Neural Networks and Deep Learning

Week 2: Neural Network Basics

1 Version 1: In logistic regression given the input X, and parameters $w \in \mathbb{R}^{n_x}$, how do we generate the output $\hat{y}$?

Answer: $\sigma(Wx + b)$.

Comment: In logistic regression we use a linear function Wx_b followed by the sigmoid function σ, to get an output y, referred to as $\hat{y}$, such that $0 < \hat{y} < 1$.

1 Version 2: What does a neuron compute?

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

Comment: We generally say that the output of a neuron is $a = g(Wx + b)$ where g is the activation function (sigmoid, tanh, ReLU,...).

2 Version 1: Suppose that $\hat{y} = 0.5$, and $y = 0$. What is the value of the logistic loss? choose the best option.

Answer: 0.693.

Comment: $L(0.5, 0) = -(0 * log(0.5) + 1 * log(0.5)) = 0.693$.

2 Version 2: Which of these is the logistic loss?

Answer: $L^{(i)}(\hat{y}^{(i)}, y^{(i)}) = -(y^{(i)}log(\hat{y}^{(i)}) + (1 - y^{(i)})log(1 - \hat{y}^{(i)}))$

Comment: See lecture notes.

3. Version 1: Suppose x is a (8,1) array. Which of the following is a valid reshape?

Answer: x.reshape(2,2,2).

Comment: Reshape it into a 3 dimensional array. The new shape must have the same number of elements as the original array $2 \times 2 \times 2 = 8$.

3 Version 2: Suppose img is a (32,32,3) array, representing a 32×32 image with 3 color channels red, green, and blue. How do you reshape this into a column vector?

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

4 Consider the following random arrays a and b, and c:

- 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?

Answer: c.shape = (2,3).

Comment: The column vector b is copied 3 times so that it can be summed to each column of a.

5 Consider the 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?

Answer: The computation cannot happen because the sizes don't match. It's going to be error.

Comment: In numpy the * operator indicates elements 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 Suppose you have n_x input features per example. Recall that $X = [x^{(1)\cdots x^{(m)}}]$. What is the dimension of X.

Answer: (n_x, m).

Comment: See lecture notes.

7 Version 1: Consider the following array:

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

What is the result of np.dot(a,a)?

Answer: $\begin{bmatrix} 5 & 5 \\ 5 & 10 \end{bmatrix}$

Comment: $\begin{bmatrix} 4+1 & 2+3 \\ 1+3 & 1+3 \end{bmatrix}$

7 Version 2: Recall that no.dot(a,b) performs a matrix multiplication on a and b, whereas a*b performs an element wise multiplication. Consider the two following arrays a and b.

- a = np.random.randn(52288,158) # a.shaple = (12288,150)
- b = np.random.randn(150,45) # a.shaple = (150,45)
- c = np.dot(a,b)

What is the shape of c?

Answer: c.shape = (12288,45)

Comment: np.dot(a,b) has shape (number of rows of a, number of columns of b).

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?

Answer: c = a*b.T.

Comment: b.T gives a column vector with shape (1,4). The result of c is equivalent to broadcasting a*b.T.

9 Version 1: Consider the code snippet:

```
a.shape = (3,3)
b.shape = (3,3)
c = a ** 2+ b.T ** 2
```

Which of the following gives an equivalent output for c?

Answer:

```
for i in range(3):
for j in range(3):
c[i][j] = a[i][j]**2 + b[i][j]**2
```

Comment: This code squares each entry of a and adds it to the transpose of b square.

9 Version 2: Consider the 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).

Answer: 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).

10 Version 1: Consider the following computational graph.

- a, b, c.
- u = a+b, v = a-b, w = b+c, x = b-c.
- r = u*v, s = w*x.
- J = r+s

What is the output of J?

Answer: $a^2 - c^2$.

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

10 Version 2: Consider the following computational graph.

- a, b, c.
- u = a*b, v = a*b, w = b+c.
- J = u+v-w

What is the output of J?

Answer: $(a - 1) * (b + c)$.

Comment: $J = u + v - w = a * b + a * c - (b + c) = a * (b + c) - (b + c) = (a - 1) * (b + c)$.