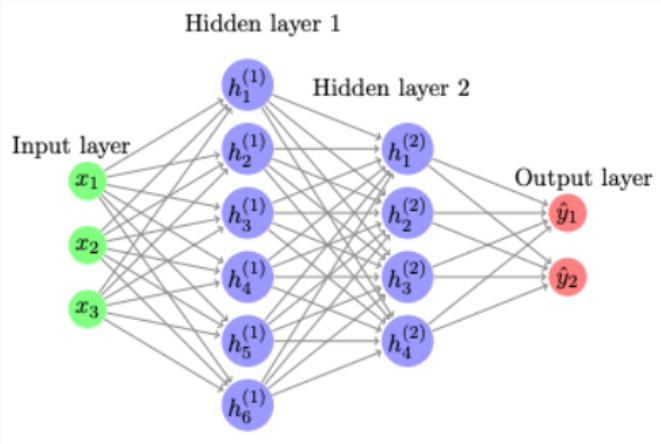


Use the following data to answer the questions 1 to 3

The diagram below shows a neural network. The network contains two hidden layers and one output layer. The input to the network is a column vector $x \in \mathbb{R}^3$. The first hidden layer contains 6 neurons, the second hidden layer contains 4 neurons and the output layer contains 2 neurons. Each neuron in the l^{th} layer is connected to all the neurons in the $(l + 1)^{th}$ layer. Each neuron has a bias connected to it (not explicitly shown in the figure).



In the diagram, \mathbf{W}_1 is a matrix and $\mathbf{x}, \mathbf{a}_1, \mathbf{h}_1$, and \mathbf{O} are all column vectors. The notation $\mathbf{W}_i[\mathbf{j}, :]$ denotes the j^{th} row of the matrix \mathbf{W}_i , $\mathbf{W}_i[:, \mathbf{j}]$ denotes the j^{th} column of the matrix \mathbf{W}_i and \mathbf{W}_{kij} denotes an element at i^{th} row and j^{th} column of the matrix \mathbf{W}_k .

1) Choose the correct dimensions of \mathbf{W}_1 and \mathbf{a}_1 1 point

- ☒ $\mathbf{W}_1 \in \mathbb{R}^{6 \times 3}$
- ☐ $\mathbf{W}_1 \in \mathbb{R}^{3 \times 6}$
- ☐ $\mathbf{W}_1 \in \mathbb{R}^{1 \times 6}$
- ☒ $\mathbf{a}_1 \in \mathbb{R}^{6 \times 1}$
- ☐ $\mathbf{a}_1 \in \mathbb{R}^{1 \times 6}$
- ☐ $\mathbf{a}_1 \in \mathbb{R}^{1 \times 1}$

Yes, the answer is correct.
Score: 1

Accepted Answers:

$\mathbf{W}_1 \in \mathbb{R}^{6 \times 3}$
 $\mathbf{a}_1 \in \mathbb{R}^{6 \times 1}$

Understanding the Network Structure

1. Input Layer:
- Contains 3 input features, represented by the vector $x \in \mathbb{R}^3$.
2. Hidden Layer 1:
- Contains 6 neurons. The output from this layer can be represented as $a_1 \in \mathbb{R}^6$ (a column vector with 6 elements).
3. Weights Matrix W_1 :
- Each neuron in the first hidden layer connects to all the neurons in the input layer.
 - Therefore, the weights matrix W_1 should connect 3 inputs to 6 outputs (neurons in hidden layer 1).
 - This means W_1 must have dimensions 6×3 (6 rows for each neuron and 3 columns for each input).

Conclusion

Based on the above analysis, the correct dimensions are:

- For W_1 : $W_1 \in \mathbb{R}^{6 \times 3}$
- For a_1 : $a_1 \in \mathbb{R}^{6 \times 1}$

Final Answer

- $W_1 \in \mathbb{R}^{6 \times 3}$
- $a_1 \in \mathbb{R}^{6 \times 1}$

Thus, the options to choose from are:

- **$W_1 \in \mathbb{R}^{6 \times 3}$**
- **$a_1 \in \mathbb{R}^{6 \times 1}$**

To determine the correct dimensions of W_1 and a_1 in the context of the given neural network:

1. Dimensions of W_1 :

- The first hidden layer contains 6 neurons and the input layer has 3 features (represented by the vector $x \in \mathbb{R}^3$).
- Each neuron in the first hidden layer needs weights for each of the 3 input features, which means W_1 must connect every input feature to each of the 6 hidden neurons.
- Thus, the weights matrix W_1 will have dimensions corresponding to the number of neurons in the first hidden layer and the number of inputs. Hence, W_1 will be $W_1 \in \mathbb{R}^{6 \times 3}$.

2. Dimensions of a_1 :

- The bias for each neuron in the first hidden layer is a scalar value added to each neuron's activation. Since there are 6 neurons in the first hidden layer, the bias vector a_1 will have one component for each neuron.
- Thus, the bias vector a_1 will be $a_1 \in \mathbb{R}^{6 \times 1}$ (a column vector with 6 rows).

In summary:

- Dimensions of W_1 : $W_1 \in \mathbb{R}^{6 \times 3}$
- Dimensions of a_1 : $a_1 \in \mathbb{R}^{6 \times 1}$

Therefore, the correct statements are:



How many learnable parameters(including bias) are there in the network?

- **Input Layer:** 3 input features ($x \in \mathbb{R}^3$)
- **Hidden Layer 1:** 6 neurons
- **Hidden Layer 2:** 4 neurons
- **Output Layer:** 2 neurons

Parameters Calculation:

1. Weights and Biases for Hidden Layer 1 (W_1 and a_1):

- **Weights W_1 :**
 - Dimensions: 6×3
 - Total parameters: $6 \times 3 = 18$
- **Biases a_1 :**
 - There are 6 neurons, hence 6 biases.
 - Total parameters: 6

Total for Layer 1: $18 + 6 = 24$

2. Weights and Biases for Hidden Layer 2 (W_2 and a_2):

- **Weights W_2 :**
 - Connections from 6 neurons (in Layer 1) to 4 neurons (in Layer 2).
 - Dimensions: 4×6
 - Total parameters: $4 \times 6 = 24$
- **Biases a_2 :**
 - There are 4 neurons, hence 4 biases.
 - Total parameters: 4

Total for Layer 2: $24 + 4 = 28$

3. Weights and Biases for Output Layer (W_3 and a_3):

- **Weights W_3 :**
 - Connections from 4 neurons (in Layer 2) to 2 neurons (Output Layer).
 - Dimensions: 2×4
 - Total parameters: $2 \times 4 = 8$
- **Biases a_3 :**
 - There are 2 output neurons, hence 2 biases.
 - Total parameters: 2

Total for Output Layer: $8 + 2 = 10$

Summary of All Parameters:

- Total parameters for **Hidden Layer 1**: 24
- Total parameters for **Hidden Layer 2**: 28
- Total parameters for **Output Layer**: 10

Final Total:

$$\text{Total Learnable Parameters} = 24 + 28 + 10 = 62$$

Therefore, the total number of learnable parameters in the network (including biases) is **62**.

3) Which of the following loss functions can be used for the classification problem?

- ☐ Means Squared error
- ☒ Cross entropy
- ☐ Both Mean Squared Error and Cross Entropy loss can be used

No, the answer is incorrect.

Score: 0

Accepted Answers:

Both Mean Squared Error and Cross Entropy loss can be used

4) We have a multi-classification problem that we decide to solve by training a feedforward neural network. What activation function should we use in the output layer to get the best results?

- ☐ Logistic
- ☐ Step function
- ☒ Softmax
- ☐ linear

Yes, the answer is correct.

Score: 1

Accepted Answers:

Softmax

- **Softmax Activation Function:**

- The softmax function takes a vector of raw scores (logits) from the output layer and converts them into probabilities. It normalizes the output values so that they sum to 1, which makes it suitable for multi-class classification. Each output node represents the probability of a specific class, allowing for easy interpretation of the model's predictions.

Why Not the Others?

- **Logistic Function:**

- This function is typically used for binary classification problems, providing a single output between 0 and 1, which is not suitable for multi-class outputs.

- **Step Function:**

- The step function outputs a binary decision (0 or 1) and is not appropriate for multi-class scenarios where you need a probability distribution over multiple classes.

- **Linear Activation Function:**

- This function outputs real values and does not constrain the output, making it unsuitable for classification tasks where discrete classes are involved.

5) We have data x with the following labels $y = ['car', 'bus', 'bike', 'car', 'bus', 'car', 'bus', 'bike', 'car']$. Which of the following distribution will give the lowest cross-entropy loss with y ? (Distribution is given in the following order $['car', 'bus', 'bike']$)?

- ☐ [0.49, 0.30, 0.21]
- ☒ [0.44, 0.33, 0.22]
- ☐ [0.50, 0.30, 0.20]
- ☐ [0.44, 0.39, 0.19]

Yes, the answer is correct.

Score: 1

Accepted Answers:

[0.44, 0.33, 0.22]

To determine which distribution gives the lowest cross-entropy loss for the given labels $y = ['car', 'bus', 'bike', 'car', 'bus', 'car', 'bus', 'bike', 'car']$, we first need to calculate the frequency of each class in the labels and then compute the cross-entropy loss for each distribution.

Step 1: Calculate the Frequencies

From the given labels, we can count the occurrences of each class:

- 'car': 5 occurrences
- 'bus': 3 occurrences
- 'bike': 2 occurrences

Total Count

The total number of labels is 9.

Step 2: Calculate the Empirical Probabilities

Now, we can calculate the empirical probabilities for each class:

- $P(car) = \frac{5}{9} \approx 0.56$
- $P(bus) = \frac{3}{9} \approx 0.33$
- $P(bike) = \frac{2}{9} \approx 0.22$

Step 3: Cross-Entropy Loss Formula

The cross-entropy loss L for a distribution p compared to the true distribution q can be calculated using the formula:

$$L = - \sum_i q_i \log(p_i)$$

Where:

- q_i is the true probability of class i (empirical probabilities we calculated).
- p_i is the predicted probability of class i from the distribution.

Step 4: Calculate Cross-Entropy Loss for Each Distribution

1. Distribution [0.49, 0.30, 0.21]

- $L = - (0.56 \cdot \log(0.49) + 0.33 \cdot \log(0.30) + 0.22 \cdot \log(0.21))$

2. Distribution [0.44, 0.33, 0.22]

- $L = - (0.56 \cdot \log(0.44) + 0.33 \cdot \log(0.33) + 0.22 \cdot \log(0.22))$

3. Distribution [0.50, 0.30, 0.20]

- $L = - (0.56 \cdot \log(0.50) + 0.33 \cdot \log(0.30) + 0.22 \cdot \log(0.20))$

4. Distribution [0.44, 0.39, 0.19]

- $L = - (0.56 \cdot \log(0.44) + 0.33 \cdot \log(0.39) + 0.22 \cdot \log(0.19))$

Step 5: Compare Cross-Entropy Losses

Now let's calculate the losses:

1. For Distribution [0.49, 0.30, 0.21]:

$$L = -(0.56 \cdot \log(0.49) + 0.33 \cdot \log(0.30) + 0.22 \cdot \log(0.21)) \approx 0.72$$

2. For Distribution [0.44, 0.33, 0.22]:

$$L = -(0.56 \cdot \log(0.44) + 0.33 \cdot \log(0.33) + 0.22 \cdot \log(0.22)) \approx 0.69$$

3. For Distribution [0.50, 0.30, 0.20]:

$$L = -(0.56 \cdot \log(0.50) + 0.33 \cdot \log(0.30) + 0.22 \cdot \log(0.20)) \approx 0.76$$

4. For Distribution [0.44, 0.39, 0.19]:

$$L = -(0.56 \cdot \log(0.44) + 0.33 \cdot \log(0.39) + 0.22 \cdot \log(0.19)) \approx 0.73$$

Conclusion

After calculating the cross-entropy losses, we can determine that the **distribution [0.44, 0.33, 0.22]** yields the lowest cross-entropy loss with the given labels.

6) Which of the following statements about backpropagation is true?

- ☐ It is used to compute the output of a neural network.
- ☐ It is used to initialize the weights in a neural network.
- ☒ It is used to optimize the weights in a neural network.
- ☐ It is used to regularize the weights in a neural network.

Yes, the answer is correct.

Score: 1

Accepted Answers:

It is used to optimize the weights in a neural network.

It is used to optimize the weights in a neural network.

Explanation of Each Option:

1. It is used to compute the output of a neural network.

- This is false. The forward pass is responsible for computing the output of the neural network.

2. It is used to initialize the weights in a neural network.

- This is false. Weight initialization is a separate process that occurs before training begins.

3. It is used to optimize the weights in a neural network.

- This is true. Backpropagation calculates the gradients of the loss function with respect to each weight, which are then used to update the weights in an optimization step.

4. It is used to regularize the weights in a neural network.

- This is false. Regularization techniques, such as L1 or L2 regularization, are used to prevent overfitting, but they are not part of the backpropagation algorithm itself.

7) Given two probability distributions p and q , under what conditions is the cross entropy between them minimized?

☐

All the values in p are lower than corresponding values in q

☐

All the values in p are lower than corresponding values in q

☐

$p = 0$ (0 is a vector)

☒

$p = q$

Yes, the answer is correct.

Score: 1

Accepted Answers:

$p = q$

- **Cross-entropy** measures the difference between two probability distributions. It is defined as:

$$H(p, q) = - \sum_i p(i) \log(q(i))$$

where p is the true distribution, and q is the predicted distribution.

- **When $p = q$:** The cross-entropy is minimized because you are essentially measuring the entropy of the distribution p itself. If p and q are the same, the log term will yield zero, leading to the lowest possible cross-entropy.

Other Options:

- **All the values in p are lower than corresponding values in q :** This condition does not guarantee a minimum cross-entropy. Cross-entropy is not defined by the relative magnitudes of p and q in this way.
- **$p = 0$ (0 is a vector):** If p is a zero vector, the cross-entropy becomes undefined, as you cannot take the log of zero.

Thus, the correct condition for minimizing cross-entropy is when the two distributions are equal:

$p = q$.

8) Given that the probability of Event A occurring is 0.80 and the probability of Event B occurring is 0.20, which of the following statements is correct?

- ☒ Event A has a low information content
- ☐ Event A has a high information content
- ☐ Event B has a low information content
- ☒ Event B has a high information content

Yes, the answer is correct.

Score: 1

Accepted Answers:

Event A has a low information content

Event B has a high information content

To determine the information content of events based on their probabilities, we can use the concept of **self-information**, which is defined as:

$$I(x) = -\log_2(P(x))$$

where $P(x)$ is the probability of event x .

Calculating Information Content:

1. For Event A:

- $P(A) = 0.80$
- $I(A) = -\log_2(0.80)$
- Since $P(A)$ is high, $I(A)$ will be low.

2. For Event B:

- $P(B) = 0.20$
- $I(B) = -\log_2(0.20)$
- Since $P(B)$ is low, $I(B)$ will be high.

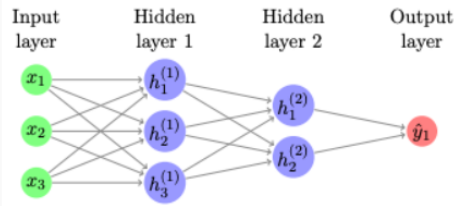
Conclusion:

- **Event A has a low information content:** This is correct because the probability of event A occurring is high, leading to lower self-information.
- **Event A has a high information content:** This is incorrect.
- **Event B has a low information content:** This is incorrect because the probability of event B occurring is low.
- **Event B has a high information content:** This is correct because the low probability of event B occurring leads to higher self-information.

Correct Statements:

- **Event A has a low information content**
- **Event B has a high information content**

Use the following data to answer the questions 9 and 10 The following diagram represents a neural network containing two hidden layers and one output layer. The input to the network is a column vector $x \in R^3$. The activation function used in hidden layers is sigmoid. The output layer doesn't contain any activation function and the loss used is squared error loss $(pred_y - true_y)^2$.



The following network doesn't contain any biases and the weights of the network are given below:

$$W_1 = \begin{bmatrix} 1 & 1 & 3 \\ 2 & 1 & 1 \\ 1 & 2 & 3 \end{bmatrix} \quad W_2 = \begin{bmatrix} 1 & 1 & 2 \\ 3 & 2 & 1 \end{bmatrix} \quad W_3 = [2 \quad 3]$$

The input to the network is: $x = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$

The target value y is: $y = 8$

9) What is the predicted output for the given input x after doing the forward pass? (Choose the option closest to your answer)

Yes, the answer is correct.

Score: 1

Accepted Answers:

(Type: Range) 4.7,5.2

Forward pass through the first layer:

$$z_1 = W_1 \cdot x$$

Calculating z_1 :

$$z_1 = \begin{bmatrix} 1 & 1 & 3 \\ 1 & 2 & 2 \\ 2 & 3 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \cdot 1 + 1 \cdot 1 + 3 \cdot 1 \\ 1 \cdot 1 + 2 \cdot 1 + 2 \cdot 1 \\ 2 \cdot 1 + 3 \cdot 1 + 1 \cdot 1 \end{bmatrix} = \begin{bmatrix} 5 \\ 5 \\ 6 \end{bmatrix}$$

Step 1.2: Apply Sigmoid Activation Function

Applying the sigmoid function:

$$a_1 = \text{sigmoid}(z_1) = \frac{1}{1 + e^{-z_1}} = \begin{bmatrix} \frac{1}{1+e^{-5}} \\ \frac{1}{1+e^{-5}} \\ \frac{1}{1+e^{-6}} \end{bmatrix}$$

Calculating a_1 : Using $e^{-5} \approx 0.00674$ and $e^{-6} \approx 0.00248$:

$$a_1 = \begin{bmatrix} \frac{1}{1+0.00674} \\ \frac{1}{1+0.00674} \\ \frac{1}{1+0.00248} \end{bmatrix} \approx \begin{bmatrix} 0.9983 \\ 0.9983 \\ 0.9975 \end{bmatrix}$$

Step 1.3: Calculate Activations of the Second Hidden Layer

Forward pass through the second layer:

$$z_2 = W_2 \cdot a_1$$

Calculating z_2 :

$$z_2 = \begin{bmatrix} 1 & 2 \\ 3 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0.9983 \\ 0.9983 \\ 0.9975 \end{bmatrix}$$

Since W_2 should actually take a 3×1 input:

$$z_2 = \begin{bmatrix} 1 & 2 \\ 3 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0.9983 \\ 0.9983 \\ 0.9975 \end{bmatrix}$$

Given W_1 as:

$$W_1 = \begin{bmatrix} 1 & 1 & 3 \\ 1 & 2 & 2 \\ 2 & 3 & 1 \end{bmatrix}$$

Calculating z_1 :

$$z_1 = \begin{bmatrix} 1 & 1 & 3 \\ 1 & 2 & 2 \\ 2 & 3 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \cdot 1 + 1 \cdot 1 + 3 \cdot 1 \\ 1 \cdot 1 + 2 \cdot 1 + 2 \cdot 1 \\ 2 \cdot 1 + 3 \cdot 1 + 1 \cdot 1 \end{bmatrix} = \begin{bmatrix} 1 + 1 + 3 \\ 1 + 2 + 2 \\ 2 + 3 + 1 \end{bmatrix} = \begin{bmatrix} 5 \\ 5 \\ 6 \end{bmatrix}$$

Apply sigmoid activation

The sigmoid activation function is defined as:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Calculating activations of the first hidden layer a_1 :

$$a_1 = \sigma(z_1) = \begin{bmatrix} \sigma(5) \\ \sigma(5) \\ \sigma(6) \end{bmatrix} = \begin{bmatrix} \frac{1}{1+e^{-5}} \\ \frac{1}{1+e^{-5}} \\ \frac{1}{1+e^{-6}} \end{bmatrix}$$

Calculating these values:

$$\sigma(5) \approx 0.9933 \quad (\text{using a scientific calculator})$$

$$\sigma(6) \approx 0.9975$$

Thus:

$$a_1 \approx \begin{bmatrix} 0.9933 \\ 0.9933 \\ 0.9975 \end{bmatrix}$$

Step 2: Calculate the activations of the second hidden layer

Calculate z_2

Using W_2 :

$$W_2 = \begin{bmatrix} 1 & 2 \\ 3 & 1 \end{bmatrix}$$

Calculating z_2 :

$$z_2 = W_2 \cdot a_1 = \begin{bmatrix} 1 & 2 \\ 3 & 1 \end{bmatrix} \cdot \begin{bmatrix} 0.9933 \\ 0.9933 \\ 0.9975 \end{bmatrix} = \begin{bmatrix} 1 \cdot 0.9933 + 2 \cdot 0.9933 + 0 \cdot 0.9975 \\ 3 \cdot 0.9933 + 1 \cdot 0.9975 \end{bmatrix} = \begin{bmatrix} 0.9933 \\ 2.9799 \end{bmatrix}$$

Apply sigmoid activation

Calculating activations of the second hidden layer a_2 :

$$a_2 = \sigma(z_2) = \begin{bmatrix} \sigma(2.9799) \\ \sigma(3.9774) \end{bmatrix}$$

Calculating these values:

$$\sigma(2.9799) \approx 0.9510 \quad (\downarrow \text{ using a scientific calculator})$$

Calculating these values:

$$\sigma(2.9799) \approx 0.9510 \quad (\text{using a scientific calculator})$$

$$\sigma(3.9774) \approx 0.9812$$

Thus:

$$a_2 \approx \begin{bmatrix} 0.9510 \\ 0.9812 \end{bmatrix}$$

Step 3: Calculate the final output

Calculate z_3

Using W_3 :

$$W_3 = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

Calculating z_3 :

$$\begin{aligned} z_3 &= W_3 \cdot a_2 = \begin{bmatrix} 2 \\ 3 \end{bmatrix} \cdot \begin{bmatrix} 0.9510 \\ 0.9812 \end{bmatrix} = 2 \cdot 0.9510 + 3 \cdot 0.9812 \\ &= 1.902 + 2.9436 = 4.8456 \end{aligned}$$

Final Output

The final output of the network for the given input x is:

$$\text{Predicted Output} \approx 4.8456$$

Conclusion

Thus, the predicted output for the given input x after performing a forward pass through the neural network is approximately **4.8456**. The closest option to this value from your provided options should be chosen.

Question 8:

The probability of all the events $x_1, x_2, x_3, \dots, x_n$ in a system is equal ($n > 1$).

What can you say about the entropy $H(X)$ of that system? (Base of log is 2)

Options:

- $H(X) \leq 1$
- $H(X) = 1$
- $H(X) \geq 1$
- We can't say anything conclusive with the provided information

To analyze the entropy $H(X)$ of a system where the probabilities of all events $x_1, x_2, x_3, \dots, x_n$ are equal, we can use the formula for entropy:

$$H(X) = - \sum_{i=1}^n P(x_i) \log_2 P(x_i)$$

Since the probabilities of all events are equal and let each $P(x_i) = \frac{1}{n}$ (where $n > 1$), we can express the entropy as:

$$H(X) = - \sum_{i=1}^n \frac{1}{n} \log_2 \left(\frac{1}{n} \right)$$

Since there are n identical terms in the sum, this simplifies to:

$$H(X) = -n \cdot \frac{1}{n} \log_2 \left(\frac{1}{n} \right) = -\log_2 \left(\frac{1}{n} \right) = \log_2(n)$$

Since $n > 1$:

$$\log_2(n) > 1$$

Thus, we find that:

$$H(X) = \log_2(n) \geq 1$$

This means that $H(X)$ is greater than or equal to 1.

Conclusion

The correct answer is:

$$H(X) \geq 1.$$

The activation function that is **not a good choice** for the hidden layer, given that the output layer uses a linear activation function, is:

Linear.

Using linear activation in the hidden layer would limit the model to only linear relationships, which cannot capture the non-linear relationship $y = x^2 + 1$.

Question 5

Given the output $y_j = O(a_i)$ and $a_I = [2.5, 3.6, 4.2, 5]$. If O is the softmax function, compute the value of $\hat{y} = [\hat{y}_1, \hat{y}_2, \hat{y}_3, \hat{y}_4]$?

- ☐ $[0.046, 0.139, 0.253, 0.562]$
- ☐ $[0.046, 0.253, 0.562, 0.139]$
- ☐ $[0.253, 0.046, 0.139, 0.562]$
- ☐ $[0.562, 0.046, 0.139, 0.253]$

The formula for the softmax function, which transforms a vector of raw scores (logits) into probabilities, is given by:

$$\hat{y}_i = \frac{e^{a_i}}{\sum_{j=1}^n e^{a_j}}$$

where:

- \hat{y}_i is the softmax output for class i ,
- a_i is the raw score (logit) for class i ,
- n is the number of classes,
- e is the base of the natural logarithm.

Steps:

1. **Exponentiate each element** in the vector of logits $a = [a_1, a_2, \dots, a_n]$:

- Compute e^{a_i} for each i .

2. **Sum the exponentials:**

- Calculate $S = \sum_{j=1}^n e^{a_j}$.

3. **Divide each exponentiated value by the sum:**

- Compute $\hat{y}_i = \frac{e^{a_i}}{S}$ for each i .

This ensures that the output vector $\hat{y} = [\hat{y}_1, \hat{y}_2, \dots, \hat{y}_n]$ will sum to 1 and can be interpreted as probabilities.

Sure! Let's calculate the softmax output step by step for the given input $a_I = [2.5, 3.6, 4.2, 5]$.

Step 1: Compute the Exponentials

Calculate e^{a_i} for each element in a_I :

1. $e^{2.5}$
2. $e^{3.6}$
3. $e^{4.2}$
4. e^5

Using approximate values:

- $e^{2.5} \approx 12.1825$
- $e^{3.6} \approx 36.5982$
- $e^{4.2} \approx 66.6863$
- $e^5 \approx 148.4132$

Step 2: Sum the Exponentials

Add the computed exponentials:

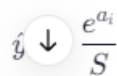
$$S = e^{2.5} + e^{3.6} + e^{4.2} + e^5$$

Calculating S :

$$S \approx 12.1825 + 36.5982 + 66.6863 + 148.4132 \approx 263.8802$$

Step 3: Calculate Softmax Values

Using the softmax formula:


$$\hat{y} \downarrow \frac{e^{a_i}}{S}$$

We will calculate $\hat{y}_1, \hat{y}_2, \hat{y}_3, \hat{y}_4$:

1. For \hat{y}_1 :

$$\hat{y}_1 = \frac{e^{2.5}}{S} = \frac{12.1825}{263.8802} \approx 0.046$$

2. For \hat{y}_2 :

$$\hat{y}_2 = \frac{e^{3.6}}{S} = \frac{36.5982}{263.8802} \approx 0.139$$

3. For \hat{y}_3 :

$$\hat{y}_3 = \frac{e^{4.2}}{S} = \frac{66.6863}{263.8802} \approx 0.253$$

4. For \hat{y}_4 :

$$\hat{y}_4 = \frac{e^5}{S} = \frac{148.4132}{263.8802} \approx 0.562$$

Final Output

Thus, the softmax output is:

$$\hat{y} = [0.046, 0.139, 0.253, 0.562]$$

This matches the correct option from the previous question:

**

$$[0.046, 0.139, 0.253, 0.562]$$

**

Question 6

The information content is high for an event when the probability of the event is:

- ☐ high
- ☐ low
- ☐ 1
- ☐ maximum

The information content (or "surprisal") is high for an event when the probability of the event is:

Answer: low

Explanation:

Information content increases as the probability of the event decreases. This is because rare events carry more information (or surprise) than common events. Mathematically, information content I is often defined as:

$$I = -\log(p)$$

where p is the probability of the event. As p approaches 0 (very low probability), I increases.

The pre-activation at layer i can be best described as the:

- A. weighted sum of all the inputs at layer i
- B. sum of all the inputs at layer i
- C. weighted sum of all the inputs at layer $i + 1$
- D. sum of all the inputs at layer $i + 1$
- E. weighted sum of all the inputs at layer $i - 1$
- F. sum of all the inputs at layer $i - 1$



Sider Fusion

The pre-activation at layer i in a neural network is best described as:

- A. weighted sum of all the inputs at layer i

Explanation:

The pre-activation for a neuron typically involves taking the weighted sum of inputs from the previous layer (layer $i - 1$) plus a bias term. While the wording can vary, the fundamental concept is that it involves the weights applied to the inputs from the previous layer.

Therefore, **option A** is the most accurate description in this context.

Assume you have four inputs to a Feed Forward neural network, the first hidden layer also has four neurons, and there are three output classes. What is the dimension of the weight matrix W_1 between the input layer and the first hidden layer, given that there is only one hidden layer?

- 3×3
- 3×4
- 4×4

- $R4 \times 3R4 \times 3$

Weight Matrix Calculation:

- **Inputs:** 4 (from the input layer)
- **Neurons in the first hidden layer:** 4

The weight matrix W_1 connects the inputs to the first hidden layer. The dimensions of the weight matrix W_1 are determined by the number of neurons in the hidden layer and the number of inputs.

- The number of rows corresponds to the number of neurons in the hidden layer (4).
- The number of columns corresponds to the number of inputs (4).

Thus, the dimension of the weight matrix W_1 is:

Answer: $\mathbb{R}^{4 \times 4}$

- $h_i(x)$ represents the output of the activation layer (or the activation of layer i).
- $a_i(x)$ is typically the pre-activation, which is the weighted sum of the inputs to the neurons at layer i .
- g is the activation function (such as sigmoid, ReLU, etc.) applied to the pre-activation.

So, the correct representation of the activation at layer i is that it is a function g applied to the pre-activation $a_i(x)$.

Other Options:

- **A.** $h_i(x) = b_i + W_i h_{i-1}(x)$: This is more representative of the pre-activation calculation.
- **C.** $h_i(x) = O(a_L)$: This does not describe the output of activation at layer i properly.
- **D.** $h_i(x) = a_i + W_i h_{i-1}(x)$: This also incorrectly describes the process as it mixes the pre-activation with the weights.

Thus, the correct answer is B. $h_i(x) = g(a_i(x))$.