Machine Learning Unit -2 Questions/ Answers

Unit II : Unsupervised Learning: Clustering: K-means, Dimensionality Reduction: PCA and kernel PCA, Generative Models (Gaussian Mixture Models and Hidden Markov Models)

Category – 1 (Easy)– A
(Short Answer Questions)
(2 Marks questions)

Unit – II
Category – 1 (Easy)

1. Write different unsupervised learning algorithms.

Different unsupervised learning algorithms:

1. K-Means algorithm
2. Mean-shift algorithm
3. DBSCAN Algorithm Expectation-Maximization Clustering using GMM
4. Agglomerative Hierarchical algorithm
5. Affinity Propagation

~~~~~~

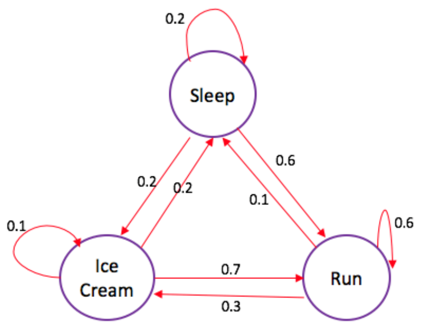
2. What is a Gaussian mixture?

Gaussian Mixture: A *Gaussian Mixture* is a function that is comprised of several Gaussians, each identified by $k \in \{1\ldots K\}$, where K is the number of clusters of our dataset. Each Gaussian k in the mixture is comprised of the following parameters:

- A mean μ that defines its centre.
- A covariance Σ that defines its width.
- A mixing probability π that defines how big or small the Gaussian function will be.

~~~~~~

3. What is the property of a Markov chain?
Answer:
Markov Chain:
A Markov chain is a process that occurs in a series of time-steps in each of which a random choice is made among a finite (or also enumerable) number of states.



Property of a Markov Chain:
An especially simple class of Markov processes are the Markov chains, which we define by the following properties:

- The range of Y is a discrete set of states.

- The time variable is discrete and takes only integer values *t* = …, −2, −1, 0, 1, 2…

~~~~~~

4. What is a principal component?

Answer:
Principal Component Analysis:
Principal Component Analysis is an unsupervised learning algorithm that is used for the dimensionality reduction in machine learning. It is a statistical process that converts the observations of correlated features into a set of linearly uncorrelated features with the help of orthogonal transformation.
Principal Component:
A principal component can be defined as a linear combination of optimally-weighted observed variables.

~~~~~~

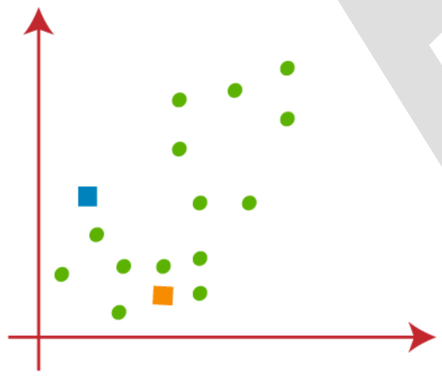
5. Write the function used in SKlearn for dimensionality reduction.
Answer:
Dimensionality reduction, an unsupervised machine learning method is used to reduce the number of feature variables for each data sample selecting set of principal features. Principal Component Analysis (PCA) is one of the popular algorithms for dimensionality reduction.

```
from sklearn.decomposition import PCA
```

~~~~~~

Category – 2 (Moderate)

1. **Initial centroids are fed as input to k-means algorithm. How do you choose them?**
   Answer:
   K-Means Algorithm: 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.
   Some random k points or centroid are chosen to form the cluster. These points can be either the points from the dataset or any other point.



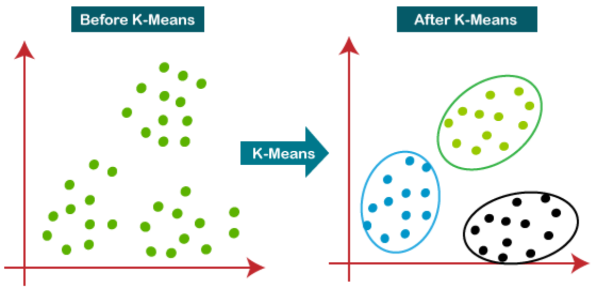
~~~~~~

2. **When do you say that good clusters are formed using k-means clustering algorithm?**
   Answer:

When we find the nearest numbers to the mean and find out its mean value, if the mean or centroid value doesn't change anymore, then then good clusters are formed using k mean clustering

Which distance measure is suitable in k-means clustering algorithm?

Answer: K-Means Clustering: 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.Euclidean distance measure is suitable in k-means clustering algorithm.



~~~~~~

3. **Differentiate between hard and soft clustering.**

Answer:

Difference between hard and soft clustering:

- **Hard Clustering:** In hard clustering, each data point either belongs to a cluster completely or not. K-Means is a famous hard clustering algorithm whereby the data items are clustered into K clusters such that each item only blogs to one cluster.

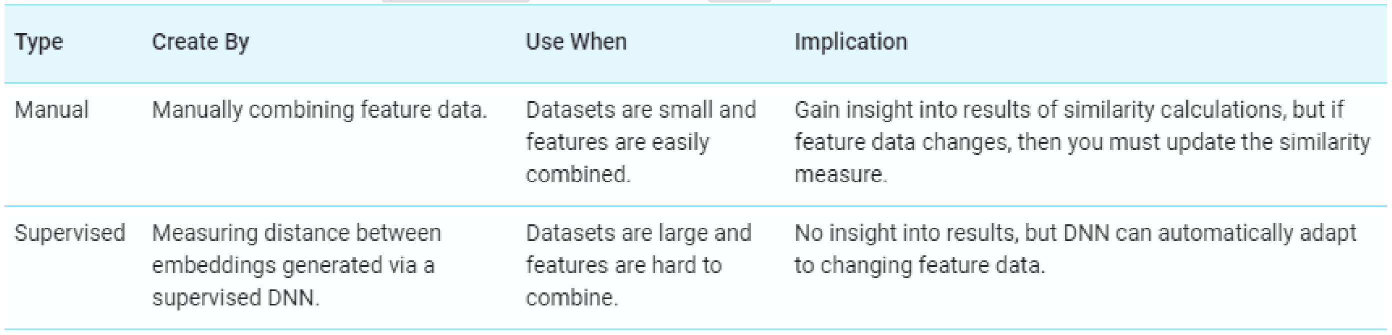
- **Soft Clustering**: In soft clustering, instead of putting each data point into a separate cluster, a probability or likelihood of that data point to be in those clusters is assigned. Fuzzy C-means is a famous soft clustering algorithm. It is based on the fuzzy logic and is often referred to as the FCM algorithm.

~~~~~~

4. **Differentiate between the similarity measures.**

Answer:

Difference between the similarity measures:



Type	Create By	Use When	Implication
Manual	Manually combining feature data.	Datasets are small and features are easily combined.	Gain insight into results of similarity calculations, but if feature data changes, then you must update the similarity measure.
Supervised	Measuring distance between embeddings generated via a supervised DNN.	Datasets are large and features are hard to combine.	No insight into results, but DNN can automatically adapt to changing feature data.

~~~~~~

SECTION – B
(Essay Questions)
(8 Marks questions)

Easy Questions

1) Broadly categorize the clustering algorithms.

Answer:
~~~~~~

Clustering Algorithms:The Clustering algorithms can be divided based on their models. The clustering algorithm is based on the kind of data that we are using. Such as, some algorithms need to guess the number of clusters in the given dataset, whereas some are required to find the minimum distance between the observations of the dataset.

Here we are discussing Clustering algorithms that are widely used in machine learning:

1. **K-Means algorithm:** The k-means algorithm is one of the most popular clustering algorithms. It classifies the dataset by dividing the samples into different clusters of equal variances. The number of clusters must be specified in this algorithm.
2. **Mean-shift algorithm:** Mean-shift algorithm tries to find the dense areas in the smooth density of data points. It is an example of a centroid-based model that works on updating the candidates for centroid to be the center of the points within a given region.
3. **DBSCAN Algorithm:** It stands for Density-Based Spatial Clustering of Applications with **Noise**. It is an example of a density-based model similar to the mean-shift, but with some remarkable advantages. In this algorithm, the areas of high density are separated by the areas of low density. Because of this, the clusters can be found in any arbitrary shape.
4. **Expectation-Maximization Clustering using GMM:** This algorithm can be used as an alternative for the k-means algorithm or for those cases where K-means can be failed. In GMM, it is assumed that the data points are Gaussian distributed.
5. **Agglomerative Hierarchical algorithm:** The Agglomerative hierarchical algorithm performs the bottom-up hierarchical clustering. In this, each data point is treated as a single cluster at the outset and then successively merged. The cluster hierarchy can be represented as a tree-structure.
6. **Affinity Propagation:** It is different from other clustering algorithms as it does not require specifying the number of clusters. In this, each data point sends a message between the pair of data points until convergence.

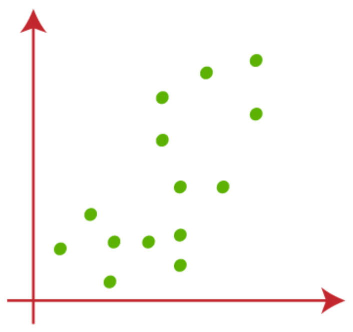
~~~~~~

2. Write the steps in finding the clusters using k-means algorithm.

Answer:

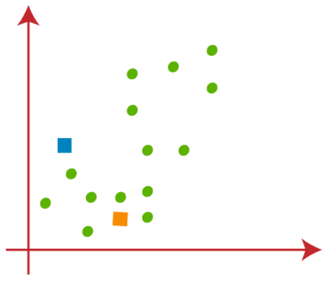
Steps in finding the clusters using K-means Algorithm:

Step-1: Select the number K to decide the number of clusters.
Step-2: Select random K points or centroid. (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 mean reassign each data point 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.
Let's understand the above steps by considering the visual plots:
Suppose we have two variables M1 and M2. The x-y axis scatter plot of these two variables is given below:

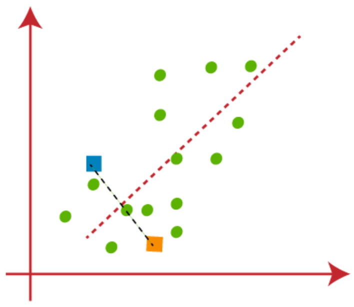


➢ Let's take number k of clusters, i.e., K=2, to identify the dataset and to put them into different clusters. It means here we will try to group these datasets into two different clusters.
~~~~~~

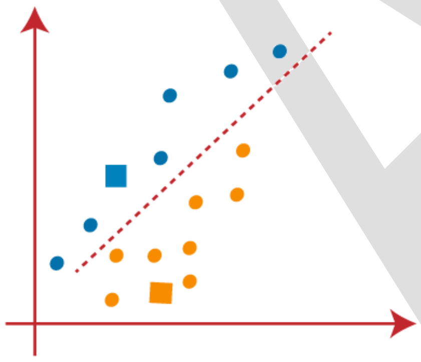
- We need to choose some random k points or centroid to form the cluster. These points can be either the points from the dataset or any other point. So, here we are selecting the below two points as k points, which are not the part of our dataset. Consider the below image:



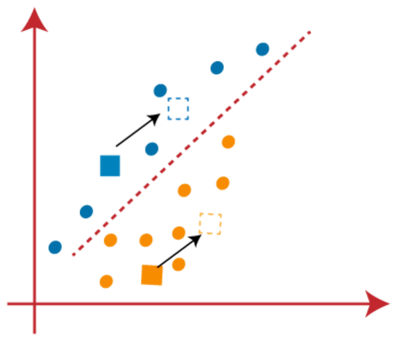
- Now we will assign each data point of the scatter plot to its closest K-point or centroid. We will compute it by applying some mathematics that we have studied to calculate the distance between two points. So, we will draw a median between both the centroids. Consider the below image:



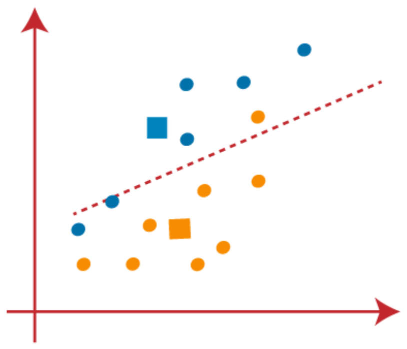
- From the above image, it is clear that points left side of the line is near to the K1 or blue centroid, and points to the right of the line are close to the yellow centroid. Let's color them as blue and yellow for clear visualization.



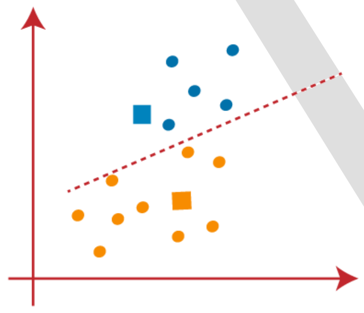
- As we need to find the closest cluster, so we will repeat the process by choosing **a new centroid**. To choose the new centroids, we will compute the center of gravity of these centroids, and will find new centroids as below:



- 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:

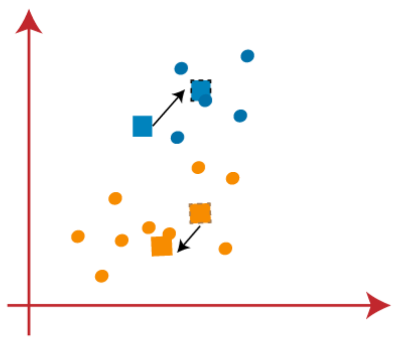


- From the above image, we can see, one yellow point is on the left side of the line, and two blue points are right to the line. So, these three points will be assigned to new centroids.

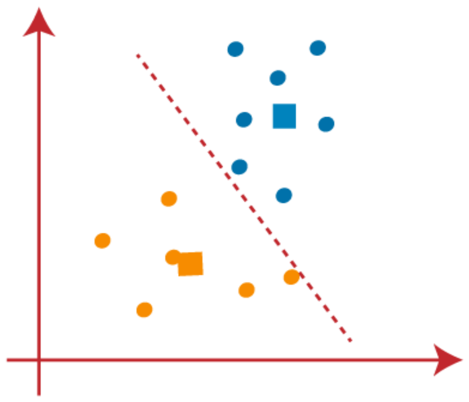


- As reassignment has taken place, so we will again go to the step-4, which is finding new centroids or K-points.

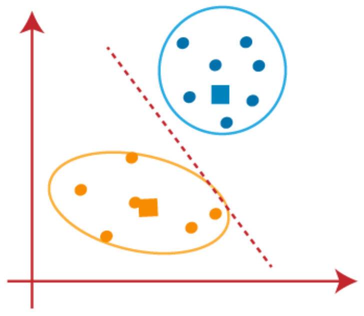
- We will repeat the process by finding the center of gravity of centroids, so the new centroids will be as shown in the below image:



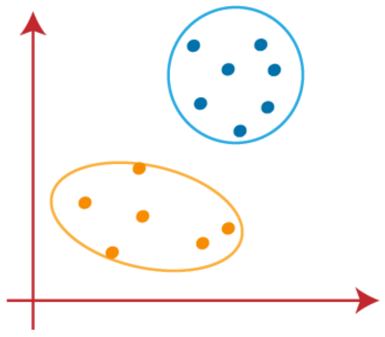
- As we got the new centroids so again will draw the median line and reassign the data points. So, the image will be:



- 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:



~~~~~~

3. What are the real world applications of clustering technique?

Answer: *Real World Applications Of Clustering Technique:*

i) Identifying Fake News:

Fake news is being created and spread at a rapid rate due to technology innovations such as social media. The issue gained attention recently during the 2016 US presidential campaign. During this campaign, the term Fake News was referenced an unprecedented number of times. The way that the algorithm works is **by taking in the content of the fake** news article, the corpus, examining the words used and **then clustering them**. These clusters are what helps the algorithm determine which pieces are genuine and which are fake news. Certain words are found more commonly in sensationalized, click-bait articles. When you see a high percentage of specific terms in an article, it gives a higher probability of the material being fake news.

ii) Spam filter: It is the place where emails that have been identified as spam by the algorithm.
Problem: Spam emails are at best annoying part of modern day marketing techniques. To avoid getting these emails in your main inbox, email companies use algorithms. The purpose of these algorithms is to flag an email as spam correctly or not.
How clustering helps: K-Means clustering techniques have proven to be an effective way of identifying spam. The way that it works is by looking at the different sections of the email (header, sender, and content). The data is then grouped together. These groups can then be classified to identify which is spam. Including clustering in the classification process improves the accuracy of the filter to 97%.

iii) Marketing and Sales: Personalization and targeting in marketing is big business. This is achieved by looking at specific characteristics of a person and sharing campaigns with them that have been successful with other similar people.

Problem: If your business is trying to get the best return on your marketing investment, it is crucial that you target people in the right way. If you get it wrong, you risk not making any sales, or worse, damaging your Customer trust.
How clustering helps: Clustering algorithms are able to group together people with similar traits and likelihood to purchase. Once you have the groups, you can run tests on each group with different marketing copy that will help you better target your messaging to them in the future.

iv) Classifying network traffic: Imagine you want to understand the different types of traffic coming to your website. You are particularly interested in understanding which traffic is spam or coming from bots.
Problem: As more and more services begin to use APIs on your application, or as your website grows, it is important you know where the traffic is coming from. For example, you want to be able to block harmful traffic and double down on areas driving growth. However, it is hard to know which is which when it comes to classifying the traffic.
How clustering helps: K-means clustering is used to group together characteristics of the traffic sources. When the clusters are created, you can then classify the traffic types. The process is faster and more accurate than the Autoclass method. By having precise information on traffic sources, you are able to grow your site and plan capacity effectively.
~~~~~~

v) Identifying fraudulent or criminal activity : In this scenario, we are going to focus on fraudulent taxi driver behavior. However, the technique has been used in multiple scenarios. **Problem:** You need to look into fraudulent driving activity. The challenge is how do you identify what is true and which is false? **How clustering helps:** By analysing the GPS logs, the algorithm is able to group similar behaviours. Based on the characteristics of the groups you are then able to classify them into those that are real and which are fraudulent.

vi) Document Analysis: There are many different reasons why you would want to run an analysis on a document. In this scenario, you want to be able to organize the documents quickly and efficiently. **Problem**: Imagine you are limited in time and need to organize information held in documents quickly. To be able to complete this ask you need to: understand the theme of the text, compare it with other documents and classify it. **How clustering helps:** Hierarchical clustering has been used to solve this problem. The algorithm is able to look at the text and group it into different themes. Using this technique, you can cluster and organize similar documents quickly using the characteristics identified in the paragraph.

vii) Fantasy Football and Sports : **Problem :** Who should you have in your team? Which players are going to perform best for your team and allow you to beat the competition? The challenge at the start of the season is that there is very little if any data available to help you identify the winning players. **How clustering helps:** When there is little performance data available to train your model on, you have an advantage for unsupervised learning. In this type of machine learning problem, you can find similar players using some of their characteristics. This has been done using K-Means clustering. Ultimately this means you can get a better team more quickly at the start of the year, giving you an advantage.
~~~~~~

4. Differentiate between hierarchical and partitioning clustering? Brief different hierarchical clustering methods.

Answer:

Difference between hierarchical and partitioning clustering:
Hierarchical and Partitional Clustering have key differences in running time, assumptions, input parameters and resultant clusters.

1. Typically, Partitional clustering is faster than hierarchical clustering. Hierarchical clustering requires only a similarity measure, while partitional clustering requires stronger assumptions such as number of clusters and the initial centers.

2. Hierarchical clustering does not require any input parameters, while partitional clustering algorithms require the number of clusters to start running.

3. Hierarchical clustering returns a much more meaningful and subjective division of clusters but Partitional clustering results in exactly k clusters.

4. Hierarchical clustering algorithms are more suitable for categorical data as long as a similarity measure can be defined accordingly.

Different hierarchical clustering methods:
There are two types of hierarchical clustering methods:

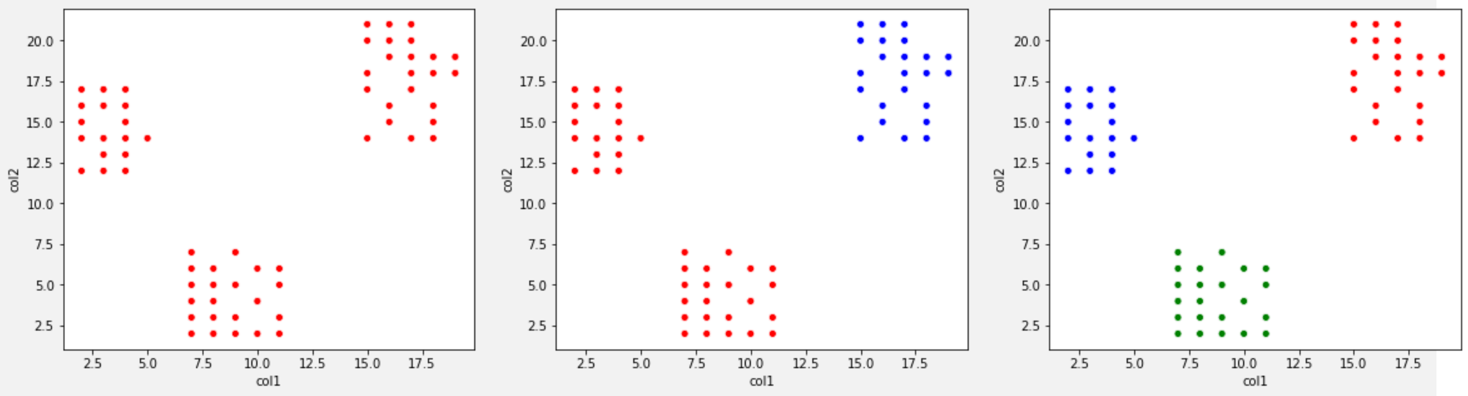
- Divisive Clustering
- Agglomerative Clustering
~~~~~~

<u>Divisive Clustering:</u>

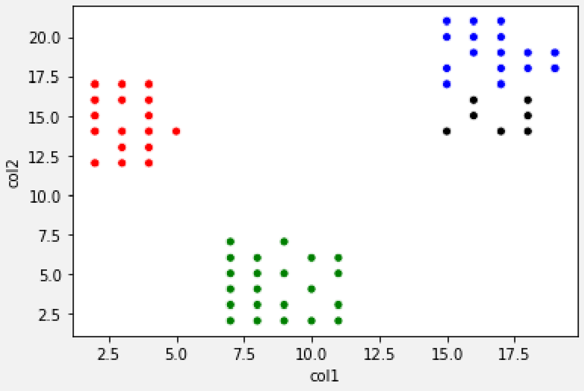
The divisive clustering algorithm is a top-down clustering approach, initially, all the points in the dataset belong to one cluster and split is performed recursively as one moves down the hierarchy.

Steps of Divisive Clustering:

- Initially, all points in the dataset belong to one single cluster.
- Partition the cluster into two least similar cluster
- Proceed recursively to form new clusters until the desired number of clusters is obtained.



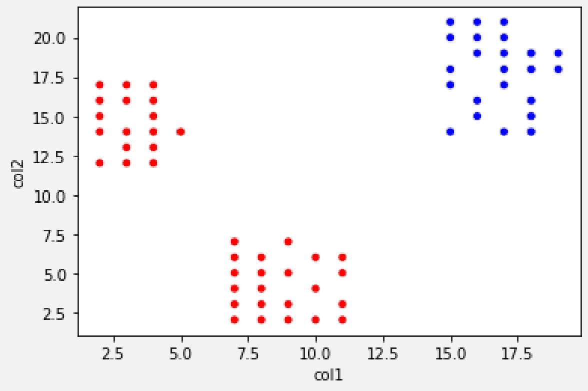
(Image by Author), 1st Image: All the data points belong to one cluster, 2nd Image: 1 cluster is separated from the previous single cluster, 3rd Image: Further 1 cluster is separated from the previous set of clusters. In the above sample dataset, it is observed that there is 3 cluster that is far separated from each other. So we stopped after getting 3 clusters. Even if start separating further more clusters, below is the obtained result.



(Image by Author), Sample dataset separated into 4 clusters

How to choose which cluster to split?

Check the sum of squared errors of each cluster and choose the one with the largest value. In the below 2-dimension dataset, currently, the data points are separated into 2 clusters, for further separating it to form the 3rd cluster find the sum of squared errors (SSE) for each of the points in a red cluster and blue cluster.



(Image by Author), Sample dataset separated into 2clustersThe cluster with the largest SSE value is separated into 2 clusters, hence forming a new cluster. In the above image, it is observed red cluster has larger SSE so it is separated into 2 clusters forming 3 total clusters.

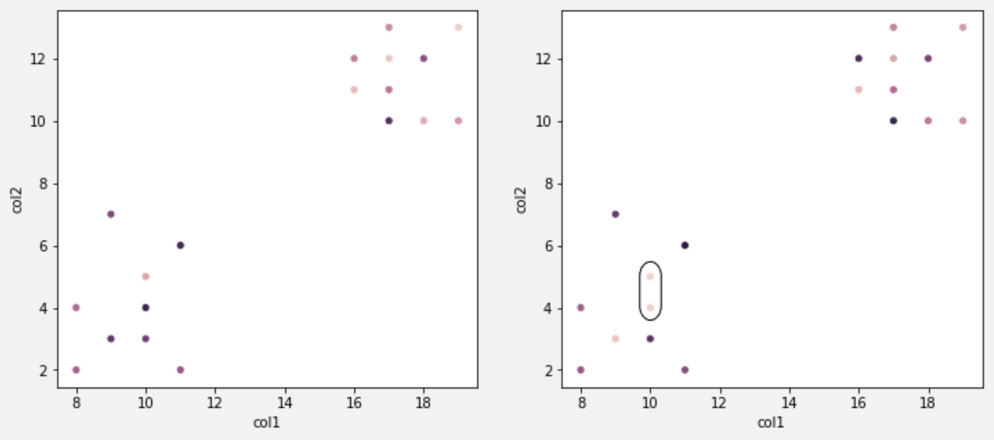
How to split the above-chosen cluster? Once we have decided to split which cluster, then the question arises on how to split the chosen cluster into 2 clusters. One way is to use Ward's criterion to chase for the largest reduction in the difference in the SSE criterion as a result of the split.

How to handle the noise or outlier? Due to the presence of outlier or noise, can result to form a new cluster of its own. To handle the noise in the dataset using a threshold to determine the termination criterion that means do not generate clusters that are too small.

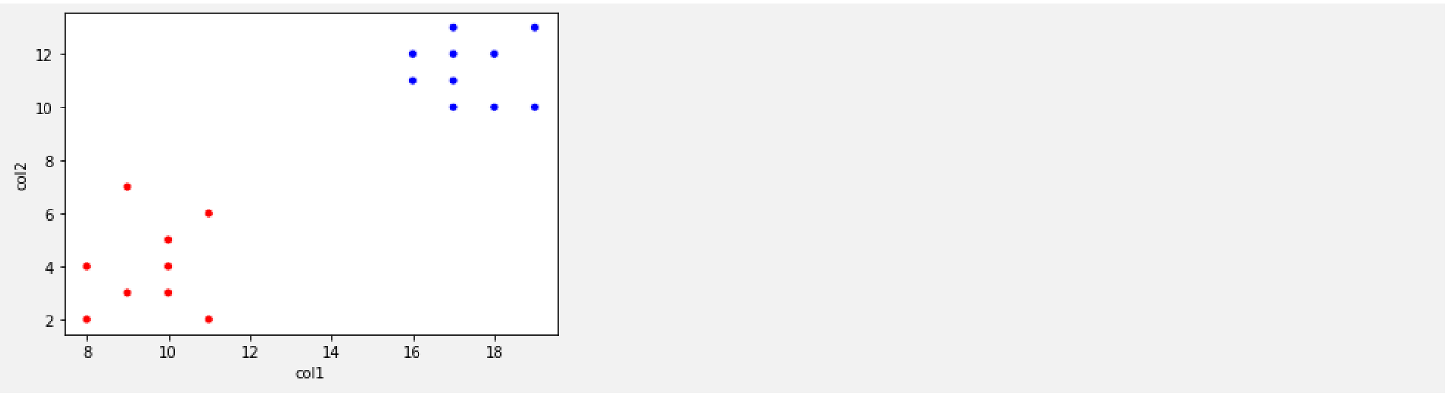
Agglomerative Clustering: Agglomerative Clustering is a bottom-up approach, initially, each data point is a cluster of its own, further pairs of clusters are merged as one moves up the hierarchy.

Steps of Agglomerative Clustering:

- Initially, all the data-points are a cluster of its own.
- Take two nearest clusters and join them to form one single cluster.
- Proceed recursively step 2 until you obtain the desired number of clusters.



(Image by Author), 1st Image: All the data point is a cluster of its own, 2nd Image: Two nearest clusters (surrounded by a black oval) joins together to form a single cluster.
In the above sample dataset, it is observed that 2 clusters are far separated from each other. So we stopped after getting 2 clusters.



(Image by Author), Sample dataset separated into 2 clusters

How to join two clusters to form one cluster?To obtain the desired number of clusters, the number of clusters needs to be reduced from initially being n cluster (n equals the total number of data-points). Two clusters are combined by computing the similarity between them.

There are some methods which are used to calculate the similarity between two clusters:

- Distance between two closest points in two clusters.
- Distance between two farthest points in two clusters.
- The average distance between all points in the two clusters.
- Distance between centroids of two clusters.

~~~~~~

Moderate Questions

1. What does k represent in k-means algorithm? How do you determine the value of k?

Answer:

K indicates – Number of clusters you want to form. User has to specify the value of k. You have algorithm to choose K- Elbow method.

How to choose the value of "K number of clusters" in K-means Clustering? The performance of the K-means clustering algorithm depends upon highly efficient clusters that it forms. But choosing the optimal number of clusters is a big task. There are some different ways to find the optimal number of clusters, but here we are discussing the most appropriate method to find the number of clusters or value of K. The method is given below:

Elbow Method: 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:

$$WCSS= \sum_{P_i \text{ in Cluster1}} distance(P_i\ C_1)^2 + \sum_{P_i \text{ in Cluster2}} distance(P_i\ C_2)^2 + \sum_{P_i \text{ in CLuster3}} distance(P_i\ C_3)^2$$

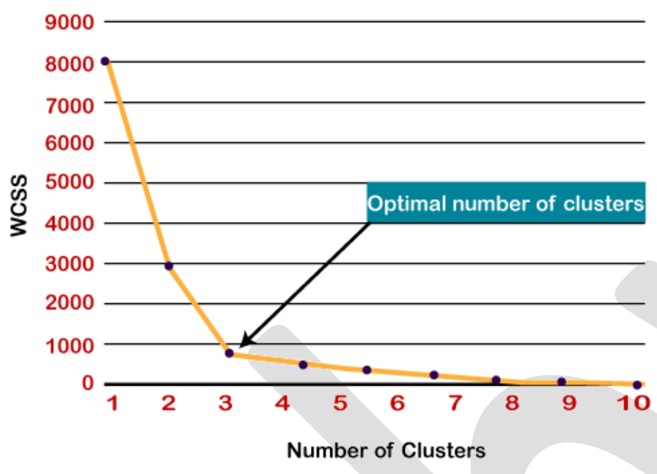
In the above formula of WCSS,

$\sum_{P_i \text{ in Cluster1}} distance(P_i\ C_1)^2$: It is the sum of the square of the distances between each data point and its centroid within a cluster1 and the same for the other two terms. To measure the distance between data points and centroid, we can use any method such as Euclidean distance or Manhattan distance.

To find the optimal value of clusters, the elbow method follows the below steps:
~~~~~~

- ➢ It executes the K-means clustering on a given dataset for different K values (ranges from 1-10).
- ➢ For each value of K, calculates the WCSS value.
- ➢ Plots a curve between calculated WCSS values and the number of clusters K.
- ➢ The sharp point of bend or a point of the plot looks like an arm, then that point is considered as the best value of K.

Since the graph shows the sharp bend, which looks like an elbow, hence it is known as the elbow method. The graph for the elbow method looks like the below image:



~~~~~~

2. What is the necessity of performing dimensionality reduction as a pre-processing step in machine learning?
Answer:

- Principal Component Analysis is an unsupervised learning algorithm that is used for the dimensionality reduction in machine learning. It is a statistical process that converts the observations of correlated features into a set of linearly uncorrelated features with the help of orthogonal transformation. These new transformed features are called the Principal Components.

Principal Components in PCA: As described above, the transformed new features or the output of PCA are the Principal Components. The number of these PCs is either equal to or less than the original features present in the dataset.

Algorithm:

Step 1: Normalize the data : First step is to normalize the data that we have so that PCA works properly. This is done by subtracting the respective means from the numbers in the respective column. So if we have two dimensions X and Y, all X become x- and all Y become y-. This produces a dataset whose mean is zero.
Step 2: Calculate the covariance matrix : Since the dataset we took is 2-dimensional, this will result in a 2x2 Covariance matrix.

$$\boxed{Matrix(Covariance) = \begin{bmatrix} Var[X_1] & Cov[X_1, X_2] \\ Cov[X_2, X_1] & Var[X_2] \end{bmatrix}}$$

Please note that $Var[X_1] = Cov[X_1,X_1]$ and $Var[X_2] = Cov[X_2,X_2]$.
~~~~~~

Step 3: Calculate the eigenvalues and eigenvectors: Next step is to calculate the eigenvalues and eigenvectors for the covariance matrix. The same is possible because it is a square matrix. ***ƛ*** is an eigenvalue for a matrix ***A*** if it is a solution of the characteristic equation:

$$det(\lambda I - A) = 0$$

Where, ***I*** is the identity matrix of the same dimension as ***A*** which is a required condition for the matrix subtraction as well in this case and '***det'*** is the determinant of the matrix. For each eigenvalue ***ƛ***, a corresponding eigen-vector ***v***, can be found by solving:

$$(\lambda I - A)v = 0$$

Step 4: Choosing components and forming a feature vector: We order the eigenvalues from largest to smallest so that it gives us the components in order or significance. Here comes the dimensionality reduction part. If we have a dataset with *n* variables, then we have the corresponding *n* eigenvalues and eigenvectors. It turns out that the eigenvector corresponding to the highest eigenvalue is the principal component of the dataset and it is our call as to how many eigenvalues we choose to proceed our analysis with. To reduce the dimensions, we choose the first *p* eigenvalues and ignore the rest. We do lose out some information in the process, but if the eigenvalues are small, we do not lose much. Next we form a feature vector which is a matrix of vectors, in our case, the eigenvectors. In fact, only those eigenvectors which we want to proceed with. Since we just have 2 dimensions in the running example, we can either choose the one corresponding to the greater eigenvalue or simply take both.

$$Feature\ Vector = (eig_1, eig_2)$$

Step 5: Forming Principal Components: This is the final step where we actually form the principal components using all the math we did till here. For the same, we take the transpose of the feature vector and left-multiply it with the transpose of scaled version of original dataset.

$$NewData = FeatureVector^T \times ScaledData^T$$

Here,

NewData is the Matrix consisting of the principal components,

FeatureVector is the matrix we formed using the eigenvectors we chose to keep, and

ScaledData is the scaled version of original dataset

('T' in the superscript denotes transpose of a matrix which is formed by interchanging the rows to columns and vice versa. In particular, a 2x3 matrix has a transpose of size 3x2). If we go back to the theory of eigenvalues and eigenvectors, we see that, essentially, eigenvectors provide us with information about the patterns in the data. In particular, in the running example of 2-D set, if we plot the eigenvectors on the scatterplot of data, we find that the principal eigenvector (corresponding to the largest eigenvalue) actually fits well with the data. The other one, being perpendicular to it, does not carry much information and hence, we are at not much loss when deprecating it, hence reducing the dimension.

All the eigenvectors of a matrix are perpendicular to each other. So, in PCA, what we do is represent or transform the original dataset using these orthogonal (perpendicular) eigenvectors instead of representing on normal ***x*** and ***y*** axes. We have now classified our data points as a combination of contributions from both ***x*** and ***y.*** The difference lies when we actually disregard one or many eigenvectors, hence, reducing the dimension of the dataset. Otherwise, in case, we take all the eigenvectors in account, we are just transforming the co-ordinates and hence, not serving the purpose.

~~~~~~

3. How do you find the hidden states of hidden markov method?

Answer:

~~~~~~

Standard Questions

1. Differentiate between expectation step and maximization step in EM algorithm.
Answer:
Expectation — Maximization algorithm: Well, at this point we have derived some expressions for the probabilities that we will find useful in determining the parameters of our model. However, in the past section we could see that simply evaluating (3) to find such parameters would prove to be very hard. Fortunately, there is an iterative method we can use to achieve this purpose. It is called the *Expectation — Maximization*, or simply *EM algorithm*. It is widely used for optimization problems where the objective function has complexities such as the one we've just encountered for the GMM case. Let the parameters of our model be

$$\theta = \{\pi, \mu, \Sigma\}$$

Let us now define the steps that the general EM algorithm wills follow.

Step 1: Initialise ϑ accordingly. For instance, we can use the results obtained by a previous K-Means run as a good starting point for our algorithm.

Step 2 (Expectation step): Evaluate

$$Q(\theta^*, \theta) = \mathbb{E}[\ln p(\mathbf{X}, \mathbf{Z}|\theta^*)] = \sum_{\mathbf{Z}} p(\mathbf{Z}|\mathbf{X}, \theta) \ln p(\mathbf{X}, \mathbf{Z}|\theta^*) \qquad (5)$$

Well, actually we have already found $p(\mathbf{Z}|\mathbf{X}, \vartheta)$. Remember the γ expression we ended up with in the previous section?

For better visibility, let's bring our earlier equation (4) here:

$$p(z_k = 1|\mathbf{x}_n) = \frac{\pi_k \mathcal{N}(\mathbf{x}_n|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\mathbf{x}_n|\mu_j, \Sigma_j)} = \gamma(z_{nk}) \qquad (4)$$

For Gaussian Mixture Models, the expectation step boils down to calculating the value of γ in (4) by using the old parameter values. Now if we replace (4) in (5), we will have:

$$\mathcal{Q}(\theta^*, \theta) = \sum_{\mathbf{Z}} \gamma(z_{nk}) \ln p(\mathbf{X}, \mathbf{Z}|\theta^*) \qquad (6)$$

Sounds good, but we are still missing *p*(**X**, **Z**|*ϑ**). How can we find it? Well, actually it's not that difficult. It is just the complete likelihood of the model, including both **X** and **Z**, and we can find it by using the following expression:

$$p(\mathbf{X}, \mathbf{Z}|\theta^*) = \prod_{n=1}^{N} \prod_{k=1}^{K} \pi^{z_{nk}} \mathcal{N}(\mathbf{x}_n|\mu_k, \Sigma_k)^{z_{nk}}$$

Which is the result of calculating the joint probability of all observations and latent variables and is an extension of our initial derivations for *p*(**x**). The log of this expression is given by

$$\ln p(\mathbf{X}, \mathbf{Z}|\theta^*) = \sum_{n=1}^{N} \sum_{k=1}^{K} z_{nk} [\ln \pi_k + \ln \mathcal{N}(\mathbf{x}_n|\mu_k, \Sigma_k)] \qquad (7)$$

Nice! And we have finally gotten rid of this troublesome logarithm that affected the summation in (3). With all of this in place, it will be much easier for us to estimate the parameters by just maximizing *Q* with respect to the parameters, but we will deal with this in the *maximization step*. Besides, remember that the latent variable *z* will **only** be 1 once everytime the summation is evaluated. With that knowledge, we can easily get rid of it as needed for our derivations.
Finally, we can replace (7) in (6) to get:

$$\mathcal{Q}(\theta^*, \theta) = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma(z_{nk}) [\ln \pi_k + \ln \mathcal{N}(\mathbf{x}_n|\mu_k, \Sigma_k)] \qquad (8)$$

In the maximization step, we will find the revised parameters of the mixture. For this purpose, we will need to make Q a restricted maximization problem and thus we will add a Lagrange multiplier to (8). Let's now review the maximization step.

Step 3 (Maximization step): Find the revised parameters *ϑ** using:

$$\theta^* = \arg\max_\theta \mathcal{Q}(\theta^*, \theta)$$

Where

$$\mathcal{Q}(\theta^*, \theta) = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma(z_{nk})[\ln \pi_k + \ln \mathcal{N}(\mathbf{x}_n|\mu_k, \Sigma_k)] \qquad (8)$$

However, Q should also take into account the restriction that all π values should sum up to one. To do so, we will need to add a suitable Lagrange multiplier. Therefore, we should rewrite (8) in this way:

$$\mathcal{Q}(\theta^*, \theta) = \sum_{n=1}^{N} \sum_{k=1}^{K} \gamma(z_{nk})[\ln \pi_k + \ln \mathcal{N}(x_n|\mu_k, \Sigma_k)] - \lambda \left(\sum_{k=1}^{K} \pi_k - 1 \right) \qquad (8)$$

And now we can easily determine the parameters by using maximum likelihood. Let’s now take the derivative of *Q* with respect to π and set it equal to zero:

$$\frac{\partial \mathcal{Q}(\theta^*, \theta)}{\partial \pi_k} = \sum_{n=1}^{N} \frac{\gamma(z_{nk})}{\pi_k} - \lambda = 0$$

Then, by rearranging the terms and applying a summation over *k* to both sides of the equation, we obtain:

$$\sum_{n=1}^{N} \gamma(z_{nk}) = \pi_k \lambda \implies \sum_{k=1}^{K} \sum_{n=1}^{N} \gamma(z_{nk}) = \sum_{k=1}^{K} \pi_k \lambda$$

From (1), we know that the summation of all mixing coefficients π equals one. In addition, we know that summing up the probabilities γ over *k* will also give us 1. Thus we get *λ* = *N*. Using this result, we can solve for *π*:

$$\pi_k = \frac{\sum_{n=1}^{N} \gamma(z_{nk})}{N}$$

Similarly, if we differentiate *Q* with respect to μ and Σ, equate the derivative to zero and then solve for the parameters by making use of the log-likelihood equation (2) we defined, we obtain:

$$\mu_k^* = \frac{\sum_{n=1}^{N} \gamma(z_{nk}) \mathbf{x}_n}{\sum_{n=1}^{N} \gamma(z_{nk})}, \qquad \Sigma_k^* = \frac{\sum_{n=1}^{N} \gamma(z_{nk}) (\mathbf{x}_n - \mu_k)(\mathbf{x}_n - \mu_k)^T}{\sum_{n=1}^{N} \gamma(z_{nk})}$$

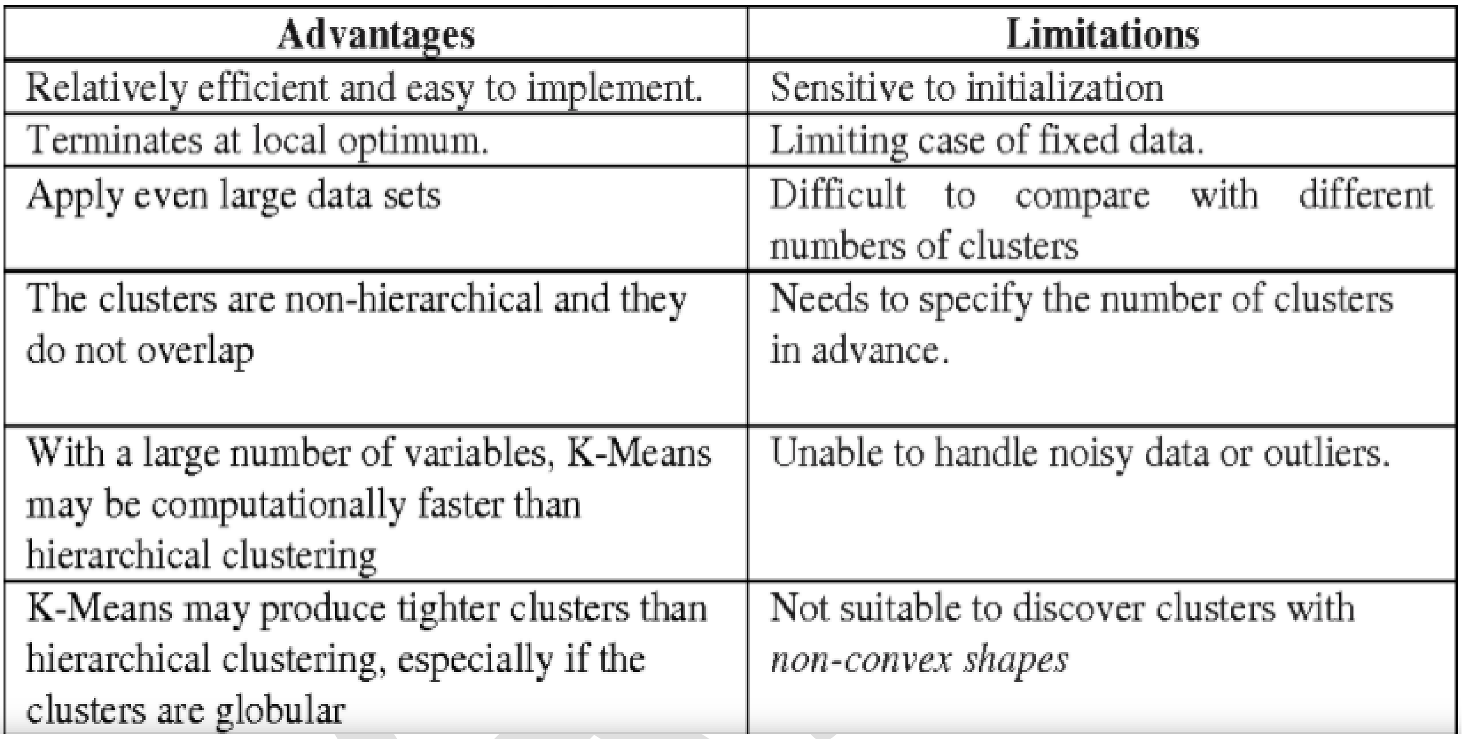
Then we will use these revised values to determine γ in the next EM iteration and so on and so forth until we see some convergence in the likelihood value. We can use equation (3) to monitor the log-likelihood in each step and we are always guaranteed to reach a local maximum.

~~~~~~

2. HMM algorithm uses different data structures. What is their purpose?
Answer:

~~~~~~

3. What are the pros and cons of k-means clustering?
Answer:



Advantages	Limitations
Relatively efficient and easy to implement.	Sensitive to initialization
Terminates at local optimum.	Limiting case of fixed data.
Apply even large data sets	Difficult to compare with different numbers of clusters
The clusters are non-hierarchical and they do not overlap	Needs to specify the number of clusters in advance.
With a large number of variables, K-Means may be computationally faster than hierarchical clustering	Unable to handle noisy data or outliers.
K-Means may produce tighter clusters than hierarchical clustering, especially if the clusters are globular	Not suitable to discover clusters with *non-convex shapes*