Week-12, Graded

Common Data

Statement

Common data for questions (1) to (4)

Consider a network that has the following architecture for a regression problem:

```
[8, 15, 25, 10, 1]
```

The first layer is the input and last layer is the output. The network shall be denoted by h, input vector by $\boldsymbol{x}$ and the output produced by the network by $\hat{y}$.

Question-1

Statement

How many hidden layers does the network have?

Answer

3

Solution

First and last layers are input and output respectively. The rest are the hidden layers.

Question-2

Statement

How many parameters (weights + biases) does the network have?

Answer

806

Solution

Refer to practice assignment solution

Question-3

Statement

What is the activation function at the output layer?

Options

(a)

Softmax

(b)

Sigmoid

(c)

ReLU

(d)

Linear

Answer

(d)

Solution

Since it is a regression problem, the output activation will be linear.

Question-4

Statement

What is the shape of the weight matrix at layer 2? Note that zero-indexing is used for the layers.

Options

(a)

8×15

(b)

15×25

(c)

25×10

(d)

10×1

Answer

(b)

Solution

Self explanatory

Question-5

Statement

The following is the activation vector output by some hidden layer in a neural network when some input vector is given to it.

$[-0.3, 0.5, 0.9, 0.4, 0, -0.5]$

Which of the following could be the activation function used in this layer?

Options

(a)

Softmax

(b)

Sigmoid

(c)

ReLU

(d)

Tanh

Answer

(d)

Solution

Since each element in the output vector lies in the range $[-1, 1]$, the activation function has to be $\tanh$.

Question-6

Statement

Which of the following techniques could be used to alleviate the problem of overfitting in neural networks? Though the question mentions neural networks, do consider all techniques that we have studied so far. [MSQ]

Options

(a)

L1/L2 regularization

(b)

Dropout

(c)

Increasing the size of the training data

(d)

Adding more hidden layers

Answers

(a), (b), (c)

Solution

Adding more layers will make the network overfit. This is because, increasing the layers will also increase the number of parameters. More the parameters, greater the chance of overfitting.

Question-7

Statement

If $g(z) = \text{ReLU}(z)$, which of the following functions is the derivative of the g with respect to z?

Options

(a)

$$g'(z) = \begin{cases} z, & z \geq 0 \\ 0, & z < 0 \end{cases}$$

(b)

$$g'(z) = \begin{cases} 1, & z \geq 0 \\ 0, & z < 0 \end{cases}$$

Answer

(b)

Solution

The function is defined piece-wise. Follow a piece-wise approach for differentiation.

Question-8

Statement

In a classification problem, for some input vector $\boldsymbol{x}$ to a neural network, $\boldsymbol{z}_l^{(g)}$ is the gradient vector of the loss with respect to the pre-activations at layer l. The activation function at layer l is ReLU:

$$\boldsymbol{z}_l^{(g)} = [0, 0, 0, 1, 2, 3]$$

Which of the following could be the pre-activation vector $\boldsymbol{z}_l$ at layer l for this input?

Options

(a)

$[-0.1, -0.1, -0.1, 5, 3, 2]$

(b)

$[-0.1, -0.1, -0.1, -0.1, 1, 2]$

(c)

$[1, 2, 3, -1, -2, -3]$

(d)

$[-6, -5, -4, -1, -2, -3]$

Answer

(a)

Solution

From the forward pass equations, we have:

$$\boldsymbol{a}_l = g(\boldsymbol{z}_l)$$

The corresponding backward pass equation is:

$$\boldsymbol{z}_l^{(g)} = \boldsymbol{a}_l^{(g)} \odot g'(\boldsymbol{z}_l)$$

Since we have ReLU as the activation function, if the gradient of the loss with respect to the pre-activations at some location is zero, then there are two possibilities:

- the gradient of the loss with respect to the activations is zero at that very location
- the pre-activations at that very location could be less than zero. This is because we are using ReLU activations and the gradient of ReLU at negative locations becomes zero.

As we don't have access to $\boldsymbol{a}_l^{(g)}$ we have to settle for the second scenario. Option-(a) is the best fit here.

Question-9

Statement

What is the output of the following snippet of code?

```python
import numpy as np

theta = np.ones(10)
W = theta[2: 8].reshape(2, 3)
W[0, :] += 1
W[1, :] += 2

print(theta)
```

Options

(a)

```
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
```

(b)

```
[1. 1. 2. 2. 2. 2. 2. 2. 1. 1.]
```

(c)

```
[1. 1. 3. 3. 3. 3. 3. 3. 1. 1.]
```

(d)

```
[1. 1. 2. 2. 2. 3. 3. 3. 1. 1.]
```

Answer

(d)

Solution

`W` is a NumPy view of a portion of the `theta` vector. This is why when `W` is updated in-place, `theta` is also updated. To learn more about NumPy views and copies, refer to this resource.

Question-10

Statement

If W_{ij} is the weight of the edge from neuron i in layer $l-1$ to neuron j in layer l, which of the following statements about the matrix W are true? Neurons in a layer are processed (indexed) from top to bottom. So, the first neuron in a layer is the top-most neuron in that layer.

Options

(a)

The first row of the matrix corresponds to all outgoing connections from the first neuron in layer $l-1$.

(b)

The first row of the matrix corresponds to all incoming connections to the first neuron in layer l.

(c)

The last column of the matrix corresponds to all incoming connections to the last neuron in layer l.

(d)

The last column of the matrix corresponds to all the outgoing connections from the last neuron in layer $l-1$.

Answer

(a), (c)

Solution

Look at the slides which talk about weights. The image there will be of help in understanding why this is the case.