

**LAPORAN KECERDASAN BUATAN**  
**UJIAN TENGAH SEMESTER**



**Disusun Oleh :**

**Shabinna Rahmadilla Santoso**

**21091397004**

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

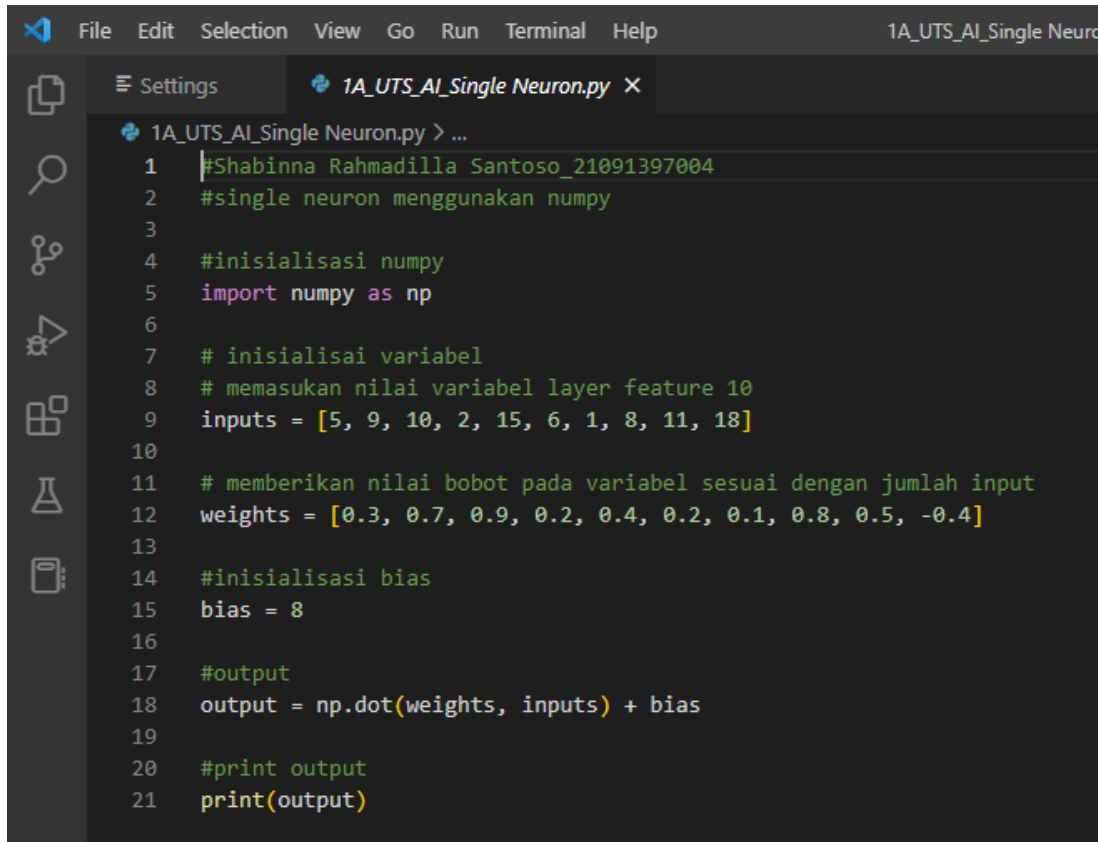
**2022**

## UTS 1

### A. Single Neuron

Input layer feature 10, Neuron 1

 Source Code



```
1A_UTS_AI_Single Neuron.py > ...
1  #Shabinnna Rahmadilla Santoso_21091397004
2  #single neuron menggunakan numpy
3
4  #inisialisasi numpy
5  import numpy as np
6
7  # inisialisai variabel
8  # memasukan nilai variabel layer feature 10
9  inputs = [5, 9, 10, 2, 15, 6, 1, 8, 11, 18]
10
11 # memberikan nilai bobot pada variabel sesuai dengan jumlah input
12 weights = [0.3, 0.7, 0.9, 0.2, 0.4, 0.2, 0.1, 0.8, 0.5, -0.4]
13
14 #inisialisasi bias
15 bias = 8
16
17 #output
18 output = np.dot(weights, inputs) + bias
19
20 #print output
21 print(output)
```

- Pada line 5 menginisialisasikan numpy ke np guna memudahkan dalam pengoperasionalkan source code ini, proses komputasi angka
- Di line 9 menginput variabel nilai sesuai dengan soal yaitu feature layer 10, jadi memasukkan angka sebanyak 10 baris angka
- Line 12 kita memasukkan weight atau bobot sesuai dengan input dan neuron yang telah ditentukan yaitu 1 neuron
- Bias yang terdapat pada line 15 berjumlah 1 karena sesuai dengan neuron yang ditetapkan yaitu 1
- Line 18 adalah output dari operasi perkalian antara weight dan input ditambah dengan bias, np.dot digunakan mengkalikan 2 matriks yaitu variabel weight dan input
- Di line 21 print output digunakan untuk menampilkan hasil operasi hitung dari source code tersebut

## Output

```
on.py*  
37.2  
PS C:\Users\Asus\Documents\vscode>  
⊗ 0 ▲ 0  
Type here to search
```

## Perhitungan output

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

Dalam perhitungan np.dot weight dikali dengan input dan hasil np.dot ditambah dengan biases

weights 10*1		input 1*10
$\begin{bmatrix} 0,3 \\ 0,7 \\ 0,9 \\ 0,2 \\ 0,4 \\ 0,2 \\ 0,1 \\ 0,8 \\ 0,5 \\ -0,4 \end{bmatrix}$		$\begin{bmatrix} 5, & 9, & 10, & 2, & 15, & 6, & 1, & 8, & 11, & 18 \end{bmatrix}$
	→	dikali (weights*input)
		$\begin{bmatrix} (0,3*5)(0,7*9)(0,9*10)(0,2*2)(0,4*15) \\ (0,2*6)(0,1*1)(0,8*8)(0,5*11)(-0,4*18) \end{bmatrix}$
		hasil np.dot
		$1,5+6,3+9+0,4+6+1,2+0,1+6,4+5,5-7,2 = 29,2$

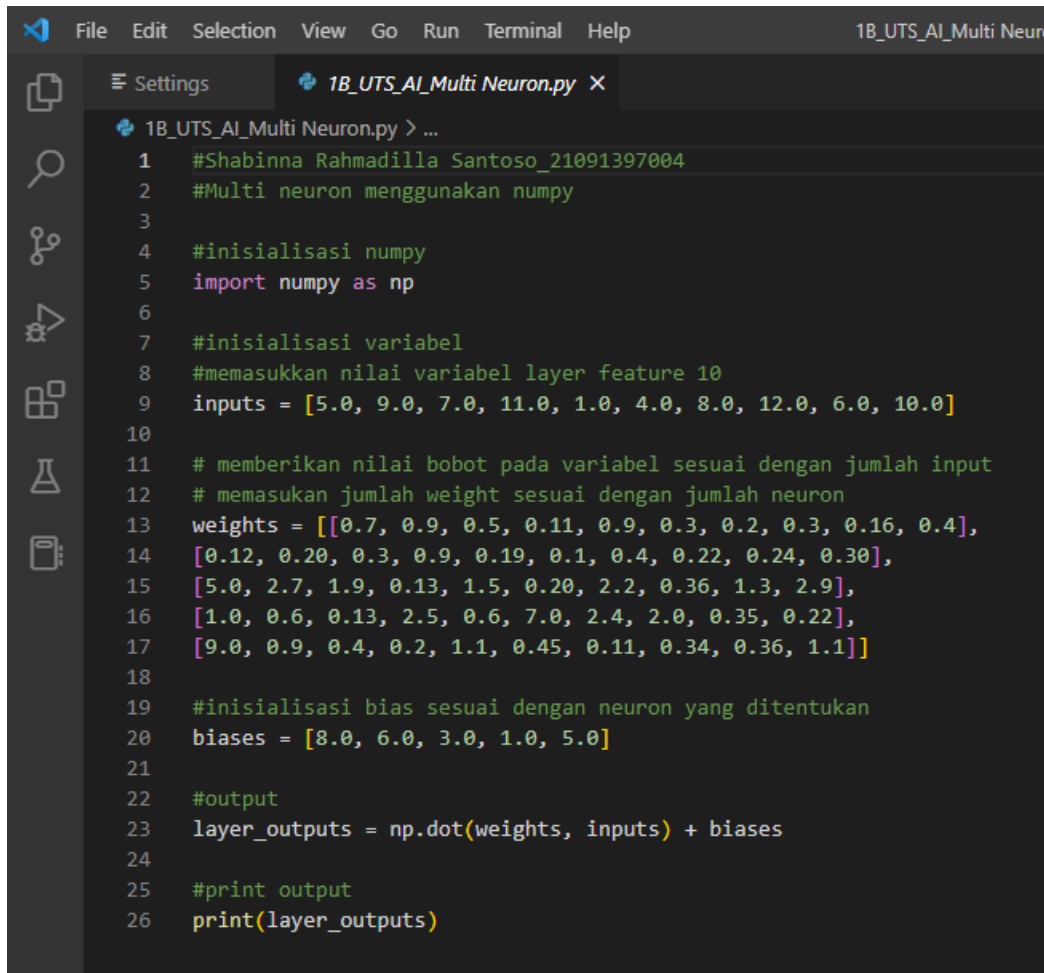
output = np.dot + biases

$$29,2 + 8 = 37,2$$

## B. Multi Neuron

Input Layer Feature 10, Neuron 5

 Source Code



```
1B_UTS_AI_Multi Neuron.py X
1B_UTS_AI_Multi Neuron.py > ...
1 #Shabinna Rahmadilla Santoso_21091397004
2 #Multi neuron menggunakan numpy
3
4 #inisialisasi numpy
5 import numpy as np
6
7 #inisialisasi variabel
8 #memasukkan nilai variabel layer feature 10
9 inputs = [5.0, 9.0, 7.0, 11.0, 1.0, 4.0, 8.0, 12.0, 6.0, 10.0]
10
11 # memberikan nilai bobot pada variabel sesuai dengan jumlah input
12 # memasukan jumlah weight sesuai dengan jumlah neuron
13 weights = [[0.7, 0.9, 0.5, 0.11, 0.9, 0.3, 0.2, 0.3, 0.16, 0.4],
14 [0.12, 0.20, 0.3, 0.9, 0.19, 0.1, 0.4, 0.22, 0.24, 0.30],
15 [5.0, 2.7, 1.9, 0.13, 1.5, 0.20, 2.2, 0.36, 1.3, 2.9],
16 [1.0, 0.6, 0.13, 2.5, 0.6, 7.0, 2.4, 2.0, 0.35, 0.22],
17 [9.0, 0.9, 0.4, 0.2, 1.1, 0.45, 0.11, 0.34, 0.36, 1.1]]
18
19 #inisialisasi bias sesuai dengan neuron yang ditentukan
20 biases = [8.0, 6.0, 3.0, 1.0, 5.0]
21
22 #output
23 layer_outputs = np.dot(weights, inputs) + biases
24
25 #print output
26 print(layer_outputs)
```

- Di line 5 menginisialisasikan numpy ke np guna memudahkan dalam pengoperasionalkan source code ini, proses komputasi angka
- Line 9 memasukkan nilai input sejumlah 10 baris angka sesuai dengan perintah yaitu feature layer 10
- Line 13 memasukkan nilai weight sejumlah 10 baris angka dan 5 kolom angka sesuai jumlah neuron karena neuron yang diminta adalah 5
- Line 20 memasukkan nilai bias sejumlah 5 baris angka karena neuron yang ditetapkan sebanyak 5
- Line 23 output dari operasi perkalian weight dan input ditambah bias, np.dot digunakan mengalikan 2 matriks yaitu variabel weight dan input
- Line 26 menampilkan hasil perhitungan output dari source code tersebut

## Output

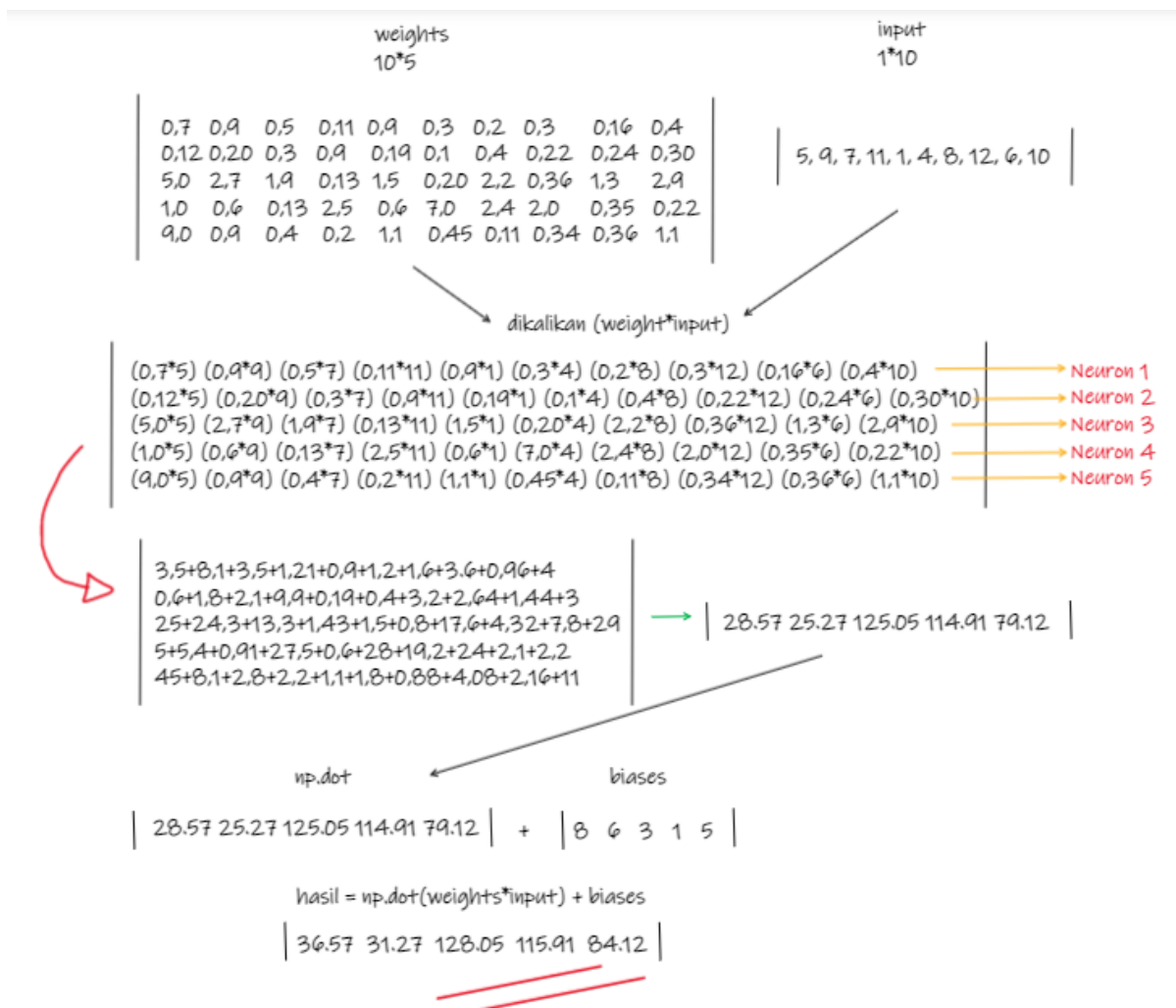
```

[ 36.57  31.27 128.05 115.91  84.12]
PS C:\Users\Asus\Documents\vscode>

```

## Perhitungan output

Dalam perhitungan np.dot weight dikali dengan input (sesuai dengan gambar) lalu hasil np.dot ditambah dengan biases



### C. Multi Neuron Batch Input

Input layer feature 10, per batchnya 6 input, Neuron 5

 Source Code

```
1C_UTS_AI_Multi Neuron Batch Input.py - vs
1 #Shabinna Rahmadilla Santoso_21091397004
2 #Multi Neuron batch input
3
4 #inisialisasi numpy
5 import numpy as np
6
7 #inisialisasi variabel
8 # memasukan nilai variabel layer feature 10 dengan batch sejumlah 6
9 inputs = [[2.5, 1.9, 7.2, 0.6, 1.3, 2.9, 1.4, 0.9, 2.4, 3.9],
10          [4.7, 1.2, 1.14, 5.5, 2.5, 3.2, 0.5, 1.9, 1.7, 1.0],
11          [3.7, 0.17, 4.3, 2.12, 0.11, 0.8, 2.1, 3.8, 1.5, 2.8],
12          [2.4, 0.20, 5.7, 3.3, 1.4, 1.3, 0.2, 0.10, 2.8, 1.4],
13          [0.11, 4.5, 3.0, 0.14, 2.9, 2.9, 1.8, 3.5, 5.3, 1.2],
14          [1.0, 1.3, 2.4, 0.15, 3.0, 3.6, 4.0, 4.2, 4.8, 5.7]]
15
16 # memberikan nilai bobot pada variabel sesuai dengan jumlah input
17 # memasukan jumlah weight sesuai dengan jumlah neuron dengan batch sejumlah 5
18 weights = [[-2.1, 3.0, 2.2, 3.8, -1.0, 0.8, -0.3, 2.4, -0.6, 1.3],
19           [4.8, 4.5, -0.7, 1.1, 4.9, 3.5, 0.4, -2.2, 1.0, 4.5],
20           [-0.1, -0.4, 1.3, 0.9, 0.2, 1.5, -1.7, 2.2, 2.8, 2.1],
21           [1.4, -0.9, 0.2, -0.10, 5.5, 3.3, -1.3, 4.9, 2.5, -3.7],
22           [3.0, 4.1, -1.3, 1.5, 1.6, -2.7, 2.9, 2.2, -0.3, -1.2]]
23
24 # inisialisasi bias sesuai dengan neuron yang ditentukan
25 biases = [0.3, 1.9, 2.3, 3.5, 1.7]
26
27 #output
28 layer_outputs = np.dot(inputs, np.array(weights).T) + biases
29
30 #print output
31 print(layer_outputs)
```

- Line 5 menginisialisasikan numpy ke np guna memudahkan dalam pengoperasionalkan source code ini, proses komputasi angka
- Line 9 menginisialisasikan variabel, memasukkan nilai input dengan jumlah 10 baris angka sesuai dengan yang diminta yaitu feature layer 10 dan 6 kolom angka dengan perintah batch = 6
- Line 18 memasukan nilai weight dengan 5 kolom karena sesuai dengan neuron yang diminta yaitu 5 dan 10 baris angka
- Di line 25 memasukkan nilai bias sejumlah dengan banyaknya neuron yang ditetapkan yaitu 5
- Line 28 output operasi dari perkalian input dan weight yang kemudian di transpose dan ditambah oleh bias, np.dot digunakan mengkalikan 2 matriks yaitu variabel weight dan input
- Line 31 untuk menampilkan hasil output dari source code tersebut

## Output

```

[25.26  53.12  30.31  17.55   3.42 ]
[22.188 60.782 23.272 42.198 26.768]
[22.316 29.666 25.452 21.7   19.743]
[20.9   34.33  25.25  21.53   4.27 ]
[26.361 48.812 31.445 49.33  23.49 ]
[21.24  61.145 38.785 38.865 14.775]]
PS C:\Users\Asus\Documents\vscode>

```

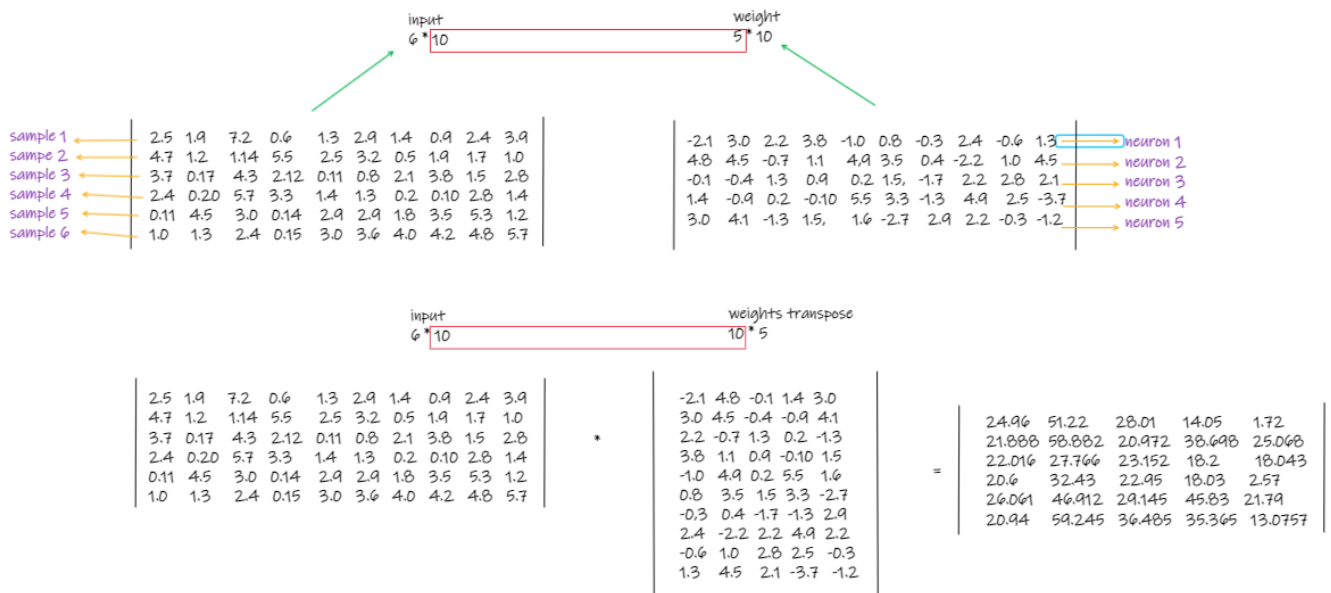
## Perhitungan output

```

#output
layer_outputs = np.dot(inputs, np.array(weights).T) + biases

```

`np.array (weight) T` adalah mentranspose weight, dilakukan transpose pada weight karena untuk menyamakan ordo dengan input. Jika ordo tidak sama maka tidak akan dapat dihitung atau dioperasikan. Setelah weight di transpose maka operasi hitung dapat dilakukan dengan cara mengkali setiap baris input dengan kolom weight.



Cara menghitung `np.dot` adalah setiap baris input dikalikan dengan setiap baris kolom. Baris 1 input dikali kolom 1 weight, baris 1 input dikali kolom 2 weight dan seterusnya hingga selesai menghasilkan `np.dot 6*5`

$$\begin{bmatrix} 2.5 & 1.9 & 7.2 & 0.6 & 1.3 & 2.9 & 1.4 & 0.9 & 2.4 & 3.9 \\ 4.7 & 1.2 & 1.14 & 5.5 & 2.5 & 3.2 & 0.5 & 1.9 & 1.7 & 1.0 \\ 3.7 & 0.17 & 4.3 & 2.12 & 0.11 & 0.8 & 2.1 & 3.8 & 1.5 & 2.8 \\ 2.4 & 0.20 & 5.7 & 3.3 & 1.4 & 1.3 & 0.2 & 0.10 & 2.8 & 1.4 \\ 0.11 & 4.5 & 3.0 & 0.14 & 2.9 & 2.9 & 1.8 & 3.5 & 5.3 & 1.2 \\ 1.0 & 1.3 & 2.4 & 0.15 & 3.0 & 3.6 & 4.0 & 4.2 & 4.8 & 5.7 \end{bmatrix} \cdot \begin{bmatrix} -2.1 & 4.8 & -0.1 & 1.4 & 3.0 \\ 3.0 & 4.5 & -0.4 & -0.9 & 4.1 \\ 2.2 & -0.7 & 1.3 & 0.2 & -1.3 \\ 3.8 & 1.1 & 0.9 & -0.10 & 1.5 \\ -1.0 & 4.9 & 0.2 & 5.5 & 1.6 \\ 0.8 & 3.5 & 1.5 & 3.3 & -2.7 \\ -0.3 & 0.4 & -1.7 & -1.3 & 2.9 \\ 2.4 & -2.2 & 2.2 & 4.9 & 2.2 \\ -0.6 & 1.0 & 2.8 & 2.5 & -0.3 \\ 1.3 & 4.5 & 2.1 & -3.7 & -1.2 \end{bmatrix} = \begin{bmatrix} 24.96 & 51.22 & 28.01 & 14.05 & 1.72 \\ 21.888 & 58.882 & 20.972 & 38.698 & 25.068 \\ 22.016 & 27.766 & 23.152 & 18.2 & 18.043 \\ 20.6 & 32.43 & 22.95 & 18.03 & 2.57 \\ 26.061 & 46.912 & 29.145 & 45.83 & 21.79 \\ 20.94 & 59.245 & 36.485 & 35.365 & 13.0757 \end{bmatrix}$$

Saat hasil np.dot dari perkalian input dan weight transpose keluar, np.dot ditambah dengan biases

$$\begin{array}{c} \text{np.dot} \\ \begin{bmatrix} 24.96 & 51.22 & 28.01 & 14.05 & 1.72 \\ 21.888 & 58.882 & 20.972 & 38.698 & 25.068 \\ 22.016 & 27.766 & 23.152 & 18.2 & 18.043 \\ 20.6 & 32.43 & 22.95 & 18.03 & 2.57 \\ 26.061 & 46.912 & 29.145 & 45.83 & 21.79 \\ 20.94 & 59.245 & 36.485 & 35.365 & 13.0757 \end{bmatrix} \end{array} + \begin{array}{c} \text{biases} \\ \begin{bmatrix} 0.3 & 1.9 & 2.3 & 3.5 & 1.7 \end{bmatrix} \end{array} = \begin{bmatrix} 25.26 & 53.12 & 30.31 & 17.55 & 3.42 \\ 22.188 & 60.782 & 23.272 & 42.198 & 26.768 \\ 22.316 & 29.666 & 25.452 & 21.7 & 19.743 \\ 20.9 & 34.33 & 25.25 & 21.53 & 4.27 \\ 26.361 & 48.812 & 31.445 & 48.33 & 23.49 \\ 21.24 & 61.145 & 38.785 & 38.865 & 14.775 \end{bmatrix}$$



## UTS 2

### D. Multi Neuron Batch Input

Input layer feature 10, batchnya 6 input, hidden layer1 (5 neuron), hidden layer2 (3 neuron)

 Source Code

```
1D_UTS_AI_MultiNeuronBatchInput2.py - vscode - Visual Studio Code
1D_UTS_AI_MultiNeuronBatchInput2.py > {} np
1 #Shabinna Rahmadilla Santoso_21091397004
2 #Multiple perceptron / Neuron batch and multiple layer 2
3
4 #inisialisasi numpy
5 import numpy as np
6
7 # inisialisasi variabel
8 # memasukan nilai variabel layer feature 10 dengan batch sejumlah 6
9 inputs = [[2.5, 1.9, 7.2, 0.6, 1.3, 2.9, 1.4, 0.9, 2.4, 3.9],
10           [4.7, 1.2, 1.14, 5.5, 2.5, 3.2, 0.5, 1.9, 1.7, 1.0],
11           [3.7, 0.17, 4.3, 2.12, 0.11, 0.8, 2.1, 3.8, 1.5, 2.8],
12           [2.4, 0.20, 5.7, 3.3, 1.4, 1.3, 0.2, 0.10, 2.8, 1.4],
13           [0.11, 4.5, 3.0, 0.14, 2.9, 2.9, 1.8, 3.5, 5.3, 1.2],
14           [1.0, 1.3, 2.4, 0.15, 3.0, 3.6, 4.0, 4.2, 4.8, 5.7]]
15
16 # memberikan nilai bobot pada variabel sesuai dengan jumlah input
17 # memasukan jumlah weight sesuai dengan jumlah neuron yaitu sejumlah 5
18 weights1 = [[-2.1, 3.0, 2.2, 3.8, -1.0, 0.8, -0.3, 2.4, -0.6, 1.3],
19             [4.8, 4.5, -0.7, 1.1, 4.9, 3.5, 0.4, -2.2, 1.0, 4.5],
20             [-0.1, -0.4, 1.3, 0.9, 0.2, 1.5, -1.7, 2.2, 2.8, 2.1],
21             [1.4, -0.9, 0.2, -0.10, 5.5, 3.3, -1.3, 4.9, 2.5, -3.7],
22             [3.0, 4.1, -1.3, 1.5, 1.6, -2.7, 2.9, 2.2, -0.3, -1.2]]
23
24 # inisialisasi biases pada layer1 sesuai dengan neuron yang ditentukan yaitu layer 1 = 5 neuron
25 biases1 = [0.3, 1.9, 2.3, 3.5, 1.7]
26
27 # inisialisasi jumlah weight 2, weight layer 2 = neuron layer 1 yaitu 5
28 # memasukan jumlah weight sesuai dengan neuron layer 2 yaitu 3 neuron
29 weights2 = [[-0.2, 1.5, 1.1, 2.1, -1.3],
30             [0.4, 1.6, 2.4, 3.0, 2.2],
31             [2.4, 2.5, 3.0, 1.1, 1.5]]
32
33 # inisialisasi biases pada layer2 dengan neuron yang ditentukan yaitu 3
34 biases2 = [2.0, 1.0, 0.4]
35
36 # output
37 # menghitung layer1 dengan (inputs*weight1) dan biases1
38 layer1_outputs = np.dot(inputs, np.array(weights1).T) + biases1
39
40 # menghitung layer2 dengan hasil perhitungan pada layer1
41 layer2_outputs = np.dot(layer1_outputs, np.array(weights2).T) + biases2
42
43 #print output layer2
44 print(layer2_outputs)
```

- Line 4 menginisialisasikan numpy ke np guna memudahkan dalam pengoperasian source code ini, proses komputasi angka
- Line 9 menginisialisasikan variabel, memasukkan nilai input dengan jumlah 10 baris angka sesuai dengan yang diminta yaitu feature layer 10 dan 6 kolom angka dengan perintah batch = 6

- Line 18 memasukkan nilai weight dengan 5 kolom karena sesuai dengan neuron yang diminta yaitu 5 dan 10 baris angka
- Line 25 memasukkan nilai bias sejumlah dengan banyaknya neuron yang ditetapkan yaitu 5
- Line 29 memasukkan nilai weight2 dengan jumlah weight samadengan neuron atau ordo di layer1 yaitu 5 dengan 3 kolom sesuai dengan neuron layer2 yaitu 3
- Line 34 memasukkan nilai biases sejumlah dengan banyaknya neuron yang ditetapkan di layer2 yaitu 3
- Line 38 melakukan perhitungan di layer1 dengan weight di transpose terlebih dahulu
- Line 41 melakukan perhitungan dari hasil layer1 dikali weight transpose ditambah biases2
- Line 44 untuk menampilkan hasil output dari source code tersebut

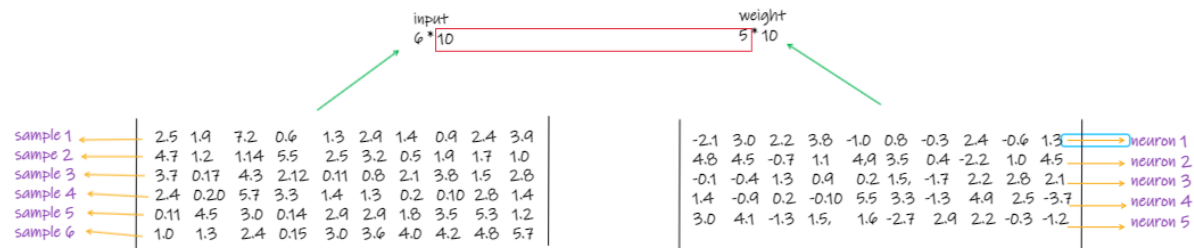
#### Output

```
[[142.378  229.014  309.189 ]
 [168.152  348.4628 361.992 ]
 [ 89.9371 227.0114 257.9639]
 [116.752  198.872  242.223 ]
 [177.5913 364.7796 369.5294]
 [194.542  349.512  385.5075]]
```

#### Perhitungan Output

```
# output
# menghitung layer1 dengan (inputs*weight1) dan biases1
layer1_outputs = np.dot(inputs, np.array(weights1).T) + biases1
```

`np.array (weight) T` adalah mentranspose weight, dilakukan transpose pada weight karena untuk menyamakan ordo dengan input. Jika ordo tidak sama maka tidak akan dapat dihitung atau dioperasikan. Setelah weight di transpose maka operasi hitung dapat dilakukan dengan cara mengkali setiap baris input dengan kolom weight.



$$\begin{array}{|c|} \hline \text{input} \\ \hline 6 \times 10 \\ \hline \end{array} \times \begin{array}{|c|} \hline \text{weights transpose} \\ \hline 10 \times 5 \\ \hline \end{array} = \begin{array}{|c|} \hline \begin{array}{cccccc} 24.96 & 51.22 & 28.01 & 14.05 & 1.72 \\ 21.888 & 58.882 & 20.972 & 38.698 & 25.068 \\ 22.016 & 27.766 & 23.152 & 18.2 & 18.043 \\ 20.6 & 32.43 & 22.95 & 18.03 & 2.57 \\ 26.061 & 46.912 & 29.145 & 45.83 & 21.79 \\ 20.94 & 59.245 & 36.485 & 35.365 & 13.0757 \end{array} \\ \hline \end{array}$$

Saat hasil layer 1 sudah ditemukan dengan mengkali input dan transpose weight 1 hasil np.dot ditambah dengan biases 1

$$\begin{array}{|c|} \hline \text{np.dot} \\ \hline \begin{array}{cccccc} 24.96 & 51.22 & 28.01 & 14.05 & 1.72 \\ 21.888 & 58.882 & 20.972 & 38.698 & 25.068 \\ 22.016 & 27.766 & 23.152 & 18.2 & 18.043 \\ 20.6 & 32.43 & 22.95 & 18.03 & 2.57 \\ 26.061 & 46.912 & 29.145 & 45.83 & 21.79 \\ 20.94 & 59.245 & 36.485 & 35.365 & 13.0757 \end{array} \\ \hline \end{array} + \begin{array}{|c|} \hline \text{biases} \\ \hline \begin{array}{ccccc} 0.3 & 1.9 & 2.3 & 3.5 & 1.7 \end{array} \\ \hline \end{array} = \begin{array}{|c|} \hline \begin{array}{ccccc} 25.26 & 53.12 & 30.31 & 17.55 & 3.42 \\ 22.188 & 60.782 & 23.272 & 42.198 & 26.768 \\ 22.316 & 29.666 & 25.452 & 21.7 & 19.743 \\ 20.9 & 34.33 & 25.25 & 21.53 & 4.27 \\ 26.361 & 48.812 & 31.445 & 48.33 & 23.49 \\ 21.24 & 61.145 & 38.785 & 38.865 & 14.775 \end{array} \\ \hline \end{array}$$

Menghitung hidden layer 2

```
# menghitung layer2 dengan hasil perhitungan pada layer1
layer2_outputs = np.dot(layer1_outputs, np.array(weights2).T) + biases2
```

Saat hasil dari layer1 didapatkan lalu lanjut masuk ke penghitungan layer 2, mencari nilai np.dot layer 2 dengan cara hasil layer 1 dikali dengan weight transpose 2. Weights 2 di transpose untuk menyamakan ordo output hidden layer 1

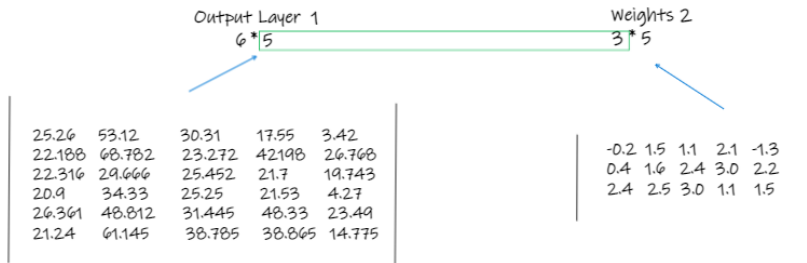


Diagram illustrating the matrix multiplication:

Output Layer 1 (6 × 5) is multiplied by Weights 2 transpose (5 × 3).

25.26	53.12	30.31	17.55	3.42
22.188	68.782	23.272	42.198	26.768
22.316	29.666	25.452	21.7	19.743
20.9	34.33	25.25	21.53	4.27
26.361	48.812	31.445	48.33	23.49
21.24	61.145	38.785	38.865	14.775

×

-0.2	0.4	2.4
1.5	1.6	2.5
1.1	2.4	3.0
2.1	3.0	1.1
-1.3	2.2	1.5

=

140.378	228.014	308.789
166.152	383.4628	361.592
87.9371	226.0114	257.5639
114.752	197.872	241.823
175.5913	363.7796	369.1294
192.542	348.512	385.1075

Setelah di dapatkan hasil np.dot layer 2 lalu ditambah dengan biases 2 dan hasil layer 2 didapatkan

Diagram illustrating the addition of biases:

np.dot layer 2 (6 × 3) is added to biases 2 (6 × 3).

140.378	228.014	308.789
166.152	383.4628	361.592
87.9371	226.0114	257.5639
114.752	197.872	241.823
175.5913	363.7796	369.1294
192.542	348.512	385.1075

+

2.0	1.0	0.4
-----	-----	-----

=

142.378	229.014	309.189
168.152	384.4628	361.992
89.9371	227.0114	257.9639
116.752	198.872	242.223
177.5913	364.7796	369.5294
194.542	349.512	385.5075