

Machine Learning (6CS4-02)

Unit-2 Notes

Vision of the Institute

To become a renowned center of outcome based learning and work towards academic, professional, cultural and social enrichment of the lives of individuals and communities.

Mission of the Institute

M1- Focus on evaluation of learning outcomes and motivate students to inculcate research aptitude by project based learning.

M2- Identify, based on informed perception of Indian, regional and global needs, the areas of focus and provide platform to gain knowledge and solutions.

M3- Offer opportunities for interaction between academia and industry.

M4- Develop human potential to its fullest extent so that intellectually capable and imaginatively gifted leaders can emerge in a range of professions.

Vision of the Department

To become renowned Centre of excellence in computer science and engineering and make competent engineers & professionals with high ethical values prepared for lifelong learning.

Mission of the Department

M1-To impart outcome based education for emerging technologies in the field of computer science and engineering.

M2-To provide opportunities for interaction between academia and industry.

M3- To provide platform for lifelong learning by accepting the change in technologies

M4- To develop aptitude of fulfilling social responsibilities.

Program Outcomes (PO)

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Program Educational Objectives (PEO)

1. To provide students with the fundamentals of Engineering Sciences with more emphasis in **Computer Science &Engineering** by way of analyzing and exploiting engineering challenges.
2. To train students with good scientific and engineering knowledge so as to comprehend, analyze, design, and create novel products and solutions for the real life problems.
3. To inculcate professional and ethical attitude, effective communication skills, teamwork skills, multidisciplinary approach, entrepreneurial thinking and an ability to relate engineering issues with social issues.
4. To provide students with an academic environment aware of excellence, leadership, written ethical codes and guidelines, and the self-motivated life-long learning needed for a successful professional career.
5. To prepare students to excel in Industry and Higher education by Educating Students along with High moral values and Knowledge

Program Specific Outcomes (PSO)

PSO1: Ability to interpret and analyze network specific and cyber security issues, automation in real word environment.

PSO2: Ability to Design and Develop Mobile and Web-based applications under realistic constraints.

Course Outcome:

CO1: Understand the concept of machine learning and apply supervised learning techniques.

CO2: Illustrate various unsupervised learning algorithm for clustering, and market basket analysis.

CO3: Analyze statistical learning theory for dimension reduction and model evaluation in machine learning.

CO4: Apply the concept of semi supervised learning, reinforcement learning and recommendation system.

CO-PO Mapping:

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
Understand the concept of machine learning and apply supervised learning techniques.	3	3	3	3	2	1	1	1	1	2	1	3
Illustrate various unsupervised learning algorithm for clustering, and market basket analysis.	3	3	3	2	2	1	1	1	1	1	1	3
Analyze statistical learning theory for dimension reduction and model evaluation in machine learning.	3	3	3	3	2	2	2	2	1	2	2	3
Apply the concept of semi supervised learning, reinforcement learning and recommendation system.	3	3	3	3	2	1	1	1	1	2	1	3

SYLLABUS:



RAJASTHAN TECHNICAL UNIVERSITY, KOTA

Syllabus

III Year-VI Semester: B.Tech. Computer Science and Engineering

6CS4-02:Machine Learning

Credit: 3

Max. Marks: 150(IA:30, ETE:120)

3L+0T+0P

End Term Exam: 3 Hours

SN	Contents	Hours
1	Introduction: Objective, scope and outcome of the course.	01
2	Supervised learning algorithm: Introduction, types of learning, application, Supervised learning: Linear Regression Model, Naive Bayes classifier Decision Tree, K nearest neighbor, Logistic Regression, Support Vector Machine, Random forest algorithm	09
3	Unsupervised learning algorithm: Grouping unlabelled items using k-means clustering, Hierarchical Clustering, Probabilistic clustering, Association rule mining, Apriori Algorithm, f-p growth algorithm, Gaussian mixture model.	08
4	Introduction to Statistical Learning Theory, Feature extraction - Principal component analysis, Singular value decomposition. Feature selection - feature ranking and subset selection, filter, wrapper and embedded methods, Evaluating Machine Learning algorithms and Model Selection.	08
5	Semi supervised learning, Reinforcement learning: Markov decision process (MDP), Bellman equations, policy evaluation using Monte Carlo, Policy iteration and Value iteration, Q-Learning, State-Action-Reward-State-Action (SARSA), Model-based Reinforcement Learning.	08
6	Recommended system, Collaborative filtering, Content-based filtering Artificial neural network, Perceptron, Multilayer network, Backpropagation, Introduction to Deep learning.	08
	Total	42

LECTURE PLAN:

Unit No./ Total Lecture Reqd.	Topics	Lect. Reqd.	Lect. No.
Unit-I (10)	1. Introduction to subject and scope	1	1
	2. Introduction to learning, Types of learning and Applications	1	2
	3. Supervised Learning	1	3
	4. Linear Regression Model	1	4
	5. Naïve Bayes Classifier	1	5
	6. Decision Tree	1	6
	7. K-nearest Neighbor	1	7
	8. Logistic Regression	1	8
	9. Support Vector Machine	1	9
	10. Random Forest Algorithm	1	10
BC-1	Logistic Regression Model	1	11
Unit-II (8)	1. Introduction to clustering, K-mean clustering	2	12
	2. Hierarchical Clustering	1	14
	3. Probabilistic Clustering	1	15
	4. Association Rule Mining	1	16
	5. Apriori Algorithm	1	17
	6. f-p Growth Algorithm	1	18
	7. Gaussian Mixture Model	1	19
Unit-III (8)	1. Feature Extraction- PCA and SVD	3	22
	2. Feature Selection- Feature Ranking and Subset Selection	2	24
	3. Filter, Wrapper and Embedded Methods	1	25
	4. Evaluating Machine Learning Algorithms	1	26
	5. Evaluating Model Selection	1	27
Unit- IV (8)	1. Semi supervised learning: Markov Decision Process (MDP)	2	29
	2. Bellman Equations	1	30
	3. Policy Evaluation using Monte Carlo	1	31
	4. Policy iteration and Value iteration	1	32
	5. Q-Learning	1	33
	6. State-Action-Reward-State-Action (SARSA)	1	34
	7. Model-based Reinforcement Learning	1	35
Unit- V	1. Recommendation system: Collaborative Filtering	1	36

(8)	2. Content based filtering	1	37
	3. Artificial neural network	1	38
	4. Perceptron	1	39
	5. Multilayer network	1	40
	6. Backpropagation	1	41
	7. Introduction to Deep learning.	2	42
BC-2	Genetic Algorithms	1	44

Text Book:

Machine learning- Tom M Mitchell

What is K-Means Clustering?

K-Means clustering is an unsupervised learning algorithm. There is no labeled data for this clustering, unlike in supervised learning. K-Means performs division of objects into clusters that share similarities and are dissimilar to the objects belonging to another cluster.

The term 'K' is a number. You need to tell the system how many clusters you need to create. For example, $K = 2$ refers to two clusters. There is a way of finding out what is the best or optimum value of K for a given data.

For a better understanding of k-means, let's take an example from cricket. Imagine you received data on a lot of cricket players from all over the world, which gives information on the runs scored by the player and the wickets taken by them in the last ten matches. Based on this information, we need to group the data into two clusters, namely batsman and bowlers.

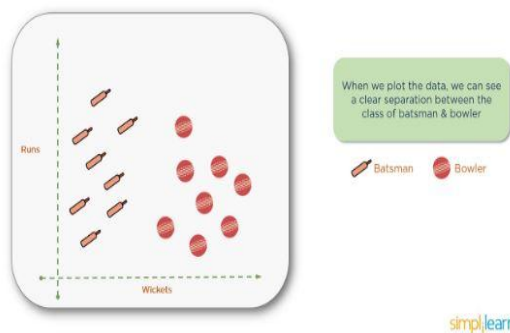
Let's take a look at the steps to create these clusters.

Solution:

Assign data points

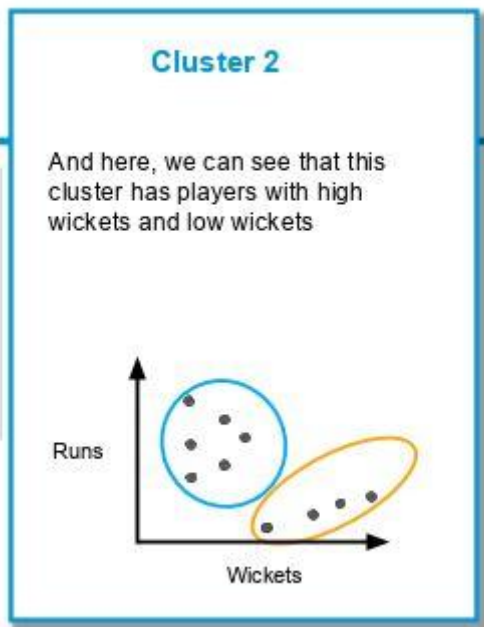
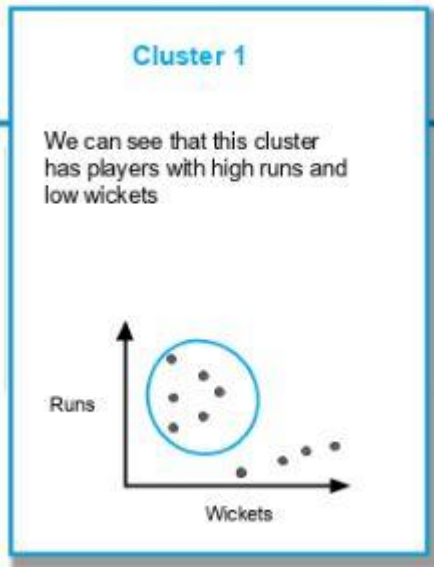
Here, we have our data set plotted on 'x' and 'y' coordinates. The information on the y-axis is about the runs scored, and on the x-axis about the wickets taken by the players.

If we plot the data, this is how it would look:



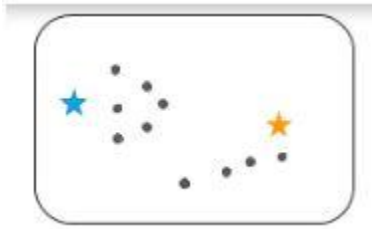
Perform Clustering

We need to create the clusters, as shown below:

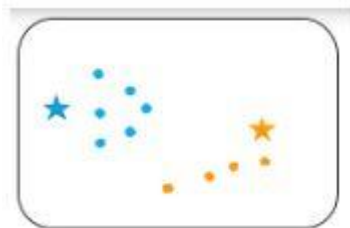


Considering the same data set, let us solve the problem using K-Means clustering (taking $K = 2$).

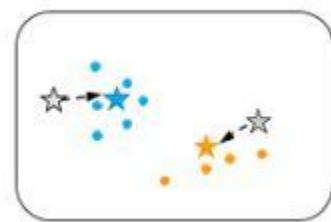
The first step in k-means clustering is the allocation of two centroids randomly (as $K=2$). Two points are assigned as centroids. Note that the points can be anywhere, as they are random points. They are called centroids, but initially, they are not the central point of a given data set.



The next step is to determine the distance between each of the data points from the randomly assigned centroids. For every point, the distance is measured from both the centroids, and whichever distance is less, that point is assigned to that centroid. You can see the data points attached to the centroids and represented here in blue and yellow.



The next step is to determine the actual centroid for these two clusters. The original randomly allocated centroid is to be repositioned to the actual centroid of the clusters.



This process of calculating the distance and repositioning the centroid continues until we obtain our final cluster. Then the centroid repositioning stops.



As seen above, the centroid doesn't need anymore repositioning, and it means the algorithm has converged, and we have the two clusters with a centroid.

Applications of K-Means Clustering

K-Means clustering is used in a variety of examples or business cases in real life, like:

- Academic performance
- Diagnostic systems
- Search engines
- Wireless sensor networks

Academic Performance

Based on the scores, students are categorized into grades like A, B, or C.

Diagnostic systems

The medical profession uses k-means in creating smarter medical decision support systems, especially in the treatment of liver ailments.

Search engines

Clustering forms a backbone of search engines. When a search is performed, the search results need to be grouped, and the search engines very often use clustering to do this.

Wireless sensor networks

The clustering algorithm plays the role of finding the cluster heads, which collect all the data in its respective cluster.

Distance Measure

Distance measure determines the similarity between two elements and influences the shape of clusters.

K-Means clustering supports various kinds of distance measures, such as:

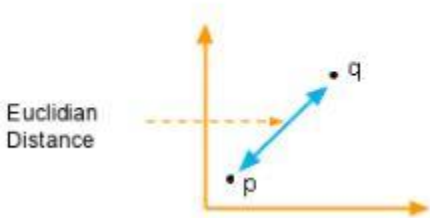
- Euclidean distance measure
- Manhattan distance measure

- A squared euclidean distance measure
- Cosine distance measure

Euclidean Distance Measure

The most common case is determining the distance between two points. If we have a point P and point Q, the euclidean distance is an ordinary straight line. It is the distance between the two points in Euclidean space.

The formula for distance between two points is shown below:

$$d = \sqrt{\sum_{i=1}^n (q_i - p_i)^2}$$


Squared Euclidean Distance Measure

This is identical to the Euclidean distance measurement but does not take the square root at the end. The formula is shown below:

$$d = \sum_{i=1}^n (q_i - p_i)^2$$

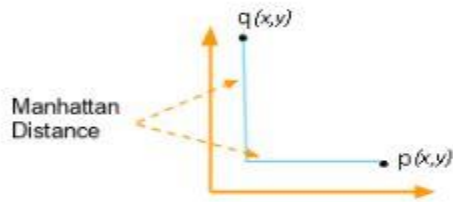
Manhattan Distance Measure

The Manhattan distance is the simple sum of the horizontal and vertical components or the distance between two points measured along axes at right angles.

Note that we are taking the absolute value so that the negative values don't come into play.

The formula is shown below:

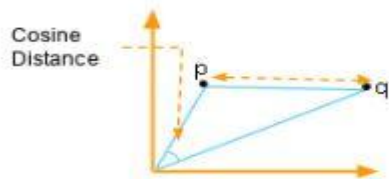
$$d = \sum_{i=1}^n |q_x - p_x| + |q_y - p_y|$$



Cosine Distance Measure

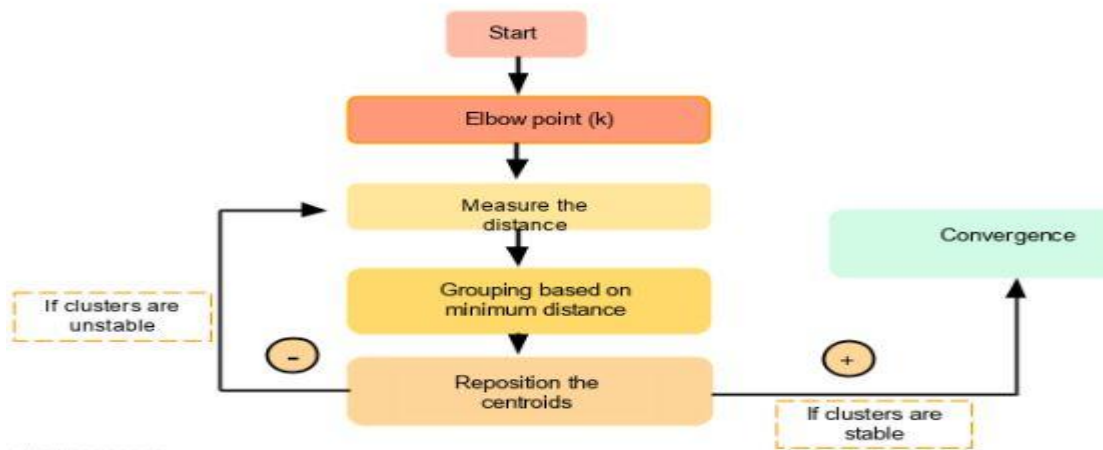
In this case, we take the angle between the two vectors formed by joining the points from the origin. The formula is shown below:

$$d = \frac{\sum_{i=0}^{n-1} q_i - p_x}{\sum_{i=0}^{n-1} (q_i)^2 \times \sum_{i=0}^{n-1} (p_i)^2}$$



How Does K-Means Clustering Work?

The flowchart below shows how k-means clustering works:



The goal of the K-Means algorithm is to find clusters in the given input data. There are a couple of ways to accomplish this. We can use the trial and error method by specifying the value of K (e.g., 3,4, 5). As we progress, we keep changing the value until we get the best clusters.

Another method is to use the Elbow technique to determine the value of K. Once we get the value of K, the system will assign that many centroids randomly and measure the distance of each of the data points from these centroids. Accordingly, it assigns those points to the corresponding centroid from which the distance is minimum. So each data point will be assigned to the centroid, which is closest to it. Thereby we have a K number of initial clusters.

For the newly formed clusters, it calculates the new centroid position. The position of the centroid moves compared to the randomly allocated one.

Once again, the distance of each point is measured from this new centroid point. If required,

the data points are relocated to the new centroids, and the mean position or the new centroid is calculated once again.

If the centroid moves, the iteration continues indicating no convergence. But once the centroid stops moving (which means that the clustering process has converged), it will reflect the result.

Let's use a visualization example to understand this better.

Post Graduate Program in AI and Machine Learning

In Partnership with Purdue University [EXPLORE COURSE](#)

We have a data set for a grocery shop, and we want to find out how many clusters this has to be spread across. To find the optimum number of clusters, we break it down into the following steps:

Step 1:

The Elbow method is the best way to find the number of clusters. The elbow method constitutes running K-Means clustering on the dataset.

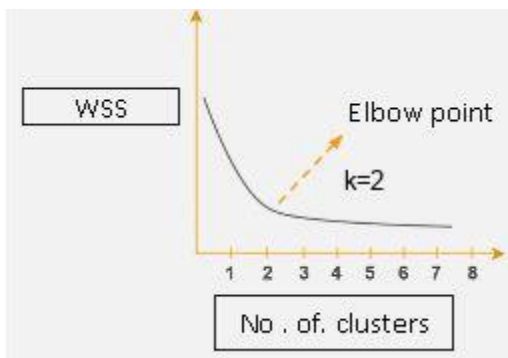
Next, we use within-sum-of-squares as a measure to find the optimum number of clusters that can be formed for a given data set. Within the sum of squares (WSS) is defined as the sum of the squared distance between each member of the cluster and its centroid.

$$WSS = \sum_{i=1}^m (x_i - c_i)^2$$

Where x_i = data point and c_i = closest point to centroid

The WSS is measured for each value of K. The value of K, which has the least amount of WSS, is taken as the optimum value.

Now, we draw a curve between WSS and the number of clusters.



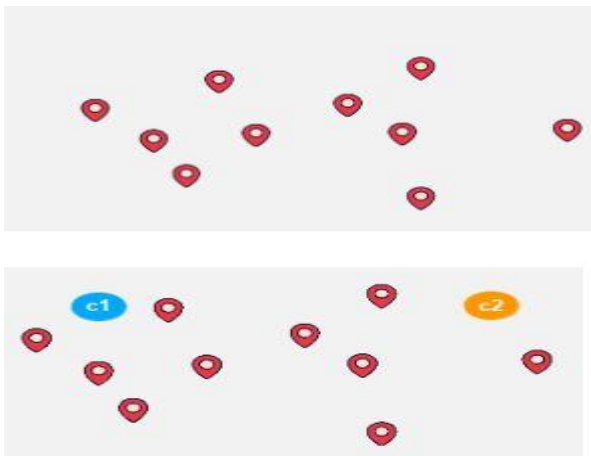
Here, WSS is on the y-axis and number of clusters on the x-axis.

You can see that there is a very gradual change in the value of WSS as the K value increases from 2.

So, you can take the elbow point value as the optimal value of K. It should be either two, three, or at most four. But, beyond that, increasing the number of clusters does not dramatically change the value in WSS, it gets stabilized.

Step 2:

Let's assume that these are our delivery points:



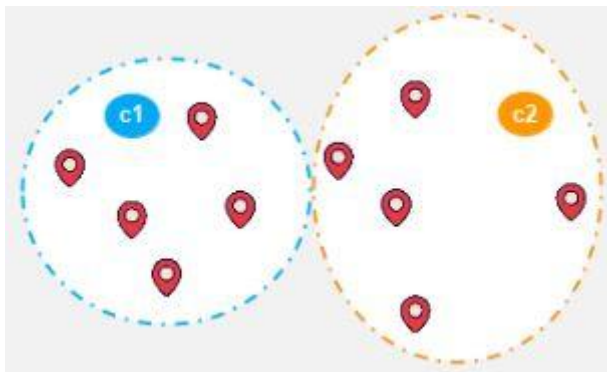
We can randomly initialize two points called the cluster centroids.

Here, C1 and C2 are the centroids assigned randomly.

Step 3:

Now the distance of each location from the centroid is measured, and each data point is assigned to the centroid, which is closest to it.

This is how the initial grouping is done:

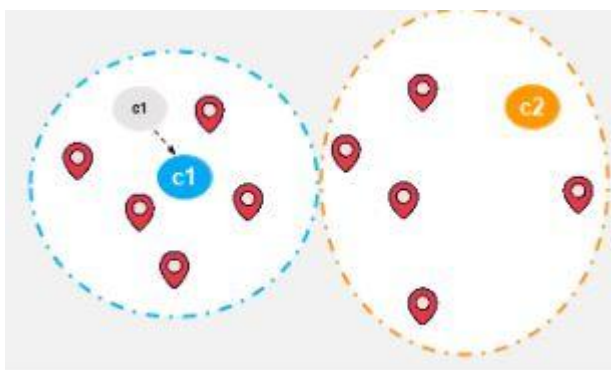


Step 4:

Compute the actual centroid of data points for the first group.

Step 5:

Reposition the random centroid to the actual centroid.

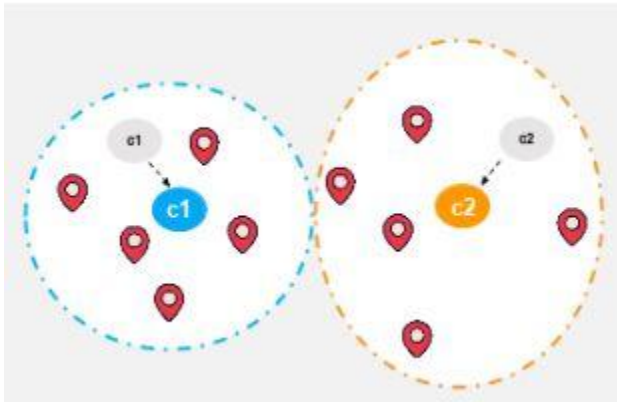


Step 6:

Compute the actual centroid of data points for the second group.

Step 7:

Reposition the random centroid to the actual centroid.



Step 8:

Once the cluster becomes static, the k-means algorithm is said to be converged.

The final cluster with centroids c1 and c2 is as shown below:



K-Means Clustering Algorithm

Let's say we have $x_1, x_2, x_3, \dots, x(n)$ as our inputs, and we want to split this into K clusters.

The steps to form clusters are:

Step 1: Choose K random points as cluster centers called centroids.

Step 2: Assign each $x(i)$ to the closest cluster by implementing euclidean distance (i.e., calculating its distance to each centroid)

Step 3: Identify new centroids by taking the average of the assigned points.

Step 4: Keep repeating step 2 and step 3 until convergence is achieved

Let's take a detailed look at it at each of these steps.

Step 1:

We randomly pick K (centroids). We name them c_1, c_2, \dots, c_k , and we can say that

$$C = \{c_1, c_2, \dots, c_k\}$$

Where C is the set of all centroids.

Step 2:

We assign each data point to its nearest center, which is accomplished by calculating the euclidean distance.

$$\arg \min_{c_i \in C} \text{dist}(c_i, x)^2$$

Where $\text{dist}()$ is the Euclidean distance.

Here, we calculate the distance of each x value from each c value, i.e. the distance between x_1-c_1 , x_1-c_2 , x_1-c_3 , and so on. Then we find which is the lowest value and assign x_1 to that particular centroid.

Similarly, we find the minimum distance for x_2 , x_3 , etc.

Step 3:

We identify the actual centroid by taking the average of all the points assigned to that cluster.

$$c_i = \frac{1}{|S_i|} \sum_{x_j \in S_i} x_j$$

Where S_i is the set of all points assigned to the i th cluster.

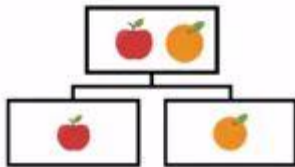
It means the original point, which we thought was the centroid, will shift to the new position, which is the actual centroid for each of these groups.

Step 4:

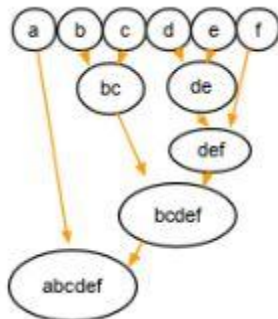
Keep repeating step 2 and step 3 until convergence is achieved.

Hierarchical Clustering

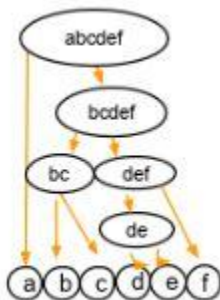
Hierarchical clustering uses a tree-like structure, like so:



In agglomerative clustering, there is a bottom-up approach. We begin with each element as a separate cluster and merge them into successively more massive clusters, as shown below:



Divisive clustering is a top-down approach. We begin with the whole set and proceed to divide it into successively smaller clusters.



How K-means works:

1. Decide the number of clusters (k)
2. Select k random points from the data as centroids
3. Assign all the points to the nearest cluster centroid
4. Calculate the centroid of newly formed clusters
5. Repeat steps 3 and 4

It is an iterative process. It will keep on running until the centroids of newly formed clusters do not change or the maximum number of iterations are reached.

But there are certain challenges with K-means. It always tries to make clusters of the same size. Also, we have to decide the number of clusters at the *beginning* of the algorithm. Ideally, we would not know how many clusters should we have, in the beginning of the algorithm and hence it a challenge with K-means.

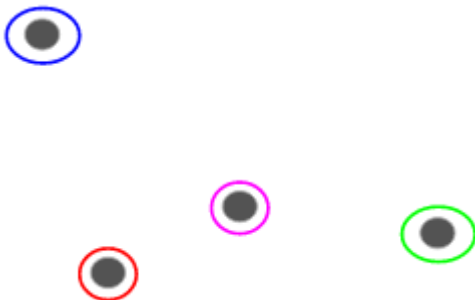
This is a gap hierarchical clustering bridges with aplomb. It takes away the problem of having to pre-define the number of clusters. Sounds like a dream! So, let's see what hierarchical clustering is and how it improves on K-means.

What is Hierarchical Clustering?

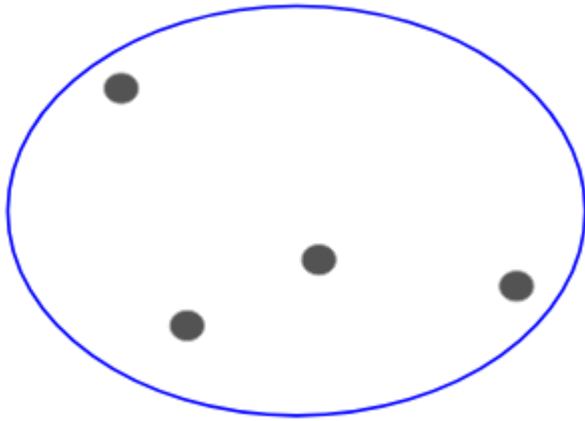
Let's say we have the below points and we want to cluster them into groups:



We can assign each of these points to a separate cluster:



Now, based on the similarity of these clusters, we can combine the most similar clusters together and repeat this process until only a single cluster is left:



We are essentially building a hierarchy of clusters. That's why this algorithm is called hierarchical clustering. I will discuss how to decide the number of clusters in a later section. For now, let's look at the different types of hierarchical clustering.

Types of Hierarchical Clustering

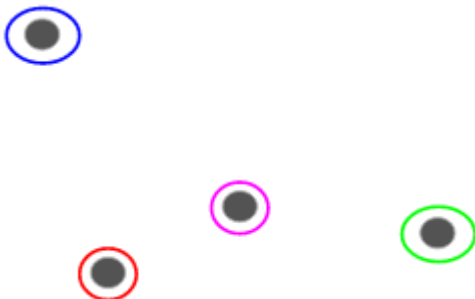
There are mainly two types of hierarchical clustering:

1. Agglomerative hierarchical clustering
2. Divisive Hierarchical clustering

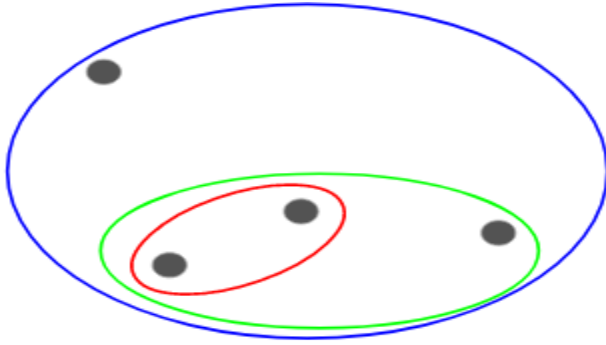
Let's understand each type in detail.

Agglomerative Hierarchical Clustering

We assign each point to an individual cluster in this technique. Suppose there are 4 data points. We will assign each of these points to a cluster and hence will have 4 clusters in the beginning:



Then, at each iteration, we merge the closest pair of clusters and repeat this step until only a single cluster is left:

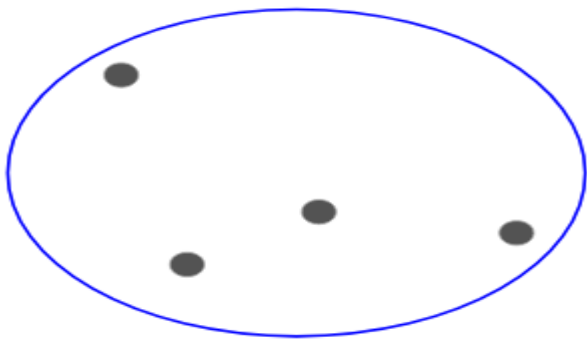


We are merging (or adding) the clusters at each step, right? Hence, this type of clustering is also known as **additive hierarchical clustering**.

Divisive Hierarchical Clustering

Divisive hierarchical clustering works in the opposite way. Instead of starting with n clusters (in case of n observations), we start with a single cluster and assign all the points to that cluster.

So, it doesn't matter if we have 10 or 1000 data points. All these points will belong to the same cluster at the beginning:



Now, at each iteration, we split the farthest point in the cluster and repeat this process until each cluster only contains a single point:



We are splitting (or dividing) the clusters at each step, hence the name divisive hierarchical clustering.

Agglomerative Clustering is widely used in the industry and that will be the focus in this article. Divisive hierarchical clustering will be a piece of cake once we have a handle on the agglomerative type.

Steps to Perform Hierarchical Clustering

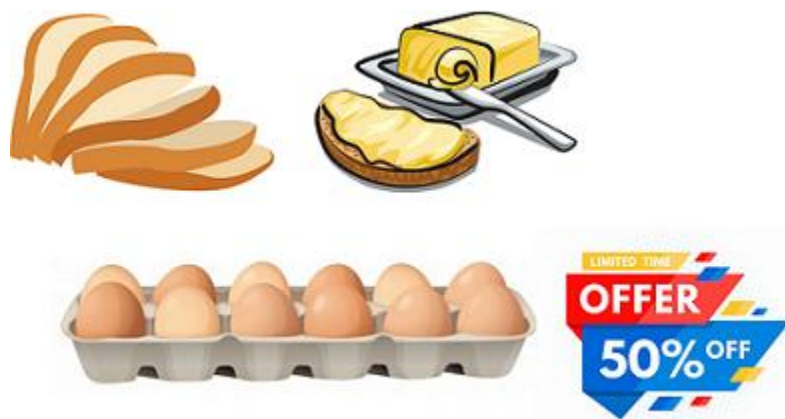
We merge the most similar points or clusters in hierarchical clustering – we know this. Now the question is – how do we decide which points are similar and which are not? It's one of the most important questions in clustering!

Here's one way to calculate similarity – Take the distance between the centroids of these clusters. The points having the least distance are referred to as similar points and we can merge them. We can refer to this as a **distance-based algorithm** as well (since we are calculating the distances between the clusters).

In hierarchical clustering, we have a concept called a proximity matrix. This stores the distances between each point. Let's take an example to understand this matrix as well as the steps to perform hierarchical clustering.

Market Basket Analysis is one of the key techniques used by large retailers to uncover associations between items. They try to find out associations between different items and products that can be sold together, which gives assisting in right product placement. Typically, it figures out what products are being bought together and organizations can place products in a similar manner. Let's understand this better with an example:

People who buy Bread usually buy Butter too. The Marketing teams at retail stores should target customers who buy bread and butter and provide an offer to them so that they buy the third item, like eggs.



So if customers buy bread and butter and see a discount or an offer on eggs, they will be encouraged to spend more and buy the eggs. This is what market basket analysis is all about.

This is just a small example. So, if you take 10000 items data of your Supermart to a Data Scientist, Just imagine the number of insights you can get. And that is why Association Rule mining is so important.

Association Rule Mining

Association rules can be thought of as an IF-THEN relationship. Suppose item **A** is being bought by the customer, then the chances of item **B** being picked by the customer too under the same **Transaction ID** is found out.



There are two elements of these rules:

Antecedent (IF): This is an item/group of items that are typically found in the Itemsets or Datasets.

Consequent (THEN): This comes along as an item with an Antecedent/group of Antecedents.

But here comes a constraint. Suppose you made a rule about an item, you still have around 9999 items to consider for rule-making. This is where the Apriori Algorithm comes into play. So before we understand the Apriori Algorithm, let's understand the math behind it. There are 3 ways to measure association:

- Support
- Confidence
- Lift

Support: It gives the fraction of transactions which contains item A and B. Basically Support tells us about the frequently bought items or the combination of items bought frequently.

$$Support = \frac{freq(A, B)}{N}$$

So with this, we can **filter out** the items that have a **low frequency**.

Confidence: It tells us how often the items A and B occur together, given the number times A occurs.

$$\text{Confidence} = \frac{\text{freq}(A, B)}{\text{freq}(A)}$$

Typically, when you work with the Apriori Algorithm, you define these terms accordingly. **But how do you decide the value?** Honestly, there isn't a way to define these terms. Suppose you've assigned the support value as 2. What this means is, until and unless the item/s frequency is not 2%, you will not consider that item/s for the Apriori algorithm. This makes sense as considering items that are bought less frequently is a waste of time.

Now suppose, after filtering you still have around 5000 items left. Creating association rules for them is a practically impossible task for anyone. This is where the concept of lift comes into play.

Lift: Lift indicates the strength of a rule over the random occurrence of A and B. It basically tells us the strength of any rule.

$$\text{Lift} = \frac{\text{Support}}{\text{Supp}(A) \times \text{Supp}(B)}$$

Focus on the denominator, it is the probability of the individual support values of A and B and not together. Lift explains the strength of a rule. **More the Lift more is the strength.** Let's say for A → B, the lift value is 4. It means that if you buy A the chances of buying B is **4 times**. Let's get started with the Apriori Algorithm now and see how it works.

Apriori Algorithm

Apriori algorithm uses frequent itemsets to generate association rules. It is based on the concept that a subset of a frequent itemset must also be a frequent itemset. Frequent Itemset is an itemset whose support value is greater than a threshold value(support).



we have the following data of a store.

TID	Items
T1	1 3 4
T2	2 3 5
T3	1 2 3 5
T4	2 5
T5	1 3 5

Iteration 1: Let's assume the support value is 2 and create the item sets of the size of 1 and calculate their support values.

TID	Items
T1	1 3 4
T2	2 3 5
T3	1 2 3 5
T4	2 5
T5	1 3 5

→

Itemset	Support
{1}	3
{2}	3
{3}	4
{4}	1
{5}	4

item 4 has a support value of 1 which is less than the min support value. So we are going to **discard {4}** in the upcoming iterations. We have the final Table F1.

Itemset	Support
{1}	3
{2}	3
{3}	4
{4}	1
{5}	4

→

Itemset	Support
{1}	3
{2}	3
{3}	4
{5}	4

Iteration 2: Next we will create itemsets of size 2 and calculate their support values. All the combinations of items set in F1 are used in this iteration.

TID	Items
T1	1 3 4
T2	2 3 5
T3	1 2 3 5
T4	2 5
T5	1 3 5

→

Itemset	Support
{1,2}	1
{1,3}	3
{1,5}	2
{2,3}	2
{2,5}	3
{3,5}	3

→

Itemset	Support
{1,3}	3
{1,5}	2
{2,3}	2
{2,5}	3
{3,5}	3

Itemsets having Support less than 2 are eliminated again. In this case $\{1,2\}$. Now, Let's understand what is pruning and how it makes Apriori one of the best algorithm for finding frequent itemsets.

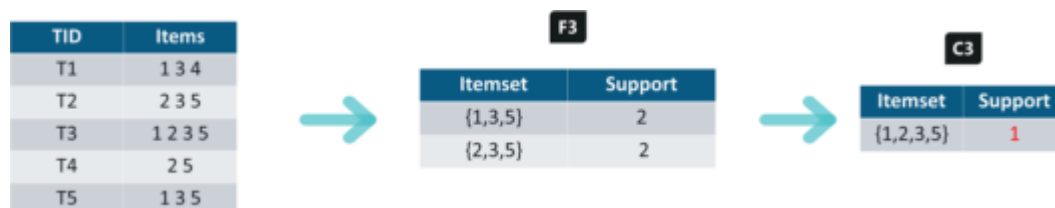
Pruning: We are going to divide the itemsets in C3 into subsets and eliminate the subsets that are having a support value less than 2.



Iteration 3: We will discard $\{1,2,3\}$ and $\{1,2,5\}$ as they both contain $\{1,2\}$. This is the main highlight of the Apriori Algorithm.



Iteration 4: Using sets of F3 we will create C4.



Since the Support of this itemset is less than 2, we will stop here and the final itemset we will have is **F3**.

Note: Till now we haven't calculated the confidence values yet.

With F3 we get the following itemsets:

For $I = \{1,3,5\}$, subsets are $\{1,3\}, \{1,5\}, \{3,5\}, \{1\}, \{3\}, \{5\}$
For $I = \{2,3,5\}$, subsets are $\{2,3\}, \{2,5\}, \{3,5\}, \{2\}, \{3\}, \{5\}$

Applying Rules: We will create rules and apply them on itemset F3. Now let's assume a minimum confidence value is **60%**.

For every subsets S of I, you output the rule

- $S \rightarrow (I-S)$ (means S recommends I-S)
- if **support(I) / support(S) \geq min_conf value**

$\{1,3,5\}$

Rule 1: $\{1,3\} \rightarrow (\{1,3,5\} - \{1,3\})$ means 1 & 3 \rightarrow 5

Confidence = $\text{support}(1,3,5) / \text{support}(1,3) = 2/3 = 66.66\% > 60\%$

Hence Rule 1 is **Selected**

Rule 2: $\{1,5\} \rightarrow (\{1,3,5\} - \{1,5\})$ means 1 & 5 \rightarrow 3

Confidence = $\text{support}(1,3,5) / \text{support}(1,5) = 2/2 = 100\% > 60\%$

Frequent Pattern Growth Algorithm

This algorithm is an improvement to the Apriori method. A frequent pattern is generated without the need for candidate generation. FP growth algorithm represents the database in the form of a tree called a frequent pattern tree or FP tree.

This tree structure will maintain the association between the itemsets. The database is fragmented using one frequent item. This fragmented part is called "pattern fragment". The itemsets of these fragmented patterns are analyzed. Thus with this method, the search for frequent itemsets is reduced comparatively.

FP Tree

Frequent Pattern Tree is a tree-like structure that is made with the initial itemsets 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 itemset.

The root node represents null while the lower nodes represent the itemsets. The association of the nodes with the lower nodes that is the itemsets with the other itemsets are maintained while forming the tree.

Frequent Pattern Algorithm Steps

The frequent pattern growth method lets us find the frequent pattern without candidate generation.

Let us see the steps followed to mine the frequent pattern using frequent pattern growth algorithm:

- 1) The first step is to scan the database to find the occurrences of the itemsets in the database. This step is the same as the first step of Apriori. The count of 1-itemsets in the database is called support count or frequency of 1-itemset.
- 2) The second step is to construct the FP tree. For this, create the root of the tree. The root is represented by null.
- 3) The next step is to scan the database again and examine the transactions. Examine the first transaction and find out the itemset in it. The itemset with the max count is taken at the top, the next itemset with lower count and so on. It means that the branch of the tree is constructed with transaction itemsets in descending order of count.
- 4) The next transaction in the database is examined. The itemsets are ordered in descending order of count. If any itemset of this transaction is already present in another branch (for example in the 1st transaction), then this transaction branch would share a common prefix to the root. This means that the common itemset is linked to the new node of another itemset in this transaction.
- 5) Also, the count of the itemset is incremented as it occurs in the transactions. Both the common node and new node count is increased by 1 as they are created and linked according to transactions.
- 6) The next step is to mine the created FP Tree. For this, the lowest node is examined first along with the links of the lowest nodes. The lowest node represents the frequency pattern length 1. From this, traverse the path in the FP Tree. This path or paths are called a conditional pattern base. Conditional pattern base is a sub-database consisting of prefix paths in the FP tree occurring with the lowest node (suffix).
- 7) Construct a Conditional FP Tree, which is formed by a count of itemsets in the path. The itemsets meeting the threshold support are considered in the Conditional FP Tree.
- 8) Frequent Patterns are generated from the Conditional FP Tree.

Example Of FP-Growth Algorithm

Support threshold=50%, Confidence= 60%

Table 1

Transaction	List of items
T1	I1,I2,I3
T2	I2,I3,I4
T3	I4,I5
T4	I1,I2,I4

Transaction	List of items
T5	I1,I2,I3,I5
T6	I1,I2,I3,I4

Solution:

Support threshold=50% $\Rightarrow 0.5*6=3 \Rightarrow \text{min_sup}=3$

1. Count of each item

Table 2

Item	Count
I1	4
I2	5
I3	4
I4	4
I5	2

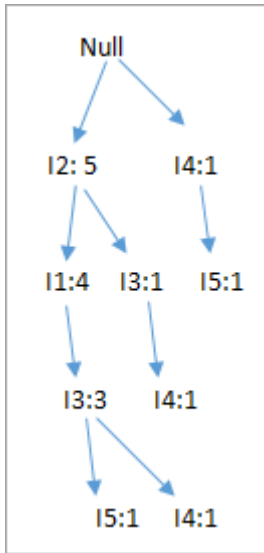
2. Sort the itemset in descending order.

Table 3

Item	Count
I2	5
I1	4
I3	4
I4	4

3. Build FP Tree

1. Considering the root node null.
2. The first scan of Transaction T1: I1, I2, I3 contains three items {I1:1}, {I2:1}, {I3:1}, where I2 is linked as a child to root, I1 is linked to I2 and I3 is linked to I1.
3. T2: I2, I3, I4 contains I2, I3, and I4, where I2 is linked to root, I3 is linked to I2 and I4 is linked to I3. But this branch would share I2 node as common as it is already used in T1.
4. Increment the count of I2 by 1 and I3 is linked as a child to I2, I4 is linked as a child to I3. The count is {I2:2}, {I3:1}, {I4:1}.
5. T3: I4, I5. Similarly, a new branch with I5 is linked to I4 as a child is created.
6. T4: I1, I2, I4. The sequence will be I2, I1, and I4. I2 is already linked to the root node, hence it will be incremented by 1. Similarly I1 will be incremented by 1 as it is already linked with I2 in T1, thus {I2:3}, {I1:2}, {I4:1}.
7. T5:I1, I2, I3, I5. The sequence will be I2, I1, I3, and I5. Thus {I2:4}, {I1:3}, {I3:2}, {I5:1}.
8. T6: I1, I2, I3, I4. The sequence will be I2, I1, I3, and I4. Thus {I2:5}, {I1:4}, {I3:3}, {I4:1}.

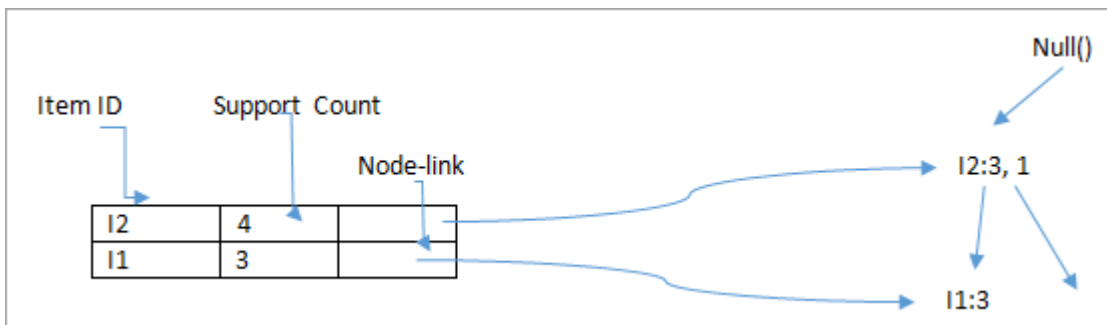


4. Mining of FP-tree is summarized below:

1. The lowest node item I5 is not considered as it does not have a min support count, hence it is deleted.
2. The next lower node is I4. I4 occurs in 2 branches , {I2,I1,I3:,I41},{I2,I3,I4:1}. Therefore considering I4 as suffix the prefix paths will be {I2, I1, I3:1}, {I2, I3: 1}. This forms the conditional pattern base.
3. The conditional pattern base is considered a transaction database, an FP-tree is constructed. This will contain {I2:2, I3:2}, I1 is not considered as it does not meet the min support count.
4. This path will generate all combinations of frequent patterns :
{I2,I4:2},{I3,I4:2},{I2,I3,I4:2}
5. For I3, the prefix path would be: {I2,I1:3},{I2:1}, this will generate a 2 node FP-tree :
{I2:4, I1:3} and frequent patterns are generated: {I2,I3:4}, {I1:I3:3}, {I2,I1,I3:3}.
6. For I1, the prefix path would be: {I2:4} this will generate a single node FP-tree: {I2:4} and frequent patterns are generated: {I2, I1:4}.

Item	Conditional Pattern Base	Conditional FP-tree	Frequent Patterns Generated
I4	{I2,I1,I3:1},{I2,I3:1}	{I2:2, I3:2}	{I2,I4:2},{I3,I4:2},{I2,I3,I4:2}
I3	{I2,I1:3},{I2:1}	{I2:4, I1:3}	{I2,I3:4}, {I1:I3:3}, {I2,I1,I3:3}
I1	{I2:4}	{I2:4}	{I2,I1:4}

The diagram given below depicts the conditional FP tree associated with the conditional node I3.



Advantages of FP Growth Algorithm

1. This algorithm needs to scan the database only twice when compared to Apriori which scans the transactions for each iteration.
2. The pairing of items is not done in this algorithm and this makes it faster.
3. The database is stored in a compact version in memory.
4. It is efficient and scalable for mining both long and short frequent patterns.

Disadvantages Of FP-Growth Algorithm

1. FP Tree is more cumbersome and difficult to build than Apriori.
2. It may be expensive.
3. When the database is large, the algorithm may not fit in the shared memory.

FP Growth vs Apriori

FP Growth	Apriori
Pattern Generation	
FP growth generates pattern by constructing a FP tree	Apriori generates pattern by pairing the items into singletons, pairs and triplets.
Candidate Generation	
There is no candidate generation	Apriori uses candidate generation
Process	
The process is faster as compared to Apriori. The runtime of process increases linearly with increase in number of itemsets.	The process is comparatively slower than FP Growth, the runtime increases exponentially with increase in number of itemsets
Memory Usage	
A compact version of database is saved	The candidates combinations are saved in memory

Gaussian Mixture Model

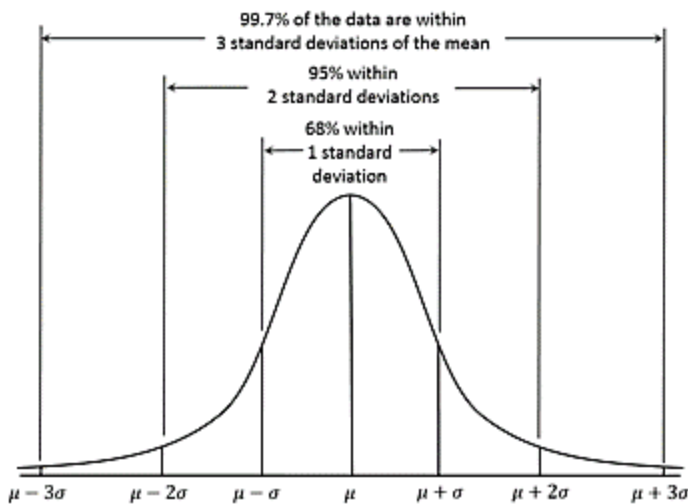
Gaussian Mixture Model or Mixture of Gaussian as it is sometimes called, is not so much a model as it is a probability distribution. It is a universally used model for generative unsupervised learning or clustering. It is also called Expectation-Maximization Clustering or EM Clustering and is based on the optimization strategy. Gaussian Mixture models are used for representing Normally Distributed subpopulations within an overall population. The advantage of Mixture models is that they do not require which subpopulation a data point belongs to. It allows the model to learn the subpopulations automatically. This constitutes a form of unsupervised learning.

A Gaussian is a type of distribution, and it is a popular and mathematically convenient type of distribution. A distribution is a listing of outcomes of an experiment and the probability

associated with each outcome. Let's take an example to understand. We have a data table that lists a set of cyclist's speeds.

Speed (Km/h)	Frequency
1	4
2	9
3	6
4	7
5	3
6	2

A cyclist reaches the speed of 1 Km/h four times, 2Km/h nine times, 3 Km/h and so on. We can notice how this follows, the frequency goes up and then it goes down. It looks like it follows a kind of bell curve the frequencies go up as the speed goes up and then it has a peak value and then it goes down again, and we can represent this using a bell curve otherwise known as a Gaussian distribution.



A Gaussian distribution is a type of distribution where half of the data falls on the left of it, and the other half of the data falls on the right of it. It's an even distribution, and one can notice just by the thought of it intuitively that it is very mathematically convenient. Gaussian distribution would be a great distribution to model the data in those cases where the data reaches a peak and then decreases. Similarly, in Multi Gaussian Distribution, we will have multiple peaks with multiple means and multiple standard deviations.

The formula for Gaussian distribution using the mean and the standard deviation called the **Probability Density Function**:

$$y = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

where μ = Mean
 σ =Standard Deviation

For a given point X, we can compute the associated Y values. Y values are the probabilities for those X values. So, for any X value, we can calculate the probability of that X value being a part of the curve or being a part of the dataset.

This is a function of a continuous random variable whose integral across an interval gives the probability that the value of the variable lies within the same interval.

What is a Gaussian Mixture Model?

Sometimes our data has multiple distributions or it has multiple peaks. It does not always have one peak, and one can notice that by looking at the data set. It will look like there are multiple peaks happening here and there. There are two peak points and the data seems to be going up and down twice or maybe three times or four times. But if there are Multiple Gaussian distributions that can represent this data, then we can build what we called a **Gaussian Mixture Model**.

In other words we can say that, if we have three Gaussian Distribution as GD1, GD2, GD3 having mean as μ_1, μ_2, μ_3 and variance 1,2,3 than for a given set of data points GMM will identify the probability of each data point belonging to each of these distributions.

It is a probability distribution that consists of multiple probability distributions and has Multiple Gaussians.

The probability distribution function of d-dimensions Gaussian Distribution is defined as:

$$N(\mu, \Sigma) = \frac{1}{(2\pi)^{\frac{d}{2}} \sqrt{|\Sigma|}} \exp\left(-\frac{1}{2}(x - \mu)^T \Sigma^{-1} (x - \mu)\right)$$

Where μ = Mean
 Σ = Covariance Matrix of the Gaussian
d= The numbers of features in our dataset
x=the number of datapoints