**Congratulations! You passed!**

**TO PASS** 80% or higher

Keep Learning

**GRADE**
90%

# Shallow Neural Networks

**LATEST SUBMISSION GRADE**

90%

1. Which of the following are true? (Check all that apply.)

**1 / 1 point**

- [ ] $a_4^{[2]}$ is the activation output of the $2^{nd}$ layer for the $4^{th}$ training example
- [ ] $a^{[2](12)}$ denotes activation vector of the $12^{th}$ layer on the $2^{nd}$ training example.
- [x] $a_4^{[2]}$ is the activation output by the $4^{th}$ neuron of the $2^{nd}$ layer

**Correct**

- [ ] $X$ is a matrix in which each row is one training example.
- [x] $a^{[2]}$ denotes the activation vector of the $2^{nd}$ layer.

**Correct**

- [x] $X$ is a matrix in which each column is one training example.

**Correct**

- [x] $a^{[2](12)}$ denotes the activation vector of the $2^{nd}$ layer for the $12^{th}$ training example

$a^{[2](12)}$ denotes the activation vector of the $2^{nd}$ layer for the $12^{th}$ training example.

**Correct**

2. The tanh activation usually works better than sigmoid activation function for hidden units because the mean of its output is closer to zero, and so it centers the data better for the next layer. True/False? — **1 / 1 point**

◉ True

○ False

**Correct**

Yes. As seen in lecture the output of the tanh is between -1 and 1, it thus centers the data which makes the learning simpler for the next layer.

3. Which of these is a correct vectorized implementation of forward propagation for layer $l$, where $1 \leq l \leq L$? — **1 / 1 point**

◉
- $Z^{[l]} = W^{[l]} A^{[l-1]} + b^{[l]}$
- $A^{[l]} = g^{[l]}(Z^{[l]})$

○
- $Z^{[l]} = W^{[l]} A^{[l]} + b^{[l]}$
- $A^{[l+1]} = g^{[l+1]}(Z^{[l]})$

○
- $Z^{[l]} = W^{[l-1]} A^{[l]} + b^{[l-1]}$
- $A^{[l]} = g^{[l]}(Z^{[l]})$

○
- $Z^{[l]} = W^{[l]} A^{[l]} + b^{[l]}$
- $A^{[l+1]} = g^{[l]}(Z^{[l]})$

**Correct**

4. You are building a binary classifier for recognizing cucumbers (y=1) vs. watermelons (y=0). Which one of these activation functions would you recommend using for the output layer? — **1 / 1 point**

○ ReLU

○ Leaky ReLU

- sigmoid
- tanh

**Correct**

Yes. Sigmoid outputs a value between 0 and 1 which makes it a very good choice for binary classification. You can classify as 0 if the output is less than 0.5 and classify as 1 if the output is more than 0.5. It can be done with tanh as well but it is less convenient as the output is between -1 and 1.

5. Consider the following code:

**1 / 1 point**

```
A = np.random.randn(4,3)
B = np.sum(A, axis = 1, keepdims = True)
```

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

- (1, 3)
- (4, 1)
- (, 3)
- (4, )

**Correct**

Yes, we use (keepdims = True) to make sure that A.shape is (4,1) and not (4, ). It makes our code more rigorous.

6. Suppose you have built a neural network. You decide to initialize the weights and biases to be zero. Which of the following statements is true?

**1 / 1 point**

- Each neuron in the first hidden layer will perform the same computation. So even after multiple iterations of gradient descent each neuron in the layer will be computing the same thing as other neurons.
- Each neuron in the first hidden layer will perform the same computation in the first iteration. But after one iteration of gradient descent they will learn to compute different things because we have "broken symmetry".
- Each neuron in the first hidden layer will compute the same thing, but neurons in different layers will compute

Each neuron in the first hidden layer will compute the same thing, but neurons in different layers will compute different things, thus we have accomplished "symmetry breaking" as described in lecture.

The first hidden layer's neurons will perform different computations from each other even in the first iteration; their parameters will thus keep evolving in their own way.

**Correct**

7. Logistic regression's weights w should be initialized randomly rather than to all zeros, because if you initialize to all zeros, then logistic regression will fail to learn a useful decision boundary because it will fail to "break symmetry", True/False?

1 / 1 point

- True
- (selected) False

**Correct**

Yes, Logistic Regression doesn't have a hidden layer. If you initialize the weights to zeros, the first example x fed in the logistic regression will output zero but the derivatives of the Logistic Regression depend on the input x (because there's no hidden layer) which is not zero. So at the second iteration, the weights values follow x's distribution and are different from each other if x is not a constant vector.

8. You have built a network using the tanh activation for all the hidden units. You initialize the weights to relative large values, using np.random.randn(..,..)*1000. What will happen?

1 / 1 point

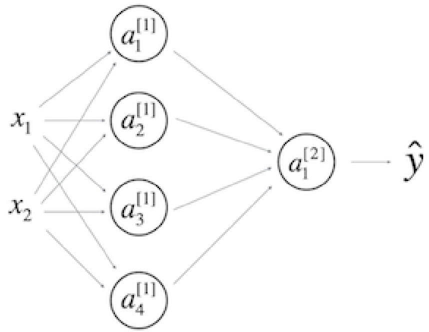
- (selected) This will cause the inputs of the tanh to also be very large, thus causing gradients to be close to zero. The optimization algorithm will thus become slow.
- This will cause the inputs of the tanh to also be very large, thus causing gradients to also become large. You therefore have to set $\alpha$ to be very small to prevent divergence; this will slow down learning.
- This will cause the inputs of the tanh to also be very large, causing the units to be "highly activated" and thus speed up learning compared to if the weights had to start from small values.
- It doesn't matter. So long as you initialize the weights randomly gradient descent is not affected by whether the weights are large or small.

**Correct**

Yes. tanh becomes flat for large values, this leads its gradient to be close to zero. This slows down the optimization algorithm.

9. Consider the following 1 hidden layer neural network:

1 / 1 point




Which of the following statements are True? (Check all that apply).

- [ ] $W^{[1]}$ will have shape (2, 4)
- [x] $b^{[1]}$ will have shape (4, 1)

✓ **Correct**

- [x] $W^{[1]}$ will have shape (4, 2)

✓ **Correct**

- [ ] $b^{[1]}$ will have shape (2, 1)
- [x] $W^{[2]}$ will have shape (1, 4)

✓ **Correct**

- [ ] $b^{[2]}$ will have shape (4, 1)
- [ ] $W^{[2]}$ will have shape (4, 1)
- [x] $b^{[2]}$ will have shape (1, 1)

Correct

10. In the same network as the previous question, what are the dimensions of $Z^{[1]}$ and $A^{[1]}$?

0 / 1 point

- ○ $Z^{[1]}$ and $A^{[1]}$ are (4,m)
- ◉ $Z^{[1]}$ and $A^{[1]}$ are (4,1)
- ○ $Z^{[1]}$ and $A^{[1]}$ are (4,2)
- ○ $Z^{[1]}$ and $A^{[1]}$ are (1,4)

**Incorrect**

Remember that $Z^{[1]}$ and $A^{[1]}$ are quantities computed over a batch of training examples, not only 1.