

**LAPORAN UJIAN TENGAH SEMESTER
KECERDASAN BUATAN**



Disusun Oleh :

Nama : Shandy Ilham Alamsyah

NIM : 21091397015

**Program Studi D4 Manajemen Informatika
Program Vokasi
Universitas Negeri Surabaya
2022**

UTS 1

- **Single Neuron**

Source Code :

```
1  # Program Single Neuron,
2  # 1. Input layer feature 10
3  # 2. Neuron 1
4
5  # inisialisasi numpy
6  import numpy as np
7
8  # inisialisasi variable
9  inputs = [2, 5.5, 9, 4, 1, 6.5, 3, 7, 2.5, 6]
10
11 # inisialisasi bobot variable
12 weights = [-3.3, 4.2, 1.8, 3.8, -4.3, -0.3, 0.9, 1.9, 2.1, 2.8]
13
14 # inisialisasi bias
15 bias = 7
16
17 # penghitungan output = (input*weight)+bias
18 output = np.dot(weights, inputs) + bias
19
20 # cetak output
21 print(output)

```

```
>>>
86.69999999999999
```

Penjelasan step by step:

- Memanggil library Numpy yang telah diinstall, untuk memproses komputasi numeric / angka.

```
# inisialisasi numpy
import numpy as np
```

- Mengset nilai dari variabel inputs, weight, dan juga bias. Dengan ketentuan jumlah tiap input layer 10 dengan 1 neuron. Kita bisa memanfaatkan numeric generator di internet.

```
# inisialisasi variable
inputs = [2, 5.5, 9, 4, 1, 6.5, 3, 7, 2.5, 6]

# inisialisasi bobot variable
weights = [-3.3, 4.2, 1.8, 3.8, -4.3, -0.3, 0.9, 1.9, 2.1, 2.8]

# inisialisasi bias
bias = 7
```

- Melakukan penghitungan untuk mendapatkan output. Dengan rumus $(inputs * weights) + bias$.

```
# penghitungan output = (input*weight)+bias
output = np.dot(weights, inputs) + bias
```

Perhitungan dot product.

$$\begin{array}{cc}
 \text{weights} & \text{inputs} \\
 10 * 1 & 1 * 10
 \end{array}$$

$$\begin{bmatrix} 4.2 \\ 1.8 \\ 3.8 \\ -4.3 \\ -0.3 \\ 0.9 \\ 1.9 \\ 2.1 \\ 2.8 \end{bmatrix} * [2, 5.5, 9, 4, 1, 6.5, 3, 7, 2.5, 6] = 79.699$$

Kemudian akan ditambahkan dengan bias.

$$\begin{array}{cc}
 \text{np.dot} & \text{bias} \\
 \uparrow & \uparrow \\
 79.699 + 7 = 86.699
 \end{array}$$

- Mencetak output.

```
# cetak output
print(output)
```

```
>>>
86.69999999999999
```

- **Multi Neuron**

Source Code :

```

1  # Program Multi Neuron
2  # 1. Input layer feature 10
3  # 2. Neuron 5
4
5  # inisialisasi numpy
6  import numpy as np
7
8  # inisialisasi variable
9  inputs = [2, 5.5, 9, 4, 1, 6.5, 3, 7, 2.5, 6]
10
11 # inisialisasi bobot variable
12 weights = [[-3.3, 4.2, 1.8, 3.8, -4.3, -0.3, 0.9, 1.9, 2.1, 2.8],
13            [4, 3.6, 4.4, -3.3, -1.8, -2.1, 1.1, 2.2, -3.9, 3.4],
14            [1, -0.8, 3, 1.5, 1.2, -1.9, -2.7, 4, 0.2, 2.3],
15            [2.1, -2.6, 3.9, 4.6, 0.3, -3.5, 2.2, -4.8, 4, 2.3],
16            [4.1, -2.2, 0.7, 1.7, 2, 0.2, 4.6, 2.6, -2.3, 3]]
17
18 # inisialisasi bias
19 bias = [7, 3, 0.5, 1.5, 4.5]
20
21 # penghitungan output = (input*weight)+bias
22 output = np.dot(weights, inputs) + bias
23
24 # cetak output
25 print(output)

```

>>>
[86.7 71.1 54.15 19.25 61.25]

Penjelasan step by step:

- Memanggil library Numpy yang telah diinstall, untuk memproses komputasi numeric / angka.

```

# inisialisasi numpy
import numpy as np

```

- Mengset nilai dari variabel inputs, weight, dan juga bias. Dengan ketentuan jumlah tiap input layer 10 dengan 5 neuron.

```

# inisialisasi variable
inputs = [2, 5.5, 9, 4, 1, 6.5, 3, 7, 2.5, 6]

# inisialisasi bobot variable
weights = [[-3.3, 4.2, 1.8, 3.8, -4.3, -0.3, 0.9, 1.9, 2.1, 2.8],
           [4, 3.6, 4.4, -3.3, -1.8, -2.1, 1.1, 2.2, -3.9, 3.4],
           [1, -0.8, 3, 1.5, 1.2, -1.9, -2.7, 4, 0.2, 2.3],
           [2.1, -2.6, 3.9, 4.6, 0.3, -3.5, 2.2, -4.8, 4, 2.3],
           [4.1, -2.2, 0.7, 1.7, 2, 0.2, 4.6, 2.6, -2.3, 3]]

# inisialisasi bias
bias = [7, 3, 0.5, 1.5, 4.5]

```

- Melakukan penghitungan untuk mendapatkan output. Dengan rumus $(\text{inputs} * \text{weights}) + \text{bias}$.

```
# penghitungan output = (input*weight)+bias
output = np.dot(weights, inputs) + bias
```

Perhitungan dot product.

$$\begin{bmatrix} -3.3 & 4.2 & 1.8 & 3.8 & -4.3 & -0.3 & 0.9 & 1.9 & 2.1 & 2.8 \\ 4 & 3.6 & 4.4 & -3.3 & -1.8 & -2.1 & 1.1 & 2.2 & -3.9 & 3.4 \\ 1 & -0.8 & 3 & 1.5 & 1.2 & -1.9 & -2.7 & 4 & 0.2 & 2.3 \\ 2.1 & -2.6 & 3.9 & 4.6 & 0.3 & -3.5 & 2.2 & -4.8 & 4 & 2.3 \\ 4.1 & -2.2 & 0.7 & 1.7 & 2 & 0.2 & 4.6 & 2.6 & -2.3 & 3 \end{bmatrix} * [2 \ 5.5 \ 9 \ 4 \ 1 \ 6.5 \ 3 \ 7 \ 2.5 \ 6]$$

$$\begin{bmatrix} (2 * -3.3) (5.5 * 4.2) (9 * 1.8) (4 * 3.8) (1 * -4.3) (6.5 * -0.3) (3 * 0.9) (7 * 1.9) (2.5 * 2.1) (6 * 2.8) \\ (2 * 4) (5.5 * 3.6) (9 * 4.4) (4 * -3.3) (1 * -1.8) (6.5 * -2.1) (3 * 1.1) (7 * 2.2) (2.5 * -3.9) (6 * 3.4) \\ (2 * 1) (5.5 * -0.8) (9 * 3) (4 * 1.5) (1 * 1.2) (6.5 * -1.9) (3 * -2.7) (7 * 4) (2.5 * 0.2) (6 * 2.3) \\ (2 * 2.1) (5.5 * -2.6) (9 * 3.9) (4 * 4.6) (1 * 0.3) (6.5 * -3.5) (3 * 2.2) (7 * -4.8) (2.5 * 4) (6 * 2.3) \\ (2 * 4.1) (5.5 * -2.2) (9 * 0.7) (4 * 1.7) (1 * 2) (6.5 * 0.2) (3 * 4.6) (7 * 2.6) (2.5 * -2.3) (6 * 3) \end{bmatrix}$$

→ Neuron 1
 → Neuron 2
 → Neuron 3
 → Neuron 4
 → Neuron 5

$$\downarrow$$

$$[79.7 \ 68.1 \ 53.65 \ 17.75 \ 56.75]$$

Kemudian akan ditambahkan dengan bias.

$$[79.7 \ 68.1 \ 53.65 \ 17.75 \ 56.75] + [7 \ 3 \ 0.5 \ 1.5 \ 4.5]$$

$$= [86.7 \ 71.1 \ 54.15 \ 19.25 \ 61.25]$$

- Mencetak output.

```
# cetak output
print(output)
```

```
>>>
[86.7  71.1  54.15 19.25 61.25]
```

- **Multi Neuron Batch Input**

Source Code :

```
1  # Program Multi Neuron Batch Input
2  # 1. Input layer feature 10
3  # 2. Per batch nya 6 input
4  # 3. Neuron 5
5
6  # inisialisasi numpy
7  import numpy as np
8
9  # inisialisasi variable
10 inputs = [[-2, 5.5, 9, 4, 1, 6.5, -3, 7, 2.5, 6],
11           [1.6, -1.7, 2.3, -0.2, 3.7, 3.3, -3.7, -3.3, 0.5, 4],
12           [4.6, -2.6, 1.3, 1, -2.7, 0.5, 3.2, 4.2, -3.4, 4.7],
13           [2.3, -4.1, -3.4, 1, 0.2, -4.5, 5, 1.4, 4, 36],
14           [-1.3, 2, -5, 4.3, 4.5, 3.6, -0.6, -0.8, 0.5, 3.5],
15           [1.2, 0.7, -4.6, -4.5, 2.2, -2.2, 3, 4.3, -1.5, 4]]
16
17 # inisialisasi bobot variable
18 weights = [[-3.3, 4.2, 1.8, 3.8, -4.3, -0.3, 0.9, 1.9, 2.1, 2.8],
19            [4, 3.6, 4.4, -3.3, -1.8, -2.1, 1.1, 2.2, -3.9, 3.4],
20            [1, -0.8, 3, 1.5, 1.2, -1.9, -2.7, 4, 0.2, 2.3],
21            [2.1, -2.6, 3.9, 4.6, 0.3, -3.5, 2.2, -4.8, 4, 2.3],
22            [4.1, -2.2, 0.7, 1.7, 2, 0.2, 4.6, 2.6, -2.3, 3]]
23
24 # inisialisasi bias
25 bias = [7, 3, 0.5, 1.5, 4.5]
26
27 # penghitungan output
28 output = np.dot(inputs, np.array(weights).T) + bias
29
30 # cetak output
31 print(output)

```

```
>>>
[[ 94.5   48.5   66.35  -2.35  17.25]
 [-16.29   0.79  14.32  25.79   9.38]
 [ 15.38  60.27  26.68   9.12  73.95]
 [ 96.72 103.65  81.87 127.22 147.21]
 [ 15.39 -39.32  -5.82  -4.83  12.81]
 [-9.28  37.8   5.41 -38.92  41.4  ]]
```

Penjelasan step by step:

- Memanggil library Numpy yang telah diinstall, untuk memproses komputasi numeric / angka.

```
# inisialisasi numpy
import numpy as np
```

- Mengset nilai dari variabel inputs, weight, dan juga bias. Dengan ketentuan per batch 6 input dan tiap input – batch layer 10 jadi inputs = 6 * 10 dan 5 neuron.

```
# inisialisasi variable
inputs = [[-2, 5.5, 9, 4, 1, 6.5, -3, 7, 2.5, 6],
          [1.6, -1.7, 2.3, -0.2, 3.7, 3.3, -3.7, -3.3, 0.5, 4],
          [4.6, -2.6, 1.3, 1, -2.7, 0.5, 3.2, 4.2, -3.4, 4.7],
          [2.3, -4.1, -3.4, 1, 0.2, -4.5, 5, 1.4, 4, 36],
          [-1.3, 2, -5, 4.3, 4.5, 3.6, -0.6, -0.8, 0.5, 3.5],
          [1.2, 0.7, -4.6, -4.5, 2.2, -2.2, 3, 4.3, -1.5, 4]]

# inisialisasi bobot variable
weights = [[-3.3, 4.2, 1.8, 3.8, -4.3, -0.3, 0.9, 1.9, 2.1, 2.8],
           [4, 3.6, 4.4, -3.3, -1.8, -2.1, 1.1, 2.2, -3.9, 3.4],
           [1, -0.8, 3, 1.5, 1.2, -1.9, -2.7, 4, 0.2, 2.3],
           [2.1, -2.6, 3.9, 4.6, 0.3, -3.5, 2.2, -4.8, 4, 2.3],
           [4.1, -2.2, 0.7, 1.7, 2, 0.2, 4.6, 2.6, -2.3, 3]]

# inisialisasi bias
bias = [7, 3, 0.5, 1.5, 4.5]
```

- Melakukan penghitungan untuk mendapatkan output. Dengan rumus :

```
# penghitungan output
output = np.dot(inputs, np.array(weights).T) + bias
```

np.array (weight).T maksudnya adalah mentransposekan weight. Kenapa harus dilakukan Transpose pada weight, karena jika menggunakan weight normal maka ordonya tidak sesuai dengan ordo input sehingga tidak bisa dioperasikan.

Diagram illustrating the matrix multiplication of inputs and weights. The inputs matrix is labeled "Input - batch" with dimensions 6×10 . The weights matrix is labeled "weights" with dimensions 5×10 . The resulting output matrix is shown as a 6×10 matrix.

$$\begin{bmatrix} -2 & 5.5 & 9 & 4 & 1 & 6.5 & -3 & 7 & 2.5 & 6 \\ 1.6 & -1.7 & 2.3 & -0.2 & 3.7 & 3.3 & -3.7 & -3.3 & 0.5 & 4 \\ 4.6 & -2.6 & 1.3 & 1 & -2.7 & 0.5 & 3.2 & 4.2 & -3.4 & 4.7 \\ 2.3 & -4.1 & -3.4 & 1 & 0.2 & -4.5 & 5 & 1.4 & 4 & 36 \\ -1.3 & 2 & -5 & 4.3 & 4.5 & 3.6 & -0.6 & -0.8 & 0.5 & 3.5 \\ 1.2 & 0.7 & -4.6 & -4.5 & 2.2 & -2.2 & 3 & 4.3 & -1.5 & 4 \end{bmatrix} \times \begin{bmatrix} -3.3 & 4.2 & 1.8 & 3.8 & -4.3 & -0.3 & 0.9 & 1.9 & 2.1 & 2.8 \\ 4 & 3.6 & 4.4 & -3.3 & -1.8 & -2.1 & 1.1 & 2.2 & -3.9 & 3.4 \\ 1 & -0.8 & 3 & 1.5 & 1.2 & -1.9 & -2.7 & 4 & 0.2 & 2.3 \\ 2.1 & -2.6 & 3.9 & 4.6 & 0.3 & -3.5 & 2.2 & -4.8 & 4 & 2.3 \\ 4.1 & -2.2 & 0.7 & 1.7 & 2 & 0.2 & 4.6 & 2.6 & -2.3 & 3 \end{bmatrix}$$

Setelah ditranspose maka kedua ordo tersebut sama dan bisa dilakukan operasi perhitungan.

Diagram illustrating the matrix multiplication of inputs and transposed weights. The inputs matrix is labeled "Input - batch" with dimensions 6×10 . The weights matrix is labeled "[weights]^T" with dimensions 10×5 . The resulting output matrix is shown as a 6×10 matrix.

$$\begin{bmatrix} -2 & 5.5 & 9 & 4 & 1 & 6.5 & -3 & 7 & 2.5 & 6 \\ 1.6 & -1.7 & 2.3 & -0.2 & 3.7 & 3.3 & -3.7 & -3.3 & 0.5 & 4 \\ 4.6 & -2.6 & 1.3 & 1 & -2.7 & 0.5 & 3.2 & 4.2 & -3.4 & 4.7 \\ 2.3 & -4.1 & -3.4 & 1 & 0.2 & -4.5 & 5 & 1.4 & 4 & 36 \\ -1.3 & 2 & -5 & 4.3 & 4.5 & 3.6 & -0.6 & -0.8 & 0.5 & 3.5 \\ 1.2 & 0.7 & -4.6 & -4.5 & 2.2 & -2.2 & 3 & 4.3 & -1.5 & 4 \end{bmatrix} \times \begin{bmatrix} -3.3 & 4 & 1 & 2.1 & 4.1 \\ 4.2 & 3.6 & -0.8 & -2.6 & -2.2 \\ 1.8 & 4.4 & 3 & 3.9 & 0.7 \\ 3.8 & -3.3 & 1.5 & 4.6 & 1.7 \\ -4.3 & -1.8 & 1.2 & 0.3 & 2 \\ -0.3 & -2.1 & -1.9 & -3.5 & 0.2 \\ 0.9 & 1.1 & -2.7 & 2.2 & 4.6 \\ 1.9 & 2.2 & 4 & 4.8 & 2.6 \\ 2.1 & -3.9 & 0.2 & 4 & -2.3 \\ 2.8 & 3.4 & 2.3 & 2.3 & 3 \end{bmatrix} = \begin{bmatrix} 87.5 & 45.5 & 65.85 & -3.85 & 12.75 \\ -23.29 & -2.21 & 13.82 & 24.29 & 4.88 \\ 8.38 & 57.27 & 26.18 & 7.62 & 69.45 \\ 89.72 & 100.65 & 81.37 & 125.72 & 142.71 \\ 8.39 & -42.32 & -6.32 & -6.33 & 8.31 \\ -16.28 & 34.8 & 4.91 & -40.42 & 36.9 \end{bmatrix}$$

Kemudian akan ditambahkan dengan bias.

$$\begin{bmatrix} 87.5 & 45.5 & 65.85 & -3.85 & 12.75 \\ -23.29 & -2.21 & 13.82 & 24.29 & 4.88 \\ 8.38 & 57.27 & 26.18 & 7.62 & 69.45 \\ 89.72 & 100.65 & 81.37 & 125.72 & 142.71 \\ 8.39 & -42.32 & -6.32 & -6.33 & 8.31 \\ -16.28 & 34.8 & 4.91 & -40.42 & 36.9 \end{bmatrix} + [7, 3, 0.5, 1.5, 4.5] = \begin{bmatrix} 94.5 & 48.5 & 66.35 & -2.35 & 17.25 \\ -16.29 & 0.79 & 14.32 & 25.79 & 9.38 \\ 15.38 & 60.27 & 26.68 & 9.12 & 73.95 \\ 96.72 & 103.65 & 81.87 & 127.22 & 147.21 \\ 15.39 & -39.32 & -5.82 & -4.83 & 12.81 \\ -9.28 & 37.8 & 5.41 & -38.92 & 41.4 \end{bmatrix}$$

- Mencetak output.

```
# cetak output
print(output)
```

```
>>>
[[ 94.5    48.5    66.35   -2.35    17.25]
 [-16.29    0.79    14.32    25.79     9.38]
 [ 15.38    60.27    26.68     9.12    73.95]
 [ 96.72  103.65    81.87   127.22   147.21]
 [ 15.39  -39.32    -5.82    -4.83    12.81]
 [-9.28    37.8     5.41   -38.92    41.4 ]]
```


UTS 2

- Multi Neuron Batch Input

Source Code :

```
1  # Program Multi Neuron Batch Input
2  # 1. Input layer feature 10
3  # 2. Per batch nya 6 input
4  # 3. Hidden layer 1, 5 neuron
5  # 4. Hidden layer 2, 3 neuron
6
7  # inisialisasi numpy
8  import numpy as np
9
10 # inisialisasi variabel inputs
11 inputs = [[-2, 5.5, 9, 4, 1, 6.5, -3, 7, 2.5, 6],
12           [1.6, -1.7, 2.3, -0.2, 3.7, 3.3, -3.7, -3.3, 0.5, 4],
13           [4.6, -2.6, 1.3, 1, -2.7, 0.5, 3.2, 4.2, -3.4, 4.7],
14           [2.3, -4.1, -3.4, 1, 0.2, -4.5, 5, 1.4, 4, 36],
15           [-1.3, 2, -5, 4.3, 4.5, 3.6, -0.6, -0.8, 0.5, 3.5],
16           [1.2, 0.7, -4.6, -4.5, 2.2, -2.2, 3, 4.3, -1.5, 4]]
17
18 # hidden layer 1
19 # inisialisasi bobot hidden layer 1
20 weights = [[-3.3, 4.2, 1.8, 3.8, -4.3, -0.3, 0.9, 1.9, 2.1, 2.8],
21            [4, 3.6, 4.4, -3.3, -1.8, -2.1, 1.1, 2.2, -3.9, 3.4],
22            [1, -0.8, 3, 1.5, 1.2, -1.9, -2.7, 4, 0.2, 2.3],
23            [2.1, -2.6, 3.9, 4.6, 0.3, -3.5, 2.2, -4.8, 4, 2.3],
24            [4.1, -2.2, 0.7, 1.7, 2, 0.2, 4.6, 2.6, -2.3, 3]]
25
26 #inisialisasi bias hidden layer 1
27 bias = [7, 3, 0.5, 1.5, 4.5]
28
29 #output
30 output = np.dot(inputs, np.array(weights).T) + bias
31
32 # hidden layer 2
33 # inisialisasi bobot hidden layer 2
34 weights2 = [[-2.1, 2.6, 2.8, -1, 3.7],
35             [-1.6, 1, -0.5, 2.1, 4],
36             [1.8, -4, -0.1, 1.3, 2.1]]
37
38 # inisialisasi bias hidden layer 2
39 bias2 = [2.5, 0.5, -4.5]
40
41 #output 2
42 output2 = np.dot(output, np.array(weights2).T) + bias2
43
44 # cetak output
45 print(output)
46 print(output2)
47
48 >>>
49 [[ 182.105  -71.31   -1.865]
50 [   87.775  111.873  14.811]
51 [  466.103  337.774 -53.413]
52 [  715.571  764.465 221.336]
53 [  -96.12   -19.437 201.686]
54 [  327.516  134.311 -136.601]]
```

Penjelasan step by step:

- Memanggil library Numpy yang telah diinstall, untuk memproses komputasi numeric / angka.

```
# inisialisasi numpy
import numpy as np
```

- Mengset nilai dari variabel inputs, weight, dan juga bias. Dengan ketentuan per batch 6 input dan tiap input – batch memiliki 10 fitur didalamnya, jadi inputs = 6 * 10, 5 neuron pada hidden layer 1 dan 3 neuron pada hidden layer 2. Perlu diingat jika weight pada hidden layer 2, mengikuti ordo dari output hidden layer 1.

```
# inisialisasi variabel inputs
inputs = [[-2, 5.5, 9, 4, 1, 6.5, -3, 7, 2.5, 6],
          [1.6, -1.7, 2.3, -0.2, 3.7, 3.3, -3.7, -3.3, 0.5, 4],
          [4.6, -2.6, 1.3, 1, -2.7, 0.5, 3.2, 4.2, -3.4, 4.7],
          [2.3, -4.1, -3.4, 1, 0.2, -4.5, 5, 1.4, 4, 36],
          [-1.3, 2, -5, 4.3, 4.5, 3.6, -0.6, -0.8, 0.5, 3.5],
          [1.2, 0.7, -4.6, -4.5, 2.2, -2.2, 3, 4.3, -1.5, 4]]
```

```
# hidden layer 1
# inisialisasi bobot hidden layer 1
weights = [[-3.3, 4.2, 1.8, 3.8, -4.3, -0.3, 0.9, 1.9, 2.1, 2.8],
           [4, 3.6, 4.4, -3.3, -1.8, -2.1, 1.1, 2.2, -3.9, 3.4],
           [1, -0.8, 3, 1.5, 1.2, -1.9, -2.7, 4, 0.2, 2.3],
           [2.1, -2.6, 3.9, 4.6, 0.3, -3.5, 2.2, -4.8, 4, 2.3],
           [4.1, -2.2, 0.7, 1.7, 2, 0.2, 4.6, 2.6, -2.3, 3]]

#inisialisasi bias hidden layer 1
bias = [7, 3, 0.5, 1.5, 4.5]
```

```
# hidden layer 2
# inisialisasi bobot hidden layer 2
weights2 = [[-2.1, 2.6, 2.8, -1, 3.7],
            [-1.6, 1, -0.5, 2.1, 4],
            [1.8, -4, -0.1, 1.3, 2.1]]

# inisialisasi bias hidden layer 2
bias2 = [2.5, 0.5, -4.5]
```

- Melakukan penghitungan untuk mendapatkan output dari hidder layer pertama, dengan rumus :

```
#output hidden layer 1
output = np.dot(inputs, np.array(weights).T) + bias
```

np.array(weight).T maksudnya adalah mentransposekan weight. Kenapa harus dilakukan Transpose pada weight, karena jika menggunakan weight normal maka ordonya tidak sesuai dengan ordo input sehingga tidak bisa dioperasikan.

$$\begin{bmatrix} -2 & 5.5 & 9 & 4 & 1 & 6.5 & -3 & 7 & 2.5 & 6 \\ 1.6 & -1.7 & 2.3 & -0.2 & 3.7 & 3.3 & -3.7 & -3.3 & 0.5 & 4 \\ 4.6 & -2.6 & 1.3 & 1 & -2.7 & 0.5 & 3.2 & 4.2 & -3.4 & 4.7 \\ 2.3 & -4.1 & -3.4 & 1 & 0.2 & -4.5 & 5 & 1.4 & 4 & 3.6 \\ -1.3 & 2 & -5 & 4.3 & 4.5 & 3.6 & -0.6 & -0.8 & 0.5 & 3.5 \\ 1.2 & 0.7 & -4.6 & -4.5 & 2.2 & -2.2 & 3 & 4.3 & -1.5 & 4 \end{bmatrix} \cdot \begin{bmatrix} -3.3 & 4.2 & 1.8 & 3.8 & -4.3 & -0.3 & 0.9 & 1.9 & 2.1 & 2.8 \\ 4 & 3.6 & 4.4 & -3.3 & -1.8 & -2.1 & 1.1 & 2.2 & -3.9 & 3.4 \\ 1 & -0.8 & 3 & 1.5 & 1.2 & -1.9 & -2.7 & 4 & 0.2 & 2.3 \\ 2.1 & -2.6 & 3.9 & 4.6 & 0.3 & -3.5 & 2.2 & -4.8 & 4 & 2.3 \\ 4.1 & -2.2 & 0.7 & 1.7 & 2 & 0.2 & 4.6 & 2.6 & -2.3 & 3 \end{bmatrix}$$

Setelah ditranspose maka kedua ordo tersebut sama dan bisa dilakukan operasi perhitungan.

$$\begin{bmatrix} -2 & 5.5 & 9 & 4 & 1 & 6.5 & -3 & 7 & 2.5 & 6 \\ 1.6 & -1.7 & 2.3 & -0.2 & 3.7 & 3.3 & -3.7 & -3.3 & 0.5 & 4 \\ 4.6 & -2.6 & 1.3 & 1 & -2.7 & 0.5 & 3.2 & 4.2 & -3.4 & 4.7 \\ 2.3 & -4.1 & -3.4 & 1 & 0.2 & -4.5 & 5 & 1.4 & 4 & 3.6 \\ -1.3 & 2 & -5 & 4.3 & 4.5 & 3.6 & -0.6 & -0.8 & 0.5 & 3.5 \\ 1.2 & 0.7 & -4.6 & -4.5 & 2.2 & -2.2 & 3 & 4.3 & -1.5 & 4 \end{bmatrix} \cdot \begin{bmatrix} -3.3 & 4 & 1 & 2.1 & 4.1 \\ 4.2 & 3.6 & -0.8 & -2.6 & -2.2 \\ 1.8 & 4.4 & 3 & 3.9 & 0.7 \\ 3.8 & -3.3 & 1.5 & 4.6 & 1.7 \\ -4.3 & -1.8 & 1.2 & 0.3 & 2 \\ -0.3 & -2.1 & -1.9 & -3.5 & 0.2 \\ 0.9 & 1.1 & -2.7 & 2.2 & 4.6 \\ 1.9 & 2.2 & 4 & 4.8 & 2.6 \\ 2.1 & -3.9 & 0.2 & 4 & -2.3 \\ 2.8 & 3.4 & 2.3 & 2.3 & 3 \end{bmatrix} = \begin{bmatrix} 87.5 & 45.5 & 65.85 & -3.85 & 12.75 \\ -23.29 & -2.21 & 13.82 & 24.29 & 4.88 \\ 8.38 & 57.27 & 26.18 & 7.62 & 69.45 \\ 89.72 & 100.65 & 81.37 & 125.72 & 142.71 \\ 8.39 & -42.32 & -6.32 & -6.33 & 8.31 \\ -16.28 & 34.8 & 4.91 & -40.42 & 36.9 \end{bmatrix}$$

Kemudian akan ditambahkan dengan bias.

$$\begin{bmatrix} 87.5 & 45.5 & 65.85 & -3.85 & 12.75 \\ -23.29 & -2.21 & 13.82 & 24.29 & 4.88 \\ 8.38 & 57.27 & 26.18 & 7.62 & 69.45 \\ 89.72 & 100.65 & 81.37 & 125.72 & 142.71 \\ 8.39 & -42.32 & -6.32 & -6.33 & 8.31 \\ -16.28 & 34.8 & 4.91 & -40.42 & 36.9 \end{bmatrix} + [7, 3, 0.5, 1.5, 4.5] = \begin{bmatrix} 94.5 & 48.5 & 66.35 & -2.35 & 17.25 \\ -16.29 & 0.79 & 14.32 & 25.79 & 9.38 \\ 15.38 & 60.27 & 26.68 & 9.12 & 73.95 \\ 96.72 & 103.65 & 81.87 & 127.22 & 147.21 \\ 15.39 & -39.32 & -5.82 & -4.83 & 12.81 \\ -9.28 & 37.8 & 5.41 & -38.92 & 41.4 \end{bmatrix}$$

Setelah hasil dari hidden layer pertama ditemukan akan langsung masuk ke hidden layer kedua.

```
#output hidden layer 2
output2 = np.dot(output, np.array(weights2).T) + bias2
```

Seperti halnya pada hidden layer 1, pada hidden layer 2 juga untuk weight akan di transpose untuk menyesuaikan dengan ordo output hidden layer 1.

$$\begin{bmatrix} 94.5 & 48.5 & 66.35 & -2.35 & 17.25 \\ -16.29 & 0.79 & 14.32 & 25.79 & 9.38 \\ 15.38 & 60.27 & 26.68 & 9.12 & 73.95 \\ 96.72 & 103.65 & 81.87 & 127.22 & 147.21 \\ 15.39 & -39.32 & -5.82 & -4.83 & 12.81 \\ -9.28 & 37.8 & 5.41 & -38.92 & 41.4 \end{bmatrix} \cdot \begin{bmatrix} -2.1 & 2.6 & 2.8 & -1 & 3.7 \\ -1.6 & 1 & -0.5 & 2.1 & 4 \\ 1.8 & -4 & -0.1 & 1.3 & 2.1 \end{bmatrix}$$

Setelah ditranspose maka dilakukan perhitungan berikut

$$\begin{array}{c}
 \text{Hidden Layer 1} \\
 6 \times 5
 \end{array}
 \begin{bmatrix}
 94.5 & 48.5 & 66.35 & -2.35 & 17.25 \\
 -16.29 & 0.79 & 14.32 & 25.79 & 9.38 \\
 15.38 & 60.27 & 26.68 & 9.12 & 73.95 \\
 96.72 & 103.65 & 81.87 & 127.22 & 147.21 \\
 15.39 & -39.32 & -5.82 & -4.83 & 12.81 \\
 -9.28 & 37.8 & 5.41 & -38.92 & 41.4
 \end{bmatrix}
 \begin{array}{c}
 \text{weights 2} \\
 5 \times 3
 \end{array}
 \begin{bmatrix}
 -2.1 & -1.6 & 1.8 \\
 2.6 & 1 & -4 \\
 2.8 & -0.5 & -0.1 \\
 -1 & 2.1 & 1.3 \\
 3.7 & 4 & 2.1
 \end{bmatrix}
 =
 \begin{bmatrix}
 179.605 & -71.81 & 2.635 \\
 85.275 & 111.373 & 19.311 \\
 463.603 & 337.274 & -48.913 \\
 713.071 & 763.965 & 225.836 \\
 -98.62 & -19.937 & 206.186 \\
 325.016 & 133.811 & -132.101
 \end{bmatrix}$$

Kemudian akan ditambahkan dengan bias 2

$$\begin{bmatrix}
 179.605 & -71.81 & 2.635 \\
 85.275 & 111.373 & 19.311 \\
 463.603 & 337.274 & -48.913 \\
 713.071 & 763.965 & 225.836 \\
 -98.62 & -19.937 & 206.186 \\
 325.016 & 133.811 & -132.101
 \end{bmatrix}
 + [2.5 \quad 0.5 \quad -4.5] =
 \begin{bmatrix}
 182.105 & -71.31 & -1.865 \\
 87.775 & 111.873 & 14.811 \\
 466.103 & 337.774 & -53.413 \\
 715.571 & 764.465 & 221.336 \\
 -96.12 & -19.437 & 201.686 \\
 327.516 & 134.311 & -136.601
 \end{bmatrix}$$

- Mencetak output.

```
# cetak output
print(output)
```

```
>>>
[[ 182.105  -71.31   -1.865]
 [  87.775  111.873   14.811]
 [ 466.103  337.774  -53.413]
 [ 715.571  764.465  221.336]
 [ -96.12   -19.437  201.686]
 [ 327.516  134.311 -136.601]]
```