

Homework 1

David A. Halloran

January 15, 2022

Theory

1) Given the following calculate h:

$$x = [1 \quad 2 \quad 3]$$

$$W = \begin{bmatrix} 1 & 2 \\ 3 & 4 \\ 5 & 6 \end{bmatrix}$$

$$b = [-1 \quad 2]$$

First multiply x into W which gives:

$$1 * 1 + 2 * 3 + 3 * 5 = 22$$

$$1 * 2 + 2 * 4 + 3 * 6 = 28$$

Simplified it looks like:

$$h = [22 \quad 28] + b$$

Next we add the biases which gives us an answer of:

$$h = [21 \quad 30]$$

2) Given matrix $h = [10 \quad -1]$ calculate the output of feeding this into different activation functions:

2a) Linear:

$$g(h) = h = h = [10 \quad -1]$$

2b) RELU: $g(h) = \max(0, h)$

$$h[0] = 10$$

$$\max(0, 10) = 10$$

$$h[1] = -1$$

$$\max(0, -1) = 0$$

$$h = [10 \quad 0]$$

2c) Sigmoid:

$$h[0] = \frac{1}{1 + e^{-10}} = 0.9999546021$$

$$h[1] = \frac{1}{1 + e^1} = 0.2689414214$$

$$h = [0.9999546021 \quad 0.2689414214]$$

2d) Hyperbolic Tangent:

$$h[0] = \frac{e^{10} - e^{-10}}{e^{10} + e^{-10}} = 0.9999999959$$

$$h[1] = \frac{e^{-1} - e^1}{e^{-1} + e^1} = -0.761594156$$

$$h = [0.9999999959 \quad -0.761594156]$$

2e) Softmax:

$$h[0] = \frac{e^{10}}{e^{10} + e^{-1}} = 0.9999832986$$

$$h[1] = \frac{e^{-1}}{e^{10} + e^{-1}} = 0.0000167014218$$

$$h = [0.9999832986 \quad 0.0000167014218]$$

Testing The Layers

This section will include the outputs of different layers with the test matrix:

$$x = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 5 & 6 & 7 & 8 \end{bmatrix}$$

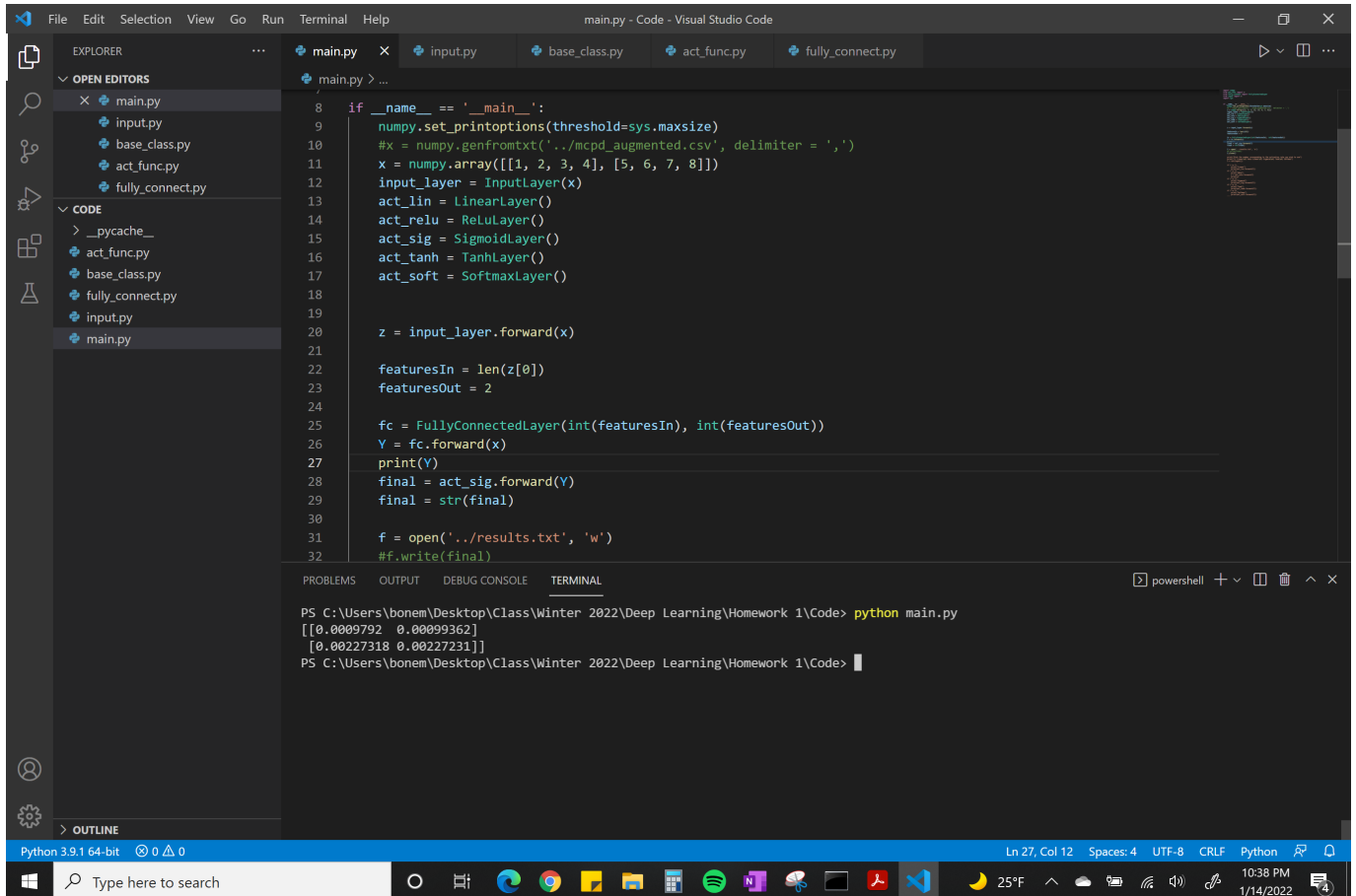
The image shows a Visual Studio Code editor window with a Python file named `main.py` open. The file contains the following code:

```
1 import numpy
2 from act_func import *
3 from fully_connect import FullyConnectedLayer
4 from input import *
5 import sys
6
7
8 if __name__ == '__main__':
9     numpy.set_printoptions(threshold=sys.maxsize)
10    #x = numpy.genfromtxt('../mcpd_augmented.csv', delimiter = ',')
11    x = numpy.array([[1, 2, 3, 4], [5, 6, 7, 8]])
12    input_layer = InputLayer(x)
13    act_lin = LinearLayer()
14    act_relu = ReLULayer()
15    act_sig = SigmoidLayer()
16    act_tanh = TanhLayer()
17    act_soft = SoftmaxLayer()
18
19
20 z = input_layer.forward(x)
21 print("Input Layer Output:")
22 print(z)
23 featuresIn = len(z[0])
24 featuresOut = 2
25
```

The terminal window at the bottom shows the execution of the script:

```
PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 1\Code> python main.py
mean:
[3. 4. 5. 6.]
stdev:
[2.82842712 2.82842712 2.82842712 2.82842712]
Input Layer Output:
[[-0.70710678 -0.70710678 -0.70710678 -0.70710678]
 [ 0.70710678  0.70710678  0.70710678  0.70710678]]
PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 1\Code>
```

Figure 1: Input Layer



```
8  if __name__ == '__main__':
9      numpy.set_printoptions(threshold=sys.maxsize)
10     #x = numpy.genfromtxt('../mcpd_augmented.csv', delimiter = ',')
11     x = numpy.array([[1, 2, 3, 4], [5, 6, 7, 8]])
12     input_layer = InputLayer(x)
13     act_lin = LinearLayer()
14     act_relu = ReLuLayer()
15     act_sig = SigmoidLayer()
16     act_tanh = TanhLayer()
17     act_soft = SoftmaxLayer()
18
19
20     z = input_layer.forward(x)
21
22     featuresIn = len(z[0])
23     featuresOut = 2
24
25     fc = FullyConnectedLayer(int(featuresIn), int(featuresOut))
26     Y = fc.forward(x)
27     print(Y)
28     final = act_sig.forward(Y)
29     final = str(final)
30
31     f = open('../results.txt', 'w')
32     #f.write(final)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 1\Code> python main.py
[[0.0009792  0.00099362]
 [0.00227318 0.00227231]]
PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 1\Code>
```

Figure 2: Fully Connected Layer

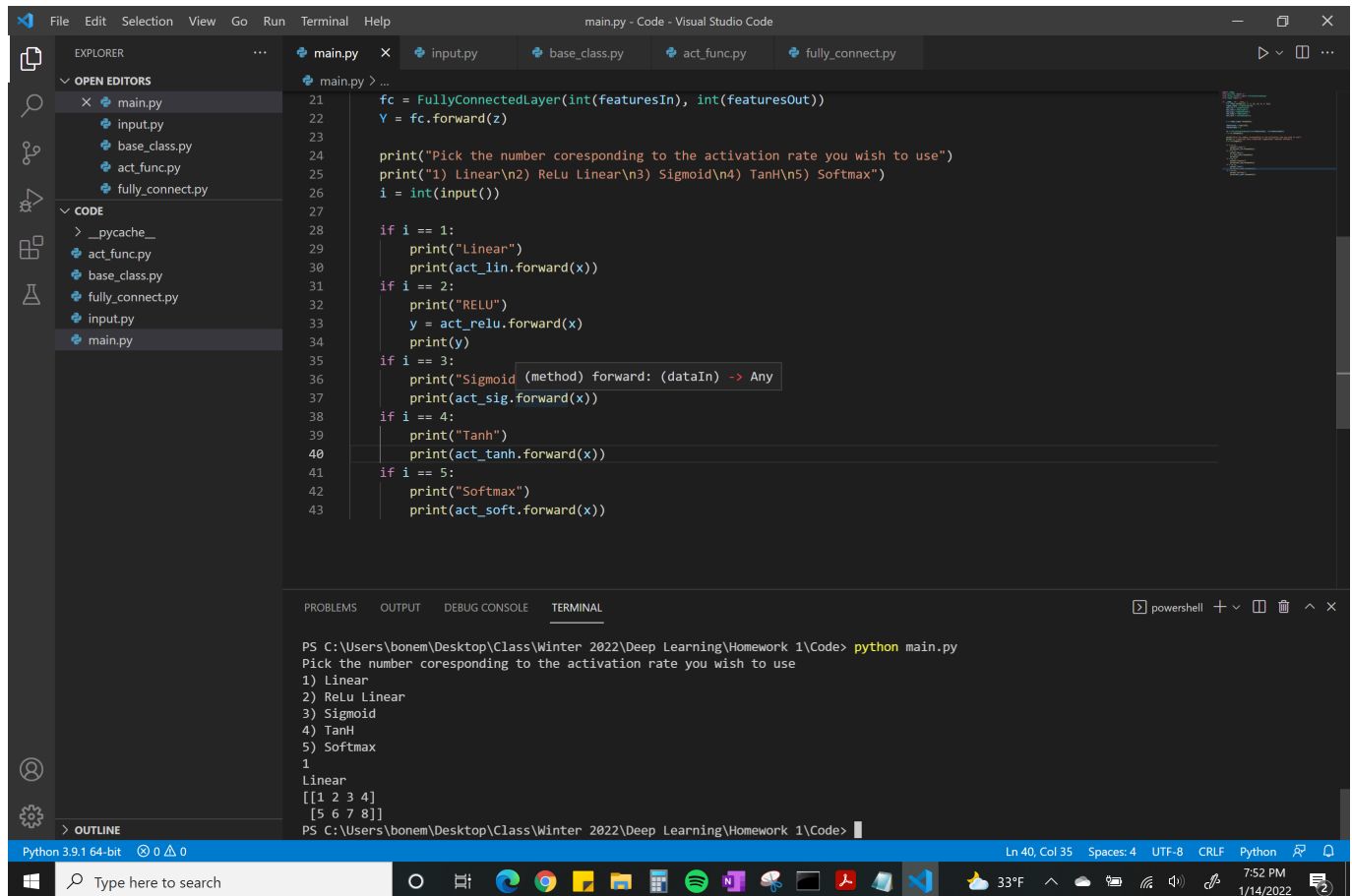


Figure 3: Linear Layer

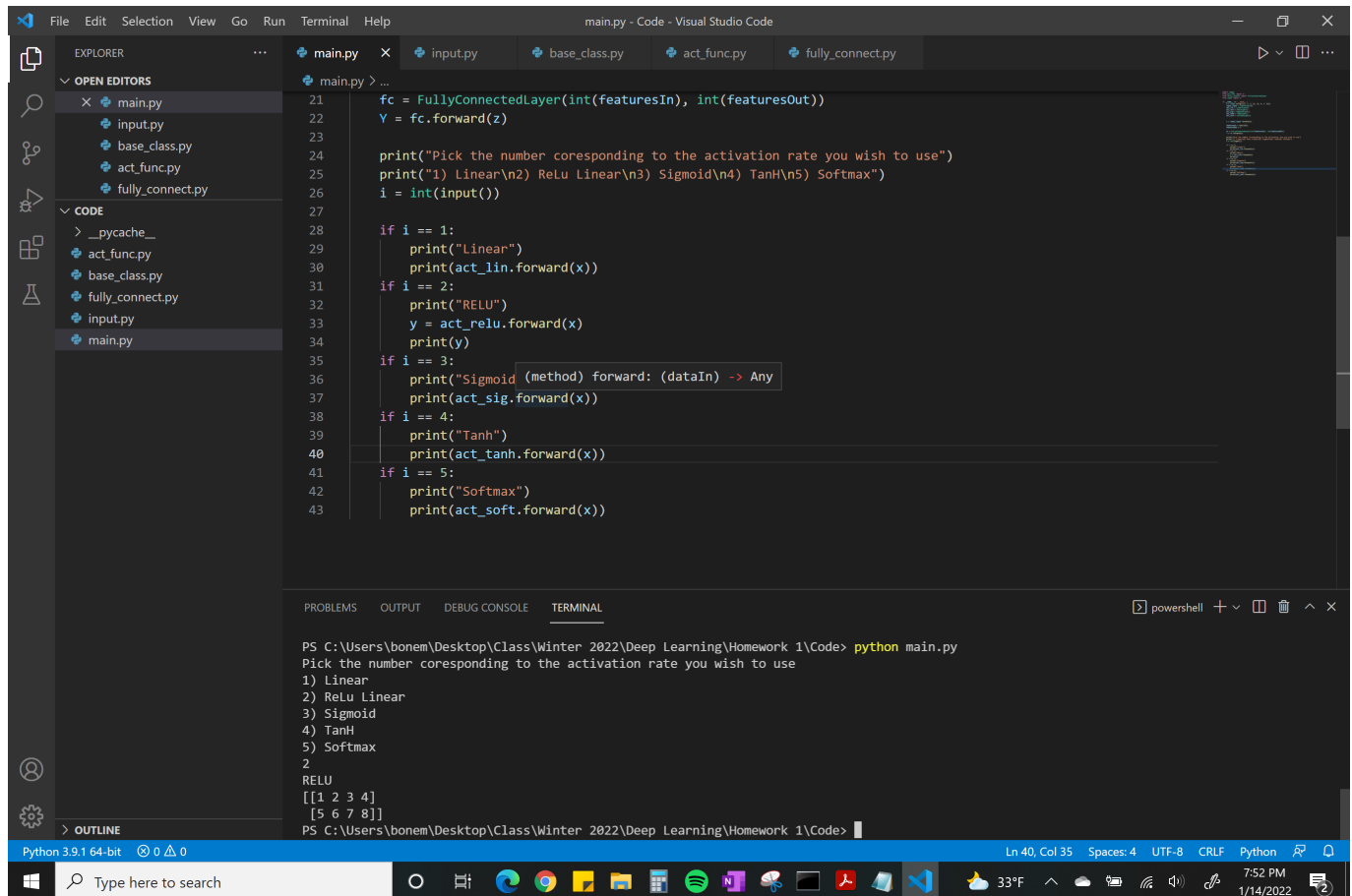


Figure 4: ReLu Layer

```
21 fc = FullyConnectedLayer(int(featuresIn), int(featuresOut))
22 Y = fc.forward(z)
23
24 print("Pick the number corresponding to the activation rate you wish to use")
25 print("1) Linear\n2) ReLu Linear\n3) Sigmoid\n4) TanH\n5) Softmax")
26 i = int(input())
27
28 if i == 1:
29     print("Linear")
30     print(act_lin.forward(x))
31 if i == 2:
32     print("ReLU")
33     y = act_relu.forward(x)
34     print(y)
35 if i == 3:
36     print("Sigmoid")
37     print(act_sig.forward(x))
38 if i == 4:
39     print("TanH")
40     print(act_tanh.forward(x))
41 if i == 5:
42     print("Softmax")
43     print(act_soft.forward(x))
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 1\Code> python main.py

Pick the number corresponding to the activation rate you wish to use

1) Linear

2) ReLu Linear

3) Sigmoid

4) TanH

5) Softmax

3

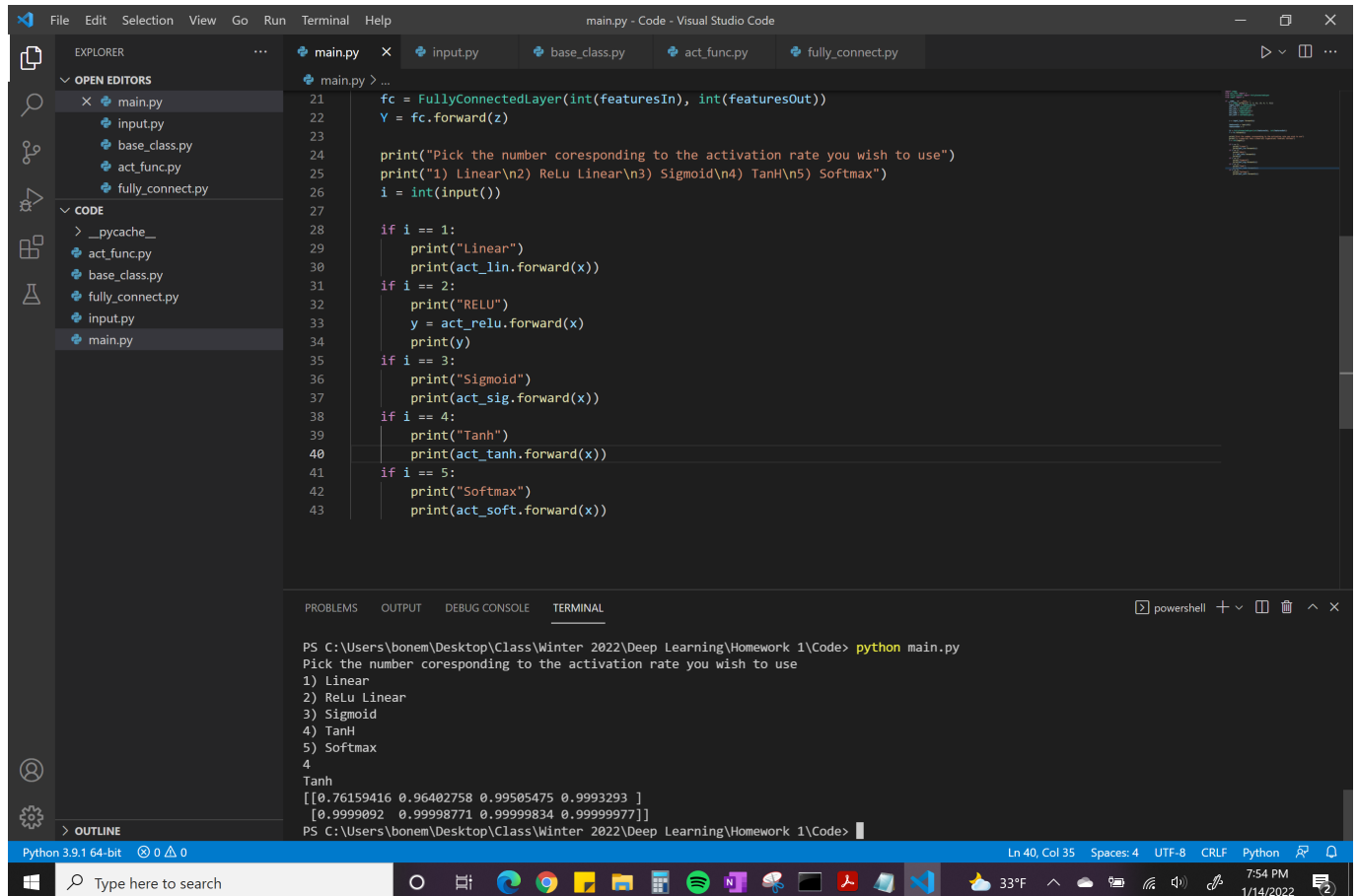
Sigmoid

[[0.73105858 0.88079708 0.95257413 0.98201379]

[0.99330715 0.99752738 0.99908895 0.99966465]]

PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 1\Code>

Figure 5: Sigmoid Layer



```
21 fc = FullyConnectedLayer(int(featuresIn), int(featuresOut))
22 Y = fc.forward(z)
23
24 print("Pick the number corresponding to the activation rate you wish to use")
25 print("1) Linear\n2) ReLu Linear\n3) Sigmoid\n4) TanH\n5) Softmax")
26 i = int(input())
27
28 if i == 1:
29     print("Linear")
30     print(act_lin.forward(x))
31 if i == 2:
32     print("ReLU")
33     y = act_relu.forward(x)
34     print(y)
35 if i == 3:
36     print("Sigmoid")
37     print(act_sig.forward(x))
38 if i == 4:
39     print("Tanh")
40     print(act_tanh.forward(x))
41 if i == 5:
42     print("Softmax")
43     print(act_soft.forward(x))
```

PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 1\Code> python main.py

Pick the number corresponding to the activation rate you wish to use

1) Linear

2) ReLu Linear

3) Sigmoid

4) TanH

5) Softmax

4

Tanh

[[0.76159416 0.96402758 0.99505475 0.9993293]

[0.9999092 0.9998771 0.99999834 0.99999977]]

Figure 6: TanH Layer

```
21 fc = FullyConnectedLayer(int(featuresIn), int(featuresOut))
22 Y = fc.forward(z)
23
24 print("Pick the number corresponding to the activation rate you wish to use")
25 print("1) Linear\n2) ReLu Linear\n3) Sigmoid\n4) TanH\n5) Softmax")
26 i = int(input())
27
28 if i == 1:
29     print("Linear")
30     print(act_lin.forward(x))
31
32 if i == 2:
33     print("ReLU")
34     y = act_relu.forward(x)
35     print(y)
36
37 if i == 3:
38     print("Sigmoid")
39     print(act_sig.forward(x))
40
41 if i == 4:
42     print("Tanh")
43     print(act_tanh.forward(x))
44
45 if i == 5:
46     print("Softmax")
47     print(act_soft.forward(x))
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 1\Code> python main.py

Pick the number corresponding to the activation rate you wish to use

1) Linear

2) ReLu Linear

3) Sigmoid

4) TanH

5) Softmax

5

Softmax

[[0.0320586 0.08714432 0.23688282 0.64391426]

[0.0320586 0.08714432 0.23688282 0.64391426]]

PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 1\Code>

Figure 7: Softmax Layer

```
10 #x = numpy.genfromtxt('../mcpd_augmented.csv', delimiter = ',')
11 x = numpy.array([[1, 2, 3, 4], [5, 6, 7, 8]])
12 input_layer = InputLayer(x)
13 act_lin = LinearLayer()
14 act_relu = ReLUlayer()
15 act_sig = SigmoidLayer()
16 act_tanh = TanhLayer()
17 act_soft = SoftmaxLayer()
18
19
20 z = input_layer.forward(x)
21 print("Input Layer:")
22 print(z)
23 featuresIn = len(z[0])
24 featuresOut = 2
25
26 fc = FullyConnectedLayer(int(featuresIn), int(featuresOut))
27 Y = fc.forward(z)
28 print("Fully Connected Layer")
29 print(Y)
30
31 final = act_sig.forward(Y)
32 print("Sigmoid:")
33 print(final)
34 final = str(final)
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 1\Code> python main.py

Input Layer:

```
[[[-0.70710678 -0.70710678 -0.70710678 -0.70710678]
 [ 0.70710678  0.70710678  0.70710678  0.70710678]]
```

Fully Connected Layer

```
[[[-3.96465590e-05 -2.11010305e-04]
 [ 4.17844215e-04  2.41075216e-04]]
```

Sigmoid:

```
[[[0.49999009 0.49994725]
 [0.50010446 0.50006027]]
```

PS C:\Users\bonem\Desktop\Class\Winter 2022\Deep Learning\Homework 1\Code>

Figure 8: From Input to FC to Sigmoid

Medical Data

For the final test the program returns a 1338 x 2 matrix with the very first observation as the following:

$$h[0] = [0.4999867 \quad 0.50005346]$$