Congratulations! You passed!

Grade received 100% **To pass** 80% or higher

Go to next item

Neural Network Basics

Latest Submission Grade 100%

1. In logistic regression given the input $\mathbf{x}$, and parameters $w \in \mathbb{R}^{n_x}$, $b \in \mathbb{R}$, how do we generate the output $\hat{y}$? 1 / 1 point

- $W\mathbf{x} + b$
- $\sigma(W\mathbf{x})$
- $\sigma(W\mathbf{x} + b)$.
- $\tanh(W\mathbf{x} + b)$

Correct
Right, in logistic regression we use a linear function $W\mathbf{x} + b$ followed by the sigmoid function σ, to get an output y, referred to as \hat{y}, such that $0 \leq \hat{y} \leq 1$.

2. Which of these is the "Logistic Loss"? 1 / 1 point

- $\mathcal{L}^{(i)}(\hat{y}^{(i)}, y^{(i)}) = -(y^{(i)} \log(\hat{y}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}))$
- $\mathcal{L}^{(i)}(\hat{y}^{(i)}, y^{(i)}) = | \, y^{(i)} - \hat{y}^{(i)} \, |$
- $\mathcal{L}^{(i)}(\hat{y}^{(i)}, y^{(i)}) = max(0, y^{(i)} - \hat{y}^{(i)})$
- $\mathcal{L}^{(i)}(\hat{y}^{(i)}, y^{(i)}) = | \, y^{(i)} - \hat{y}^{(i)} \, |^2$

Correct
Correct, this is the logistic loss you've seen in lecture!

3. Suppose img is a (32,32,3) array, representing a 32x32 image with 3 color channels red, green and blue. How do you reshape this into a column vector x? 1 / 1 point

- x = img.reshape((32*32,3))
- x = img.reshape((3,32*32))
- x = img.reshape((32*32*3,1))
- x = img.reshape((1,32*32,3))

Correct

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

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

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

$c = a + b$

What will be the shape of c?

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

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?

- [x] c.shape = (4, 3)

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

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

6. Suppose our input batch consists of 8 grayscale images, each of dimension 8x8. We reshape these images into feature column vectors $\mathbf{x}^{j}$. Remember that $X = \begin{bmatrix} \mathbf{x}^{(1)} \mathbf{x}^{(2)} \cdots \mathbf{x}^{(8)} \end{bmatrix}$. What is the dimension of X? 1 / 1 point

- (8, 64)
- (8, 8, 8)
- (64, 8)
- (512, 1)

Correct
Yes. After converting the 8x8 gray scale images to a column vector we get a vector of size 64, thus X has dimension $(64, 8)$.

7. Recall that $np.dot(a, b)$ performs a matrix multiplication on a and b, whereas $a * b$ performs an element-wise multiplication. 1 / 1 point

Consider the two following random arrays a and b:

$a = np.random.randn(12288, 150)$ # $a.shape = (12288, 150)$

$b = np.random.randn(150, 45)$ # $b.shape = (150, 45)$

$c = np.dot(a, b)$

What is the shape of c?

- c.shape = (150,150)
- The computation cannot happen because the sizes don't match. It's going to be "Error"!
- c.shape = (12288, 150)
- c.shape = (12288, 45)

Correct
Correct, remember that a np.dot(a, b) has shape (number of rows of a, number of columns of b). The sizes match because :

"number of columns of a = 150 = number of rows of b"

8. Consider the following code snippet: 1 / 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.T
- c = a.T + b.T
- c = a + b
- c = a.T + b

Correct

9. Consider the following code: 1 / 1 point

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

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

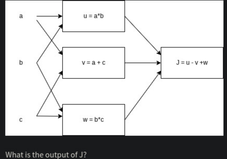
$c = a * b$

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

- 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).
- 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)
- 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)

Correct

10. Consider the following computational graph. 1 / 1 point



What is the output of J?

- $(a - 1)(b + c)$
- $(a + c)(b - 1)$
- $(c - 1)(a + c)$
- $ab + bc + ac$

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