 2) of Cluster-03	Point belongs to Cluster
A1(2, 10)	0	5	9	C1
A2(2, 5)	5	6	4	C3
A3(8, 4)	12	7	9	C2
A4(5, 8)	5	0	10	C2
A5(7, 5)	10	5	9	C2
A6(6, 4)	10	5	7	C2
A7(1, 2)	9	10	0	C3
A8(4, 9)	3	2	10	C2

From here, New clusters are-

Cluster-01:

First cluster contains points-

A1(2, 10)

Cluster-02:

Second cluster contains points-

A3(8, 4)

A4(5, 8)

A5(7, 5)

A6(6, 4)

A8(4, 9)

Cluster-03:

Third cluster contains points-

A2(2, 5)

A7(1, 2)

Similarly, we can apply this for iteration 2, 3 and so on.

Pros:

- K-means is fast and scalable, making it ideal for large datasets.
- It is relatively simple to implement and can work well with high-dimensional data.
- K-means is efficient and can handle noisy data.
- It can be used for a wide range of applications, including customer segmentation, image analysis, and text mining.

Cons:

- The performance of K-means depends on the initial placement of centroids, which can lead to suboptimal solutions.
- The algorithm may converge to local optima, which may not be the global optimum.
- K-means assumes that all clusters have the same variance, which may not be the case in some datasets.
- It may not work well with non-linearly separable data.

Applications:

- Customer Segmentation: K-means clustering can be used to group customers based on their behavior, preferences, or needs, allowing businesses to tailor their marketing strategies and offers to specific customer segments.
- Image Segmentation: K-means can be used to segment images by grouping

similar pixels together into clusters, allowing for image compression or object recognition.

- Text Clustering: K-means can be used to group similar text documents together based on their content, improving search results and recommendation systems.
- Anomaly Detection: K-means can be used to identify outliers or anomalies in data that may indicate fraud, intrusion, or equipment malfunction.
- Bioinformatics: K-means can be used to identify patterns in gene expression data or to cluster proteins based on their properties, improving drug discovery or disease diagnosis.

5.4 Finding pattern using Association Rule

The **Association Rule** is a rule-based machine learning method for identifying associations between unrelated elements using pattern recognition.

Support and confidence are two important measures used in association rule mining to evaluate the significance of frequent itemsets and association rules.

Support refers to the frequency of occurrence of an itemset in a dataset, expressed as a percentage or proportion of the total transactions in the dataset that contain the itemset.

Support(A) = (Number of transactions containing A) / (Total number of transactions)

It measures the degree of association between items and is used to identify frequent itemsets that occur frequently enough to be considered interesting or significant.

For example, if a transaction dataset contains 100 transactions and a particular itemset occurs in 20 of them, the support of the itemset would be 20%.

Confidence, on the other hand, refers to the conditional probability that a transaction containing one set of items will also contain another set of items. It measures the strength of the association between items and is used to generate association rules that express the conditional relationships between items.

Confidence(X => Y) = (Number of transactions containing X and Y) / (Number of transactions containing X)

For example, if a transaction dataset contains 100 transactions and a particular association rule has a confidence of 80%, it means that in 80% of the transactions that contain the antecedent (left-hand side) of the rule, the consequent (right-hand side) also occurs.

5.4.1 Apriori Algorithm

Purpose: The Apriori Algorithm is an influential algorithm for mining frequent itemsets for Boolean association rules.

Key Concepts:

Frequent Itemsets: The sets of item which has minimum support (denoted by Li for i^{th}-Itemset).

Apriori Property: Any subset of frequent itemset must be frequent.

Join Operation: To find L_k, a set of candidate k-itemsets is generated by joining Lk-1 itself.

Find the frequent itemsets: the sets of items that have minimum support – A subset of a frequent itemset must also be a frequent itemset (Apriori Property)

i.e., if {AB} is a frequent itemset, both {A} and {B} should be a frequent itemset

– Iteratively find frequent itemsets with cardinality from 1 to k (k-itemset)

Use the frequent itemsets to generate association rules.

The Apriori Algorithm :

The steps of the Apriori algorithm are as follows:

1. Set the minimum support threshold to a desired value.
2. Generate all frequent 1-itemsets by scanning the dataset and counting the support of each item.
3. Repeat the following steps until no more frequent itemsets can be generated:
   a. Generate candidate itemsets by joining pairs of frequent (k-1)-itemsets.
   b. Prune the candidate itemsets that contain infrequent (k-1)-itemsets.
   c. Count the support of each candidate itemset by scanning the dataset.
   d. Keep only the frequent itemsets that meet the minimum support threshold.
4. Generate association rules from the frequent itemsets by applying a minimum confidence threshold.

Pseudo code

- Join Step: C k is generated by joining L_k-1with itself
- Prune Step: Any (k-1)-itemset that is not frequent cannot be a subset of frequent k-itemset

Algorithm: *Apriori algorithm*

Input: *D*: Input Dataset
minSup: minimum support threshold

Output: All 2 to *k*-frequent itemsets

```
1.  L1 = {1-frequent itemset}; // found separately
2.  for (k = 2; Lk-1 ≠ φ; k++)
3.    Ck = apriori_gen(Lk-1) // finds k-candidate itemsets by
        joining and pruning Lk-1 with itself
4.    for each transaction t in D
5.        Ct = subset (Ck, t) // finds candidate itemsets in t
6.        for each c in Ct
7.            c.count++
8.        end for each
9.    end for each
10.   Lk = {c ∈ Ck | c.count ≥ minSup}
11. end for
12. Return ∪k Lk
```

Let's see an example of the Apriori Algorithm.

Transaction ID	Items
T1	Hot Dogs, Buns, Ketchup
T2	Hot Dogs, Buns
T3	Hot Dogs, Coke, Chips
T4	Chips, Coke
T5	Chips, Ketchup
T6	Hot Dogs, Coke, Chips

Find the **frequent itemsets** and generate **association rules** on this. Assume that minimum support threshold (s = 33.33%) and minimum confident threshold (c = 60%)
Let's start,

$$\text{minimum support count} = \frac{33.33}{100} \times 6$$
$$= 2$$

Item set	Sup-count
Hot Dogs	4
Buns	2
Ketchup	2
Coke	3
Chips	4

Item set	Sup-count
Hot Dogs	4
Buns	2
Ketchup	2
Coke	3
Chips	4

Item set	Sup-count
Hot Dogs, Buns	2
Hot Dogs, Ketchup	1
Hot Dogs, Coke	2
Hot Dogs, Chips	2
Buns, Ketchup	1
Buns, Coke	0
Buns, Chips	0
Ketchup, Coke	0
Ketchup, Chips	1
Coke, Chips	3

Item set	Sup-count
Hot Dogs, Buns	2
Hot Dogs, Coke	2
Hot Dogs, Chips	2
Coke, Chips	3

Item set	Sup-count
Hot Dogs, Buns, Coke	0
Hot Dogs, Buns, Chips	0
Hot Dogs, Coke, Chips	2

Item set	Sup-count
Hot Dogs, Coke, Chips	2

There is only one itemset with minimum support 2.

So only one itemset is frequent.

Frequent Itemset (I) = {Hot Dogs, Coke, Chips}

Association rules,

- [Hot Dogs^Coke]=>[Chips] // confidence = sup(Hot Dogs^Coke^Chips)/sup(Hot Dogs^Coke) = 2/2*100=100% // **Selected**
- [Hot Dogs^Chips]=>[Coke] // confidence = sup(Hot Dogs^Coke^Chips)/sup(Hot Dogs^Chips) = 2/2*100=100% // **Selected**
- [Coke^Chips]=>[Hot Dogs] // confidence = sup(Hot Dogs^Coke^Chips)/sup(Coke^Chips) = 2/3*100=66.67% //**Selected**
- [Hot Dogs]=>[Coke^Chips] // confidence = sup(Hot Dogs^Coke^Chips)/sup(Hot Dogs) = 2/4*100=50% //**Rejected**
- [Coke]=>[Hot Dogs^Chips] // confidence = sup(Hot Dogs^Coke^Chips)/sup(Coke) = 2/3*100=66.67% //**Selected**
- [Chips]=>[Hot Dogs^Coke] // confidence = sup(Hot Dogs^Coke^Chips)/sup(Chips) = 2/4*100=50% //**Rejected**

There are four strong results (minimum confidence greater than 60%)

Advantages of Apriori Algorithm:

- Scalability : The Apriori algorithm is a scalable algorithm and can handle large datasets efficiently.
- Interpretability : The Apriori algorithm generates frequent itemsets and association rules that are easy to understand and interpret.
- Flexibility : The algorithm can be customized by adjusting the minimum support and confidence thresholds to suit the specific requirements of the user.
- Applicability : The Apriori algorithm can be used to mine association rules from various types of data, including market basket transactions, web logs, and biological sequences.

Disadvantages of Apriori Algorithm:

- Computational Complexity : The Apriori algorithm has a high computational complexity and may take a long time to generate frequent itemsets and association rules from large datasets.
- Memory Requirements : The algorithm requires a significant amount of memory to store the candidate itemsets and their support counts.
- Curse of Dimensionality : The performance of the algorithm may degrade rapidly as the number of items or attributes in the data increases.

Applications of Apriori Algorithm:

- Market Basket Analysis : The Apriori algorithm is commonly used in market basket analysis to identify the co-occurrence of items in customer transactions and generate recommendations for cross-selling and up-selling.
- Web Usage Mining : The algorithm can be used to mine patterns and trends in web logs, such as frequently visited pages or clickstreams, to improve website design and user experience.
- Bioinformatics : The Apriori algorithm can be applied to biological sequences, such as DNA or protein sequences, to discover frequent patterns or motifs that may be related to gene expression or function.
- Fraud Detection : The algorithm can be used to detect fraudulent behavior in financial transactions by identifying patterns of unusual or suspicious activities.
- Social Network Analysis : The algorithm can be applied to social network data to identify groups or communities of individuals with similar interests or behaviors.

Question Bank

■ Short-Answer Questions (3 or 4 marks) :

1. Give the difference between supervised learning and unsupervised learning.
2. State any four applications of unsupervised learning.
3. Differentiate clustering with classification.
4. Write pseudo code of k-means clustering algorithm.
5. Write strength and weakness of k-means clustering algorithm.
6. Define: Support, Confidence
7. State apriori property.
8. Explain any two applications of apriori algorithm.
9. Write strength and weakness of apriori clustering algorithm.
10. How unsupervised learning is useful in fraud detection?

■ Long-Answer Questions (7 marks):

1. Write and explain applications of unsupervised learning.
2. Write and explain apriori algorithm in detail.
3. Write and explain k-means clustering approach in detail.
4. You are given a set of one-dimensional data points: {5, 10, 15, 20, 25, 30, 35}. Assume that k = 2 and first set of random centroid is selected as {15, 32} and then it is refined with {12, 30}. Create two clusters with each set of centroid mentioned above following the k-means approach.
5. Generate frequent itemsets and generate association rules based on it using apriori algorithm. Minimum support is 50% and minimum confidence is 70%.

TID	Items
100	1, 3, 4
200	2, 3, 5
300	1, 2, 3, 5
400	2, 5

■ Check your knowledge:

1. Which of the following is a clustering algorithm ?

a) Linear regression **b) K-means**

c) Naive Bayes d) SVM

2. Which of the following is an example of an unsupervised learning problem?
   a) Classifying images of animals into different categories
   b) Predicting the price of a house
   c) Clustering customers based on their purchasing behavior
   d) All of above

   hi

3. Apriori algorithm is used for:
   a) Clustering
   b) Classification
   c) Association rule mining
   d) Regression

4. What is the minimum support threshold in Apriori algorithm ?
   a) The minimum number of transactions that an itemset must appear in.
   b) The maximum number of transactions that an itemset must appear in.
   c) The minimum frequency with which an item must appear in the transaction database.
   d) The maximum frequency with which an item must appear in the transaction database.

5. What is the main drawback of the k-means algorithm?
   a) It is only effective for datasets with a small number of dimensions.
   b) It can get stuck in local optima.
   c) It requires the number of clusters to be specified in advance.
   d) It is not effective for datasets with categorical variables.

■