Shallow Neural Networks

✓ **Congratulations! You passed!**

Grade received 100%

Latest Submission Grade 100%

To pass 80% or higher

Retake the assignment in **23h 56m**

Go to next item

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

1 / 1 point

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

☐ X is a matrix in which each row is one training example.

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

✓ **Correct**

☑ X is a matrix in which each column is one training example.

✓ **Correct**

☐ $a^{[2]}_4$ is the activation output of the 2^{nd} layer for the 4^{th} training example

☑ $a^{[2]}$ denotes the activation vector of the 2^{nd} layer.

✓ **Correct**

☑ $a^{[2]}_4$ is the activation output by the 4^{th} neuron of the 2^{nd} layer

✓ **Correct**

Correct
Great, you got all the right answers.

2. In which of the following cases is the linear (identity) activation function most likely used?

1 / 1 point

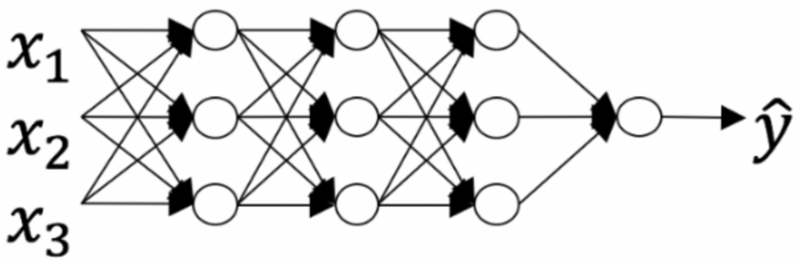
- ○ As activation function in the hidden layers.
- ○ The linear activation function is never used.
- ○ For binary classification problems.
- ● When working with regression problems.

Expand

Correct
Yes. In problems such as predicting the price of a house it makes sense to use the linear activation function as output.

3. Which of the following represents the activation output of the second neuron of the third layer applied to the fourth example?

1 / 1 point



[✓ $a_2^{[3](4)}$]

$a_2^{[4](3)}$

$a_4^{[3](2)}$

$a_3^{[4]2}$

Expand

Correct
Yes. The superscript in brackets indicates the layer number, the superscript in parenthesis represents the number of examples, and the subscript the number of the neuron.

4. The use of the ReLU activation function is becoming more rare because the ReLU function has no derivative for $c = 0$. True/False?

1 / 1 point

[✓ False]

True

Expand

Correct
Yes. Although the ReLU function has no derivative at $c = 0$ this rarely causes any problems in practice. Moreover it has become the default activation function in many cases, as explained in the lectures.

5. Consider the following code:

1 / 1 point

#+begin_src python

x = np.random.rand(3, 2)

y = np.sum(x, axis=0, keepdims=True)

#+end_src

What will be y.shape?

- [✓ (1, 2)]
- (2,)
- (3,)
- (3, 1)

Expand

✓ **Correct**
Yes. By choosing the axis=0 the sum is computed over each column of the array, thus the resulting array is a row vector with 2 entries. Since the option keepdims=True is used the first dimension is kept, thus (1, 2).

6. Suppose you have built a neural network with one hidden layer and tanh as activation function for the hidden layer. You decide to initialize the weights to small random numbers and the biases to zero. The first hidden layer's neurons will perform different computations from each other even in the first iteration. True/False?

1 / 1 point

- False No. Since the weights are most likely different, each neuron will do a different computation.
- [✓ True Yes. Since the weights are most likely different, each neuron will do a different computation.]

Expand

✓ **Correct**

7. Logistic regression's weights 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
- [✓ False]

Expand

✓ **Correct**
Yes, Logistic Regression doesn't have a hidden layer. If you initialize the weights to zeros, the first example x fed into 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 relatively large values, using np.random.randn(..,..)*1000. What will happen?

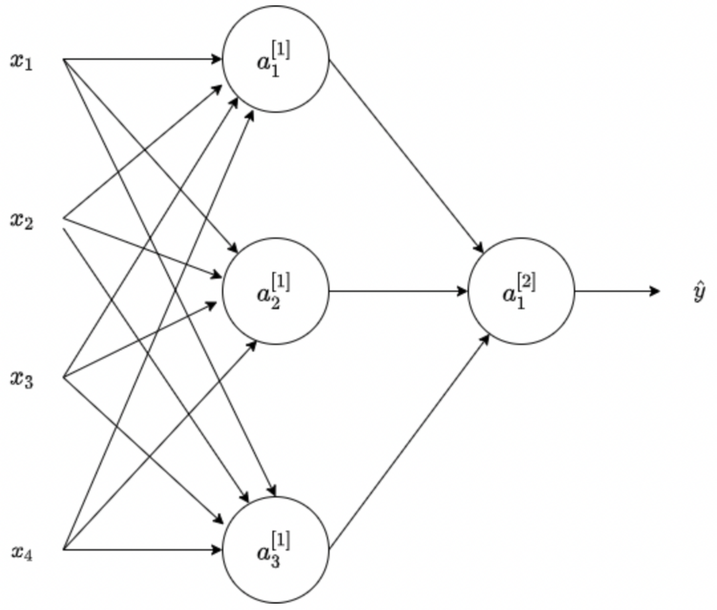
1 / 1 point

- ○ 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.
- [✓ 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.]
- ○ So long as you initialize the weights randomly gradient descent is not affected by whether the weights are large or small.
- ○ 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 α to a very small value to prevent divergence; this will slow down learning.

✓ 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 (4, 3).
- ☑ *E: $b^{[2]}$ will have shape (1,1)

✓ **Correct**
Yes. $b^{[k]}$ is a column vector and has the same number of rows as neurons in the k-th layer.

[✓] $b^{[1]}$ will have shape (3, 1).

✓ **Correct**
Yes. $b^{[k]}$ is a column vector and has the same number of rows as neurons in the k-th layer.

[ ] $b^{[2]}$ will have shape (3, 1)

[✓] $W^{[1]}$ will have shape (3, 4).

✓ **Correct**
Yes. The number of rows in $W^{[k]}$ is the number of neurons in the k-th layer and the number of columns is the number of inputs of the layer.

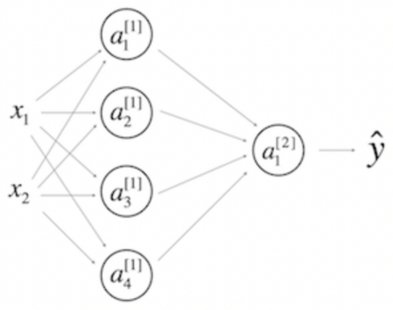
[ ] $b^{[1]}$ will have shape (1, 3)

Expand

✓ **Correct**
Great, you got all the right answers.

10. What are the dimensions of $Z^{[1]}$ and $A^{[1]}$?

1 / 1 point



○ $Z^{[1]}$ and $A^{[1]}$ are (4,2)

● $Z^{[1]}$ and $A^{[1]}$ are (4,m)

○ $Z^{[1]}$ and $A^{[1]}$ are (1,4)

○ $Z^{[1]}$ and $A^{[1]}$ are (4,1)

Expand

✓ **Correct**