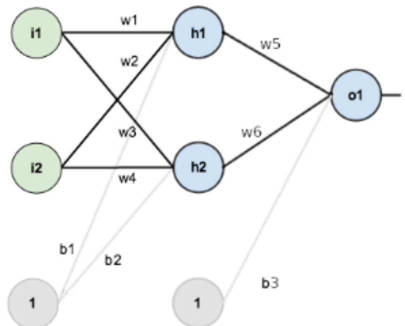
Given the following Neural Network and input values, compute the output value. Use the sigmoid activation function. Show your work for partial credit.



Input	Level 1 Weights	Level 1 Weights	Bias
i1 = .8	W1 = .2	W5 = .48	B1 = .23
i2 = .3	W2 = .35	W6 = .34	B2 = .41
	W3 = .15		B3 = .5
	W4 = .6		

The work for partial credit is shown in the attached Excel file, but here are screenshots for simplicity.

Answer:

o1 output value is:	**0.7361693**

Work for partial credit (actual calculations were done in the Excel file that was included in the submission

	J	K	L	M	N	O	P	Q
1								
2	Input	Level 1 Weights	Level 1 Weights	Bias	Learning Rate	Target Value		
3	i1 = .8	W1.1 = .2	W5 = .48	B1 = .23	0.1	0		
4	i2 = .3	W2.1 = .35	W6 = .34	B2 = .41				
5		W1.2 = .15		B3 = .5				
6		W2.2 = .6						
7								

Forward Pass

$$h_n = a\left(\sum_{k=1}^{3} i_k * W_{k,n} + b_n\right)$$

	J	K	L	M	N	O	P	Q
14	From inputs to hidden layer 1					Sum and add bias	Then sigmoid	
15		I x	w	=	`+b		"a" values	
16	w1: i1.h1	0.8	0.2	0.16	0.23	0.495	**0.6212836**	:h1_output
17	w2: i2.h1	0.3	0.35	0.105				
18								
19	w3: i1.h2	0.8	0.15	0.12	0.41	0.71	**0.67040116**	:h2_output
20	w4: i2.h2	0.3	0.6	0.18				
21								
22	From hidden layer 1 to output 1					Sum and add bias	Then sigmoid	
23		I x	w	=	`+b			
24	w5: h1.o1	0.621283595	0.48	0.29821613	0.5	1.02615252	**0.7361693**	:o1_output
25	w6: h2.o1	0.67040116	0.34	0.22793639				
26								
27	o1 output value is:		**0.7361693**					

Using the output from the previous question, apply backpropagation assuming the target value is 0 and the learning rate is .1. What are the new weights. Show your work for partial credit.

Answer:

	Updated Weights		
	From:	To:	
w1	0.2	0.19870814	
w2	0.35	0.34951555	
w3	0.15	0.14914065	
w4	0.6	0.59967774	
From hidden layer 1 to output 1			
	From:	To:	
w5	0.48	0.47111678	
w6	0.34	0.33041448	
	Updated Biases		
	From:	To:	
b1	0.23	0.22838517	
b2	0.41	0.40892581	
b3	0.5	0.48570182	

Work for partial credit (actual calculations were done in the Excel file that was included in the submission

	S	T	U	V	W	X
2	0) Calculate the error					
3	Formula: Output - Target					
4	=L26-O3					
5	0.7361693					
6						
7	**1) Calculate the gradient of the error with respect to the output value**					
8	Formula: dError = Error * output * (1-output)					
9	=R5*L26*(1-L26)					
10	dError =	0.14298179				
11						
12	**2) Gradient of Output layer**					
13	Formula: Output of previous layer * dError					
14	d_w5:	=P16*T$10				
15	d_w5:	0.08883224				
16						
17	d_w6:	=P19*T10				
18	d_w6:	0.09585516				
19						
20	**3) Error gradient of hidden layer neurons**					
21	Formula: d_error * w5 * h1_output * (1 - h1_output)					
22	d_h1:	=T10*L24*P16*(1-P16)				
23	d_h1:	0.01614827				
24						
25	d_h2:	=T10*L25*P19*(1-P19)				
26	d_h2:	0.01074187				

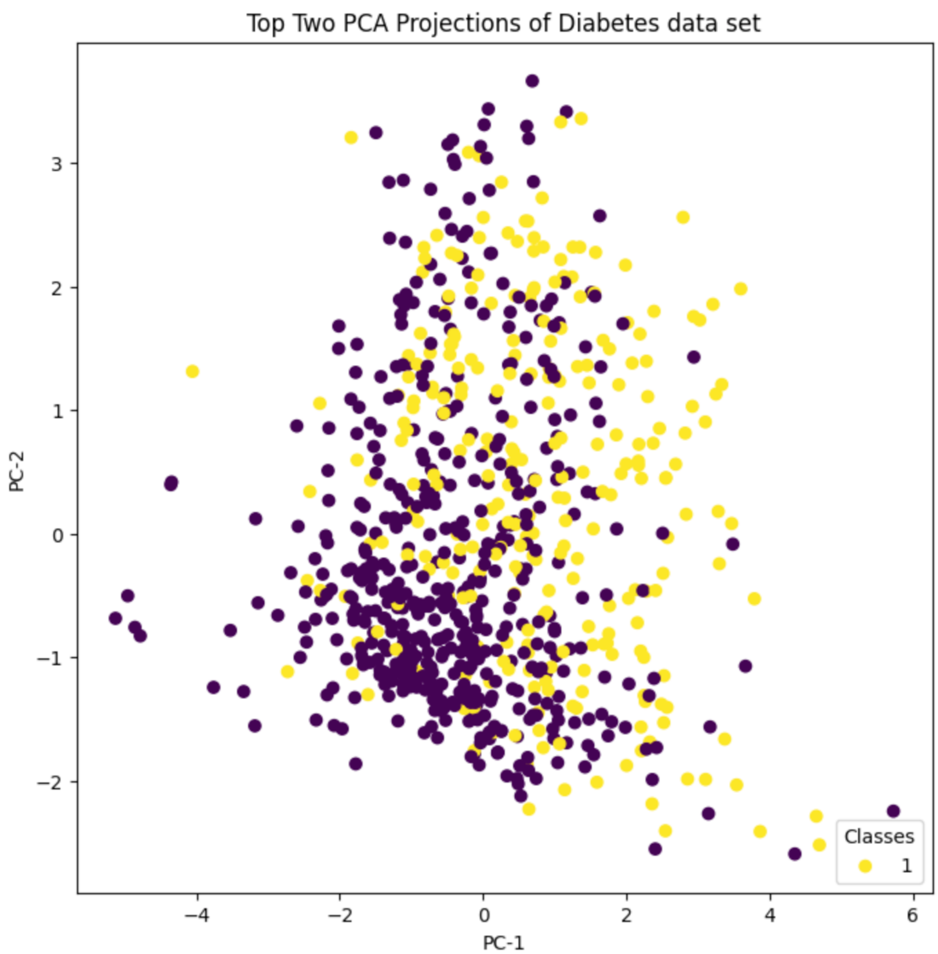
	S	T	U	V	W	X
28	**4) Gradient for the input layer weight**					
29	Formula: Input to the neuron * gradient of error at the following neuron					
30	d_w1:	=K16*T23				
31	d_w1:	0.01291862				
32						
33	d_w2:	=K17*T23				
34	d_w2:	0.00484448				
35						
36	d_w3:	=K19*T26				
37	d_w3:	0.0085935				
38						
39	d_w4:	=K20*T26				
40	d_w4:	0.00322256				
41						
42	**5) Update the weights**					
43	Formula: Old weight - Learning Rate * d_w					
44	new_w1:	=L16-N3*T31				
45	new_w1:	0.19870814				
46						
47	new_w2:	=L17-N3*T34				
48	new_w2:	0.34951555				
49						
50	new_w3:	=L19-N3*T37				
51	new_w3:	0.14914065				
52						
53	new_w4:	=L20-N3*T40				
54	new_w4:	0.59967774				
55						
56	new_w5:	=L24-N3*T15				
57	new_w5:	0.47111678				
58						
59	new_w6:	=L25-N3*T18				
60	new_w6:	0.33041448				

	S	T	U	V
62	**6) Gradient of dError with respect to the Bias**			
63	Formula: Gradient of the error on that edge			
64	d_b1:	=T23 (aka d_h1)		
65	d_b1:	0.01614827		
66				
67	d_b2:	=T26 (aka d_h1)		
68	d_b2:	0.01074187		
69				
70	d_b3:	=T10 (aka d_Error)		
71	d_b3:	0.14298179		
72				
73	**5) Update the biases**			
74	Formula: Old bias - Learning Rate * d_B			
75	new_b1:	=N16-N3*T65		
76	new_b1:	0.22838517		
77				
78	new_b2:	=N19-N3*T68		
79	new_b2:	0.40892581		
80				
81	new_b3:	=N24-N3*T71		
82	new_b3:	0.48570182		

Apply PCA to the Diabetes dataset. What are the resulting projections for the top 2 dimensions?

The projections themselves are stored in the attached diabetes_pca_top_2_projections.csv

Nevertheless, here is a plot of the values with component-1 on the X axis and component 2 on the Y axis.



Also of note:

```
********************************
Here are the top 2 components of the Diabetes data set after applying PCA
```

	preg	plas	pres	skin	insu	mass	pedi	age
PC-1	0.128432	0.393083	0.360003	0.439824	0.435026	0.451941	0.270611	0.198027
PC-2	0.593786	0.174029	0.183892	-0.331965	-0.250781	-0.100960	-0.122069	0.620589

```
For component 1 the percentage of the variance explained is: 26.18%
For component 2 the percentage of the variance explained is: 21.64%
```

Apply LDA and PCA to the iris dataset. Using the top 2 dimensions, graph the resulting subspace and include the graphs here. Which one produces a better separation between classes.

LDA provides a better separation between the classes. This is illustrated by noting how many of the dots shown in green "bleed" into the yellow class with PDA whereas there is almost no intermixing with LDA