**1. Description ANN, or Artificial Neural Network**

An ANN, or Artificial Neural Network, is a type of machine learning model that is inspired by the structure and function of the human brain. ANNs are composed of interconnected units called artificial neurons or simply "neurons," which can process and transmit information.

Each neuron receives input from other neurons or external sources and produces an output based on that input. The output of a neuron is determined by a set of weights that represent the strength of the connections between neurons, and a bias term that represents the neuron's threshold for activation.

ANNs can be used for a wide range of tasks, including classification, regression, and prediction. They are particularly useful for tasks that require the processing of large amounts of data or the identification of patterns and relationships within that data.

There are several different types of ANNs, including feedforward networks, convolutional neural networks, and recurrent neural networks. Each type is designed for a specific purpose and has its own unique characteristics and capabilities.

**2. Artificial neural networks (ANNs) are well-suited for a wide range of tasks, including:**

Classification: ANNs can be used to classify data into different categories or classes based on certain features or characteristics. For example, an ANN could be used to classify images of animals as cats, dogs, or birds based on their visual characteristics.

Regression: ANNs can be used to predict a continuous value based on a set of input features. For example, an ANN could be used to predict the price of a house based on its size, location, and other characteristics.

Prediction: ANNs can be used to make predictions about future events or outcomes based on past data. For example, an ANN could be used to predict the stock price of a company based on its historical performance.

Pattern recognition: ANNs are good at identifying patterns and relationships within large amounts of data. For example, an ANN could be used to identify patterns in customer behavior that might indicate a likelihood to purchase a particular product.

Decision making: ANNs can be used to make decisions based on input data. For example, an ANN could be used to determine whether to approve or deny a loan application based on the applicant's credit history and other factors.

Natural language processing: ANNs can be used to process and understand natural language text, making it possible to build language translation systems and chatbots, for example.

These are just a few examples of the many types of tasks that ANNs can be used for. Overall, ANNs are a flexible and powerful tool that can be applied to a wide range of problems in many different fields.

**3. What is Backpropagation**

Backpropagation is an algorithm used to train artificial neural networks, particularly multi-layer perceptrons. It is a supervised learning algorithm, which means that it requires a labeled training dataset in order to learn how to perform a specific task.

The basic idea behind backpropagation is to adjust the weights and biases of the neurons in the network in order to minimize the error between the predicted output of the network and the true desired output. This is done by propagating the error back through the network, starting at the output layer and working backwards through the hidden layers until the input layer is reached.

**4. The steps involved in the backpropagation algorithm.**

-Initialize the weights and biases of the neurons in the network randomly.

-Feed the input data through the network and compute the predicted output.

-Calculate the error between the predicted output and the true desired output.

Propagate the error back through the network, starting at the output layer and working backwards through the hidden layers.

-For each layer, calculate the gradient of the error with respect to the weights and biases of the neurons.

-Update the weights and biases of the neurons in the network using the gradient and a learning rate.

-Repeat the process until the error is minimized to an acceptable level or the maximum number of iterations is reached.

-Use the trained network to make predictions on new input data.

It's important to note that the specific implementation details of the backpropagation algorithm can vary depending on the specific task and network architecture being used. However, the general process of propagating the error back through the network and adjusting the weights and biases based on the gradient of the error is a key component of the algorithm.

**5. what is activation function**

An activation function is a mathematical function used in artificial neural networks to determine the output of a neuron given an input or set of inputs. Activation functions are an important part of the backpropagation algorithm, as they allow the network to learn and adapt to new data.

There are several different types of activation functions that can be used in neural networks, including sigmoid, tanh, ReLU, and Leaky ReLU. Each type of activation function has its own characteristics and properties that make it more or less suitable for different types of tasks.

In the backpropagation algorithm, the activation function is used to determine the output of a neuron based on its input and weights. When the input data is fed through the network, it is multiplied by the weights of the connections between neurons and passed through the activation function to produce the output of the neuron.

The activation function plays a crucial role in the learning process of the network, as it determines how the output of a neuron is influenced by its inputs and weights. By adjusting the weights and biases of the neurons in the network during training, the network can learn to recognize patterns and relationships in the data and make accurate predictions.

**6. The sigmoid activation function**

is a type of activation function used in artificial neural networks. It is a smooth, continuous function that maps any real-valued number to a value between 0 and 1. It is defined as:

f(x) = 1 / (1 + e^(-x))

where x is the input to the activation function and e is the base of the natural logarithm.

The sigmoid activation function has several properties that make it useful in neural networks:

It is differentiable, which means that it has a well-defined derivative that can be used in the backpropagation algorithm to update the weights and biases of the neurons in the network.

It has a simple, smooth shape, which makes it easy to optimize and train the network.

It maps the input to an output that is bounded between 0 and 1, which can be useful for modeling probabilities and binary classification tasks.

**7. Weights in an artificial neural network**

Weights in an artificial neural network are numerical values that are associated with the connections between neurons. These weights represent the strength of the connection between neurons and the influence that one neuron has on another. In other words, they determine how much impact a given neuron's output has on the output of other neurons in the network.

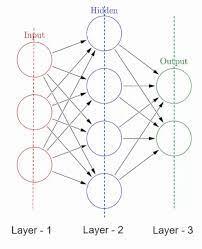
During the training process of a neural network, the weights are adjusted in order to minimize the error between the predicted output of the network and the true desired output. This process is known as weight optimization and is typically done using an algorithm such as backpropagation.

The weights of a neural network can have a significant impact on its performance and ability to learn. If the weights are not initialized correctly or are not updated correctly during training, the network may not be able to learn effectively and may produce poor results. Therefore, it is important to choose appropriate initialization and update methods for the weights in order to ensure that the network is able to learn effectively.

There are several different approaches that can be used to initialize and update the weights in a neural network, including random initialization, Glorot initialization, and He initialization. The choice of initialization and update method can depend on the specific task and network architecture being used.

Overall, the weights of a neural network are an important aspect of its structure and function, and they play a crucial role in the learning process of the network. Properly initialized and updated weights can help the network to learn effectively and achieve good performance on a given task.

**8. The standard structure of an artificial neural network (ANN) consists of three main types of layers:**



Input layer: The input layer is the first layer of the network and is responsible for receiving input data and passing it on to the next layer. The number of neurons in the input layer is equal to the number of features in the input data.

Hidden layer(s): The hidden layer(s) of the network are located between the input and output layers and are responsible for processing and transforming the input data. The number of hidden layers and the number of neurons in each layer can vary depending on the complexity of the task and the specific needs of the network.

Output layer: The output layer is the last layer of the network and is responsible for producing the final output of the network based on the input data and the processing done in the hidden layer(s). The number of neurons in the output layer is equal to the number of classes or output values that the network is designed to predict.

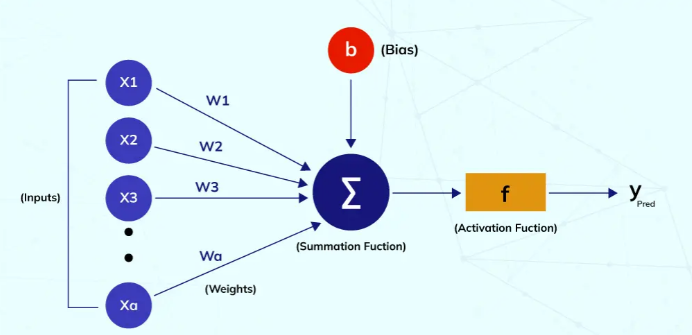
**9. Feedforward neural network and Feedback neural network**

A feedforward neural network is a type of ANN that consists of an input layer, one or more hidden layers, and an output layer. The information flows only in one direction through the network, from the input layer to the output layer, without any loops or feedback.

A feedback neural network, also known as a recurrent neural network (RNN), is a type of ANN that allows information to flow in both directions through the network. In a feedback network, the output of a neuron can be fed back into itself or into other neurons in the network, creating a loop or feedback path.

**10. What is a perceptron**

A perceptron is a simple model of a biological neuron in an artificial neural network.A perceptron works by taking in some numerical inputs along with what is known as weights and a bias. It then multiplies these inputs with the respective weights(this is known as the weighted sum). These products are then added together along with the bias.



**Each perceptron comprises four different parts:**

Input Values: A set of values or a dataset for predicting the output value. They are also described as a dataset’s features and dataset.

Weights: The real value of each feature is known as weight. It tells the importance of that feature in predicting the final value.

Bias: The activation function is shifted towards the left or right using bias. You may understand it simply as the y-intercept in the line equation.

Summation Function: The summation function binds the weights and inputs together. It is a function to find their sum.

Activation Function: It introduces non-linearity in the perceptron model.

**11. What is a Bias in neural network**

Bias in an artificial neural network (ANN) is a parameter that represents the threshold for activation of a neuron. It is added to the weighted sum of the input data and determines whether the neuron will be activated or not. Bias helps to adjust the output of a neuron based on the input data and the connections between neurons, making it possible to perform tasks more accurately.

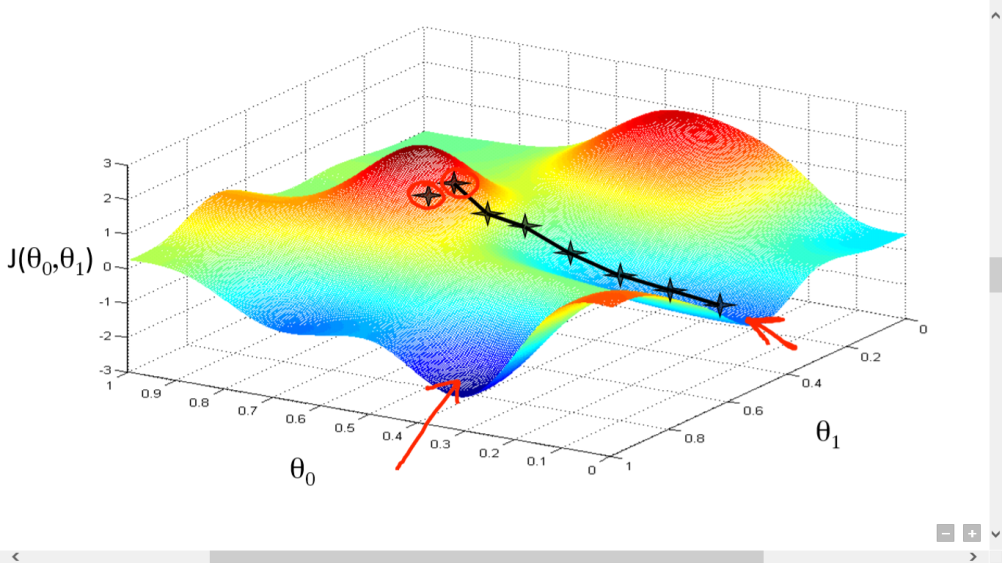
If bias was not added in an artificial neural network (ANN), the output of a neuron would be determined solely by the weighted sum of the input data and the connections between neurons. This could potentially result in the following issues:

The output of the neuron may be more sensitive to the scale and range of the input data. If the input data has a large range or scale, it may be difficult to adjust the weights of the connections between neurons to produce the desired output.

The output of the neuron may be more prone to overfitting or underfitting. Without a bias term, the network may be more likely to learn patterns and relationships in the training data that are specific to that dataset, but may not generalize well to new data.

The output of the neuron may be more difficult to control and optimize. Without a bias term, it may be more difficult to adjust the output of the neuron to produce the desired output, particularly for tasks that require precise control or prediction.

**12. What is a gradient descent in backpropagation algorithm**



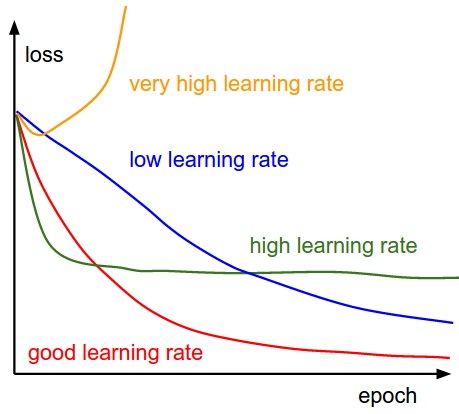
In the backpropagation algorithm, gradient descent is used to optimize the weights and biases of the neurons in an artificial neural network (ANN) in order to minimize the error between the predicted output of the network and the true desired output.The gradient represents the slope of the error function and indicates the direction in which the weights and biases should be adjusted in order to minimize the error. The algorithm then updates the weights and biases of the neurons in the network using the gradient and a learning rate, which determines the size of the update.

**13. What is learning rate in backpropogation algorithm**

In the backpropagation algorithm, the learning rate is a hyperparameter that determines the step size at which the algorithm updates the weights of the neural network during training. It is used to control the speed at which the algorithm converges on a solution.

The learning rate is a scalar value that is multiplied by the gradient of the loss function with respect to the weights at each iteration of the algorithm. The size of the update to the weights is then determined by the product of the learning rate and the gradient.

A smaller learning rate means that the algorithm takes smaller steps and requires more iterations to converge on a solution, but it also means that the algorithm is less likely to overshoot the optimal solution and may be more stable. On the other hand, a larger learning rate means that the algorithm takes larger steps and may converge faster, but it also means that the algorithm is more prone to overshooting the optimal solution and may be less stable.



**14. Applications of backpropogation algorithm**

Some specific applications of backpropagation include:

Computer vision: Backpropagation is often used to train convolutional neural networks (CNNs), which are widely used for image classification, object detection, and segmentation tasks.

Natural language processing: Backpropagation is used to train recurrent neural networks (RNNs), which are commonly used for tasks such as language translation, language generation, and sentiment analysis.

Speech recognition: Backpropagation is used to train neural networks for tasks such as speech-to-text transcription and speaker identification.

Robotics: Backpropagation is used to train neural networks for tasks such as robot navigation and control.

Medicine: Backpropagation is used to train neural networks for tasks such as diagnosis and prognosis of diseases, and predicting the effects of treatments.

Finance: Backpropagation is used to train neural networks for tasks such as stock price prediction and fraud detection.

Overall, the backpropagation algorithm is a widely used and powerful tool for training neural networks, and it has many applications in a variety of fields.

**15. What is an epoch**

The act of sending the data from the input layer to the output layer then all the way back is called an Epoch. In each epoch, the Neural Network updates the weights of the Neurons which is also known as Learning.

**16. Explanation of code**

This code defines a neural network that takes in 3 data points, each with 2 features, and trains a model to predict a single output value (a score between 0 and 100) for each data point. The neural network has 2 input nodes (one for each feature), 3 hidden nodes, and 1 output node.

The input data, stored in the X array, and the output data, stored in the y array, are first preprocessed by normalizing the values to be between 0 and 1. This is done by dividing each element in X by the maximum value in X along the longitudinal axis (axis=0). The output data is divided by 100 to scale the values down.

The neural network uses the sigmoid function as its activation function. The sigmoid function maps any input value to a value between 0 and 1. The derivative of the sigmoid function is also defined in this code.

The neural network is trained using the backpropagation algorithm. The model makes a prediction using the forward propagation step, then compares the predicted output to the actual output using the error (EO). The error is then used to update the weights and biases of the model during the backpropagation step. This process is repeated for a fixed number of epochs (10 in this case).

The weights and biases of the model are initialized randomly using a uniform distribution. The model makes predictions for the input data at each epoch and prints out the input data, the actual output, and the predicted output. Finally, the model makes a final prediction on the input data and prints out the input data, the actual output, and the final predicted output.