**Neural Network Basics**

**Latest Submission Grade 80%**

**1.**

**Question 1**

In logistic regression given the input \mathbf{x}**x**, and parameters w \in \mathbb{R}^{n\_x}*w*∈R*nx*​, b \in \mathbb{R}*b*∈R, how do we generate the output \hat{y}*y*^​?

**1 / 1 point**

Expand

**Correct**

Right, in logistic regression we use a linear function W**x**+b followed by the sigmoid function σ, to get an output y, referred to as ŷ, such that 0<ŷ<1.

**2.**

**Question 2**

Which of these is the "Logistic Loss"?

**1 / 1 point**

Expand

**Correct**

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

**3.**

**Question 3**

Consider the Numpy array x*x*:

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

What is the shape of x?

**1 / 1 point**

Expand

**Correct**

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

**4.**

**Question 4**

Consider the following random arrays a*a* and b*b*, and c*c*:

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

b = np.random.randn(1, 4)*b*=*np*.*random*.*randn*(1,4) # b.shape = (1, 4)*b*.*shape*=(1,4)

c = a + b*c*=*a*+*b*

What will be the shape of c*c*?

**1 / 1 point**

Expand

**Correct**

Yes. Broadcasting is used, so row b is copied 3 times so it can be summed to each row of a.

**5.**

**Question 5**

Consider the two following random arrays a*a* and b*b*:

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

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

c = a\*b*c*=*a*∗*b*

What will be the shape of c*c*?

**0 / 1 point**

Expand

**Incorrect**

You didn't select all the correct answers

**6.**

**Question 6**

Suppose you have n\_x*nx*​ input features per example. Recall that X = [x^{(1)} x^{(2)} ... x^{(m)}]*X*=[*x*(1)*x*(2)...*x*(*m*)]. What is the dimension of X?

**0 / 1 point**

Expand

**Incorrect**

**7.**

**Question 7**

Consider the following array:

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

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

**1 / 1 point**

Expand

**Correct**

Yes, recall that \* indicates the element wise multiplication and that np.dot() is the matrix multiplication. Thus ((2)(2)+(1)(1)(1)(2)+(3)(1)(2)(1)+(1)(3)(1)(1)+(3)(3)).

**8.**

**Question 8**

Consider the following code snippet:

a.shape = (3, 4)*a*.*shape*=(3,4)

b.shape = (4, 1)*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?

**1 / 1 point**

Expand

**Correct**

Yes. b.T gives a column vector with shape (1, 4). The result of c is equivalent to broadcasting a\*b.T.

**9.**

**Question 9**

Consider the code snippet:

a.shape = (3, 3)*a*.*shape*=(3,3)

b.shape = (3, 3)*b*.*shape*=(3,3)

c = a\*\*2 + b.T\*\*2*c*=*a*∗∗2+*b*.*T*∗∗2

Which of the following gives an equivalent output for c*c*?

**1 / 1 point**

Expand

**Correct**

Yes. This code squares each entry of a and adds it to the transpose of b square.

**10.**

**Question 10**

Consider the following computational graph.

Diagram

Description automatically generated

What is the output of J?

**1 / 1 point**

Expand

**Correct**

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