Q.1 Distinguish between artificial neuron & biological neuron?

Answer:

Feature	Biological Neuron	Artificial Neuron
Structure	Complex, organic cell with dendrites, soma, axon, synapses	Simple mathematical model with inputs, weights, bias, activation function
Signal Transmission	Electrochemical process involving neurotransmitters across synapses	Mathematical operations (weighted sum, activation)
Processing	Parallel and distributed	Primarily sequential (though can be parallelized in hardware)
Learning	Synaptic plasticity (strengthening/weakening of connections)	Weight adjustment through algorithms like backpropagation
Speed	Relatively slow (milliseconds)	Very fast (nanoseconds)
Fault Tolerance	High (can adapt to some damage)	Lower (sensitive to component failures)

Export to Sheets

Q.2 Draw the basic model of Adaline network and Madaline network?

Answer:

- **Adaline (Adaptive Linear Neuron):**
    - A single layer network with linear activation function.
    - Diagram:
        - Inputs (x1, x2, ..., xn) are fed into nodes.
        - Each input has an associated weight (w1, w2, ..., wn).
        - The weighted sum is calculated: z = w1*x1 + w2*x2 + ... + wn*xn + b (where b is the bias).
        - The linear activation function (f(z) = z) is applied.
        - The output is the result of the activation function.
- **Madaline (Multiple Adaline):**
    - A network with multiple Adaline units in the first layer and a single output unit with a hard-limit activation function.
    - Diagram:
        - Inputs (x1, x2, ..., xn) are fed into multiple Adaline units (each with its own weights and bias).
        - Each Adaline unit produces an output.

- These outputs are fed into a single output unit.
- The output unit applies a hard-limit activation function (e.g., sign function) to produce the final output.

Q.3 Distinguish between supervised learning and unsupervised learning algorithm.

Answer:

Feature	Supervised Learning	Unsupervised Learning
Data	Labeled data (input-output pairs)	Unlabeled data (only inputs)
Goal	Learn a mapping function to predict outputs for new inputs	Discover patterns and structures in the data
Examples	Classification, Regression	Clustering, Dimensionality Reduction, Association Rule Mining
Evaluation	Based on prediction accuracy (comparing predicted outputs to true labels)	Based on internal metrics (e.g., cluster separation, data reconstruction error)

Export to Sheets

Q.4 Draw and explain the architecture of back propagation algorithm.

Answer:

- **Architecture:**
  - Multi-layered feedforward network with an input layer, one or more hidden layers, and an output layer.
  - Each layer consists of neurons (nodes).
  - Neurons are connected by weighted connections.
  - Activation functions are used in neurons (typically sigmoid, ReLU, etc.).
- **Algorithm Explanation:**

1. **Forward Pass:**
   - Input is fed into the network.
   - Signals propagate through the layers, with weighted sums and activation functions applied at each neuron.
   - Output is produced at the output layer.
2. **Error Calculation:**
   - The error (difference between the predicted output and the actual output) is calculated.
3. **Backward Pass (Backpropagation):**

- The error is propagated backward through the network, layer by layer.
- Weights are adjusted based on their contribution to the error, using gradient descent.
- The goal is to minimize the error.

4. **Iteration:**
    - Steps 1-3 are repeated for multiple training examples (epochs) until the error is minimized.

Q.5 What is a back propagation NN? What are the factors affecting back propagation training?

Answer:

- **Backpropagation NN (Neural Network):**
    - A neural network trained using the backpropagation algorithm.
    - It's a supervised learning algorithm commonly used for tasks like classification and regression.
- **Factors Affecting Training:**
    - **Learning Rate:** Controls the step size for weight updates. Too high can lead to instability, too low can slow down convergence.
    - **Momentum:** Helps accelerate learning in consistent directions and avoid local minima.
    - **Network Architecture:** Number of layers and neurons per layer.
    - **Activation Function:** Choice of activation function (sigmoid, ReLU, etc.) can impact learning.
    - **Initialization of Weights:** Random initialization is common, but methods like Xavier/Glorot initialization can improve convergence.
    - **Training Data:** Size, quality, and representativeness of the training data.
    - **Overfitting:** Occurs when the network memorizes the training data instead of generalizing.
    - **Local Minima:** The algorithm might get stuck in a local minimum of the error function.

Q.6 Discuss in detail the various types of activation function used in neural network with aid of graphical as well as mathematical representation and its output.

Answer:

- **Sigmoid (Logistic):**
    - **Mathematical Representation:** f(z) = 1 / (1 + e^(-z))
    - **Output Range:** (0, 1)
    - **Use:** Binary classification, output as probability.
    - **Graph:** S-shaped curve.
- **Tanh (Hyperbolic Tangent):**
    - **Mathematical Representation:** f(z) = (e^z - e^(-z)) / (e^z + e^(-z))

- **Output Range:** (-1, 1)
- **Use:** Similar to sigmoid, but centered around 0.
- **Graph:** S-shaped curve, steeper than sigmoid.

- **ReLU (Rectified Linear Unit):**
  - **Mathematical Representation:** f(z) = max(0, z)
  - **Output Range:** [0, ∞)
  - **Use:** Most common activation function in deep learning, avoids vanishing gradient problem.
  - **Graph:** Linear for positive values, zero for negative values.
- **Leaky ReLU:**
  - **Mathematical Representation:** f(z) = z if z > 0, else f(z) = αz (where α is a small constant)
  - **Output Range:** (-∞, ∞)
  - **Use:** Addresses the "dying ReLU" problem (neurons getting stuck with zero output).
  - **Graph:** Similar to ReLU, but with a small slope for negative values.
- **Softmax:**
  - **Mathematical Representation:** f(z_i) = e^(z_i) / Σ(e^(z_j)) (for all j)
  - **Output Range:** (0, 1) for each output, sum of outputs is 1.
  - **Use:** Multi-class classification, output as probability distribution.
  - **Graph:** Not easily represented as a single curve, but as a distribution over multiple outputs.

Q.7 Using McCulloch-Pitts neuron model, design a neural network for 2-input XOR functions?

Answer:

- **McCulloch-Pitts Neuron Model:**
  - A simplified model of a biological neuron with binary inputs and outputs.
  - It uses a threshold function to determine the output.
- **XOR Network Design:**
  - XOR function: Output is 1 if inputs are different, 0 if inputs are the same.
  - Need two hidden neurons and one output neuron.
  - Weights and thresholds need to be carefully chosen.
- **Network Representation:**
  - Input Layer: x1, x2
  - Hidden Layer: h1, h2

- Output Layer: y

- **Weights and Thresholds:**
  - h1: w11 = 1, w21 = 1, θ1 = 1.5
  - h2: w12 = -1, w22 = -1, θ2 = -0.5
  - y: wh1 = 1, wh2 = 1, θy = 0.5
- **Equations:**
  - h1 = 1 if (x1 + x2) >= 1.5, else 0
  - h2 = 1 if (-x1 - x2) >= -0.5, else 0
  - y = 1 if (h1 + h2) >= 0.5, else 0

Q.8 Explain with a neat diagram the neural network architecture of multilayer feed forward network.

Answer:

- **Diagram:**

Input Layer Hidden Layer 1 Hidden Layer 2 Output Layer

x1 h11 h21 y1

x2 h12 h22 y2

... ...

xn h1m h2p yk

- **Explanation:**
  - **Input Layer:** Receives the input data (x1, x2, ..., xn).
  - **Hidden Layers:** One or more layers between the input and output layers. They perform non-linear transformations on the data.
  - **Output Layer:** Produces the final output (y1, y2, ..., yk).
  - **Connections:** Neurons are connected by weighted connections.
  - **Feedforward:** Information flows in one direction, from input to output.

Q.9 Explain briefly the full counter propagation with architecture and its functioning.

Answer:

- **Full Counterpropagation Network:**
  - A type of neural network that combines features of self-organizing maps (SOMs) and Grossberg networks.
  - It's used for pattern association and function approximation.

- **Architecture:**
    - **Input Layer:** Receives input patterns.
    - **Kohonen Layer (Competitive Layer):** Performs competitive learning to find the best matching unit (BMU) for the input pattern.
    - **Grossberg Layer (Output Layer):** Learns the association between the BMU and the desired output pattern.
- **Functioning:**

1. **Competition:** Input pattern is presented to the Kohonen layer, and the BMU is determined based on the similarity between the input and the weights of the Kohonen neurons.

2. **Weight Adaptation:** The weights of the BMU and its neighbors are updated to become more similar to the input pattern.

3. **Association:** The Grossberg layer learns to associate the BMU with the corresponding output pattern.

4. **Output Generation:** When a new input is presented, the Kohonen layer finds the BMU, and the Grossberg layer generates the associated output.

Q.10 With neat architecture, explain the training algorithm of Kohonen self-organizing feature maps.

Answer:

- **Architecture:**
    - **Input Layer:** Receives input patterns.
    - **Kohonen Layer (Competitive Layer):** Arranged in a grid or other topology. Neurons in this layer compete to respond to the input.
- **Training Algorithm:**

1. **Initialization:** Initialize the weights of the Kohonen neurons to small random values.

2. **Input Presentation:** Present an input pattern to the network.

3. **Best Matching Unit (BMU) Selection:** Calculate the distance (e.g., Euclidean distance) between the input pattern and the weight vectors of all Kohonen neurons. Select the neuron with the smallest distance as the BMU.

4. **Weight Update:** Update the weights of the BMU and its neighbors to become more similar to the input pattern. The learning rate and neighborhood size decrease over time.

5. **Iteration:** Repeat steps 2-4 for multiple input patterns and epochs.