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

```python
# Alvin Febrianto
# 21091397031
# 2021 A / D4 Manajemen Informatika

# a. Single Neuron
#    i.  Input layer feature 10
#    ii. Neuron 1

import numpy as np

# Inisialisasi variabel inputs
inputs = [0.8, 4, 7, -3.2, 1.3, 3, -8, -2.0, 4.5, 2.4]

# Inisialisasi variabel weights
weights = [-3.6, 3.2, 6, -1.8, 2.8, -7, 5.7, 3.4, 3.1, 4.2]

# Inisialisasi variabel bias
bias = 6

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

# Menampilkan output
print(output)
```

```
17.95
[Finished in 0.8s]
```

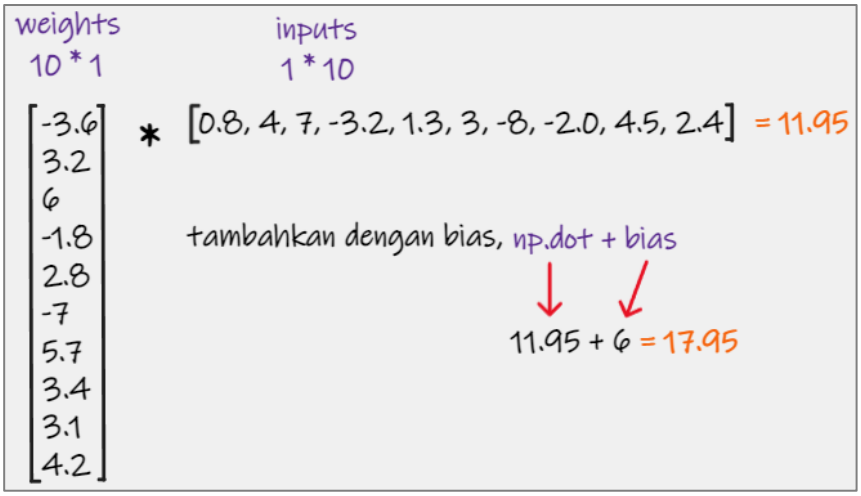
b) Penjelasan

- 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.

```python
import numpy as np

# Inisialisasi variabel inputs
inputs = [0.8, 4, 7, -3.2, 1.3, 3, -8, -2.0, 4.5, 2.4]

# Inisialisasi variabel weights
weights = [-3.6, 3.2, 6, -1.8, 2.8, -7, 5.7, 3.4, 3.1, 4.2]

# Inisialisasi variabel bias
bias = 6
```

- Melakukan operasi perhitungan dot product dengan rumus (inputs × weights) + bias.

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



- Menampilkan output.

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

```
17.95
[Finished in 0.8s]
```

B. Multi Neuron

i. Input layer feature 10

ii. Neuron 5

a) Source Code

1b_MultiNeuron_031_AlvinFebrianto.py

```
# Alvin Febrianto
# 21091397031
# 2021 A / D4 Manajemen Informatika

# b. Multi Neuron
#    i.  Input layer feature 10
#    ii. Neuron 5

import numpy as np

# Inisialisasi variabel inputs
inputs = [-8, 7.1, -5, 4.2, 5.3, 2.0, 7.9, 2.4, 8.4, -4]

# Inisialisasi variabel weights
weights = [[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]]

# Inisialisasi variabel biases
biases = [2.0, 4, 0.8, 5.2, 3]

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

# Menampilkan output
print(output)
```

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

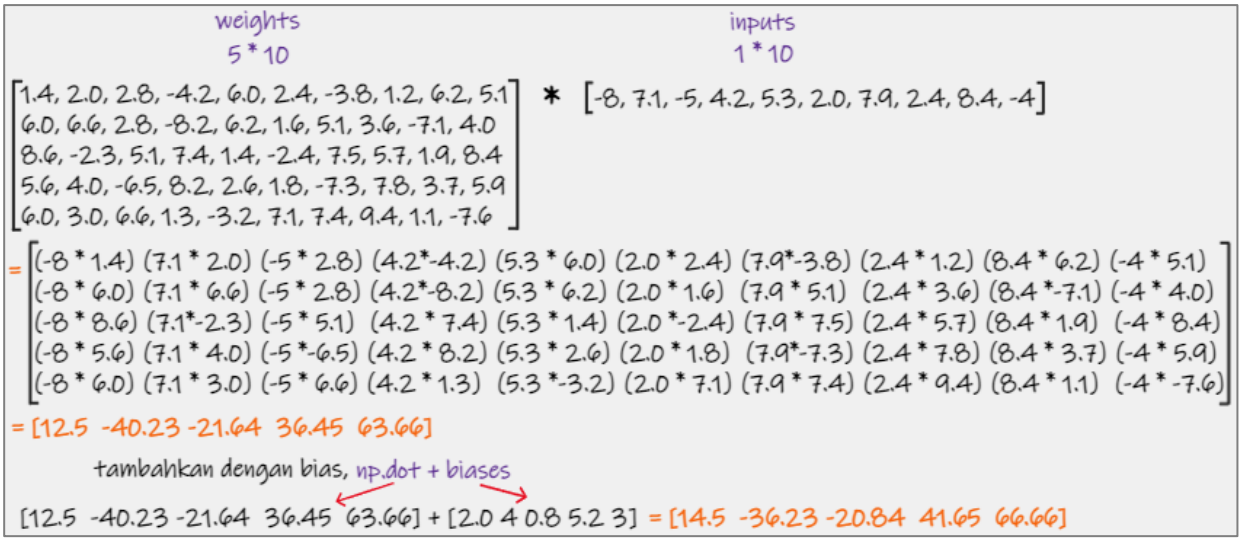
b) Penjelasan

- 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.

```python
import numpy as np

# Inisialisasi variabel inputs
inputs = [-8, 7.1, -5, 4.2, 5.3, 2.0, 7.9, 2.4, 8.4, -4]

# Inisialisasi variabel weights
weights = [[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]]

# Inisialisasi variabel biases
biases = [2.0, 4, 0.8, 5.2, 3]
```

- Melakukan perhitungan dot product dengan rumus (inputs × weights) + biases.

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



- Menampilkan output.

```python
# Menampilkan output
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

```python
# Alvin Febrianto
# 21091397031
# 2021 A / D4 Manajemen Informatika

# c. Multi Neuron Batch Input
#    i.   Input layer feature 10
#    ii.  Per batch nya 6 input
#    iii. Neuron 5

import numpy as np

# Inisialisasi variabel inputs
inputs = [[-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]]

# Inisialisasi variabel weights
weights = [[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]]

# Inisialisasi variabel biases
biases = [2.0, 4, 0.8, 5.2, 3]

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

# Menampilkan output
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

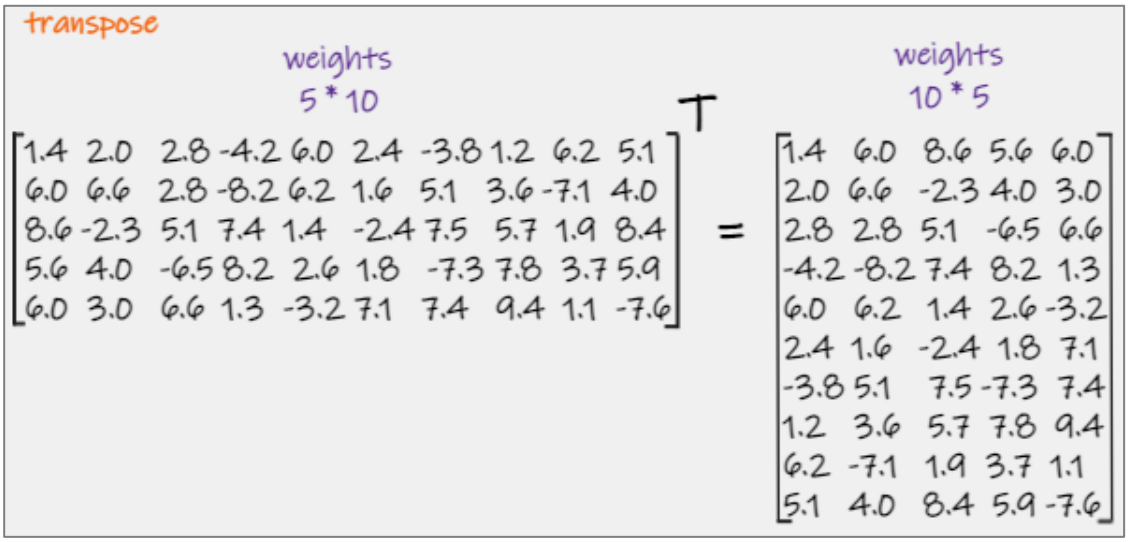
- Langkah pertama yaitu mengimpor NumPy (library Python) yang bisa digunakan untuk melakukan operasi vektor dan matriks. Lalu mengisi nilai dari setiap variabel inputs, weights, dan biases sesuai ketentuan soal yaitu per batch-nya 6 input dengan jumlah tiap input layer 10 sehingga inputs = 6 * 10 dan 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.

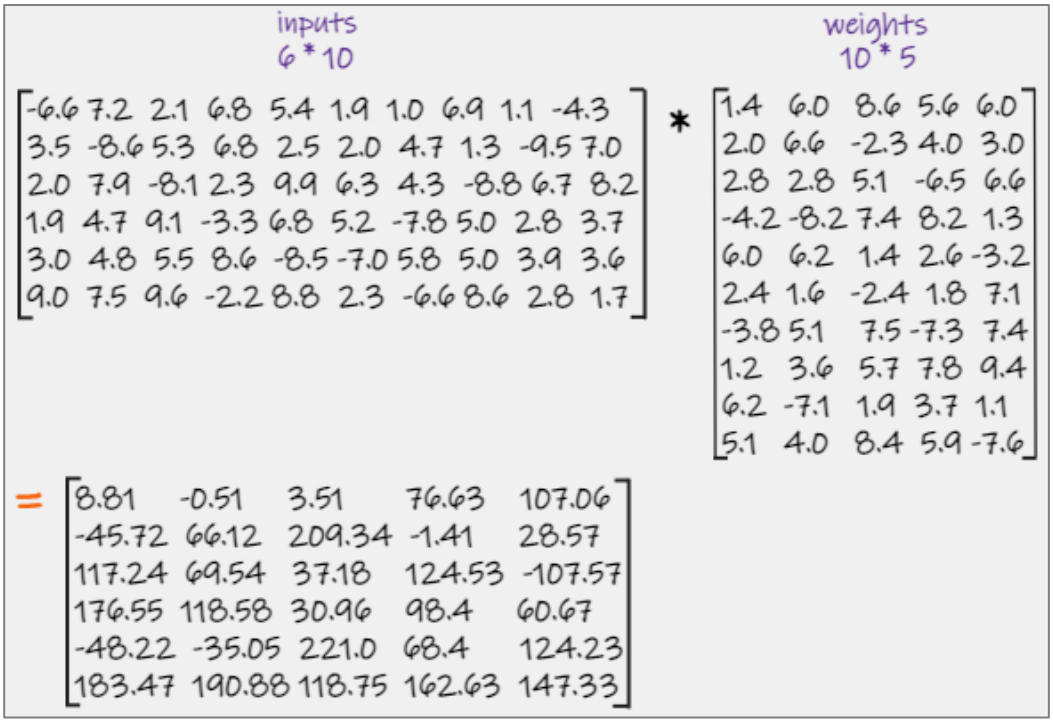
```
10  import numpy as np
11
12  # Inisialisasi variabel inputs
13  inputs = [[-6.6, 7.2, 2.1, 6.8, 5.4, 1.9, 1.0, 6.9, 1.1, -4.3],
14            [3.5, -8.6, 5.3, 6.8, 2.5, 2.0, 4.7, 1.3, -9.5, 7.0],
15            [2.0, 7.9, -8.1, 2.3, 9.9, 6.3, 4.3, -8.8, 6.7, 8.2],
16            [1.9, 4.7, 9.1, -3.3, 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]
```

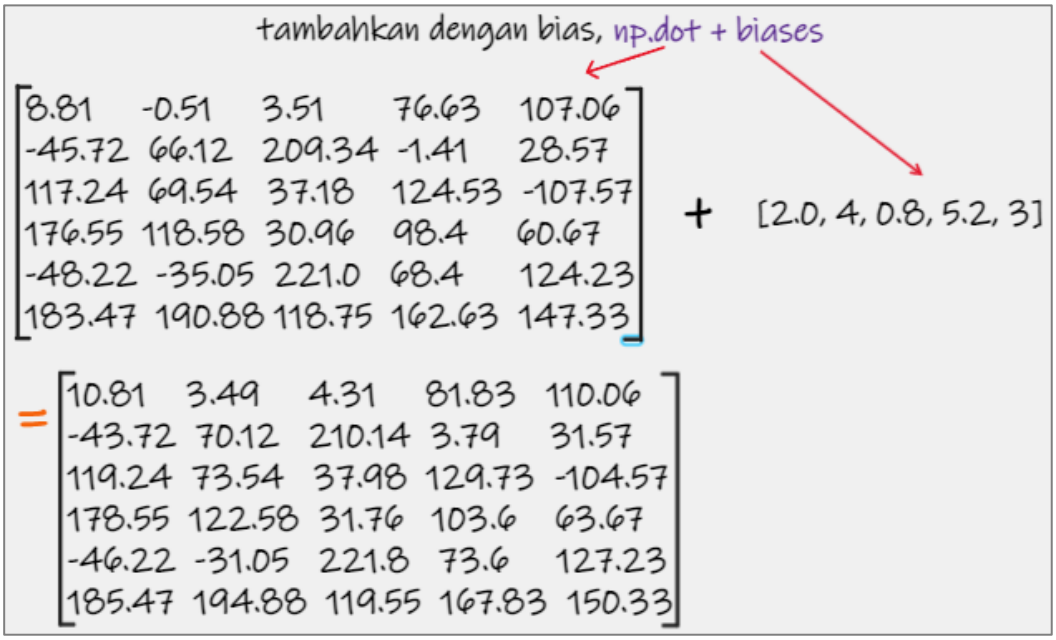
- Melakukan operasi penghitungan.

```
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.







- 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  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]
```

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

```python
# Alvin Febrianto | 21091397031 | 2021 A - D4 Manajemen Informatika
# 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

# Inisialisasi library NumPy
import numpy as np

# Inisialisasi variabel inputs
inputs = [[-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]]

# Inisialisasi variabel weights1 [hidden layer 1, 5 neuron]
weights1 = [[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]]

# Inisialisasi variabel biases1 [hidden layer 1, 5 neuron]
biases1 = [2.0, 4, 0.8, 5.2, 3]

# Inisialisasi variabel weights2 [hidden layer 2, 3 neuron]
weights2 = [[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]]

# Inisialisasi variabel biases2 [hidden layer 2, 3 neuron]
biases2 = [1.8, 6, 4.5]

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

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

# Menampilkan output
print(layer2_outputs)
```

b) Penjelasan

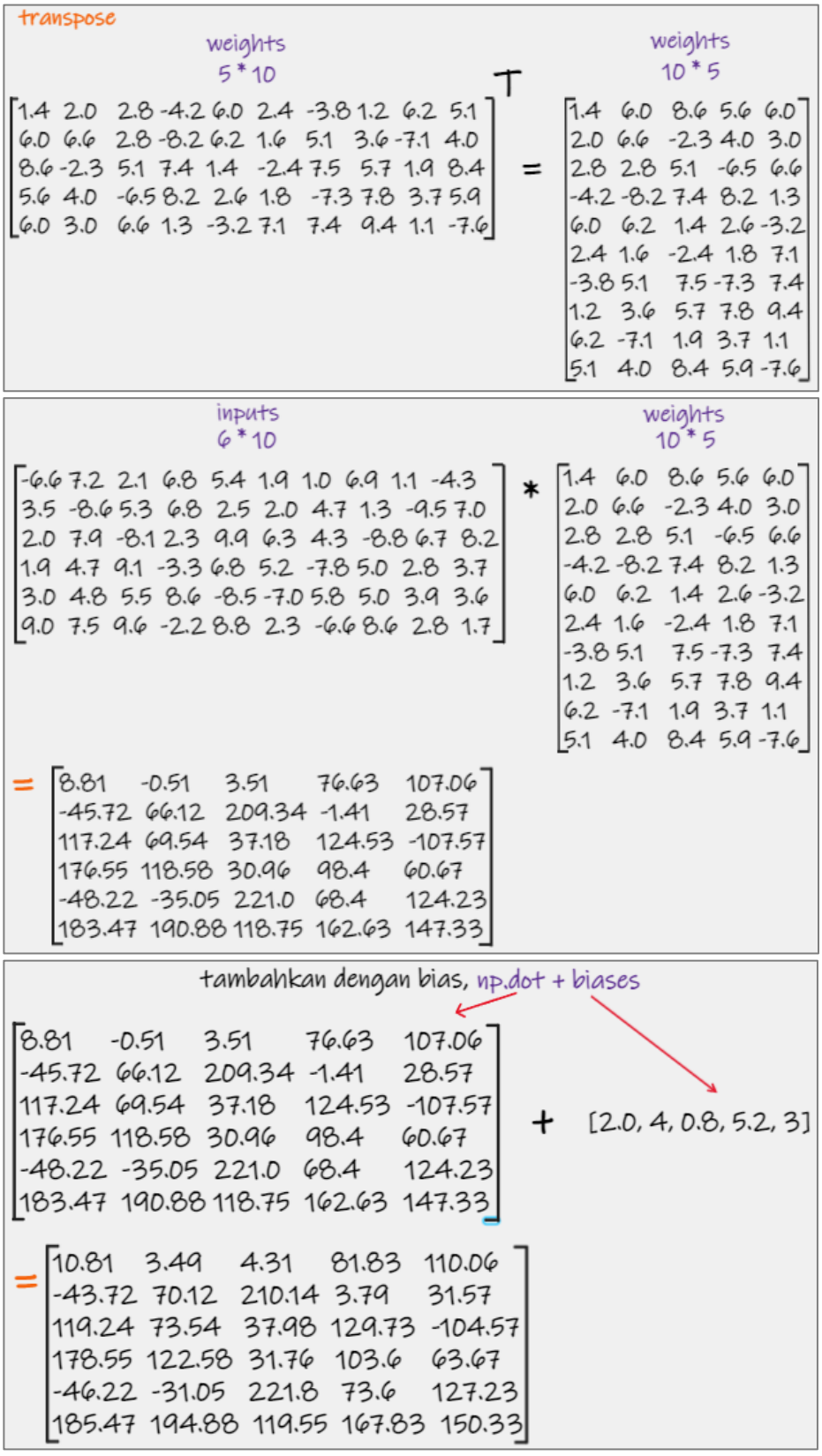
- Langkah pertama yaitu mengimpor NumPy (library Python) yang bisa digunakan untuk melakukan operasi vektor dan matriks. Lalu mengisi nilai dari setiap variabel inputs, weights, dan biases sesuai ketentuan soal yaitu per batch-nya 6 input dengan jumlah tiap input layer 10 sehingga inputs = 6 * 10 dan 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.

```
# Inisialisasi library NumPy
import numpy as np

# Inisialisasi variabel inputs
inputs = [[-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]]

# Inisialisasi variabel weights1 [hidden layer 1, 5 neuron]
weights1 = [[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]]

# Inisialisasi variabel biases1 [hidden layer 1, 5 neuron]
biases1 = [2.0, 4, 0.8, 5.2, 3]

# Inisialisasi variabel weights2 [hidden layer 2, 3 neuron]
weights2 = [[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]]

# Inisialisasi variabel biases2 [hidden layer 2, 3 neuron]
biases2 = [1.8, 6, 4.5]
```

- Melakukan operasi penghitungan pada hidden layer 1.

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

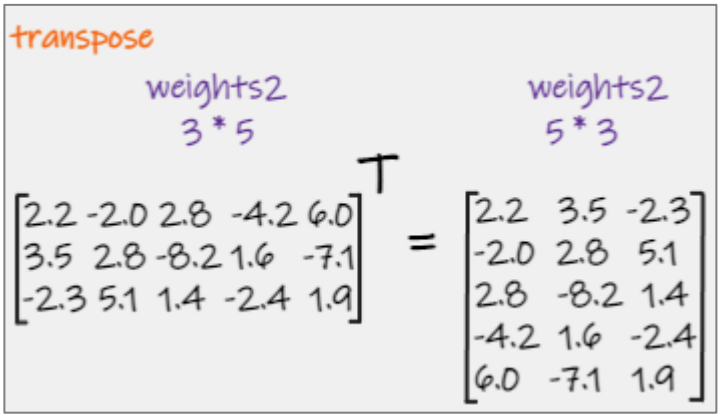
Transpose weights terlebih dahulu agar ordonya sesuai dengan variabel inputs sehingga operasi perhitungan bisa dilakukan.

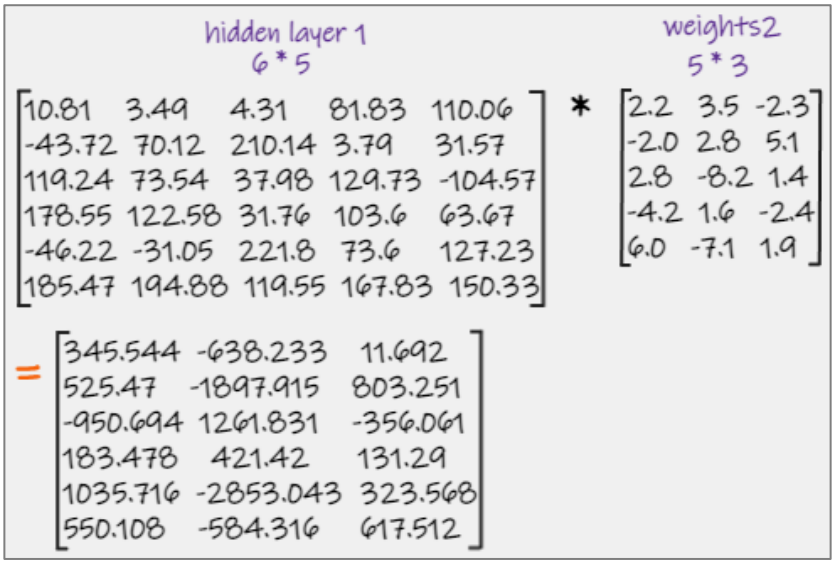


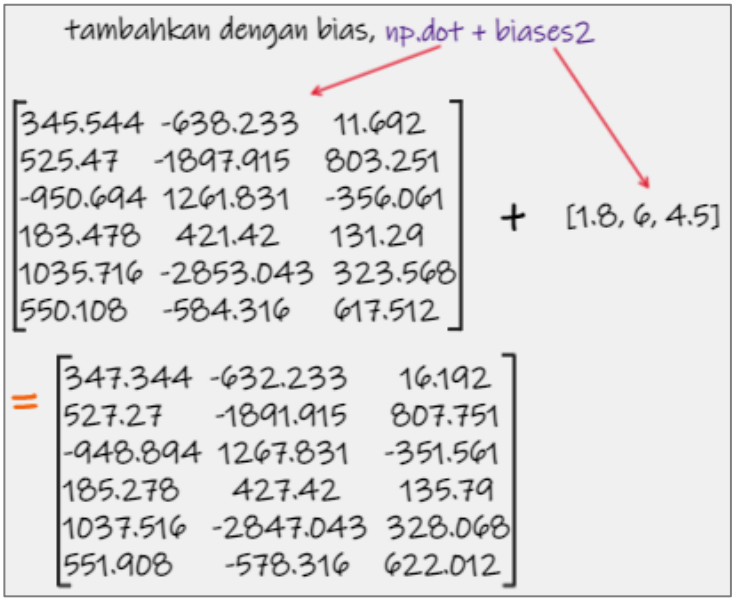
- Melakukan operasi penghitungan pada hidden layer 2.

```
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.







- 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]
```