

**LAPORAN KECERDASAN BUATAN
UJIAN TENGAH SEMESTER**



Disusun oleh :
Muvidha Fatmawati Putri (21091397057)
A2021 MI

**PROGRAM STUDI D4 MANAJEMEN INFORMATIKA
FAKULTAS VOKASI
UNIVERSITAS NEGERI SURABAYA**

2022

1. UTS 1

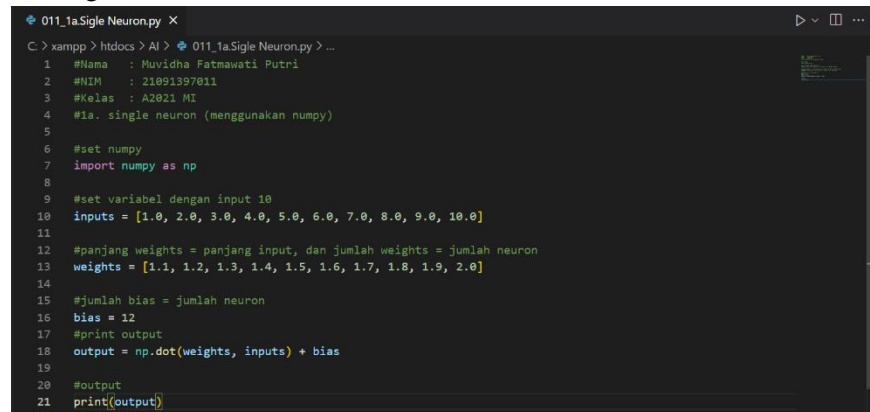
Buat dokumentasi dan jelaskan

a. Single Neuron

Input layer feature 10

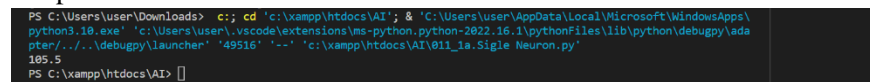
Neuron 1

- Kodingan :



```
011_1aSingle Neuron.py X
C:\xampp\htdocs\AI> 011_1aSingle Neuron.py > ...
1 #Nama : Muvridha Fatmawati Putri
2 #NIM : 21091397011
3 #Kelas : A2021 MI
4 #1a. single neuron (menggunakan numpy)
5
6 #set numpy
7 import numpy as np
8
9 #set variabel dengan input 10
10 inputs = [1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0]
11
12 #panjang weights = panjang input, dan jumlah weights = jumlah neuron
13 weights = [1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0]
14
15 #jumlah bias = jumlah neuron
16 bias = 12
17 #print output
18 output = np.dot(weights, inputs) + bias
19
20 #output
21 print(output)
```

- Output :



```
PS C:\Users\user\Downloads> cd 'c:\xampp\htdocs\AI'; & 'C:\Users\user\AppData\Local\Microsoft\WindowsApps\python3.10.exe' 'c:\Users\user\.vscode\extensions\ms-python.python-2022.16.1\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '49516' '--' 'c:\xampp\htdocs\AI\011_1a.Single Neuron.py'
105.5
PS C:\xampp\htdocs\AI>
```

- Penjelasan :

1. Program dimulai dengan meng-set numpy dengan np pada baris 6.
2. Membuat variabel input pada baris 9 (untuk menginput nilai neuron)
3. Bobot pada baris 12 (bobot neuron)
4. Bias pada baris 15 (untuk menyeimbangkan keluaran positif atau negatif).
5. Mengisikan setiap variable dengan nilai yang telah diinputkan, yaitu 10 value untuk input layers, dan Weight serta bias = 1, karena single neuron.
6. Baris 17 mendefinisikan output dari perhitungan produk dalam dot.product (untuk mengalikan nilai input dengan bobot) dan ditambahkan ke nilai bias .
7. Baris 20 terdapat fungsi print yang memanggil hasil perhitungan dengan output.

- Step perhitungan

Perhitungan inputs dan weights dapat menggunakan rumus dot product vector,

Rumus dot Product Vektor

$$\vec{a} \cdot \vec{b} = \sum_{i=1}^n a_i b_i = a_1 b_1 + a_2 b_2 + \dots + a_n b_n$$

misal,

$$\begin{matrix} \text{inputs} = \vec{a} \\ \text{weights} = \vec{b} \end{matrix} \quad \text{maka} \quad \begin{matrix} \vec{a} = [1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0] \\ \vec{b} = [1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0] \end{matrix}$$

$$\begin{aligned} \vec{a} \cdot \vec{b} &= (1.0 \times 1.1) + (2.0 \times 1.2) + (3.0 \times 1.3) + (4.0 \times 1.4) + (5.0 \times 1.5) + (6.0 \times 1.6) + \\ &\quad (7.0 \times 1.7) + (8.0 \times 1.8) + (9.0 \times 1.9) + (10.0 \times 2.0) \\ &= 1.1 + 2.4 + 3.9 + 5.6 + 7.5 + 9.6 + 11.9 + 14.4 + 17.1 + 20.0 \\ &= 93.5 \end{aligned}$$

Setelah mengalikan vektor a dan vektor b maka hasil ditambahkan dengan bias = 12

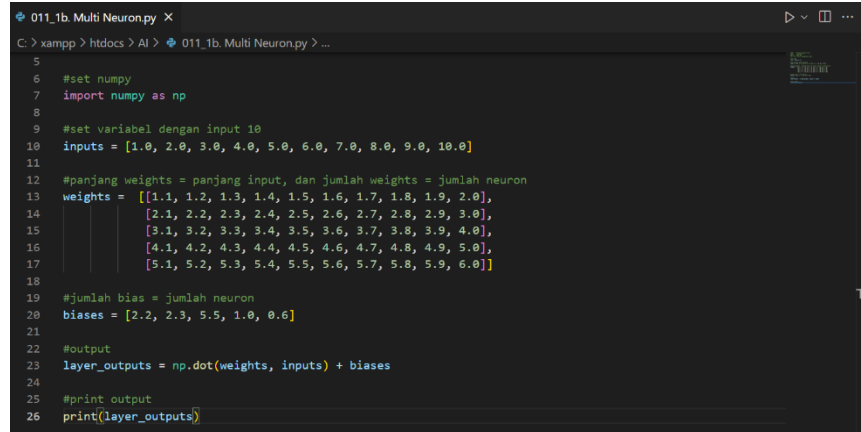
$$\text{np.dot(weights, inputs) + bias}$$

$$= 93.5 + 12$$

$$= 105.5 //$$

- b. Multi Neuron
Input layer feature 10
Neuron 5

- Kodingan :

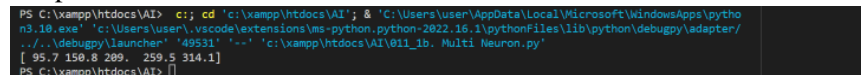


```

5
6 #set numpy
7 import numpy as np
8
9 #set variabel dengan input 10
10 inputs = [1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0]
11
12 #panjang weights = panjang input, dan jumlah weights = jumlah neuron
13 weights = [[1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0],
14            [2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0],
15            [3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 4.0],
16            [4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 5.0],
17            [5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 6.0]]
18
19 #jumlah bias = jumlah neuron
20 biases = [2.2, 2.3, 5.5, 1.0, 0.6]
21
22 #output
23 layer_outputs = np.dot(weights, inputs) + biases
24
25 #print output
26 print(layer_outputs)

```

- Output :



```

PS C:\xampp\htdocs\AI> cd "c:\xampp\htdocs\AI"; & "C:\Users\User\AppData\Local\Microsoft\WindowsApps\python3.10.exe" "c:\Users\User\.vscode\extensions\ms-python.python-2022.16.1\pythonFiles\lib\python\debugpy\adapter/../../debugpy/launcher" "49531" "--" "c:\xampp\htdocs\AI\01_1b_Multi_Neuron.py"
[ 95.7 150.8 209. 259.5 314.1]
PS C:\xampp\htdocs\AI>

```

- Penjelasan :

1. Seperti Single Neuron, Multi Neuron dimulai dengan set numpy pada baris 6.
 2. Set variabel input (baris 10)
 3. Bobot (baris 12)
 4. Bias (baris 19).
 5. Untuk lapisan input = 10, bobot dan bias = 5 (neuron = 5).
 6. Kemudian, pada baris 22 , kami membuat produk titik perhitungan dot.product (mengkalikan bobot dengan input) pada lapisan output dan menambahkan bias .
 7. Selanjutnya pada baris ke - 25 , hasil perhitungan output layer dipanggil dengan perintah print .
- Step perhitungan multi neuron hampir sama dengan single neuron tetapi dihitung per neuron satu per satu terlebih dahulu

Rumus dot Product Vektor

$$\vec{a} \cdot \vec{b} = \sum_{i=1}^n a_i b_i = a_1 b_1 + a_2 b_2 + \dots + a_n b_n$$

misal,

inputs = \vec{a}
weights = \vec{b} } Perhitungan urutan sesuai neuron

• Neuron 1 : $\vec{a} = [1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0]$

$$\vec{b} = [1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0]$$

$$\begin{aligned} \therefore \vec{a} \cdot \vec{b} &= (1.0 \times 1.1) + (2.0 \times 1.2) + (3.0 \times 1.3) + (4.0 \times 1.4) + (5.0 \times 1.5) + (6.0 \times 1.6) + \\ &\quad (7.0 \times 1.7) + (8.0 \times 1.8) + (9.0 \times 1.9) + (10.0 \times 2.0) \\ &= 1.1 + 2.4 + 3.9 + 5.6 + 7.5 + 9.6 + 11.9 + 14.4 + 17.1 + 20.0 \\ &= 93.5 \end{aligned}$$

Setelah itu ditambahkan dengan bias

$$\rightarrow \text{np.dot(weights, inputs)} + \text{bias}$$

$$= 93.5 + 2.5$$

$$= 95.7 //$$

• Neuron 2 : $\vec{a} = [1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0]$

$$\vec{b} = [2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0]$$

$$\therefore \vec{a} \cdot \vec{b} = 148.5$$

$$\rightarrow \text{np.dot(weights, inputs)} + \text{bias}$$

$$= 148.5 + 2.3 = 150.8 //$$

• Neuron 3 : $\vec{a} = [1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0]$

$$\vec{b} = [3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 4.0]$$

$$\therefore \vec{a} \cdot \vec{b} = 203.5$$

$$\rightarrow \text{np.dot(weights, inputs)} + \text{bias}$$

$$= 203.5 + 5.5$$

$$= 209.0 //$$

• Neuron 4 : $\vec{a} = [1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0]$

$$\vec{b} = [4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 5.0]$$

$$\therefore \vec{a} \cdot \vec{b} = 258.5$$

$$\rightarrow \text{np.dot(weights, inputs)} + \text{bias}$$

$$= 258.5 + 1.0$$

$$= 259.5 //$$

• Neuron 5 : $\vec{a} = [1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0]$

$$\vec{b} = [5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 6.0]$$

$$\therefore \vec{a} \cdot \vec{b} = 313.5$$

$$\rightarrow \text{np.dot(weights, inputs)} + \text{bias}$$

$$= 313.5 + 0.6$$

$$= 314.1 //$$

Maka hasil berurutan sesuai urutan neuron yaitu [95.7 150.8 209.0 259.5 314.1]

c. Multi Neuron Batch Input

Input layer Feature 10

Per batch nya 6 input

Neuron 5

- Kodongan :

```
011_1cMulti Neuron Batch Input.py X
C:\> xampp > htdocs > AI > 011_1cMulti Neuron Batch Input.py > ...
1 #Nama : Muvidha Fatmawati Putri
2 #NIM : 21091397011
3 #Kelas : A2021 MI
4 #ic. multi neuron batch input (menggunakan numpy)
5
6 #set numpy
7 import numpy as np
8
9 #set variabel dengan matriks 6x10 (input 10 dan batch 6)
10 inputs = [[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0],
11 [1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0],
12 [2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0],
13 [3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 4.0],
14 [4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 5.0],
15 [5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 6.0]]
16
17 #panjang weights = panjang input, dan jumlah weights = jumlah neuron
18 weights = [[1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0],
19 [2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0],
20 [3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 4.0],
21 [4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 5.0],
22 [5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 6.0]]
23 biases = [2.2, 2.3, 5.5, 1.0, 0.6]
24
25 #ouputs
26 layer_outputs = np.dot(inputs, np.array(weights).T) + biases
27
28 #print outputs
29 print(layer_outputs)
```

- Output :

```
PS C:\xampp\htdocs\AI> cd "c:\xampp\htdocs\AI"; & "C:\Users\User\AppData\Local\Microsoft\WindowsApps\python3.10.exe" "c:\Users\User\.vscode\extensions\ms-python.python-2022.16.1\pythonFiles\lib\python\debugpy\adapter/.../debugpy/launcher" "49539" "--" "c:\xampp\htdocs\AI\011_1cMulti Neuron Batch Input.py"
[[ 11.55  17.15  25.85  26.85  31.95]
 [ 27.05  42.65  61.35  72.35  87.45]
 [ 42.55  68.15  96.85  117.85  142.95]
 [ 58.05  93.65  132.35  163.35  198.45]
 [ 73.55  119.15  167.85  208.85  253.95]
 [ 89.05  144.65  203.35  254.35  309.45]]
PS C:\xampp\htdocs\AI>
```

- Penjelasan :

1. Pada baris 5, pertama - tama kita meng-set numpy (yang nantinya akan digunakan dalam perhitungan dot product).
2. Meng-set variabel input (baris 9)
3. Bobot (baris 17)
4. Bias (baris 23).
5. Untuk lapisan input = 10, input per batch = 6, bobot dan bias = 5 (neuron = 5).
6. Baris 25 melakukan perhitungan hasil kali dalam dot.product (mengkalikan setiap nilai input dengan bobot transposnya (T)) dan menambahkan setiap nilai bias .
7. Ini memanggil fungsi " Lapisan PrintOutput " pada baris 28 dan menampilkan hasil perhitungan dalam bentuk Himpunan.

- Step perhitungan
Dimisalkan

Inputs = matriks a
Weights = matriks b
Maka,

```
a = [[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0],  
      [1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0],  
      [2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0],  
      [3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 4.0],  
      [4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 5.0],  
      [5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 6.0]]  
  
b = [[1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0],  
      [2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0],  
      [3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 4.0],  
      [4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 5.0],  
      [5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 6.0]]
```

Untuk perhitungan multi neuron batch input dapat menggunakan perkalian matriks, namun baris dan kolom dari matriks a dan b tidak sama, sehingga matriks b harus di transposkan (T) terlebih dahulu

- ✓ Hasil transpose matriks b

```
b transpose = [[1.1, 2.1, 3.1, 4.1, 5.1],  
               [1.2, 2.2, 3.2, 4.2, 5.2],  
               [1.3, 2.3, 3.3, 4.3, 5.3],  
               [1.4, 2.4, 3.4, 4.4, 5.4],  
               [1.5, 2.5, 3.5, 4.5, 5.5],  
               [1.6, 2.6, 3.6, 4.6, 5.6],  
               [1.7, 2.7, 3.7, 4.7, 5.7],  
               [1.8, 2.8, 3.8, 4.8, 5.8],  
               [1.9, 2.9, 3.9, 4.9, 5.9],  
               [2.0, 3.0, 4.0, 5.0, 6.0]]
```

- ✓ Setelah menemukan hasil transpose matriks b, selanjutnya adalah mengalikan matriks a dengan matriks transpose b sesuai kaidah perkalian matriks
- ✓ Maka hasil perkalian matriks dan ditambahkan dengan bias yang di outputkan adalah

```
[ [ 11.55 17.15 25.85 38.85 51.95]  
  [ 27.85 42.85 61.35 72.35 87.45]  
  [ 42.55 68.15 96.85 117.85 142.95]  
  [ 58.85 93.85 132.35 163.35 198.45]  
  [ 75.55 115.15 167.85 208.85 253.95]  
  [ 89.85 144.85 203.35 254.35 309.45]]
```

2. UTS 2

Buat dokumentasi dan jelaskan

Input layer Feature 10

Per batch nya 6 input

Hidden layer 1,5 neuron

Hidden layer 2,3 neuron

- Kodingan :

```
C:\> xampp > htdocs > AI > 011_uts_2_Multi Neuron Batch input hidden layer.py > ...
1 #Nama : Muvidha Fatmawati Putri
2 #NIM : 21091397011
3 #Kelas : A2021 MI
4 #lc. multi neuron batch input (menggunakan numpy)
5
6 #set numpy
7 import numpy as np
8
9 #set variabel inputs dengan matriks 6x10 (input 10 dan batch 6)
10 inputs = [[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0],
11 [1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0],
12 [2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0],
13 [3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 4.0],
14 [4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 5.0],
15 [5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 6.0]]
16
17 #hidden layer 1
18 #set weights hidden layer 1
19 weights = [[1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0],
20 [2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0],
21 [3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 4.0],
22 [4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 5.0],
23 [5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 6.0]]
24
25 #set biases hidden layer 1
26 biases = [2.2, 2.3, 5.5, 1.0, 0.6]
27
28 #ouputs hidden layer 1
29 layer_outputs = np.dot(inputs, np.array(weights).T) + biases
30
31 #hidden layer 2
32 #set weights hidden layer 2
33 weights2 = [[6.2, 6.4, 6.6, 6.8, 7],
34 [7.2, 7.4, 7.6, 7.8, 8],
35 [8.2, 8.4, 8.6, 8.8, 9]]
36
37 #set biases hidden layer 2
38 biases2 = [1, 2, 3]
39
40 #ouputs hidden layer 2
41 layer_outputs2 = np.dot(layer_outputs, np.array(weights2).T) + biases2
42
43 #print outputs
44 print(layer_outputs2)
45
```

- Output :

```
PS C:\xampp\htdocs\AI\AI 2> cd 'c:\xampp\htdocs\AI\AI 2'; & 'C:\Program Files\WindowsApps\PythonSoftwareFoundation.Python.3.10.3.10.2288.0_x64_qbz5n2kfra8p0\python3.10.exe' 'c:\Users\User\.vscode\extensions\ms-python.python-2022.16.1\pythonfiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '5362'
4' '--' 'c:\xampp\htdocs\AI\AI 2\011_uts_2_Multi Neuron Batch input hidden layer.py'
[[ 759.21  873.56  987.91]
 [1950.71 2242.56 2534.41]
 [3142.21 3611.56 4080.91]
 [4333.71 4980.56 5627.41]
 [5525.21 6349.56 7173.91]
 [6716.71 7718.56 8720.41]]
PS C:\xampp\htdocs\AI\AI 2>
```

- Penjelasan :

1. Pada baris 6 meng-set numpy
2. Baris 9 menginisialisasikan variabel, memasukkan nilai input dengan jumlah 10 baris dan batch dengan jumlah 6 baris
3. Pada baris 18 memasukan nilai weight

4. Pada baris 25 memasukkan nilai biases
5. Pada baris 32 memasukkan nilai weight2
6. Pada baris 37 memasukkan nilai biases2
7. Pada baris 29 melakukan perhitungan di layer1 dengan weight di transpose terlebih dahulu
8. Pada baris 40 melakukan perhitungan dari hasil layer1 dikali weight transpose ditambah biases2
9. untuk menampilkan pemanggilan hasil output dari kodingan tersebut

- Step perhitungan

Dimisalkan

Inputs = matriks a

Weights1 = matriks b

Maka,

```
a = [[0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0],
      [1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0],
      [2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0],
      [3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 4.0],
      [4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 5.0],
      [5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 6.0]]

b = [[1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0],
      [2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 2.7, 2.8, 2.9, 3.0],
      [3.1, 3.2, 3.3, 3.4, 3.5, 3.6, 3.7, 3.8, 3.9, 4.0],
      [4.1, 4.2, 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 5.0],
      [5.1, 5.2, 5.3, 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 6.0]]
```

Untuk perhitungan layer 1 multi neuron batch input hidden layer dapat menggunakan perkalian matriks, namun baris dan kolom dari matriks a dan b tidak sama, sehingga matriks b harus di transposkan (T) terlebih dahulu

- ✓ Hasil transpose matriks b

```
b transpose = [[1.1, 2.1, 3.1, 4.1, 5.1],
                [1.2, 2.2, 3.2, 4.2, 5.2],
                [1.3, 2.3, 3.3, 4.3, 5.3],
                [1.4, 2.4, 3.4, 4.4, 5.4],
                [1.5, 2.5, 3.5, 4.5, 5.5],
                [1.6, 2.6, 3.6, 4.6, 5.6],
                [1.7, 2.7, 3.7, 4.7, 5.7],
                [1.8, 2.8, 3.8, 4.8, 5.8],
                [1.9, 2.9, 3.9, 4.9, 5.9],
                [2.0, 3.0, 4.0, 5.0, 6.0]]
```

- ✓ Setelah menemukan hasil transpose matriks b, selanjutnya adalah mengalikan matriks a dengan matriks transpose b sesuai kaidah perkalian matriks
- ✓ Maka hasil perkalian ditambahkan dengan bias yang di outputkan adalah

```
[[ 11.98  17.13  23.83  30.83  38.98]
 [ 27.83  42.83  61.33  72.33  87.43]
 [ 42.33  68.13  98.83  117.83  142.93]
 [ 58.83  93.83  132.33  163.33  198.43]
 [ 73.33  113.13  167.83  208.83  253.93]
 [ 89.83  144.83  203.33  254.33  309.43]]
```

- ✓ Selanjutnya, setelah menemukan hasil layer 1, hasil tersebut dikalikan dengan transpose weight2

$$\begin{pmatrix}
 \begin{bmatrix}
 11.55 & 17.15 & 25.85 & 36.85 & 51.55 \\
 27.65 & 42.65 & 61.35 & 72.35 & 87.45 \\
 42.55 & 66.15 & 96.85 & 127.85 & 142.55 \\
 58.65 & 93.65 & 132.35 & 163.35 & 198.45 \\
 73.55 & 119.15 & 167.85 & 208.85 & 233.55 \\
 89.65 & 144.65 & 203.35 & 234.35 & 269.45
 \end{bmatrix}
 &
 \begin{bmatrix}
 71.45 & 77.45 & 88.45 \\
 21.45 & 27.45 & 38.45 \\
 36.45 & 42.45 & 53.45 \\
 51.45 & 57.45 & 68.45 \\
 66.45 & 72.45 & 83.45 \\
 81.45 & 87.45 & 98.45
 \end{bmatrix}
 \end{pmatrix}
 *
 \begin{bmatrix}
 759.21 & 873.54 & 987.91 \\
 1959.71 & 2242.54 & 2534.41 \\
 3142.21 & 3631.54 & 4089.91 \\
 4333.71 & 4999.54 & 5627.41 \\
 5525.21 & 6349.54 & 7179.91 \\
 6716.71 & 7718.54 & 8729.41
 \end{bmatrix}
 + (1, 2, 3)$$