**[Skip to Main Content](https://www.coursera.org/learn/neural-networks-deep-learning/exam/9uiEN/neural-network-basics/view-attempt" \l "main)**

[[](https://www.coursera.org/)](https://www.coursera.org/)

Explore

Top of Form



Bottom of Form

* **1**
* Vijay Misra

1. [Neural Networks and Deep Learning](https://www.coursera.org/learn/neural-networks-deep-learning/home/welcome)
2. [Week 2](https://www.coursera.org/learn/neural-networks-deep-learning/home/week/2)
3. Neural Network Basics

[**Previous**](https://www.coursera.org/learn/neural-networks-deep-learning/supplement/wa3Si/lectures-in-pdf)[**Next**](https://www.coursera.org/learn/neural-networks-deep-learning/supplement/muGtL/deep-learning-honor-code)

* **Logistic Regression as a Neural Network**
* **Python and Vectorization**
* **Lecture Notes (Optional)**
* **Quiz**

**[Quiz:](https://www.coursera.org/learn/neural-networks-deep-learning/exam/9uiEN/neural-network-basics)**[Neural Network Basics](https://www.coursera.org/learn/neural-networks-deep-learning/exam/9uiEN/neural-network-basics)

[10 questions](https://www.coursera.org/learn/neural-networks-deep-learning/exam/9uiEN/neural-network-basics)

* **Programming Assignments**
* **Heroes of Deep Learning (Optional)**

**QUIZQuiz • 20 MIN20 minutes**

**Neural Network Basics**

**Submit your assignment**

**DUE DATE**May 9, 11:59 PM PDTMay 9, 11:59 PM PDT

Try again

**Receive grade**

**TO PASS**80% or higher

**Grade**

90%

View Feedback

We keep your highest score

Neural Network Basics

Graded Quiz • 20 min

**Due** May 9, 11:59 PM PDT

**Congratulations! You passed!**

**TO PASS**80% or higher

Keep Learning

**GRADE**

90%

**Neural Network Basics**

**LATEST SUBMISSION GRADE**

90%

1.

Question 1

What does a neuron compute?

**1 / 1 point**



A neuron computes the mean of all features before applying the output to an activation function



A neuron computes a function g that scales the input x linearly (Wx + b)



A neuron computes a linear function (z = Wx + b) followed by an activation function



A neuron computes an activation function followed by a linear function (z = Wx + b)

**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.

Question 2

Which of these is the "Logistic Loss"?

**1 / 1 point**



\mathcal{L}^{(i)}(\hat{y}^{(i)}, y^{(i)}) = max(0, y^{(i)} - \hat{y}^{(i)})L(*i*)(*y*^​(*i*),*y*(*i*))=*max*(0,*y*(*i*)−*y*^​(*i*))



\mathcal{L}^{(i)}(\hat{y}^{(i)}, y^{(i)}) = \mid y^{(i)} - \hat{y}^{(i)} \midL(*i*)(*y*^​(*i*),*y*(*i*))=∣*y*(*i*)−*y*^​(*i*)∣



\mathcal{L}^{(i)}(\hat{y}^{(i)}, y^{(i)}) = \mid y^{(i)} - \hat{y}^{(i)} \mid^{2}L(*i*)(*y*^​(*i*),*y*(*i*))=∣*y*(*i*)−*y*^​(*i*)∣2



\mathcal{L}^{(i)}(\hat{y}^{(i)}, y^{(i)}) = -( y^{(i)}\log(\hat{y}^{(i)}) + (1- y^{(i)})\log(1-\hat{y}^{(i)})L(*i*)(*y*^​(*i*),*y*(*i*))=−(*y*(*i*)log(*y*^​(*i*))+(1−*y*(*i*))log(1−*y*^​(*i*)))

**Correct**

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

3.

Question 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?

**1 / 1 point**



x = img.reshape((3,32\*32))



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



x = img.reshape((32\*32,3))



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

**Correct**

4.

Question 4

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

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"?

**1 / 1 point**



c.shape = (2, 3)



c.shape = (3, 2)



c.shape = (2, 1)



The computation cannot happen because the sizes don't match. It's going to be "Error"!

**Correct**

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

5.

Question 5

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

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"?

**1 / 1 point**



c.shape = (4, 3)



c.shape = (4,2)



c.shape = (3, 3)



The computation cannot happen because the sizes don't match. It's going to be "Error"!

**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.

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?

**1 / 1 point**



(m,n\_x)(*m*,*nx*​)



(1,m)(1,*m*)



(n\_x, m)(*nx*​,*m*)



(m,1)(*m*,1)

**Correct**

7.

Question 7

Recall that "np.dot(a,b)" performs a matrix multiplication on a and b, whereas "a\*b" performs an element-wise multiplication.

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?

**0 / 1 point**



The computation cannot happen because the sizes don't match. It's going to be "Error"!



c.shape = (12288, 150)



c.shape = (12288, 45)



c.shape = (150,150)

**Incorrect**

No, 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.

Question 8

Consider the following code snippet:

# 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?

**1 / 1 point**



c = a + b



c = a.T + b



c = a.T + b.T



c = a + b.T

**Correct**

9.

Question 9

Consider the following code:

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).

**1 / 1 point**



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 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 multiply a 3x3 matrix a with a 3x1 vector, thus resulting in a 3x1 vector. That is, c.shape = (3,1).



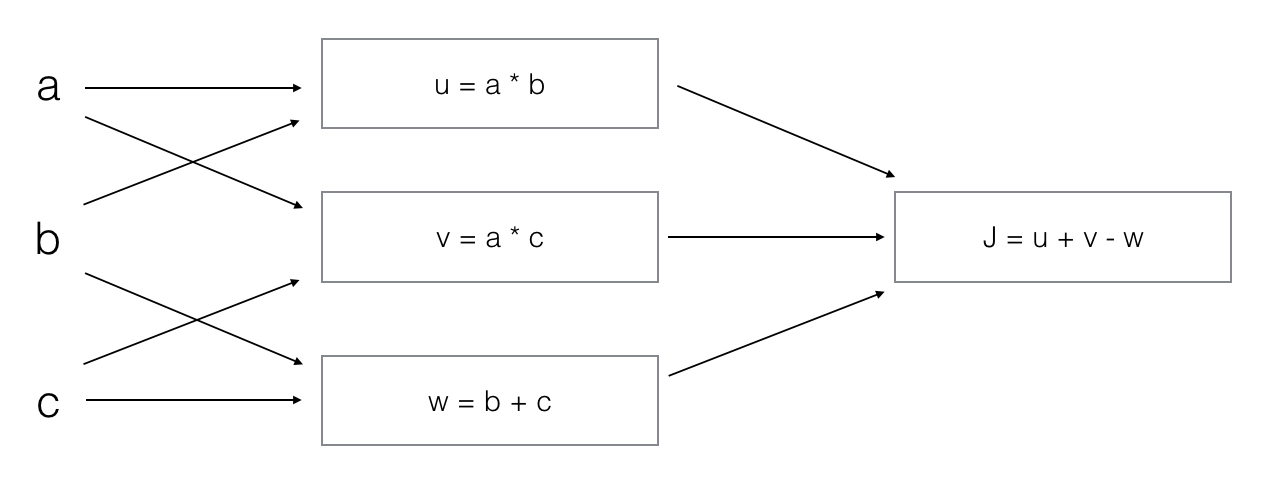
This will invoke broadcasting, so b is copied three times to become (3, 3), and \*∗ invokes a matrix multiplication operation of two 3x3 matrices so c.shape will be (3, 3)

**Correct**

10.

Question 10

Consider the following computation graph.



What is the output J?

**1 / 1 point**



J = (b - 1) \* (c + a)



J = (c - 1)\*(b + a)



J = a\*b + b\*c + a\*c



J = (a - 1) \* (b + c)

**Correct**

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