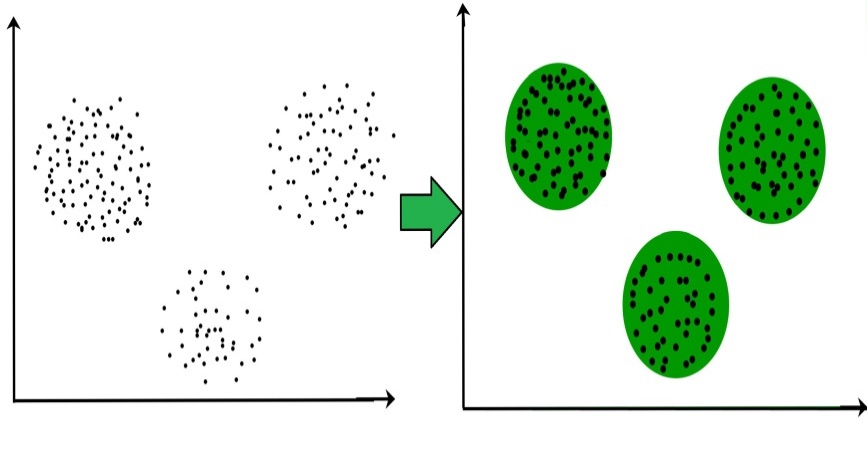
**1. What is clustering? Explain in detail.**

The task of grouping data points based on their similarity with each other is called Clustering or Cluster Analysis. This method is defined under the branch of [Unsupervised Learning](https://www.geeksforgeeks.org/supervised-unsupervised-learning/), which aims at gaining insights from unlabelled data points, that is, unlike [supervised learning](https://www.geeksforgeeks.org/supervised-unsupervised-learning/) we don’t have a target variable.

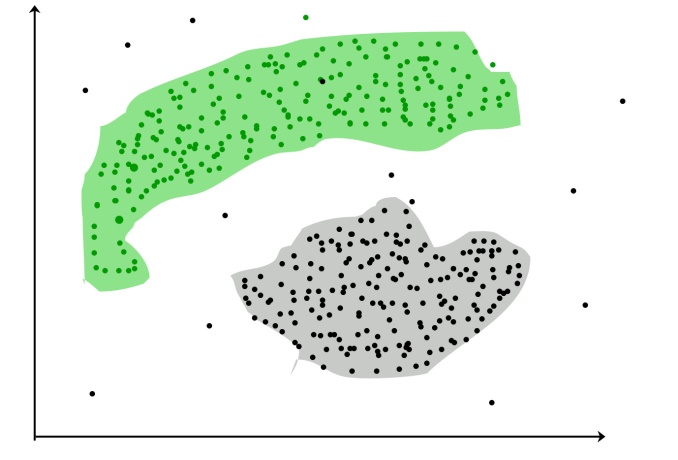
Clustering aims at forming groups of homogeneous data points from a heterogeneous dataset. It evaluates the similarity based on a metric like Euclidean distance, Cosine similarity, Manhattan distance, etc. and then group the points with highest similarity score together.

For Example, In the graph given below, we can clearly see that there are 3 circular clusters forming on the basis of distance.



Now it is not necessary that the clusters formed must be circular in shape. The shape of clusters can be arbitrary. There are many algortihms that work well with detecting arbitrary shaped clusters.

For example, In the below given graph we can see that the clusters formed are not circular in shape.



**Types of Clustering**

Broadly speaking, there are 2 types of clustering that can be performed to group similar data points:

* **Hard Clustering:**In this type of clustering, each data point belongs to a cluster completely or not. For example, Let’s say there are 4 data point and we have to cluster them into 2 clusters. So each data point will either belong to cluster 1 or cluster 2.

| **Data Points** | **Clusters** |
| --- | --- |
| A | C1 |
| B | C2 |
| C | C2 |
| D | C1 |

* **Soft Clustering:**In this type of clustering, instead of assigning each data point into a separate cluster, a probability or likelihood of that point being that cluster is evaluated. For example, Let’s say there are 4 data point and we have to cluster them into 2 clusters. So we will be evaluating a probability of a data point belonging to both clusters. This probability is calculated for all data points.

| Data Points | Probability of C1 | Probability of C2 |
| --- | --- | --- |
| A | 0.91 | 0.09 |
| B | 0.3 | 0.7 |
| C | 0.17 | 0.83 |
| D | 1 | 0 |

**Uses of Clustering**

Now before we begin with types of clustering algorithms, we will go through the use cases of Clustering algorithms. Clustering algorithms are majorly used for:

* [Market Segmentation](https://www.geeksforgeeks.org/customer-segmentation-using-unsupervised-machine-learning-in-python/) – Businesses use clustering to group their customers and use targeted advertisements to attract more audience.

1. [Market Basket Analysis](https://www.geeksforgeeks.org/market-basket-analysis-in-data-mining/) – Shop owners analyze their sales and figure out which items are majorly bought together by the customers. For example, In USA, according to a study diapers and beers were usually bought together by fathers.
2. [Social Network Analysis](https://www.geeksforgeeks.org/social-network-analysis-using-r-programming/) – Social media sites use your data to understand your browsing behaviour and provide you with targeted friend recommendations or content recommendations.
3. Medical Imaging – Doctors use Clustering to find out diseased areas in diagnostic images like X-rays.
4. [Anomaly Detection](https://www.geeksforgeeks.org/machine-learning-for-anomaly-detection/) – To find outliers in a stream of real-time dataset or forecasting fraudulent transactions we can use clustering to identify them.
5. Simplify working with large datasets – Each cluster is given a cluster ID after clustering is complete. Now, you may reduce a feature set’s whole feature set into its cluster ID. Clustering is effective when it can represent a complicated case with a straightforward cluster ID. Using the same principle, clustering data can make complex datasets simpler.

There are many more use cases for clustering but there are some of the major and common use cases of clustering. Moving forward we will be discussing Clustering Algorithms that will help you perform the above tasks.

**2. Explain K Means clustering.**

K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science.

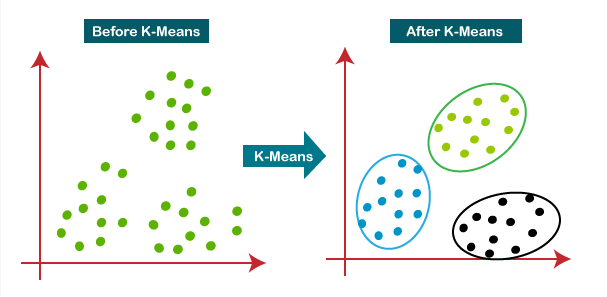
K-Means Clustering is an [Unsupervised Learning algorithm](https://www.javatpoint.com/unsupervised-machine-learning), which groups the unlabeled dataset into different clusters. Here K defines the number of pre-defined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on.

* It is an iterative algorithm that divides the unlabeled dataset into k different clusters in such a way that each dataset belongs only one group that has similar properties.
* It allows us to cluster the data into different groups and a convenient way to discover the categories of groups in the unlabeled dataset on its own without the need for any training.
* It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.
* The algorithm takes the unlabeled dataset as input, divides the dataset into k-number of clusters, and repeats the process until it does not find the best clusters. The value of k should be predetermined in this algorithm.

The k-means [clustering](https://www.javatpoint.com/clustering-in-machine-learning) algorithm mainly performs two tasks:

* Determines the best value for K center points or centroids by an iterative process.
* Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

The below diagram explains the working of the K-means Clustering Algorithm:



Working of K-Means Algorithm:

The working of the K-Means algorithm is explained in the below steps:

**Step-1:** Select the number K to decide the number of clusters.

**Step-2:** Select random K points or centroids. (It can be other from the input dataset).

**Step-3:** Assign each data point to their closest centroid, which will form the predefined K clusters.

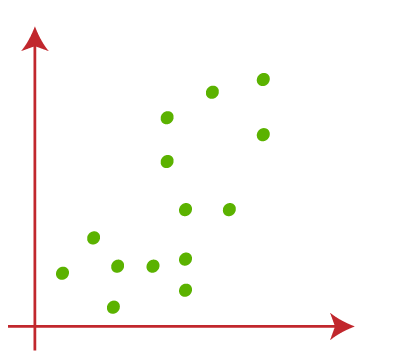
**Step-4:** Calculate the variance and place a new centroid of each cluster.

**Step-5:** Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

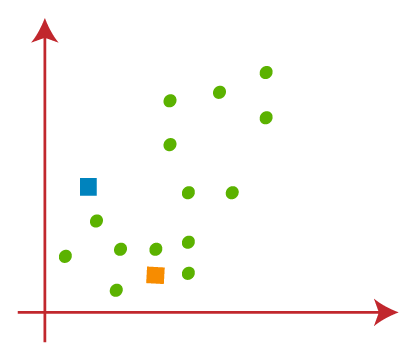
**Step-6:** If any reassignment occurs, then go to step-4 else go to FINISH.

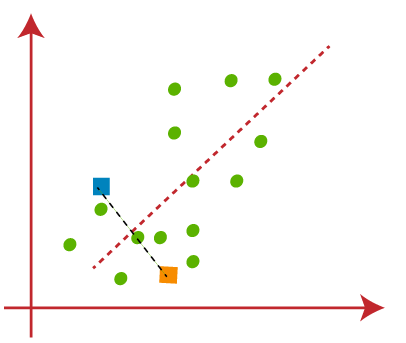
**Step-7**: The model is ready.

Suppose we have two variables M1 and M2. The x-y axis scatter plot of these two variables is given below:

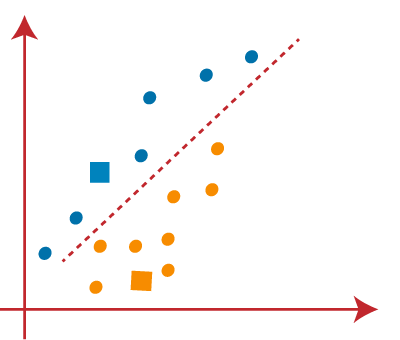


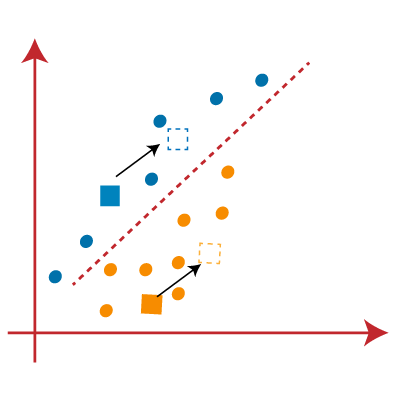
* Let's take number k of clusters, i.e., K=2, to identify the dataset and to put them into different clusters. It means here we will try to group these datasets into two different clusters.
* We need to choose some random k points or centroid to form the cluster. These points can be either the points from the dataset or any other point. So, here we are selecting the below two points as k points, which are not the part of our dataset. Consider the below image:

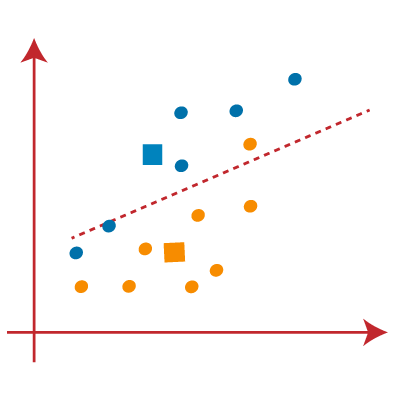


Now we will assign each data point of the scatter plot to its closest K-point or centroid. We will compute it by applying some mathematics that we have studied to calculate the distance between two points. So, we will draw a median between both the centroids. Consider the below image: 

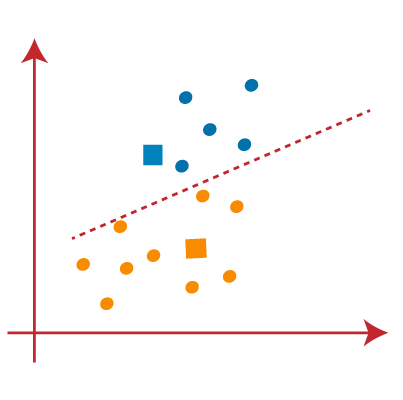
From the above image, it is clear that points left side of the line is near to the K1 or blue centroid, and points to the right of the line are close to the yellow centroid. Let's color them as blue and yellow for clear visualization.



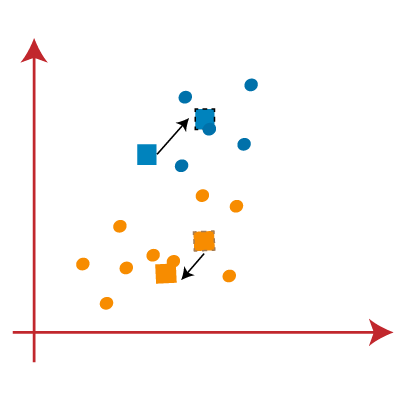
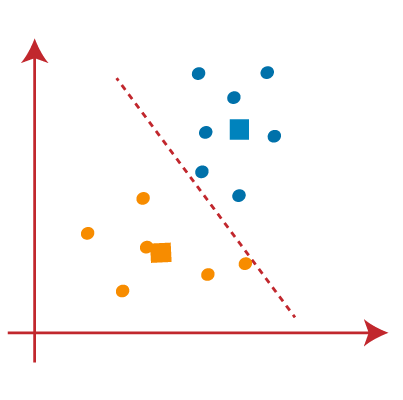
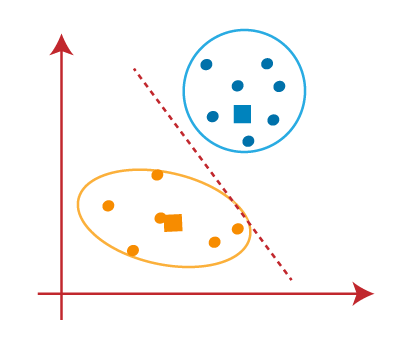
As we need to find the closest cluster, so we will repeat the process by choosing **a new centroid**. To choose the new centroids, we will compute the center of gravity of these centroids, and will find new centroids as below:  


* Next, we will reassign each datapoint to the new centroid. For this, we will repeat the same process of finding a median line. The median will be like below image:  
  

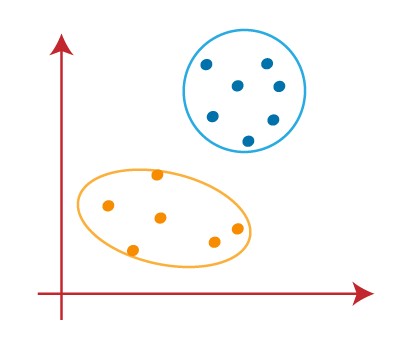
From the above image, we can see, one yellow point is on the left side of the line, and two blue points are right to the line. So, these three points will be assigned to new centroids.



As reassignment has taken place, so we will again go to the step-4, which is finding new centroids or K-points.

* We will repeat the process by finding the center of gravity of centroids, so the new centroids will be as shown in the below image:  
  
* As we got the new centroids so again will draw the median line and reassign the data points. So, the image will be:  
  
* We can see in the above image; there are no dissimilar data points on either side of the line, which means our model is formed. Consider the below image:  
  

As our model is ready, so we can now remove the assumed centroids, and the two final clusters will be as shown in the below image:



### **Advantages**

The following are some advantages of K-Means clustering algorithms −

* It is very easy to understand and implement.
* If we have large number of variables then, K-means would be faster than Hierarchical clustering.
* On re-computation of centroids, an instance can change the cluster.
* Tighter clusters are formed with K-means as compared to Hierarchical clustering.

### **Disadvantages**

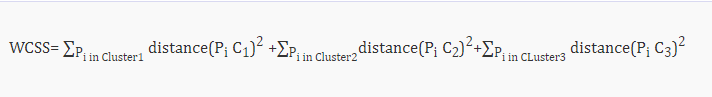
The following are some disadvantages of K-Means clustering algorithms −

* It is a bit difficult to predict the number of clusters i.e. the value of k.
* Output is strongly impacted by initial inputs like number of clusters (value of k).
* Order of data will have strong impact on the final output.
* It is very sensitive to rescaling. If we will rescale our data by means of normalization or standardization, then the output will completely change final output.
* It is not good in doing clustering job if the clusters have a complicated geometric shape.

**3. Explain Elbow Method in K Means clustering**

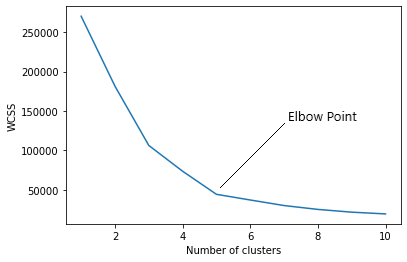
The Elbow Method is a technique used in data analysis and machine learning for determining the optimal number of clusters in a dataset. It involves plotting the variance explained by different numbers of clusters and identifying the “elbow” point, where the rate of variance decreases sharply levels off, suggesting an appropriate cluster count for analysis or model training.

The Elbow method is one of the most popular ways to find the optimal number of clusters. This method uses the concept of WCSS value. WCSS stands for Within Cluster Sum of Squares, which defines the total variations within a cluster. The formula to calculate the value of WCSS (for 3 clusters) is given below:

****

**K Means Clustering Using the Elbow Method**

1. In the Elbow method, we are actually varying the number of clusters (K) from 1 – 10.
2. For each value of K, we are calculating WCSS (Within-Cluster Sum of Square).
3. WCSS is the sum of the squared distance between each point and the centroid in a cluster.
4. When we plot the WCSS with the K value, the plot looks like an Elbow.
5. As the number of clusters increases, the WCSS value will start to decrease. WCSS value is largest when K = 1.
6. When we analyze the graph, we can see that the graph will rapidly change at a point and thus creating an elbow shape.
7. From this point, the graph moves almost parallel to the X-axis. The K value corresponding to this point is the optimal value of K or an optimal number of clusters.



**Pros of Elbow Method:**

1. Simplicity: It's easy to understand and implement. You just plot the WCSS vs number of clusters (k) and look for the elbow.
2. Visualization: The graph provides a visual aid for choosing k, making it easier to understand the relationship between cluster number and data fit.
3. Efficiency: It's computationally cheap compared to other methods for choosing k.

**Elbow Method Drawbacks:**

The elbow method, while a useful tool for determining the optimal number of clusters in K-means clustering, has some drawbacks:

1. Subjectivity: The choice of the “elbow point” can be subjective and might vary between individuals analyzing the same data.
2. Non-Gaussian Data: It assumes that clusters are spherical and equally sized, which may not hold for complex datasets with irregularly shaped or differently sized clusters.
3. Sensitivity to Initialization: K-means itself is sensitive to initial cluster centroids, which can affect the WCSS values and, consequently, the choice of the optimal K.
4. Inefficient for Large Datasets: For large datasets, calculating WCSS for a range of K values can be computationally expensive and time-consuming.
5. Unsuitable for All Distributions: The elbow method is not suitable for all data distributions, especially when clusters have varying densities or are non-convex.
6. Limited to K-means: It specifically applies to K-means clustering and may not be suitable for other clustering algorithms with different objectives.

**4) . Explain Hierarchical clustering.**

# **Hierarchical Clustering in Machine Learning**

Hierarchical clustering is another unsupervised machine learning algorithm, which is used to group the unlabeled datasets into a cluster and also known as **hierarchical cluster analysis** or HCA.

In this algorithm, we develop the hierarchy of clusters in the form of a tree, and this tree-shaped structure is known as the **dendrogram**.

Sometimes the results of K-means clustering and hierarchical clustering may look similar, but they both differ depending on how they work. As there is no requirement to predetermine the number of clusters as we did in the K-Means algorithm.

The hierarchical clustering technique has two approaches:

1. **Agglomerative:** Agglomerative is a **bottom-up** approach, in which the algorithm starts with taking all data points as single clusters and merging them until one cluster is left.
2. **Divisive:** Divisive algorithm is the reverse of the agglomerative algorithm as it is a **top-down approach.**

### **Why hierarchical clustering?**

As we already have other [clustering](https://www.javatpoint.com/clustering-in-machine-learning) algorithms such as [K-Means Clustering](https://www.javatpoint.com/k-means-clustering-algorithm-in-machine-learning), then why we need hierarchical clustering? So, as we have seen in the K-means clustering that there are some challenges with this algorithm, which are a predetermined number of clusters, and it always tries to create the clusters of the same size. To solve these two challenges, we can opt for the hierarchical clustering algorithm because, in this algorithm, we don't need to have knowledge about the predefined number of clusters.

In this topic, we will discuss the Agglomerative Hierarchical clustering algorithm.

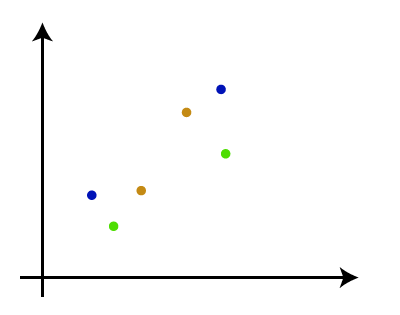
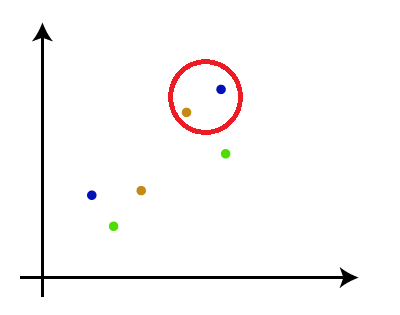
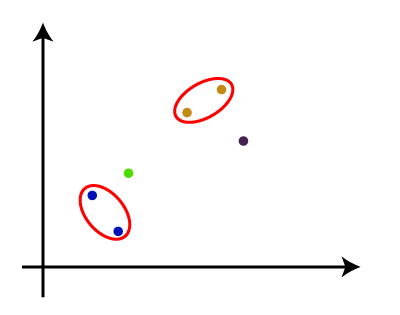
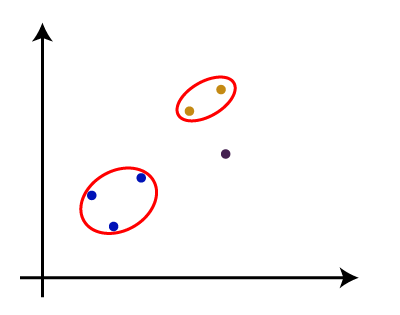
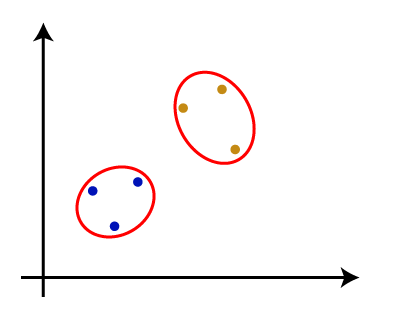
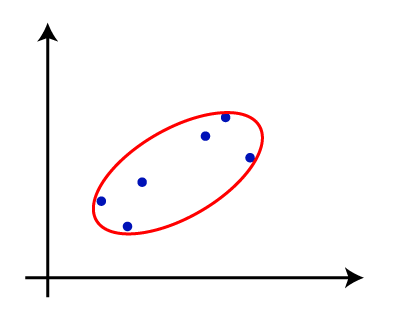
## Agglomerative Hierarchical clustering

The agglomerative hierarchical clustering algorithm is a popular example of HCA. To group the datasets into clusters, it follows the **bottom-up approach**. It means, this algorithm considers each dataset as a single cluster at the beginning, and then start combining the closest pair of clusters together. It does this until all the clusters are merged into a single cluster that contains all the datasets.

This hierarchy of clusters is represented in the form of the dendrogram.

## How the Agglomerative Hierarchical clustering Work?

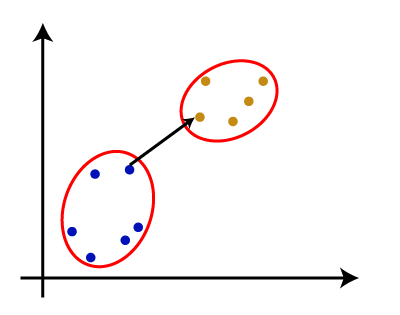
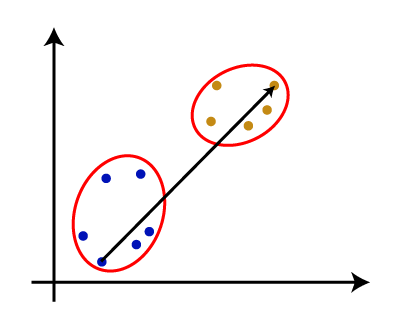
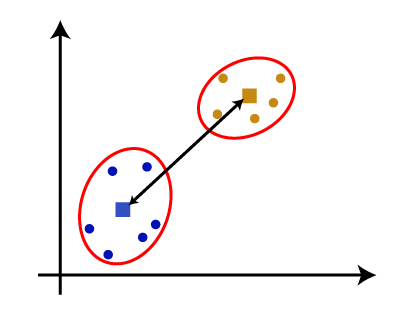
The working of the AHC algorithm can be explained using the below steps:

* **Step-1:** Create each data point as a single cluster. Let's say there are N data points, so the number of clusters will also be N.  
  
* **Step-2:** Take two closest data points or clusters and merge them to form one cluster. So, there will now be N-1 clusters.  
  
* **Step-3**: Again, take the two closest clusters and merge them together to form one cluster. There will be N-2 clusters.  
  
* **Step-4:** Repeat Step 3 until only one cluster left. So, we will get the following clusters. Consider the below images:  
    
    
  
* **Step-5:** Once all the clusters are combined into one big cluster, develop the dendrogram to divide the clusters as per the problem.

#### **Note: To better understand hierarchical clustering, it is advised to have a look on k-means clustering**

### **Measure for the distance between two clusters**

As we have seen, the **closest distance** between the two clusters is crucial for the hierarchical clustering. There are various ways to calculate the distance between two clusters, and these ways decide the rule for clustering. These measures are called **Linkage methods**. Some of the popular linkage methods are given below:

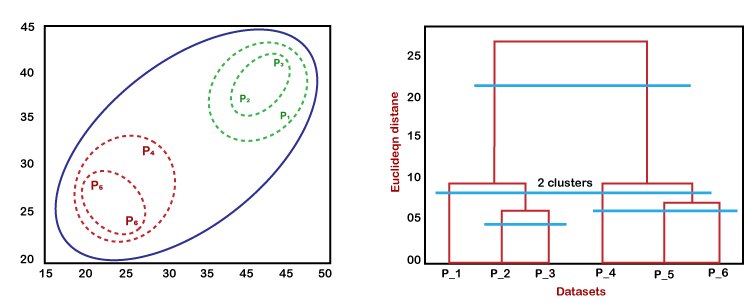
1. **Single Linkage:** It is the Shortest Distance between the closest points of the clusters. Consider the below image:  
   
2. **Complete Linkage:** It is the farthest distance between the two points of two different clusters. It is one of the popular linkage methods as it forms tighter clusters than single-linkage.  
   
3. **Average Linkage:** It is the linkage method in which the distance between each pair of datasets is added up and then divided by the total number of datasets to calculate the average distance between two clusters. It is also one of the most popular linkage methods.
4. **Centroid Linkage:** It is the linkage method in which the distance between the centroid of the clusters is calculated. Consider the below image:  
   

From the above-given approaches, we can apply any of them according to the type of problem or business requirement.

### **Woking of Dendrogram in Hierarchical clustering**

The dendrogram is a tree-like structure that is mainly used to store each step as a memory that the HC algorithm performs. In the dendrogram plot, the Y-axis shows the Euclidean distances between the data points, and the x-axis shows all the data points of the given dataset.

The working of the dendrogram can be explained using the below diagram:



In the above diagram, the left part is showing how clusters are created in agglomerative clustering, and the right part is showing the corresponding dendrogram.

* As we have discussed above, firstly, the datapoints P2 and P3 combine together and form a cluster, correspondingly a dendrogram is created, which connects P2 and P3 with a rectangular shape. The hight is decided according to the Euclidean distance between the data points.
* In the next step, P5 and P6 form a cluster, and the corresponding dendrogram is created. It is higher than of previous, as the Euclidean distance between P5 and P6 is a little bit greater than the P2 and P3.
* Again, two new dendrograms are created that combine P1, P2, and P3 in one dendrogram, and P4, P5, and P6, in another dendrogram.
* At last, the final dendrogram is created that combines all the data points together.

We can cut the dendrogram tree structure at any level as per our requirement.

1. **What is Agglomerative Hierarchical clustering?**

The agglomerative hierarchical clustering algorithm is a popular example of HCA. To group the datasets into clusters, it follows the **bottom-up approach**. It means, this algorithm considers each dataset as a single cluster at the beginning, and then start combining the closest pair of clusters together. It does this until all the clusters are merged into a single cluster that contains all the datasets.

It is also known as the bottom-up approach or hierarchical agglomerative clustering (HAC). A structure that is more informative than the unstructured set of clusters returned by flat clustering. This clustering algorithm does not require us to prespecify the number of clusters. Bottom-up algorithms treat each data as a singleton cluster at the outset and then successively agglomerate pairs of clusters until all clusters have been merged into a single cluster that contains all data.

**Algorithm :**

given a dataset (d1, d2, d3, ....dN) of size N

# compute the distance matrix

for i=1 to N:

# as the distance matrix is symmetric about

# the primary diagonal so we compute only lower

# part of the primary diagonal

for j=1 to i:

dis\_mat[i][j] = distance[di, dj]

each data point is a singleton cluster

**repeat**

merge the two cluster having minimum distance

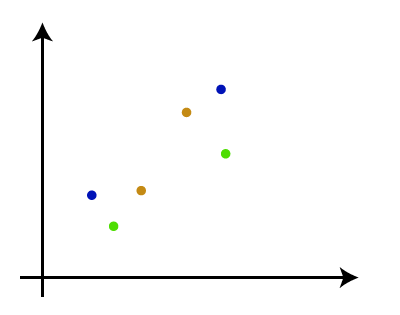
update the distance matrix

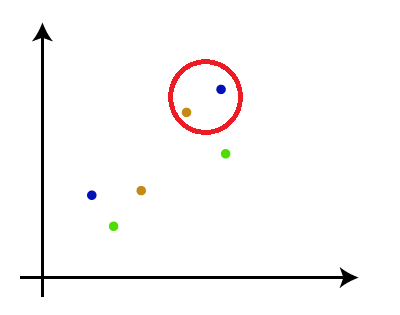
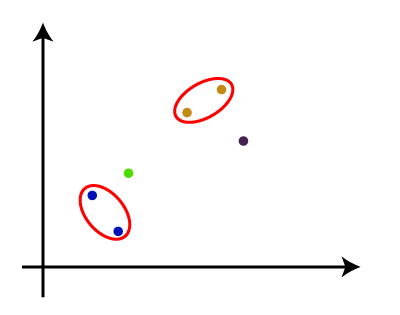
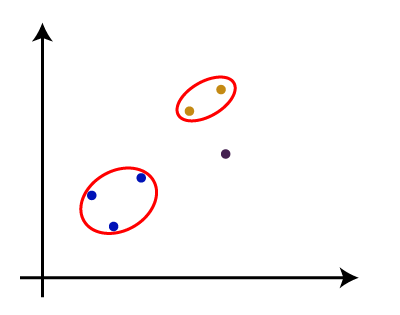
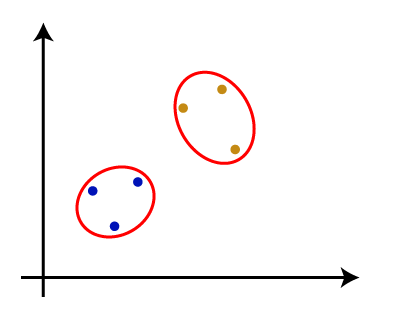
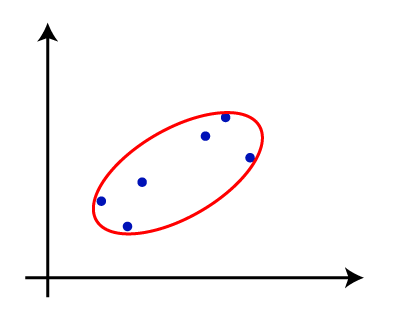
**until** only a single cluster remains

This hierarchy of clusters is represented in the form of the dendrogram.

How the Agglomerative Hierarchical clustering Work?

The working of the AHC algorithm can be explained using the below steps:

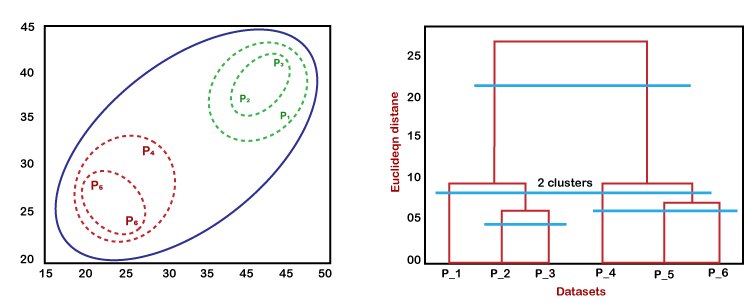
**Step-1:** Create each data point as a single cluster. Let's say there are N data points, so the number of clusters will also be N.  


* **Step-2:** Take two closest data points or clusters and merge them to form one cluster. So, there will now be N-1 clusters.  
  
* **Step-3**: Again, take the two closest clusters and merge them together to form one cluster. There will be N-2 clusters.  
  
* **Step-4:** Repeat Step 3 until only one cluster left. So, we will get the following clusters. Consider the below images:  
    
    
  
* **Step-5:** Once all the clusters are combined into one big cluster, develop the dendrogram to divide the clusters as per the problem.

### **Woking of Dendrogram in Hierarchical clustering**

The dendrogram is a tree-like structure that is mainly used to store each step as a memory that the HC algorithm performs. In the dendrogram plot, the Y-axis shows the Euclidean distances between the data points, and the x-axis shows all the data points of the given dataset.

The working of the dendrogram can be explained using the below diagram:



In the above diagram, the left part is showing how clusters are created in agglomerative clustering, and the right part is showing the corresponding dendrogram.

* As we have discussed above, firstly, the datapoints P2 and P3 combine together and form a cluster, correspondingly a dendrogram is created, which connects P2 and P3 with a rectangular shape. The hight is decided according to the Euclidean distance between the data points.
* In the next step, P5 and P6 form a cluster, and the corresponding dendrogram is created. It is higher than of previous, as the Euclidean distance between P5 and P6 is a little bit greater than the P2 and P3.
* Again, two new dendrograms are created that combine P1, P2, and P3 in one dendrogram, and P4, P5, and P6, in another dendrogram.
* At last, the final dendrogram is created that combines all the data points together.

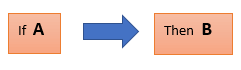
We can cut the dendrogram tree structure at any level as per our requirement.

1. **Explain Association Rule mining.**

* Association rule learning is a type of unsupervised learning technique that checks for the dependency of one data item on another data item and maps accordingly so that it can be more profitable.
* It tries to find some interesting relations or associations among the variables of dataset. It is based on different rules to discover the interesting relations between variables in the database.
* The association rule learning is one of the very important concepts of machine learning, and it is employed in Market Basket analysis, Web usage mining, continuous production, etc.
* Here market basket analysis is a technique used by the various big retailer to discover the associations between items.
* We can understand it by taking an example of a supermarket, as in a supermarket, all products that are purchased together are put together.
* For example, if a customer buys bread, he most likely can also buy butter, eggs, or milk, so these products are stored within a shelf or mostly nearby.  
    
  Association rule learning can be divided into three types of algorithms:

1. Apriori
2. Eclat
3. F-P Growth Algorithm

Association rule learning works on the concept of If and Else Statement, such as if A then B.



Here the If element is called **antecedent**, and then statement is called as **Consequent**. These types of relationships where we can find out some association or relation between two items is known *as single cardinality*. It is all about creating rules, and if the number of items increases, then cardinality also increases accordingly. So, to measure the associations between thousands of data items, there are several metrics. These metrics are given below:

* **Support**
* **Confidence**
* **Lift**

Support

Support is the frequency of A or how frequently an item appears in the dataset. It is defined as the fraction of the transaction T that contains the itemset X. If there are X datasets, then for transactions T, it can be written as:

Association Rule Learning

Confidence

Confidence indicates how often the rule has been found to be true. Or how often the items X and Y occur together in the dataset when the occurrence of X is already given. It is the ratio of the transaction that contains X and Y to the number of records that contain X.

Association Rule Learning

Lift

It is the strength of any rule, which can be defined as below formula:

Association Rule Learning

It is the ratio of the observed support measure and expected support if X and Y are independent of each other. It has three possible values:

* If **Lift= 1**: The probability of occurrence of antecedent and consequent is independent of each other.
* **Lift>1**: It determines the degree to which the two itemsets are dependent to each other.
* **Lift<1**: It tells us that one item is a substitute for other items, which means one item has a negative effect on another.

Types of Association Rule Lerning

Association rule learning can be divided into three algorithms:

Apriori Algorithm

This algorithm uses frequent datasets to generate association rules. It is designed to work on the databases that contain transactions. This algorithm uses a breadth-first search and Hash Tree to calculate the itemset efficiently.

It is mainly used for market basket analysis and helps to understand the products that can be bought together. It can also be used in the healthcare field to find drug reactions for patients.

Eclat Algorithm

Eclat algorithm stands for **Equivalence Class Transformation**. This algorithm uses a depth-first search technique to find frequent itemsets in a transaction database. It performs faster execution than Apriori Algorithm.

F-P Growth Algorithm

The F-P growth algorithm stands for **Frequent Pattern**, and it is the improved version of the Apriori Algorithm. It represents the database in the form of a tree structure that is known as a frequent pattern or tree. The purpose of this frequent tree is to extract the most frequent patterns.

Applications of Association Rule Learning

It has various applications in machine learning and data mining. Below are some popular applications of association rule learning:

* **Market Basket Analysis:** It is one of the popular examples and applications of association rule mining. This technique is commonly used by big retailers to determine the association between items.
* **Medical Diagnosis:** With the help of association rules, patients can be cured easily, as it helps in identifying the probability of illness for a particular disease.
* **Protein Sequence:** The association rules help in determining the synthesis of artificial Proteins.
* It is also used for the **Catalog Design** and **Loss-leader Analysis** and many more other applications.

Summary:

* **Association Rule Learning Overview:**
  + Type of unsupervised learning.
  + Checks dependency between data items.
  + Finds relations among variables in datasets.
  + Employed in Market Basket analysis, Web usage mining, etc.
  + Market Basket analysis example: Items purchased together in supermarkets.
* **Types of Association Rule Learning Algorithms:**
  + **Apriori:**
    - Uses frequent datasets for association rules.
    - Utilizes breadth-first search and Hash Tree.
    - Mainly for market basket analysis.
    - Also applicable in healthcare for finding drug reactions.
  + **Eclat:**
    - Stands for Equivalence Class Transformation.
    - Uses depth-first search to find frequent itemsets.
    - Faster execution compared to Apriori Algorithm.
  + **F-P Growth Algorithm:**
    - Stands for Frequent Pattern Growth.
    - Improved version of Apriori.
    - Represents the database as a frequent pattern tree.
    - Extracts the most frequent patterns.
* **How Association Rule Learning Works:**
  + Works on If-Else statements (e.g., if A then B).
  + If element = antecedent, Then statement = consequent.
  + Single cardinality relationships: association between two items.
  + Metrics for Measuring Associations:
    - **Support:** Frequency of an item in the dataset.
    - **Confidence:** Likelihood of finding X and Y together.
    - **Lift:** Strength of a rule; indicates dependency between itemsets.
* **Applications:**
  + **Market Basket Analysis:** Determines item associations in retail.
  + **Medical Diagnosis:** Helps identify illness probabilities.
  + **Protein Sequence:** Assists in artificial protein synthesis.
  + **Catalog Design, Loss-leader Analysis, etc.:** Additional applications of association rule learning.

1. **Explain apriori algorithm.**

**Apriori Algorithm in Machine Learning:**

* The Apriori algorithm uses frequent item sets to generate association rules, and it is designed to work on the databases that contain transactions.
* With the help of these association rule, it determines how strongly or how weakly two objects are connected.
* This algorithm uses a breadth-first search and Hash Tree to calculate the item set associations efficiently.
* It is the iterative process for finding the frequent item sets from the large dataset.

This algorithm was given by the R. Agrawal and Srikant in the year 1994. It is mainly used for market basket analysis and helps to find those products that can be bought together. It can also be used in the healthcare field to find drug reactions for patients.

To improve the efficiency of level-wise generation of frequent item sets, an important property is used called Apriori property which helps by reducing the search space.

Apriori Property –

All non-empty subset of frequent item set must be frequent. The key concept of Apriori algorithm is its anti-monotonicity of support measure. Apriori assumes that

All subsets of a frequent item set must be frequent(Apriori property).

If an itemset is infrequent, all its supersets will be infrequent.

**Steps for Apriori Algorithm**

Below are the steps for the apriori algorithm:

**Step-1:** Determine the support of item sets in the transactional database, and select the minimum support and confidence.

**Step-2:** Take all supports in the transaction with higher support value than the minimum or selected support value.

**Step-3:** Find all the rules of these subsets that have higher confidence value than the threshold or minimum confidence.

**Step-4:** Sort the rules as the decreasing order of lift.

## Components of Apriori algorithm

The given three components comprise the apriori algorithm.

1. Support
2. Confidence
3. Lift

Let's take an example to understand this concept.

We have already discussed above; you need a huge database containing a large no of transactions. Suppose you have 4000 customers transactions in a Big Bazar. You have to calculate the Support, Confidence, and Lift for two products, and you may say Biscuits and Chocolate. This is because customers frequently buy these two items together.

Out of 4000 transactions, 400 contain Biscuits, whereas 600 contain Chocolate, and these 600 transactions include a 200 that includes Biscuits and chocolates. Using this data, we will find out the support, confidence, and lift.

### **Support**

Support refers to the default popularity of any product. You find the support as a quotient of the division of the number of transactions comprising that product by the total number of transactions. Hence, we get

Support (Biscuits) = (Transactions relating biscuits) / (Total transactions)

= 400/4000 = 10 percent.

### **Confidence**

Confidence refers to the possibility that the customers bought both biscuits and chocolates together. So, you need to divide the number of transactions that comprise both biscuits and chocolates by the total number of transactions to get the confidence.

Hence,

Confidence = (Transactions relating both biscuits and Chocolate) / (Total transactions involving Biscuits)

= 200/400

= 50 percent.

It means that 50 percent of customers who bought biscuits bought chocolates also.

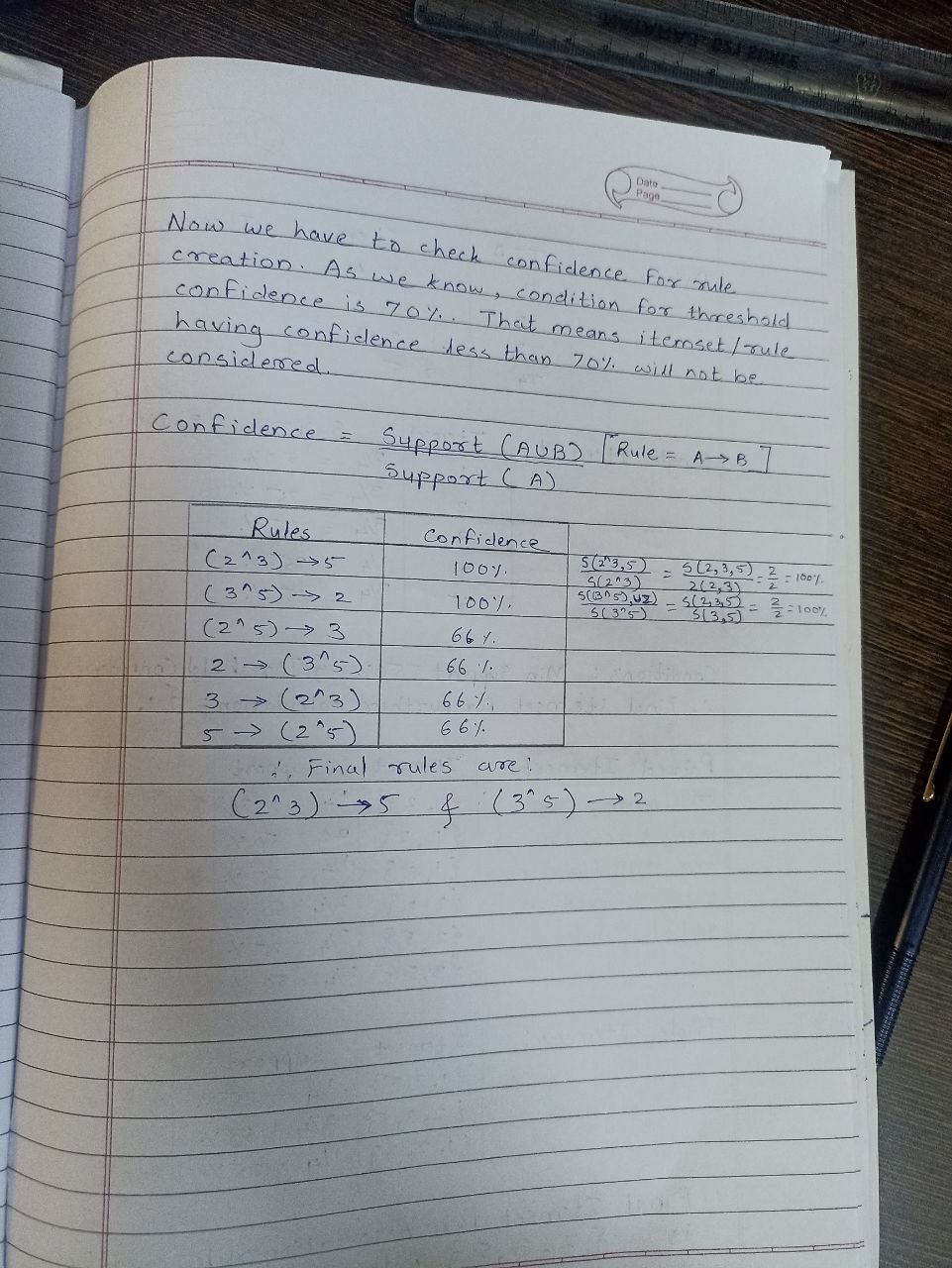
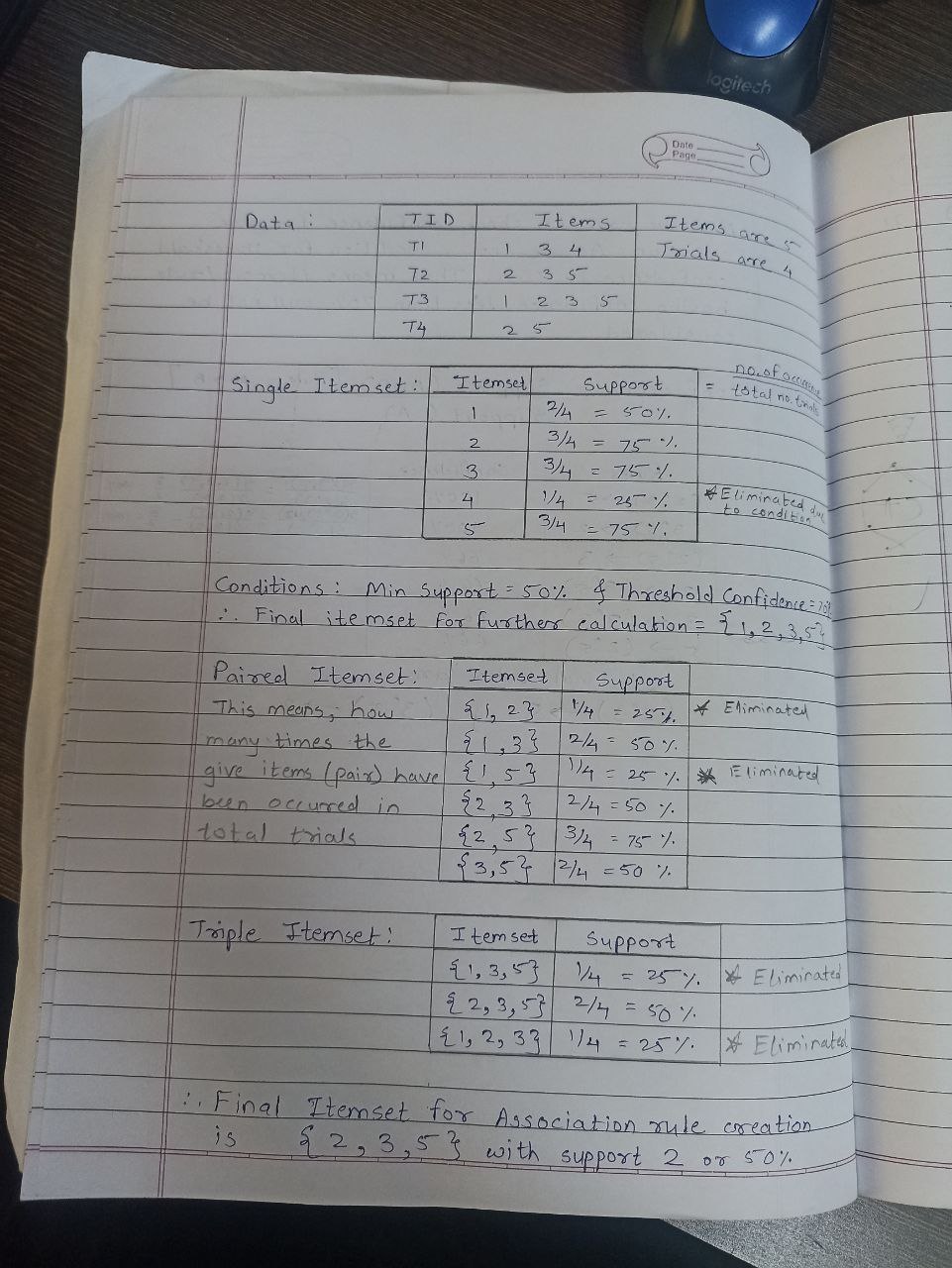
### **Lift**

Consider the above example; lift refers to the increase in the ratio of the sale of chocolates when you sell biscuits. The mathematical equations of lift are given below.

Lift = (Confidence (Biscuits - chocolates)/ (Support (Biscuits)

= 50/10 = 5

It means that the probability of people buying both biscuits and chocolates together is five times more than that of purchasing the biscuits alone. If the lift value is below one, it requires that the people are unlikely to buy both the items together. Larger the value, the better is the combination.

Example:  


Advantages of Apriori Algorithm

* This is easy to understand algorithm
* The join and prune steps of the algorithm can be easily implemented on large datasets.

Disadvantages of Apriori Algorithm

* The apriori algorithm works slow compared to other algorithms.
* The overall performance can be reduced as it scans the database for multiple times.
* The time complexity and space complexity of the apriori algorithm is O(2D), which is very high. Here D represents the horizontal width present in the database.

1. **Explain F-P Growth algorithm**

# **FP Growth Algorithm in Data Mining**

In Data Mining, finding frequent patterns in large databases is very important and has been studied on a large scale in the past few years. Unfortunately, this task is computationally expensive, especially when many patterns exist.

The FP-Growth Algorithm proposed by **Han in**. This is an efficient and scalable method for mining the complete set of frequent patterns by pattern fragment growth, using an extended prefix-tree structure for storing compressed and crucial information about frequent patterns named frequent-pattern tree (FP-tree). In his study, Han proved that his method outperforms other popular methods for mining frequent patterns, e.g. the Apriori Algorithm and the TreeProjection. In some later works, it was proved that FP-Growth performs better than other methods, including **Eclat** and **Relim**. The popularity and efficiency of the FP-Growth Algorithm contribute to many studies that propose variations to improve its performance.

### **What is FP Growth Algorithm?**

The FP-Growth Algorithm is an alternative way to find frequent item sets without using candidate generations, thus improving performance. For so much, it uses a divide-and-conquer strategy. The core of this method is the usage of a special data structure named frequent-pattern tree (FP-tree), which retains the item set association information.

**This algorithm works as follows:**

ADVERTISEMENT

ADVERTISEMENT

* First, it compresses the input database creating an FP-tree instance to represent frequent items.
* After this first step, it divides the compressed database into a set of conditional databases, each associated with one frequent pattern.
* Finally, each such database is mined separately.

Using this strategy, the FP-Growth reduces the search costs by recursively looking for short patterns and then concatenating them into the long frequent patterns.

In large databases, holding the FP tree in the main memory is impossible. A strategy to cope with this problem is to partition the database into a set of smaller databases (called projected databases) and then construct an FP-tree from each of these smaller databases.

### **FP-Tree**

The frequent-pattern tree (FP-tree) is a compact data structure that stores quantitative information about frequent patterns in a database. Each transaction is read and then mapped onto a path in the FP-tree. This is done until all transactions have been read. Different transactions with common subsets allow the tree to remain compact because their paths overlap.

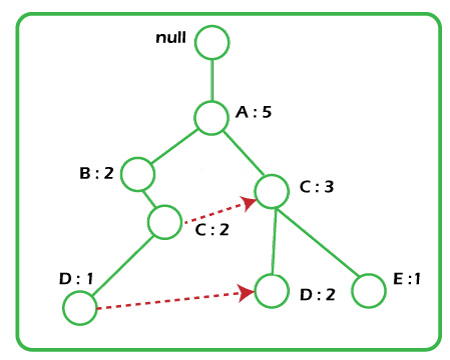
A frequent Pattern Tree is made with the initial item sets of the database. The purpose of the FP tree is to mine the most frequent pattern. Each node of the FP tree represents an item of the item set.

The root node represents null, while the lower nodes represent the item sets. The associations of the nodes with the lower nodes, that is, the item sets with the other item sets, are maintained while forming the tree.

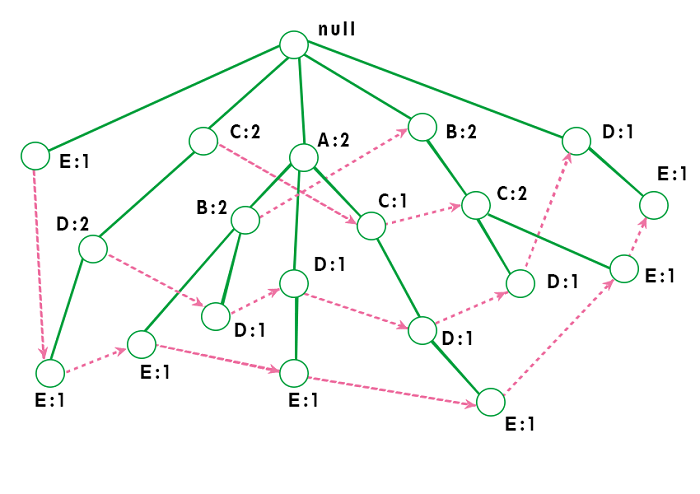
Han defines the FP-tree as the tree structure given below:

1. One root is labelled as "null" with a set of item-prefix subtrees as children and a frequent-item-header table.
2. Each node in the item-prefix subtree consists of three fields:
   * Item-name: registers which item is represented by the node;
   * Count: the number of transactions represented by the portion of the path reaching the node;
   * Node-link: links to the next node in the FP-tree carrying the same item name or null if there is none.
3. Each entry in the frequent-item-header table consists of two fields:
   * Item-name: as the same to the node;
   * Head of node-link: a pointer to the first node in the FP-tree carrying the item name.

Additionally, the frequent-item-header table can have the count support for an item. The below diagram is an example of a best-case scenario that occurs when all transactions have the same itemset; the size of the FP-tree will be only a single branch of nodes.



The worst-case scenario occurs when every transaction has a unique item set. So the space needed to store the tree is greater than the space used to store the original data set because the FP-tree requires additional space to store pointers between nodes and the counters for each item. The diagram below shows how a worst-case scenario FP-tree might appear. As you can see, the tree's complexity grows with each transaction's uniqueness.



9. Explain Eclat algorithm

10. Explain working of dendrogram in Hierarchical clustering.