LAPORAN UJIAN TENGAH SEMESTER
(KECERDASAN BUATAN)

Oleh:

Nur Haslinda (21091397035)

PROGRAM STUDI D4 MANAJEMEN INFORMATIKA

FAKULTAS VOKASI

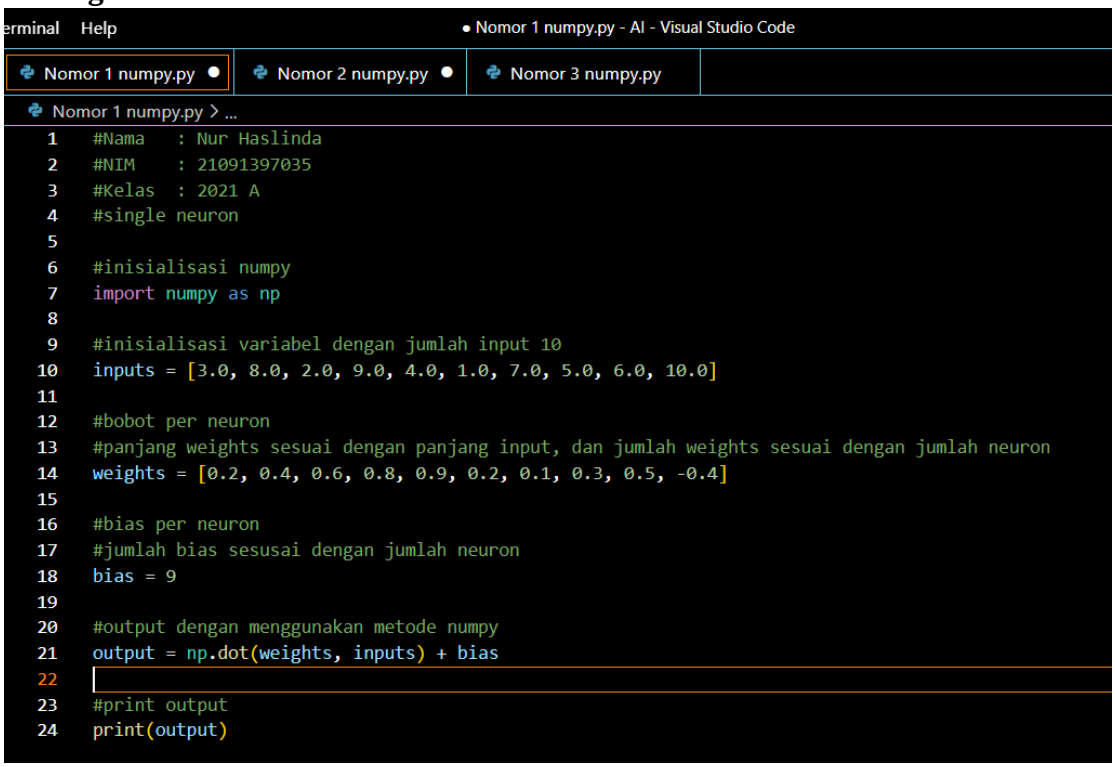
UNIVERSITAS NEGERI SURABAYA

2022

UTS 1

1. Single neuron
   - Input layer feature 10
   - Neuron 1

Codingan

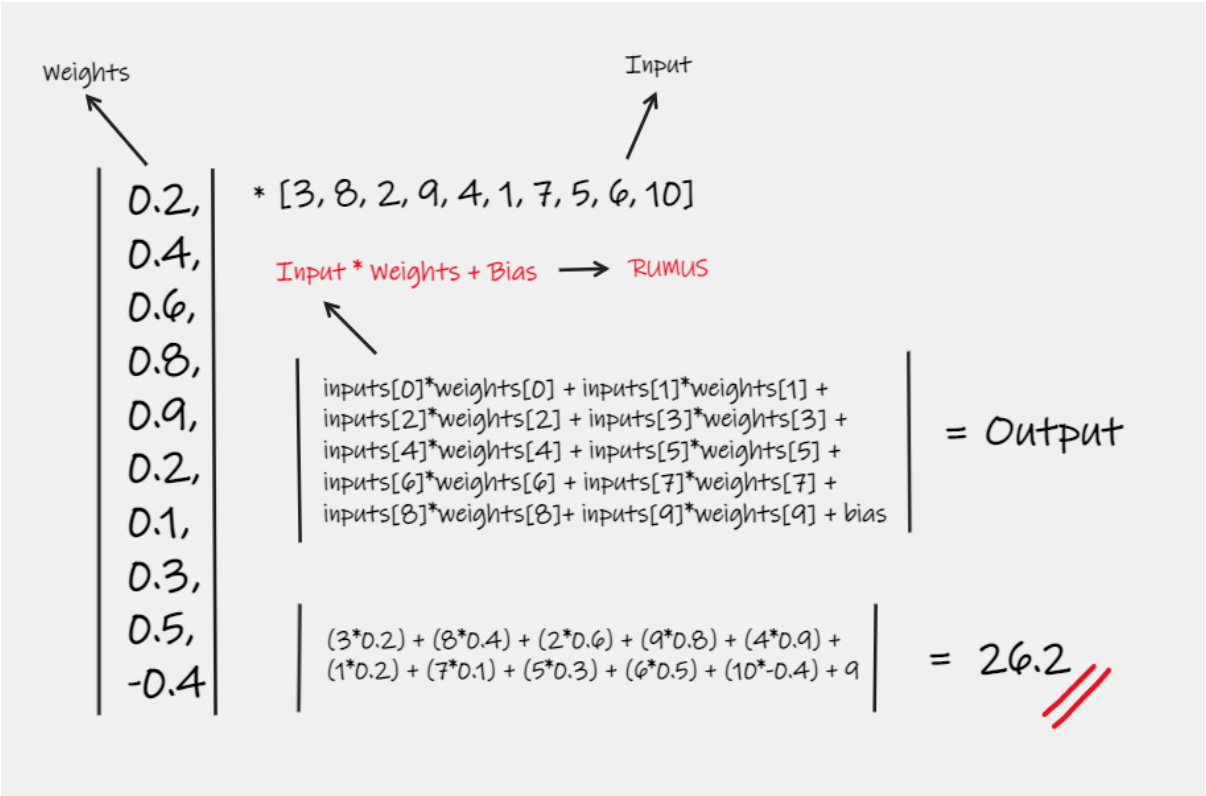


Output

Analisis

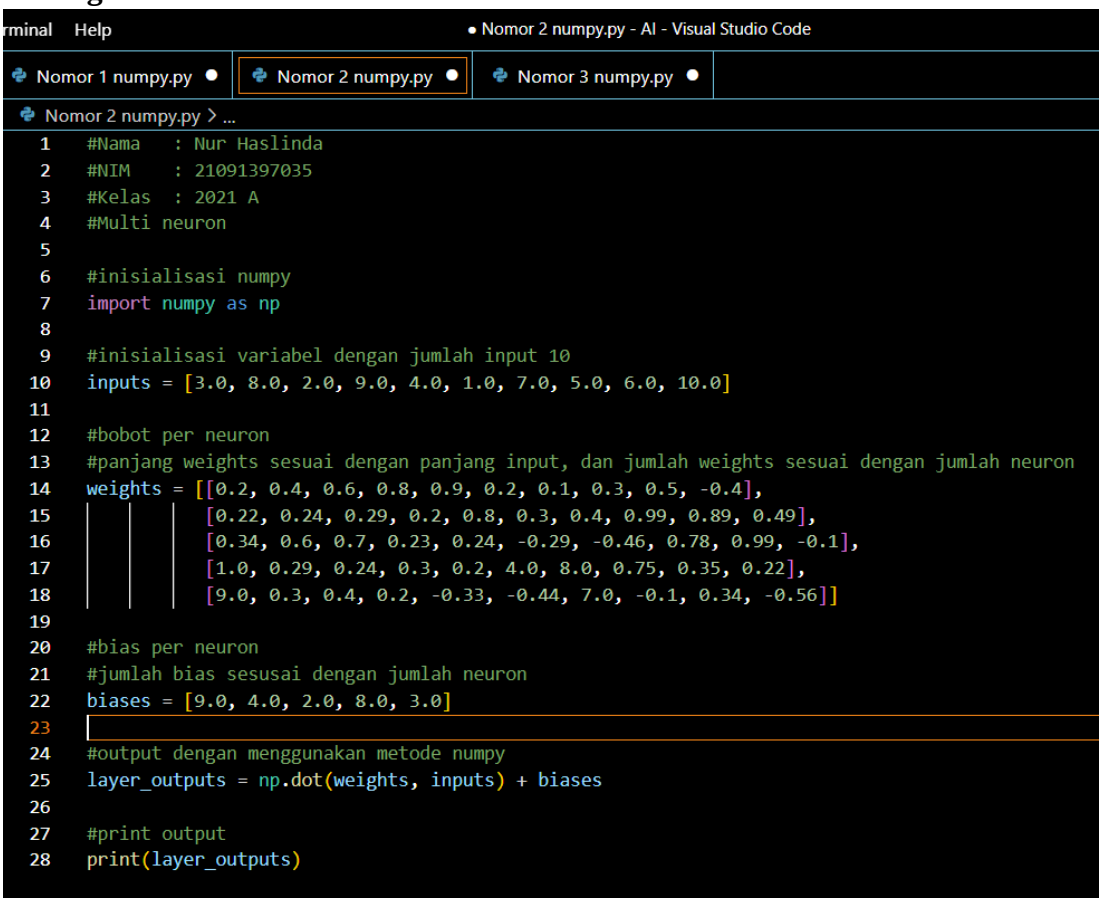
- Berdasarkan program single neuron diatas, jumlah input yang dimiliki adalah 10, jumlah weightnya adalah 1, dan jumlah biasnya adalah 1.
- Perhitungan dot product pada program ini menggunakan fungsi np.dot yang digunakan untuk menghitung input dan weightnya secara lebih praktis, kemudian pada akhir perhitungan ditambahkan dengan bias.

Cara Perhitungan

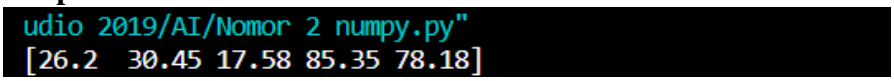


2. Single multi neuron
   - Input layer feature 10
   - Neuron 5

Codingan



Output



Analisis

- Berdasarkan program diatas, multi neuron memiliki jumlah neuron yang lebih dari satu. Dari soal diatas jumlah input yang dimasukkan adalah 10, jumlah weightsnya adalah 5 (berdasarkan dengan jumlah neuron), dan jumlah biasnya adalah 5 (berdasarkan dengan jumlah neuron).
- Kemudian perhitungan dot product pada program ini dengan menggunakan fungsi np.dot. Fungsi ini digunakan untuk mengoperasikan perkalian input, dan wights, kemudian menambahkan bias pada akhir perhitungan.
- Dengan menggunakan fungsi np.dot penulisan pada pada programnya juga menjadi lebih praktis.

Cara Perhitungan

Input =

Jumlah inputnya adalah 10 (1x10)

```
[3.0, 8.0, 2.0, 9.0, 4.0, 1.0, 7.0, 5.0, 6.0, 10.0]
```

Weights =

Jumlah baris dalam weights sesuai dengan jumlah neuron dan jumlah kolom dalam weights (5x10)

- Weights baris satu merupakan neuron 1
- Weights baris dua merupakan neuron 2
- Weights baris tiga merupakan neuron 3
- Weights baris empat merupakan neuron 4
- Weights baris lima merupakan neuron 5

```
[[0.2, 0.4, 0.6, 0.8, 0.9, 0.2, 0.1, 0.3, 0.5, -0.4],
 [0.22, 0.24, 0.29, 0.2, 0.8, 0.3, 0.4, 0.99, 0.89, 0.49],
 [0.34, 0.6, 0.7, 0.23, 0.24, -0.29, -0.46, 0.78, 0.99, -0.1],
 [1.0, 0.29, 0.24, 0.3, 0.2, 4.0, 8.0, 0.75, 0.35, 0.22],
 [9.0, 0.3, 0.4, 0.2, -0.33, -0.44, 7.0, -0.1, 0.34, -0.56]]
```

Kemudian input dikalikan dengan weights lalu dijumlahkan dengan biases.

Biases =

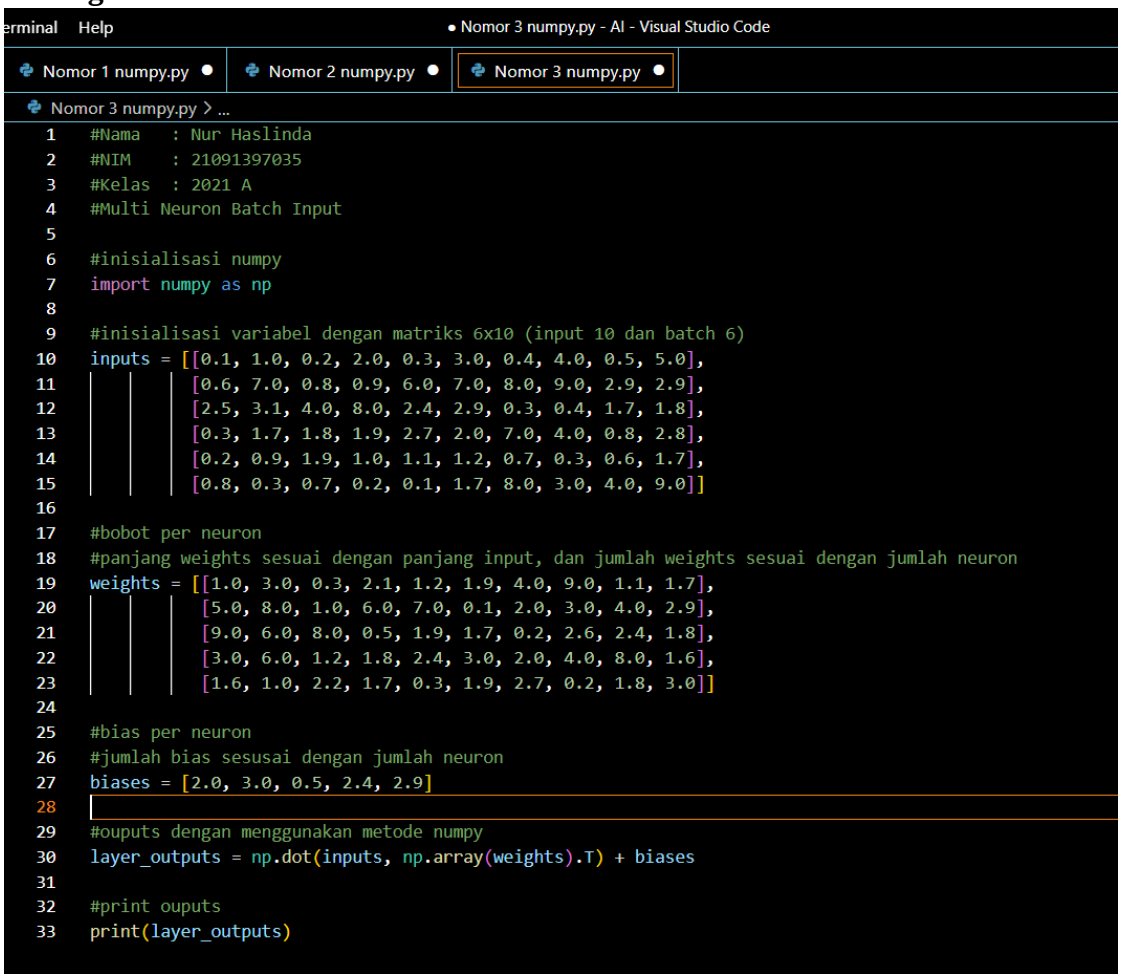
```
[9.0, 4.0, 2.0, 8.0, 3.0]
```

Rumus = np.dot(weights, inputs)+ Biases

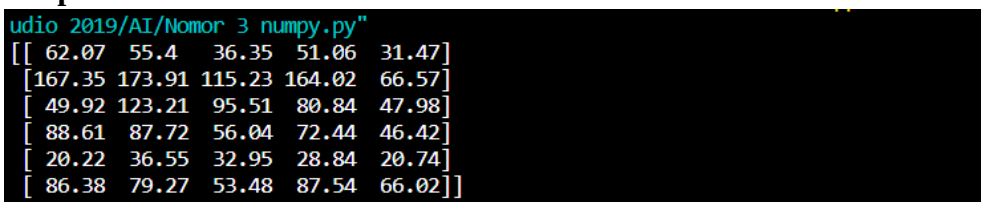
```
#output
outputs = [
    #neuron1
    inputs[0]* (variable) weights1: list[float] ] + inputs[2]*weights1[2] +
    inputs[3]*weights1[3] + inputs[4]*weights1[4] + inputs[5]*weights1[5] +
    inputs[6]*weights1[6] + inputs[7]*weights1[7] + inputs[8]*weights1[8] +
    inputs[9]*weights1[9] + bias1,

    #neuron2
    inputs[0]*weights2[0] + inputs[1]*weights2[1] + inputs[2]*weights2[2] +
    inputs[3]*weights2[3] + inputs[4]*weights2[4] + inputs[5]*weights2[5] +
    inputs[6]*weights2[6] + inputs[7]*weights2[7] + inputs[8]*weights2[8] +
    inputs[9]*weights2[9] + bias2,

    #neuron3
    inputs[0]*weights3[0] + inputs[1]*weights3[1] + inputs[2]*weights3[2] +
    inputs[3]*weights3[3] + inputs[4]*weights3[4] + inputs[5]*weights3[5] +
    inputs[6]*weights3[6] + inputs[7]*weights3[7] + inputs[8]*weights3[8] +
    inputs[9]*weights3[9] + bias3,

    #neuron4
    inputs[0]*weights4[0] + inputs[1]*weights4[1] + inputs[2]*weights4[2] +
    inputs[3]*weights4[3] + inputs[4]*weights4[4] + inputs[5]*weights4[5] +
    inputs[6]*weights4[6] + inputs[7]*weights4[7] + inputs[8]*weights4[8] +
    inputs[9]*weights4[9] + bias4,

    #neuron5
    inputs[0]*weights5[0] + inputs[1]*weights5[1] + inputs[2]*weights5[2] +
    inputs[3]*weights5[3] + inputs[4]*weights5[4] + inputs[5]*weights5[5] +
    inputs[6]*weights5[6] + inputs[7]*weights5[7] + inputs[8]*weights5[8] +
    inputs[9]*weights5[9] + bias5
]
```

3. Multi neuron batch input
   - Input layer feature 10
   - Per batch nya 6 input
   - Neuron 5

Codingan



Output



Analisis

- Program diatas merupakan program multi neuron batch input. Input yang dimiliki pada program tersebut berjumlah 10 dengan jumlah batch 6 (6x10), weights pada program tersebut berjumlah 5 (berdasarkan dengan jumlah neuron), dan bias pada program tersebut berjumlah 5 (berdasarkan dengan jumlah neuron).

- Perhitungan dot product pada program ini menggunakan fungsi np.dot dengan mengoperasikan input dan array pada weights, lalu ditranspose, dan yang terakhir ditambahkan dengan bias.

Cara Perhitungan

Inputs =

Input berjumlahkan 10 dengan batch 6. Jumlah baris sesuai dengan jumlah input layer, dan jumlah kolom sesuai dengan jumlah batch input (6x10)

```
[[0.1, 1.0, 0.2, 2.0, 0.3, 3.0, 0.4, 4.0, 0.5, 5.0],
 [0.6, 7.0, 0.8, 0.9, 6.0, 7.0, 8.0, 9.0, 2.9, 2.9],
 [2.5, 3.1, 4.0, 8.0, 2.4, 2.9, 0.3, 0.4, 1.7, 1.8],
 [0.3, 1.7, 1.8, 1.9, 2.7, 2.0, 7.0, 4.0, 0.8, 2.8],
 [0.2, 0.9, 1.9, 1.0, 1.1, 1.2, 0.7, 0.3, 0.6, 1.7],
 [0.8, 0.3, 0.7, 0.2, 0.1, 1.7, 8.0, 3.0, 4.0, 9.0]]
```

Weights =

Jumlah baris dalam weights sesuai dengan jumlah neuron, dan jumlah kolom dalam weights sesuai dengan jumlah input (5x10)

- Weights baris satu merupakan neuron 1
- Weights baris dua merupakan neuron 2
- Weights baris tiga merupakan neuron 3
- Weights baris empat merupakan neuron 4
- Weights baris lima merupakan neuron 5

```
[[1.0, 3.0, 0.3, 2.1, 1.2, 1.9, 4.0, 9.0, 1.1, 1.7],
 [5.0, 8.0, 1.0, 6.0, 7.0, 0.1, 2.0, 3.0, 4.0, 2.9],
 [9.0, 6.0, 8.0, 0.5, 1.9, 1.7, 0.2, 2.6, 2.4, 1.8],
 [3.0, 6.0, 1.2, 1.8, 2.4, 3.0, 2.0, 4.0, 8.0, 1.6],
 [1.6, 1.0, 2.2, 1.7, 0.3, 1.9, 2.7, 0.2, 1.8, 3.0]]
```

Kemudian weights ditranspose kan menjadi jumlah barisnya adalah 10 dan jumlah kolomnya adalah 5 (10x5).

```
[1.0, 5.0, 9.0, 3.0, 1.6]
[3.0, 8.0, 6.0, 6.0, 1.0]
[0.3, 1.0, 8.0, 1.2, 2.2]
[2.1, 6.0, 0.5, 1.8, 1.7]
[1.2, 7.0, 1.9, 2.4, 3.0]
[1.9, 0.1, 1.7, 3.0, 1.9]
[4.0, 2.0, 0.2, 2.0, 2.7]
[9.0, 3.0, 2.6, 4.0, 0.2]
[1.1, 4.0, 2.4, 8.0, 1.8]
[1.7, 2.9, 1.8, 1.6, 3.0]
```

Weights yang sudah ditransposekan, dikalikan dengan inputs lalu dijumlahkan dengan biases.

Biases =

```
[2.0, 3.0, 0.5, 2.4, 2.9]
```

Rumus = np.dot (inputs, np.array(weights).T) + biases

Output yang dihasilkan adalah

```
[[ 62.07  55.4   36.35  51.06  31.47]
 [167.35 173.91 115.23 164.02  66.57]
 [ 49.92 123.21  95.51  80.84  47.98]
 [ 88.61  87.72  56.04  72.44  46.42]
 [ 20.22  36.55  32.95  28.84  20.74]
 [ 86.38  79.27  53.48  87.54  66.02]]
```

UTS 2

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

Codingan

```python
#Nama    : Nur Haslinda
#NIM     : 21091397035
#Kelas   : 2021 A
#Multi Neuron Batch Input 2 layer

#inisialisasi numpy
import numpy as np

#inisialisasi variabel dengan matriks 6x10 (input 10 dan batch 6)
inputs = [[0.1, 1.0, 0.2, 2.0, 0.3, 3.0, 0.4, 4.0, 0.5, 5.0],
          [0.6, 7.0, 0.8, 0.9, 6.0, 7.0, 8.0, 9.0, 2.9, 2.9],
          [2.5, 3.1, 4.0, 8.0, 2.4, 2.9, 0.3, 0.4, 1.7, 1.8],
          [0.3, 1.7, 1.8, 1.9, 2.7, 2.0, 7.0, 4.0, 0.8, 2.8],
          [0.2, 0.9, 1.9, 1.0, 1.1, 1.2, 0.7, 0.3, 0.6, 1.7],
          [0.8, 0.3, 0.7, 0.2, 0.1, 1.7, 8.0, 3.0, 4.0, 9.0]]

#bobot per neuron layer 1
#panjang weights layer 1 sesuai dengan panjang input, dan jumlah weights sesuai dengan jumlah neuron layer 1 (10x5)
weights1 = [[1.0, 3.0, 0.3, 2.1, 1.2, 1.9, 4.0, 9.0, 1.1, 1.7],
            [5.0, 8.0, 1.0, 6.0, 7.0, 0.1, 2.0, 3.0, 4.0, 2.9],
            [9.0, 6.0, 8.0, 0.5, 1.9, 1.7, 0.2, 2.6, 2.4, 1.8],
            [3.0, 6.0, 1.2, 1.8, 2.4, 3.0, 2.0, 4.0, 8.0, 1.6],
            [1.6, 1.0, 2.2, 1.7, 0.3, 1.9, 2.7, 0.2, 1.8, 3.0]]

#bias per neuron layer 1
#jumlah bias layer 1 sesuai dengan jumlah neuron layer 1
biases = [2.0, 3.0, 0.5, 2.4, 2.9]

#bobot per neuron layer 2
#panjang weights layer 2 sesuai dengan panjang outputs layer 1 dan jumlah weights sesuai dengan jumlah neuron layer 2 (5x3)
weights2 = [[2.5, 2.4, 0.1, 4.8, 2.9,],
            [3.6, 2.5, 7.0, 1.0, 6.0],
            [0.8, 0.4, 0.3, 0.2, 0.9]]

#bias per neuron layer 2
#Jumlah bias layer 2 sesuai dengan jumlah neuron layer 2
biases2 = [4.0, 9.0, 1.7]

#ouputs dengan menggunakan metode numpy
layer1_outputs = np.dot(inputs, np.array(weights1).T) + biases
layer2_outputs = np.dot (layer1_outputs, np.array(weights2).T) + biases2

#print ouputs
print(layer2_outputs)
```

Output

```
PS C:\Users\nrhas\Documents\Visual Studio 2019\AI> & C:/
ments/Visual Studio 2019/AI/Nur Haslinda_035_UTS2.py"
[[ 632.121  865.282  122.956]
 [1831.631 2416.285  332.43 ]
 [ 961.229 1534.027  178.923]
 [ 923.987 1290.536  180.754]
 [ 344.143  557.097   66.815]
 [1027.196 1376.163  195.482]]
PS C:\Users\nrhas\Documents\Visual Studio 2019\AI>
```

Analisa

- Program diatas merupakan program multi neuron batch input 2 layer. Input yang dimiliki pada program tersebut berjumlah 10 dengan jumlah batch 6 (6x10). Pada layer 1, jumlah baris weights pada program tersebut berjumlah 5 (berdasarkan dengan jumlah neuron), jumlah kolom pada weights layer 1 adalah 10 (Berdasarkan jumlah input) (5x10), dan biases pada program tersebut berjumlah 5 (berdasarkan dengan jumlah neuron).
- Perhitungan dot product pada layer 1 menggunakan fungsi np.dot dengan mengoperasikan input dan array pada weights, lalu ditranspose, dan yang terakhir ditambahkan dengan bias layer 1.
- Kemudian pada layer ke 2 jumlah baris weights pada program tersebut adalah 3 (Berdasarkan jumlah neuron), jumlah kolom weights layer 2 adalah 5 (Berdasarkan jumlah kolom pada layer 1) (5x3), dan biases pada layer 2 berjumlahkan 3 (Berdasarkan dengan jumlah neuron)
- Perhitungan dot product pada layer 2 menggunakan fungsi np.dot dengan mengoperasikan output dari layer 1 dan array pada weights2, lalu ditranspose, dan yang terakhir ditambahkan dengan bias layer 2.

Cara Kerja

Inputs =

Input berjumlahkan 10 dengan batch 6. Jumlah baris sesuai dengan jumlah input layer, dan jumlah kolom sesuai dengan jumlah batch input (6x10)

```
[[0.1, 1.0, 0.2, 2.0, 0.3, 3.0, 0.4, 4.0, 0.5, 5.0],
 [0.6, 7.0, 0.8, 0.9, 6.0, 7.0, 8.0, 9.0, 2.9, 2.9],
 [2.5, 3.1, 4.0, 8.0, 2.4, 2.9, 0.3, 0.4, 1.7, 1.8],
 [0.3, 1.7, 1.8, 1.9, 2.7, 2.0, 7.0, 4.0, 0.8, 2.8],
 [0.2, 0.9, 1.9, 1.0, 1.1, 1.2, 0.7, 0.3, 0.6, 1.7],
 [0.8, 0.3, 0.7, 0.2, 0.1, 1.7, 8.0, 3.0, 4.0, 9.0]]
```

Weights1 =

Jumlah baris dalam weights1 sesuai dengan jumlah neuron, dan jumlah kolom dalam weights sesuai dengan jumlah input (5x10)

- Weights baris satu merupakan neuron 1
- Weights baris dua merupakan neuron 2
- Weights baris tiga merupakan neuron 3
- Weights baris empat merupakan neuron 4
- Weights baris lima merupakan neuron 5

```
[[1.0, 3.0, 0.3, 2.1, 1.2, 1.9, 4.0, 9.0, 1.1, 1.7],
 [5.0, 8.0, 1.0, 6.0, 7.0, 0.1, 2.0, 3.0, 4.0, 2.9],
 [9.0, 6.0, 8.0, 0.5, 1.9, 1.7, 0.2, 2.6, 2.4, 1.8],
 [3.0, 6.0, 1.2, 1.8, 2.4, 3.0, 2.0, 4.0, 8.0, 1.6],
 [1.6, 1.0, 2.2, 1.7, 0.3, 1.9, 2.7, 0.2, 1.8, 3.0]]
```

Kemudian weights ditranspose kan menjadi jumlah barisnya adalah 10 dan jumlah kolomnya adalah 5 (10x5).

```
[1.0, 5.0, 9.0, 3.0, 1.6]
[3.0, 8.0, 6.0, 6.0, 1.0]
[0.3, 1.0, 8.0, 1.2, 2.2]
[2.1, 6.0, 0.5, 1.8, 1.7]
[1.2, 7.0, 1.9, 2.4, 3.0]
[1.9, 0.1, 1.7, 3.0, 1.9]
[4.0, 2.0, 0.2, 2.0, 2.7]
[9.0, 3.0, 2.6, 4.0, 0.2]
[1.1, 4.0, 2.4, 8.0, 1.8]
[1.7, 2.9, 1.8, 1.6, 3.0]
```

Weights yang sudah ditransposekan, dikalikan dengan inputs lalu dijumlahkan dengan biases layer 2.

Biases layer 1 =

```
[2.0, 3.0, 0.5, 2.4, 2.9]
```

Rumus layer 1= np.dot (inputs, np.array(weights1).T) + biases

Output yang dihasilkan adalah

```
[[ 62.07  55.4   36.35  51.06  31.47]
 [167.35 173.91 115.23 164.02  66.57]
 [ 49.92 123.21  95.51  80.84  47.98]
 [ 88.61  87.72  56.04  72.44  46.42]
 [ 20.22  36.55  32.95  28.84  20.74]
 [ 86.38  79.27  53.48  87.54  66.02]]
```

Weights 2

Jumlah baris dalam weights 2 sesuai dengan jumlah neuron, dan jumlah kolom dalam weights 2 sesuai dengan jumlah kolom pada output layer 1 (3x5).

- Weights baris satu merupakan neuron 1
- Weights baris dua merupakan neuron 2
- Weights baris tiga merupakan neuron 3

```
[[2.5, 2.4, 0.1, 4.8, 2.9,],
 [3.6, 2.5, 7.0, 1.0, 6.0],
 [0.8, 0.4, 0.3, 0.2, 0.9]]
```

Kemudian weights 2 ditranspose menjadi jumlah barisnya 5 dan jumlah kolomnya 3 (5x3).

```
[2.5, 3.6, 0.8]
[2.4, 2.5, 0.4]
[0.1, 7.0, 0.3]
[4.8, 1.0, 0.2]
[2.9, 6.0, 0.9]
```

Kemudian output layer 1 dikalikan dengan weights 2 yang sudah ditranspose, lalu dijumlahkan dengan biases layer 2.

Biases layer 2 =

```
[4.0, 9.0, 1.7]
```

Rumus layer 1= np.dot (layer1_outputs, np.array(weights2).T) + biases2

Output yang dihasilkan adalah

```
[[ 632.121  865.282  122.956]
 [1831.631 2416.285  332.43 ]
 [ 961.229 1534.027  178.923]
 [ 923.987 1290.536  180.754]
 [ 344.143  557.097   66.815]
 [1027.196 1376.163  195.482]]
```