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 :

```
# Program Single Neuron,
# 1. Input layer feature 10
# 2. Neuron 1

# inisialisasi numpy
import numpy as np

# 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

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

# cetak output
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.

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

Perhitungan dot product.

$$\begin{array}{ccc} \text{weights} & & \text{inputs} \\ 10 * 1 & & 1 * 10 \\ \begin{bmatrix} 4.2 \\ 1.8 \\ 3.8 \\ -4.3 \\ -0.3 \\ 0.9 \\ 1.9 \\ 2.1 \\ 2.8 \end{bmatrix} & * & [2, 5.5, 9, 4, 1, 6.5, 3, 7, 2.5, 6] = 79.699 \end{array}$$

Kemudian akan ditambahkan dengan bias.

$$\overset{\text{np.dot}}{79{,}699} + \overset{\text{bias}}{7} = 86{,}699$$

- Mencetak output.

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

```
>>>
86.69999999999999
```

- **Multi Neuron**

Source Code :

```
# Program Multi Neuron
# 1. Input layer feature 10
# 2. Neuron 5

# inisialisasi numpy
import numpy as np

# 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]

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

# cetak output
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 (inputs*weights) + bias.

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

Perhitungan dot product.

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

$$\begin{array}{cl} \begin{bmatrix} (2*-3.3) & (5.5*4.2) & (9*1.8) & (4*3.8) & (1*-4.3) & (6.5*-0.3) & (3*0.9) & (7*1.9) & (2.5*2.1) & (6*2.8) \\ (2*4) & (5.5*3.6) & (9*4.4) & (4*-3.3) & (1*-1.8) & (6.5*-2.1) & (3*1.1) & (7*2.2) & (2.5*-3.9) & (6*3.4) \\ (2*1) & (5.5*-0.8) & (9*3) & (4*1.5) & (1*1.2) & (6.5*-1.9) & (3*-2.7) & (7*4) & (2.5*0.2) & (6*2.3) \\ (2*2.1) & (5.5*-2.6) & (9*3.9) & (4*4.6) & (1*0.3) & (6.5*-3.5) & (3*2.2) & (7*-4.8) & (2.5*4) & (6*2.3) \\ (2*4.1) & (5.5*-2.2) & (9*0.7) & (4*1.7) & (1*2) & (6.5*0.2) & (3*4.6) & (7*2.6) & (2.5*-2.3) & (6*3) \end{bmatrix} & \begin{array}{l} \rightarrow \text{Neuron 1} \\ \rightarrow \text{Neuron 2} \\ \rightarrow \text{Neuron 3} \\ \rightarrow \text{Neuron 4} \\ \rightarrow \text{Neuron 5} \end{array} \end{array}$$

$$\downarrow$$

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

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.

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

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

- **Multi Neuron Batch Input**

Source Code :

```python
# Program Multi Neuron Batch Input
# 1. Input layer feature 10
# 2. Per batch nya 6 input
# 3. Neuron 5

# inisialisasi numpy
import numpy as np

# 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]

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

# 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 ]]
```

Penjelasan step by step:

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

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

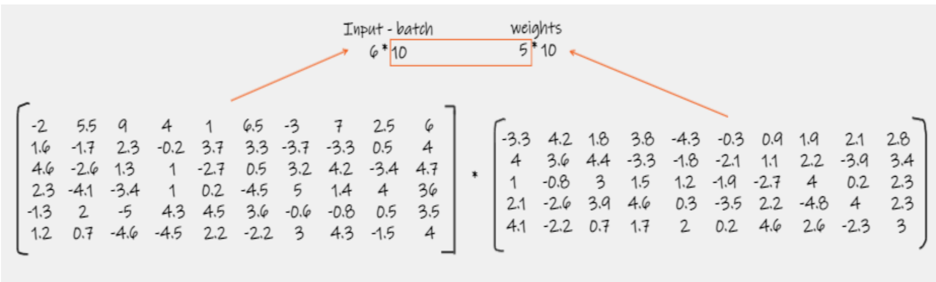
- Mengset nilai dari variabel inputs, weight, dan juga bias. Dengan ketentuan per batch 6 input dan tiap input – batch layer 10 jadi inputs = 6 * 10 dan 5 neuron.

```
# 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]
```

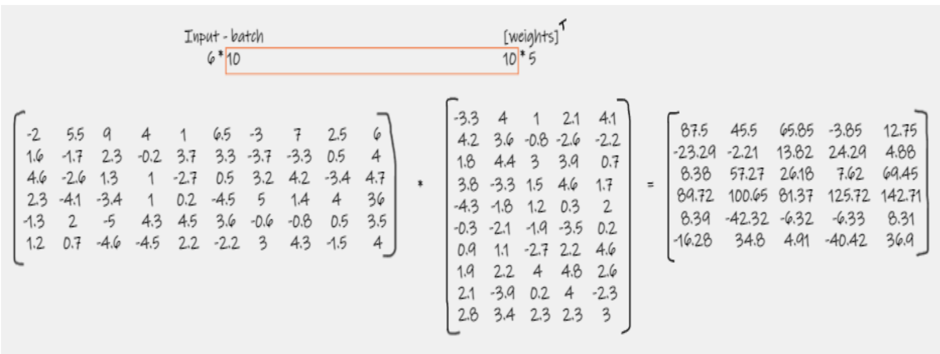
- Melakukan penghitungan untuk mendapatkan output. Dengan rumus :

```
# 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.



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



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 :

```
# Program Multi Neuron Batch Input
# 1. Input layer feature 10
# 2. Per batch nya 6 input
# 3. Hidden layer 1, 5 neuron
# 4. Hidden layer 2, 3 neuron

# inisialisasi numpy
import numpy as np

# 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]

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

# 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]

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

# cetak output
print(output)
print(output2)
```

```
>>>
[[ 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]]
```

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]]
```

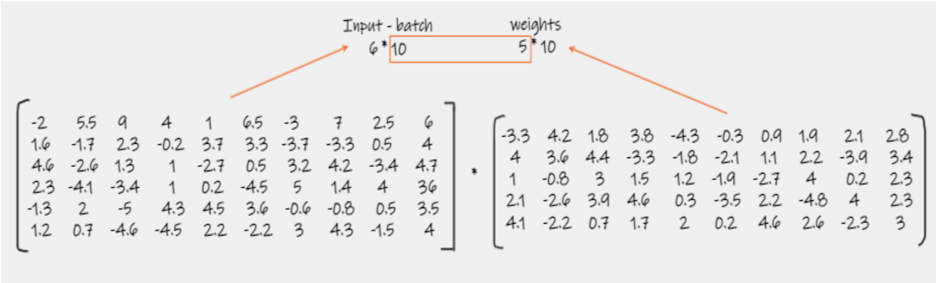
```
# hidden layer 1
# inisialisasi bobot hidden layer 1
weights =  [[-3.3, 4.2, 1.8, 3.8, -4.3, -0.3, 0.9, 1.9, 2.1, 2.8],
            [4, 3.6, 4.4, -3.3, -1.8, -2.1, 1.1, 2.2, -3.9, 3.4],
            [1, -0.8, 3, 1.5, 1.2, -1.9, -2.7, 4, 0.2, 2.3],
            [2.1, -2.6, 3.9, 4.6, 0.3, -3.5, 2.2,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

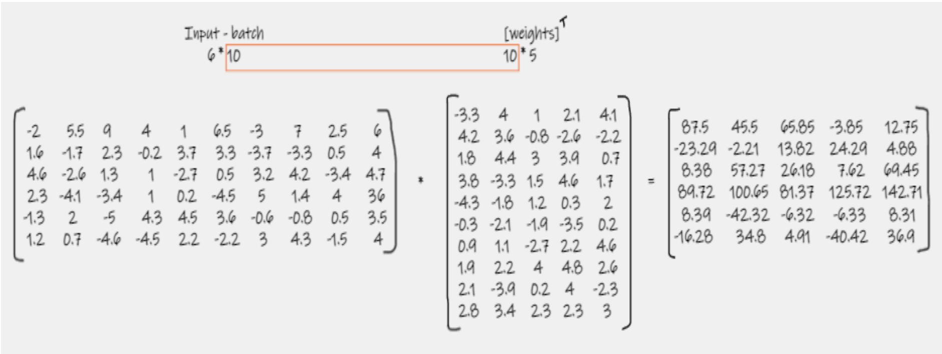
- Melakukan penghitungan untuk mendapatkan output dari hidder layer pertama, dengan rumus :

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

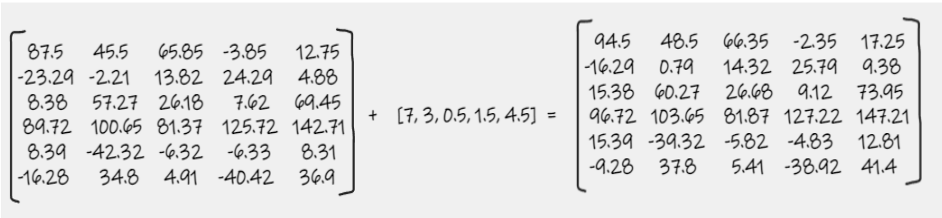
np.array (weight).T maksudnya adalah mentransposekan weight. Kenapa harus dilakukan Transpose pada weight, karena jika menggunakan weight normal maka ordonya tidak sesuai dengan ordo input sehingga tidak bisa dioperasikan.



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



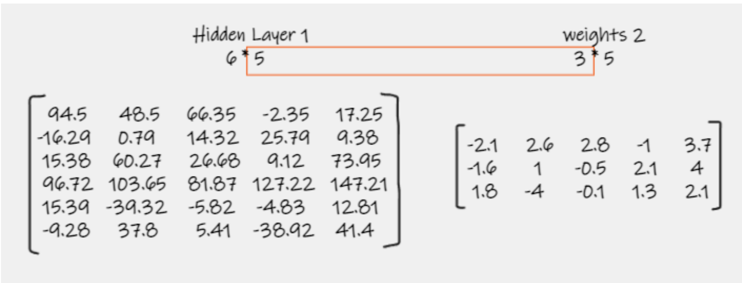
Kemudian akan ditambahkan dengan bias.



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.



Setelah ditranspose maka dilakukan perhitungan berikut

Hidden Layer 1 $6 * \boxed{5}$ weights 2 $\boxed{5} * 3$

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

Kemudian akan ditambahkan dengan bias 2

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

- 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]]
```