MACHINE LEARNING
SCHEME 2026

a) How would you defined Machine Learning?

Machine learning is a technique that enables computers to learn patterns from data and make predictions without being explicitly programmed.

b) What is an online learning system?

It is a ML system that train the system incrementally by feeding it data instances sequentially or in small groups called minibatches.

c) Define under-fitting and over-fitting.

Under-fitting :-

A model with low variance and high bias.

Over-fitting :-

A model with low bias and high variance.

d) What is Bayes rule.

It gives the probability of a hypothesis h given observed data D.

$$P(h|D) = \frac{P(D|h) \cdot P(h)}{P(D)}$$

e) Why use Decision trees.

Decision trees are the supervised learning model that are used for making predictions because they are easy to handle both numerical and categorical data.

f. Define Information Gain and Gini Index?

Information Gain:-

Information gain is the reduction in entropy after splitting data on an attribute.

$$IG(S,A) = \text{Entropy}(S) - \sum_{v} \frac{|S_v|}{|S|} \text{Entropy}(S_v)$$

Gini Index :-

It is a measure of impurity in the data set with respect to the distribution of the samples in a dataset as a fraction of the target class.

$$\text{Gini}(D) = 1 - \sum_{i=1}^{n} P_i^2$$

g) What is an ROC curve?

- The receiver operating characteristic (ROC) curve is another common tool used with binary classifiers.
- It is very similar to the precision-recall curve, but instead of plotting precision versus recall, the ROC plots the positive versus false positive rate.

h) What is ensemble learning?

Ensemble learning:-

Combinations of models are generally known as model ensembles.

A group of predictors is called an ensemble thus, this technique is called ensemble learning.

(OR)

An ensemble model is a machine learning model that combines the predictions from two or more models.

i) Difference between softvoting and hardvoting.

softvoting:-

uses the predicted probabilities and the final prediction is based on average probabilities

Hard voting:-

uses the predicted class labels and the final prediction is based on majority voting.

j) Define hypothesis space.

The set of all possible hypotheses that a learning algorithm can consider.

k) Difference between clustering and classification.

clustering:-

1. Data points are unsupervised learning technique.
2. It works only with unlabeled data.

classification:-

1. Data points are supervised learning technique.
2. It works with both labeled and unlabeled data.

l) Define unsupervised learning.

unsupervised learning:-

It is a type of learning where the training data is unlabeled and the system tries to learn without a teacher.

m) Short notes on hierarchical clustering.

It is a unsupervised machine learning algorithm where it builds a tree like structure (dendogram) of cluster.

There are two approaches:-

1. Agglomerative (bottom-up)
2. Divisive (top-down)

n) Which clustering algorithm is based on the concept of centroids?

K-means clustering algorithm.

Unit-I

2a).Explain different types of Machine learning techniques.

1.Supervised Learning:

In supervised learning, the training set you feed to the algorithm includes the desired solutions, called labels

A typical supervised learning task is classification. The spam filter is a good example of this.

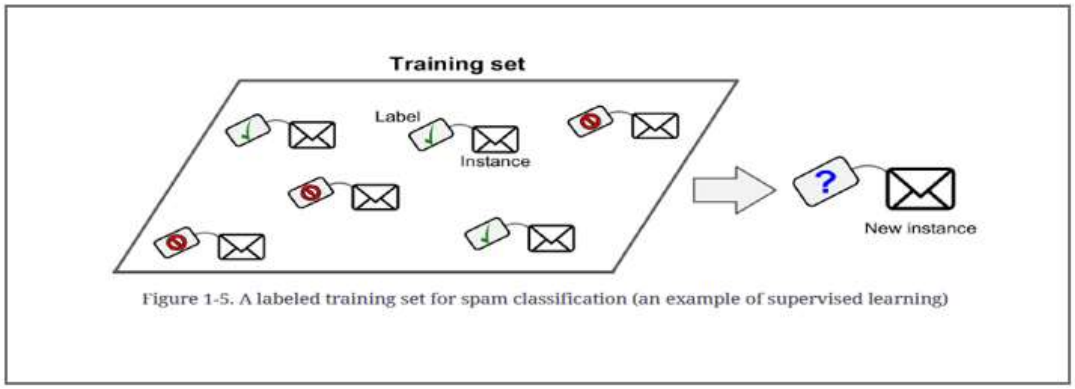


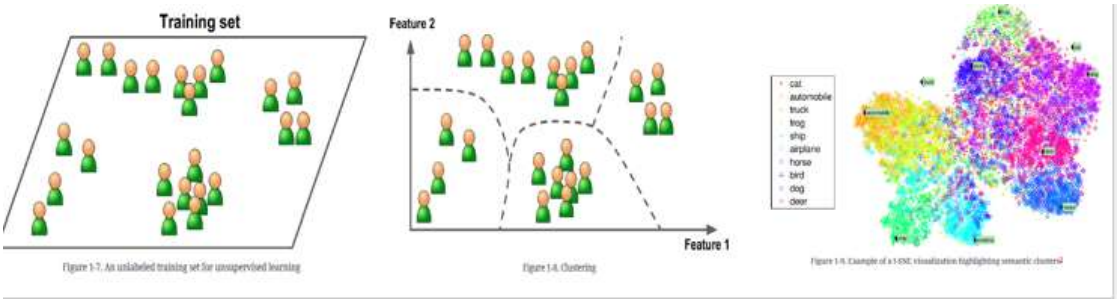
Figure 1-5. A labeled training set for spam classification (an example of supervised learning)

2.Un Supervised Learning:

In unsupervised learningthe training data is unlabelled (Figure 1-7). The system tries to learn without a teacher.

Visualization algorithms are also good examples of unsupervised learning algorithms: you feed them a lot of complex and unlabelled data, and they output a 2D or 3D representation of your data that can easily be plotted.

Another important unsupervised task is anomaly detection—for example, detecting unusual credit card transactions to prevent fraud.



3.Semi Supervised Learning:

Some algorithms can deal with partially labelled training data, usually a lot of unlabelled data and a little bit of labelled data. This is called semi supervised learning.

Most semi supervised learning algorithms are combinations of un supervised and supervised algorithms.

Google Photos, are good examples of this. Once you upload all your family photos to the service, it automatically recognizes that the same person A shows up in photos 1, 5, and 11,while another person B shows up in photos 2, 5, and 7. This is the unsupervised part of the algorithm (clustering). Now all the system needs is for you to tell it who these people are. Just add one label per person and it is able to name everyone in every photo, which is useful for searching photos

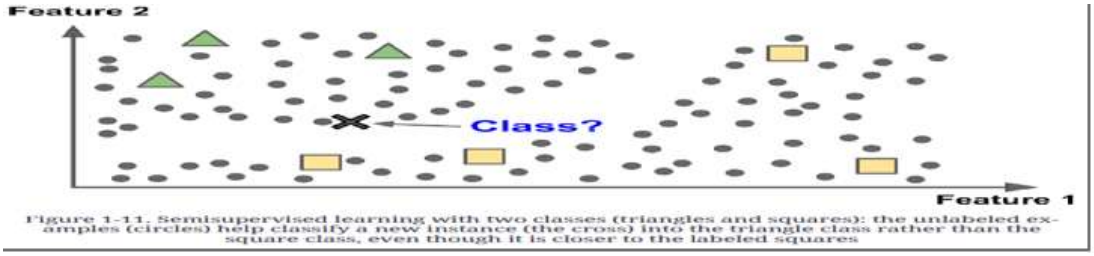


Figure 1-11. Semisupervised learning with two classes (triangles and squares): the unlabeled examples (circles) help classify a new instance (the cross) into the triangle class rather than the square class, even though it is closer to the labeled squares

4.Self Supervised Learning:

Another approach to machine learning involves actually generating a fully labelled dataset from a fully unlabelled one. Again, once the whole dataset is labelled, any supervised learning algorithm can be used. This approach is called self-supervised learning.

For example→ if you have a large dataset of unlabelled images, you can randomly mask a small part of each image and then train a model to recover the original image (figure 1-12).The resulting model may be quite useful in itself- for example, to repair damaged images or to erase unwanted objects from pictures.

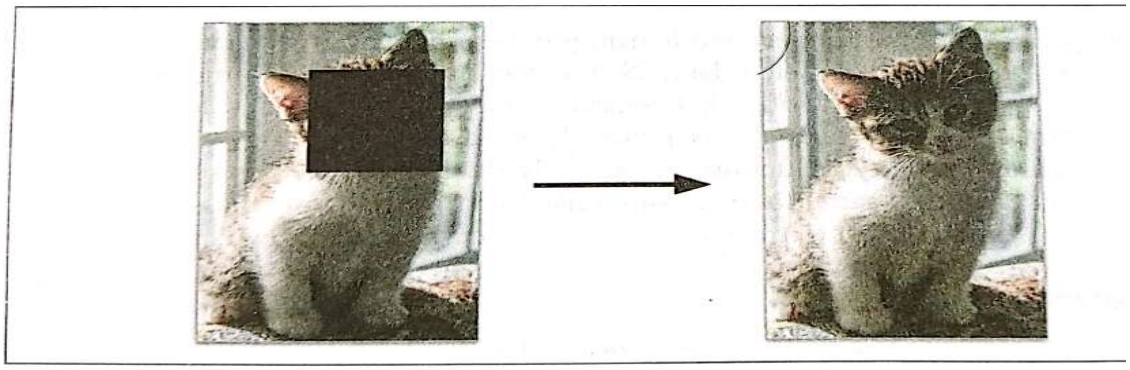


Figure 1-12. Self-supervised learning example: input (left) and target (right)

5.Reinforcement Learning:

Reinforcement learning (RL) is a machine learning technique that teaches software to make decisions that produce the best results.

The learning system, called an agent in this context, can observe the environment, select and perform actions, and get rewards in return.

It must then learn by itself what is the best strategy, called a policy, to get the most reward over time. A policy defines what action the agent should choose when it is in a given situation.

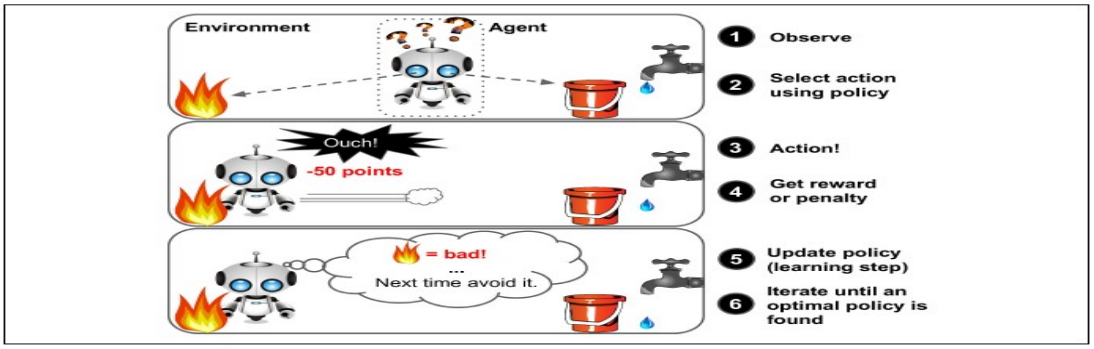


Figure 1-12. Reinforcement Learning

2b).Build a simple Linear Regression Model for the given sample data. And use the model to predict the price of a house with area of 100 Square Yards

Area(In square yards)	House-Price(In Lakhs)
70	26
75	28
80	32
85	38
90	40
95	45

Q2(b) Build a simple Linear Regression Model for the given sample data. And use the model to predict the price of a house with area of 100 Square Yards.

Given Data :

Area (In square yards) :	70	75	80	85	90	95
House-Price (In Lakhs) :	26	28	32	38	40	45

Step 1 : Prepare the table

X (Area)	Y (Price)	X^2	XY
70	26	4900	1820
75	28	5625	2100
80	32	6400	2560
85	38	7225	3230
90	40	8100	3600
95	45	9025	4275
$\Sigma X = 495$	$\Sigma Y = 209$	$\Sigma X^2 = 41275$	$\Sigma XY = 17585$

$n = 6$

Step 2 : Calculate slope (b)

$$b = \frac{n\Sigma XY - \Sigma X \Sigma Y}{n\Sigma X^2 - (\Sigma X)^2}$$

$$= \frac{6(17585) - (495)(209)}{6(41275) - (495)^2}$$

$$= \frac{105510 - 103455}{247650 - 245025}$$

$$= \frac{2055}{2625} = 0.783$$

Step 3 : Calculate intercept (a)

$$a = \frac{\Sigma Y - b\Sigma X}{n}$$

$$= \frac{209 - (0.783)(495)}{6} = \frac{209 - 387.585}{6}$$

$$= \frac{-299.585}{6} = \frac{-178.585}{6} = -29.764$$

Step 4 : Regression equation

$$Y = a + bX \qquad Y = -29.764 + 0.783X$$

Step 5 : Predict the price for area = 100 square yards

For $X = 100$,

$$Y = -29.764 + 0.783(100)$$
$$= -29.764 + 78.3$$
$$= 48.536$$

Answer : The predicted price of a house with area of 100 square yards is 48.54 Lakhs (approx).

(OR)

3a).What are Gradient decent algorithms? Explain different types of Gradient decent algorithms?

Gradient Descent is a very generic optimization algorithm capable of finding optimal solutions to a wide range of problems.

The general idea of Gradient Descent is to tweak parameters iteratively in order to minimize a cost function.

Three Types of gradient decent algorithm

1) Batch Gradient Descent Algorithm
2) Stochastic Gradient Descent Algorithm
3) Mini-Batch Gradient Descent Algorithm

1).Batch gradient decent algorithm:

In this variant, the entire training dataset is used to calculate the gradient and update the parameters**.**

In batch gradient descent, the gradient of the loss function (MSE) is computed with respect to the weights for the entire training dataset, and the weights are updated after eachiteration. This provides a more accurate estimate of the gradient, but it can be computationally expensive for large datasets.

Equation 4-5. Partial derivatives of the cost function

$$\frac{\partial}{\partial \theta_j}\text{MSE}(\boldsymbol{\theta}) = \frac{2}{m}\sum_{i=1}^{m}\left(\boldsymbol{\theta}^T\mathbf{x}^{(i)} - y^{(i)}\right)x_j^{(i)}$$

Batch gradient decent algorithm:

eta= 0.1 # learning rate

theta = np.random.randn(2,1) # random initialization

Gradient=0

for iteration in range(n_iterations):

thetaX= dot(X_train,theta)

gradient = gradient+((thetaX- y).X_train)

Gadient=gradient/n

theta = theta - eta * gradient

$$\theta^{(\text{next step})} = \theta - \eta \nabla_\theta \text{MSE}(\theta)$$

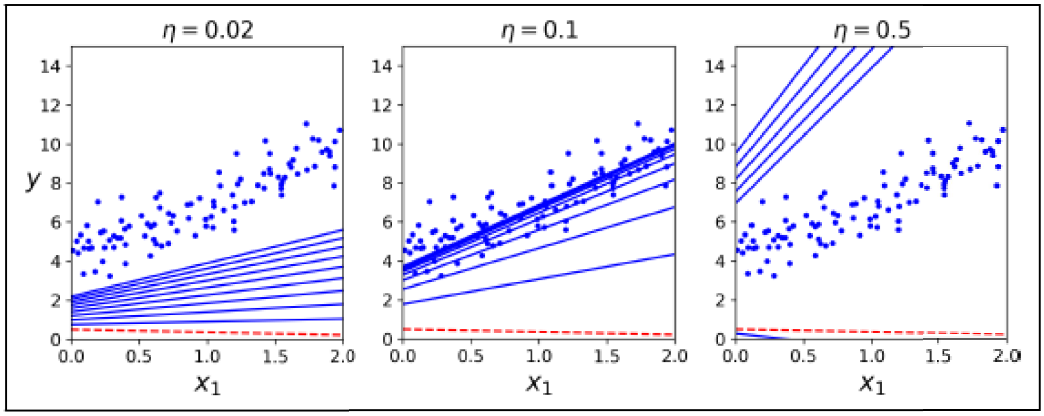


Figure 4-8. Gradient Descent with various learning rates

On the left, the learning rate is too low it will take a long time. In the middle, the learning rate looks pretty good.On the right, the learning rate is too high.

2).Stochastic Gradient Descent Algorithm:

Stochastic Gradient Descent just picks a random instance in the training set at every step and computes the gradients based only on that single instance. Obviously this makes the algorithm much faster since it has very little data to manipulate at every iteration. It also

makes it possible to train on huge training sets, since only one instance needs to be in memory at each iteration.

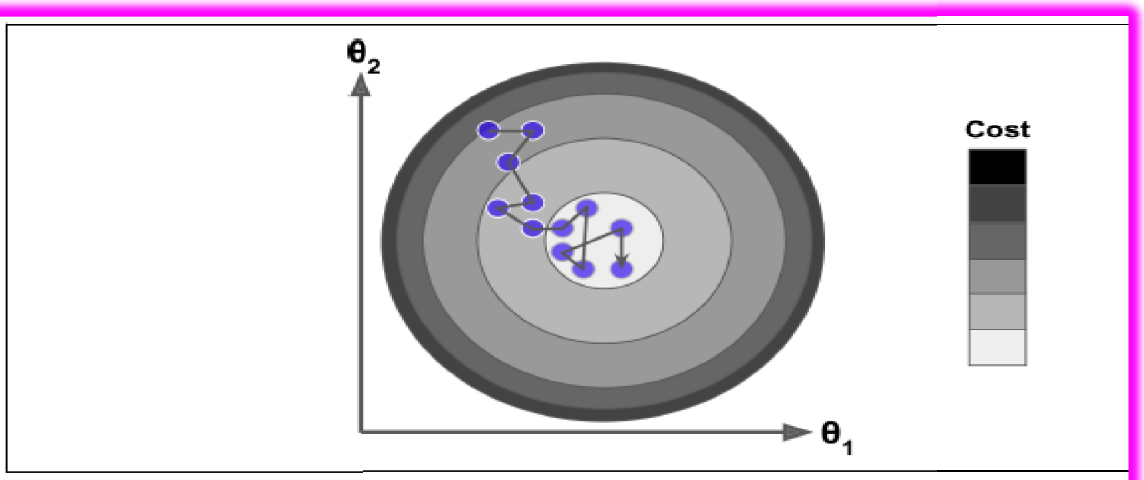


Figure 4-9. Stochastic Gradient Descent

Stochastic gradient decent algorithm:

eta = 0.1 # learning rate

theta = np.random.randn(2,1) # random initialization

for iteration in range(n_iterations):

thetaX= dot(X_train,theta)

gradient = 2/m*(thetaX- y).X_train)

theta = theta - eta * gradient

$\theta = \theta - \eta \nabla J(\theta; x^{(i)}, y^{(i)})$

where:

- θ represents the parameters of the model,
- $\eta$is the learning rate,
- $\nabla J(\theta; x^{(i)}, y^{(i)})$ is the gradient of the loss function$J(\theta)$ with respect to θ for the i-th training example $(x^{(i)}, y^{(i)})$.

3).Mini-Batch Gradient Descent Algorithm:

At each step in Mini-Batch Gradient Descent, instead of computing the gradients based on the full training set (as in Batch GD) or based on just one instance (as in Stochastic GD), MinibatchGD computes the gradients on small random sets of instances called *minibatches*.

The main advantage of Mini-batch GD over Stochastic GD is that you can get a performance boost from hardware optimization of matrix operations, especially when using GPUs.

$$\theta = \theta - \eta \nabla J(\theta; \{x^{(i)}, y^{(i)}\}_{i=1}^{m})$$

where:

- θ represents the parameters of the model,
- $\eta$is the learning rate,
- $\{x^{(i)}, y^{(i)}\}_{i=1}^{m}$represents a mini-batch of mmm training examples,
- $\nabla J(\theta; \{x^{(i)}, y^{(i)}\}_{i=1}^{m})$ is the gradient of the loss function $J(\theta)$ with respect to θ for the mini-batch.

3b).Explain Ridge Regression and Lasso Regression.

1.Ridge Regression:

Ridge regression is a type of regression that involves adding a penalty term to the cost function to prevent overfitting. The penalty term helps to reduce the magnitude of the coefficients of the input variables.

2.The hyperparameter α controls how much you want to regularize the model. If $\alpha = 0$ then Ridge Regression is just Linear Regression. If α is very large, then all weights end up very close to zero and the result is a flat line going through the data's mean

Equation 4-8. Ridge Regression cost function

$$J(\theta) = \text{MSE}(\theta) + \alpha \frac{1}{2} \sum_{i=1}^{n} \theta_i^2$$

Ridge Regression is most suitable when all predictors are expected to contribute to the outcome and none should be excluded from the model. It reduces overfitting by shrinking

the coefficients, ensuring they don't become too large, while still keeping all the predictors in the model.

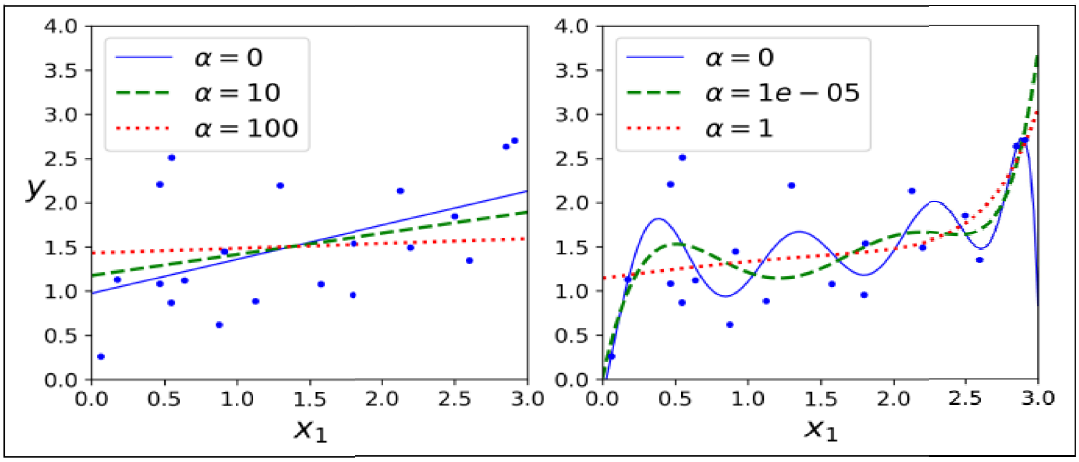


Figure 4-17. Ridge Regression

2).**Lasso Regression:**

Least Absolute Shrinkage and Selection Operator Regression (simply called *Lasso Regression*) is another regularized version of Linear Regression: just like Ridge Regression, it adds a regularization term to the cost function, but it uses the ℓ_1 norm of the weight vector instead of half the square of the ℓ_2 norm (see Equation 4-10).

Lasso Regression cost function=>J(θ)=MSE(θ)+$2\alpha\sum_{i=1}|\theta_i|$

For example, in genetic research, where thousands of genes are analyzed for their effect on a disease, Lasso Regression helps by identifying only the most impactful genes and ignoring the irrelevant ones, leading to a simpler and more interpretable model.

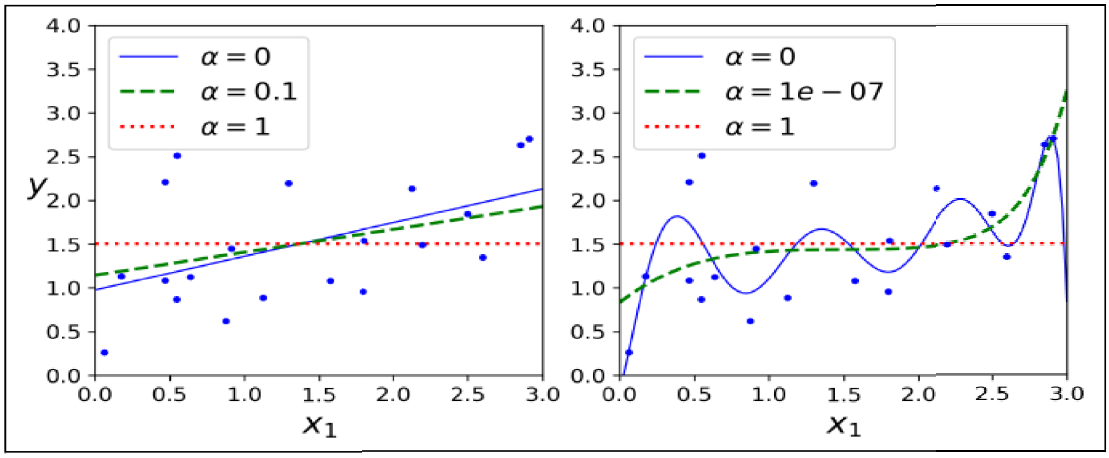


Figure 4-18. Lasso Regression

Unit-II

4a).What is classification. Explain the different types of classifiers with examples.

Logistic Regression:

Logistic Regression is a classification algorithm used to predict binary outcomes such as Pass/Fail or Yes/No. It uses the sigmoid function to estimate probabilities.

Example: Predict whether a student will **Pass** or **Fail** based on study hours.

Decision Tree:

A Decision Tree is a tree-structured classifier where internal nodes represent decisions and leaf nodes represent outcomes. It is easy to understand and interpret.

Ex:Play Tennis prediction based on attributes like Outlook, Temperature, Humidity, and Wind.

Naïve Bayes:

Naïve Bayes is a probabilistic classifier based on Bayes' theorem. It assumes that all features are independent and is widely used for spam filtering and text classification.

Example:Play Tennis dataset – predicting whether tennis will be played based on weather conditions.

Support Vector Machine (SVM):

SVM is a powerful classifier that finds the optimal hyperplane to separate different classes while maximizing the margin between them.

Example: Separating data points of two classes using an **optimal hyperplane**.

K-Nearest Neighbors (KNN):

KNN classifies a new data point based on the majority class of its K nearest neighbors. It is simple and effective for small dataset.

Example: Classify a flower as Rose, Lily, or Sunflower based on similar flowers nearby.

Random Forest:

Random Forest is an ensemble learning method that combines multiple decision trees and uses majority voting to make accurate predictions while reducing overfitting.

Example: Predict whether a patient has a disease using multiple decision trees.

4b).Explain in detail about CART algorithm with an example.

- **CART (Classification and Regression Trees)** is a popular algorithm used in machine learning for both classification and regression tasks.
- It works by recursively partitioning the training data into subsets based on the values of features, aiming to create a decision tree that can predict the target variable.
- **Initial Split:** CART begins by examining all the features in the dataset and selecting the one that best separates the target class(classification) or has the strongest correlation with the target class (regression).
- This is done by evaluating a splitting criterion such as Gini index for classification or mean squared error for regression.
- **Recursive Partitioning:** Once CART has selected the best feature to split on, it divides the dataset into two subsets based on a threshold / value for that feature.
- This creates two child nodes in the decision tree with each node associated with a subset.
- Partitioning continues on each node till the samples in a node belong to the same target class.

CART Algorithm

- **Gini index/impurity** is a measure of impurity in the data set with respect to the distribution of the samples in a dataset as a fraction of the target class.
- The formula for Gini Index (GI) is
- $GI(D) = 1 - \sum_{i=1}^{n} p_i^2$,

where n is the number of classes and p_i is the probability of the ith calss

- $GI(D) = 1 - \left(\frac{4}{6}\right)^2 - \left(\frac{2}{6}\right)^2 = 1 - \frac{20}{36} = \frac{16}{36} = 0.44$
- In the CART algorithm we choose a parameter/ attribute which decreases the impurity of the partitions of the dataset made by a particular that feature.
- For example, let us consider the Job Type as the feature.
- It will divide the dataset into three sets of samples with the value as Teacher, Engineer, Doctor,

#	Job Type	Age	Buys
D1	Teacher	Young	No
D2	Engineer	Middle	No
D3	Doctor	Middle	No
D4	Teacher	Young	No
D5	Engineer	Young	Yes
D6	Doctor	Young	Yes

CART Algorithm

- $GI(Job\ Type = Teacer) = 1 - \left(\frac{0}{2}\right)^2 - \left(\frac{2}{2}\right)^2 = 1 - 1 = 0$

(since $GI(D) = 1 - \sum_{i=1}^{n} p_i^2$,)

- $GI(Job\ Type = Engineer) = 1 - \left(\frac{1}{2}\right)^2 - \left(\frac{1}{2}\right)^2$ =1-0.5=0.5
- $GI(Job\ Type = Teacer) = 1 - \left(\frac{1}{2}\right)^2 - \left(\frac{1}{2}\right)^2$ =0.5
- $Weighted\ GI(Job\ Type)$

$= \sum Weighted\ average\ *Gini\ index$

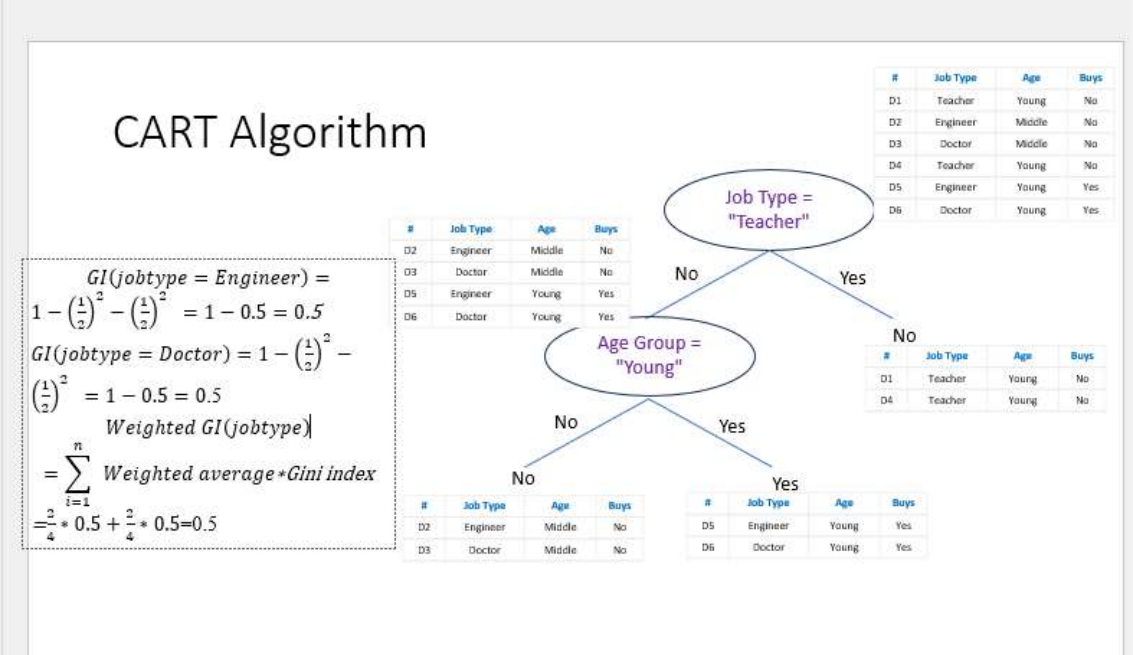
$= \frac{2}{6} * 0 + \frac{2}{6} * 0.5 + \frac{2}{6} * 0.5 = 0.33$

Job Type	Yes	No
Teacher	0	2
Engineer	1	1
Doctor	1	1

#	Job Type	Age	Buys
D1	Teacher	Young	No
D4	Teacher	Young	No

#	Job Type	Age	Buys
D2	Engineer	Middle	No
D5	Engineer	Young	Yes

#	Job Type	Age	Buys
D3	Doctor	Middle	No
D6	Doctor	Young	Yes



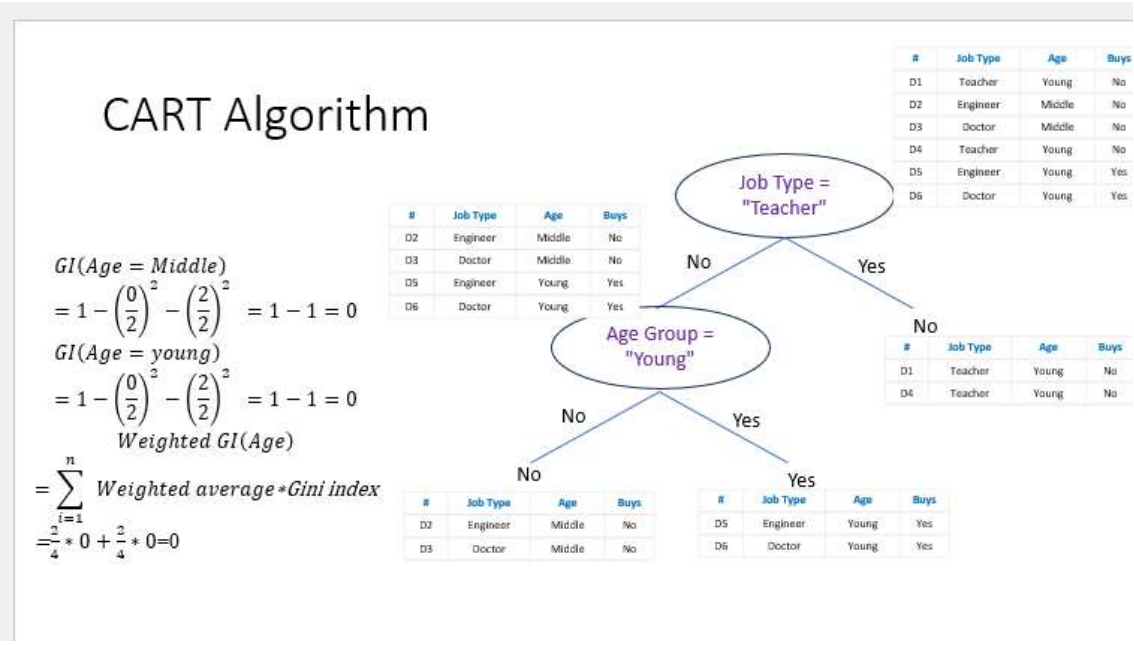
#	Job Type	Age	Buys
D1	Teacher	Young	No
D2	Engineer	Middle	No
D3	Doctor	Middle	No
D4	Teacher	Young	No
D5	Engineer	Young	Yes
D6	Doctor	Young	Yes

#	Job Type	Age	Buys
D2	Engineer	Middle	No
D3	Doctor	Middle	No
D5	Engineer	Young	Yes
D6	Doctor	Young	Yes

#	Job Type	Age	Buys
D1	Teacher	Young	No
D4	Teacher	Young	No

#	Job Type	Age	Buys
D2	Engineer	Middle	No
D3	Doctor	Middle	No

#	Job Type	Age	Buys
D5	Engineer	Young	Yes
D6	Doctor	Young	Yes



#	Job Type	Age	Buys
D1	Teacher	Young	No
D2	Engineer	Middle	No
D3	Doctor	Middle	No
D4	Teacher	Young	No
D5	Engineer	Young	Yes
D6	Doctor	Young	Yes

#	Job Type	Age	Buys
D2	Engineer	Middle	No
D3	Doctor	Middle	No
D5	Engineer	Young	Yes
D6	Doctor	Young	Yes

#	Job Type	Age	Buys
D1	Teacher	Young	No
D4	Teacher	Young	No

#	Job Type	Age	Buys
D2	Engineer	Middle	No
D3	Doctor	Middle	No

#	Job Type	Age	Buys
D5	Engineer	Young	Yes
D6	Doctor	Young	Yes

(OR)

5a).Explain about Logistic Regression with python code Implementation.

Logistic Regression:

Logistic regression is a type of regression used for classification problems, where the output variable is categorical in nature. Logistic regression uses a logistic function to predict the probability of the input belonging to a particular category.

Logistic regression (logit regression) is a type of regression analysis used for predicting the outcome of a categorical dependent variable.

In logistic regression, dependent variable (Y) is binary (0,1) and independent variables (X) are continuous in nature.

Sigmoid Function:

It is a mathematical function having a characteristic that can take any real value and map it to between 0 to 1 shaped like the letter "S". The sigmoid function also called a logistic function. $Y = 1 / 1+e^{-z}$.

 # Logistic Regression using Iris Dataset

```
from sklearn.datasets import load_iris

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LogisticRegression

from sklearn.metrics import accuracy_score

iris = load_iris()

X = iris.data

y = iris.target

X_train, X_test, y_train, y_test = train_test_split( X, y, test_size=0.2, random_state=42)

model = LogisticRegression(max_iter=200)

model.fit(X_train, y_train)

y_pred = model.predict(X_test)

print("Accuracy:", accuracy_score(y_test, y_pred))
```

5b).Consider the following data set.

Confident	Studied	Sick	Result
Yes	No	No	Fail
Yes	No	Yes	Pass
No	Yes	Yes	Fail
No	Yes	No	Pass
Yes	Yes	Yes	Pass

Find out whether the object with attribute **Confident = Yes, Sick = No** will Fail or Pass using Naïve Bayes Classification.

1. $P(\text{Fail} \mid \text{yes}, \text{NO}) = P(\text{fail}) \times P(\text{confident} = \text{yes} \mid \text{fail}) \; P(\text{sick} = \text{NO} \mid \text{fail})$

$= 0.4 \times (1/2) \times (1/2)$

$= 0.4 \times 0.25$

$= 0.100$

Since, $0.133 > 0.100$, the Naive Bayes classifier assigns the higher probability to pass.

2. Naive Bayes classification:

$$V_{NB} = \underset{v_j \in V}{\text{argmax}} \; P(v_j) \prod_i P(a_i \mid v_j)$$

confident yes/no	studied NO/yes	sick NO/yes	result fail/pass
yes	NO	NO	fail
yes	NO	yes	pass
NO	yes	yes	fail
NO	yes	yes	pass
yes	yes	yes	pass

$P(\text{pass}) = 3/5 = 0.6$

$P(\text{fail}) = 2/5 = 0.4$

3. Calculate Likelihoods:

Attribute	Value	P(value | pass)
confident	yes	$2/3 \approx 0.667$
sick	NO	$1/3 \approx 0.333$

4. Apply Naive Bayes:

$P(\text{pass} \mid \text{yes}, \text{NO}) = P(\text{pass}) \times P(\text{confident} = \text{yes} \mid \text{pass}) \; P(\text{sick} = \text{NO} \mid \text{pass})$

$= 0.6 \times (2/3) \times (1/3) = 0.6 \times 0.222$

$= 0.133$

$P(\text{fail} \mid \text{yes}, \text{NO}) = P(\text{fail}) \times P(\text{confident} = \text{yes} \mid \text{fail}) \; P(\text{sick} = \text{NO} \mid \text{fail})$

$= 0.4 \times (1/2) \times (1/2) = 0.4 \times 0.25$

$= 0.100$

Since, $0.133 > 0.100$, the Naive Bayes classifier assigns the higher probability to pass.

Unit-III

6a).What is Confusion Matrix? Explain in detailed about Confusion Matrix.

CONFUSION MATRIX:

- The confusion matrix is a matrix used to determine the performance of the classification models for a given set of test data.

			Predicted values		
			Positive	**Negative**	Totals
	Actual Values	**Positive**	TP	FN	**P = (TP + FN) = Actual Total Positives**
		Negative	FP	TN	**N = (FP + TN) = Actual Total Negatives**
		Totals	Predicted Total Positives	Predicted Total Negatives	

The four possible outcomes of a binary classification problem are:

True Positive (TP): The model predicted the positive class correctly, and the true class is also positive.

False Positive (FP): The model predicted the positive class incorrectly, and the true class is negative.

True Negative (TN): The model predicted the negative class correctly, and the true class is also negative.

False Negative (FN): The model predicted the negative class incorrectly, and the true class is positive

The terminology used in a confusion matrix includes:

1. **Accuracy:**

→The proportion of correctly classified instances out of the total number of instances.

$$\text{Accuracy} \quad \frac{TP + TN}{(TP + TN + FP + FN)}$$

2.Precision:

→The proportion of correctly predicted positive instances out of the total number of instances predicted as positive.

$$\text{Precision} \quad \frac{TP}{(TP + FP)}$$

3.Recall OR Sensitivity OR True Positive Rate:

→The proportion of correctly predicted positive instances out of the total number of actual positive instances.

$$\text{Sensitivity} \quad \frac{TP}{(TP + FN)}$$

3.Specificity:

→The proportion of correctly predicted negative instances out of the total number of actual negative instances.

$$\text{Specificity} = \frac{TN}{(TN + FP)}$$

5.F1-score:

→The harmonic mean of precision and recall

$$F_1\text{-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2TP}{2TP + FP + FN}$$

Example

n=165	Predicted: NO	Predicted: YES	
Actual: NO	TN = 50	FP = 10	60
Actual: YES	FN = 5	TP = 100	105
	55	110	

Recall OR Sensitivity OR TPR

$= \frac{TP}{TP+FN} \Rightarrow \frac{100}{100+5} \Rightarrow 0.95$

$\text{Specificity} = \frac{TN}{TN+FP} \rightarrow \frac{50}{50+10} \rightarrow 0.833$

$\text{FPR} = \frac{FP}{TN+FP} \Rightarrow \frac{10}{50+10} \Rightarrow 0.166$

$\text{Accuracy} = \frac{TN \mid TP}{T \mid P} \Rightarrow \frac{50 \mid 100}{50 \mid 10 \mid 5 \mid 100} \Rightarrow 0.90$

Error rate= 1 – accuracy
= 1 – 0.90
= 0.1

$\text{Precision} = \frac{TP}{TP+FP} \rightarrow \frac{100}{100+10} \rightarrow 0.909$

6b).Explain cross-validation in detail. Discuss different types and their advantages.

CROSS VALIDATION:

A good way to evaluate a model is to use cross-validation.

cross_val_score() function to evaluate cross validation.

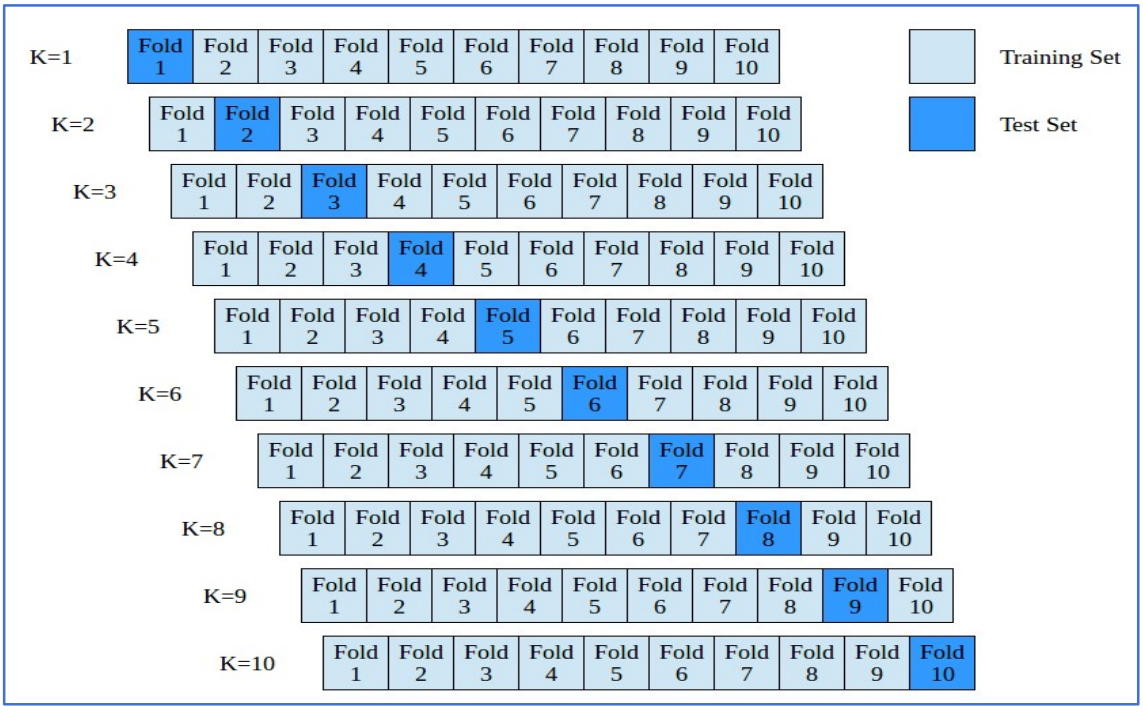
The entire dataset is used for the cross-validation techniques.

Cross validation is used for perform the model on **unseen data**

TYPES OF CROSS-VALIDATION:

1.K FOLD CROSS VALIDATION:

- It is used to divide the dataset into K Folds. , each and every iteration one fold for testing and remaining another folds for training dataset is called K Fold Cross validation.

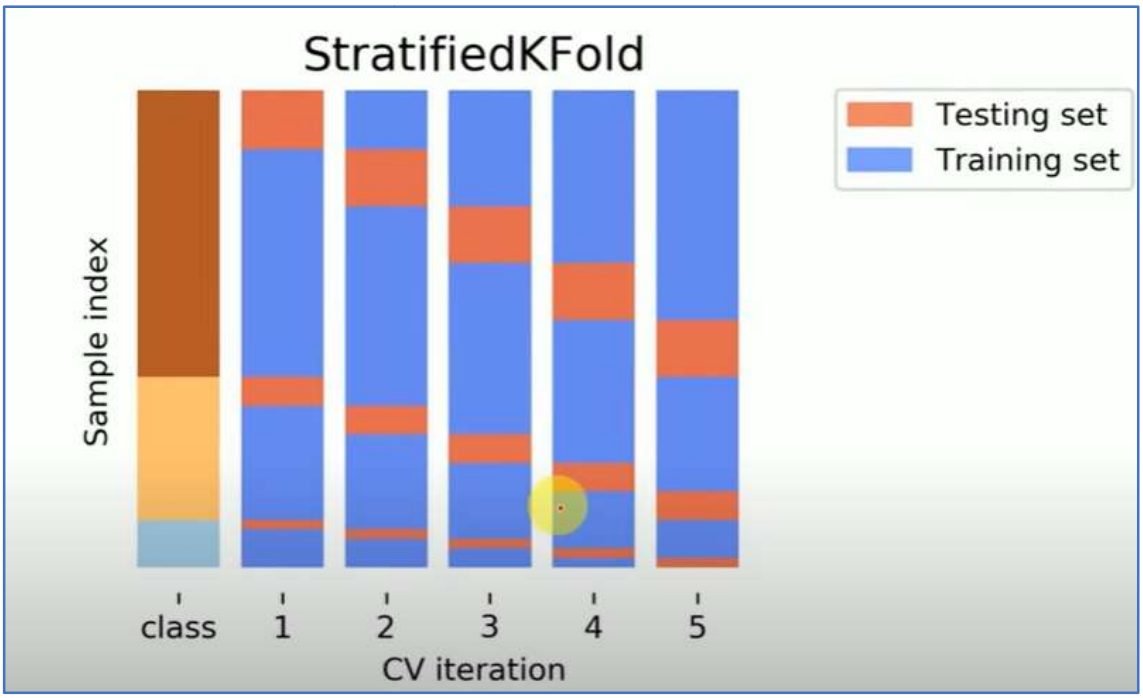


Advantages:

- Better performance estimation.
- Efficient use of data.

2.STRATIFIED K FOLD CROSS VALIDATION:

- It is used to divide the dataset into stratified K Folds. , each and every iteration is small partition each class for testing and another for training called Stratified K Fold Cross validation.

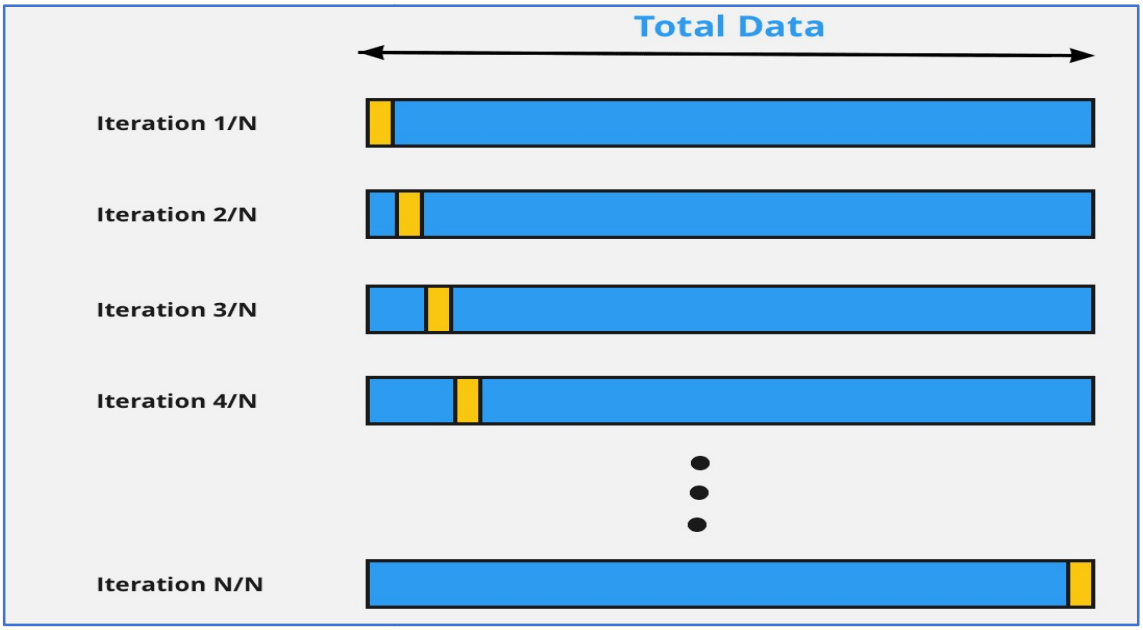


Advantages:

- Works well for imbalanced datasets.
- Produces more reliable results.

3.LEAVE-ONE-OUT CROSS VALIDATION:

→ The model is trained on n−1samples and tested on the one sample.

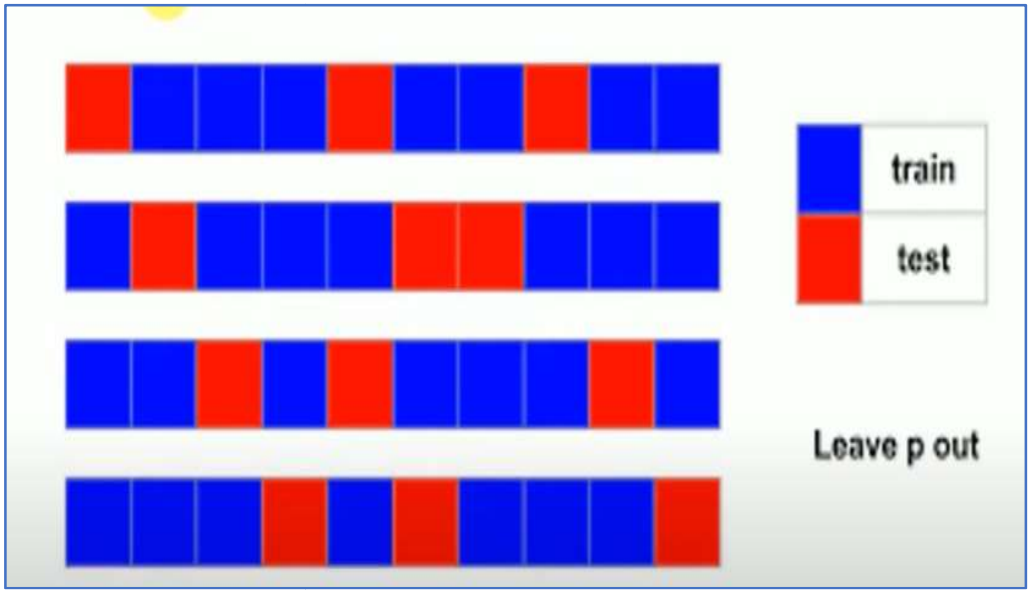


Advantages:

- Uses almost all data for training.
- Less biased estimate.

4.LEAVE-P-OUT CROSS VALIDATION:

→ The model is trained on n−psamples and tested on the p samples.

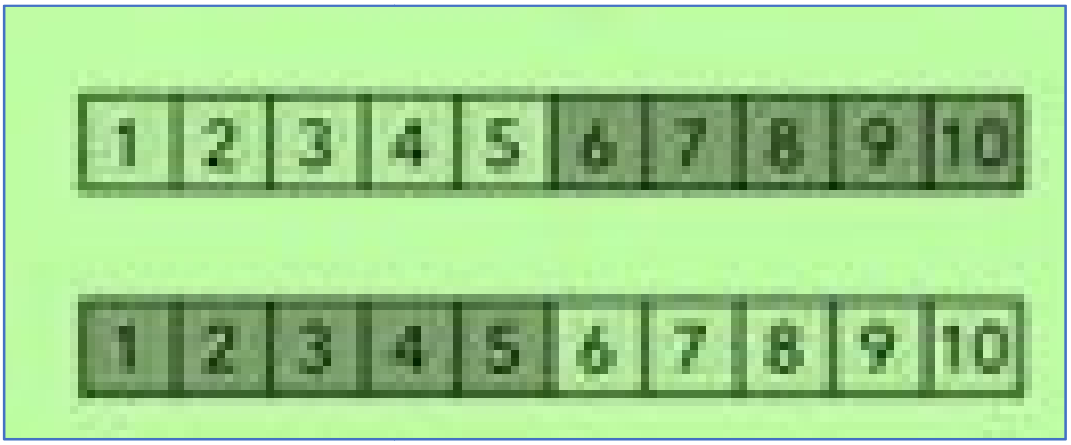


Advantages:

- Provides accurate performance estimates.
- Makes maximum use of data

5.HOLD OUT CROSS VALIDATION:

- We perform training on the 50% of the given dataset and rest 50% is used for the testing purpose. It's a simple and quick way to evaluate a model.

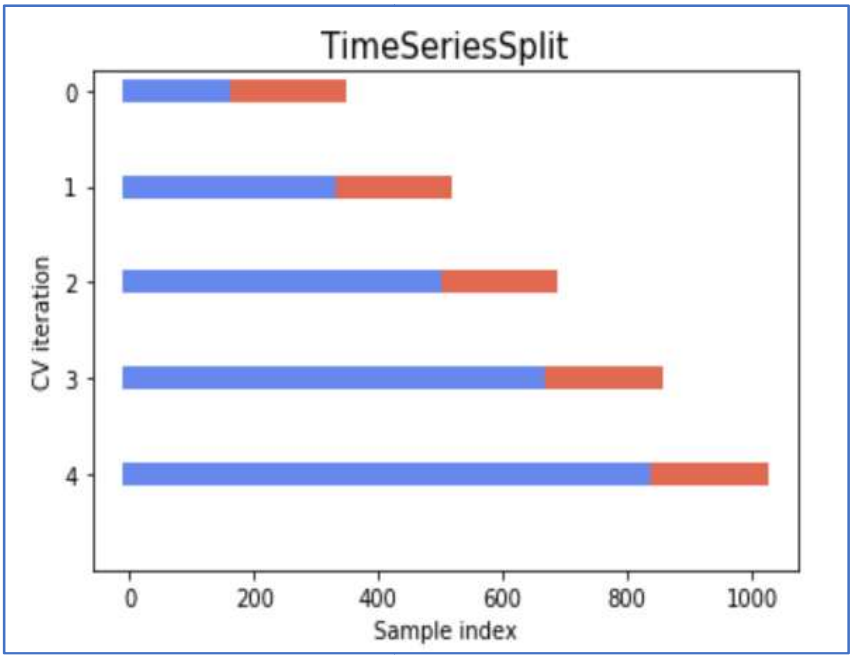


Advantages:

- Simple and easy to implement.
- Fast execution.

6.TIME SERIES CROSS VALIDATION:

The method that can be used for cross-validating the time-series model is cross-validation on a rolling basis.



Advantages:

- Prevents data leakage.
- Suitable for forecasting problems

(OR)

7A).Explain ensemble learning and its importance in machine learning.

ENSEMBLE LEARNING:

- Combinations of models are generally known as *model ensembles*.
- A group of predictors is called an *ensemble*thus, this technique is called *Ensemble Learning*.

(OR)

- *An ensembled model is a machine learning model that combines the predictions from two or more models."*

There are 3 most common ensemble learning methods in machine learning. These are as

- **Simple Ensemble Techniques**
- **Bagging:**

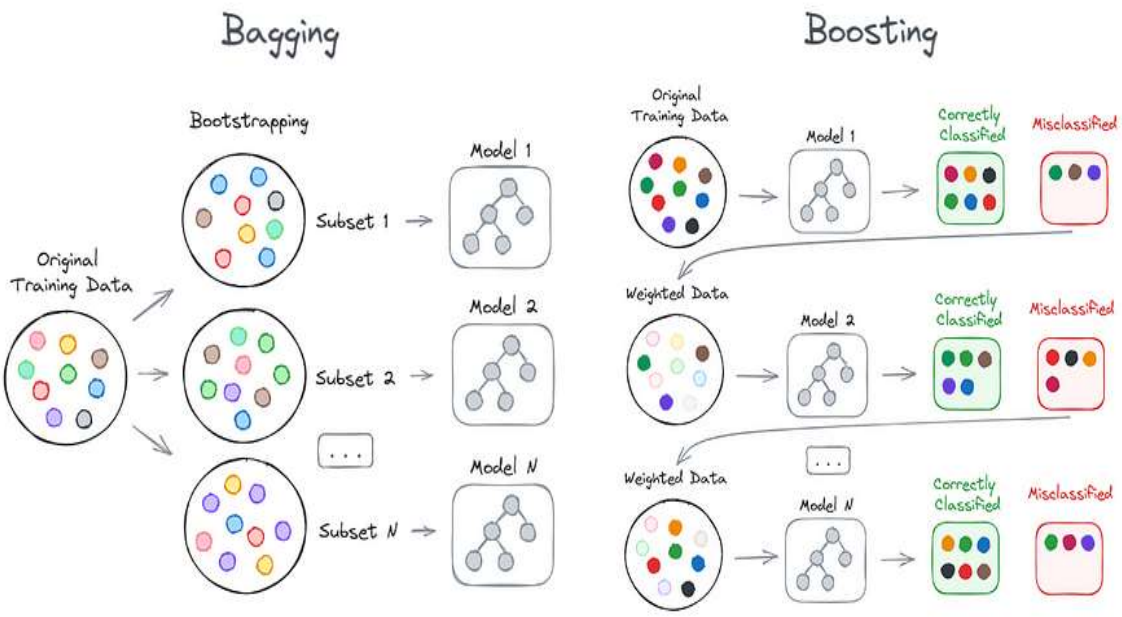
  Various training data subsets are randomly drawn with replacement from the whole training dataset.

  Bagging attempts to tackle the over-fitting issue.

- **Boosting:**

  Each new subset contains the components that were misclassified by previous models.

Boosting tries to reduce bias



Importance of Ensemble Learning

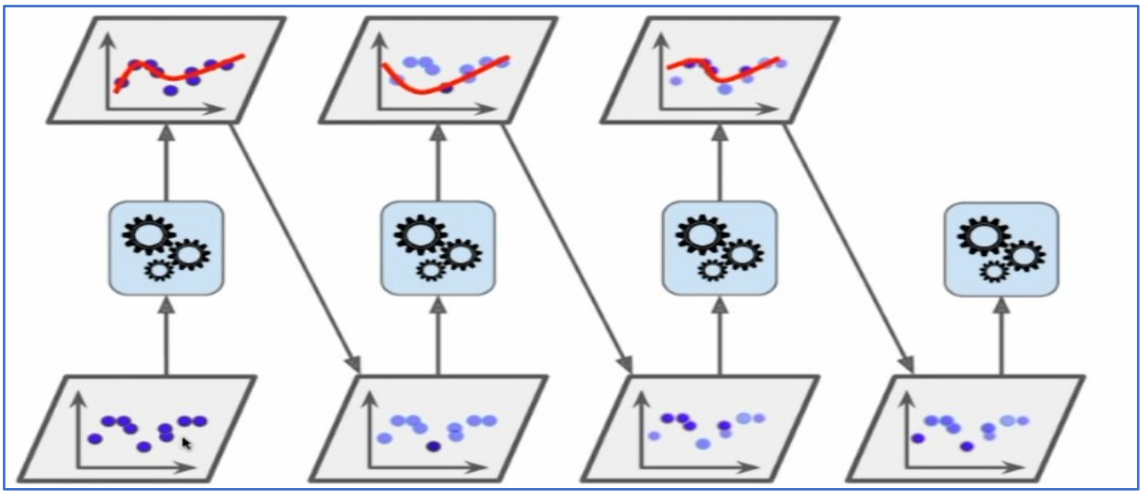
- Increases prediction accuracy.
- Reduces overfitting.
- Improves model stability and robustness.
- Handles complex datasets effectively.

- Provides better generalization on unseen data.

7b)Describe the working of AdaBoost with an example.

ADABOOST:

- AdaBoost short for Adaptive Boosting is an ensemble learning used in machine learning for **classification and regression problems.**
- AdaBoost aims to combine multiple weak learners by adjusting weights on misclassified examples, creating a stronger classifier that focuses on difficult instances.



WORKING:

STEP1: Initial weight assigned for each item = 1/n

STEP2: Iterating each weak classifier.

We have to calculate the decision stump for each attribute.

2a) By selecting instances of the attribute, we may predict '**YES**' or else '**NO**'.

2b)compute weighted error for the hypothesis of attribute.

Here:

$H_l(d_j)$ ==hypothesis of data ,**$H_l(d_j) = 0$** → wrong classification, **$H_l(d_j) = 1$** → **correct**

Wt(dj)=weight of sample.

2c) compute the **weight for each classifier.**

$\alpha=\frac{1}{2}\ln\left(\frac{(1-eattribute)\ )}{eattribut}\right)$

2d) calculate the **normalizing factor.**

$Z_{attribute}$= wt(correctly classified instances) * no of correctly classified instances * $e^{-\alpha}$ + wt(wrongly classified instances) * no of wrongly classified instances * e^{α}

2(e) :Update the weights for all data instances.

$$wt(c) = \frac{wt(c)attribute\ of correct\ instances * e^{-\alpha}}{Z_{attribute}}$$

$$wt(w) = \frac{wt(w)attribute\ of\ wrong\ instances * e^{\alpha}}{Z_{attribute}}$$

STEP3:

Compute the final predicted value for each data instance.

$$H_F(d_j) = \sum_{i=1}^{m} \propto_i * H_i(d_j)$$

ML SCHEME 2026

Unit-IV

8a). Write about sample complexity for finite hypothesis spaces

Sample Complexity For Finite Hypothesis Spaces:

- **Sample Complexity for Finite Hypothesis Spaces** is a concept from statistical learning theory that determines the number of training samples required to ensure that a learning algorithm can find a hypothesis close to the target function with high probability.

Key Terms

1. **Hypothesis Space (H)**:
    1. A set of all possible hypotheses (functions) that a learning algorithm.
2. **Target Concept**:
    1. The true underlying function or concept the algorithm tries to approximate.
3. **Error of a Hypothesis**:
    1. The difference between the hypothesis and the target concept.
4. **Sample Complexity Formula**
5. For a finite hypothesis space H, the number of samples m required to ensure PAC learning is given by:

$$m \geq \frac{1}{\epsilon}\left(\ln|\mathcal{H}| + \ln\frac{1}{\delta}\right)$$

Where:

- **ϵ: Desired accuracy (maximum allowable error).**

1. →Relates to the accuracy. A smaller error tolerance requires more samples.

- **δ: Desired confidence level (probability of success is 1−δ.**

→Reflects the confidence level. Higher confidence demands more samples.

- **|H|: Size of the hypothesis space (finite).**

→Relates to the accuracy. A smaller error tolerance requires more samples.

Problem Statement

The hypothesis space $\mathcal{H}$ contains $|\mathcal{H}| = 500$ hypotheses. You want to select a hypothesis $h \in \mathcal{H}$ such that the error ϵ (difference between the selected hypothesis and the true function) is at most 0.1, with a confidence level of $1 - \delta = 0.95$ ($\delta = 0.05$).

Question: How many training samples m are required to achieve this?

Solution

The formula for sample complexity is:

$$m \geq \frac{1}{\epsilon}\left(\ln|\mathcal{H}| + \ln\frac{1}{\delta}\right)$$

Step 1: Substitute the known values:

$$\epsilon = 0.1, \quad |\mathcal{H}| = 500, \quad \delta = 0.05$$

$$m \geq \frac{1}{0.1}\left(\ln 500 + \ln\frac{1}{0.05}\right)$$

Step 2: Compute the logarithmic terms:

1. $\ln 500 \approx 6.2146$
2. $\ln \frac{1}{0.05} = \ln 20 \approx 2.9957$

Add these:

$$\ln |\mathcal{H}| + \ln \frac{1}{\delta} = 6.2146 + 2.9957 = 9.2103$$

Step 3: Plug into the formula:

$$m \geq \frac{1}{0.1} \cdot 9.2103 = 10 \cdot 9.2103 = 92.103$$

Round up:

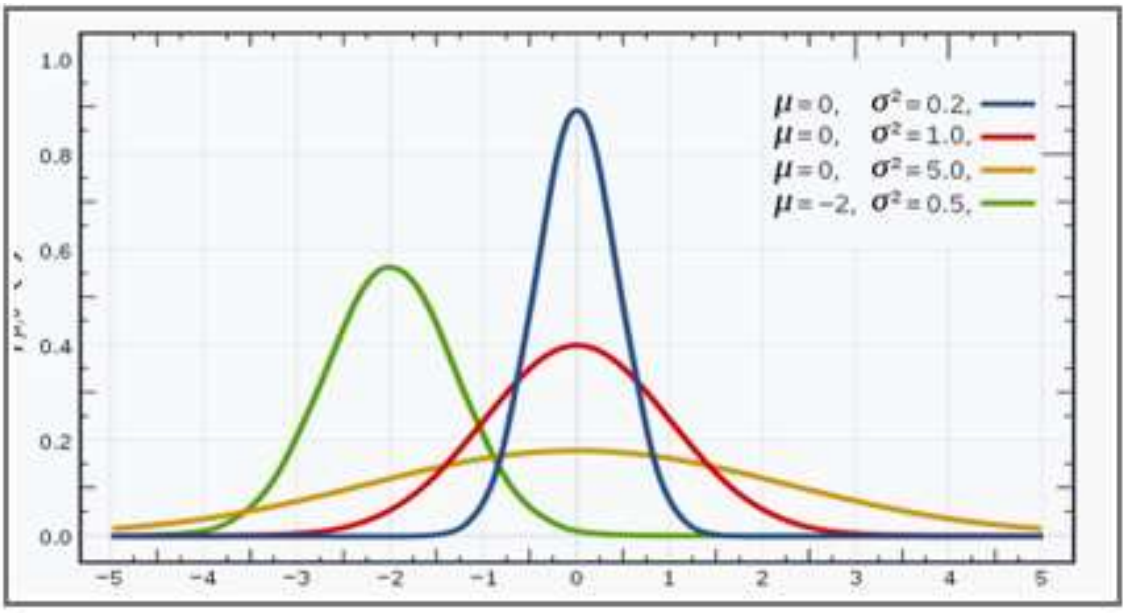
$$m \geq 93$$

❖ You need at least **93 training samples** to ensure that the hypothesis is within 0.1 error of the target function with a 95% confidence level.

8b).What is the Gaussian mixture model (GMM) in unsupervised learning? Explain the concept of representing the underlying data distribution as a mixture of Gaussian components and using the Expectation-Maximization (EM) algorithm to estimate the parameters?

GAUSSIAN MIXTURE MODELS:

- A Gaussian mixture model is a probabilistic model that assumes all the data points are generated from a mixture of a finite number of Gaussian distributions with unknown parameters.
- A normal distribution is a type of continuous probability distribution in which most data points cluster toward the middle of the range, while the rest taper off symmetrically toward either extreme.



ALGORITHM:

Step 1: Initialization

Assume K Gaussian components.

For each component k:

Initial means: μk (random guess or based on data clustering).

Initial variances: σ_k^2 (random or uniform values).

Mixing coefficients: $\pi_k = \frac{1}{k}$ (equal weights for all components).

Step 2: Expectation Step (E-Step)

For each data point, calculate the **responsibility** (E_P), which is the probability that the data point x_i belongs to the k-th Gaussian component.

Formula:

$$\gamma_{ik} = \frac{\pi_k \cdot \mathcal{N}(x_i|\mu_k, \sigma_k^2)}{\sum_{j=1}^{K} \pi_j \cdot \mathcal{N}(x_i|\mu_j, \sigma_j^2)}$$

Where:

- $\mathcal{N}(x_i|\mu_k, \sigma_k^2)$ is the Gaussian probability density function (PDF):

$$\mathcal{N}(x_i|\mu_k, \sigma_k^2) = \frac{1}{\sqrt{2\pi\sigma_k^2}} \cdot \exp\left(-\frac{(x_i - \mu_k)^2}{2\sigma_k^2}\right)$$

Step 3: Maximization Step (M-Step)

1. Update the Means (μ_k):

$$\mu_k = \frac{\sum_{i=1}^{n} \gamma_{ik} \cdot x_i}{\sum_{i=1}^{n} \gamma_{ik}}$$

2. Update the Variances (σ_k^2):

$$\sigma_k^2 = \frac{\sum_{i=1}^{n} \gamma_{ik} \cdot (x_i - \mu_k)^2}{\sum_{i=1}^{n} \gamma_{ik}}$$

3. Update the Mixing Coefficients (π_k):

$$\pi_k = \frac{\sum_{i=1}^{n} \gamma_{ik}}{n}$$

Step 4: Convergence Check

Statement:

Repeat Steps 2 and 3 until the parameters (μ_k, σ_k^2, π_k) converge (i.e., the change in values becomes very small).

(OR)

9a).Explain in detail about K-means Clustering Algorithm with an example.

K-means Clustering Algorithm:

- K-Means Clustering is 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. K means no of clusters.
- K-Means clustering is an unsupervised iterative clustering technique.
- *A centroid-based technique.*
- It is Distance Based Clustering method.
- Algorithm:
- Step 1: Select *K* points in the data space and mark them as initial centroids
- Loop
- Step 2: Assign each point in the data space to the nearest centroid to form K clusters
- Step 3: Measure the distance of each point in the cluster from the centroid
- Step 5: Identify the new centroid of each cluster on the basis of distance between points
- Step 6: Repeat Steps 2 to 5 to refine until centroids do not change (When the updated centroids from one iteration are the same (or very close to) the centroids from the previous iteration, the algorithm will stop.)
- end loop

Step 1: Select K points in the data space and mark them as **initial centroids.**

k=3, then initial points → (1.5, 1) ,(5.5,6.5),(4.3,4.8)

Measure the distance of each point in the cluster from the centroid

DATA POINTS	CENTRIOD -1 (1.5,1) D= $\sqrt{(X_2-X_1)^2+(Y_2-Y_1)^2}$	CENTRIOD-2 (5.5,6.5) D= $\sqrt{(X_2-X_1)^2+(Y_2-Y_1)^2}$	CENTRIOD-3 (4.3,4.8) D= $\sqrt{(X_2-X_1)^2+(Y_2-Y_1)^2}$	ASSIGN CLUSTER
(1.5,1)	C1= $\sqrt{(1.5-1.5)^2+(1-1)^2}$ C1=0	C2= $\sqrt{(5.5-1.5)^2+(6.5-1)^2}$ C2=6.800	C3= $\sqrt{(4.3-1.5)^2+(4.8-1)^2}$ C3=4.720	min(c1,c2, c3) **C1**
(5.5,6.5)	C1= $\sqrt{(5.5-1.5)^2+(6.5-1)^2}$ C1=6.800	C2= $\sqrt{(5.5-5.5)^2+(6.5-6.5)^2}$ C2=0	C3= $\sqrt{(5.5-4.3)^2+(6.5-4.8)^2}$ C3=2.080	C2
(4.3,4.8)	C1= $\sqrt{(1.5-4.3)^2+(1-4.8)^2}$ C1=4.720	C2= $\sqrt{(5.5-4.3)^2+(6.5-4.8)^2}$ C2=2.080	C3= $\sqrt{(4.3-4.3)^2+(4.8-4.8)^2}$ C3=0	C3
(1.5,1.9)	C1= $\sqrt{(1.5-1.5)^2+(1.9-1)^2}$ C1=0.9	C2= $\sqrt{(1.5-5.5)^2+(1.9-6.5)^2}$ C2=6.09	C3= $\sqrt{(1.5-4.3)^2+(1.9-4.8)^2}$ C3=4.0311	C1
(0.5,1.5)	C1= $\sqrt{(0.5-1.5)^2+(1.5-1)^2}$ C1=1.1180	C2= $\sqrt{(0.5-5.5)^2+(1.5-6.5)^2}$ C2=7.07	C3= $\sqrt{(0.5-4.3)^2+(1.5-4.8)^2}$ C3=5.032	C1

(2.5,3.8)	**C1= $\sqrt{(2.5-1.5)^2+(3.8-1)^2}$ C1=2.9732**	**C2= $\sqrt{(2.5-5.5)^2+(3.8-6.5)^2}$ C2=4.0360**	**C3= $\sqrt{(2.5-4.3)^2+(3.8-4.8)^2}$ C3=2.0591**	**C3**
(3.4,3.5)	C1= $\sqrt{(3.4-1.5)^2+(3.5-1)^2}$ C1=3.1400	C2= $\sqrt{(3.4-5.5)^2+(3.5-6.5)^2}$ C2=3.661	C3= $\sqrt{(3.4-4.3)^2+(3.5-4.8)^2}$ C3=1.5811	C3
(5,5.2)	C1= $\sqrt{(5.0-1.5)^2+(5.2-1)^2}$ C1=5.467	C2= $\sqrt{(5-5.5)^2+(5.2-6.5)^2}$ C2=1.3928	C3= $\sqrt{(5.0-4.3)^2+(5.2-4.8)^2}$ C3=0.8062	C3
(4.2,5.3)	C1= $\sqrt{(4.2-1.5)^2+(5.3-1)^2}$ C1=5.0774	C2= $\sqrt{(4.2-5.5)^2+(5.3-6.5)^2}$ C2=1.769	C3= $\sqrt{(4.2-4.3)^2+(5.3-4.8)^2}$ C3=0.5099	C3
(2,2)	C1= $\sqrt{(2-1.5)^2+(2-1)^2}$ C1=1.1180	C2= $\sqrt{(2-5.5)^2+(2-6.5)^2}$ C2=5.700	C3= $\sqrt{(2-4.3)^2+(2-4.8)^2}$ C3=3.6235	C1

Points:

C1→ (2,2) , (0.5,1.5) , (1.5,1.9) , (1.5,1)

C2→ (5.5,6.5)

C3→ (4.2,5.3) , (5,5.2) , (3.4,3.5) , (2.5,3.8) , (4.3,4.8)

$x_1 = \frac{x1+x2+x3+x4}{4}$ → $\frac{2+0.5+1.5+1.5}{4}$ → **1.375** $y_1 = \frac{y1+y2+y3+y4}{4}$ → $\frac{2+1.5+1.9+1}{4}$ → **1.6**

C1→(x1,y1) → (1.375,1.6)

C2 → (x2,y2) → (5.5,6.5)

$x_3 = \frac{4.2+5+3.4+2.5+4.3}{5} \rightarrow 3.88$ $\quad y_3 = \frac{5.3+5.2+3.5+3.8+4.8}{5} \rightarrow 4.52$

C3→(x3,y3) → (3.88,4.52)

The updated centroids are

C1→ (1.375,1.6)	**C2 → (5.5,6.5)**	**C3→ (3.88,4.52)**

There is no movement of centroids from iteration →1 to iteration→ 2 . So we can terminate the process.

Finally the values of updated centroids are

Centroid 1 = (1.375, 1.6)

Centroid 2 = (5.5, 6.5)

Centroid 3 = (3.88, 4.52)

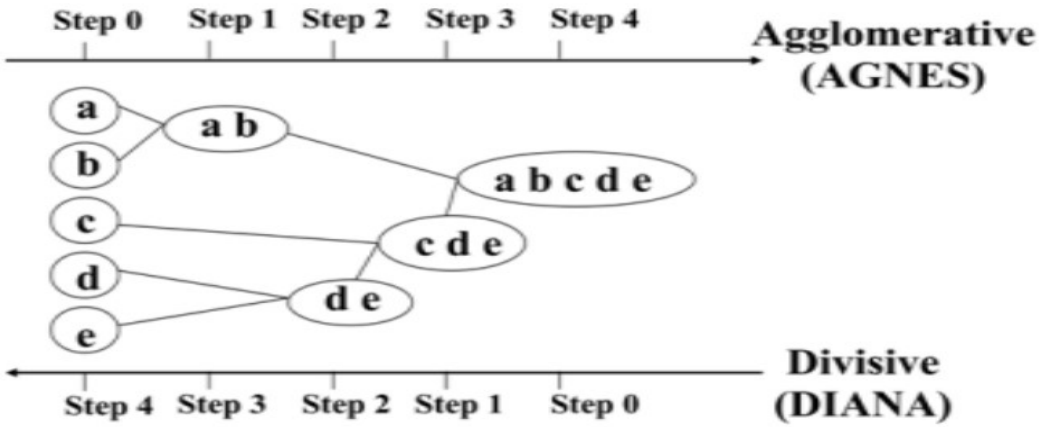
- **Repeat this process , When the updated centroids from one iteration are the same (or very close to) the centroids from the previous iteration, the algorithm will stop**

9b).How does hierarchical clustering work? Explain the top-down and bottom-up approaches of building a hierarchical cluster tree by iteratively merging or splitting clusters based on similarity measures?

Hierarchical Clustering :

- Hierarchical clustering is another unsupervised machine learning algorithm, which is used to group the unlabeled datasets into a cluster(group the data into hierarchy or tree-like structure)
- 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.
- **The hierarchical clustering technique has two approaches:**

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.	**Divisive:** Divisive algorithm is the reverse of the agglomerative algorithm as it is a **top-down approach.**



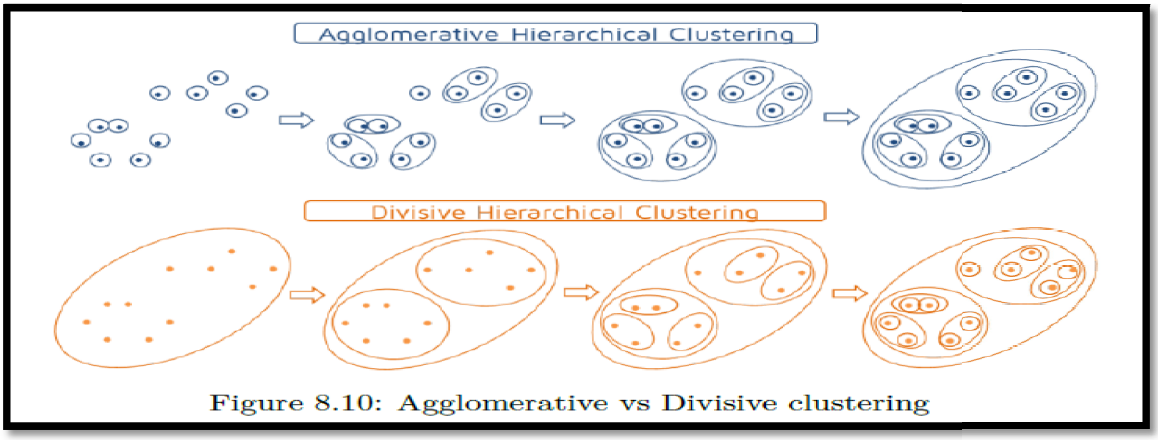


Figure 8.10: Agglomerative vs Divisive clustering

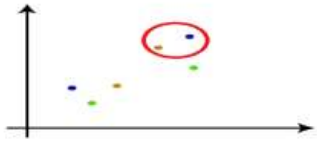
Agglomerative Hierarchical clustering Work:

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

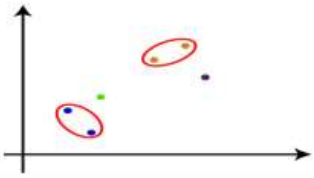
Step-1: Create each data point as a single cluster. Let's say there are N data points, so the number of clusters will also be N.



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: Once all the clusters are combined into one big cluster, develop the dendrogram to divide the clusters as per the problem.

