Congratulations! You passed!

Grade received 90% | Latest Submission Grade 90% | To pass 80% or higher

Go to next item

1. In logistic regression given $\mathbf{x}$ and parameters $w \in \mathbb{R}^{n_x}, b \in \mathbb{R}$. Which of the following best expresses what we want $\hat{y}$ to tell us? **0 / 1 point**

- $\sigma(W\mathbf{x})$
- $\sigma(W\mathbf{x}+b)$
- $P(y=1|\mathbf{x})$
- $P(y=\hat{y}|\mathbf{x})$

Expand

Incorrect
No. We want the output $\hat{y}$ to tell us the probability that $y=1$ given x.

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

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

Expand

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

Expand

Correct

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

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

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

$c = a + b$

What will be the shape of c?

- c.shape = (2, 1)
- c.shape = (2, 3, 3)
- The computation cannot happen because it is not possible to broadcast more than one dimension
- c.shape = (3,3)

Expand

Correct
Yes. It is not possible to broadcast together a and b. In this case there is no way to generate copies of one of the arrays to match the size of the other.

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

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

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

$c = a * b$

What will be the shape of c?

- c.shape = (4, 3)
- c.shape = (3, 3)
- c.shape = (4,2)
- The computation cannot happen because the sizes don't match. It's going to be "Error"!

Expand

Correct
Indeed! In numpy the "*" operator indicates element-wise multiplication. It is different from "np.dot()". If you would try "c = np.dot(a,b)" you would get c.shape = (4, 2).

6. 1 / 1 point

Suppose you have n_x input features per example. If we decide to use row vectors $\mathbf{x}_j$ for the features and $X = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \vdots \\ \mathbf{x}_m \end{bmatrix}$.

What is the dimension of X?

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

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

Expand

Correct
Yes. Each $\mathbf{x}_j$ has dimension $1 \times n_x$, X is built stacking all rows together into a $m \times n_x$ array.

7. Consider the following array: 1 / 1 point

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

What is the result of $a * a$?

- The computation cannot happen because the sizes don't match. It's going to be an "Error"!
- $\begin{pmatrix} 4 & 1 \\ 1 & 9 \end{pmatrix}$
- $\begin{pmatrix} 5 & 5 \\ 5 & 10 \end{pmatrix}$

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

Expand

Correct
Yes, recall that * indicates element-wise multiplication.

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

$a.shape = (4, 3)$

$b.shape = (4, 1)$

for i in range(3):

for j in range(4):

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

How do you vectorize this?

- c = a + b
- c = a.T + b.T
- c = a.T + b
- c = a + b.T

Expand

Correct
Yes. a[j][i] being used for a[i][j] indicates we are using a.T, and the element in the row j is used in the column j thus we are using b.T.

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

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

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

$c = a * b$

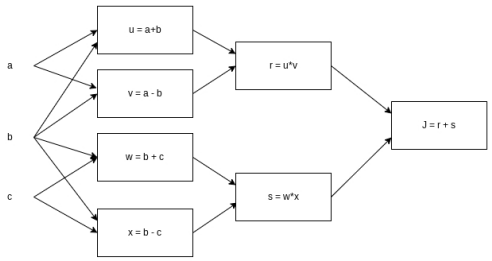
What will be c? (If you're not sure, feel free to run this in python to find out).

- (●) This will invoke broadcasting, so b is copied three times to become (3,3), and $*$ 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)
- ( ) This will multiply a 3x3 matrix a with a 3x1 vector, thus resulting in a 3x1 vector. That is, c.shape = (3,1).
- ( ) 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)

Expand

Correct

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



What is the output of J?

- ( ) $a^2 - b^2$
- ( ) $(a - b) * (a - c)$
- (●) $a^2 - c^2$
- ( ) $a^2 + b^2 - c^2$

Expand

Correct

Yes.

$J = r + s = u * v + w * x = (a + b) * (a - b) + (b + c) * (b - c) = a^2 - b^2 + b^2 - c^2 = a^2 - c^2$

.