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

  ✓ **Correct**

- [x] $a_4^{[2]}$ is the activation output by the 4^{th} neuron of the 2^{nd} layer

  ✓ **Correct**

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

  ✓ **Correct**

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

  ✓ **Correct**

Expand

Correct
Great, you got all the right answers.

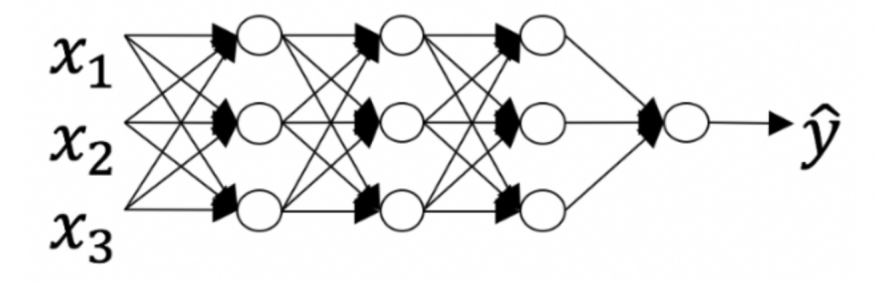
2. The tanh activation is not always better than sigmoid activation function for hidden units because the mean of its output is closer to zero, and so it centers the data, making learning complex for the next layer. True/False? **1 / 1 point**

- (•) False
- ( ) True

Expand

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 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. When building a binary classifier for recognizing cats (y=1) vs raccoons (y=0). Is better to use the sigmoid function as activation function for the hidden layers. True/False **1 / 1 point**

- True
- False

Expand

Correct
Yes. Using tanh almost always works better than the sigmoid function for hidden layers.

5. Consider the following code: **0 / 1 point**

#+begin_src python

x = np.random.rand(4, 5)

y = np.sum(x, axis=1)

#+end_src

What will be y.shape?

- (4, 1)
- (1, 5)
- (4,)
- (5,)

Expand

Incorrect
No. By using axis=1 the sum is computed over each row of the array, thus the resulting array is a column vector with 4 entries. Since the option keepdims was not used the array doesn't keep the second dimension.

6. Suppose you have built a neural network with one hidden layer and tanh as activation function for the hidden layers. Which of the following is a best option to initialize the weights? **0 / 1 point**

- Initialize all weights to a single number chosen randomly.
- Initialize the weights to large random numbers.
- Initialize the weights to small random numbers.
- Initialize all weights to 0.

Expand

Incorrect
No. In this case all neurons in the 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.

7. A single output and single layer neural network that uses the sigmoid function as activation is equivalent to the logistic regression. True/False **1 / 1 point**

- False
- True

Expand

Correct
Yes. The logistic regression model can be expressed by $$ŷ = σ\left(W \, x + b \right)$$. This is the same as $$a^{[1]} = σ(W^{[1]}\, X + b)$$.

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**

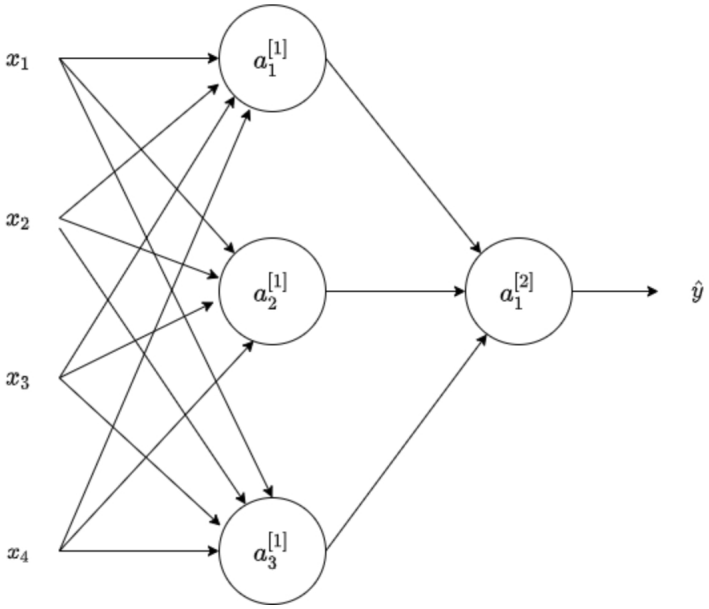
np.random.randn(...,...) * 1000. What will happen?

- 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, 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 also become large. You therefore have to set α to a very small value to prevent divergence; this will slow down learning.
- 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.

Expand

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).

- [x] $W^{[1]}$ 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.

- [ ] $b^{[2]}$ will have shape (3, 1)
- [ ] $W^{[1]}$ will have shape (4, 3).
- [x] *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 (1, 3)
- [x] $$b^{[1]}$$ 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.

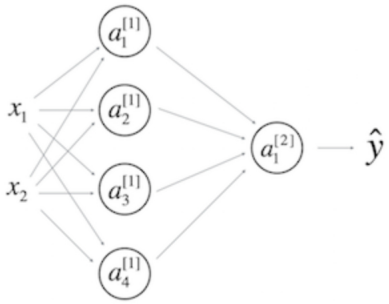
Expand

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

Expand

Correct