SRINIVAS UNIVERSITY
Question Bank - III SEMESTER MCA 2024

UNIT 1: INTRODUCTION TO MACHINE LEARNING

7 Mark Questions:

1. **Define Machine learning and explain the concept of machine learning with a neat diagram.**

Machine Learning (ML) is a **subfield** of artificial intelligence (**AI**) that focuses on the development of algorithms and models that enable computers to learn and make predictions or decisions without being explicitly programmed for a specific task.

The first step in machine learning activity starts with **data**. In case of supervised learning, it is the labelled training data set followed by **test data which is not labelled**. In case of unsupervised learning, there is no question of labelled data but the task is to find patterns in the input data.

A thorough review and exploration of the data is needed to understand the type of the data, the quality of the data and relationship between the different data elements. Based on that, multiple pre-processing activities may need to be done on the input data before we can go ahead with core machine learning activities. Following are the typical preparation activities done once the input data comes into the machine learning system:

- Understand the type of data in the given input data set (For example Numerical Data).
- Explore the data to understand the nature and quality.
- Explore the relationships amongst the data elements, e.g. inter-feature relationship.
- Find potential issues in data. (you might find missing values, outliers, duplicate entries, or data entry errors)
- Do the necessary remediation, e.g. impute missing data values, etc., if needed. Once issues are identified, you take steps to address them.
- Apply the following pre-processing steps, as necessary.

1. Dimensionality Reduction

2. Feature sub-set selection

Once the data is prepared for modelling, then the learning tasks start off.

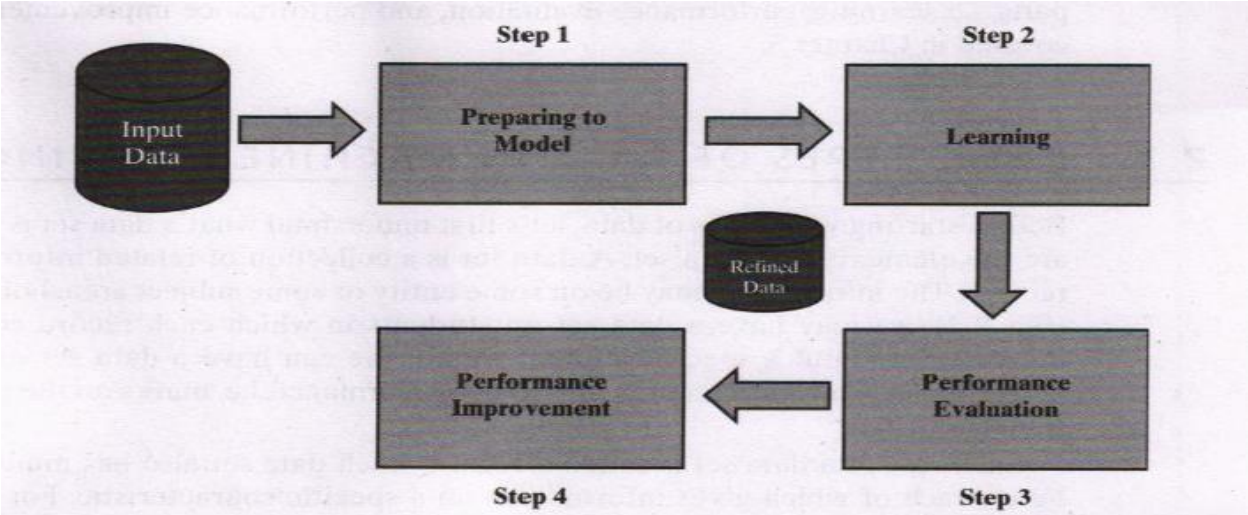


FIG. 2.1
Detailed process of machine learning

Once the data is prepared for modelling, then the learning tasks start off. As a part of it, do the following activities:

1. " The input data is **first divided into parts** — the **training data** and the **test data** (called holdout). This step is applicable for supervised learning only.
2. Consider different **models** or **learning algorithms** for selection. "Train the model based on the training data for supervised learning problem and apply to unknown data. Directly apply the chosen unsupervised model on the input data for unsupervised learning problem.
3. After the model is selected, trained (for supervised learning), and applied on input data, the performance of the model is evaluated. Based on options available, specific actions can be taken to improve the performance of the model, if possible.

2. Explain the various applications of machine learning

Machine learning, a subset of artificial intelligence, has revolutionized numerous industries by enabling computers to learn from data and make

predictions or decisions without explicit programming. Its applications span across diverse domains, including healthcare, finance, e-commerce, marketing, transportation, and more. Below are some key applications:

1. **Healthcare**: Machine learning is instrumental in medical diagnosis, analyzing medical images (like X-rays and MRI scans) to detect diseases early. Additionally, it powers personalized medicine by analyzing patient data to recommend tailored treatments.

2. **Finance**: Machine learning algorithms play a vital role in fraud detection by identifying unusual patterns in financial transactions. They are also used in algorithmic trading, where models analyze market data to make automated trading decisions.

3. E-**commerce**: Recommendation systems powered by machine learning algorithms suggest products to users based on their past behavior and preferences. Moreover, machine learning enables dynamic pricing, adjusting prices in real-time based on demand and other factors.

4. **Marketing**: Machine learning techniques are employed for customer segmentation and targeted advertising, enabling companies to personalize marketing campaigns. Sentiment analysis, which analyzes text data from social media to gauge public opinion about products or brands, is another important application.

5. **Transportation**: In the transportation sector, machine learning algorithms are utilized in autonomous vehicles for tasks such as object detection, path planning, and decision-making. Moreover, they optimize transportation systems by predicting traffic patterns and improving route efficiency.

6. **Image Recognition:**

   It is one of the most common machine learning applications. There are many situations where you can classify the object as a digital image. For

digital images, the measurements describe the outputs of each pixel in the image.

In the case of a black and white image, the intensity of each pixel serves as one measurement.

So if a black and white image has N*N pixels, the total number of pixels and hence measurement is N2. In the coloured image, each pixel considered as providing 3 measurements of the intensities

of 3 main colour components ie RGB. So N*N coloured image there are 3 N2 measurements.

- For face detection – The categories might be face versus no face present. There might be

a separate category for each person in a database of several individuals.

- For character recognition – We can segment a piece of writing into smaller images, each

containing a single character. The categories might consist of the 26 letters of the English alphabet, the 10 digits, and some special characters.

7. Speech Recognition

Speech recognition (SR) is the translation of spoken words into text. It is also known as “automatic speech recognition” (ASR), “computer speech recognition”, or “speech to text” (STT). In speech recognition, a software application recognizes spoken words. The measurements in

this Machine Learning application might be a set of numbers that represent the speech signal. We can segment the signal into portions that contain distinct words or phonemes. In each segment, we can represent the speech signal by the intensities or energy in different time frequency bands.

Although the details of signal representation are outside the scope of this program, we can represent the signal by a set of real values.

Speech recognition, Machine Learning applications include voice user interfaces. Voice user interfaces are such as voice dialing, call routing, domotic appliance control. It can also use as simple data entry, preparation of structured documents, speech-to-text processing, and plane.

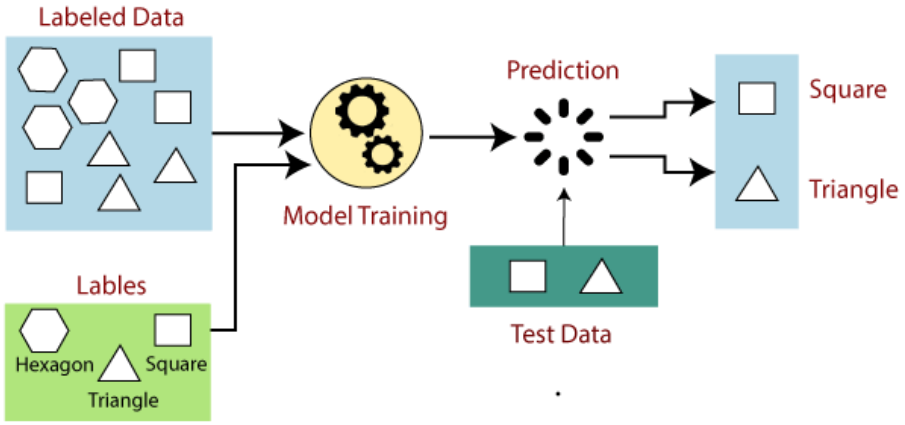
3. **Briefly Explain the types of Supervised and Unsupervised Machine Learning with appropriate examples.**

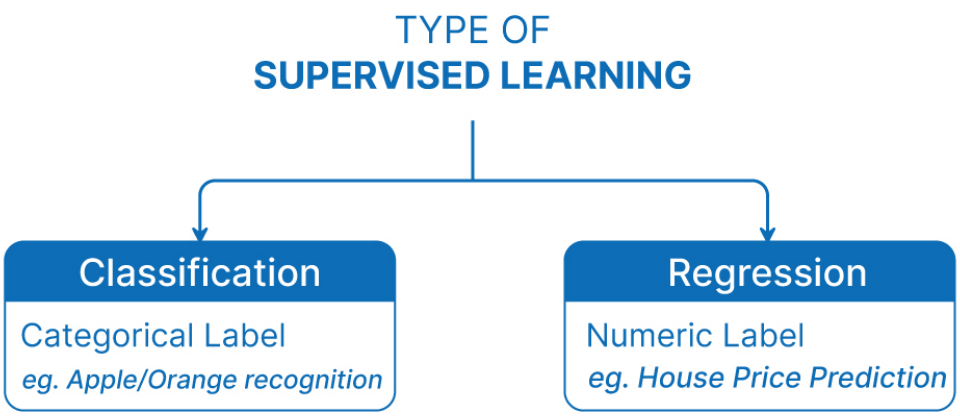
Supervised Machine Learning

Supervised learning is the types of machine learning in which machines are trained using well "**labelled**" **training data**, and on basis of that data, machines predict the output. The labelled data means some input data is already **tagged** or **labelled** with the **correct output.**

In supervised learning, **the training data provided to the machines work as the supervisor** that teaches the machines to predict the output correctly. It applies the same concept as a student learns in the **supervision of the teacher**. Supervised learning is a process of providing input data as well as correct output data to the machine learning model. The aim of a supervised learning algorithm is to find a mapping function to **map** the **input variable(x)** with the **output variable(y).**

Example: Optional





1. **Classification:**

Classification is a type of supervised machine learning task where the goal is to **predict the category or class** that a new instance or observation belongs to, **on the basis of training data**. The output variable in classification is **discrete** and represents different classes or labels.

In Classification, a program learns from the given dataset or observations and then classifies new observation into a number of classes or groups. Such as, Yes or No, 0 or 1, Spam or Not Spam, cat or dog, etc. Classes can be called as targets/labels or categories.

2. **Regression :**

Regression is a type of supervised machine learning task where the goal is to predict a **continuous numerical value** or outcome based on input features.

Regression is a type of supervised machine learning where algorithms **learn** from the data to predict continuous values such as sales, salary, weight, or temperature.

Note:

In the context of regression in machine learning, a continuous numerical value refers to an outcome or target variable that can take on an infinite number of values within a specific range. In case of predicting a person's age, age is considered a continuous numerical value because it can theoretically take on any value within a certain range (for example, from 0 to 100+ years). **There are**

no gaps or intervals between possible ages, and age can be expressed as a decimal or fraction if necessary (e.g., 25.5 years).

It is a variable that can have any real number value, and there are no distinct categories or classes. The term "continuous" implies that the variable can vary over a continuous range, and there are no gaps or interruptions in the possible values it can take. In contrast, in a classification task, the target variable would be a **discrete set of categories or classes.**

Example:

1. **Predicting age of a person:** Given certain features or attributes of a person, such as **height, weight, gender**, and other relevant factors, the task is to predict the person's age in years.
2. **Predicting the price of houses based on their features:**

   In real estate markets, house prices can vary continuously based on factors such as location, size, amenities, market conditions, and other features. The House prices can range from a few thousand dollars for smaller properties in certain areas to millions of dollars for luxury properties in prime locations.
3. Predicting the **salary** of an employee on the basis of the **year of experience**.

Unsupervised Learning:

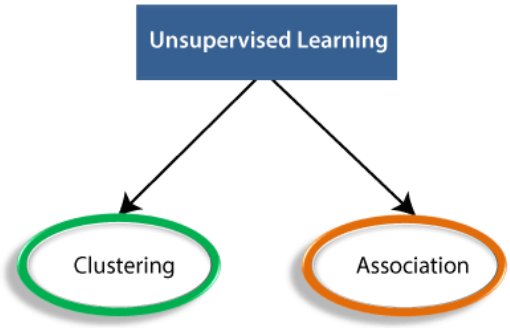
Unsupervised Machine learning is a type of machine learning where the algorithm is trained on **unlabeled data**, and the **objective is to find patterns, relationships, or structures within the data** without explicit guidance or labeled outcomes. i.e Unsupervised learning is a method we use to **group data** when no **labels are present**.

As the **name suggests**, unsupervised learning is a machine learning technique in which models are **not supervised using training dataset**. Instead, **models itself find** the hidden patterns and insights from the given data. It can be compared to **learning which takes place in the human brain while learning new things**.

It can be defined as: **"**Unsupervised learning is a type of machine learning in which models are trained using unlabeled dataset and are allowed to act on that data without any supervision.**"**

Example:

Suppose the unsupervised learning algorithm is given an input dataset containing **images of different types of cats and dogs**. The algorithm is never trained upon the given dataset, which means it does not have any idea about the features of the dataset. The task of the unsupervised learning algorithm is to identify the image features on their own. Unsupervised learning algorithm will perform this task **by clustering the image dataset into the groups according to similarities between images**.

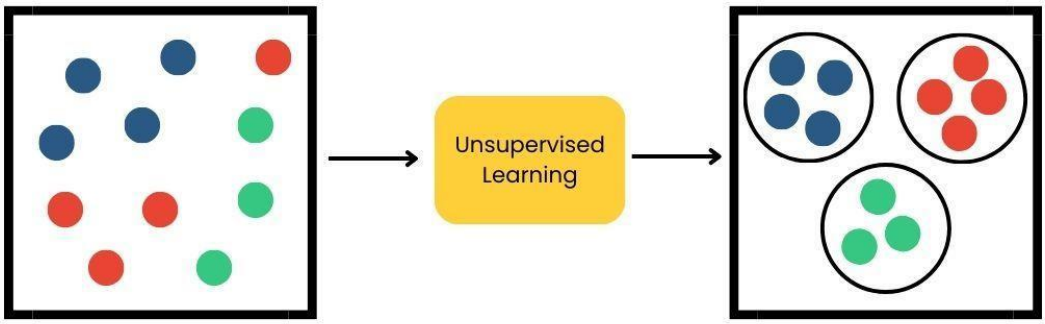


1. **Clustering:**

Clustering methods involve **grouping untagged** data based on their similarities and differences. When two instances appear in different groups, we can infer they have dissimilar properties.

Clustering is a type of unsupervised learning, meaning that we do not need labeled data for clustering algorithms; this is one of the biggest advantages of clustering over other supervised learning like Classification.

Clustering is the process of **arranging** a group of objects in such a manner that the objects in the same group (which is referred to as a cluster) are more **similar** to **each other** than to the objects in any **other group**.



2. **Association Rule:**

An association rule is an unsupervised learning method which is used **for finding the relationships between variables in the large database**. It determines the set of items that occurs together in the dataset. Association rule makes marketing strategy more effective. Such as people who buy X item (suppose a bread) are also tend to purchase Y (Butter/Jam) item. A typical example of Association rule is **Market Basket Analysis(MBA)**.

We typically see association rule mining used for market basket analysis: this is a data mining technique retailers use to gain a better understanding of customer purchasing patterns based on the relationships between various products.

So Association is the process of discovering interesting relationships, associations, or patterns within a dataset. This type of analysis is often applied to **transactional data**, where the goal is to identify associations between items or events that frequently co-occur. Association rules are used to express these relationships, and they help reveal hidden connections in the data.

4. **Explain the various activities of machine learning with a neat diagram**

Machine Learning (ML) is a **subfield** of artificial intelligence (**AI**) that focuses on the development of algorithms and models that enable computers to learn

and make predictions or decisions without being explicitly programmed for a specific task.

The first step in machine learning activity starts with **data**. In case of supervised learning, it is the labelled training data set followed by **test data which is not labelled**. In case of unsupervised learning, there is no question of labelled data but the task is to find patterns in the input data.

A thorough review and exploration of the data is needed to understand the type of the data, the quality of the data and relationship between the different data elements. Based on that, multiple pre-processing activities may need to be done on the input data before we can go ahead with core machine learning activities. Following are the typical preparation activities done once the input data comes into the machine learning system:

- Understand the type of data in the given input data set (For example Numerical Data).
- Explore the data to understand the nature and quality.
- Explore the relationships amongst the data elements, e.g. inter-feature relationship.
- Find potential issues in data. (you might find missing values, outliers, duplicate entries, or data entry errors)
- Do the necessary remediation, e.g. impute missing data values, etc., if needed. Once issues are identified, you take steps to address them.
- Apply the following pre-processing steps, as necessary.

1. Dimensionality Reduction
2. Feature sub-set selection

Once the data is prepared for modelling, then the learning tasks start off.

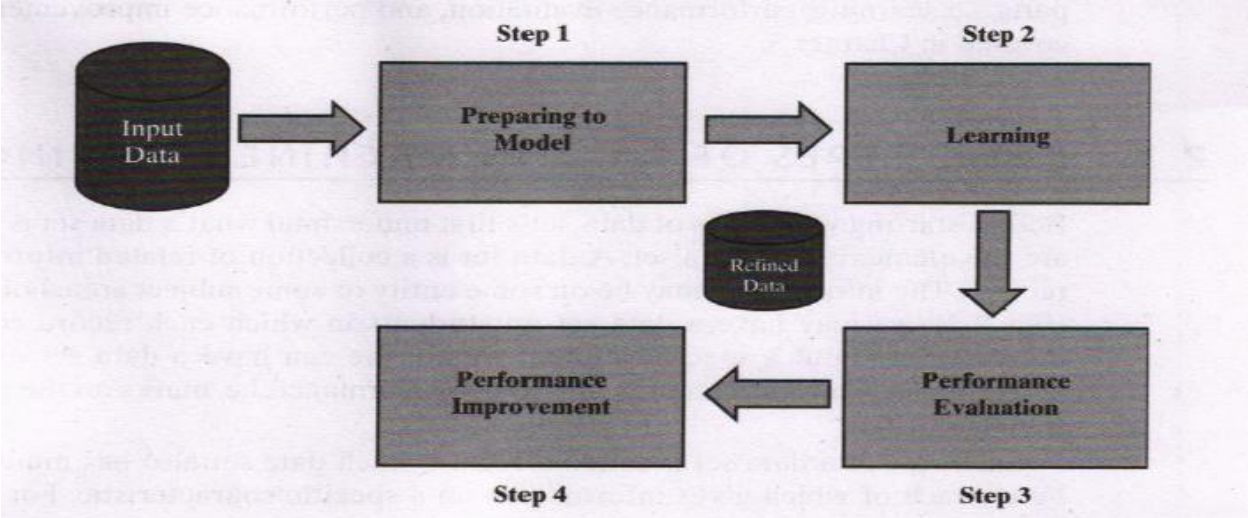


FIG. 2.1
Detailed process of machine learning

Once the data is prepared for modelling, then the learning tasks start off. As a part of it, do the following activities:

4. " The input data is **first divided into parts** — the **training data** and the **test data** (called holdout). This step is applicable for supervised learning only.
5. Consider different **models** or **learning algorithms** for selection. "Train the model based on the training data for supervised learning problem and apply to unknown data. Directly apply the chosen unsupervised model on the input data for unsupervised learning problem.
6. After the model is selected, trained (for supervised learning), and applied on input data, the performance of the model is evaluated. Based on options available, specific actions can be taken to improve the performance of the model, if possible.

5. Explain the concept of dimensionality reduction in preprocessing.

High-dimensional data sets need a high amount of computational space and time. At the same time, not all features are useful — they degrade the performance of machine learning algorithms. Most of the machine learning algorithms perform better if the dimensionality of data set, i.e. the number of features in the data set, is reduced. Dimensionality reduction helps

in reducing irrelevance and redundancy in features. Also, it is easier to understand a model if the number of features involved in the learning activity is less. Dimensionality reduction refers to the techniques of reducing the dimensionality of a data set by creating new attributes by combining the original attributes. The most common approach for dimensionality reduction is known as Principal Component Analysis (PCA).

PCA is a statistical technique to convert a set of correlated variables into a set of transformed, uncorrelated variables called principal components. The principal components area linear combination of the original variables. They are orthogonal to each other. Since principal components are uncorrelated, they capture the maximum amount of variability in the data. However, the only challenge is that the original attributes are lost due to the transformation. Another commonly used technique which is used for dimensionality reduction is Singular Value Decomposition (SVD).

Dimensionality reduction is a fundamental concept in preprocessing data for machine learning tasks. It involves reducing the number of input variables or features under consideration by selecting a subset of relevant features or transforming the data into a lower-dimensional space while preserving essential information. This process is essential for simplifying complex datasets, improving computational efficiency, and mitigating the curse of dimensionality.

There are two primary approaches to dimensionality reduction:

1. Feature Selection: Feature selection involves selecting a subset of the original features and discarding the irrelevant or redundant ones. It aims to retain the most informative features that contribute significantly to the predictive performance of the model. Feature selection techniques include filter methods, wrapper methods, and embedded methods. Filter methods evaluate the relevance of features based on statistical measures like correlation or mutual information. Wrapper methods employ a search strategy, such as forward selection or backward elimination, to find the optimal subset of features. Embedded methods integrate feature selection

into the model training process, allowing the model to select features based on their contribution to the objective function.

2. Feature Extraction: Feature extraction transforms the original features into a lower-dimensional space using linear or nonlinear transformations. It aims to capture the essential information present in the original features while reducing redundancy and noise. Principal Component Analysis (PCA) is a popular linear dimensionality reduction technique that projects the data onto a lower-dimensional subspace while maximizing the variance of the projected data. Other techniques, such as t-distributed Stochastic Neighbor Embedding (t-SNE) and autoencoders, perform nonlinear dimensionality reduction by preserving local or global structure in the data.

Dimensionality reduction offers several benefits in preprocessing data for machine learning tasks:

- Improved Model Performance: By reducing the number of features, dimensionality reduction can prevent overfitting and improve the generalization performance of machine learning models, especially in high-dimensional datasets.

- Enhanced Interpretability: Simplifying the dataset makes it easier to interpret and visualize the relationships between features and the target variable, leading to better insights and decision-making.

- Reduced Computational Complexity: Dimensionality reduction reduces the computational complexity of machine learning algorithms, making them more efficient and scalable, especially in real-time or resource-constrained environments.

6. What is Reinforcement Learning in Machine Learning? Explain with Example.

Reinforcement Learning:

Reinforcement Learning (RL) is a type of machine learning paradigm in which an **agent learns to make decisions by interacting with an environment**. The agent takes actions in the environment, and in **return, it receives feedback in the form of rewards or punishments**.

Reinforcement Learning is a **feedback-based Machine learning technique** in which an agent learns to behave in an environment by performing the actions and seeing the results of actions. For each good action, the agent gets **positive feedback**, and for each bad action, the agent gets **negative feedback or penalty**. In Reinforcement Learning, the agent learns automatically using feedbacks without any labeled data, unlike supervised learning Since there is no labeled data, so the agent is bound to learn by its experience only. RL solves a specific type of problem where decision making is sequential, and the goal is long-term, such as game-playing, robotics, etc. The agent interacts with the environment and explores it by itself. The primary goal of an agent in reinforcement learning is to **improve the performance by getting the maximum positive rewards.**

The agent learns with the process of **hit and trial**, and based on the experience, it learns to perform the task in a better way. Hence, we can say that "Reinforcement learning is a type of machine learning method where an intelligent agent (computer program) interacts with the environment and learns to act within that." How a **Robotic dog learns** the movement of his arms is an example of Reinforcement learning.

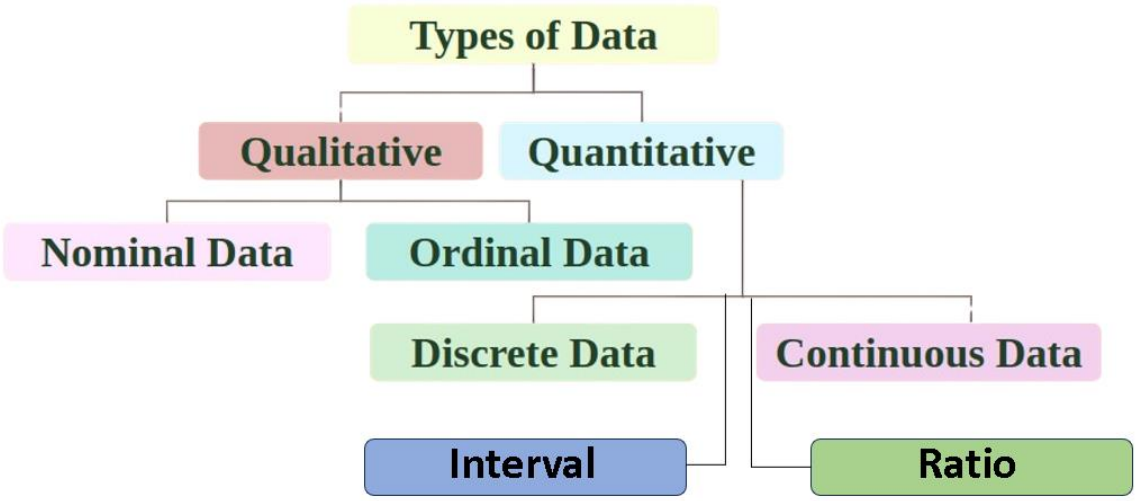
It is a **core part of Artificial Intelligence**, and all AI agent works on the concept of reinforcement learning. Here we do not need to pre-program the agent, as it learns from its own experience without any human intervention

Example:

Consider training an autonomous vehicle to navigate a maze. The vehicle (agent) interacts with the maze environment, receiving positive rewards for moving closer to the maze's exit and negative rewards for hitting walls or going further from the exit. Through trial and error, the vehicle learns a policy

(sequence of actions) to efficiently navigate the maze and reach the exit, optimizing its path to maximize cumulative rewards.

7. **Discuss the broad classification of data used in machine learning along with appropriate examples**



1. **Qualitative Data: (Non- Measurable One)**

Definition: Qualitative data, also known as **categorical data**, represents **characteristics** or **attributes** that are not measured on a numerical scale. Instead, they are categorical in nature and represent different categories or groups.

For example, if we consider the quality of performance of students in terms of 'Good' 'Average', and 'Poor' it falls under the category of qualitative data.

Examples:

- Gender (Male, Female)
- Color (Red, Blue, Green)
- Marital Status (Married, Single, Divorced)
- Type of Vehicle (Car, Truck, Motorcycle)

There are 2 Types:

1. **Nominal data (Unordered data):**
   Nominal data represents **categories or groups with no inherent order or ranking**. Nominal data does not follow any hierarchy. The categories are **distinct** and, **mutually exclusive (separate**) but there is no meaningful numerical value associated with them.
   Examples of nominal data are :
   - Blood group: A, B, O, AB, etc.
   - Nationality: Indian, American, British, etc.
   - Gender: Male, Female, Other.

2. **Ordinal data (Ordered):**
   Ordinal data represents categories or groups with a **specific order or ranking**. While **the categories themselves are distinct** and **mutually exclusive**, they also have a meaningful sequence or hierarchy.
   In addition to possessing the properties of nominal data, can also be **naturally ordered**. This means ordinal data also assigns named values to attributes but unlike nominal data, they can be **arranged in a sequence of increasing or decreasing value** so that we can say whether a value is better than or greater than another value.
   Examples of ordinal data are:
   - Customer satisfaction: 'Very Happy' 'Happy' 'Unhappy; etc.
   - Grades: A, B, C, etc.
   - Hardness of Metal: 'Very Hard', 'Hard', 'Soft', etc.
   - Educational levels :High School, Bachelor's, Master's, Ph.D.
   - Performance ratings :Poor, Fair, Good, Excellent

2. **Quantitative Data: (Measurable One):**
   Definition: Quantitative data, also known as numerical data, consists of numerical measurements or quantities that can be expressed as numbers and subjected to mathematical operations. It involves values that can be expressed as numbers and subjected to mathematical operations.

For example, if we consider the attribute 'marks' it can be measured using a scale of measurement. Quantitative data is also termed as numeric data.

Examples:

- Height (in centimeters or inches)
- Age (in years)
- Income (in dollars)
- Temperature (in Celsius or Fahrenheit)
- Number of products sold.
- Test scores.

There are Four types of quantitative data:

1. **Discrete Data: (Whole Number or a Number without Fractional Part):** Data that can only take certain values is called discrete data or discrete values. This is data that can be counted and has a limited number of values. It usually comes in the form of **whole numbers or integers.**

   Examples:

   - Number of siblings.
   - Number of goals scored in a soccer match.
   - Number of defects in a manufacturing process.
   - Number of customers in a store at a given time.
   - Age of a Person
   - Number of cars in a parking lot

2. **Continues Data:**

   Continuous data is data **that can take any value.** Height, weight, temperature and length are all examples of continuous data. Some continuous data will change over time; the weight of a baby in its first year or the temperature in a room throughout the day.

   Continuous data represents measurements that can take on any value within a certain range. These values are not restricted to whole numbers and can include decimals or fractions.

   Example:

- Height of individuals.
- Weight of objects.
- Temperature readings.
- Time taken to complete a task.
- Distance traveled by a vehicle.

8. How do you handle missing values in data preprocessing? Explain

Missing values refer to the absence of data for one or more variables in a dataset. These missing values can occur for various reasons, such as data entry errors, equipment malfunction, non-response in surveys, or intentional omission. Missing values can adversely affect data analysis and modeling if not handled properly, as they can bias statistical estimates, reduce the effectiveness of machine learning models, or lead to incorrect conclusions.

Handling missing values is a critical step in data preprocessing to ensure the quality and integrity of the dataset. There are several techniques available to address missing values effectively.

Diff Approaches to handle missing values:

1. **Removing Missing Values:**
   One approach is to remove observations or features with missing values entirely from the dataset.
   This method is suitable when the missing values are few and randomly distributed across the dataset.
   For example, if a small percentage of rows contain missing values, those rows can be removed without significantly affecting the overall dataset.

2. **Imputation:**
   Imputation involves replacing missing values with estimated or calculated values based on the available data.
   Common imputation techniques include mean, median, mode imputation, or using predictive models to estimate missing values.

For numerical features, replacing missing values with the mean or median of the respective feature is a straightforward approach.
For categorical features, replacing missing values with the mode (most frequent value) is often used.
Example: In a dataset containing age values with missing entries, missing values can be replaced with the mean age of the non-missing entries.

3. **Advanced Imputation Techniques:**
   More advanced imputation techniques include K-nearest neighbors (KNN) imputation or regression imputation.
   KNN imputation involves finding the K-nearest neighbors of a data point with missing values and using their values to impute the missing values.
   Regression imputation fits a regression model to predict missing values based on other features in the dataset.
   Example: In a dataset with missing values in a particular feature, KNN imputation identifies the nearest neighbors with similar characteristics and imputes missing values based on their values.

4. **Using Indicator Variables:**
   Another approach is to create indicator variables (also known as dummy variables) to denote the presence of missing values.
   This approach preserves information about the missingness of values and can be used as a feature in predictive modeling.
   Example: In a dataset where missing values are imputed with zeros or some other placeholder value, an additional binary indicator variable can be created to indicate whether the original value was missing.

5. **Domain-specific Methods:**
   In some cases, domain-specific knowledge may guide the handling of missing values.
   For example, in time-series data, missing values may be filled with the most recent available value or interpolated based on trends in the data.

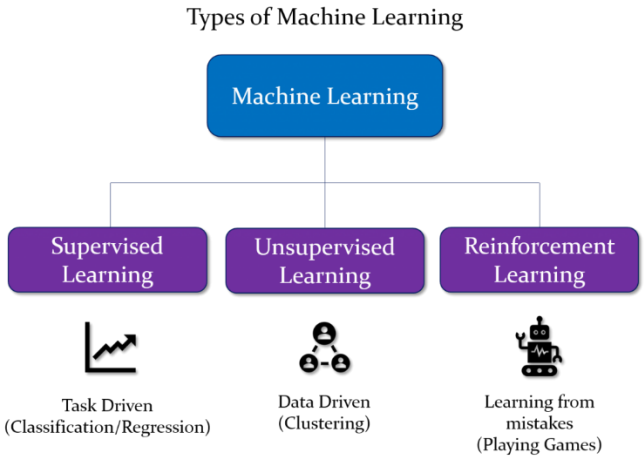
By employing appropriate techniques to handle missing values, data preprocessing ensures that machine learning models can effectively learn from the available data, leading to more accurate and reliable predictions or insights. Each method has its advantages and limitations, and the choice of technique depends on factors such as the nature of the data, the extent of missingness, and the requirements of the analysis or modeling task.

10 Mark Questions:

1. What is Machine Learning? Explain the broad classification of machine learning techniques with appropriate examples?

   Machine Learning (ML) is a **subfield** of artificial intelligence (**AI**) that focuses on the development of algorithms and models that enable computers to learn and make predictions or decisions without being explicitly programmed for a specific task.

   According to **Tom Mitchell** Machine Learning **is the study of algorithms that improve** their **performance** P at some task T with experience E. A well-defined learning task for a system is given by <P, T, E>.



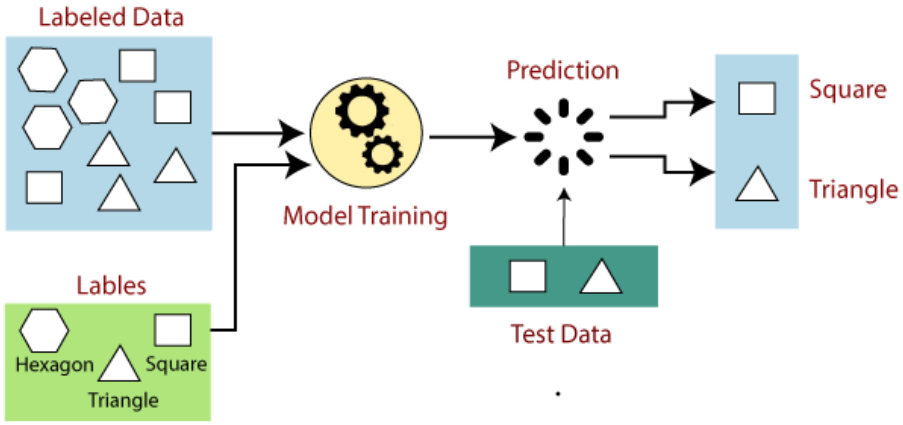
Supervised Machine Learning

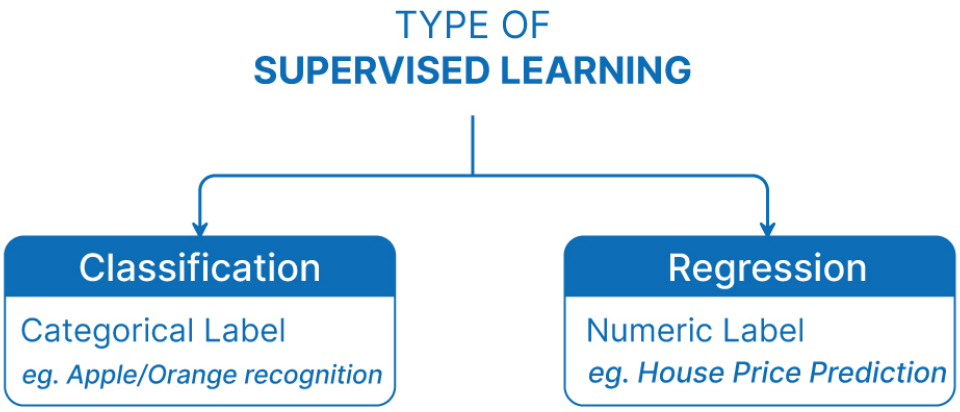
Supervised learning is the types of machine learning in which machines are trained using well "**labelled**" **training data**, and on basis of that data, machines

predict the output. The labelled data means some input data is already **tagged** or **labelled** with the **correct output.**

In supervised learning, **the training data provided to the machines work as the supervisor** that teaches the machines to predict the output correctly. It applies the same concept as a student learns in the **supervision of the teacher**. Supervised learning is a process of providing input data as well as correct output data to the machine learning model. The aim of a supervised learning algorithm is to find a mapping function to **map** the **input variable(x)** with the **output variable(y).**

Example: Optional





3. **Classification:**

Classification is a type of supervised machine learning task where the goal is to **predict the category or class** that a new instance or observation belongs

to, **on the basis of training data**. The output variable in classification is **discrete** and represents different classes or labels.

In Classification, a program learns from the given dataset or observations and then classifies new observation into a number of classes or groups. Such as, Yes or No, 0 or 1, Spam or Not Spam, cat or dog, etc. Classes can be called as targets/labels or categories.

4. **Regression :**

Regression is a type of supervised machine learning task where the goal is to predict a **continuous numerical value** or outcome based on input features.

Regression is a type of supervised machine learning where algorithms **learn** from the data to predict continuous values such as sales, salary, weight, or temperature.

Note:

In the context of regression in machine learning, a continuous numerical value refers to an outcome or target variable that can take on an infinite number of values within a specific range. In case of predicting a person's age, age is considered a continuous numerical value because it can theoretically take on any value within a certain range (for example, from 0 to 100+ years). **There are no gaps or intervals between possible ages**, and age can be expressed as a decimal or fraction if necessary (e.g., 25.5 years).

It is a variable that can have any real number value, and there are no distinct categories or classes. The term "continuous" implies that the variable can vary over a continuous range, and there are no gaps or interruptions in the possible values it can take. In contrast, in a classification task, the target variable would be a **discrete set of categories or classes.**

Example:

4. **Predicting age of a person:** Given certain features or attributes of a person, such as **height, weight, gender**, and other relevant factors, the task is to predict the person's age in years.
5. **Predicting the price of houses based on their features:**

In real estate markets, house prices can vary continuously based on factors such as location, size, amenities, market conditions, and other features. The

House prices can range from a few thousand dollars for smaller properties in certain areas to millions of dollars for luxury properties in prime locations.

6. Predicting the **salary** of an employee on the basis of the **year of experience**.

Unsupervised Learning:

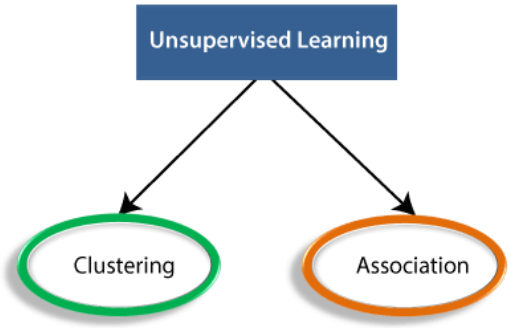
Unsupervised Machine learning is a type of machine learning where the algorithm is trained on **unlabeled data**, and the **objective is to find patterns, relationships, or structures within the data** without explicit guidance or labeled outcomes. i.e Unsupervised learning is a method we use to **group data** when no **labels are present**.

As the **name suggests**, unsupervised learning is a machine learning technique in which models are **not supervised using training dataset**. Instead, **models itself find** the hidden patterns and insights from the given data. It can be compared to **learning which takes place in the human brain while learning new things**.

It can be defined as: **"**Unsupervised learning is a type of machine learning in which models are trained using unlabeled dataset and are allowed to act on that data without any supervision.**"**

Example:

Suppose the unsupervised learning algorithm is given an input dataset containing **images of different types of cats and dogs**. The algorithm is never trained upon the given dataset, which means it does not have any idea about the features of the dataset. The task of the unsupervised learning algorithm is to identify the image features on their own. Unsupervised learning algorithm will perform this task **by clustering the image dataset into the groups according to similarities between images**.

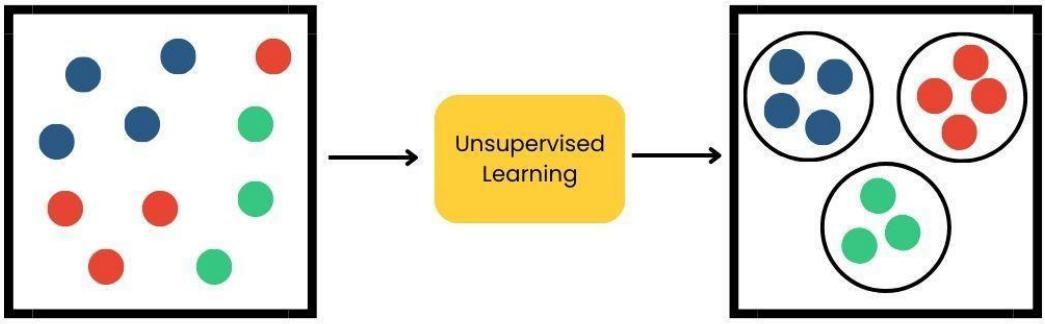


3. **Clustering:**

Clustering methods involve **grouping untagged** data based on their similarities and differences. When two instances appear in different groups, we can infer they have dissimilar properties.

Clustering is a type of unsupervised learning, meaning that we do not need labeled data for clustering algorithms; this is one of the biggest advantages of clustering over other supervised learning like Classification.

Clustering is the process of **arranging** a group of objects in such a manner that the objects in the same group (which is referred to as a cluster) are more **similar** to **each other** than to the objects in any **other group**.



4. **Association Rule:**

An association rule is an unsupervised learning method which is used **for finding the relationships between variables in the large database**. It determines the set of items that occurs together in the dataset. Association rule makes

marketing strategy more effective. Such as people who buy X item (suppose a bread) are also tend to purchase Y (Butter/Jam) item. A typical example of Association rule is **Market Basket Analysis(MBA)**.

We typically see association rule mining used for market basket analysis: this is a data mining technique retailers use to gain a better understanding of customer purchasing patterns based on the relationships between various products. So Association is the process of discovering interesting relationships, associations, or patterns within a dataset. This type of analysis is often applied to **transactional data**, where the goal is to identify associations between items or events that frequently co-occur. Association rules are used to express these relationships, and they help reveal hidden connections in the data.

Reinforcement Learning:

Reinforcement Learning (RL) is a type of machine learning paradigm in which an **agent learns to make decisions by interacting with an environment**. The agent takes actions in the environment, and in **return, it receives feedback in the form of rewards or punishments**.

Reinforcement Learning is a **feedback-based Machine learning technique** in which an agent learns to behave in an environment by performing the actions and seeing the results of actions. For each good action, the agent gets **positive feedback**, and for each bad action, the agent gets **negative feedback or penalty**. In Reinforcement Learning, the agent learns automatically using feedbacks without any labeled data, unlike supervised learning Since there is no labeled data, so the agent is bound to learn by its experience only. RL solves a specific type of problem where decision making is sequential, and the goal is long-term, such as game-playing, robotics, etc. The agent interacts with the environment and explores it by itself. The primary goal of an agent in reinforcement learning is to **improve the performance by getting the maximum positive rewards.**

2. **Explain the broad classification of data used in machine learning, including structured, unstructured, and semi-structured data.**

1. **Qualitative Data: (Non- Measurable One)**

Definition: Qualitative data, also known as **categorical data**, represents **characteristics** or **attributes** that are not measured on a numerical scale. Instead, they are categorical in nature and represent different categories or groups.

For example, if we consider the quality of performance of students in terms of 'Good' 'Average', and 'Poor' it falls under the category of qualitative data.

Examples:

- Gender (Male, Female)
- Color (Red, Blue, Green)
- Marital Status (Married, Single, Divorced)
- Type of Vehicle (Car, Truck, Motorcycle)

There are 2 Types:

1. **Nominal data (Unordered data):**
   Nominal data represents **categories or groups with no inherent order or ranking**. Nominal data does not follow any hierarchy. The categories are **distinct** and, **mutually exclusive (separate**) but there is no meaningful numerical value associated with them.
   Examples of nominal data are :
   - Blood group: A, B, O, AB, etc.
   - Nationality: Indian, American, British, etc.
   - Gender: Male, Female, Other.

2. **Ordinal data (Ordered):**
   Ordinal data represents categories or groups with a **specific order or ranking**. While **the categories themselves are distinct** and **mutually exclusive**, they also have a meaningful sequence or hierarchy.
   In addition to possessing the properties of nominal data, can also be **naturally ordered**. This means ordinal data also assigns named values to attributes but unlike nominal data, they can be

arranged in a sequence of increasing or decreasing value so that we can say whether a value is better than or greater than another value.

Examples of ordinal data are:

- Customer satisfaction: 'Very Happy' 'Happy' 'Unhappy; etc.
- Grades: A, B, C, etc.
- Hardness of Metal: 'Very Hard', 'Hard', 'Soft', etc.
- Educational levels :High School, Bachelor's, Master's, Ph.D.
- Performance ratings :Poor, Fair, Good, Excellent

2. **Quantitative Data: (Measurable One):**

Definition: Quantitative data, also known as numerical data, consists of numerical measurements or quantities that can be expressed as numbers and subjected to mathematical operations. It involves values that can be expressed as numbers and subjected to mathematical operations.

For example, if we consider the attribute 'marks' it can be measured using a scale of measurement. Quantitative data is also termed as numeric data.

Examples:

- Height (in centimeters or inches)
- Age (in years)
- Income (in dollars)
- Temperature (in Celsius or Fahrenheit)
- Number of products sold.
- Test scores.

There are Four types of quantitative data:

3. **Discrete Data: (Whole Number or a Number without Fractional Part):** Data that can only take certain values is called discrete data or discrete values. This is data that can be counted and has a limited number of values. It usually comes in the form of **whole numbers or integers.**

   Examples:

   - Number of siblings.
   - Number of goals scored in a soccer match.

- Number of defects in a manufacturing process.
- Number of customers in a store at a given time.
- Age of a Person
- Number of cars in a parking lot

4. **Continues Data:**

   Continuous data is data **that can take any value.** Height, weight, temperature and length are all examples of continuous data. Some continuous data will change over time; the weight of a baby in its first year or the temperature in a room throughout the day.

   Continuous data represents measurements that can take on any value within a certain range. These values are not restricted to whole numbers and can include decimals or fractions.

   Example:

   - Height of individuals.
   - Weight of objects.
   - Temperature readings.
   - Time taken to complete a task.
   - Distance traveled by a vehicle.

Structured, unstructured, and semi-structured data are classifications based on the organization and format of data. Here's an explanation of each:

1. **Structured Data:**

   Structured data refers to data that has a well-defined and organized structure, typically stored in databases or tabular formats.

   It is characterized by a fixed schema, where each data element is organized into rows and columns.

Examples of structured data include relational databases, spreadsheets, CSV files, and tables in SQL databases.

Structured data is easily queryable, making it suitable for analysis using traditional database management systems (DBMS) and SQL queries.

2. **Unstructured Data:**
   Unstructured data refers to data that does not have a predefined data model or organization, making it more challenging to analyze using traditional methods.

   It lacks a formal structure and can include text documents, images, audio files, videos, social media posts, emails, and web pages.

   Unstructured data is typically stored in formats that do not adhere to a specific schema, making it difficult to process and analyze using traditional database systems.

   Analyzing unstructured data often requires advanced techniques such as natural language processing (NLP), image recognition, and machine learning algorithms to extract insights and patterns.

3. **Semi-Structured Data:**

   Semi-structured data is a hybrid form of data that falls between structured and unstructured data.

   It has some organizational properties similar to structured data, such as tags, labels, or keys, but does not conform to a rigid schema.

   Examples of semi-structured data include XML (eXtensible Markup Language), JSON (JavaScript Object Notation), log files, and NoSQL databases like MongoDB.

While semi-structured data does not have a fixed schema, it often retains some level of hierarchy or organization, making it more flexible than structured data but less chaotic than unstructured data.

Analyzing semi-structured data may require specialized tools or techniques that can handle its flexible nature, such as XML parsers or JSON processing libraries.

UNIT 2: FEATURE ENGINEERING& BAYESIAN CONCEPT LEARNING

7 Mark Questions:

1. **Explain the concept of Feature Engineering**

   FEATURE ENGINEERING:

   Feature engineering refers to the process of **translating** a **data set into features** such that these features are able to **represent the data set more effectively** and result in a **better learning performance**. Feature engineering is an important pre-processing step for machine learning. It has two major elements:

   1. **Feature transformation**
   2. **Feature subset selection**

   Feature engineering is the process of **creating new features** or **modifying existing ones** from the raw data to improve the performance of machine learning models. It involves **selecting**, **transforming**, and **creating features** that are relevant and informative for the task at hand. Effective feature engineering can lead to better model performance and **generalization**.

   1. **Feature transformation:**

      Feature transformation **transforms the data** — structured or unstructured, into a **new set of features** which can represent the underlying problem which machine learning is trying to solve.

      Feature transformation involves **changing the representation** of the features in the dataset to make them more suitable for the machine learning algorithm.

      Engineering a good feature space is a crucial prerequisite for the **success** of any machine learning model. However, **often it is not clear which feature is more important**. For that reason, all available attributes of the data set are used as features and the problem of identifying the important features is left to the learning model. This is definitely not a feasible approach, particularly for certain domains e.g. medical image classification, text categorization, etc. In case a model has to be trained to classify a document as spam or non-spam, we can represent a document as

a bag of words. Then the feature space will contain all unique words occurring across all documents. This will easily be a feature space of a few hundred thousand features. If we start including bigrams or trigrams along with words, the count of features will run in millions. To deal with this problem, feature transformation comes into play. **Feature transformation is used as an effective tool for dimensionality reduction** and hence **for boosting** learning model performance. **Broadly, there are two distinct goals of feature transformation:**

1. Achieving best reconstruction of the original features in the data set
2. Achieving highest efficiency in the learning task

There are two variants of feature transformation:

1. **Feature construction (or Generation):**
2. **Feature extraction:**

1. **Feature construction (or Generation):**

   Feature construction involves **creating new features by combining or transforming the existing features** in the dataset. This process discovers missing information about the relationships between features and increases the feature space by creating additional features. Hence, if there are 'n' features or dimensions in a data set, after feature construction 'm' more features or dimensions may get added. So at the end, the data set will become '**n + m**' dimensional.

   Feature construction involves transforming a given set of input features to generate a new set of **more powerful features**. To understand more clearly, let's take the example of a **real estate data set having details of all apartments sold in a specific region**. The data set has three features — **apartment length, apartment breadth, and price of the apartment**. If it is used as an input to a regression problem, such data can be training data for the regression model. So given the training data, the model should be able to predict the price of an apartment whose price is not known or which has just come up for sale. However, instead of using length and breadth of the apartment as a predictor, it is

much convenient and makes more sense to use the **area of the apartment**, which is not an existing feature of the data set. Hence, such a feature, namely apartment area, can be added to the data set. In other words, we transform **the three-dimensional data set** to a **four-dimensional data set**, with the newly 'discovered' feature apartment area being added to the original data set.

2. Feature extraction:

Feature extraction involves **reducing the dimensionality** of the data **by selecting or extracting a subset of relevant features** from the original feature set. This process aims to retain as much relevant information as possible **while discarding redundant or irrelevant features.**

Unlike feature construction, which creates entirely new features, feature extraction aims to capture the essence of the original features in a more compact or meaningful representation.

2. **Feature subset selection: (or simply feature selection)**

In Feature subset selection **no new feature is generated**. Feature subset selection is a technique in feature engineering that involves choosing a subset of most relevant features from the original set of features in a dataset. The goal is to improve the **model's performance by reducing the dimensionality** of the data and eliminating irrelevant or redundant features. This process can lead to more efficient and interpretable models, as well as potentially faster training times.

Certainly! Feature selection is a **critical process** in machine learning aimed at identifying the most relevant subset of features from the original set of features in a dataset. Initially, we have a set of features $F=\{F1,F2,...,Fn\}$, representing various attributes of the data. The goal is to derive a subset $F'=\{Fj,Fo,...,Fm\}$ from F, where $m<n$ and $Fj,Fo,...,Fm$ are the selected features. Among these selected features, Fy represents the subset deemed most meaningful and relevant for the machine learning task at hand.

2. **Briefly Explain the Bayes Theorem in Machine Learning?**

Bayes Theorem: (Conditional Probability)

Bayes theorem is one of the most popular machine learning concepts that **helps to calculate the probability of occurring one event with uncertain knowledge while other one has already occurred.**

In ML, Bayes Theorem is used to calculate the probability of a **Hypothesis (theory)** or an **event** based on prior knowledge or evidence.

Bayes' Theorem is a fundamental concept in probability theory and statistics, and it plays a crucial role in various machine learning algorithms, particularly those based on Bayesian inference. In machine learning, Bayes' Theorem is **used to calculate the posterior probability of a hypothesis given observed data.**

Here "observed data" refers to the dataset or information that is available for analysis. It consists of the input features and corresponding outcomes or labels that are used to make inferences or predictions.

Definition:

Bayes theorem One of the most well-known theories in machine learning, the Bayes theorem helps determine the **likelihood that one event will occur with unclear information while another has already happened.**

Conditional Probability: (Pre requisite for Bayes Theorem)

Conditional probability is a measure of the probability of **an event occurring** given that **another event has already occurred**. It is denoted as $P(A|B)$ and is read as "**the probability of event A given event B**."

Formula for Conditional Probability for Event A given that (|) event B already occurred:

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

Probability of A given B

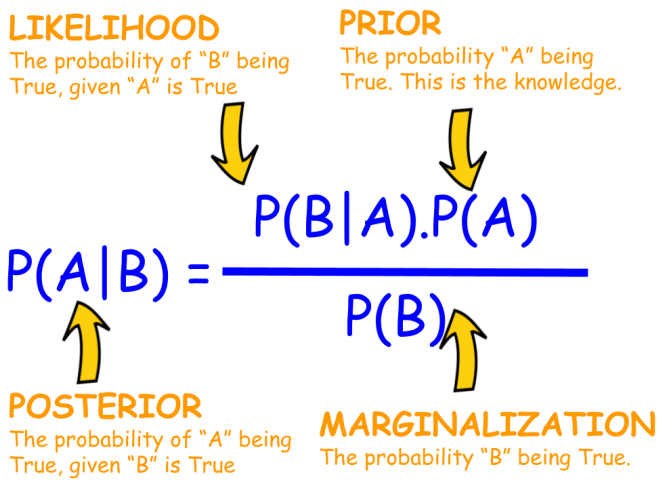
Probability of A and B

Probability of B

Formula for Conditional Probability for Event B given that (|) event A already occurred:

$$P(B|A) = \frac{P(B \cap A)}{P(A)}$$

Derivation of **Bayes Theorem:**



Example of Playing Card = 52. Now The card that I chose is a Face Card and the Problem is to find whether the face card is King or Not

Total Face Card: **12**

Now we have to find the Following:

P(King | Face) = P(Face | King) P(King) / P(Face)

Here,

1. P(Face | King) : Means the card is King but the Probablity fo Face we have to find. Which is nothing but 1 as if we choose any king card it will be a face card.
2. P(King): Totally we have 4 King cards out of 52 so the probability of King is 4/52.
3. P(Face): Totally we have 12 Face cards out of 52 so the probability of Face is 12/52.

Finally,

P(King | Face) = (1 * 4/52) / (12/52)

= 1/3 **1/3 of 12 Face is = 4**

3. **Explain the various terms associated with Bayes Theorem**

Bayes Theorem: (Conditional Probability)

Bayes theorem is one of the most popular machine learning concepts that **helps to calculate the probability of occurring one event with uncertain knowledge while other one has already occurred.**

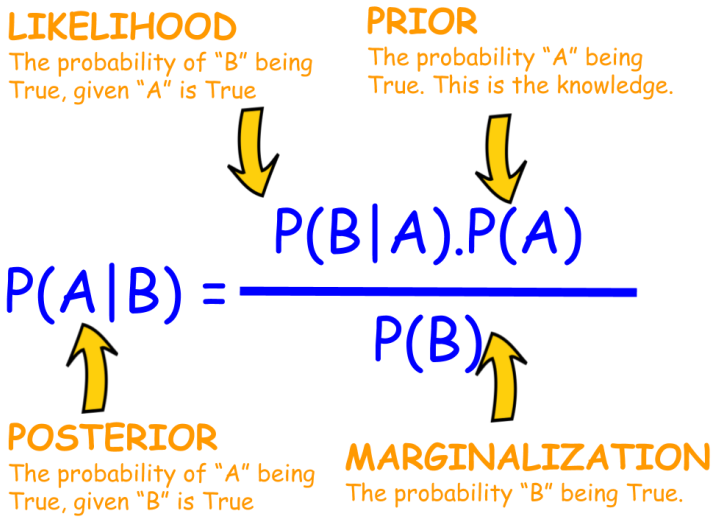
In ML, Bayes Theorem is used to calculate the probability of a **Hypothesis (theory)** or an **event** based on prior knowledge or evidence.

Bayes' Theorem is a fundamental concept in probability theory and statistics, and it plays a crucial role in various machine learning algorithms, particularly those based on Bayesian inference. In machine learning, Bayes' Theorem is **used to calculate the posterior probability of a hypothesis given observed data.**

Here "observed data" refers to the dataset or information that is available for analysis. It consists of the input features and corresponding outcomes or labels that are used to make inferences or predictions.

Definition:

Bayes theorem One of the most well-known theories in machine learning, the Bayes theorem helps determine the **likelihood that one event will occur with unclear information while another has already happened.**



1. **Prior Probability (Prior)**:

The prior probability represents **our initial belief or knowledge** about the likelihood of an event or hypothesis before we have observed any new evidence or data.

- The prior probability serves as the **starting point** for Bayesian inference and **influences the posterior probability.**
- The prior probability can be based on previous experience, domain knowledge, or assumptions.

Also called as Prior **Probability** is the prior knowledge or belief about the probabilities of various Hypothesis H is called Prior.

It is based on our prior knowledge, assumptions, or belief about the probability of the event.

For Example:

1. Suppose we want to determine the probability of a **patient having a particular disease** let's call it Disease X) before any diagnostic tests are performed. Our prior belief about the prevalence or spread of Disease X in the population is 0.05, meaning that 5% of the population is estimated to

have Disease X based on historical data. This prior probability is denoted as P(Disease X)=0.05.

2. Let's say we want to predict whether a **customer will buy a product or not** based on their age, gender and income, before we predict we should have some prior knowledge.
3. if we have to determine whether a particular type of tumour is malignant for a patient, the prior knowledge of such tumours becoming malignant can be used to validate our current hypothesis and is a prior probability or simply called Prior.

2. Posterior:

The probability that a **particular hypothesis holds for a data set based on the Prior** is called the posterior probability or simply Posterior.

In the above example, the probability of the hypothesis that the patient has a malignant tumour considering the Prior of correctness of the malignancy testis a posterior probability.

The posterior probability represents our **updated belief or probability of an event or hypothesis being true** after observing new evidence or data.

The posterior probability is calculated using Bayes' theorem, which **combines the prior probability, the likelihood, and the evidence or data.**

The posterior probability reflects our updated understanding of the event or hypothesis based on the observed evidence.

Example:

After the patient undergoes a diagnostic test for Disease X and the test results come back **positive**, we want to update our belief about the probability of the patient having the disease. Using Bayes' theorem, we calculate the posterior probability of the patient having Disease X given the positive test result. Let's say the likelihood of a positive test result given that the patient has Disease X is 0.95, and the likelihood of a positive test result given that the patient does not have Disease X (false positive rate) is 0.10. Using Bayes' theorem, we update our prior probability to calculate the posterior probability:

P(Disease X | Positive Test) = P(Positive Test | Disease X)×P(Disease X) / P(Positive Test)

3. **Likelihood**:

The likelihood represents the probability of **observing the evidence or data given** that a particular hypothesis or event is **true**. It measures **how well the Hypothesis explains the observed data**. i.e, The likelihood quantifies how well the observed data supports the hypothesis or event A.

The likelihood plays a crucial role in Bayesian inference as it helps update the prior probability to the posterior probability.

For Example:

The likelihood represents the probability of observing the evidence (test results) given the hypothesis (presence or absence of Disease X). In our example, the likelihood of a positive test result given that the patient has Disease X is 0.95, and the likelihood of a positive test result given that the patient does not have Disease X (false positive rate) is 0.10.

In summary, the prior probability represents **our initial belief** about the likelihood of an event (presence of Disease X), the posterior probability represents **our updated belief after observing new evidence** (positive test result), and the likelihood represents the probability of observing the evidence given the hypothesis. These concepts are fundamental to Bayesian inference and help us make informed decisions in uncertain situations, such as medical diagnosis.

4. **Explain the concept of Naive Bayes Classifier**

Naive Bayes classifier:

The Naive Bayes classifier is a simple yet powerful probabilistic classifier based on **Bayes' theorem**. So the base of Naive Bayes classifier is Bayes' theorem.

Consider the Following Example:

Let's say we have a Data Set with Features {x1, x2, x3, …… , xn} and the output {y}. Now the Task is to Classify the given data which is **y** by making use of the

Features. For Navie Bayes Classifier we have to follow the Bayes Theorem, So according to the Bayes Theorem Formula we have to simplify our Problem.

$$P(y|x_1, x_2, x_3, \ldots, x_n) = \frac{P(x_1,x_2,x_3,\ldots,x_n|y)\times P(y)}{P(x_1,x_2,x_3,\ldots,x_n)}$$

$$P(y|x_1, x_2, x_3, \ldots, x_n) = \frac{P(y)\times\prod_{i=1}^{n} P(x_i|y)}{P(x_1,x_2,x_3,\ldots,x_n)}$$

Here, P(x1, x2, x3, ……, xn) is common for all the classes or records in the data set. So we can ignore it

$$P(y|x_1, x_2, x_3, \ldots, x_n) \propto P(y) \times \prod_{i=1}^{n} P(x_i|y)$$

So, we're essentially looking for the class y that maximizes $P(y) \times \prod_{i=1}^{n} P(x_i|y)$.

Mathematically, we can express this as:

$$y = \arg\max_{y} \left(P(y) \times \prod_{i=1}^{n} P(x_i|y)\right)$$

This equation implies that it is used to find the value of *y* that **maximizes the expression In RHS**. In other words, it returns the class label *y* that has the **highest probability** given the input features *x*1,*x*2,*x*3,…,*xn*.
For Example: Let say we have Binary Classifiier for Spam or Not Spam if we are getting the Probability = 0.7 for the Mail is Spam and 0.3 that the mail is Not Spam then we have to consider the maximum among these that is 0.7 which means the mail is Spam.

Steps in Naives Bayes Classifier:

The term "**naive**" in Naive Bayes classifier refers to the **simplifying assumption made by the model** regarding the independence of features.

Specifically, it assumes that all features are conditionally independent given the class label. So **all features of the dataset are equally important and independent,** this is called Naive Bayes classifier

Naive Bayes classifier calculates the probability of an event in the following steps:

Step 1: Calculate the prior probability for given class labels

Step 2: Find Likelihood probability with each **attribute** for each class

Step 3: Put these value in Bayes Formula and calculate posterior probability.

Step 4: See which class has a higher probability, given the input belongs to the higher probability class.

Example:

Suppose we have a dataset with **two** classes, "**spam**" (denoted as y = spam) and "not spam" (denoted as *y*=not spam), and two features, *x*1 and *x*2.

We want to classify a new email with the following **features**:

***x*1=buy**

***x*2=discount**

Let's assume we have already calculated the following probabilities from our training dataset:

1. Prior probabilities:

- *P*(spam)=0.4 // Consider it as 40% out of 100%
- *P*(not spam)=0.6 // Consider it as 60%

2. Likelihoods:

- *P*(buy|spam)=0.8
- *P*(discount|spam)=0.6
- *P*(buy|not spam)=0.3
- *P*(discount|not spam)=0.5

Now, let's plug these values into the Naive Bayes classifier equation:

$$y = \arg\max_y \left(P(y) \times \prod_{i=1}^{2} P(x_i|y) \right)$$

For $y = \text{spam}$:
$P(\text{spam}) \times P(\text{buy}|\text{spam}) \times P(\text{discount}|\text{spam}) = 0.4 \times 0.8 \times 0.6 = 0.192$

For $y = \text{not spam}$:
$P(\text{not spam}) \times P(\text{buy}|\text{not spam}) \times P(\text{discount}|\text{not spam}) = 0.6 \times 0.3 \times 0.5 = 0.09$

Comparing the two values, we see that *y*=spam gives the higher result. Therefore, according to the **Naive Bayes classifier**, the predicted class for the given features "buy" and "discount" is "**spam**".

5. Discuss the applications of Naive Bayes Classifier

Naive Bayes classifier:

The Naive Bayes classifier is a simple yet powerful probabilistic classifier based on **Bayes' theorem**. So the base of Naive Bayes classifier is Bayes' theorem.

The Naive Bayes classifier is a versatile algorithm with various applications in machine learning. Here are some applications along with relevant excerpts from the provided text:

1. **Text Classification:**
   In text classification, the Naive Bayes classifier is used to categorize text documents into predefined classes or categories based on the words or features present in the text.
   The classifier calculates the probability of a document belonging to each class using Bayes' theorem and assigns the document to the class with the highest probability.

Example: Classifying news articles into categories such as sports, politics, or entertainment based on their content.

2. **Spam Filtering:**
   Spam filtering aims to automatically identify and filter out unwanted or unsolicited emails (spam) from legitimate emails (ham).
   The Naive Bayes classifier analyzes the content and features of emails, such as words, sender information, and email headers, to determine the probability of an email being spam or ham.
   Example: Gmail's spam filter uses a Naive Bayes classifier to classify incoming emails as spam or not spam based on various criteria.

3. **Hybrid Recommender System:**
   Recommender systems aim to provide personalized recommendations to users by predicting their preferences or interests.
   A hybrid recommender system combines multiple recommendation techniques, such as content-based filtering, collaborative filtering, and demographic-based filtering, to improve recommendation accuracy and coverage.
   The Naive Bayes classifier can be used in conjunction with collaborative filtering to predict whether a user would like a given resource based on their past behavior and preferences.
   Example: Amazon uses a hybrid recommender system that combines user browsing history, purchase behavior, and product attributes to recommend products to customers.

4. **Online Sentiment Analysis:**
   Sentiment analysis, also known as opinion mining, involves analyzing text data to determine the sentiment or opinion expressed by users.
   The Naive Bayes classifier can classify text data into positive, negative, or neutral sentiments based on the presence of sentiment-related words or features.

Example: Analyzing customer reviews or social media comments to determine the sentiment towards a product, service, or event.

6. What is Conditional Probability? Explain how it is related to Bayes theorem.

Conditional probability is a measure of the probability of an event occurring given that another event has already occurred. It is denoted as P(A|B) and is read as "the probability of event A given event B."

Formula for Condi**tional Probability for Event A given that (|) event B already occurred:**

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

Probability of A given B — Probability of A and B — Probability of B

Here:

- $P(A|B)$ is the conditional probability of event A given event B. So A and B are 2 events.
- $P(A \cap B)$ is the joint probability of events A and B occurring together.
- $P(B)$ is the probability of event B.

Formula for Conditional Probability for Event B given that (|) event A already occurred:

$$P(B|A) = \frac{P(B \cap A)}{P(A)}$$

Relationship between Conditional Probability and Bayes theorem:

Bayes theorem is completely depends on the Conditional Probability. Conditional probability and Bayes' theorem are closely related concepts in probability theory. Conditional probability deals with the probability of an event occurring given that

another event has already occurred. Bayes' theorem provides a mathematical formula to calculate conditional probability in certain situations.

Write the Proper Derivation Step by Step:

Derivation:

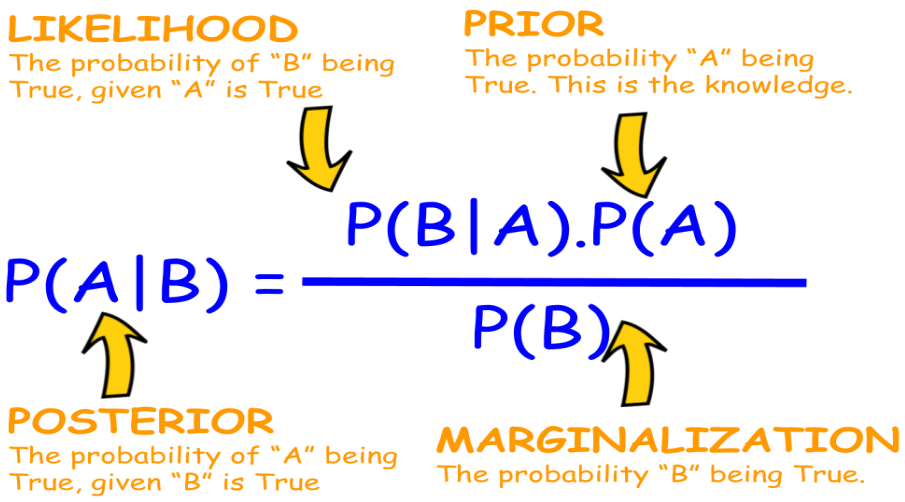
According to the definition of conditional probability,

$$P(A|B) = \frac{P(A \cap B)}{P(B)}, P(B) \neq 0 \text{ and we know that}$$

$$P(A \cap B) = P(B \cap A) = P(B|A)P(A), \text{ which implies,}$$

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Bayes theorem is one of the most popular machine learning concepts that **helps to calculate the probability of occurring one event with uncertain knowledge while other one has already occurred.**



Bayes theorem One of the most well-known theories in machine learning, the Bayes theorem helps determine the **likelihood that one event will occur with unclear information while another has already happened.**

7. Explain the concept of Feature transformation

Dd

Feature transformation:

Feature transformation **transforms the data** — structured or unstructured, into a **new set of features** which can represent the underlying problem which machine learning is trying to solve.

Feature transformation involves **changing the representation** of the features in the dataset to make them more suitable for the machine learning algorithm.

Engineering a good feature space is a crucial prerequisite for the **success** of any machine learning model. However, **often it is not clear which feature is more important**. For that reason, all available attributes of the data set are used as features and the problem of identifying the important features is left to the learning model. This is definitely not a feasible approach, particularly for certain domains e.g. medical image classification, text categorization, etc. In case a model has to be trained to classify a document as spam or non-spam, we can represent a document as a bag of words. Then the feature space will contain all unique words occurring across all documents. This will easily be a feature space of a few hundred thousand features. If we start including bigrams or trigrams along with words, the count of features will run in millions. To deal with this problem, feature transformation comes into play. **Feature transformation is used as an effective tool for dimensionality reduction** and hence **for boosting** learning model performance. **Broadly, there are two distinct goals of feature transformation:**

3. Achieving best reconstruction of the original features in the data set
4. Achieving highest efficiency in the learning task

There are two variants of feature transformation:

1. **Feature construction (or Generation):**

2. **Feature extraction:**

1. **Feature construction (or Generation):**

Feature construction involves **creating new features by combining or transforming the existing features** in the dataset. This process discovers missing information about the relationships between features and increases the feature space by creating additional features. Hence, if there are 'n' features or dimensions in a data set, after feature construction 'm' more features or dimensions may get added. So at the end, the data set will become '**n + m**' dimensional.

Feature construction involves transforming a given set of input features to generate a new set of **more powerful features**. To understand more clearly, let's take the example of a **real estate data set having details of all apartments sold in a specific region**. The data set has three features — **apartment length, apartment breadth, and price of the apartment**. If it is used as an input to a regression problem, such data can be training data for the regression model. So given the training data, the model should be able to predict the price of an apartment whose price is not known or which has just come up for sale. However, instead of using length and breadth of the apartment as a predictor, it is much convenient and makes more sense to use the **area of the apartment**, which is not an existing feature of the data set. Hence, such a feature, namely apartment area, can be added to the data set. In other words, we transform **the three-dimensional data set** to a **four-dimensional data set**, with the newly 'discovered' feature apartment area being added to the original data set.

2. Feature extraction:

Feature extraction involves **reducing the dimensionality** of the data **by selecting or extracting a subset of relevant features** from the original feature set. This process aims to retain as much relevant information as possible **while discarding redundant or irrelevant features.**

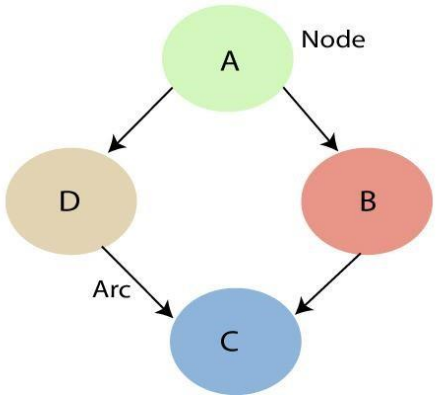
Unlike feature construction, which creates entirely new features, feature extraction aims to capture the essence of the original features in a more compact or meaningful representation.

8. **Explain Bayesian Belief Network with appropriate example?**

Bayesian Belief Network: (BBN)

Bayesian Belief Network or **Bayesian Network** or **Belief Network** is a Probabilistic Graphical Model (PGM) that represents a **set of random variables** and **their conditional dependencies** through a Directed Acyclic Graph (DAG).

It is a graphical representation of different probabilistic relationships among random variables in a particular set. Due to its feature of **joint probability,** the probability in Bayesian Belief Network is derived, based on a condition — P(attribute/parent) i.e probability of an attribute, true over parent attribute.



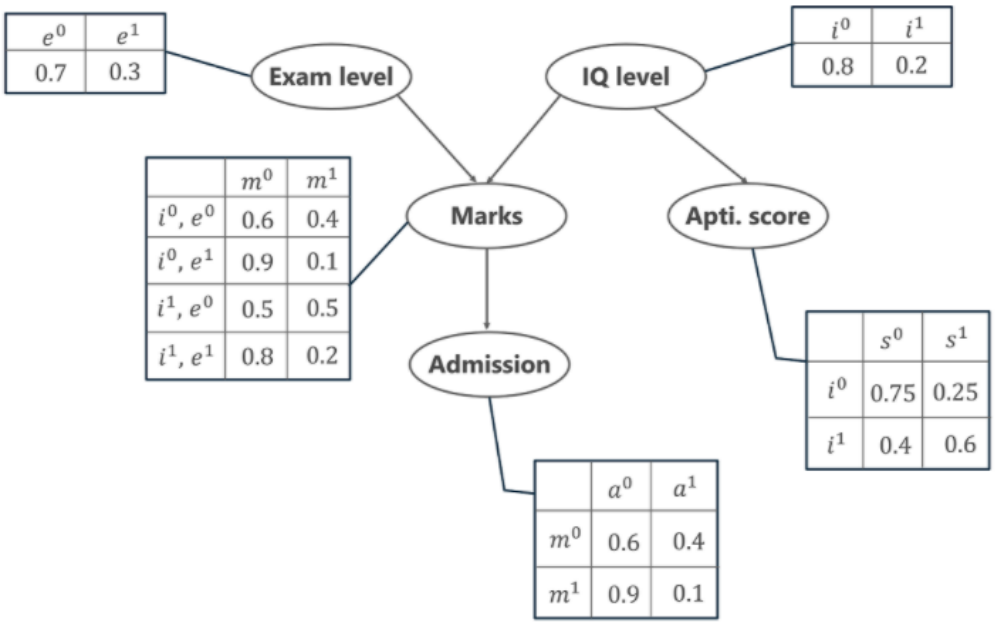
As mentioned above, by making use of the relationships which are specified by the Bayesian Network, we can obtain the **Joint Probability Distribution** (JPF) with the **conditional probabilities.** Each node in the graph represents **a random variable** and the arc (or directed arrow) represents the **relationship between the nodes**. They can be either continuous or discrete in nature.

Example of Bayesian Networks:

Let us now understand the mechanism of Bayesian Networks and their advantages with the help of a simple example. In this example, let us imagine that we are given the task of **Modeling a student's marks (*m*)** for an exam he has just given. From the given Bayesian Network Graph below, we see that the marks depend upon two other variables. They are,

1. **Exam Level (*e*)–** This **discrete variable** denotes the difficulty of the exam and has two values (**0 for easy and 1 for difficult**)
2. **IQ Level (*i*)** – This represents the Intelligence Quotient level of the student and is also discrete in nature having two values (**0 for low and 1 for high)**

Additionally, the **IQ level of the student also leads us to another variable**, which is the **Aptitude Score** of the student (***s***). Now, with marks the student has scored, he can secure **admission** to a particular university. The probability distribution for getting admitted (***a***) to a university is also given below.



e^0	e^1
0.7	0.3

i^0	i^1
0.8	0.2

	m^0	m^1
i^0, e^0	0.6	0.4
i^0, e^1	0.9	0.1
i^1, e^0	0.5	0.5
i^1, e^1	0.8	0.2

	s^0	s^1
i^0	0.75	0.25
i^1	0.4	0.6

	a^0	a^1
m^0	0.6	0.4
m^1	0.9	0.1

10 Mark Questions:

1. **What is Bayes' theorem? Briefly explain Bayes' theorem and the various terms associated with Bayesian theory, along with its derivation.**

Bayes Theorem: (Conditional Probability)

Bayes theorem is one of the most popular machine learning concepts that **helps to calculate the probability of occurring one event with uncertain knowledge while other one has already occurred.**

In ML, Bayes Theorem is used to calculate the probability of a **Hypothesis (theory)** or an **event** based on prior knowledge or evidence.

Bayes' Theorem is a fundamental concept in probability theory and statistics, and it plays a crucial role in various machine learning algorithms, particularly those based on Bayesian inference. In machine learning, Bayes' Theorem is **used to calculate the posterior probability of a hypothesis given observed data.**

Here "observed data" refers to the dataset or information that is available for analysis. It consists of the input features and corresponding outcomes or labels that are used to make inferences or predictions.

Definition:

Bayes theorem One of the most well-known theories in machine learning, the Bayes theorem helps determine the **likelihood that one event will occur with unclear information while another has already happened.**

Conditional probability is a measure of the probability of an event occurring given that another event has already occurred. It is denoted as P(A|B) and is read as "the probability of event A given event B."

Formula for Condi**tional Probability for Event A given that (|) event B already occurred:**

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

Probability of A given B — Probability of A and B — Probability of B

Here:

- $P(A|B)$ is the conditional probability of event A given event B. So A and B are 2 events.
- $P(A \cap B)$ is the joint probability of events A and B occurring together.
- $P(B)$ is the probability of event B.

Formula for Conditional Probability for Event B given that (|) event A already occurred:

$$P(B|A) = \frac{P(B \cap A)}{P(A)}$$

Relationship between Conditional Probability and Bayes theorem:

Bayes theorem is completely depends on the Conditional Probability. Conditional probability and Bayes' theorem are closely related concepts in probability theory. Conditional probability deals with the probability of an event occurring given that another event has already occurred. Bayes' theorem provides a mathematical formula to calculate conditional probability in certain situations.

Write the Proper Derivation Step by Step:

Derivation:

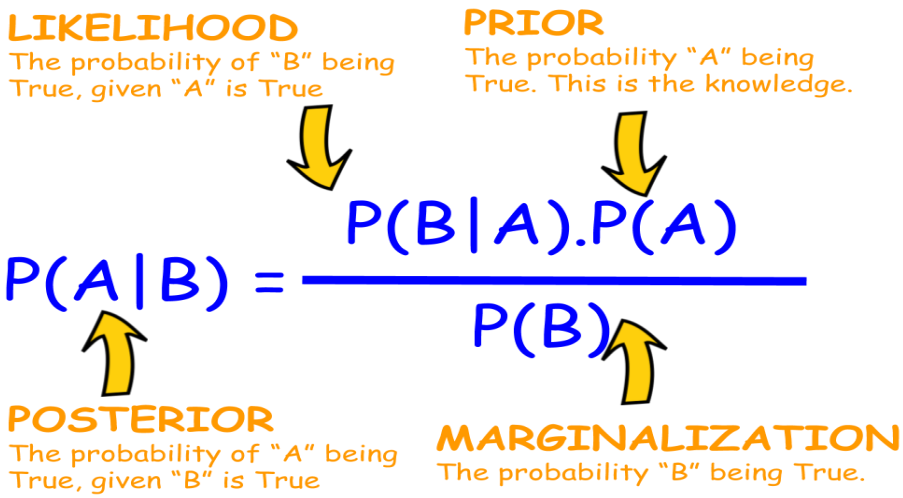
According to the definition of conditional probability,

$$P(A|B) = \frac{P(A \cap B)}{P(B)}, P(B) \neq 0 \text{ and we know that}$$

$$P(A \cap B) = P(B \cap A) = P(B|A)P(A), \text{ which implies,}$$

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Bayes theorem is one of the most popular machine learning concepts that **helps to calculate the probability of occurring one event with uncertain knowledge while other one has already occurred.**



Bayes theorem One of the most well-known theories in machine learning, the Bayes theorem helps determine the **likelihood that one event will occur with unclear information while another has already happened.**

1. **Prior Probability (Prior)**:

The prior probability represents **our initial belief or knowledge** about the likelihood of an event or hypothesis before we have observed any new evidence or data.

- The prior probability serves as the **starting point** for Bayesian inference and **influences the posterior probability.**
- The prior probability can be based on previous experience, domain knowledge, or assumptions.

Also called as Prior **Probability** is the prior knowledge or belief about the probabilities of various Hypothesis H is called Prior.

It is based on our prior knowledge, assumptions, or belief about the probability of the event.

For Example:

Suppose we want to determine the probability of a **patient having a particular disease** let's call it Disease X) before any diagnostic tests are performed. Our prior belief about the prevalence or spread of Disease X in the population is 0.05, meaning that 5% of the population is estimated to

have Disease X based on historical data. This prior probability is denoted as P(Disease X)=0.05.

Let's say we want to predict whether a **customer will buy a product or not** based on their age, gender and income, before we predict we should have some prior knowledge.

if we have to determine whether a particular type of tumour is malignant for a patient, the prior knowledge of such tumours becoming malignant can be used to validate our current hypothesis and is a prior probability or simply called Prior.

2. Posterior:

The probability that a **particular hypothesis holds for a data set based on the Prior** is called the posterior probability or simply Posterior.

In the above example, the probability of the hypothesis that the patient has a malignant tumour considering the Prior of correctness of the malignancy testis a posterior probability.

The posterior probability represents our **updated belief or probability of an event or hypothesis being true** after observing new evidence or data.

The posterior probability is calculated using Bayes' theorem, which **combines the prior probability, the likelihood, and the evidence or data.**

The posterior probability reflects our updated understanding of the event or hypothesis based on the observed evidence.

Example:

After the patient undergoes a diagnostic test for Disease X and the test results come back **positive**, we want to update our belief about the probability of the patient having the disease. Using Bayes' theorem, we calculate the posterior probability of the patient having Disease X given the positive test result. Let's say the likelihood of a positive test result given that the patient has Disease X is 0.95, and the likelihood of a positive test result given that the patient does not have Disease X (false positive rate) is 0.10. Using Bayes' theorem, we update our prior probability to calculate the posterior probability:

$P(\text{Disease X} \mid \text{Positive Test}) = P(\text{Positive Test} \mid \text{Disease X}) \times P(\text{Disease X}) / P(\text{Positive Test})$

3. **Likelihood**:

The likelihood represents the probability of **observing the evidence or data given** that a particular hypothesis or event is **true**. It measures **how well the Hypothesis explains the observed data**. i.e, The likelihood quantifies how well the observed data supports the hypothesis or event A.

The likelihood plays a crucial role in Bayesian inference as it helps update the prior probability to the posterior probability.

For Example:

The likelihood represents the probability of observing the evidence (test results) given the hypothesis (presence or absence of Disease X). In our example, the likelihood of a positive test result given that the patient has Disease X is 0.95, and the likelihood of a positive test result given that the patient does not have Disease X (false positive rate) is 0.10.

In summary, the prior probability represents **our initial belief** about the likelihood of an event (presence of Disease X), the posterior probability represents **our updated belief after observing new evidence** (positive test result), and the likelihood represents the probability of observing the evidence given the hypothesis. These concepts are fundamental to Bayesian inference and help us make informed decisions in uncertain situations, such as medical diagnosis.

2. **What is Naïve Bayes Classifier? Write the algorithm of Navie Bayes Classifier with appropriate example.**

Naive Bayes classifier:

The Naive Bayes classifier is a simple yet powerful probabilistic classifier based on **Bayes' theorem**. So the base of Naive Bayes classifier is Bayes' theorem.

Consider the Following Example:

Let's say we have a Data Set with Features {x1, x2, x3, …… , xn} and the output {y}. Now the Task is to Classify the given data which is **y** by making use of the Features. For Navie Bayes Classifier we have to follow the Bayes Theorem, So according to the Bayes Theorem Formula we have to simplify our Problem.

$$P(y|x_1, x_2, x_3, \ldots, x_n) = \frac{P(x_1,x_2,x_3,\ldots,x_n|y)\times P(y)}{P(x_1,x_2,x_3,\ldots,x_n)}$$

$$P(y|x_1, x_2, x_3, \ldots, x_n) = \frac{P(y)\times\prod_{i=1}^{n} P(x_i|y)}{P(x_1,x_2,x_3,\ldots,x_n)}$$

Here, P(x1, x2, x3, ……, xn) is common for all the classes or records in the data set. So we can ignore it

$$P(y|x_1, x_2, x_3, \ldots, x_n) \propto P(y) \times \prod_{i=1}^{n} P(x_i|y)$$

So, we're essentially looking for the class y that maximizes $P(y) \times \prod_{i=1}^{n} P(x_i|y)$.

Mathematically, we can express this as:

$$y = \arg\max_{y} \left(P(y) \times \prod_{i=1}^{n} P(x_i|y)\right)$$

This equation implies that it is used to find the value of *y* that **maximizes the expression In RHS**. In other words, it returns the class label *y* that has the **highest probability** given the input features *x*1,*x*2,*x*3,…,*xn*.
For Example: Let say we have Binary Classifiier for Spam or Not Spam if we are getting the Probability = 0.7 for the Mail is Spam and 0.3 that the mail is Not Spam then we have to consider the maximum among these that is 0.7 which means the mail is Spam.

Steps in Naives Bayes Classifier:

The term "**naive**" in Naive Bayes classifier refers to the **simplifying assumption made by the model** regarding the independence of features. Specifically, it assumes that all features are conditionally independent given the class label. So **all features of the dataset are equally important and independent,** this is called Naive Bayes classifier

Naive Bayes classifier calculates the probability of an event in the following steps:

Step 1: Calculate the prior probability for given class labels

Step 2: Find Likelihood probability with each **attribute** for each class

Step 3: Put these value in Bayes Formula and calculate posterior probability.

Step 4: See which class has a higher probability, given the input belongs to the higher probability class.

Example:

Suppose we have a dataset with **two** classes, "**spam**" (denoted as y = spam) and "not spam" (denoted as *y*=not spam), and two features, *x*1 and *x*2.

We want to classify a new email with the following **features**:

***x*1=buy**

***x*2=discount**

Let's assume we have already calculated the following probabilities from our training dataset:

1. Prior probabilities:

- *P*(spam)=0.4 // Consider it as 40% out of 100%
- *P*(not spam)=0.6 // Consider it as 60%

2. Likelihoods:

- *P*(buy|spam)=0.8
- *P*(discount|spam)=0.6
- *P*(buy|not spam)=0.3
- *P*(discount|not spam)=0.5

Now, let's plug these values into the Naive Bayes classifier equation:

$$y = \arg\max_y \left(P(y) \times \prod_{i=1}^{2} P(x_i|y) \right)$$

For $y = \text{spam}$:
$$P(\text{spam}) \times P(\text{buy}|\text{spam}) \times P(\text{discount}|\text{spam}) = 0.4 \times 0.8 \times 0.6 = 0.192$$

For $y = \text{not spam}$:
$$P(\text{not spam}) \times P(\text{buy}|\text{not spam}) \times P(\text{discount}|\text{not spam}) = 0.6 \times 0.3 \times 0.5 = 0.09$$

Comparing the two values, we see that *y*=spam gives the higher result. Therefore, according to the **Naive Bayes classifier**, the predicted class for the given features "buy" and "discount" is "**spam**".

UNIT 3: SUPERVISED LEARNING

7 Mark Questions:

1. **Explain steps in classification learning with a neat diagram**

Classification :

Classification is a type of supervised machine learning task where the goal is to **predict the category or class** that a new instance or observation belongs to, **on the basis of training data**. The output variable in classification is **discrete** and represents different classes or labels.

In Classification, a program learns from the given dataset or observations and then classifies new observation into a number of classes or groups. Such as, Yes or No, 0 or 1, Spam or Not Spam, cat or dog, etc. Classes can be called as targets/labels or categories.

Classification model:

Let us consider two examples, say '**predicting whether a tumour is malignant or benign**' and '**price prediction in the domain of real estate**? **Are these two problems same in nature?** The answer is 'no' It is true that both of them are problems related to prediction. However, for tumour prediction, we

are trying to predict which **category or class**, i.e. 'malignant' or 'benign', an unknown input data related to tumour belongs to. In the other case, that is, for price prediction, we are trying to predict an **absolute value and not a class.** When we are trying to predict a **categorical or nominal variable**, the problem is known as a classification problem. A classification problem is one where the output variable is a **category** such as 'red' or 'blue' or 'malignant tumour' or 'benign tumour' etc. Whereas when we are trying to predict a numerical variable such as 'price', 'weight', etc. the problem falls under the category of **regression**.

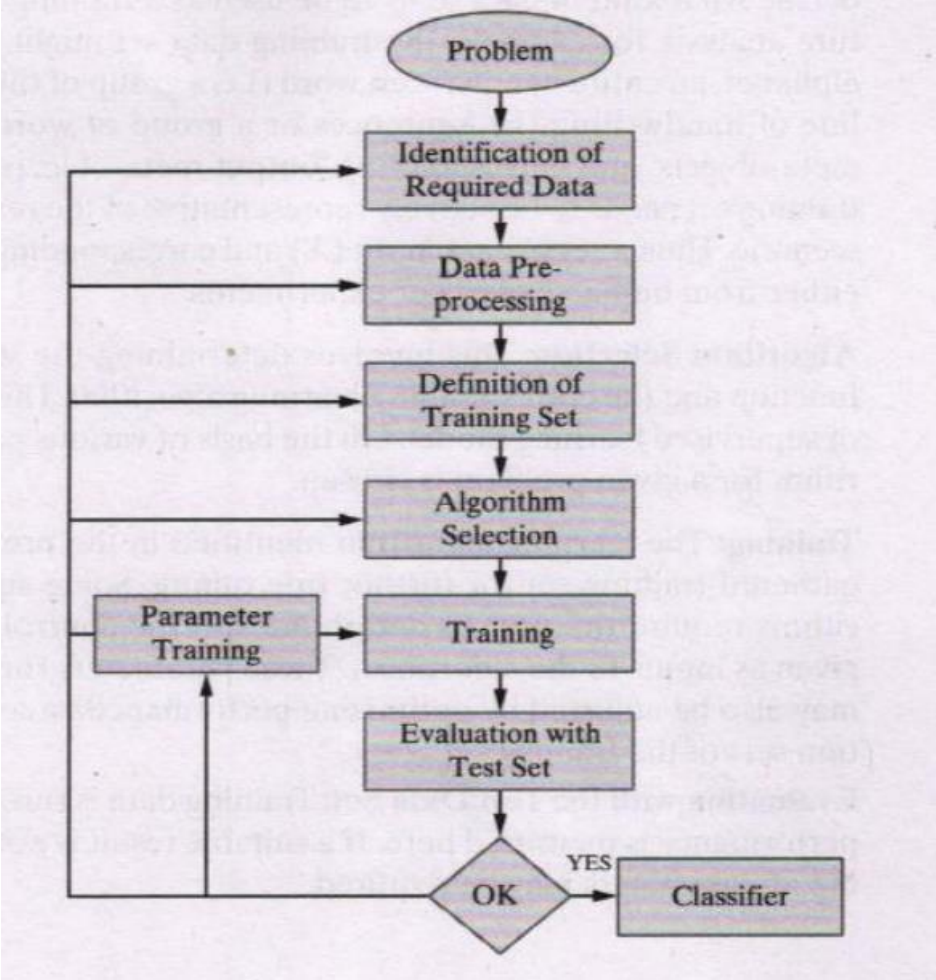
Classification Learning Steps:

1. **Problem Identification:**

Identifying the problem is the first step in the supervised learning model. The problem needs to be a well-formed problem, i.e. a **problem with well-defined goals and benefit**, which has a long-term impact.

2. **Identification of Required Data:**

On the basis of the problem identified above, the required data set that precisely represents the identified problem needs to be identified/evaluated. For example: If the problem is to predict whether a tumour is malignant or benign, then the corresponding patient data sets related to malignant tumour and benign tumours are to be identified.



3. Data Pre-processing:

This is related to the **cleaning/transforming the dataset**. This step ensures that all the unnecessary/irrelevant data elements are removed. Data pre-processing refers to the transformations applied to the identified data **before feeding the same into the algorithm**. Because the data is gathered from different sources, it is usually collected in a raw format and is not ready for immediate analysis. This step ensures that the data is ready to be fed into the machine learning algorithm.

4. Definition of Training Data Set:

Before starting the analysis, the user should decide **what kind of data set is to be used as a training set.** In the case of signature analysis, for example, the training data set might be a single handwritten alphabet, an entire handwritten word(i.e. a group of the alphabets) or an entire line of

handwriting (i.e. sentences or a group of words). Thus, a set of 'input meta-objects' and corresponding 'output meta-objects' are also gathered. The training set needs to be actively representative of the real-world use of the given scenario. Thus, a set of data input (X) and corresponding outputs (Y) is gathered either from human experts or experiments.

5. Algorithm Selection:

This involves determining the structure of the learning function and the corresponding learning algorithm. This is the **most critical step** of supervised learning model. On the basis of various parameters, the best algorithm for a given problem is chosen.

6. Training:

The learning algorithm identified in the previous step is run on the gathered training set for further fine tuning. Some supervised learning algorithms require the user to determine specific control parameters (which are given as inputs to the algorithm). These parameters (inputs given to algorithm) may also be adjusted by optimizing performance on a subset (called as validation set) of the training set.

7. Evaluation with the Test Data Set:

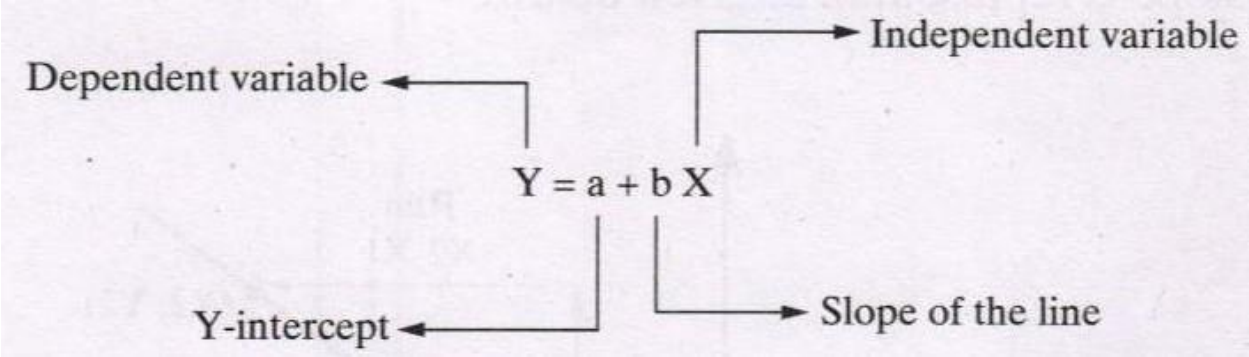
Training data is run on the algorithm, and its performance is measured here. If a suitable result is not obtained, further training of parameters may be required.

2. Explain the concept of Linear regression

Simple linear regression:

As the name indicates, simple linear regression is the simplest regression model which involves **only one independent variable or predictor** and **only one Dependent Variable or the Response Variable.**

This model assumes a **linear relationship** between the dependent variable and the predictor variable i.e, The relationship between Dependent and Independent Variable is **Linear.** Here Linear means if the value of independent variable increases or decreases then the value of Dependent variable will increases or decreases for sure.



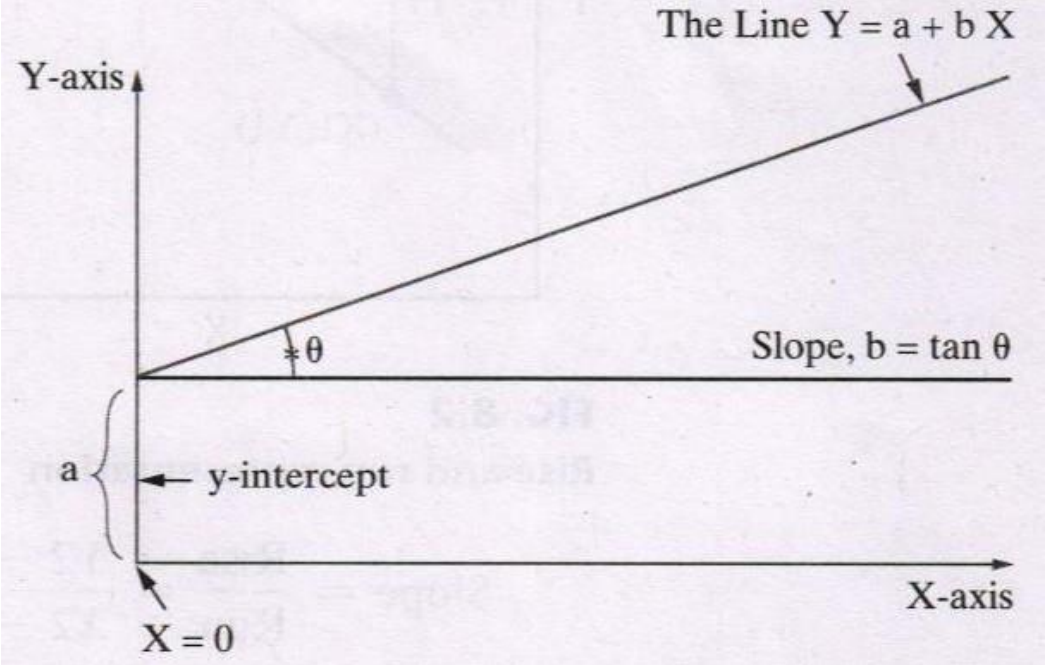
Note:

When we are trying to predict a **categorical or nominal variable**, the problem is known as a classification problem. A classification problem is one where the output variable is a **category** such as 'red' or 'blue' or 'malignant tumour' or 'benign tumour' etc. Whereas when we are trying to predict a numerical variable such as 'price', 'weight', etc. the problem falls under the category of **regression**.

Working of Simple Linear Regression:

The Working of Simple Linear Regression is very Simple that it will draw a straight line in a 2D Plane which is called **Best Fit Line**. So our aim is to find the Best Fit Line with minimal Error.

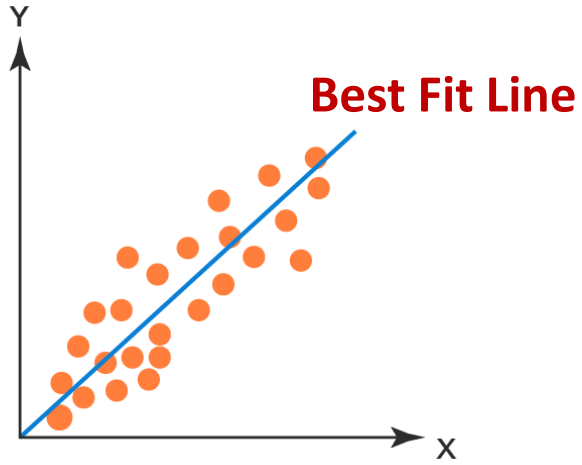
Let's Take an Example of Height and Weight such that the Problem is to Find a Height of a Person given the Weight.



Y-Intercept:

The y-intercept represents the **point where the regression line intersects the y-axis**. It is the value of the dependent variable (y) **when all independent variables (x) are equal to zero.**

Slope of the line indicates that 1 unit of increase in X is effecting the increase of slope in y.



Note: Best Fit Line is draws based on the Slope of the Line

Error in simple regression:

The regression equation model in machine learning uses the above slope—intercept format in algorithms. X and Y values are provided to the machine, and it identifies the values of a (intercept) and b (slope) by relating the values of X and Y. However, **identifying the exact match of values for a and b is not always possible.** There will be some error value (¢) associated with it. This error is called marginal or residual error.

$\hat{y}=(a+bX) + \varepsilon$

3. **Explain the concept of Logistic regression**

Logistic Regression:

Logistic regression is a versatile technique that serves **both classification and regression tasks**, depending on the context in which it is applied. It's primarily utilized as a classification method and is often referred to as logit regression.

This statistical approach is employed for predicting the **outcome of a categorical dependent variable**. In logistic regression, the **dependent variable (Y) typically takes on binary values (0 or 1),** representing two possible outcomes. Meanwhile, the **independent variables (X) are continuous in nature**, providing predictive features for the model.

Logistic regression is used when **our dependent variable is dichotomous or binary.** It just means a variable that has **only 2 outputs**, for example**, A person will survive this accident or not, The student will pass this exam or not.** The outcome can either be yes or no (2 outputs). This regression technique is similar to linear regression and can be used to predict the **Probabilities** for classification problems. Like all regression analyses, logistic regression is a **predictive analysis**.

Here's how logistic regression works:

1. **Binary Classification Task**: Logistic regression is primarily used for binary classification, where the target variable *y* has only two possible outcomes (e.g., 0 or 1, "Yes" or "No", "True" or "False").

2. **Sigmoid Function**: Logistic regression applies the **logistic function** also called the **sigmoid function** to transform the output of a linear equation **into a value between 0 and 1.**

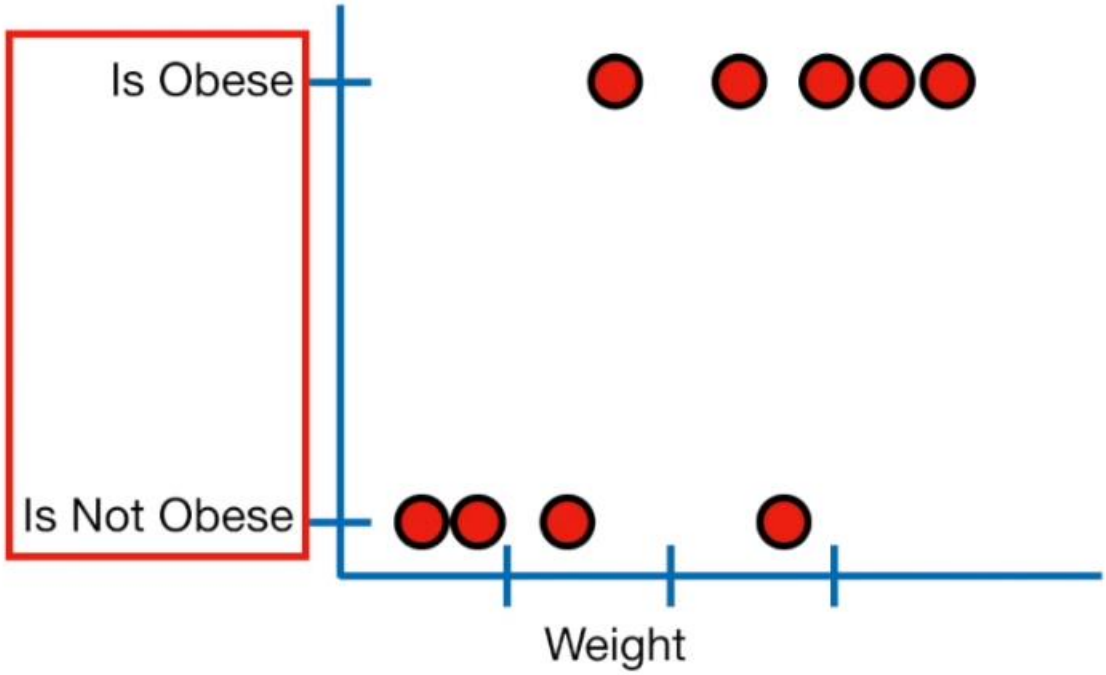
Logistic Regression is **similar to Linear Regression Except Logistic Regression predicts whether something is 0 or 1 i.e, True or False, Instead of predicting something continuous like size, house price, salary etc.**

The logistic regression model works **by transforming the Linear Regression equation using a Logistric Function,** which maps any value between **-∞ to +∞ to a range between 0 and 1**. The logistic funtion used in Logistic Regression is called as **Sigmoid** and it's equation is as follows.

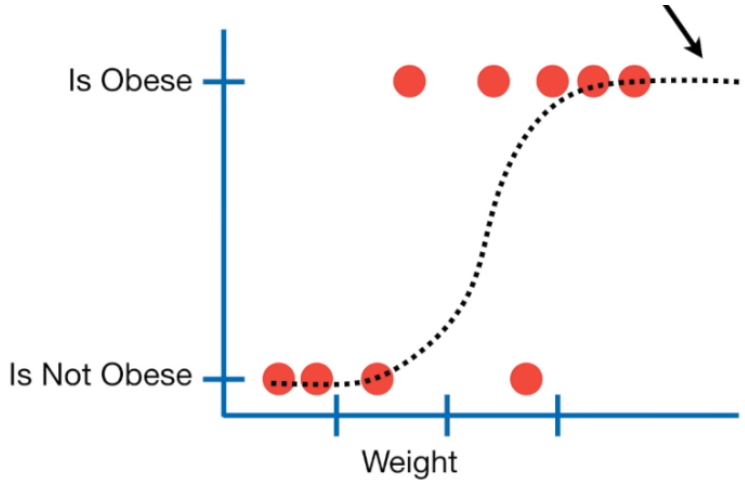
ln(P/ 1-P) = a + bX

ln(P/1 - P) = β0 + β1X1 + β2X2+ .....+ βnXn

The Logistic formula are stated in terms of the probability that Y=1 which is referred as P and the Probability that Y = 0 is 1-P



Also, instead of fitting **a line to the data**, logistic regression fits an "S" **shaped** "Logistic Function" or "**Sigmoid Function".**



4. **Discuss the SVM model in detail with different scenarios**

Support Vector Machines:

Support Vector Machines (SVM) is a **powerful supervised machine learning algorithm** used for **classification** and **regression** tasks. The primary objective of SVM is **to find the optimal hyperplane that best separates different classes** in the feature space.

The goal of the SVM algorithm is to **create the Best line or decision boundary** that can segregate the n-dimensional space into classes so that we can easily put the new data point in the correct category in the future.

SVM is a model, which can **do linear classification as well as regression**. SVM is **based on the concept of a surface, called a hyperplane,** which draws a boundary between data instances plotted in the multi-dimensional feature space. The output prediction of an SVM is one of two conceivable classes which are already defined in the training data. In summary, the SVM algorithm builds an N-dimensional hyperplane model that assigns future instances into one of the two possible output classes.

Note:

The SVM model is **does not depends on the single Hyperplane**. Instead it will create 2 more hyperplane where, one is created which is passing through

the nearest point in one class or category and one more is for another classs or category.

The goal of the SVM analysis is to find a plane, or rather a hyperplane, which separates the instances on the basis of their classes. New examples (i.e. new instances) are then mapped into that same space and predicted to belong to a class on the basis of which side of the gap the new instance will fall on. In summary, in the overall training process, the SVM algorithm analyses input data and identifies a surface in the multidimensional feature space called the hyperplane. There may be many possible hyperplanes, and one of the **challenges** with the SVM model is to find the **optimal hyperplane**.

Important Terminologies:

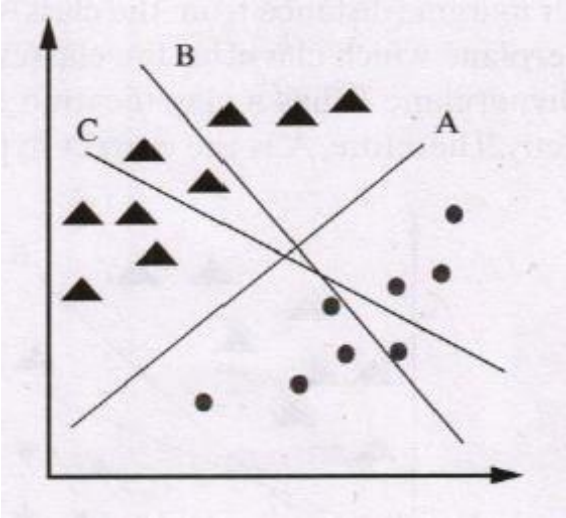
1. **Marginal Plane:**
   In Support Vector Machines (SVM), the marginal plane refers to the hyperplane or decision boundary that maximizes the margin between the support vectors of different classes. The margin is the distance between the decision boundary and the closest data points (support vectors) from each class.
2. **Support Vectors:** The data points which are passing through the Marginal Plane is called Support Vectors. It is possible to have more than one Data Points.
3. **Marginal Distance :**In Support Vector Machines (SVM), the marginal distance refers to the distance between the data point and the decision boundary (hyperplane) of the SVM model.
   Higher the marginal distance is more the Generalized our Model is.

Diff Scenario to Identifying the correct hyperplane in SVM

There may be multiple options for hyperplanes dividing the data instances belonging to the different classes. We need to identify which one will result in the best classification. Let us examine a few scenarios before arriving to that conclusion. For the sake of simplicity of visualization, the hyperplanes have been shown as straight lines in most of the diagrams.

Scenario 1:

As depicted in Figure, in this scenario, we have three hyperplanes: A, B, and C. Now, we need to identify the correct hyperplane which better segregates the two classes represented by the triangles and circles. As we can see, hyperplane 'A' has performed this task quite well.

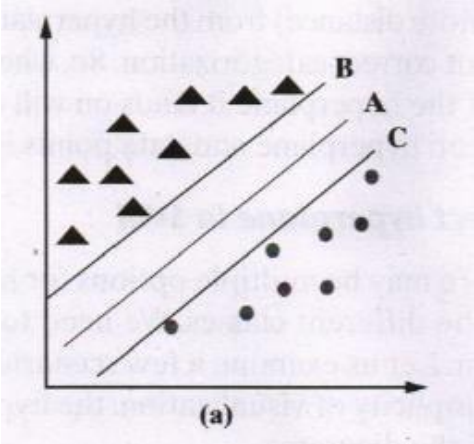


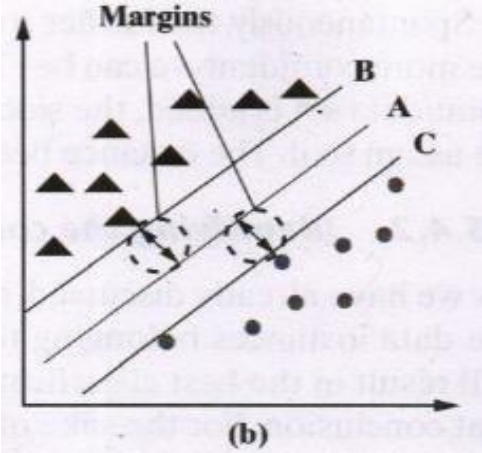
Scenario 2:

As depicted in figure below, we have three hyperplanes: A, B, and C. We have to identify the correct hyperplane which classifies the triangles and circles in the best possible way. Here, maximizing the distances between the nearest data points of both the classes and hyperplane will help us decide the correct hyperplane. This distance is called as margin.

We can see that the margin for hyperplane A is high as compared to those for both B and C. Hence, hyperplane A is the correct hyperplane.

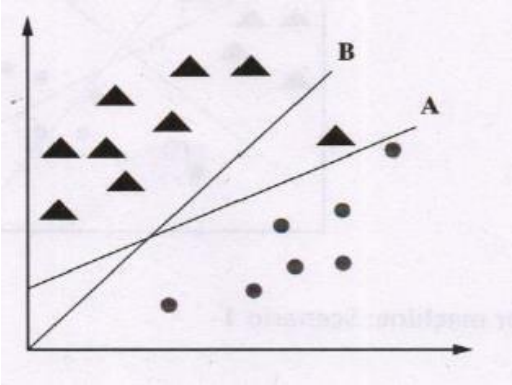
Another quick reason for selecting the hyperplane with higher margin (distance) is robustness. If we select a hyperplane having a lower margin (distance), then there is a **high probability of misclassification.**





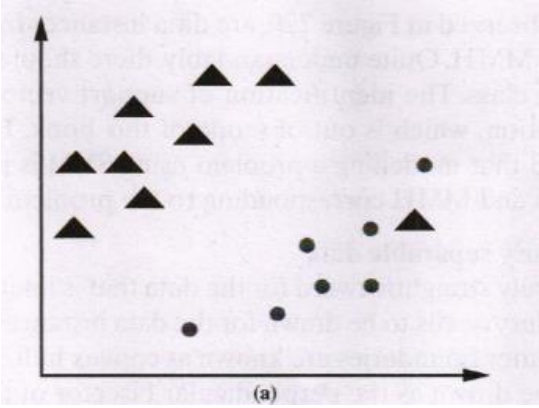
Scenario 3:

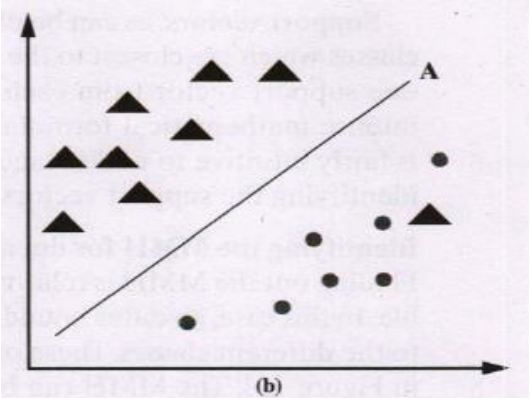
When we use the rules as discussed in the previous section to identify the correct hyperplane in the scenario shown in Figure there may a chance of selecting hyperplane B as it has a higher margin (distance from the class) than A. But, **here is the catch**; SVM selects the hyperplane which classifies the classes accurately before maximizing the margin. Here, hyperplane B has a **classification error**, and A has classified all data instances correctly. Therefore, A is the correct hyperplane.



Scenario 4:

In this scenario, as shown in Figure a, it is not possible to distinctly segregate the two classes by using a straight line, as one data instance belonging to one of the classes (triangle) lies in the territory of the other class (circle) as an outlier. One triangle at the other end is like an outlier for the triangle class. SVM has a feature to **ignore outliers** and find the hyperplane that has the maximum margin. Hence, we can say that SVM is robust to outliers. So Consider the Figure b as the Final Hyperplane.





5. **Explain the concept of KNN with an example**
   k-Nearest Neighbour (kNN)

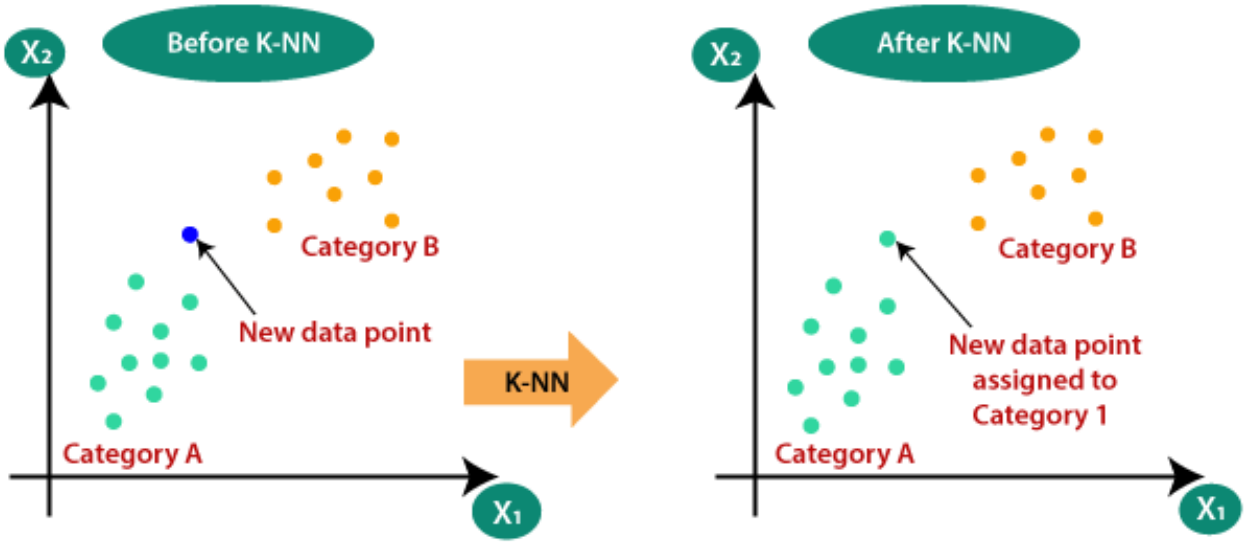
The kNN algorithm is a simple but extremely powerful **Classification** and **Regression** algorithm. The name of the algorithm originates from the underlying philosophy of KNN —i.e. **people having similar background or mindset tend to stay close to each other**. In other words, neighbours in a locality have a similar background.

In the same way, as a part of the KNN algorithm, **the unknown and unlabelled data which comes for a prediction problem is judged on the basis of the training data set elements which are similar to the unknown element. So, the class label of the unknown element is assigned on the basis of the class labels of the similar training data set elements.**

The k-nearest neighbors (KNN) algorithm is often referred to as a "**LAZY**" learning algorithm because it does **not involve a training phase** where the model learns explicit patterns from the training data. Instead, during the training phase, KNN **simply memorizes the entire training dataset**. This means that the model does not attempt to generalize or build a compact representation of the data.

Hyper Parameter: K

Hyperparameters are parameters whose values are set before the learning process begins.



Algorithm:

Step 1 – For implementing any algorithm, we need dataset. So in the first step of KNN, we
must load the training as well as test data.

Step 2 – Next, we need to choose the value of K (Initialize the K) i.e. the nearest data points. K can be any integer. **K is a Hyper Parameter.**

Step 3 – For each point in the test data do the following

3.1 – Calculate the distance between test data and each row of training data with the help of **Euclidean distance.**

3.2 – Now, based on the distance value, sort them in ascending order.

3.3 – Next, it will choose the top K rows from the **sorted array**. i.e First K entries from the array.

3.4 – Now, it will assign a class to the test point based on most frequent class of these
rows.

3.5 – If it is **a Regression problem** then return the **Mean** of the K labels.

3.6 – If it is **a Classification Problem** ten return the **Mode** of the K labels.

Step 4 – End

Example: Write a Python Code

```
import numpy as np                    # Numerical computations.
import matplotlib.pyplot as plt       # Data Visualization
from sklearn.neighbors import NearestNeighbors

# Define the Dataset using the 2-D Array

A = np.array(
             [
                 [3.1, 2.3],
                 [2.3, 4.2],
                 [3.9, 3.5],
                 [3.7, 6.4],
                 [4.8, 1.9],
                 [8.3, 3.1],
                 [5.2, 7.5],
                 [4.8, 4.7],
                 [3.5, 5.1],
                 [4.4, 2.9],
             ]
           )

plt.figure()
plt.title('Input data')
plt.scatter(A[:,0], A[:,1], marker = 'x', s = 50, color = 'red')

# Here A[:] means the slicing has no range it will take entire data set
# A[:, 0] - This means consider full data set and from the data set take only the
1st column

# Find the nearest neighbour for the new data point [5.2, 2.9]
test_data = [5.2, 2.9]

knn_model = NearestNeighbors(n_neighbors = 3, algorithm = 'auto')


knn_model.fit(A)

distances, indices = knn_model.kneighbors([test_data])

print("\nK Nearest Neighbors:")
```

```
for rank, index in enumerate(indices[0][:3], start = 1):    # Iterates over the
indices of the k-nearest neighbors. indices[0][:3] because indices  is a 2D array
which has only one row which contains the index number of the Nearest Neighbors.
    print(str(rank) + " is", A[index])                      # Print Nearest
Neighbors with their x and y axis value.


plt.figure()
plt.title('Nearest neighbors')
plt.scatter(A[:, 0], A[:, 1], marker = 'x', s = 100, color = 'red')
plt.scatter(test_data[0], test_data[1],marker = 'x', s = 100, color = 'blue')
plt.show()
```

6. Explain the concept of decision trees with an example

Decision tree learning is one of the **most widely adopted algorithms for classification**. As the name indicates, it builds a model in the form of a **tree structure**.

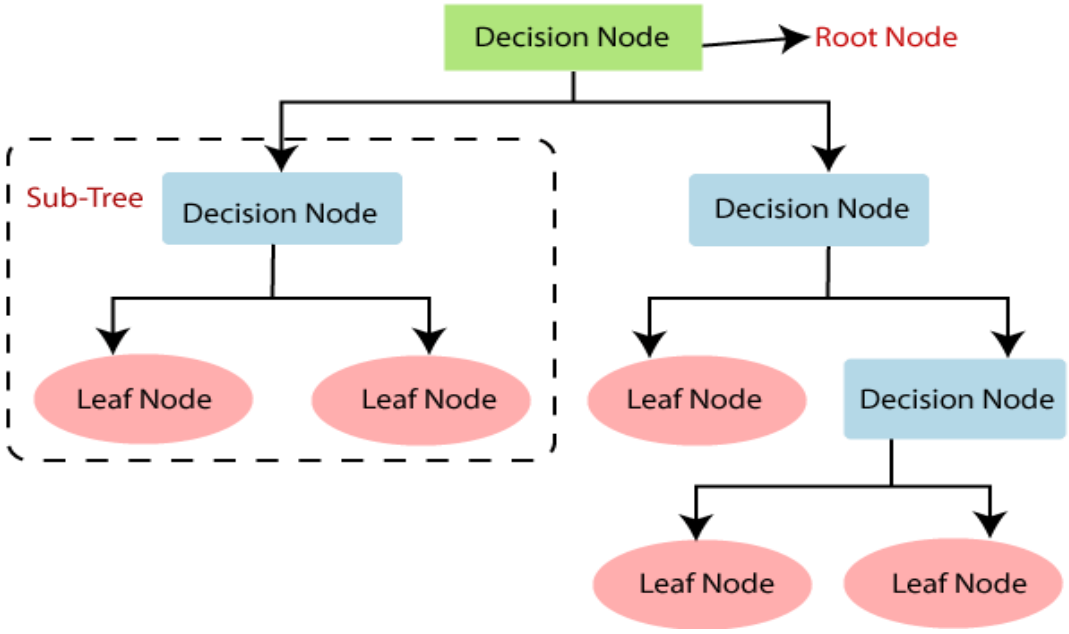
It has a **hierarchical tree structure consisting of a root node, branches, internal nodes, and leaf nodes.** Decision trees are used for **classification** and **regression** tasks, providing easy-to-understand models.

Its grouping exactness is focused with different strategies, and it is exceptionally productive. **A decision tree is used for multi-dimensional analysis with multiple classes**. It is characterized by fast execution time and ease in the interpretation of the rules.

A Decision tree is usually represented in the format given Below

Algorithm for decision tree

Step-1: Begin the tree with the root node, says S, which contains the complete dataset.
Step-2: Find the best attribute in the dataset using Attribute Selection Measure (ASM).
Step-3: Divide the S into subsets that contains possible values for the best attributes.
Step-4: Generate the decision tree node, which contains the best attribute.
Step-5: Recursively make new decision trees using the subsets of the dataset created in **step -3:** Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.



Example: Build a decision tree using the given following data

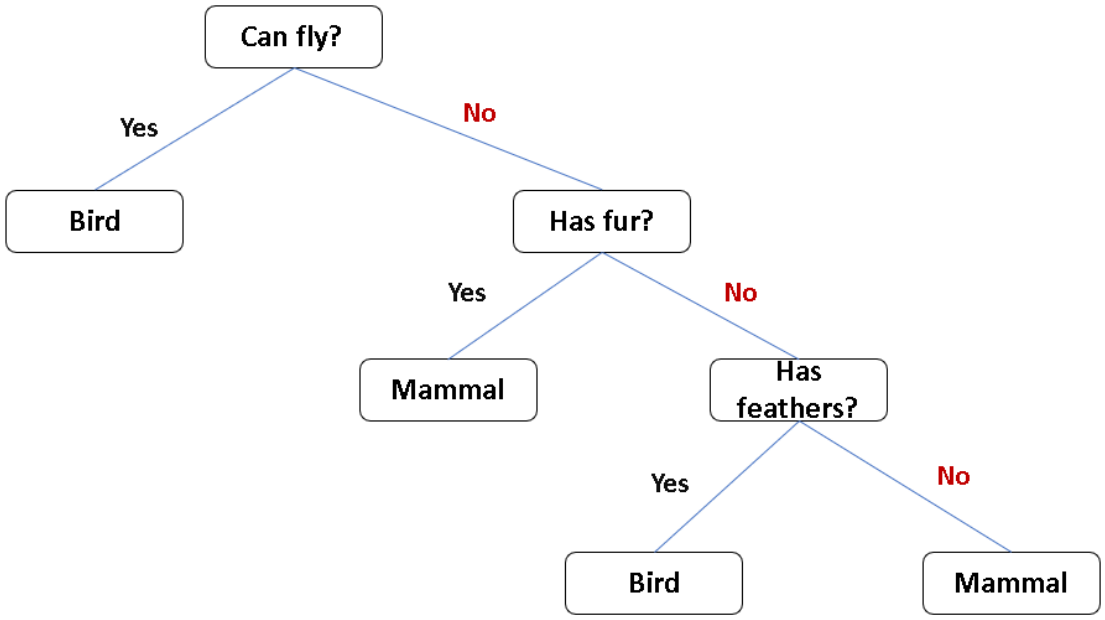
Animal	Has fur (Hair)	Has feathers	Lays eggs	Can fly	Type
Dog	Yes	No	No	No	**Mammal**
Eagle	No	Yes	No	Yes	**Bird**
Platypus	Yes	No	Yes	No	**Mammal**
Sparrow	No	Yes	No	Yes	**Bird**
Bat	Yes	No	No	Yes	**Mammal**
Ostrich	No	Yes	Yes	No	**Bird**

Output: Decision Tree

1. **Choosing the root node:** We start by selecting the feature that best splits the data. We can use metrics like Gini impurity or information gain to measure the effectiveness of

each feature. In this case, let's choose "Can fly" as the root node because it best separates birds from mammals.

2. **Splitting the data:** We partition the dataset based on the values of the selected feature. For "Can fly," we have two branches: "Yes" and "No."
3. **Repeat:** We repeat the process for each branch, selecting the best feature to split the data until we reach a stopping criterion (e.g., maximum depth, minimum number of samples per leaf).

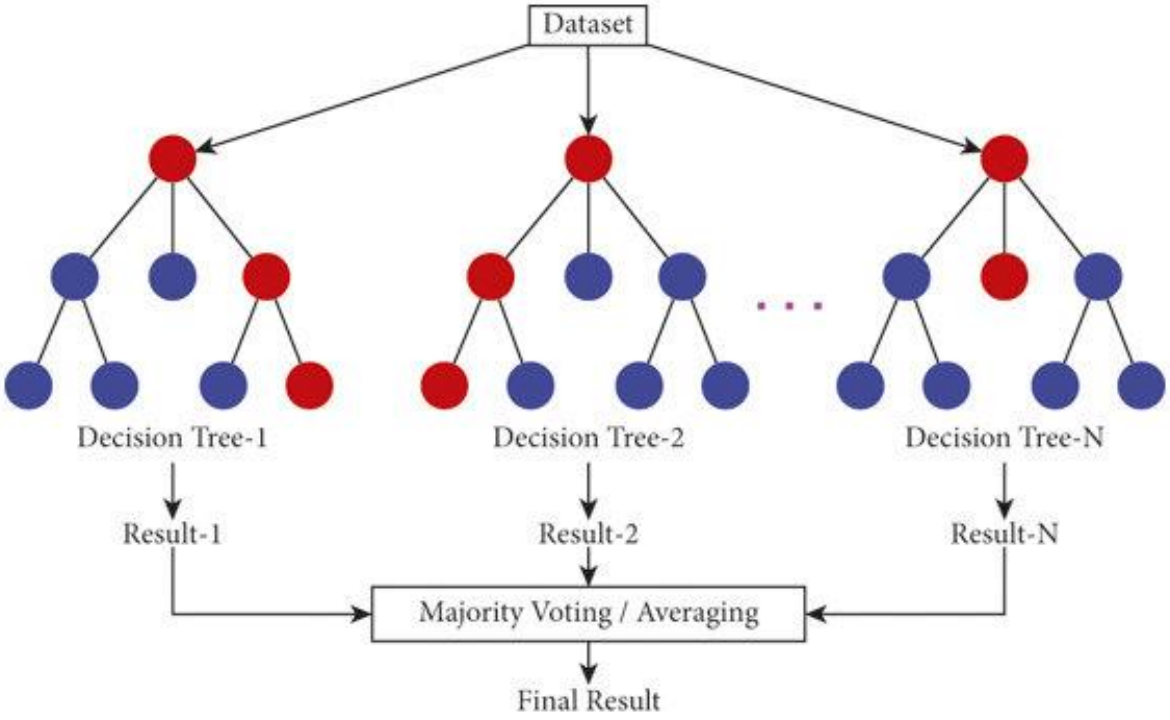


7. Explain the concept of Random Forest with a neat diagram

Random Forest:

Random forest is an **ensemble** classifier, i.e. a combining classifier that uses and **combines many decision tree classifiers.** It is also used for Regression.

Random Forest **combines the output of multiple decision trees** to reach a single result. It combines the **opinions of many** "**trees**" i.e, individual models to make better predictions, creating a more robust and accurate overall model.



Note: Random Subset includes both **Random Subset of the Full Dataset** as well as **Random Features** from the Full Dataset

Working of the Random Forest Model:

1. Random Forest is a popular group learning method that builds multiple Decision trees and aggregates their outcomes or output to make a final prediction.
2. It works by creating a set of decision trees, where each tree is trained on **a randomly selected subset of the training data** and a **randomly selected subset of features**. (Sampling)
3. The final prediction is then made by aggregating the prediction of all the trees on the basis of the majority votes from the 'n' trees. The randomness in selecting the data and feaures helps to increase the accuracy of the model.

Random Forest algorithm :

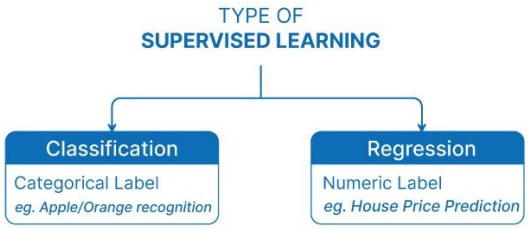
1. If there are N variables or features in the input data set, select a subset of 'm' (m<N) features at random out of the N features. Also, the observations or data instances should be picked randomly.
2. Use the best split principle on these 'm' features to calculate the number of nodes 'd'
3. Keep splitting the nodes to child nodes till the tree is grown to the maximum possible extent.
4. Select a different subset of the training data 'with replacement' to train another decision tree following steps (1) to (3). Repeat this to build and train 'n' decision trees.
5. Final class assignment is done on the basis of the majority votes from the 'n' trees.

8. Explain the types of supervised learning algorithms with examples

Supervised Machine Learning

Supervised learning is the types of machine learning in which machines are trained using well "**labelled**" **training data**, and on basis of that data, machines predict the output. The labelled data means some input data is already **tagged** or **labelled** with the **correct output.**

In supervised learning, **the training data provided to the machines work as the supervisor** that teaches the machines to predict the output correctly. It applies the same concept as a student learns in the **supervision of the teacher**. Supervised learning is a process of providing input data as well as correct output data to the machine learning model. The aim of a supervised learning algorithm is to find a mapping function to **map** the **input variable(x)** with the **output variable(y).**



Classification Learning Algorithms:

1. KNN
2. SVM
3. Decision Tree
4. Random Forest

Regression Learning Algorithms:

1. Linear Regression
2. Multiple Linear Regression
3. Polynomial Regression
4. Logistic Regression

Classification:

1. **k-Nearest Neighbour (kNN):**

The kNN algorithm is a simple but extremely powerful **Classification** and **Regression** algorithm. The name of the algorithm originates from the underlying philosophy of KNN —i.e. **people having similar background or mindset tend to stay close to each other**. In other words, neighbours in a locality have a similar background.

In the same way, as a part of the KNN algorithm, **the unknown and unlabelled data which comes for a prediction problem is judged on the basis of the training data set elements which are similar to the unknown element. So, the class label of the unknown element is assigned on the basis of the class labels of the similar training data set elements.**

The k-nearest neighbors (KNN) algorithm is often referred to as a "**LAZY**" learning algorithm because it does **not involve a training phase** where the model learns explicit patterns from the training data. Instead, during the training phase, KNN **simply memorizes the entire training dataset**. This means that the model does not attempt to generalize or build a compact representation of the data.

2. Support Vector Machines:

SVM is a **powerful supervised machine learning algorithm** used for **classification** and **regression** tasks. The primary objective of SVM is **to find the optimal hyperplane that best separates different classes** in the feature space.

The goal of the SVM algorithm is to **create the Best line or decision boundary** that can segregate the n-dimensional space into classes so that we can easily put the new data point in the correct category in the future.

SVM is a model, which can **do linear classification as well as regression**. SVM is **based on the concept of a surface, called a hyperplane,** which draws a boundary between data instances plotted in the multi-dimensional feature space. The output prediction of an SVM is one of two conceivable classes which are already defined in the training data. In summary, the SVM algorithm builds an N-dimensional hyperplane model that assigns future instances into one of
the two possible output classes.

3. Decision Tree

Decision tree learning is one of the **most widely adopted algorithms for classification**. As the name indicates, it builds a model in the form of a **tree structure**.

It has a **hierarchical tree structure consisting of a root node, branches, internal nodes, and leaf nodes.** Decision trees are used for **classification** and **regression** tasks, providing easy-to-understand models.

Its grouping exactness is focused with different strategies, and it is exceptionally productive. **A decision tree is used for multi-dimensional analysis with multiple classes**. It is characterized by fast execution time and ease in the interpretation of the rules.

4. **Random Forest:**

   Random forest is an **ensemble** classifier, i.e. a combining classifier that uses and **combines many decision tree classifiers.** It is also used for Regression.

   Random Forest **combines the output of multiple decision trees** to reach a single result. It combines the **opinions of many** "**trees**" i.e, individual models to make better predictions, creating a more robust and accurate overall model.

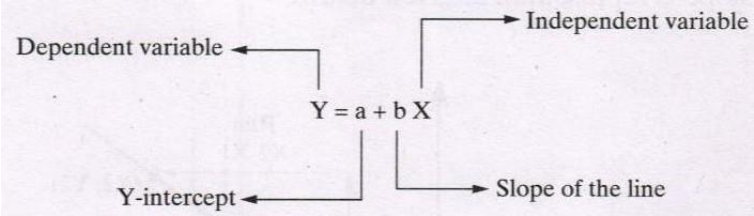
Regression:

1. Linear Regression

   Simple linear regression:

   As the name indicates, simple linear regression is the simplest regression model which involves **only one independent variable or predictor** and **only one Dependent Variable or the Response Variable.**

   This model assumes a **linear relationship** between the dependent variable and the predictor variable i.e, The relationship between Dependent and Independent Variable is **Linear.** Here Linear means if the value of independent variable increases or decreses then the value of Dependent varible will increases or decreases for sure.



2. **Multiple Linear Regression:**

In a multiple regression model, two or more independent variables, i.e. predictors are involved in the model. In the context of simple linear regression, we considered Price of a Property as the dependent variable and the Area of the Property (in sq. m.) as the predictor variable. However, location, floor, number of years since purchase, amenities available, etc. are also important predictors which should not be ignored. Thus, if we consider Price of a Property (in ₹) as the dependent variable and Area of the Property (in sq.m.), location, floor, number of years since purchase and amenities available as the independent variables, we can form a multiple regression equation as shown below:

Priceproperty = f (Areaproperty, location, floor, Ageing, Amenities)

The simple linear regression model and the multiple regression model assume that the dependent variable is continuous.

3. Polynomial Regression:

A simple linear regression algorithm **only works when the relationship between the data is linear**. But **suppose we have non-linear data**, then linear regression **will not be able to draw a best-fit line**. Simple regression analysis fails in such conditions. Consider the below diagram, which has a non-linear relationship, and you can see the linear regression results on it, which does not perform well, meaning it does not come close to reality. Hence, we introduce polynomial regression to overcome this problem, which helps identify the curvilinear relationship between independent and dependent variables.

4. Logistic Regression:

   Logistic regression is a versatile technique that serves **both classification and regression tasks**, depending on the context in which it is applied. It's primarily utilized as a classification method and is often referred to as logit regression.

   This statistical approach is employed for predicting the **outcome of a categorical dependent variable**. In logistic regression, the **dependent variable (Y) typically takes on binary values (0 or 1),** representing two possible outcomes. Meanwhile, the **independent variables (X) are continuous in nature**, providing predictive features for the model.

   Logistic regression is used when **our dependent variable is dichotomous or binary.** It just means a variable that has **only 2 outputs**, for example**, A person will survive this accident or not, The student will pass this exam or not.** The outcome can either be yes or no (2 outputs). This regression technique is similar to linear regression and can be used to predict the **Probabilities** for classification problems. Like all regression analyses, logistic regression is a **predictive analysis**.

10 Mark Questions:

1. **What is Supervised Machine Learning? List different algorithms used in Classification and Regression and Explain any two algorithms with python code for each type.**

Supervised Machine Learning

Supervised learning is the types of machine learning in which machines are trained using well "**labelled**" **training data**, and on basis of that data, machines predict the output. The labelled data means some input data is already **tagged** or **labelled** with the **correct output.**

In supervised learning, **the training data provided to the machines work as the supervisor** that teaches the machines to predict the output correctly. It

applies the same concept as a student learns in the **supervision of the teacher**. Supervised learning is a process of providing input data as well as correct output data to the machine learning model. The aim of a supervised learning algorithm is to find a mapping function to **map** the **input variable(x)** with the **output variable(y).**

Classification Learning Algorithms: (Pick one algorithm from this a write the Python Code)

1. KNN
2. SVM
3. Decision Tree
4. Random Forest

Regression Learning Algorithms: (Pick one algorithm from this a write the Python Code)

1. Linear Regression
2. Multiple Linear Regression
3. Polynomial Regression
4. Logistic Regression

Here I will choose KNN for Classification and Linear Regression for Regression:

1. k-Nearest Neighbour (kNN)

The kNN algorithm is a simple but extremely powerful **Classification** and **Regression** algorithm. The name of the algorithm originates from the underlying philosophy of KNN —i.e. **people having similar background or mindset tend to stay close to each other**. In other words, neighbours in a locality have a similar background.

In the same way, as a part of the KNN algorithm, **the unknown and unlabelled data which comes for a prediction problem is judged on the basis**

of the training data set elements which are similar to the unknown element. So, the class label of the unknown element is assigned on the basis of the class labels of the similar training data set elements.

The k-nearest neighbors (KNN) algorithm is often referred to as a “**LAZY**" learning algorithm because it does **not involve a training phase** where the model learns explicit patterns from the training data. Instead, during the training phase, KNN **simply memorizes the entire training dataset**. This means that the model does not attempt to generalize or build a compact representation of the data.

Hyper Parameter: K

Hyperparameters are parameters whose values are set before the learning process begins.

Example: Write a Python Code

```
import numpy as np                    # Numerical computations.
import matplotlib.pyplot as plt       # Data Visualization
from sklearn.neighbors import NearestNeighbors

# Define the Dataset using the 2-D Array

A = np.array(
              [
                  [3.1, 2.3],
                  [2.3, 4.2],
                  [3.9, 3.5],
                  [3.7, 6.4],
                  [4.8, 1.9],
                  [8.3, 3.1],
                  [5.2, 7.5],
                  [4.8, 4.7],
                  [3.5, 5.1],
                  [4.4, 2.9],
              ]
            )

plt.figure()
plt.title('Input data')
plt.scatter(A[:,0], A[:,1], marker = 'x', s = 50, color = 'red')
```

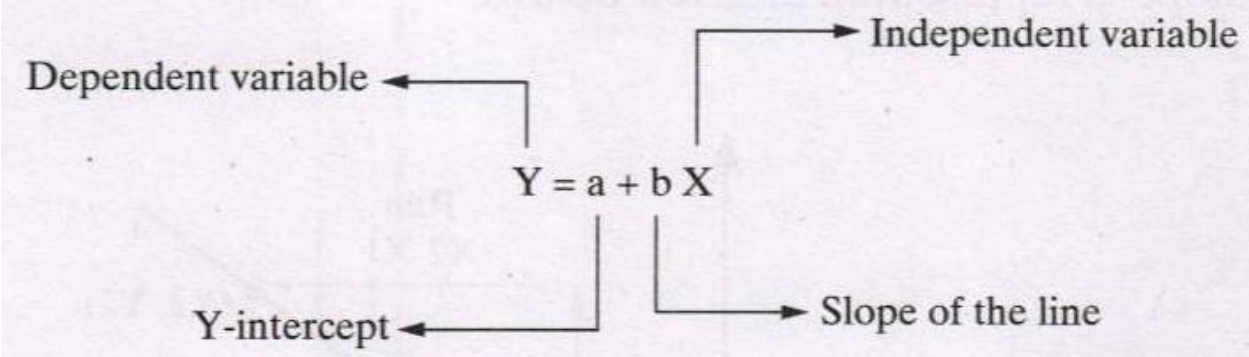
```
# Here A[:] means the slicing has no range it will take entire data set
# A[:, 0] - This means consider full data set and from the data set take only the
1st column

# Find the nearest neighbour for the new data point [5.2, 2.9]
test_data = [5.2, 2.9]

knn_model = NearestNeighbors(n_neighbors = 3, algorithm = 'auto')


knn_model.fit(A)

distances, indices = knn_model.kneighbors([test_data])

print("\nK Nearest Neighbors:")
for rank, index in enumerate(indices[0][:3], start = 1):    # Iterates over the
indices of the k-nearest neighbors. indices[0][:3] because indices  is a 2D array
which has only one row which contains the index number of the Nearest Neighbors.
    print(str(rank) + " is", A[index])                      # Print Nearest
Neighbors with their x and y axis value.


plt.figure()
plt.title('Nearest neighbors')
plt.scatter(A[:, 0], A[:, 1], marker = 'x', s = 100, color = 'red')
plt.scatter(test_data[0], test_data[1],marker = 'x', s = 100, color = 'blue')
plt.show()
```

2. Linear Regression

Simple linear regression:

As the name indicates, simple linear regression is the simplest regression model which involves **only one independent variable or predictor** and **only one Dependent Variable or the Response Variable.**

This model assumes a **linear relationship** between the dependent variable and the predictor variable i.e, The relationship between Dependent and Independent Variable is **Linear.** Here Linear means if the value of independent variable increases or decreases then the value of Dependent variable will increases or decreases for sure.



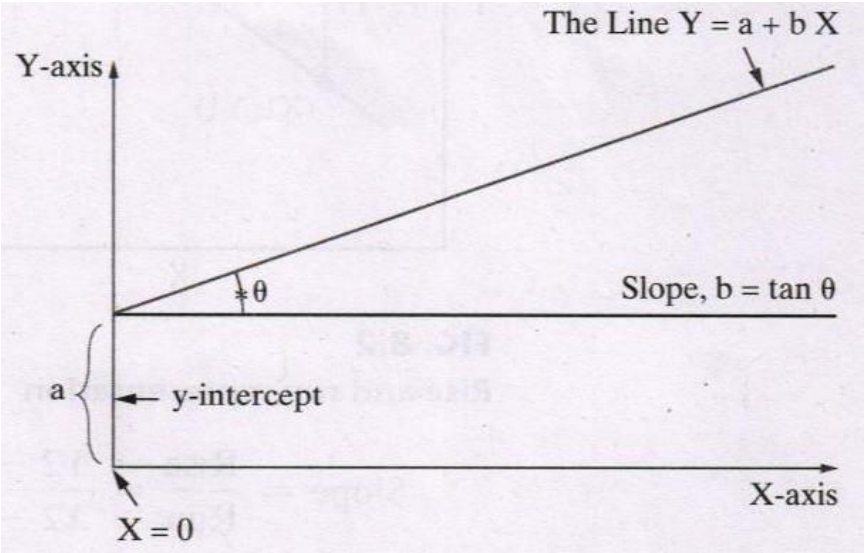
Note:

When we are trying to predict a **categorical or nominal variable**, the problem is known as a classification problem. A classification problem is one where the output variable is a **category** such as 'red' or 'blue' or 'malignant tumour' or 'benign tumour' etc. Whereas when we are trying to predict a numerical variable such as 'price', 'weight', etc. the problem falls under the category of **regression**.

Working of Simple Linear Regression:

The Working of Simple Linear Regression is very Simple that it will draw a straight line in a 2D Plane which is called **Best Fit Line**. So our aim is to find the Best Fit Line with minimal Error.

Let's Take an Example of Height and Weight such that the Problem is to Find a Height of a Person given the Weight.



Y-Intercept:

The y-intercept represents the **point where the regression line intersects the y-axis**. It is the value of the dependent variable (y) **when all independent variables (x) are equal to zero.**

Slope of the line indicates that 1 unit of increase in X is effecting the increase of slope in y.

```
# step 1 : select the ML algorithm to apply
from sklearn.linear_model import LinearRegression

#step 2 -- load training data (historic data)
# data = [year, GDP]

X =
[[2001,5.2],[2002,5.1],[2003,5.1],[2004,4.9],[2005,5.0],[2006,5.1],[2007,5.4],[20
08,5.6],[2009,5.9],[2010,5.8],[2011,6.2],
    [2012,6.0],[2013,5.8],[2014,6.1],[2015,6.4],[2016,6.6],[2017,6.6],[2018,6.8],
[2019,6.85],[2020,5.9]]
Y =
[2.5,2.52,2.54,2.48,2.52,2.54,2.55,2.7,2.9,3.2,3.16,3.28,3.2,3.15,3.26,3.29,3.17,
3.25,3.29,3.18]

len(X), len(Y)

# step 3 -- create a model
LinR_model = LinearRegression()

# step 4 -- fit model to data (training the model)
LinR_model.fit(X, Y)

# step 5 -- predict outcome for the new observaton
prediction = LinR_model.predict([[2021,6.1]])
print(prediction)

#predict outcome for another new observaton
prediction_2022 = LinR_model.predict([[2022,6.4]])
print(prediction_2022)
```

2. **Define Regression and list different types of Regression? Explain the linear regression algorithm. Outline the steps in python code to implement linear regression algorithm using standard Machine Learning toolkit.**

Regression is a type of supervised machine learning task where the goal is to predict a **continuous numerical value** or **outcome** based on input features.

Regression is a type of supervised machine learning where algorithms **learn** from the data to predict continuous values such as sales, salary, weight, or temperature.

Let us take the example of real estate to understand the concept of regression. To **know the price of an apartment**, if we can build a model which can predict the correct value of a real estate if it has certain standard inputs such as **area** (sq. m.) **of the property**, **location**, **floor**, **amenities available** etc., then we can solve real estate price prediction problem.

Many problems related to prediction of **numerical value** can be solved using the regression model. In the context of regression, **dependent variable (Y) is the one whose value is to be predicted**. This variable is presumed to be functionally related to one (say, X) or more **independent variables** called **predictors**. In other words, the dependent variable depends on independent variable(s) or predictor(s).

Regression is essentially finding a relationship (or) association between the dependent variable (Y) and the independent variable(s) (X), i.e. to find the function 'f' for the association Y = f (X).

Common Regression Algorithms:

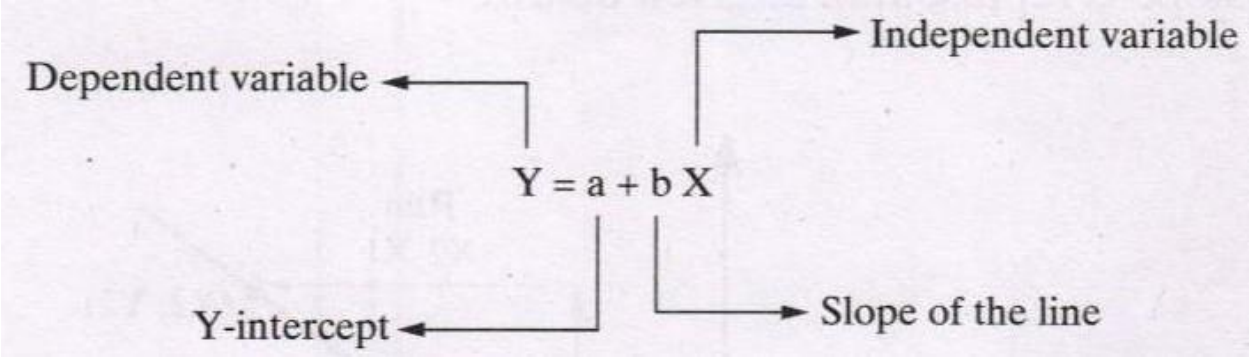
The most common regression algorithms are as follows:

1. **Simple linear regression**
2. **Multiple linear regression**
3. **Polynomial regression**
4. Multivariate adaptive regression splines
5. **Logistic regression**
6. Maximum likelihood estimation (least squares)

Simple linear regression:

As the name indicates, simple linear regression is the simplest regression model which involves **only one independent variable or predictor** and **only one Dependent Variable or the Response Variable.**

This model assumes a **linear relationship** between the dependent variable and the predictor variable i.e, The relationship between Dependent and Independent Variable is **Linear.** Here Linear means if the value of independent variable increases or decreases then the value of Dependent variable will increases or decreases for sure.



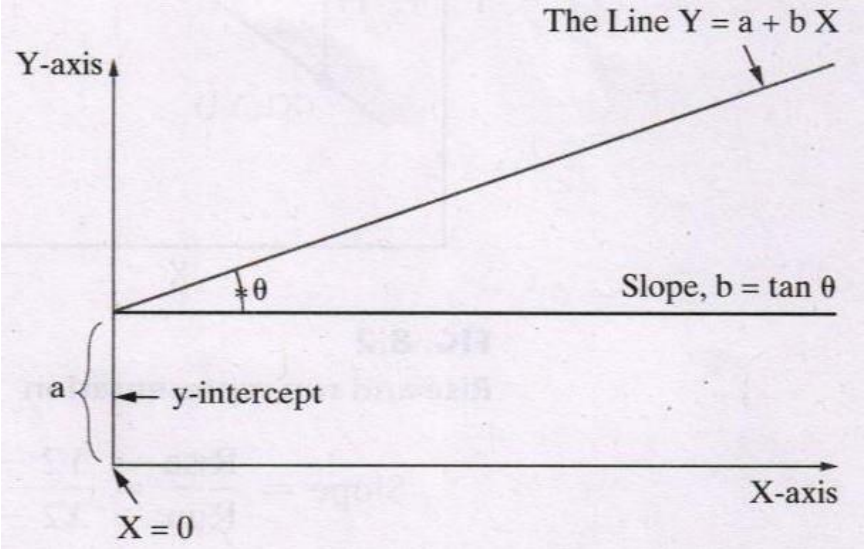
Note:

When we are trying to predict a **categorical or nominal variable**, the problem is known as a classification problem. A classification problem is one where the output variable is a **category** such as 'red' or 'blue' or 'malignant tumour' or 'benign tumour' etc. Whereas when we are trying to predict a numerical variable such as 'price', 'weight', etc. the problem falls under the category of **regression**.

Working of Simple Linear Regression:

The Working of Simple Linear Regression is very Simple that it will draw a straight line in a 2D Plane which is called **Best Fit Line**. So our aim is to find the Best Fit Line with minimal Error.

Let's Take an Example of Height and Weight such that the Problem is to Find a Height of a Person given the Weight.



Y-Intercept:

The y-intercept represents the **point where the regression line intersects the y-axis**. It is the value of the dependent variable (y) **when all independent variables (x) are equal to zero.**

Slope of the line indicates that 1 unit of increase in X is effecting the increase of slope in y.

```
# step 1 : select the ML algorithm to apply
from sklearn.linear_model import LinearRegression

#step 2 -- load training data (historic data)
# data = [year, GDP]

X =
[[2001,5.2],[2002,5.1],[2003,5.1],[2004,4.9],[2005,5.0],[2006,5.1],[2007,5.4],[20
08,5.6],[2009,5.9],[2010,5.8],[2011,6.2],
    [2012,6.0],[2013,5.8],[2014,6.1],[2015,6.4],[2016,6.6],[2017,6.6],[2018,6.8],
[2019,6.85],[2020,5.9]]
Y =
[2.5,2.52,2.54,2.48,2.52,2.54,2.55,2.7,2.9,3.2,3.16,3.28,3.2,3.15,3.26,3.29,3.17,
3.25,3.29,3.18]

len(X), len(Y)

# step 3 -- create a model
LinR_model = LinearRegression()
```

```
# step 4 -- fit model to data (training the model)
LinR_model.fit(X, Y)

# step 5 -- predict outcome for the new observaton
prediction = LinR_model.predict([[2021,6.1]])
print(prediction)

#predict outcome for another new observaton
prediction_2022 = LinR_model.predict([[2022,6.4]])
print(prediction_2022)
```

UNIT 4: UNSUPERVISED LEARNING

7 Mark Questions:

1. **Compare and contrast supervised and unsupervised learning**

Aspect	**Supervised Learning**	**Unsupervised Learning**
Definition	**Learning from labeled data**, where input-output pairs are given.	Learning from **unlabeled data**, where the algorithm must infer patterns without explicit output labels.
Objective	To **predict or classify new data** based on past observations.	To **discover hidden patterns or structures in data.**
Input Data	Requires **labeled data** (input-output pairs).	Works with **unlabeled or unstructured data.**
Feedback	Receives feedback **during training** (error or loss function).	No feedback during training; relies on inherent data properties.
Types	**Classification, Regression.**	**Clustering, Dimensionality Reduction, Association.**

Usage	Use **Off-line analysis** of Data.	Use **Real-Time analysis** of Data.
Training Complexity	Often more complex due to the need for labeled data.	Generally simpler as it doesn't require labeled data.
Applicability	Commonly used in tasks where labeled data is available.	Used when there's no labeled data or to explore data structure.
Examples of Algorithms	Decision Trees, Support Vector Machines, Neural Networks.	K-Means, Hierarchical Clustering, Principal Component Analysis (PCA).

2. **Explain the types of unsupervised algorithms with examples.**
   Aa

3. **Explain the concept of Market Basket Analysis with examples**
   ASSOCIATION RULES :

   Association rule presents a methodology that is useful for **identifying interesting relationships hidden in large data sets**. It is also known as association analysis, and the discovered relationships can be represented in the form of **association rules comprising a set of frequent items**.

   A common application of this analysis is the **Market Basket Analysis (MBA)** where the aim is to find **associations between items purchased together**, that retailers use for **cross-selling of their products**.

   For example, every large grocery store accumulates a large volume of data about the **buying pattern of the customers**. On the basis of the items purchased together, the retailers can push some cross-selling either by **placing the items bought together in adjacent areas or creating some combo offer with those different product types.**

The below association rule signifies that **people who have bought bread and milk have often bought egg also;** so, for the retailer, it makes sense that these items are placed together for new opportunities for cross-selling.

Market Basket Problem :

- The **problem of deriving associations from data** is of utmost **importance** in unsupervised learning. This problem is often referred to as **Market basket problem**.
- In this problem, we are given a **set of items** and **a large collection of transactions** that are subsets of these items. **The task here is to find relationship** between the presences of various items within the basket.

Key components of association rules include:

1. **Support: (Indicates how the items support the Association rule)**

Support refers to the **frequency** or **occurrence** of a particular **itemset in a dataset**. It measures how frequently **a specific combination of items appears together in the dataset.**

Support is used to identify itemsets that occur **frequently enough to be considered significant for generating association rules**. Higher support values **indicate stronger relationships between items.**

For example,

let's calculate the support for {Milk, Bread}.

Support({Milk, Bread}) = (Number of transactions containing {Milk, Bread}) / (Total number of transactions)

Support({Milk, Bread}) = 3 / 5 **= 0.6**

This means that the support for the itemset {Milk, Bread} is 0.6, indicating that it appears in 60% of the transactions.

2. Confidence:

Confidence measures the **reliability of the association rule**.

Confidence measures the reliability of the inference made by an association rule. It is the probability of seeing the **consequent** (B) in a transaction given that the transaction also contains the **antecedent** (A). A high confidence indicates a strong association between the antecedent and consequent.

For example, **let's calculate the confidence for the rule {Milk} → {Bread}.**

Confidence({Milk} → {Bread}) = (Number of transactions containing {Milk, Bread}) / (Number of transactions containing {Milk})

Confidence({Milk} → {Bread}) = 3 / 4 = 0.75

This means that the confidence for the rule {Milk} → {Bread} is 0.75, indicating that in 75% of the transactions where Milk is purchased, Bread is also purchased.

Example: Apply the Market Basket Analysis to bellow Transactions:

```
# market data
transactions = [('butter', 'milk', 'bread'),
                ('butter', 'milk', 'apple'),
                ('bread', 'milk', 'banana'),
                ('milk','bread','butter')]

We can use Apriori algorithm to solve this problem:
# creating baskets
itemsets, rules = apriori(transactions, min_support=0.5,  min_confidence=1)
# association rules
print(rules)
```

It uses **Frequent itemsets** to generate association rules and it is designed to **work on databases that contains transactions.**

This algorithm uses the Breadth First Search (BFS) and Hash Tree to calculate the itemset association efficiently.

Step-1: Determine the **Support of itemsets** and **select minimum support and Confidence**.

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

Step-3: Find all the rules of these sub itemsets that have higher confidence value than the threshold or decreesing order of Lift.

4. Explain the Apriori algorithm with its application.

Dd

APRIORI ALGORITHM:

The Apriori algorithm is a **classical algorithm used for association rule mining** in machine learning and data mining.

It is designed to **discover frequent itemsets** in transactional databases and **derive association rules from these itemsets.** The algorithm is named "**Apriori**" because **it uses the "prior" knowledge of frequent itemsets** to efficiently generate candidate itemsets.

The main idea for the Apriori algorithm is: All non-empty subsets of a frequent itemset must also be frequent.

The Apriori algorithm detects the **most frequent itemsets or elements** in a transaction database and **establishes association rules between the items**.

The method employs a “bottom-up” strategy, in which frequent subsets are expanded one item at a time (candidate generation), and groups of candidates are checked against the data. When no more successful rules can be obtained from the data, the algorithm stops.

Algorithm:

It uses **Frequent itemsets** to generate association rules and it is designed to **work on databases that contains transactions.**

This algorithm uses the Breadth First Search (BFS) and Hash Tree to calculate the itemset association efficiently.

Step-1: Determine the **Support of itemsets** and **select minimum support and Confidence**.

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

Step-3: Find all the rules of these sub itemsets that have higher confidence value than the threshold or decreesing order of Lift

The Apriori algorithm is a classic algorithm used for association rule mining in data mining and machine learning. It is particularly useful for discovering frequent itemsets in transactional datasets and extracting association rules between items.

Applications of the Apriori algorithm:

1. Market Basket Analysis:
   One of the primary applications of the Apriori algorithm is in market basket analysis, where it helps identify patterns of co-occurrence among items purchased together.
   Retailers use market basket analysis to understand customer purchasing behavior, optimize product placement, and design targeted marketing strategies such as cross-selling and upselling.

2. E-commerce Recommendations:

E-commerce platforms leverage the Apriori algorithm to generate personalized product recommendations for users based on their browsing and purchase history.
By identifying frequent itemsets in historical transaction data, e-commerce websites can recommend related or complementary products to users, enhancing their shopping experience and increasing sales.

3. Inventory Management:
   In inventory management, the Apriori algorithm can assist in optimizing stock levels and inventory replenishment strategies.
   By analyzing transaction data and identifying frequently co-purchased items, businesses can better predict demand for certain products and ensure that they have adequate stock on hand to meet customer needs.

4. Healthcare Data Analysis:
   Healthcare organizations use the Apriori algorithm to analyze patient treatment data and identify associations between medical procedures, medications, and patient outcomes.
   By uncovering patterns in treatment protocols and patient responses, healthcare providers can improve treatment efficacy, reduce costs, and enhance patient care.

5. Web Usage Mining:
   In web usage mining, the Apriori algorithm can be applied to analyze clickstream data and identify patterns of user navigation on websites.
   Website owners use this information to optimize website layout and content, personalize user experiences, and increase user engagement and conversion rates.

6. Fraud Detection:
   The Apriori algorithm can be used in fraud detection applications to identify patterns of fraudulent behavior in financial transactions or insurance claims data.

By detecting frequent combinations of suspicious activities or transactions, organizations can implement preventive measures and mitigate the risk of fraud.

5. Explain the concept of K-Means algorithm.

k-Means Clustering: (A centroid-based technique):

K-Means is one of the **most popular unsupervised machine learning algorithms used for clustering.** The algorithm aims to partition a set of **data points into 'k' clusters**, where each data point belongs to the **cluster with the nearest mean or centroid.**

The goal of clustering is to divide the **population** or set of data points into a number of groups **so that the data points within each group are more comparable to one another and different from the data points within the other groups**. It is essentially a grouping of things based on how similar and different they are to one another.

The principle of the k-means algorithm is to assign each of the 'n' data points to one of the K clusters where 'K' is a **user-defined parameter** as the number of clusters desired.

Simple algorithm of K-means:

Step 1:

Choose the number of clusters, 'k', that you want to identify.

Randomly initialize 'k' cluster 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 4: Calculate the Sum of Squared Error (SSE) to measure the quality of the clusters.

This is to measure whether the K value is good or bad.

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.

End Loop.

Algorithm Implementation:

 # step 1 -- Importing Modules

```
from sklearn.cluster import KMeans

#step 2 -- load data
# Persons with varied hieght and age
data_features = ["Hieght", "Age"]
X = [[165,19],[175,32],[136,35],[174,65],[141,28],[176,15],[131,32],
[166,6],[128,32],[179,10],[136,34],[186,20],[126,25],[176,28],[112,38],[16
9,9],[171,36],[116,25],[196,25]]


# Step-3: Declaring Model
model = KMeans(n_clusters=3)
# Fitting Model
model.fit(X)

# Step-4: Prediction on the entire data
cluster_labels = model.predict(X)
# Printing Predictions
print(cluster_labels)

x1 = []  # hieght
x2 = []  # age
for item in X:
   x1.append(item[0])
   x2.append(item[1])
print(x1)
print(x2)

# Step-5
import matplotlib.pyplot as plt
plt.scatter(x1,x2, c=model.labels_)
```

6. Explain the concept of K-Medoid algorithm.

k-Medoids Cluster:

The k-Medoids algorithm is a **variation of the k-Means algorithm** that focuses on **finding representative objects** or **medoids in the dataset to form clusters**. Instead of using the **mean** or **centroid** of the data points within a

cluster, the k-Medoids algorithm **selects actual data points** or **medoids** as cluster representatives. This makes the k-Medoids algorithm more robust to outliers compared to k-Means, as it directly uses data points as cluster centers rather than **relying on the mean**, which can be sensitive to outliers.

Because of the use of medoids from the actual representative data points, k-medoids is less influenced by the **outliers** in the data. One of the practical implementation of the k-medoids principle is the **Partitioning Around Medoids (PAM) algorithm.**

What Do you mean by medoids:

A medoid in a data set is a central point within a cluster minimizing the sum of distances to the other point.

Outliers:

Outliers refer to data points that **significantly differ** from other observations in a dataset.

For Example in a dataset of student exam scores where most students score between 60 and 90, but there is one student who scores 10. This score of 10 would be considered an outlier because it significantly differs from the rest of the scores in the dataset.

There are 2 Type of K-Medoid Clustering Algorithm:

1. **Partitioning Around Medoids (PAM)-** Suitable for Small Datsets
2. **CLustering LARge Application (CLARA) –** Large Data Sets

PAM:

Step 1: Initially Select k Random points as the **medoid** or **representative points** from the given n data points of the data set.

loop

Step 2: Assign each of the remaining points to the cluster which has the nearest representative point by finding the distance using Euclidian Distance or other.

Step 3: Randomly select a non-representative point or, in each cluster

Step 4: Swap the representative point oj with or, and compute the new SSE after swapping

Step 5: If SSEnew < SSEold, then swap oj with or, to form the new set of k representative objects;

Step 6: Refine the k clusters on the basis of the nearest representative point.

Logic continues until there is no change

end loop

Implementation: (Not Suitable for Large Dataset)

Step-1: Initially **Select k Random points** as the **medoid** or **representative points** from the given n data points of the data set.

Step-2: **Associate each data point to the closest medoid** by using any of the most common distance metrics. For example Euclidian Or Manhatten Distance.

Step-3: Once the Cluster is formed then we have to calculate the **Total Cost** of forming these Clusters. So calculate the total cost as the total sum of the distacne of the data points from the assigned medoid. **The Cost is nothing but sum of the distances of all points from the medoid of the cluster they belongs to**.

$$c = \sum_{Ci} \sum_{Pi \in Ci} |Pi - Ci|$$

Where,

Ci : Cluster Number (C1, C2, ….. , Cn) which is nothing but the Medoid

Pi : is the data point

| |: Cardinality to consider only +ve value.

Step-4: Swap one medoid point with a non-medoid point and recalculate the cost.

Step-5: If the calculated cost with new medoid point **is grater than** the previous cost, we **undo the swap** and the algorithm coverges else; we repeat step 4

7. **Explain the concept of DBSCAN algorithm in unsupervised learning**

Density-based methods – DBSCAN:

Density-based Spatial Clustering of Applications with Noise (DBSCAN) is a popular clustering algorithm in machine learning used for **grouping together data points based on their density (mass) in a given feature space.**

Unlike k-Means, which assumes that **clusters are spherical** and require the number of clusters to be predefined, DBSCAN can find **clusters of arbitrary shapes and sizes** without needing the number of clusters as an input.

The density-based clustering approach provides a solution to **identify clusters of arbitrary shapes**. The principle is based on identifying the **dense area** and **sparse area** within the data set and then run the clustering algorithm. DBSCAN is one of the popular density-based algorithm which creates clusters by using connected regions with high density.

DBSCAN is a **base algorithm** for **density-based clustering**. **It can discover clusters of different shapes and sizes from a large amount of data, which is containing noise and outliers.**

Dense and **Sparse Area:**

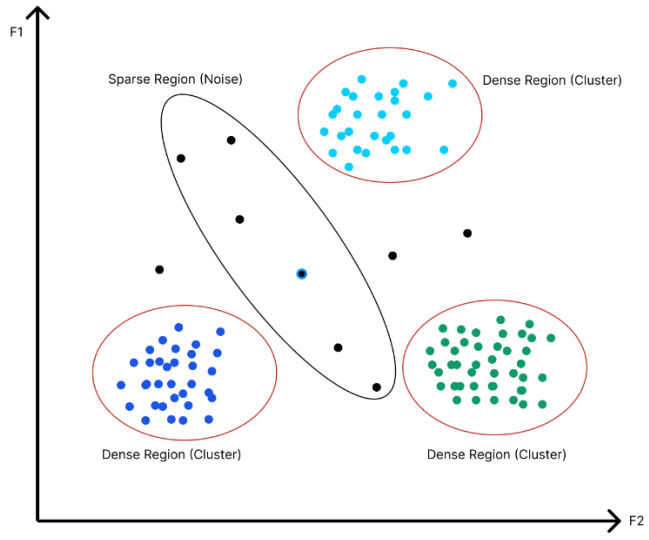
In the context of clustering algorithms like DBSCAN (Density-Based Spatial Clustering of Applications with Noise), "dense" and "sparse" areas refer to regions of the dataset **with different concentrations of data points.**

1. **Dense Area:**

A dense area in the dataset is a region where there is a **high concentration of data points**. In a dense area, data points are **closely packed together**, and there are relatively **many data points within a small area**. Dense areas often correspond to clusters in clustering algorithms, as they represent regions where the data points share similar characteristics or properties.

2. Sparse Area:

A sparse area in the dataset is a region where there is a **low concentration of data points**. In a sparse area, data points are more sparsely distributed, and there are relatively **few data points within a given area**. Sparse areas often occur between clusters or in regions of the feature space where there is little or no data present.



The DBSCAN algorithm uses two parameters:

1. **minPts:** The **minimum number of points (a threshold) clustered together for a region to be considered dense.**
2. **eps (ε):** A **distance measure** that will be used **to locate the points** in the neighborhood of any point.

   These parameters can be understood if we explore two concepts called :

   1. **Density Reachability and**
   2. **Density Connectivity.**

Reachability in terms of density **establishes a point to be reachable from another if it lies within a particular distance (eps) from it.**

Connectivity, on the other hand, involves **a transitivity based chaining-approach to determine whether points are located in a particular cluster.** For example, p and q points could be connected if p->r->s->t->q, where a->b means b is in the **neighborhood** of a.

Algorithmic steps for DBSCAN clustering :

1. The algorithm proceeds by **arbitrarily picking up a point** in the dataset (until all points have been visited).
2. If there are at least '**minPoint**' points within a radius of '**ε**' to the point then we consider **all these points to be part of the same cluster.**
3. The clusters are then **expanded by recursively repeating the neighborhood calculation** for each neighboring point

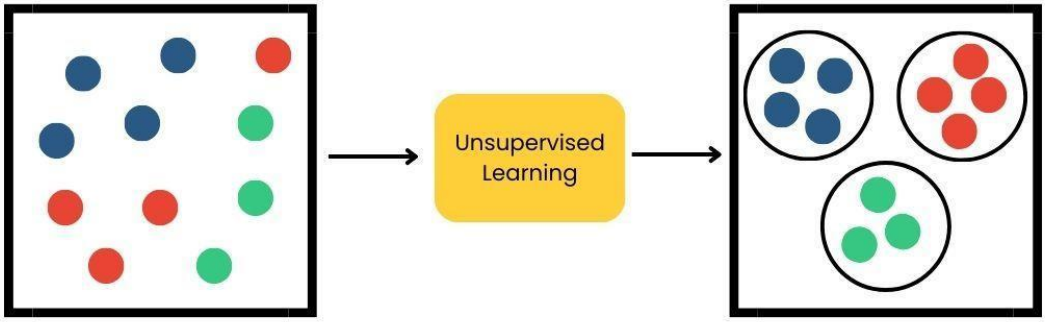
10 Mark Questions:

1. **What is Clustering? Discuss K-Means and K-Medoid algorithms and mention the difference between the two.**

Clustering methods involve **grouping untagged** data based on their similarities and differences. When two instances appear in different groups, we can infer they have dissimilar properties.

Clustering is a type of unsupervised learning, meaning that we do not need labeled data for clustering algorithms; this is one of the biggest advantages of clustering over other supervised learning like Classification.

Clustering is the process of **arranging** a group of objects in such a manner that the objects in the same group (which is referred to as a cluster) are more **similar** to **each other** than to the objects in any **other group**.



k-Means Clustering: (A centroid-based technique):

K-Means is one of the **most popular unsupervised machine learning algorithms used for clustering.** The algorithm aims to partition a set of **data points into 'k' clusters**, where each data point belongs to the **cluster with the nearest mean or centroid.**

The goal of clustering is to divide the **population** or set of data points into a number of groups **so that the data points within each group are more comparable to one another and different from the data points within the other groups**. It is essentially a grouping of things based on how similar and different they are to one another.

The principle of the k-means algorithm is to assign each of the 'n' data points to one of the K clusters where 'K' is a **user-defined parameter** as the number of clusters desired.

Simple algorithm of K-means:

Step 1:

Choose the number of clusters, 'k', that you want to identify.

Randomly initialize 'k' cluster 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 4: Calculate the Sum of Squared Error (SSE) to measure the quality of the clusters.

This is to measure whether the K value is good or bad.

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.

End Loop.

Algorithm Implementation:

```
# step 1 -- Importing Modules
from sklearn.cluster import KMeans

#step 2 -- load data
# Persons with varied hieght and age
data_features = ["Hieght", "Age"]
X = [[165,19],[175,32],[136,35],[174,65],[141,28],[176,15],[131,32],
[166,6],[128,32],[179,10],[136,34],[186,20],[126,25],[176,28],[112,38],[16
9,9],[171,36],[116,25],[196,25]]
```

```
# Step-3: Declaring Model
model = KMeans(n_clusters=3)
# Fitting Model
model.fit(X)

# Step-4: Prediction on the entire data
cluster_labels = model.predict(X)
# Printing Predictions
print(cluster_labels)

x1 = []  # hieght
x2 = []  # age
for item in X:
   x1.append(item[0])
   x2.append(item[1])
print(x1)
print(x2)

# Step-5
import matplotlib.pyplot as plt
plt.scatter(x1,x2, c=model.labels_)
```

k-Medoids Cluster:

The k-Medoids algorithm is a **variation of the k-Means algorithm** that focuses on **finding representative objects** or **medoids in the dataset to form clusters**. Instead of using the **mean** or **centroid** of the data points within a cluster, the k-Medoids algorithm **selects actual data points** or **medoids** as cluster representatives. This makes the k-Medoids algorithm more robust to outliers compared to k-Means, as it directly uses data points as cluster centers rather than **relying on the mean**, which can be sensitive to outliers.

Because of the use of medoids from the actual representative data points, k-medoids is less influenced by the **outliers** in the data. One of the practical

implementation of the k-medoids principle is the **Partitioning Around Medoids (PAM) algorithm.**

What Do you mean by medoids:

A medoid in a data set is a central point within a cluster minimizing the sum of distances to the other point.

Outliers:

Outliers refer to data points that **significantly differ** from other observations in a dataset.

For Example in a dataset of student exam scores where most students score between 60 and 90, but there is one student who scores 10. This score of 10 would be considered an outlier because it significantly differs from the rest of the scores in the dataset.

There are 2 Type of K-Medoid Clustering Algorithm:

3. **Partitioning Around Medoids (PAM)-** Suitable for Small Datsets
4. **CLustering LARge Application (CLARA) –** Large Data Sets

PAM:

Step 1: Initially Select k Random points as the **medoid** or **representative points** from the given n data points of the data set.

loop

Step 2: Assign each of the remaining points to the cluster which has the nearest representative point by finding the distance using Euclidian Distance or other.

Step 3: Randomly select a non-representative point or, in each cluster

Step 4: Swap the representative point oj with or, and compute the new SSE after swapping

Step 5: If SSEnew < SSEold, then swap oj with or, to form the new set of k representative objects;

Step 6: Refine the k clusters on the basis of the nearest representative point.

Logic continues until there is no change

end loop

Implementation: (Not Suitable for Large Dataset)

Step-1: Initially **Select k Random points** as the **medoid** or **representative points** from the given n data points of the data set.

Step-2: **Associate each data point to the closest medoid** by using any of the most common distance metrics. For example Euclidian Or Manhatten Distance.

Step-3: Once the Cluster is formed then we have to calculate the **Total Cost** of forming these Clusters. So calculate the total cost as the total sum of the distacne of the data points from the assigned medoid. **The Cost is nothing but sum of the distances of all points from the medoid of the cluster they belongs to**.

$$c = \sum_{Ci} \sum_{Pi \in Ci} |Pi - Ci|$$

Where,

Ci : Cluster Number (C1, C2, ….. , Cn) which is nothing but the Medoid

Pi : is the data point

| |: Cardinality to consider only +ve value.

Step-4: Swap one medoid point with a non-medoid point and recalculate the cost.

Step-5: If the calculated cost with new medoid point **is grater than** the previous cost, we **undo the swap** and the algorithm coverges else; we repeat step 4.

UNIT 5: NEURAL NETWORKS

7 Mark Questions:

1. **With a neat diagram explain the structure of biological neuron.**

Introduction:

Machine learning, as we have seen, **mimics the human form of learning**. On the other hand, human learning, or for that matter every action of a human being, is **controlled by the nervous system**.

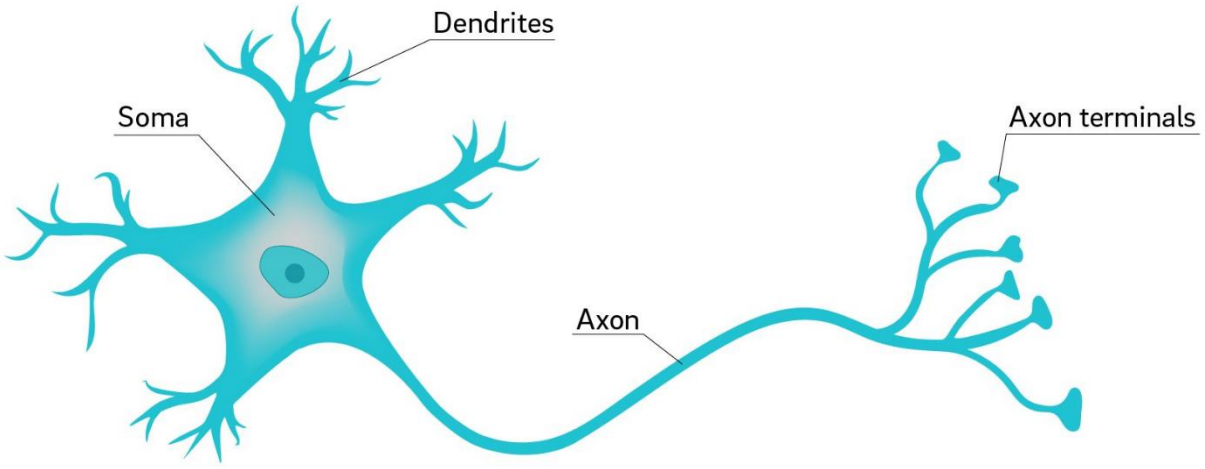
In **any human being**, the **nervous system coordinates the different actions by transmitting signals** to and from different parts of the body.

The nervous system is constituted of a special type of cell, called **neuron** or **nerve cell**, which has special structures allowing it to **receive** or **send signals** to other neurons. Neurons connect with each other to transmit signals to or receive signals from other neurons. This structure essentially forms **a network of neurons or a Neural Network.**

THE BIOLOGICAL NEURON :

The human nervous system is divided into two main parts:

1. The **Central Nervous System (CNS),** which includes the **brain** and **spinal cord**.
2. The **Peripheral Nervous System**, which includes nerves and ganglia (clusters of nerve cells) outside the brain and spinal cord.



Note:

1. The **CNS integrates all information**, in the form of **signals**, from the different parts of the body.
2. The **peripheral nervous system**, on the other hand, **connects the CNS with the limbs and organs.**
3. **Neurons** are basic structural units of the CNS.
4. A neuron is able to **receive, process**, and **transmit** information in the form of **chemical** and **electrical** signals.

A **biological neuron** has a cell body or soma to **process the impulses or signals**, dendrites to receive them, and an axon that transfers them to other neurons.

The Above figure presents the structure of a neuron. It has **three** main parts to carry out its primary functionality of receiving and transmitting information:

A biological neuron has a cell body or soma to process the impulses, dendrites to receive them, and an axon that transfers them to other neurons.

1. **Dendrites :** to **receive signals** from neighboring or surrounding neurons and the **axon transmits the signal to the other neurons.**
2. **Soma** : Main body of the neuron **which accumulates** or **collects** the signals coming from **the different dendrites.** It '**fires**' when a **sufficient amount of signal is Collected.**
3. **Axon** :The last part of the neuron which **receives signal from soma, once the neuron 'fires' and passes it on to the neighboring neurons through the axon terminals** (to the adjacent dendrite of the neighboring neurons).

There is a very small **gap** between the axon **terminal of one neuron and the adjacent dendrite of the neighboring neuron.** This small gap is known as **synapse**. The signals transmitted through synapse may be excitatory or inhibitory.

2. With a neat diagram explain the structure of artificial neuron.

Gg

Artificial Neuron or **Perceptron :**

An artificial neuron, often referred to as a **perceptron**, is a **fundamental building block of artificial neural networks (ANNs). Modeled after biological neurons in the human brain**, artificial neurons are **mathematical entities** designed to process and transmit information.

The biological neural network has been modelled in the form of ANN with artificial neurons simulating the function of biological neurons.

Artificial Neural Network (ANN):

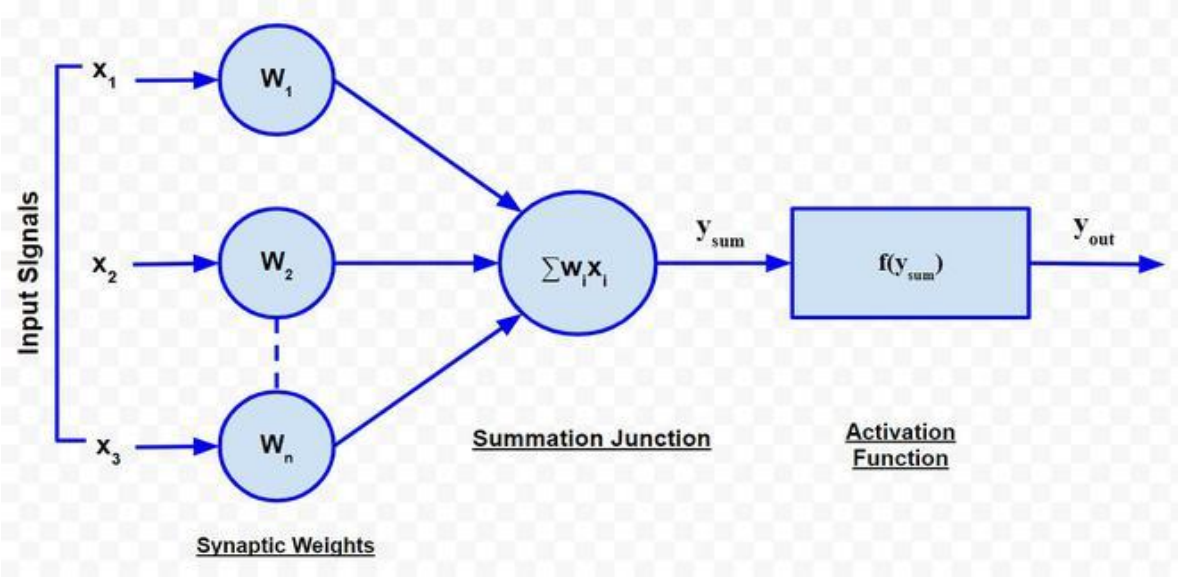
An artificial neural network (ANN), often simply referred to as a **neural network**, is a **computational model inspired by the structure and function of biological neural networks in the human brain.** It consists of **interconnected nodes**, called **neurons** or units, **organized in layers**. **ANN processes information in a manner similar to how the human brain operates**, enabling it to learn from data and make predictions or decisions.

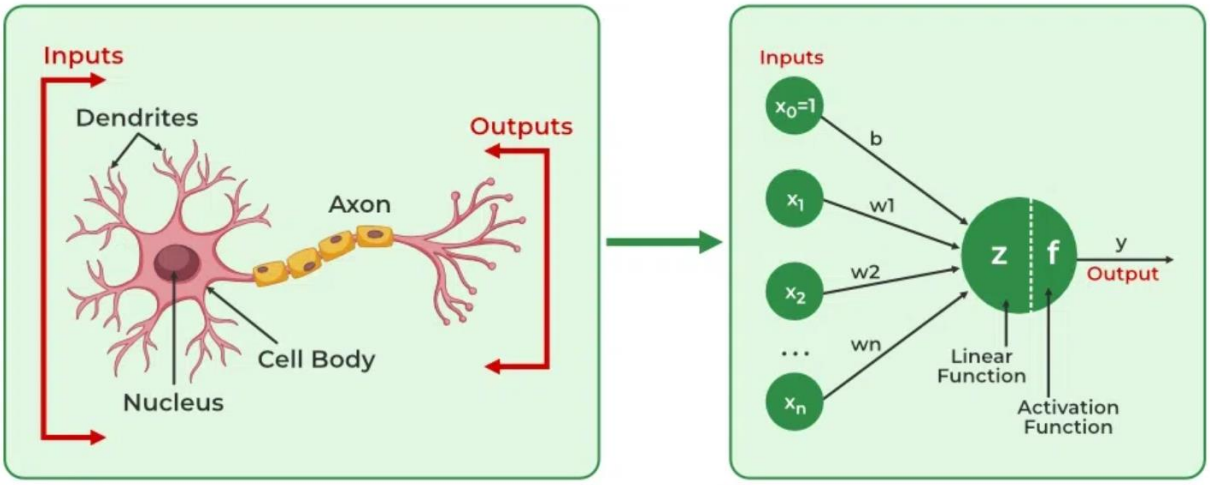
Key components and concepts of an ANN:

1. **Neurons (Nodes)**
2. **Connections (Edges)**
3. **Layers**

Structure of an Artificial Neuron: (Single Artificial Neuron)

An **artificial neuron**, also known as a **perceptron**, **is the building block of artificial neural networks** (ANNs). It **receives one or more input signals**, **processes them using weights** and **a transfer function**, and **produces an output signal.** The structure of an artificial neuron typically consists of the following component.





1. **Inputs:**

An artificial neuron **receives input signals** from **other neurons** or **external sources**. Each input is **associated with a weight** that represents the **strength or importance of that input signal** to the neuron.

1. **Weights:**

The weights **assigned to the inputs determine how much influence each input has on the neuron's output**. A higher weight **amplifies** the input signal's **contribution**, while a lower weight **diminishes** it. The weights are parameters of the neuron that are adj**usted during the training process** to **optimize** the **network's performance**.

3. Summation:

The **neuron computes a weighted sum of its inputs by multiplying each input signal by its corresponding weight and summing up the results.** Mathematically, this can be represented as the dot product of the input vector and weight vector, followed by adding a bias term

$$y_{\text{sum}} = \sum_{i=1}^{n} w_i x_i$$

4. Activation Function: (**Threshold Activation Function** or **Squashing function**)

The **weighted sum** is then **passed through an activation function**, which introduces **non-linearity** to the neuron's output.

Non-linearity refers to the property of activation functions that **enables neural networks to model and learn complex, non-linear relationships in the data**.

Without activation functions, neural networks would only be capable of representing linear relationships between inputs and outputs.

Activation functions **introduce non-linearity by applying a non-linear transformation to the weighted sum of inputs received by the neuron**. This non-linear transformation allows the neuron to **capture** and **represent complex patterns and relationships in the data.**

The complex problems cannot be solved by Linear equation and the Hidden patterns are cannot be expressed using linear equation. So, that's why we need non-linear equation and the Activation function will helps us to build the non-linear equation.

Output of the activation function, yout can be expressed as follows:

$$y_{\text{out}} = f(y_{\text{sum}})$$

5. Output:

The output of the activation function represents the **neuron's response to the input signals.** It can be interpreted as the **neuron's activation level or firing rate**.

This output is either **transmitted to other neurons as input** or **serves as the final output of the neural network.**

Note:

Artificial neurons are arranged in layers to form A**rtificial Neural Networks (ANN)**. Information flows through the network, with each neuron **receiving inputs from neurons in the previous layer, processing them, and passing the results to neurons in the next layer**. **By adjusting the weights and biases of the neurons, neural networks can learn to perform tasks such as classification, regression, and pattern recognition.**

3. What is deep learning? Explain the architecture DNN.

Deep Learning:

Deep learning is a **branch of machine learning** which is **completely based on artificial neural networks**, as neural network is **going to mimic the human brain** so **deep learning is also a kind of mimic of human brain**.

In deep learning, **we don't need to explicitly program everything**. The concept of deep learning is not new. It has been around for a couple of years now. It's on hype nowadays because earlier we did not have that much **processing power** and a **lot of data**. As in the last 20 years, the processing power increases exponentially, deep learning and machine learning came in the picture.

Deep learning deals with **algorithms inspired by the structure and function of the brain's neural networks**. It aims to mimic the way **humans learn** and **process information**, enabling **computers to learn from large amounts of data and make predictions or decisions without being explicitly programmed.**

Architectures in Deep Learning:

Deep learning **encompasses a wide range of neural network architectures,** each designed to **solve specific types of problems** and **address various challenges in machine learning**. These architectures differ in **their structure, connectivity**, and **functionality**, **allowing them to excel in different domains and tasks.**

Here are some popular Techniques in deep learning:

1. **Deep Neural Network :**
2. **Convolutional Neural Networks (CNNs)**:
3. **Recurrent Neural Networks (RNNs)**:
4. **Deep Belief Network(DBN)**

1. **Deep Neural Network:**

It is a neural network with a **certain level of complexity** having multiple hidden layers in between input and output layers. They are capable of modeling and processing **non-linear relationships**.

2. Convolutional Neural Networks (CNNs):

CNNs are widely used for **image recognition**, **computer vision**, and **other** tasks **involving grid-like data.**

- They consist of convolutional layers that apply filters (kernels) to input data, capturing spatial hierarchies of features.
- CNNs are known for their ability to automatically learn relevant features **from raw pixel data**, making them highly effective for tasks like **image classification**, **object detection**, and **image segmentation**.

3. Recurrent Neural Networks (RNNs):

- RNNs are designed to **process sequential data**, making them suitable for **tasks involving time-series data** or sequences of varying lengths.
- They have recurrent connections that allow **information to persist over time**, enabling them to capture temporal dependencies in data.

- RNNs are commonly used for tasks such as **natural language processing** (e.g., language modeling, machine translation), **speech recognition**, and **sequence generation**.
- RNN Allows for **parallel** and **sequential computation**. Similar to the human brain (large feedback network of connected neurons). They are **able to remember important things about the input they received and hence enables them to be more precise.**

4. Deep Belief Network(DBN):

A Deep Belief Network (DBN) is a type of artificial neural network that is **composed of multiple layers of latent variables, often referred to as hidden layers, which are arranged in a hierarchical manner**. DBNs are generative models that learn to **represent complex patterns in data through unsupervised learning**.

Deep Belief Networks are powerful models for representation learning, capable of capturing intricate patterns and relationships in data by learning hierarchical representations through unsupervised pre-training and fine-tuning. They have played a significant role in advancing the field of deep learning and continue to be an active area of research and development.

4. Explain the concept of multi-layer perceptron

Multi-layer Perceptron: (MLP)

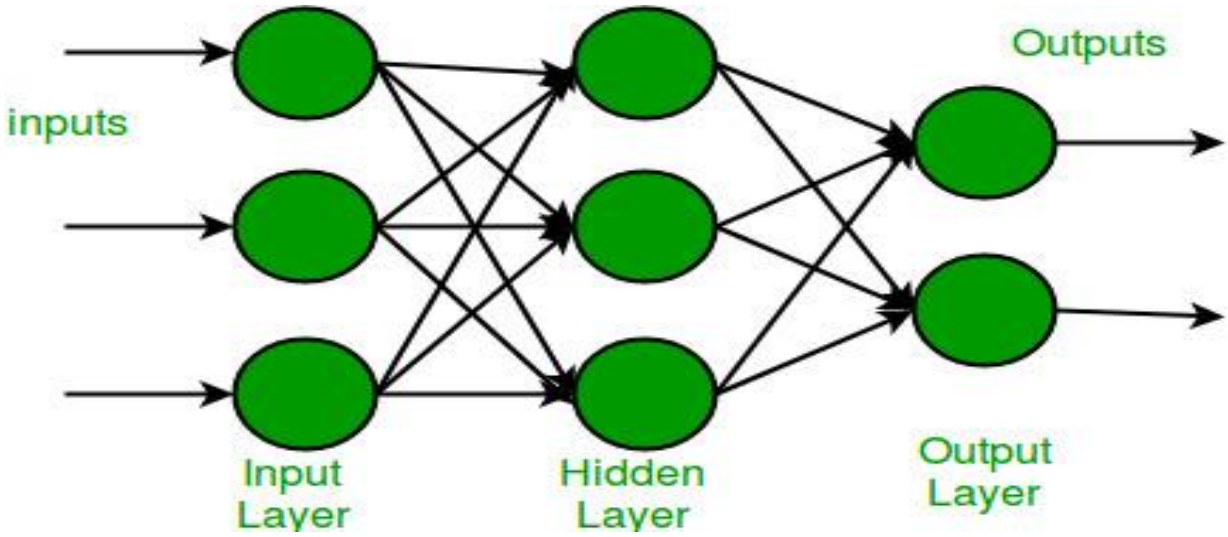
In the context of artificial neural networks (ANNs), a perceptron is a **computational unit that models a simplified version of a biological neuron's functionality.**

It takes **multiple input values**, **multiplies each input by a corresponding weight**, **sums up these weighted inputs**, applies an **activation function**, and **produces a single output**.

Perceptron's are typically **organized into layers within a neural network**, and **multiple perceptrons can be stacked together to form more complex architectures, such as multi-layer perceptron's (MLPs).**

A multi-layer perception is a neural network **that has multiple layers**. To create a neural network we combine neurons together so that the outputs of some neurons are inputs of other neurons.

A multi-layer perceptron has **one input layer** and **for each input**, **there is one neuron**(or node), it **has one output layer with a single node for each output** and it **can have any number of hidden layers** and **each hidden layer can have any number of nodes**. A schematic diagram of a Multi-Layer Perceptron (MLP) is depicted below.



In the multi-layer perceptron **diagram above**, we can see that **there are three inputs and thus three input nodes** and the **hidden layer has three nodes**. The output layer gives **two outputs**, therefore there are two output nodes. The nodes in the input layer take input and forward it for further process, in the diagram above the nodes in the input layer forwards their output to each of the three nodes in the hidden layer, and in the same way, the **hidden layer processes the information and passes it to the output layer**.

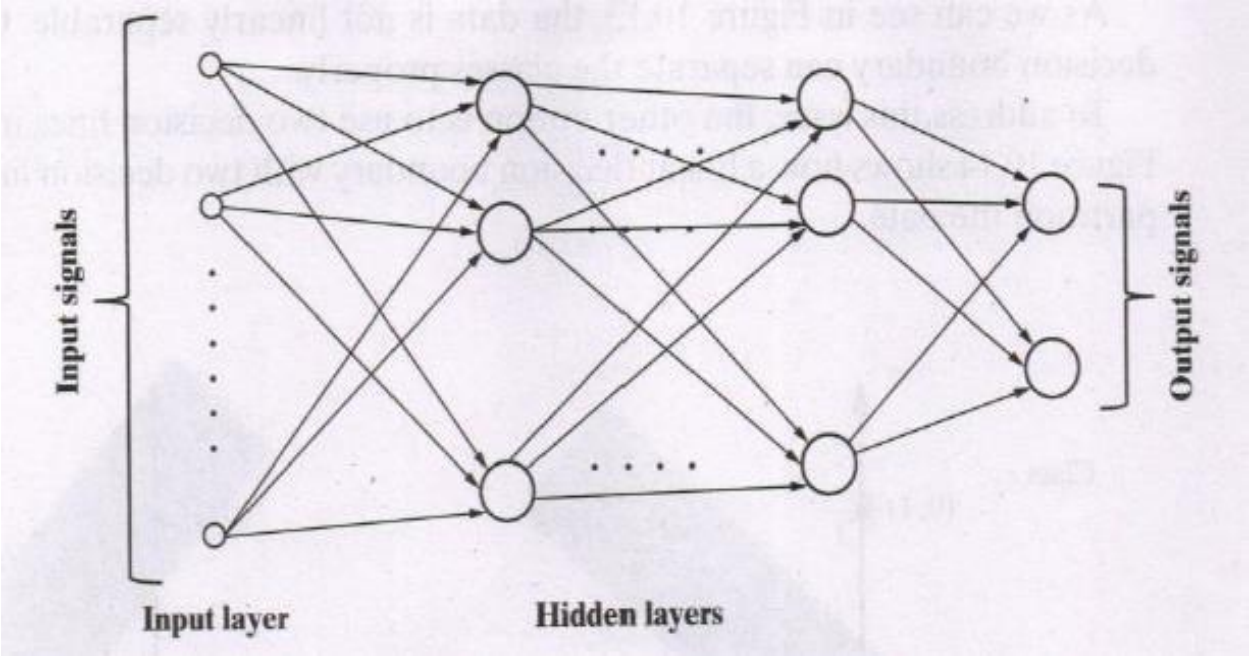
Every node in the multi-layer perception uses a **sigmoid activation function**. The sigmoid activation function **takes real values as input** and **converts them to numbers between 0 and 1** using the sigmoid formula.

A basic perceptron works very successfully for data sets which **possess linearly separable patterns**. **This is the philosophy used to design the multi-layer perceptron model**.

The major highlights of this model are as follows:

1. The neural network contains **one or more intermediate layers** between the input and the output nodes, which are **hidden from both input and output nodes.**
2. Each neuron in the network includes a **non-linear activation function** that is differentiable.
3. The **neurons in each layer are connected** with **some or all the neurons in the previous layer**.

The diagram in the figure below resembles a fully connected multi-layer perceptron with multiple hidden layers between the input and output layers. It is called **fully connected** because any neuron in any layer of the perceptron is connected with all neurons (or input nodes in the case of the first hidden layer) in the previous layer. The signals flow from one layer to another layer **from left to right**.



5. **Explain the sigmoid function in neural networks**

Sigmoid Function or Logistic Function:

The sigmoid function is a common activation function used in artificial neural networks. It is a **smooth, S-shaped function** that **squashes** or **compresses** the input values into the range between **0 and 1**.

The sigmoid function is particularly useful for **binary classification tasks**, where the goal is to produce a probability score indicating the likelihood of an input belonging to one of two classes.

There are two types of sigmoid function:

1. Binary sigmoid function
2. Bipolar sigmoid function

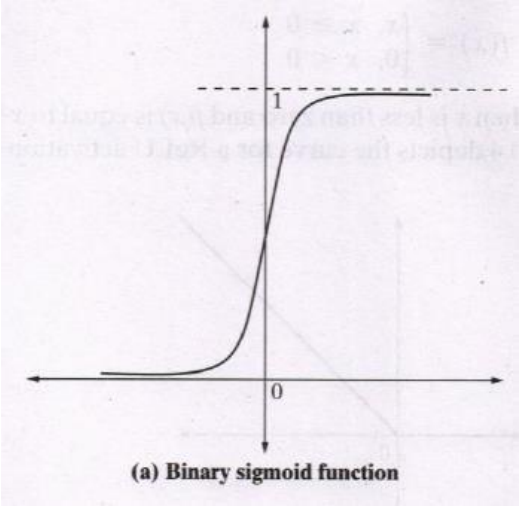
1. Binary sigmoid function:

The binary sigmoid function is a type of sigmoid function that **produces binary output value**s, **typically 0 or 1.**

Mathematically, the binary sigmoid function can be defined as:

$$y_{out} = f(x) = \frac{1}{1 + e^{-kx}}$$

- In this definition, *x* represents the input to the function, and the threshold is a predetermined value that separates the two classes.
- where k = steepness or slope parameter of the sigmoid function. By varying the value of k, sigmoid functions with different slopes can be obtained. It has range of (0,1).



(a) Binary sigmoid function

- The binary sigmoid function is commonly used in binary classification tasks, where the goal is to classify inputs into one of two categories.
- The slope at the origin is k/4. As the value of k becomes very large, the sigmoid function becomes a threshold function

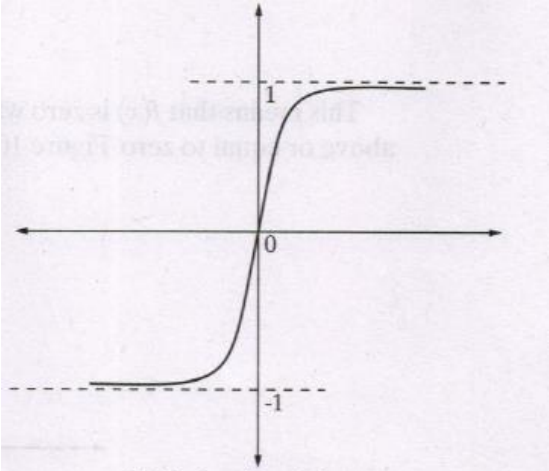
2. Bipolar sigmoid function:

The bipolar sigmoid function is another type of sigmoid function that **produces bipolar (or signed) output values, typically -1 or 1.**

- Mathematically, the bipolar sigmoid function can be defined as:

$$y_{out} = f(x) = \frac{1 - e^{-kx}}{1 + e^{-kx}}$$

- Like the binary sigmoid function, the bipolar sigmoid function produces output values bounded between -1 and 1.
- The bipolar sigmoid function is useful in contexts where inputs and outputs are **naturally signed**, such as in certain types of neural networks or signal processing applications.
- It can also be advantageous in scenarios where the mean of the inputs is close to zero, as it allows the **network to capture both positive and negative information.**



(b) Bipolar sigmoid function

6. Discuss the various types of activation functions in neural networks.

Activation Function:

- The Activation Function is **applied** over the **net input** i.e, **ysum to calculate the output of an ANN.**
- The activation function is a **mathematical "Gate"** in **between the input feeding in the current node and it's output to the next layer.**

TYPES OF ACTIVATION FUNCTIONS:

There are different types of activation functions. The most commonly used activation functions are highlighted below:

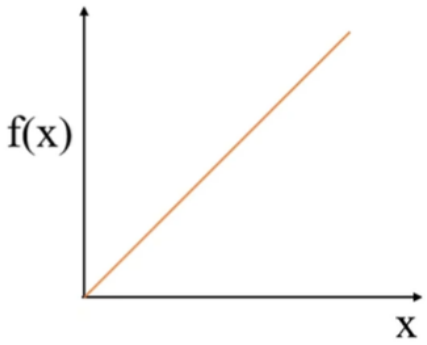
1. **Linear Activation Function:**
    1. Identity Function
2. **Non-Linear Activation Function**
    1. Step Function
    2. Threshold Function
    3. ReLU
    4. Sigmoid Function or Logistic Function

1. Identity Function:

The identity function, also known as the "**linear activation function**," is a simple mathematical function commonly used as an activation function for the input layer of neural networks.

Unlike other activation functions that **introduce non-linearity to the model**, the identity function **preserves the original input values**, resulting in a linear relationship between the input and output.

yout = f(x) = **x**, for all x



2. Step function:

The threshold function, also known as the **step function** or **Heaviside step function**, is a simple mathematical function commonly used in artificial neural networks as an activation function.

It is **a binary function** that outputs one of two possible values based on whether the input is greater than or equal to a specified threshold.

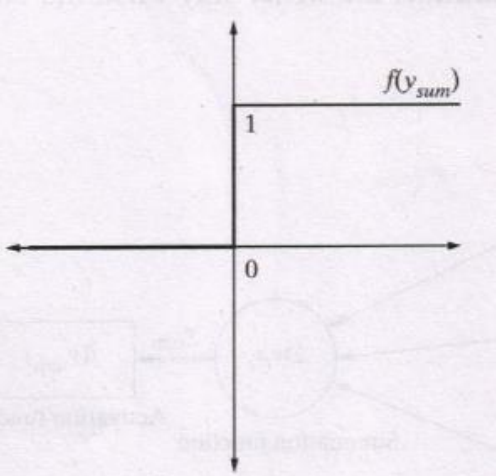
Mathematical Form:

$$y_{out} = f(y_{sum}) = \begin{cases} 1, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

Where, x represent the **weighted sum of inputs** to the neuron

In other words, if the input *x* is greater than or equal to the specified threshold, the output of the function is 1; otherwise, it is 0.

For example if the value ysum is greater thant or equal to zero then the value of f(ysum) will be 1 else 0.



(a) Step function

3. Threshold Function:

The threshold function is **almost like the step function,** with the only difference being the fact that **θ is used as a threshold value instead of 0**.

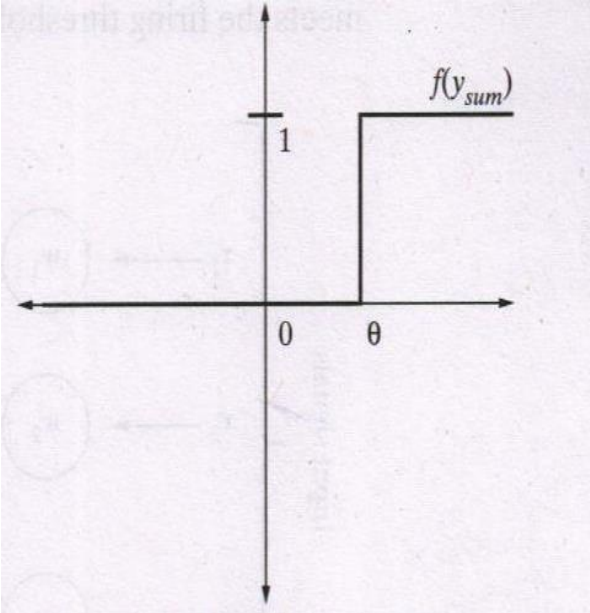
It can mathematically be expressed as follows:

$$y_{out} = f(y_{sum}) = \begin{cases} 1, & x \geq \theta \\ 0, & x < \theta \end{cases}$$

Note:

θ represents a **threshold value** that determines the point at which the function transitions from one state to another.

Here, θ acts as the boundary or threshold. If the input x is greater than or equal to θ, the function outputs 1. If x is less than θ, the function outputs 0.



4. ReLU (Rectified Linear Unit) function:

The **Rectified Linear Unit (ReLU)** function is a popular activation function used in artificial neural networks, particularly in deep learning models.

It **introduces non-linearity to the network by outputting the input directly if it is positive, and zero otherwise.** Mathematically, the ReLU function can be defined as:

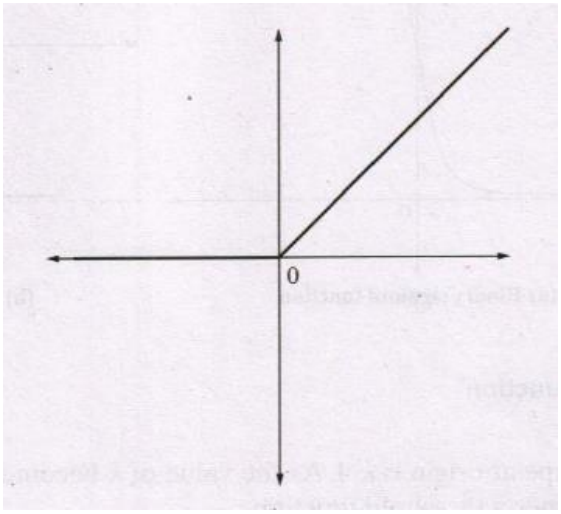
$f(x)=\max(0,x)$

$$f(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases}$$

Where,

- For any input ***x***, if ***x*** is greater than zero, the **function outputs *x***.
- If *x* is less than zero, the function outputs zero.

Graphically, the ReLU function appears as a linear function with a positive slope for positive input values, and it flattens out to zero for negative input values.



5. Sigmoid Function or Logistic Function:

The sigmoid function is a common activation function used in artificial neural networks. It is a **smooth, S-shaped function** that **squashes** or **compresses** the input values into the range between **0 and 1**.

The sigmoid function is particularly useful for **binary classification tasks**, where the goal is to produce a probability score indicating the likelihood of an input belonging to one of two classes.

There are two types of sigmoid function:

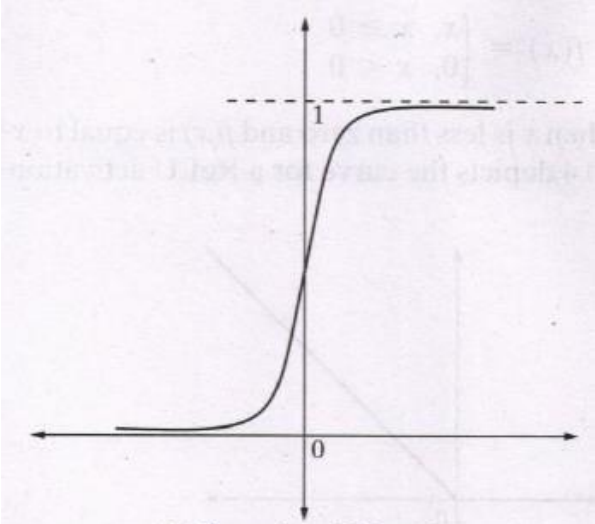
1. Binary sigmoid function
2. Bipolar sigmoid function

1. Binary sigmoid function:

The binary sigmoid function is a type of sigmoid function that **produces binary output value**s, **typically 0 or 1.**

Mathematically, the binary sigmoid function can be defined as:

$$y_{out} = f(x) = \frac{1}{1 + e^{-kx}}$$



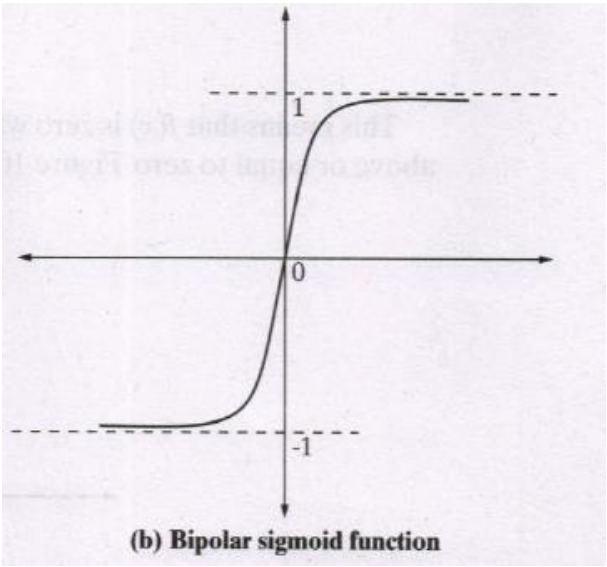
(a) Binary sigmoid function

2. Bipolar sigmoid function:

The bipolar sigmoid function is another type of sigmoid function that **produces bipolar (or signed) output values, typically -1 or 1.**

- Mathematically, the bipolar sigmoid function can be defined as:

$$y_{out} = f(x) = \frac{1 - e^{-kx}}{1 + e^{-kx}}$$



(b) Bipolar sigmoid function

7. Explain the concept of back propagation in neural network.

Backpropagation :

Backpropagation is a **fundamental algorithm used for training** artificial neural networks, including **Multi-layer Perceptron's** and **other deep learning models**.

In 1986, an **efficient method of training an ANN was discovered**. In this method, errors, i.e. **difference in output values of the output layer and the expected values,** are **propagated back** from the **output layer to the preceding layers**. Hence, the algorithm implementing this method is **known as backpropagation**, i.e. **propagating the errors backward to the preceding layers.**

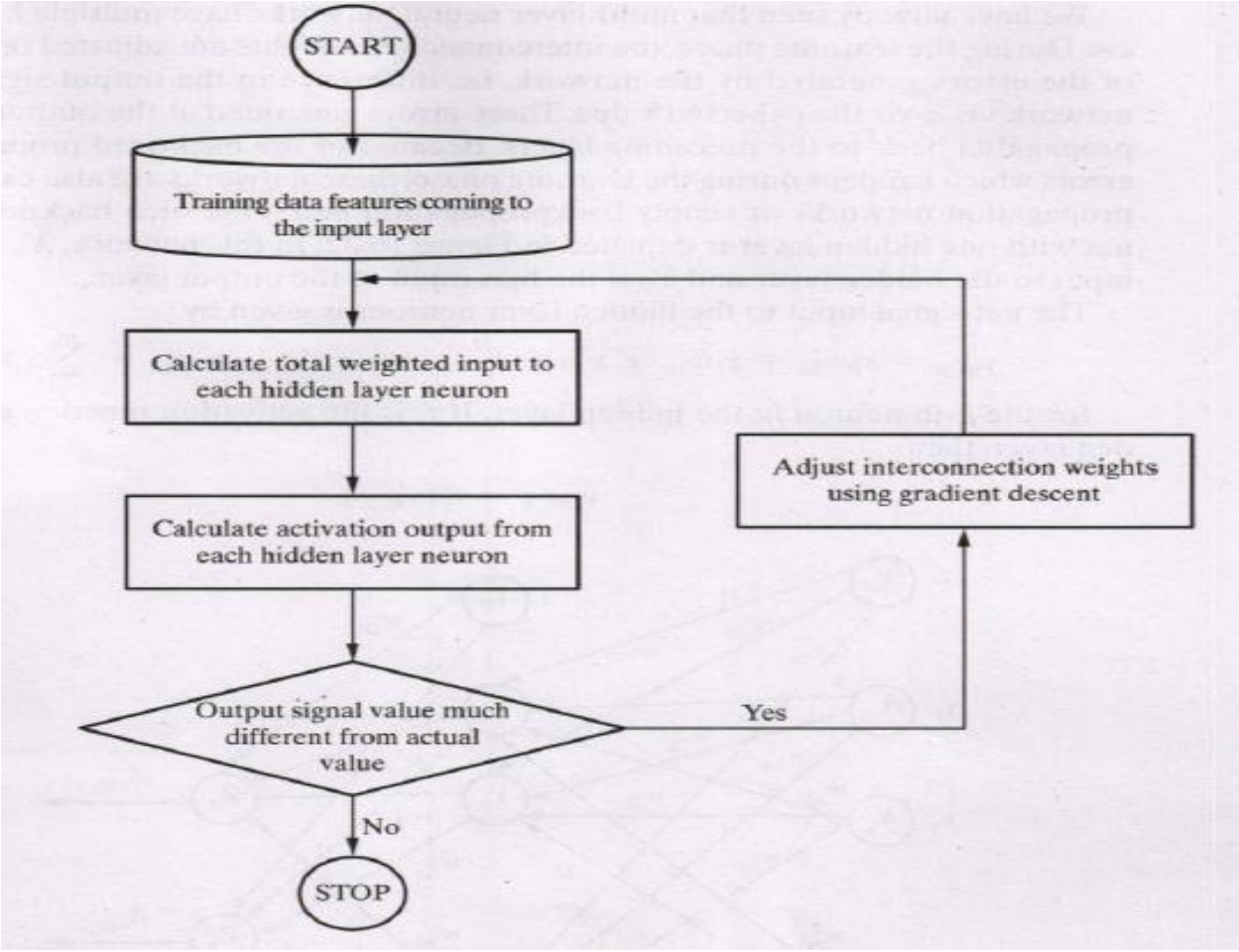
The backpropagation algorithm is **applicable for multi-layer feed forward networks**. It is a supervised learning algorithm which **continues adjusting the weights of the connected neurons with an objective to reduce the deviation of the output signal from the target output.**

This algorithm consists of multiple iterations, also known as **epochs**.

Each epoch consists of two phases —

1. **A forward phase** in which the **signals flow from the neurons in the input layer to the neurons in the output layer through the hidden layers**. The weights of the interconnections and activation functions are used during the flow. In the output layer, the **output signals are generated**.
2. **A backward phase in which the output signal is compared with the expected value**. **The computed errors are propagated backwards from the output to the preceding layers.** The errors propagated back are **used to adjust the interconnection weights between the layers.**

The iterations continue till a **stopping criterion is reached**. The figure below depicts a reasonably simplified version of the backpropagation algorithm.



Here's an overview of how backpropagation works:

1. **Forward Pass**:

During the forward pass, **input data is passed through the network, layer by layer**, to produce a predicted output. Each layer applies a set of linear transformations (matrix multiplications) and non-linear activation functions to the input data.

1. **Loss Calculation**:

After the forward pass, the **difference between the predicted outputs and the actual targets** (the ground truth) is **computed using a loss function**. Common loss functions include **mean squared error (MSE)** for regression tasks and **cross-entropy loss** for classification tasks.

1. **Backward Pass (Backpropagation)**:

In the backward pass, **gradients (**derivatives of the loss function**)of the loss function with respect to the network's parameters such as** weights and biases are computed **recursively** using the chain rule of calculus. Gradients are computed layer by layer, **starting from the output layer and moving backward through the network.**

1. **Gradient Descent**:

Once the gradients are computed, the **network's parameters are updated in the opposite direction of the gradient** (i.e., descending along the gradient) **to minimize the loss function**. **This process is known as gradient descent.** The magnitude of the parameter updates is controlled by a learning rate hyperparameter.

5. **Iterative Training**:

The **forward pass**, **loss calculation**, **backward pass**, and **parameter updates** are repeated iteratively for **multiple epochs** (passes through the entire training dataset). During training, the network's parameters gradually adjust to minimize the error between predicted outputs and actual targets, improving the network's performance on the task.

Backpropagation **enables neural networks to learn complex patterns** and **relationships in data by iteratively adjusting their parameters based on the error feedback from the training data**.

It is a **key algorithm in the field of deep learning** and has enabled the development of powerful models for a wide range of tasks, including image classification, natural language processing, and reinforcement learning.

10 Mark Questions:

1. Briefly explain the Artificial Neural Network? Discuss various Architectures of neural network with diagrams?

Artificial Neural Network:

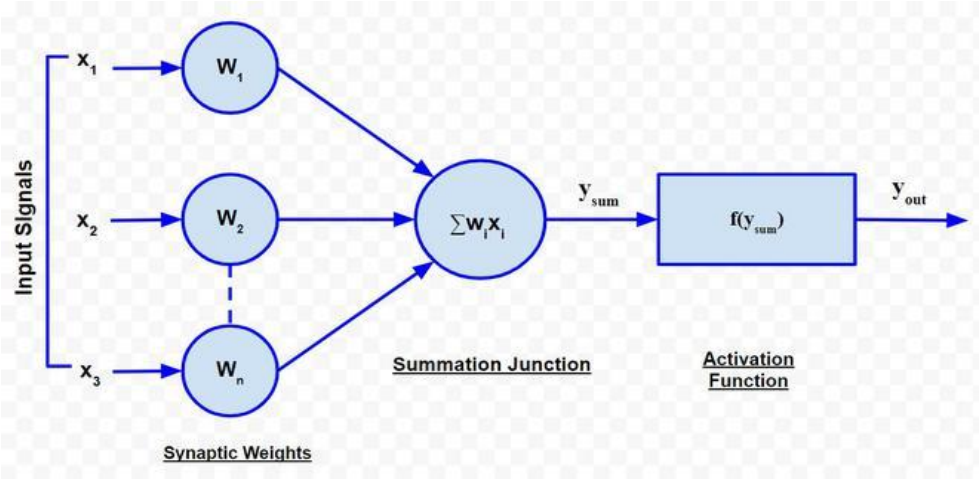
An artificial neural network (ANN), often simply referred to as a **neural network**, is a **computational model inspired by the structure and function of biological**

neural networks in the human brain. It consists of **interconnected nodes**, called **neurons** or units, **organized in layers**. **ANN processes information in a manner similar to how the human brain operates**, enabling it to learn from data and make predictions or decisions.

Key components and concepts of an ANN:

1. **Neurons (Nodes)**
2. **Connections (Edges)**
3. **Layers**

Structure of an Artificial Neuron:



Neural networks consist of **interconnected layers of neurons** that **process input data to produce desired outputs.**

The architecture of a neural network **refers to its structure, including the number of layers, the number of neurons in each layer, and the connections between neurons.** There are several common architectures of neural networks, each designed for specific tasks and applications.

The various Architectures of neural networks are as follows:

1. Single Layer Feed Forward Network
2. Multi-Layered Feed forward Network
3. Competitive Network
4. Recurrent Network

1. Single Layer Feed Forward Network:

Single-layer feed forward, also known as a **Single Layer Perceptron (SLP**), is the simplest and most basic architecture of ANNs. It **consists of only two layers** as

depicted in Figure — the **input layer and the output layer** with **no hidden layers in between**.

The input layer consists of a set of 'm' input neurons **X1, X2, ..., Xm** connected to each of the 'n' output neurons **Y1,Y2, ..., Yn.** The connections carry weights **w11,w12,w13,** …, wmn.

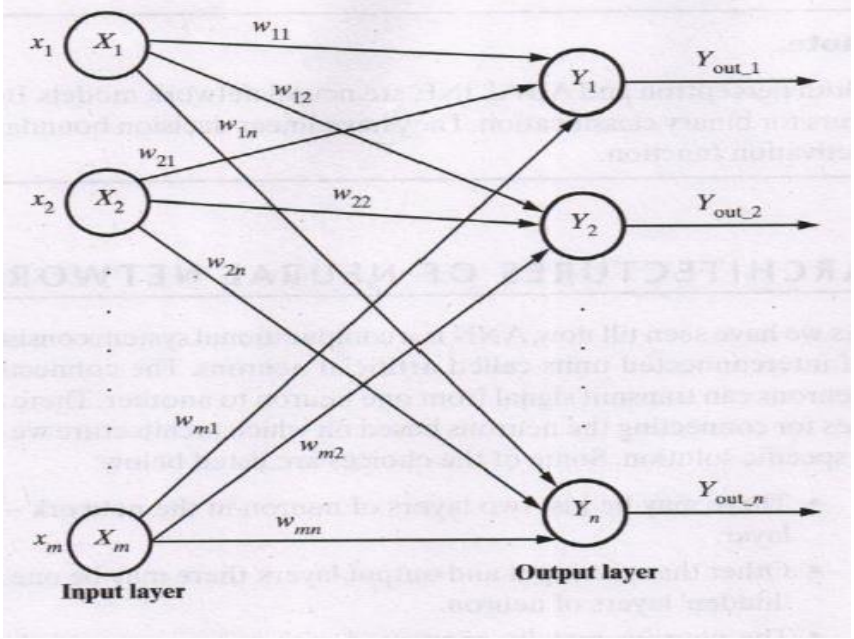
The input layer of neurons does not conduct any processing — they pass the input signals to the output neurons. The **computations are performed only by** the neurons in the **output layer**. So, though it has two layers of neurons, **only one layer is performing the computation. This is the reason why the network is known as single layer in spite of having two layers of neurons.**

Also, the **signals always flow from the input layer to the output layer.** Hence, this network is known as **feed forward**.

The net signal input (*uj*) to the output neuron *j* in a Single-layer Feedforward Network (SLFN) can be **mathematically** expressed as follows:

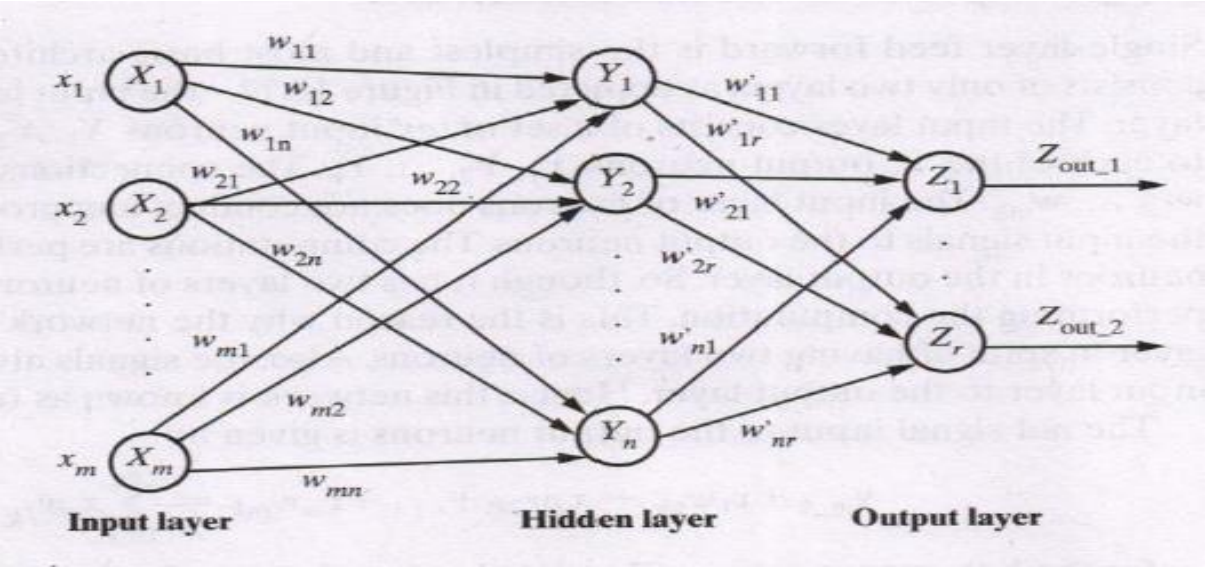
$$u_j = \sum_{i=1}^{m} w_{ij} \cdot x_i$$

$$y_{in\_k} + x_1 w_{1k} + x_2 w_{2k} + \ldots + x_m w_{mk} = \sum_{i=1}^{m} x_i w_{ik},$$



2.Multi-Layered Feedforward Network:

A Multi-Layered Feedforward Network, also known as a **Multi-Layer Perceptron (MLP**), is a type of artificial neural network **with multiple layers of neurons**, including **input**, **hidden**, and **output layers**. Unlike Single-Layer Feedforward Networks, MLPs **can learn complex non-linear relationships between input and output data.**



Each of the layers may have varying number of neurons. For example, the one shown above has **'m' neurons** in the **input layer** and '**r' neurons** in the **output layer**, and there is **only one hidden layer with 'n' neurons.** The net signal input to the neuron in the hidden layer is given by

$$y_{in\_k} = x_1 w_{1k} + x_2 w_{2k} + \ldots + x_m w_{mk} = \sum_{i=1}^{m} x_i w_{ik},$$

for the k-th output neuron. The net signal input to the neuron in the output layer is given by

$$z_{in\_k} = y_{out\_1} w'_{1k} + y_{out\_2} w'_{2k} + \ldots + y_{out\_n} w'_{nk} = \sum_{i=1}^{n} y_{out\_i} w'_{ik},$$

for the k-th output neuron.

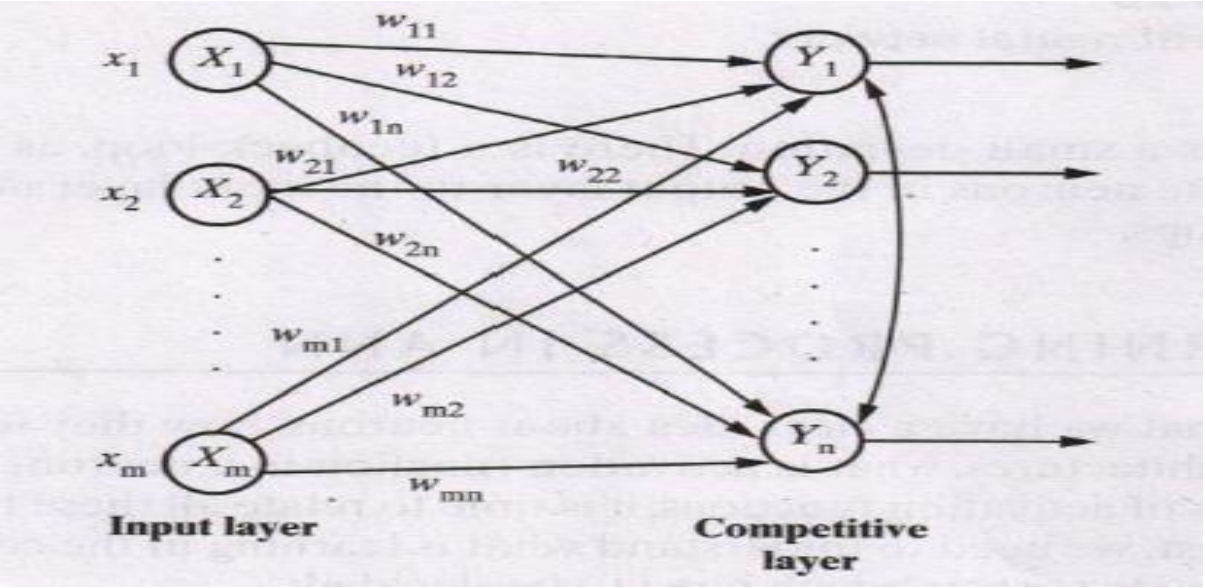
3. Competitive Network:

A Competitive Network, also known as a **Self-Organizing Map (SOM)** or **Kohonen Network**. The competitive network is **almost the same in structure as the single-layer feed forward network**.

The only difference is that the output neurons are connected with each other (either partially or fully). The figure depicts a fully connected competitive network**. In competitive networks, for a given input, the output neurons compete amongst**

themselves to represent the input. It represents a form of unsupervised learning algorithm in ANN that is **suitable to find clusters in a data set**.

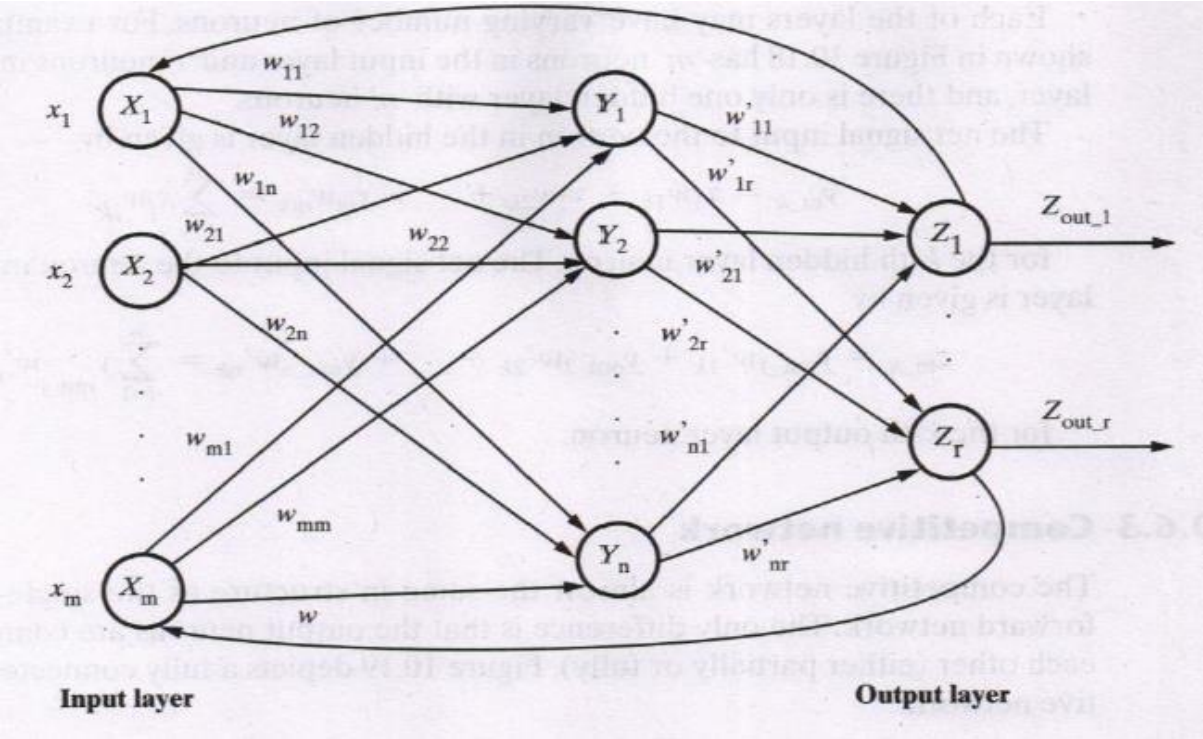
Competitive Network is a type of neural network used for unsupervised learning and dimensionality reduction. It is commonly used for clustering and visualization of high-dimensional data.



4. Recurrent Network:

We have seen that in feed forward networks, **signals always flow from the input layer towards the output layer** (through the hidden layers in the case of multi-layer feed forward networks), i.e. in **one direction**.

In the case of recurrent neural networks, **there is a small deviation**. **There is a feedback loop**, as depicted in Figure from the neurons in the **output layer to the input layer neurons**. There may also be self-loops.



2. Write a note on Artificial Neuron and Biological Neuron along with their differences? Briefly explain the Learning process in Artificial Neural Network?

Machine learning, as we have seen, **mimics the human form of learning**. On the other hand, human learning, or for that matter every action of a human being, is **controlled by the nervous system**.

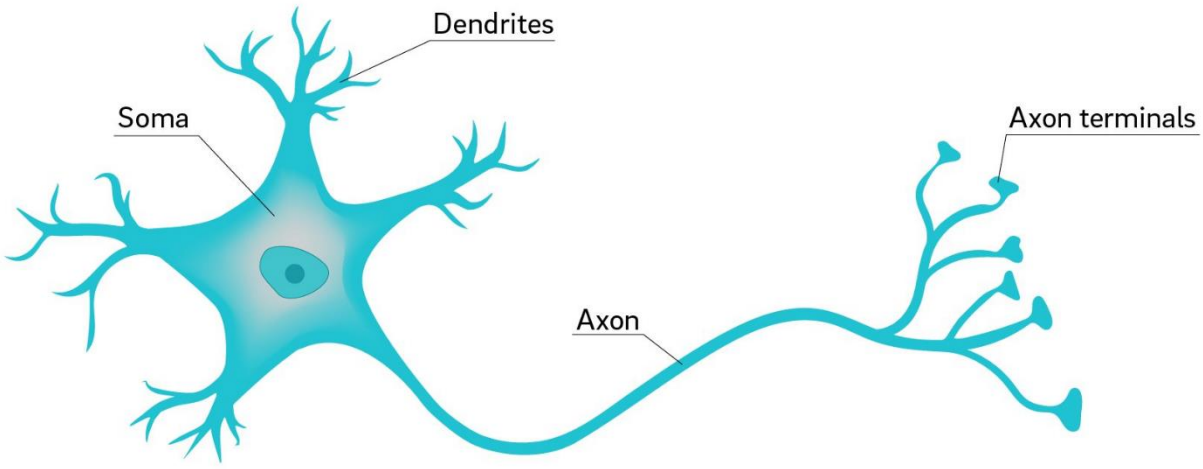
In **any human being**, the **nervous system coordinates the different actions by transmitting signals** to and from different parts of the body.

The nervous system is constituted of a special type of cell, called **neuron** or **nerve cell**, which has special structures allowing it to **receive** or **send signals** to other neurons. Neurons connect with each other to transmit signals to or receive signals from other neurons. This structure essentially forms **a network of neurons or a Neural Network.**

THE BIOLOGICAL NEURON :

The human nervous system is divided into two main parts:

1. The **Central Nervous System (CNS),** which includes the **brain** and **spinal cord**.
2. The **Peripheral Nervous System**, which includes nerves and ganglia (clusters of nerve cells) outside the brain and spinal cord.



Note:

1. The **CNS integrates all information**, in the form of **signals**, from the different parts of the body.
2. The **peripheral nervous system**, on the other hand, **connects the CNS with the limbs and organs.**
3. **Neurons** are basic structural units of the CNS.
4. A neuron is able to **receive, process**, and **transmit** information in the form of **chemical** and **electrical** signals.

A **biological neuron** has a cell body or soma to **process the impulses or signals**, dendrites to receive them, and an axon that transfers them to other neurons.

The Above figure presents the structure of a neuron. It has **three** main parts to carry out its primary functionality of receiving and transmitting information:

A biological neuron has a cell body or soma to process the impulses, dendrites to receive them, and an axon that transfers them to other neurons.

1. **Dendrites :** to **receive signals** from neighboring or surrounding neurons and the **axon transmits the signal to the other neurons.**
2. **Soma** : Main body of the neuron **which accumulates** or **collects** the signals coming from **the different dendrites.** It '**fires**' when a **sufficient amount of signal is Collected.**
3. **Axon** :The last part of the neuron which **receives signal from soma, once the neuron 'fires' and passes it on to the neighboring neurons through the axon terminals** (to the adjacent dendrite of the neighboring neurons).

There is a very small **gap** between the axon **terminal of one neuron and the adjacent dendrite of the neighboring neuron.** This small gap is known as **synapse**. The signals transmitted through synapse may be excitatory or inhibitory.

Artificial Neuron or **Perceptron :**

An artificial neuron, often referred to as a **perceptron**, is a **fundamental building block of artificial neural networks (ANNs). Modeled after biological neurons in the human brain**, artificial neurons are **mathematical entities** designed to process and transmit information.

The biological neural network has been modelled in the form of ANN with artificial neurons simulating the function of biological neurons.

Artificial Neural Network (ANN):

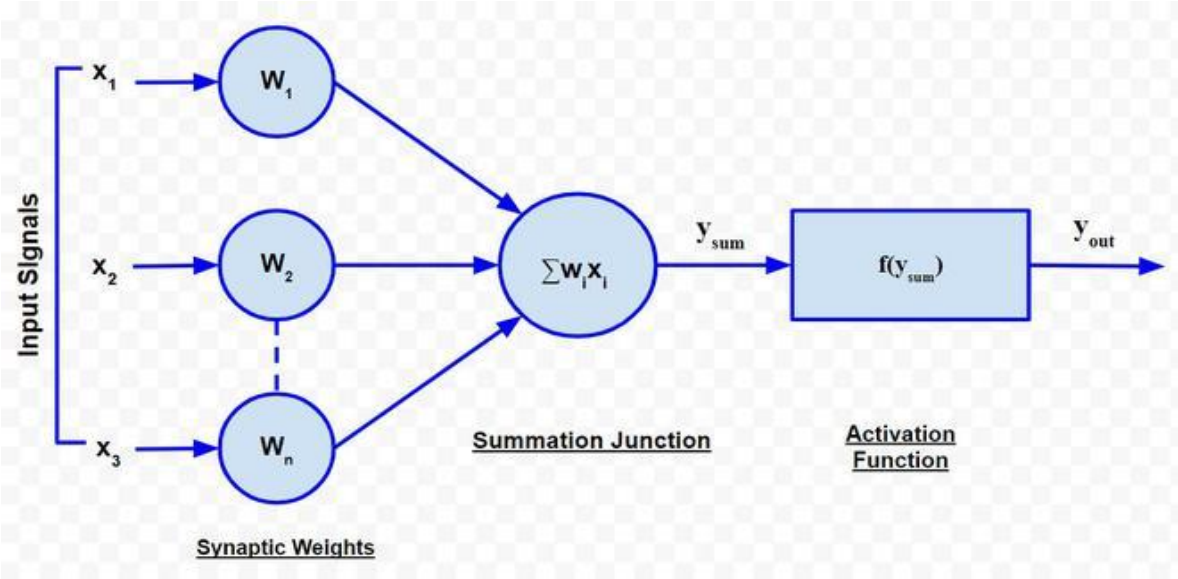
An artificial neural network (ANN), often simply referred to as a **neural network**, is a **computational model inspired by the structure and function of biological neural networks in the human brain.** It consists of **interconnected nodes**, called **neurons** or units, **organized in layers**. **ANN processes information in a manner similar to how the human brain operates**, enabling it to learn from data and make predictions or decisions.

Key components and concepts of an ANN:

4. **Neurons (Nodes)**
5. **Connections (Edges)**
6. **Layers**

Structure of an Artificial Neuron: (Single Artificial Neuron)

An **artificial neuron**, also known as a **perceptron**, **is the building block of artificial neural networks** (ANNs). It **receives one or more input signals**, **processes them using weights** and **a transfer function**, and **produces an output signal.** The structure of an artificial neuron typically consists of the following component.



1. **Inputs:**

An artificial neuron **receives input signals** from **other neurons** or **external sources**. Each input is **associated with a weight** that represents the **strength or importance of that input signal** to the neuron.

2. **Weights:**

The weights **assigned to the inputs determine how much influence each input has on the neuron's output**. A higher weight **amplifies** the input signal's **contribution**, while a lower weight **diminishes** it. The weights are parameters of the neuron that are adj**usted during the training process** to **optimize** the **network's performance**.

3. **Summation:**

The **neuron computes a weighted sum of its inputs by multiplying each input signal by its corresponding weight and summing up the results.** Mathematically, this can be represented as the dot product of the input vector and weight vector, followed by adding a bias term

$$y_{\text{sum}} = \sum_{i=1}^{n} w_i x_i$$

4. Activation Function: (**Threshold Activation Function** or **Squashing function**)

The **weighted sum** is then **passed through an activation function**, which introduces **non-linearity** to the neuron's output.

Output of the activation function, yout can be expressed as follows:

$$y_{\text{out}} = f(y_{\text{sum}})$$

5. Output:

The output of the activation function represents the **neuron's response to the input signals.** It can be interpreted as the **neuron's activation level or firing rate**.

This output is either **transmitted to other neurons as input** or **serves as the final output of the neural network.**

Learning process in ANN:

We need to understand what is learning in the context of ANNs. There are **four major aspects** which need to be decided:

1. The **number of layers** in the network
2. The **direction of signal flow**
3. The **number of nodes in each layer**

4. The **value of weights** attached with each interconnection between neurons

1. **Number of Layers:**

As we have seen, a neural network may have a **single layer or multi-layer**. In the case of a single layer, a set of neurons in the input layer receives signal, i.e. a single feature per neuron, from the data set. The value of the feature is transformed by the activation function of the input neuron. The **signals processed by the neurons in the input layer** are then **forwarded to the neurons in the output layer**. The neurons in the **output layer use their own activation function to generate the final prediction.**

More complex networks may be designed with multiple hidden layers between the input layer and the output layer. Most of the multi-layer networks are **fully connected**.

2. **Direction of signal flow:**

In certain networks, termed **as feed forward networks**, **signal is always fed in one direction,** i.e. **from the input layer towards the output layer through the hidden layers**, if there is any. However, certain networks, such as the **recurrent network**, also allow signals to **travel from the output layer to the input layer**. This is also an important consideration for **choosing the correct learning model**.

3. **Number of nodes in layers:**

In the case of a multi-layer network, the **number of nodes in each layer can** be **varied**. However, the number of nodes or neurons in the input layer is **equal to the number of features of the input data set**. Similarly, the **number of output nodes** will depend on **possible outcomes,** e.g. number of classes in the case of supervised learning. So, the **number of nodes in each of the hidden layers** is to be **chosen by the user**. **A larger number of nodes in the hidden layer help in improving the performance.** However, too many nodes may result in overfitting as well as an increased computational expense.

4. Weight of interconnection between neurons:

Deciding the value of weights attached with each interconnection between neurons so that a specific learning problem can be solved correctly is quite a difficult problem by itself.

For solving a learning problem using ANN, we can start with a set of values for the synaptic weights and **keep doing changes** to those values in **multiple iterations**. In the case of supervised learning, the objective to be pursued is to reduce the number of misclassifications. Ideally, the iterations for making changes in weight values should be continued **till there is no misclassification.**

However, in practice, such a stopping criterion may not be possible to achieve. Practical stopping criteria maybe the rate of misclassification less than a specific threshold value, say 1%, or the maximum number. This may become a bigger problem when the number of interconnections and hence the number of weights keeps increasing.