

LAPORAN UTS KECERDASAN BUATAN



Oleh:

Alvin Febrianto

21091397031

D4 MANAJEMEN INFORMATIKA

FAKULTAS VOKASI

UNIVERSITAS NEGERI SURABAYA

TAHUN AJARAN 2022/2023

UTS 1

A. Single Neuron

- i. Input layer feature 10
- ii. Neuron 1

a) Source Code

```
1a_SingleNeuron_031_AlvinFebrianto.py x
1  # Alvin Febrianto
2  # 21091397031
3  # 2021 A / D4 Manajemen Informatika
4
5  # a. Single Neuron
6  # i. Input layer feature 10
7  # ii. Neuron 1
8
9  import numpy as np
10
11 # Inisialisasi variabel inputs
12 inputs = [0.8, 4, 7, -3.2, 1.3, 3, -8, -2.0, 4.5, 2.4]
13
14 # Inisialisasi variabel weights
15 weights = [-3.6, 3.2, 6, -1.8, 2.8, -7, 5.7, 3.4, 3.1, 4.2]
16
17 # Inisialisasi variabel bias
18 bias = 6
19
20 # Penghitungan output menggunakan rumus dot product vector (inputs*weights)+bias
21 output = np.dot(weights, inputs) + bias
22
23 # Menampilkan output
24 print(output)

17.95
[Finished in 0.8s]
```

b) Penjelasan

- Langkah pertama yaitu mengimpor NumPy (library Python) yang bisa digunakan untuk melakukan operasi vektor dan matriks. Lalu mengisi nilai dari setiap variabel inputs, weights, dan bias sesuai ketentuan soal dengan jumlah tiap input layer 10 dan 1 neuron. Maksudnya, pada variabel weights berisi sebanyak 10 angka setiap 1 barisnya (sesuai banyaknya angka dalam 1 baris inputs) dengan total 1 baris (karena 1 neuron). Karena hanya 1 neuron maka isi variabel bias sebanyak 1 angka.

```
9  import numpy as np
10
11 # Inisialisasi variabel inputs
12 inputs = [0.8, 4, 7, -3.2, 1.3, 3, -8, -2.0, 4.5, 2.4]
13
14 # Inisialisasi variabel weights
15 weights = [-3.6, 3.2, 6, -1.8, 2.8, -7, 5.7, 3.4, 3.1, 4.2]
16
17 # Inisialisasi variabel bias
18 bias = 6
```

- Melakukan operasi perhitungan dot product dengan rumus $(\text{inputs} \times \text{weights}) + \text{bias}$.

```
20 # Penghitungan output menggunakan rumus dot product vector (inputs*weights)+bias
21 output = np.dot(weights, inputs) + bias
```

weights 10×1 inputs 1×10

$$\begin{bmatrix} -3.6 \\ 3.2 \\ 6 \\ -1.8 \\ 2.8 \\ -7 \\ 5.7 \\ 3.4 \\ 3.1 \\ 4.2 \end{bmatrix} * [0.8, 4, 7, -3.2, 1.3, 3, -8, -2.0, 4.5, 2.4] = 11.95$$

tambahkan dengan bias, $\text{np.dot} + \text{bias}$

$11.95 + 6 = 17.95$

- Menampilkan output.

```
23 # Menampilkan output
24 print(output)

17.95
[Finished in 0.8s]
```

B. Multi Neuron

- Input layer feature 10
- Neuron 5

a) Source Code

```
1b_MultiNeuron_031_AlvinFebrianto.py x
1 # Alvin Febrianto
2 # 21091397031
3 # 2021 A / D4 Manajemen Informatika
4
5 # b. Multi Neuron
6 # i. Input layer feature 10
7 # ii. Neuron 5
8
9 import numpy as np
10
11 # Inisialisasi variabel inputs
12 inputs = [-8, 7.1, -5, 4.2, 5.3, 2.0, 7.9, 2.4, 8.4, -4]
13
14 # Inisialisasi variabel weights
15 weights = [[1.4, 2.0, 2.8, -4.2, 6.0, 2.4, -3.8, 1.2, 6.2, 5.1],
16            [6.0, 6.6, 2.8, -8.2, 6.2, 1.6, 5.1, 3.6, -7.1, 4.0],
17            [8.6, -2.3, 5.1, 7.4, 1.4, -2.4, 7.5, 5.7, 1.9, 8.4],
18            [5.6, 4.0, -6.5, 8.2, 2.6, 1.8, -7.3, 7.8, 3.7, 5.9],
19            [6.0, 3.0, 6.6, 1.3, -3.2, 7.1, 7.4, 9.4, 1.1, -7.6]]
20
21 # Inisialisasi variabel biases
22 biases = [2.0, 4, 0.8, 5.2, 3]
23
24 # Penghitungan output menggunakan rumus dot product vector (inputs*weights)+biases
25 output = np.dot(weights, inputs) + biases
26
27 # Menampilkan output
28 print(output)

[ 14.5 -36.23 -20.84  41.65  66.66]
[Finished in 0.7s]
```

b) Penjelasan

- Langkah pertama yaitu mengimpor NumPy (library Python) yang bisa digunakan untuk melakukan operasi vektor dan matriks. Lalu mengisi nilai dari setiap variabel inputs, weights, dan biases sesuai ketentuan soal dengan jumlah tiap input layer 10 dan 5 neuron. Maksudnya, pada variabel weights berisi sebanyak 10 angka setiap 1 barisnya (sesuai banyaknya angka dalam 1 baris inputs) dengan total 5 baris (karena 5 neuron). Karena ada 5 neuron maka isi variabel biases sebanyak 5 angka.

```
9   import numpy as np
10
11   # Inisialisasi variabel inputs
12   inputs = [-8, 7.1, -5, 4.2, 5.3, 2.0, 7.9, 2.4, 8.4, -4]
13
14   # Inisialisasi variabel weights
15   weights = [[1.4, 2.0, 2.8, -4.2, 6.0, 2.4, -3.8, 1.2, 6.2, 5.1],
16              [6.0, 6.6, 2.8, -8.2, 6.2, 1.6, 5.1, 3.6, -7.1, 4.0],
17              [8.6, -2.3, 5.1, 7.4, 1.4, -2.4, 7.5, 5.7, 1.9, 8.4],
18              [5.6, 4.0, -6.5, 8.2, 2.6, 1.8, -7.3, 7.8, 3.7, 5.9],
19              [6.0, 3.0, 6.6, 1.3, -3.2, 7.1, 7.4, 9.4, 1.1, -7.6]]
20
21   # Inisialisasi variabel biases
22   biases = [2.0, 4, 0.8, 5.2, 3]
```

- Melakukan perhitungan dot product dengan rumus $(\text{inputs} \times \text{weights}) + \text{biases}$.

```
24   # Penghitungan output menggunakan rumus dot product vector (inputs*weights)+biases
25   output = np.dot(weights, inputs) + biases
```

$$\begin{array}{c} \text{weights} \\ 5 \times 10 \end{array} \begin{array}{c} \text{inputs} \\ 1 \times 10 \end{array}$$
$$\begin{bmatrix} 1.4 & 2.0 & 2.8 & -4.2 & 6.0 & 2.4 & -3.8 & 1.2 & 6.2 & 5.1 \\ 6.0 & 6.6 & 2.8 & -8.2 & 6.2 & 1.6 & 5.1 & 3.6 & -7.1 & 4.0 \\ 8.6 & -2.3 & 5.1 & 7.4 & 1.4 & -2.4 & 7.5 & 5.7 & 1.9 & 8.4 \\ 5.6 & 4.0 & -6.5 & 8.2 & 2.6 & 1.8 & -7.3 & 7.8 & 3.7 & 5.9 \\ 6.0 & 3.0 & 6.6 & 1.3 & -3.2 & 7.1 & 7.4 & 9.4 & 1.1 & -7.6 \end{bmatrix} * \begin{bmatrix} -8 & 7.1 & -5 & 4.2 & 5.3 & 2.0 & 7.9 & 2.4 & 8.4 & -4 \end{bmatrix}$$
$$= \begin{bmatrix} (-8 * 1.4) & (7.1 * 2.0) & (-5 * 2.8) & (4.2 * -4.2) & (5.3 * 6.0) & (2.0 * 2.4) & (7.9 * -3.8) & (2.4 * 1.2) & (8.4 * 6.2) & (-4 * 5.1) \\ (-8 * 6.0) & (7.1 * 6.6) & (-5 * 2.8) & (4.2 * -8.2) & (5.3 * 6.2) & (2.0 * 1.6) & (7.9 * 5.1) & (2.4 * 3.6) & (8.4 * -7.1) & (-4 * 4.0) \\ (-8 * 8.6) & (7.1 * -2.3) & (-5 * 5.1) & (4.2 * 7.4) & (5.3 * 1.4) & (2.0 * -2.4) & (7.9 * 7.5) & (2.4 * 5.7) & (8.4 * 1.9) & (-4 * 8.4) \\ (-8 * 5.6) & (7.1 * 4.0) & (-5 * -6.5) & (4.2 * 8.2) & (5.3 * 2.6) & (2.0 * 1.8) & (7.9 * -7.3) & (2.4 * 7.8) & (8.4 * 3.7) & (-4 * 5.9) \\ (-8 * 6.0) & (7.1 * 3.0) & (-5 * 6.6) & (4.2 * 1.3) & (5.3 * -3.2) & (2.0 * 7.1) & (7.9 * 7.4) & (2.4 * 9.4) & (8.4 * 1.1) & (-4 * -7.6) \end{bmatrix}$$
$$= [12.5 \quad -40.23 \quad -21.64 \quad 36.45 \quad 63.66]$$

tambahkan dengan bias, $\text{np.dot} + \text{biases}$

$$[12.5 \quad -40.23 \quad -21.64 \quad 36.45 \quad 63.66] + [2.0 \quad 4 \quad 0.8 \quad 5.2 \quad 3] = [14.5 \quad -36.23 \quad -20.84 \quad 41.65 \quad 66.66]$$

- Menampilkan output.

```
27   # Menampilkan output
28   print(output)
```

```
[ 14.5  -36.23 -20.84  41.65  66.66]
[Finished in 0.7s]
```

C. Multi Neuron Batch Input

- i. Input layer feature 10
- ii. Per batch nya 6 input
- iii. Neuron 5

a) Source Code

```
1c_MultiNeuronBatchInput_031_AlvinFebrianto.py x
1  # Alvin Febrianto
2  # 21091397031
3  # 2021 A / D4 Manajemen Informatika
4
5  # c. Multi Neuron Batch Input
6  # i. Input layer feature 10
7  # ii. Per batch nya 6 input
8  # iii. Neuron 5
9
10 import numpy as np
11
12 # Inisialisasi variabel inputs
13 inputs = [[-6.6, 7.2, 2.1, 6.8, 5.4, 1.9, 1.0, 6.9, 1.1, -4.3],
14           [3.5, -8.6, 5.3, 6.8, 2.5, 2.0, 4.7, 1.3, -9.5, 7.0],
15           [2.0, 7.9, -8.1, 2.3, 9.9, 6.3, 4.3, -8.8, 6.7, 8.2],
16           [1.9, 4.7, 9.1, -3.3, 6.8, 5.2, -7.8, 5.0, 2.8, 3.7],
17           [3.0, 4.8, 5.5, 8.6, -8.5, -7.0, 5.8, 5.0, 3.9, 3.6],
18           [9.0, 7.5, 9.6, -2.2, 8.8, 2.3, -6.6, 8.6, 2.8, 1.7]]
19
20 # Inisialisasi variabel weights
21 weights = [[1.4, 2.0, 2.8, -4.2, 6.0, 2.4, -3.8, 1.2, 6.2, 5.1],
22           [6.0, 6.6, 2.8, -8.2, 6.2, 1.6, 5.1, 3.6, -7.1, 4.0],
23           [8.6, -2.3, 5.1, 7.4, 1.4, -2.4, 7.5, 5.7, 1.9, 8.4],
24           [5.6, 4.0, -6.5, 8.2, 2.6, 1.8, -7.3, 7.8, 3.7, 5.9],
25           [6.0, 3.0, 6.6, 1.3, -3.2, 7.1, 7.4, 9.4, 1.1, -7.6]]
26
27 # Inisialisasi variabel biases
28 biases = [2.0, 4, 0.8, 5.2, 3]
29
30 # Penghitungan output menggunakan rumus dot product vector
31 output = np.dot(inputs, np.array(weights).T) + biases
32
33 # Menampilkan output
34 print(output)

[[ 10.81   3.49   4.31  81.83 110.06]
 [ -43.72  70.12 210.14   3.79  31.57]
 [ 119.24  73.54  37.98 129.73 -104.57]
 [ 178.55 122.58  31.76 103.6   63.67]
 [ -46.22 -31.05 221.8   73.6   127.23]
 [ 185.47 194.88 119.55 167.83 150.33]]
[Finished in 0.8s]
```

b) Penjelasan

- Langkah pertama yaitu mengimpor NumPy (library Python) yang bisa digunakan untuk melakukan operasi vektor dan matriks. Lalu mengisi nilai dari setiap variabel inputs, weights, dan biases sesuai ketentuan soal yaitu per batch-nya 6 input dengan jumlah tiap input layer 10 sehingga inputs = 6 * 10 dan 5 neuron. Maksudnya, pada variabel inputs berisi sebanyak 10 angka setiap 1 barisnya dengan total 6 baris. Pada variabel weights berisi sebanyak 10 angka setiap 1 barisnya (sesuai banyaknya angka dalam 1 baris inputs) dengan total 5 baris (karena 5 neuron). Karena ada 5 neuron maka isi variabel biases sebanyak 5 angka.

```

10 import numpy as np
11
12 # Inisialisasi variabel inputs
13 inputs = [[-6.6, 7.2, 2.1, 6.8, 5.4, 1.9, 1.0, 6.9, 1.1, -4.3],
14           [3.5, -8.6, 5.3, 6.8, 2.5, 2.0, 4.7, 1.3, -9.5, 7.0],
15           [2.0, 7.9, -8.1, 2.3, 9.9, 6.3, 4.3, -8.8, 6.7, 8.2],
16           [1.9, 4.7, 9.1, -3.3, 6.8, 5.2, -7.8, 5.0, 2.8, 3.7],
17           [3.0, 4.8, 5.5, 8.6, -8.5, -7.0, 5.8, 5.0, 3.9, 3.6],
18           [9.0, 7.5, 9.6, -2.2, 8.8, 2.3, -6.6, 8.6, 2.8, 1.7]]
19
20 # Inisialisasi variabel weights
21 weights = [[1.4, 2.0, 2.8, -4.2, 6.0, 2.4, -3.8, 1.2, 6.2, 5.1],
22            [6.0, 6.6, 2.8, -8.2, 6.2, 1.6, 5.1, 3.6, -7.1, 4.0],
23            [8.6, -2.3, 5.1, 7.4, 1.4, -2.4, 7.5, 5.7, 1.9, 8.4],
24            [5.6, 4.0, -6.5, 8.2, 2.6, 1.8, -7.3, 7.8, 3.7, 5.9],
25            [6.0, 3.0, 6.6, 1.3, -3.2, 7.1, 7.4, 9.4, 1.1, -7.6]]
26
27 # Inisialisasi variabel biases
28 biases = [2.0, 4, 0.8, 5.2, 3]

```

- Melakukan operasi penghitungan.

```

30 # Penghitungan output menggunakan rumus dot product vector
31 output = np.dot(inputs, np.array(weights).T) + biases

```

Transpose weights terlebih dahulu agar ordonya sesuai dengan variabel inputs sehingga operasi perhitungan bisa dilakukan.

transpose

$$\begin{array}{c}
 \text{weights} \\
 5 \times 10
 \end{array}
 \begin{bmatrix}
 1.4 & 2.0 & 2.8 & -4.2 & 6.0 & 2.4 & -3.8 & 1.2 & 6.2 & 5.1 \\
 6.0 & 6.6 & 2.8 & -8.2 & 6.2 & 1.6 & 5.1 & 3.6 & -7.1 & 4.0 \\
 8.6 & -2.3 & 5.1 & 7.4 & 1.4 & -2.4 & 7.5 & 5.7 & 1.9 & 8.4 \\
 5.6 & 4.0 & -6.5 & 8.2 & 2.6 & 1.8 & -7.3 & 7.8 & 3.7 & 5.9 \\
 6.0 & 3.0 & 6.6 & 1.3 & -3.2 & 7.1 & 7.4 & 9.4 & 1.1 & -7.6
 \end{bmatrix}
 \begin{array}{c}
 T
 \end{array}
 =
 \begin{array}{c}
 \text{weights} \\
 10 \times 5
 \end{array}
 \begin{bmatrix}
 1.4 & 6.0 & 8.6 & 5.6 & 6.0 \\
 2.0 & 6.6 & -2.3 & 4.0 & 3.0 \\
 2.8 & 2.8 & 5.1 & -6.5 & 6.6 \\
 -4.2 & -8.2 & 7.4 & 8.2 & 1.3 \\
 6.0 & 6.2 & 1.4 & 2.6 & -3.2 \\
 2.4 & 1.6 & -2.4 & 1.8 & 7.1 \\
 -3.8 & 5.1 & 7.5 & -7.3 & 7.4 \\
 1.2 & 3.6 & 5.7 & 7.8 & 9.4 \\
 6.2 & -7.1 & 1.9 & 3.7 & 1.1 \\
 5.1 & 4.0 & 8.4 & 5.9 & -7.6
 \end{bmatrix}$$

inputs
6 * 10

$$\begin{bmatrix} -6.6 & 7.2 & 2.1 & 6.8 & 5.4 & 1.9 & 1.0 & 6.9 & 1.1 & -4.3 \\ 3.5 & -8.6 & 5.3 & 6.8 & 2.5 & 2.0 & 4.7 & 1.3 & -9.5 & 7.0 \\ 2.0 & 7.9 & -8.1 & 2.3 & 9.9 & 6.3 & 4.3 & -8.8 & 6.7 & 8.2 \\ 1.9 & 4.7 & 9.1 & -3.3 & 6.8 & 5.2 & -7.8 & 5.0 & 2.8 & 3.7 \\ 3.0 & 4.8 & 5.5 & 8.6 & -8.5 & -7.0 & 5.8 & 5.0 & 3.9 & 3.6 \\ 9.0 & 7.5 & 9.6 & -2.2 & 8.8 & 2.3 & -6.6 & 8.6 & 2.8 & 1.7 \end{bmatrix} * \begin{bmatrix} 1.4 & 6.0 & 8.6 & 5.6 & 6.0 \\ 2.0 & 6.6 & -2.3 & 4.0 & 3.0 \\ 2.8 & 2.8 & 5.1 & -6.5 & 6.6 \\ -4.2 & -8.2 & 7.4 & 8.2 & 1.3 \\ 6.0 & 6.2 & 1.4 & 2.6 & -3.2 \\ 2.4 & 1.6 & -2.4 & 1.8 & 7.1 \\ -3.8 & 5.1 & 7.5 & -7.3 & 7.4 \\ 1.2 & 3.6 & 5.7 & 7.8 & 9.4 \\ 6.2 & -7.1 & 1.9 & 3.7 & 1.1 \\ 5.1 & 4.0 & 8.4 & 5.9 & -7.6 \end{bmatrix}$$

=

$$\begin{bmatrix} 8.81 & -0.51 & 3.51 & 76.63 & 107.06 \\ -45.72 & 66.12 & 209.34 & -1.41 & 28.57 \\ 117.24 & 69.54 & 37.18 & 124.53 & -107.57 \\ 176.55 & 118.58 & 30.96 & 98.4 & 60.67 \\ -48.22 & -35.05 & 221.0 & 68.4 & 124.23 \\ 183.47 & 190.88 & 118.75 & 162.63 & 147.33 \end{bmatrix}$$

tambahkan dengan bias, np.dot + biases

$$\begin{bmatrix} 8.81 & -0.51 & 3.51 & 76.63 & 107.06 \\ -45.72 & 66.12 & 209.34 & -1.41 & 28.57 \\ 117.24 & 69.54 & 37.18 & 124.53 & -107.57 \\ 176.55 & 118.58 & 30.96 & 98.4 & 60.67 \\ -48.22 & -35.05 & 221.0 & 68.4 & 124.23 \\ 183.47 & 190.88 & 118.75 & 162.63 & 147.33 \end{bmatrix} + [2.0, 4, 0.8, 5.2, 3]$$

=

$$\begin{bmatrix} 10.81 & 3.49 & 4.31 & 81.83 & 110.06 \\ -43.72 & 70.12 & 210.14 & 3.79 & 31.57 \\ 119.24 & 73.54 & 37.98 & 129.73 & -104.57 \\ 178.55 & 122.58 & 31.76 & 103.6 & 63.67 \\ -46.22 & -31.05 & 221.8 & 73.6 & 127.23 \\ 185.47 & 194.88 & 119.55 & 167.83 & 150.33 \end{bmatrix}$$

- Menampilkan output.

```
33 # Menampilkan output
34 print(output)

[[ 10.81   3.49   4.31  81.83 110.06]
 [-43.72  70.12 210.14   3.79  31.57]
 [119.24  73.54  37.98 129.73 -104.57]
 [178.55 122.58  31.76  83.6   63.67]
 [-46.22 -31.05 221.8   73.6  127.23]
 [185.47 194.88 119.55 167.83 150.33]]
[Finished in 0.8s]
```


UTS 2

A. Multi Neuron Batch Input

- i. Input layer feature 10
 - ii. Per batch nya 6 input
 - iii. Hidden layer 1, 5 neuron
 - iv. Hidden layer 2, 3 neuron
- a) Source Code

```
UTS2_1a_MultiNeuronBatchInput_031_AlvinFebrianto.py x
1  # Alvin Febrianto | 21091397031 | 2021 A - D4 Manajemen Informatika
2  # a. Multi Neuron Batch Input
3  #   i.   Input layer feature 10
4  #   ii.  Per batch nya 6 input
5  #   iii. Hidden layer 1, 5 neuron
6  #   iv.  Hidden layer 2, 3 neuron
7
8  # Inisialisasi library NumPy
9  import numpy as np
10
11 # Inisialisasi variabel inputs
12 inputs = [[-6.6, 7.2, 2.1, 6.8, 5.4, 1.9, 1.0, 6.9, 1.1, -4.3],
13           [3.5, -8.6, 5.3, 6.8, 2.5, 2.0, 4.7, 1.3, -9.5, 7.0],
14           [2.0, 7.9, -8.1, 2.3, 9.9, 6.3, 4.3, -8.8, 6.7, 8.2],
15           [1.9, 4.7, 9.1, -3.3, 6.8, 5.2, -7.8, 5.0, 2.8, 3.7],
16           [3.0, 4.8, 5.5, 8.6, -8.5, -7.0, 5.8, 5.0, 3.9, 3.6],
17           [9.0, 7.5, 9.6, -2.2, 8.8, 2.3, -6.6, 8.6, 2.8, 1.7]]
18
19 # Inisialisasi variabel weights1 [hidden layer 1, 5 neuron]
20 weights1 = [[1.4, 2.0, 2.8, -4.2, 6.0, 2.4, -3.8, 1.2, 6.2, 5.1],
21            [6.0, 6.6, 2.8, -8.2, 6.2, 1.6, 5.1, 3.6, -7.1, 4.0],
22            [8.6, -2.3, 5.1, 7.4, 1.4, -2.4, 7.5, 5.7, 1.9, 8.4],
23            [5.6, 4.0, -6.5, 8.2, 2.6, 1.8, -7.3, 7.8, 3.7, 5.9],
24            [6.0, 3.0, 6.6, 1.3, -3.2, 7.1, 7.4, 9.4, 1.1, -7.6]]
25
26 # Inisialisasi variabel biases1 [hidden layer 1, 5 neuron]
27 biases1 = [2.0, 4, 0.8, 5.2, 3]
28
29 # Inisialisasi variabel weights2 [hidden layer 2, 3 neuron]
30 weights2 = [[2.2, -2.0, 2.8, -4.2, 6.0],
31            [3.5, 2.8, -8.2, 1.6, -7.1],
32            [-2.3, 5.1, 1.4, -2.4, 1.9]]
33
34 # Inisialisasi variabel biases2 [hidden layer 2, 3 neuron]
35 biases2 = [1.8, 6, 4.5]
36
37 # Penghitungan output pada layer 1
38 layer1_outputs = np.dot(inputs, np.array(weights1).T) + biases1
39
40 # Penghitungan output pada layer 2
41 layer2_outputs = np.dot(layer1_outputs, np.array(weights2).T) + biases2
42
43 # Menampilkan output
44 print(layer2_outputs)
```


b) Penjelasan

- Langkah pertama yaitu mengimpor NumPy (library Python) yang bisa digunakan untuk melakukan operasi vektor dan matriks. Lalu mengisi nilai dari setiap variabel inputs, weights, dan biases sesuai ketentuan soal yaitu per batch-nya 6 input dengan jumlah tiap input layer 10 sehingga inputs = 6 * 10 dan 5 neuron pada hidden layer 1 serta 3 neuron pada hidden layer 2. Maksudnya, pada variabel inputs berisi sebanyak 10 angka setiap 1 barisnya dengan total 6 baris. Pada variabel weights1 berisi sebanyak 10 angka setiap 1 barisnya (sesuai banyaknya angka dalam 1 baris inputs) dengan total 5 baris (karena 5 neuron). Karena ada 5 neuron maka isi variabel biases1 sebanyak 5 angka. Pada variabel weights2 (berada di hidden layer 2) susunannya mengikuti ordo dari output pada hidden layer 1 sehingga setiap 1 barisnya ada sebanyak 5 angka dengan total 3 baris. Karena ada 3 neuron di hidden layer 2 maka berisi sebanyak 3 angka.

```
8 # Inisialisasi library NumPy
9 import numpy as np
10
11 # Inisialisasi variabel inputs
12 inputs = [[-6.6, 7.2, 2.1, 6.8, 5.4, 1.9, 1.0, 6.9, 1.1, -4.3],
13           [3.5, -8.6, 5.3, 6.8, 2.5, 2.0, 4.7, 1.3, -9.5, 7.0],
14           [2.0, 7.9, -8.1, 2.3, 9.9, 6.3, 4.3, -8.8, 6.7, 8.2],
15           [1.9, 4.7, 9.1, -3.3, 6.8, 5.2, -7.8, 5.0, 2.8, 3.7],
16           [3.0, 4.8, 5.5, 8.6, -8.5, -7.0, 5.8, 5.0, 3.9, 3.6],
17           [9.0, 7.5, 9.6, -2.2, 8.8, 2.3, -6.6, 8.6, 2.8, 1.7]]
18
19 # Inisialisasi variabel weights1 [hidden layer 1, 5 neuron]
20 weights1 = [[1.4, 2.0, 2.8, -4.2, 6.0, 2.4, -3.8, 1.2, 6.2, 5.1],
21             [6.0, 6.6, 2.8, -8.2, 6.2, 1.6, 5.1, 3.6, -7.1, 4.0],
22             [8.6, -2.3, 5.1, 7.4, 1.4, -2.4, 7.5, 5.7, 1.9, 8.4],
23             [5.6, 4.0, -6.5, 8.2, 2.6, 1.8, -7.3, 7.8, 3.7, 5.9],
24             [6.0, 3.0, 6.6, 1.3, -3.2, 7.1, 7.4, 9.4, 1.1, -7.6]]
25
26 # Inisialisasi variabel biases1 [hidden layer 1, 5 neuron]
27 biases1 = [2.0, 4, 0.8, 5.2, 3]
28
29 # Inisialisasi variabel weights2 [hidden layer 2, 3 neuron]
30 weights2 = [[2.2, -2.0, 2.8, -4.2, 6.0],
31             [3.5, 2.8, -8.2, 1.6, -7.1],
32             [-2.3, 5.1, 1.4, -2.4, 1.9]]
33
34 # Inisialisasi variabel biases2 [hidden layer 2, 3 neuron]
35 biases2 = [1.8, 6, 4.5]
```

- Melakukan operasi penghitungan pada hidden layer 1.

```
37 # Penghitungan output pada layer 1
38 layer1_outputs = np.dot(inputs, np.array(weights1).T) + biases1
```

Transpose weights terlebih dahulu agar ordonya sesuai dengan variabel inputs sehingga operasi perhitungan bisa dilakukan.

transpose

$$\begin{array}{c} \text{weights} \\ 5 \times 10 \end{array} \begin{bmatrix} 1.4 & 2.0 & 2.8 & -4.2 & 6.0 & 2.4 & -3.8 & 1.2 & 6.2 & 5.1 \\ 6.0 & 6.6 & 2.8 & -8.2 & 6.2 & 1.6 & 5.1 & 3.6 & -7.1 & 4.0 \\ 8.6 & -2.3 & 5.1 & 7.4 & 1.4 & -2.4 & 7.5 & 5.7 & 1.9 & 8.4 \\ 5.6 & 4.0 & -6.5 & 8.2 & 2.6 & 1.8 & -7.3 & 7.8 & 3.7 & 5.9 \\ 6.0 & 3.0 & 6.6 & 1.3 & -3.2 & 7.1 & 7.4 & 9.4 & 1.1 & -7.6 \end{bmatrix}^T = \begin{array}{c} \text{weights} \\ 10 \times 5 \end{array} \begin{bmatrix} 1.4 & 6.0 & 8.6 & 5.6 & 6.0 \\ 2.0 & 6.6 & -2.3 & 4.0 & 3.0 \\ 2.8 & 2.8 & 5.1 & -6.5 & 6.6 \\ -4.2 & -8.2 & 7.4 & 8.2 & 1.3 \\ 6.0 & 6.2 & 1.4 & 2.6 & -3.2 \\ 2.4 & 1.6 & -2.4 & 1.8 & 7.1 \\ -3.8 & 5.1 & 7.5 & -7.3 & 7.4 \\ 1.2 & 3.6 & 5.7 & 7.8 & 9.4 \\ 6.2 & -7.1 & 1.9 & 3.7 & 1.1 \\ 5.1 & 4.0 & 8.4 & 5.9 & -7.6 \end{bmatrix}$$

$$\begin{array}{c} \text{inputs} \\ 6 \times 10 \end{array} \begin{bmatrix} -6.6 & 7.2 & 2.1 & 6.8 & 5.4 & 1.9 & 1.0 & 6.9 & 1.1 & -4.3 \\ 3.5 & -8.6 & 5.3 & 6.8 & 2.5 & 2.0 & 4.7 & 1.3 & -9.5 & 7.0 \\ 2.0 & 7.9 & -8.1 & 2.3 & 9.9 & 6.3 & 4.3 & -8.8 & 6.7 & 8.2 \\ 1.9 & 4.7 & 9.1 & -3.3 & 6.8 & 5.2 & -7.8 & 5.0 & 2.8 & 3.7 \\ 3.0 & 4.8 & 5.5 & 8.6 & -8.5 & -7.0 & 5.8 & 5.0 & 3.9 & 3.6 \\ 9.0 & 7.5 & 9.6 & -2.2 & 8.8 & 2.3 & -6.6 & 8.6 & 2.8 & 1.7 \end{bmatrix} * \begin{array}{c} \text{weights} \\ 10 \times 5 \end{array} \begin{bmatrix} 1.4 & 6.0 & 8.6 & 5.6 & 6.0 \\ 2.0 & 6.6 & -2.3 & 4.0 & 3.0 \\ 2.8 & 2.8 & 5.1 & -6.5 & 6.6 \\ -4.2 & -8.2 & 7.4 & 8.2 & 1.3 \\ 6.0 & 6.2 & 1.4 & 2.6 & -3.2 \\ 2.4 & 1.6 & -2.4 & 1.8 & 7.1 \\ -3.8 & 5.1 & 7.5 & -7.3 & 7.4 \\ 1.2 & 3.6 & 5.7 & 7.8 & 9.4 \\ 6.2 & -7.1 & 1.9 & 3.7 & 1.1 \\ 5.1 & 4.0 & 8.4 & 5.9 & -7.6 \end{bmatrix}$$

$$= \begin{bmatrix} 8.81 & -0.51 & 3.51 & 76.63 & 107.06 \\ -45.72 & 66.12 & 209.34 & -1.41 & 28.57 \\ 117.24 & 69.54 & 37.18 & 124.53 & -107.57 \\ 176.55 & 118.58 & 30.96 & 98.4 & 60.67 \\ -48.22 & -35.05 & 221.0 & 68.4 & 124.23 \\ 183.47 & 190.88 & 118.75 & 162.63 & 147.33 \end{bmatrix}$$

tambahkan dengan bias, $np.dot + \text{biases}$

$$\begin{bmatrix} 8.81 & -0.51 & 3.51 & 76.63 & 107.06 \\ -45.72 & 66.12 & 209.34 & -1.41 & 28.57 \\ 117.24 & 69.54 & 37.18 & 124.53 & -107.57 \\ 176.55 & 118.58 & 30.96 & 98.4 & 60.67 \\ -48.22 & -35.05 & 221.0 & 68.4 & 124.23 \\ 183.47 & 190.88 & 118.75 & 162.63 & 147.33 \end{bmatrix} + [2.0, 4, 0.8, 5.2, 3]$$

$$= \begin{bmatrix} 10.81 & 3.49 & 4.31 & 81.83 & 110.06 \\ -43.72 & 70.12 & 210.14 & 3.79 & 31.57 \\ 119.24 & 73.54 & 37.98 & 129.73 & -104.57 \\ 178.55 & 122.58 & 31.76 & 103.6 & 63.67 \\ -46.22 & -31.05 & 221.8 & 73.6 & 127.23 \\ 185.47 & 194.88 & 119.55 & 167.83 & 150.33 \end{bmatrix}$$

- Melakukan operasi penghitungan pada hidden layer 2.

```
40 # Penghitungan output pada layer 2
41 layer2_outputs = np.dot(layer1_outputs, np.array(weights2).T) + biases2
```

Transpose weights2 lebih dahulu agar sesuai dengan ordo dari output hidden layer 1.

transpose

$$\begin{array}{c} \text{weights2} \\ 3 \times 5 \end{array} \begin{array}{c} \text{weights2} \\ 5 \times 3 \end{array}$$

$$\begin{bmatrix} 2.2 & -2.0 & 2.8 & -4.2 & 6.0 \\ 3.5 & 2.8 & -8.2 & 1.6 & -7.1 \\ -2.3 & 5.1 & 1.4 & -2.4 & 1.9 \end{bmatrix}^T = \begin{bmatrix} 2.2 & 3.5 & -2.3 \\ -2.0 & 2.8 & 5.1 \\ 2.8 & -8.2 & 1.4 \\ -4.2 & 1.6 & -2.4 \\ 6.0 & -7.1 & 1.9 \end{bmatrix}$$

$$\begin{array}{c} \text{hidden layer 1} \\ 6 \times 5 \end{array} \begin{array}{c} \text{weights2} \\ 5 \times 3 \end{array}$$

$$\begin{bmatrix} 10.81 & 3.49 & 4.31 & 81.83 & 110.06 \\ -43.72 & 70.12 & 210.14 & 3.79 & 31.57 \\ 119.24 & 73.54 & 37.98 & 129.73 & -104.57 \\ 178.55 & 122.58 & 31.76 & 103.6 & 63.67 \\ -46.22 & -31.05 & 221.8 & 73.6 & 127.23 \\ 185.47 & 194.88 & 119.55 & 167.83 & 150.33 \end{bmatrix} * \begin{bmatrix} 2.2 & 3.5 & -2.3 \\ -2.0 & 2.8 & 5.1 \\ 2.8 & -8.2 & 1.4 \\ -4.2 & 1.6 & -2.4 \\ 6.0 & -7.1 & 1.9 \end{bmatrix}$$

$$= \begin{bmatrix} 345.544 & -638.233 & 11.692 \\ 525.47 & -1897.915 & 803.251 \\ -950.694 & 1261.831 & -356.061 \\ 183.478 & 421.42 & 131.29 \\ 1035.716 & -2853.043 & 323.568 \\ 550.108 & -584.316 & 617.512 \end{bmatrix}$$

tambahkan dengan bias, np.dot + biases2

$$\begin{bmatrix} 345.544 & -638.233 & 11.692 \\ 525.47 & -1897.915 & 803.251 \\ -950.694 & 1261.831 & -356.061 \\ 183.478 & 421.42 & 131.29 \\ 1035.716 & -2853.043 & 323.568 \\ 550.108 & -584.316 & 617.512 \end{bmatrix} + [1.8, 6, 4.5]$$

$$= \begin{bmatrix} 347.344 & -632.233 & 16.192 \\ 527.27 & -1891.915 & 807.751 \\ -948.894 & 1267.831 & -351.561 \\ 185.278 & 427.42 & 135.79 \\ 1037.516 & -2847.043 & 328.068 \\ 551.908 & -578.316 & 622.012 \end{bmatrix}$$

- Menampilkan output.

```
43 # Menampilkan output
44 print(layer2_outputs)
```

```
[[ 347.344 -632.233  16.192]
 [ 527.27  -1891.915 807.751]
 [-948.894 1267.831 -351.561]
 [ 185.278  427.42  135.79 ]
 [1037.516 -2847.043  328.068]
 [ 551.908 -578.316  622.012]]
[Finished in 0.6s]
```