Congratulations! You passed!

Grade received 80% | Latest Submission Grade 80% | To pass 80% or higher

Go to next item

1. What does a neuron compute? **1 / 1 point**

- A neuron computes an activation function followed by a linear function $z = Wx + b$
- A neuron computes the mean of all features before applying the output to an activation function
- A neuron computes a linear function $z = Wx + b$ followed by an activation function
- A neuron computes a function g that scales the input x linearly (Wx + b)

Expand

Correct
Correct, we generally say that the output of a neuron is a = g(Wx + b) where g is the activation function (sigmoid, tanh, ReLU, ...).

2. Suppose that $\hat{y} = 0.5$ and $y = 0$. What is the value of the "Logistic Loss"? Choose the best option. **0 / 1 point**

- 0.693
- $+\infty$
- 0.5
- $$ \mathcal{L}(\hat{y}, y) = - \left(y \, \log \hat{y} + (1-y)\ \log (1 - \hat{y}) \right) $$

Loading [MathJax]/jax/output/CommonHTML/jax.js

Expand

Incorrect
No. This is only the definition of Logistic Loss.

3. Consider the Numpy array x: **1 / 1 point**

$x = np.array([[[1], [2]], [[3], [4]]])$

What is the shape of x?

- (1, 2, 2)
- (4,)
- (2, 2)
- (2,2,1)

Expand

Correct
Yes. This array has two rows and in each row it has 2 arrays of 1x1.

4. Consider the following random arrays a and b, and c: **1 / 1 point**

$a = np.random.randn(2, 3)$ # $a.shape = (2, 3)$

$a = np.random.randn(2, 3)$ # $a.shape = (2, 3)$

$b = np.random.randn(2, 1)$ # $b.shape = (2, 1)$

$c = a + b$

What will be the shape of c?

- The computation cannot happen because the sizes don't match. It's going to be "Error"!
- c.shape = (3, 2)
- c.shape = (2, 1)
- c.shape = (2, 3)

Expand

Correct
Yes! This is broadcasting. b (column vector) is copied 3 times so that it can be summed to each column of a.

5. Consider the two following random arrays a and b: **1 / 1 point**

$a = np.random.randn(4, 3)$ # $a.shape = (4, 3)$

$b = np.random.randn(1, 3)$ # $b.shape = (1, 3)$

$c = a * b$

What will be the shape of c?

- The computation cannot happen because it is not possible to broadcast more than one dimension.
- c.shape = (1, 3)
- The computation cannot happen because the sizes don't match.
- c.shape = (4, 3)

Expand

Correct
Yes. Broadcasting is invoked, so row b is multiplied element-wise with each row of a to create c.

6. Suppose you have n_x input features per example. Recall that $X = [x^{(1)}x^{(2)}...x^{(m)}]$. What is the dimension of X? **1 / 1 point**

- (n_x, m)
- $(1, m)$
- (m, n_x)
- $(m, 1)$

Expand

Correct

7. Consider the following array: **1 / 1 point**

$a = np.array([[2, 1], [1, 3]])$

What is the result of $np.dot(a, a)$?

- $\begin{pmatrix} 4 & 1 \\ 1 & 9 \end{pmatrix}$

- The computation cannot happen because the sizes don't match. It's going to be an "Error"!
- $\begin{pmatrix} 4 & 2 \\ 2 & 6 \end{pmatrix}$
- $\begin{pmatrix} 5 & 5 \end{pmatrix}$

Expand

Correct

Yes, recall that * indicates the element wise multiplication and that np.dot() is the matrix multiplication.

Thus $\begin{pmatrix} (2)(2)+(1)(1) & (2)(1)+(1)(3) \\ (1)(2)+(3)(1) & (1)(1)+(3)(3) \end{pmatrix}$.

8. Consider the following code snippet: **0 / 1 point**

$a.shape = (3, 4)$

$b.shape = (4, 1)$

for i in range(3):

for j in range(4):

c[i][j] = a[i][j]*b[j]

How do you vectorize this?

- c = a*b
- c = a.T*b
- c = np.dot(a,b)
- c = a*b.T

Expand

Incorrect

No. Because of the dimensions of a and b, it is not possible to invoke broadcasting.

9. Consider the following code: **1 / 1 point**

$a = np.random.randn(3, 3)$

$b = np.random.randn(3, 1)$

$c = a * b$

What will be c? (If you're not sure, feel free to run this in python to find out).

- This will invoke broadcasting, so b is copied three times to become (3, 3), and $*$ invokes a matrix multiplication operation of two 3x3 matrices so c.shape will be (3, 3)
- This will invoke broadcasting, so b is copied three times to become (3,3), and

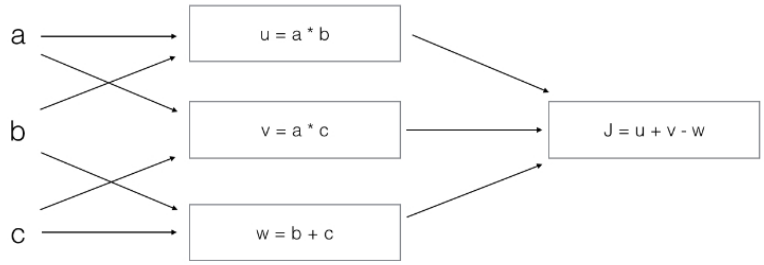
  $*$

  $*$ is an element-wise product so c.shape will be (3, 3)
- It will lead to an error since you cannot use "*" to operate on these two matrices. You need to instead use np.dot(a,b)
- This will multiply a 3x3 matrix a with a 3x1 vector, thus resulting in a 3x1 vector. That is, c.shape = (3, 1).

Expand

Correct

10. Consider the following computation graph. **1 / 1 point**



What is the output J?

- (•) $J = (a - 1) * (b + c)$
- ( ) $J = a * b + b * c + a * c$
- ( ) $J = (b - 1) * (c + a)$
- ( ) $J = (c - 1) * (b + a)$

Expand

Correct

Yes. $J = u + v - w = a * b + a * c - (b + c) = a * (b + c) - (b + c) = (a - 1) * (b + c)$.