

Ujian Tengah Semester Kecerdasan Buatan
Single Neuron, Multiple Neuron, Multiple Neuron Batch
Multiple Neuron Batch and Layers

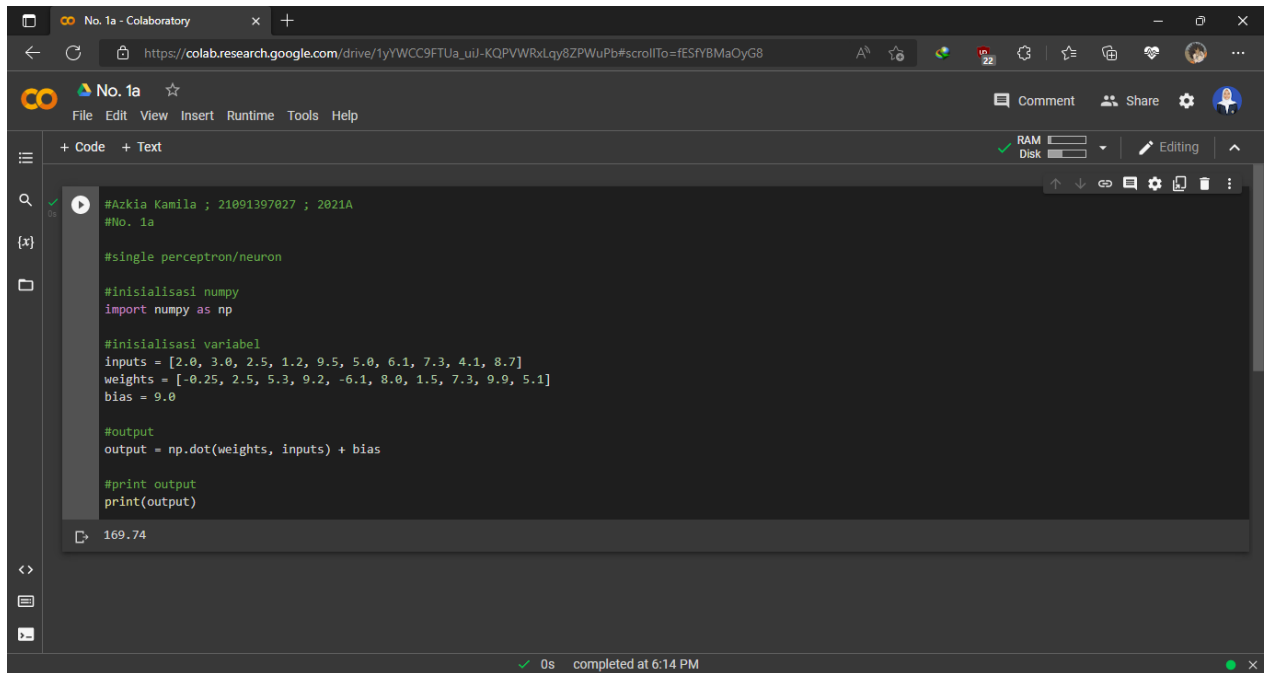


Disusun oleh:
Azkia Kamila (21091397027)

Program Studi D4 Manajemen Informatika
Fakultas Vokasi
Universitas Negeri Surabaya
2022

UTS 1

a. Source Code:



```
#Azkia Kamila ; 21091397027 ; 2021A
#No. 1a

#single perceptron/neuron

#inisialisasi numpy
import numpy as np

#inisialisasi variabel
inputs = [2.0, 3.0, 2.5, 1.2, 9.5, 5.0, 6.1, 7.3, 4.1, 8.7]
weights = [-0.25, 2.5, 5.3, 9.2, -6.1, 8.0, 1.5, 7.3, 9.9, 5.1]
bias = 9.0

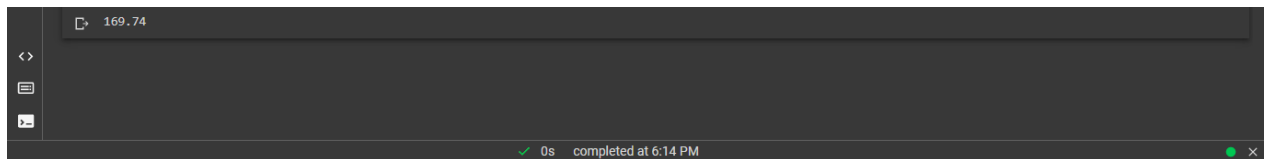
#output
output = np.dot(weights, inputs) + bias

#print output
print(output)
```

169.74

0s completed at 6:14 PM

Output:



```
169.74
```

0s completed at 6:14 PM

Output Result: 169.74

Analisis:

1. Inisialisasi numpy sebagai method perhitungan
2. Masukkan variabel untuk inputs, weights, dan bias sesuai dengan ketentuan
Inputs = 10*1
Weights = 1*10
Neuron = 1
Bias = 1
3. Buat output untuk menghitung variabel yang telah kita masukkan/buat
np.dot = untuk penghitungan vektor weight dan input
kemudian hasil penghitungan vektor ditambahkan dengan bias
4. Buat command print untuk menampilkan hasil perhitungan output

perhitungan dot product

weights
10*1

inputs
1*10

$\begin{bmatrix} -0.25 \\ 2.5 \\ 5.3 \\ 9.2 \\ -6.1 \\ 8.0 \\ 1.5 \\ 7.3 \\ 9.9 \\ 5.1 \end{bmatrix}$

*

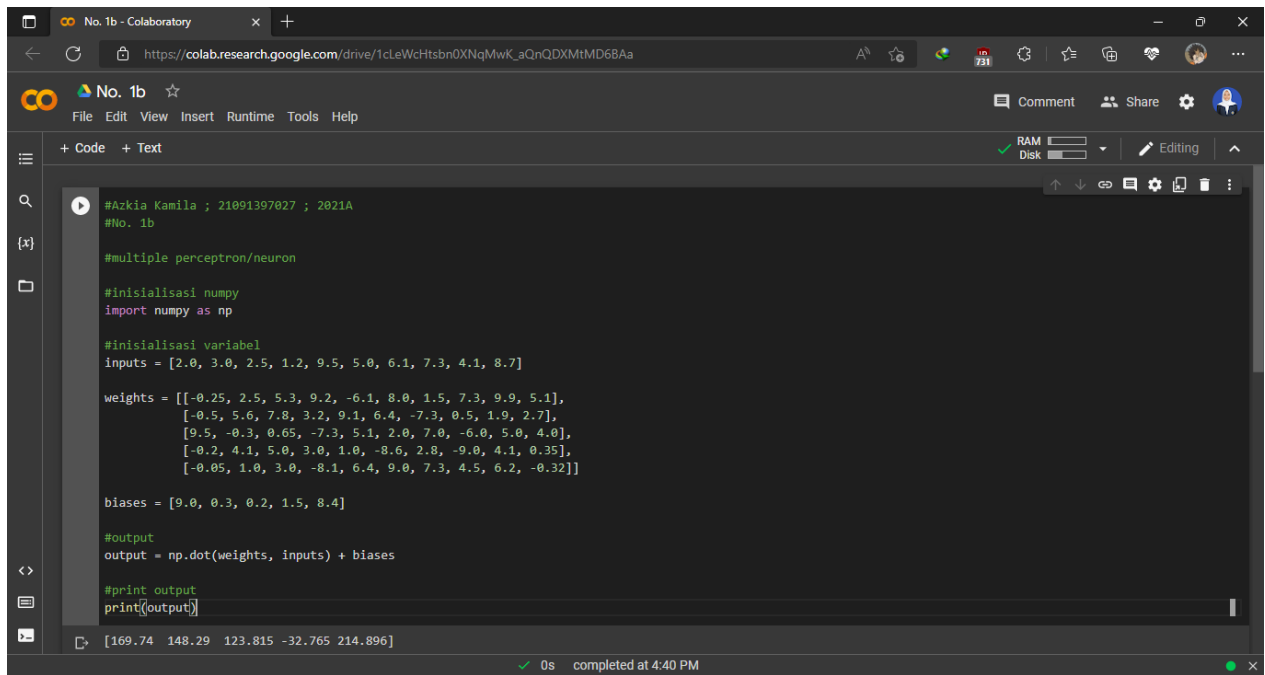
[2.0, 3.0, 2.5, 1.2, 9.5, 5.0, 6.1, 7.3, 4.1, 8.7]

weights * inputs = 160.74

kemudian np.dot + bias

$$160.74 + 9.0 = 169.74 //$$

b. Source Code:



The screenshot shows a Google Colab notebook interface. The browser address bar displays the URL: https://colab.research.google.com/drive/1cLeWcHtsbn0XNqMwK_aQnQDXMtMD68Aa. The notebook is titled "No. 1b" and has tabs for "Code" and "Text". The code editor contains the following Python code:

```
#Azkia Kamila ; 21091397027 ; 2021A
#No. 1b

#multiple perceptron/neuron

#inisialisasi numpy
import numpy as np

#inisialisasi variabel
inputs = [2.0, 3.0, 2.5, 1.2, 9.5, 5.0, 6.1, 7.3, 4.1, 8.7]

weights = [[-0.25, 2.5, 5.3, 9.2, -6.1, 8.0, 1.5, 7.3, 9.9, 5.1],
            [-0.5, 5.6, 7.8, 3.2, 9.1, 6.4, -7.3, 0.5, 1.9, 2.7],
            [9.5, -0.3, 0.65, -7.3, 5.1, 2.0, 7.0, -6.0, 5.0, 4.0],
            [-0.2, 4.1, 5.0, 3.0, 1.0, -8.6, 2.8, -9.0, 4.1, 0.35],
            [-0.05, 1.0, 3.0, -8.1, 6.4, 9.0, 7.3, 4.5, 6.2, -0.32]]

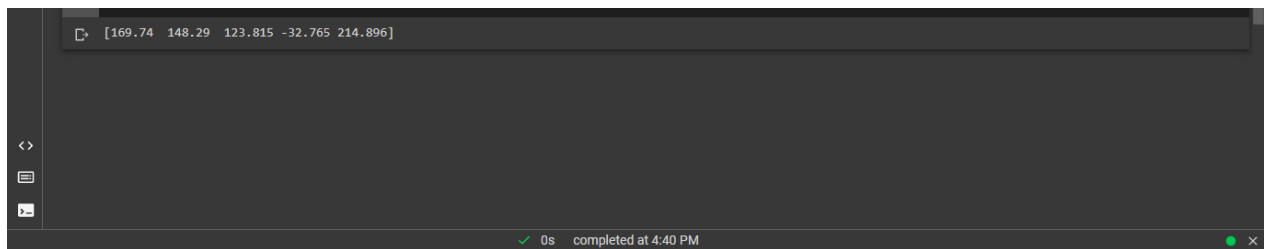
biases = [9.0, 0.3, 0.2, 1.5, 8.4]

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

#print output
print(output)
```

The output of the code is displayed at the bottom of the notebook: `[169.74 148.29 123.815 -32.765 214.896]`. The status bar indicates that the code was executed successfully (green checkmark) and completed at 4:40 PM.

Output:



The screenshot shows the output of the code execution, which is a NumPy array: `[169.74 148.29 123.815 -32.765 214.896]`. The status bar indicates that the code was executed successfully (green checkmark) and completed at 4:40 PM.

Output Result: [169.75 148.29 123.815 -32.765 214.896]

Analisis:

1. Inisialisasi numpy sebagai method perhitungan
2. Masukkan variabel untuk inputs, weights, dan bias sesuai dengan ketentuan
Inputs = 10
Weights = 5*10
Neuron = 5
Biases = 5
3. Buat output untuk menghitung variabel yang telah kita masukkan/buat
np.dot = untuk penghitungan vektor weight dan input
kemudian hasil penghitungan vektor ditambahkan dengan biases
4. Buat command print untuk menampilkan hasil perhitungan output

perhitungan dot product

weights 10*5		inputs 1*10
$\begin{bmatrix} -0.25 & 2.5 & 5.3 & 9.2 & -6.1 & 8.0 & 1.5 & 7.3 & 9.9 & 5.1 \\ -0.5 & 5.6 & 7.8 & 3.2 & 9.1 & 6.4 & -7.3 & 0.5 & 1.9 & 2.7 \\ 9.5 & 0.3 & 0.65 & -7.3 & 5.1 & 2.0 & 7.0 & -6.0 & 5.0 & 4.0 \\ -0.2 & 4.1 & 5.0 & 3.0 & 1.0 & -8.6 & 2.8 & -9.0 & 4.1 & 0.35 \\ -0.05 & 1.0 & 3.0 & -8.1 & 6.4 & 9.0 & 7.3 & 4.5 & 6.2 & -0.32 \end{bmatrix}$	*	$[2.0, 3.0, 2.5, 1.2, 9.5, 5.0, 6.1, 7.3, 4.1, 8.7]$

$(2 * -0.25)$	$(3 * 2.5)$	$(2.5 * 5.3)$	$(1.2 * 9.2)$	$(9.5 * -6.1)$	$(5 * 8)$	$(6.1 * 1.5)$	$(7.3 * 7.3)$	$(4.1 * 9.9)$	$(8.7 * 5.1)$	neuron 1
$(2 * 0.5)$	$(3 * 5.6)$	$(2.5 * 7.8)$	$(1.2 * 3.2)$	$(9.5 * 9.1)$	$(5 * 6.4)$	$(6.1 * -7.3)$	$(7.3 * 0.5)$	$(4.1 * 1.9)$	$(8.7 * 2.7)$	neuron 2
$(2 * 9.5)$	$(3 * -0.3)$	$(2.5 * 0.65)$	$(1.2 * -7.3)$	$(9.5 * 5.1)$	$(5 * 2)$	$(6.1 * 7)$	$(7.3 * -6)$	$(4.1 * 5)$	$(8.7 * 4)$	neuron 3
$(2 * -0.2)$	$(3 * 4.1)$	$(2.5 * 5)$	$(1.2 * 3)$	$(9.5 * 1)$	$(5 * -8.6)$	$(6.1 * 2.8)$	$(7.3 * -9)$	$(4.1 * 4.1)$	$(8.7 * 0.35)$	neuron 4
$(2 * -0.05)$	$(3 * 1)$	$(2.5 * 3)$	$(1.2 * -8.1)$	$(9.5 * 6.4)$	$(5 * 9)$	$(6.1 * 7.3)$	$(7.3 * 4.5)$	$(4.1 * 6.2)$	$(8.7 * -0.32)$	neuron 5

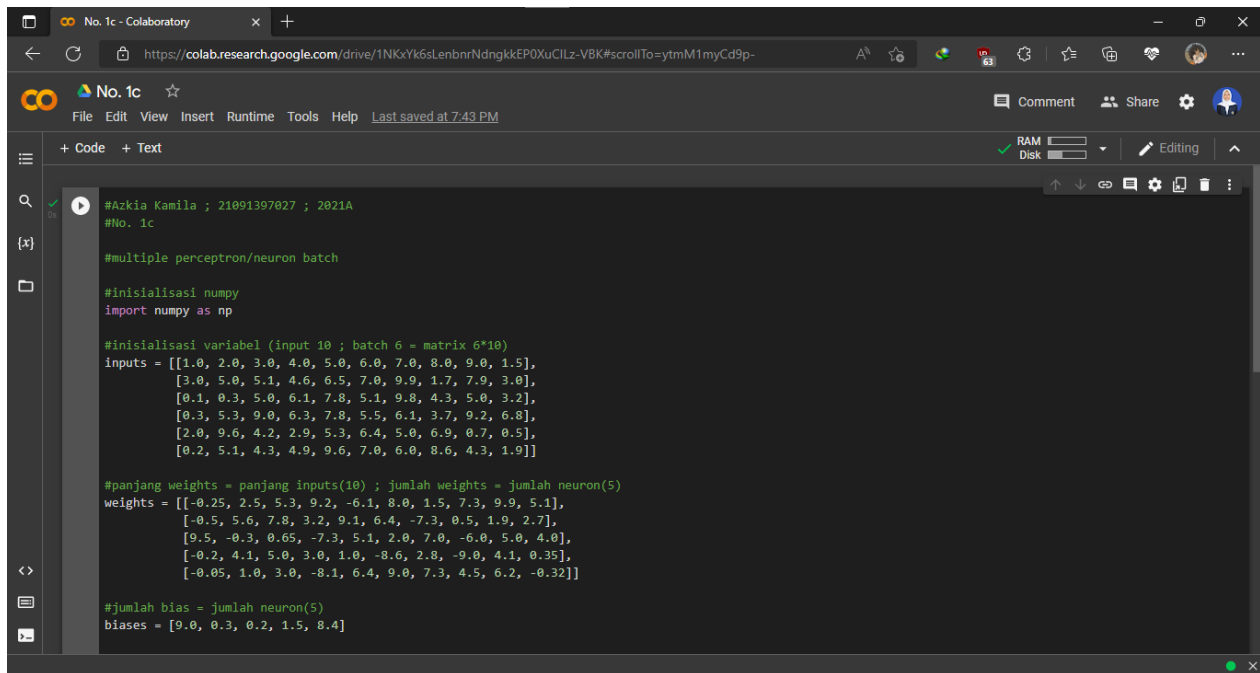
weights * inputs = $[160.74 ; 147.99 ; 123.615 ; -34.265 ; 206.496]$

kemudian np.dot + biases

$[160.74 ; 147.99 ; 123.615 ; -34.265 ; 206.496] + [9.0, 0.3, 0.2, 1.5, 8.4]$

$= [169.74 ; 148.29 ; 123.815 ; -32.765 ; 214.896]$

c. Source Code:



The screenshot shows a Google Colab notebook titled "No. 1c". The code defines inputs, weights, and biases for a neural network. The inputs are a 6x10 matrix, weights are a 6x5 matrix, and biases are a 1x5 vector.

```
#Azkia Kamila ; 21091397027 ; 2021A
#No. 1c

#multiple perceptron/neuron batch

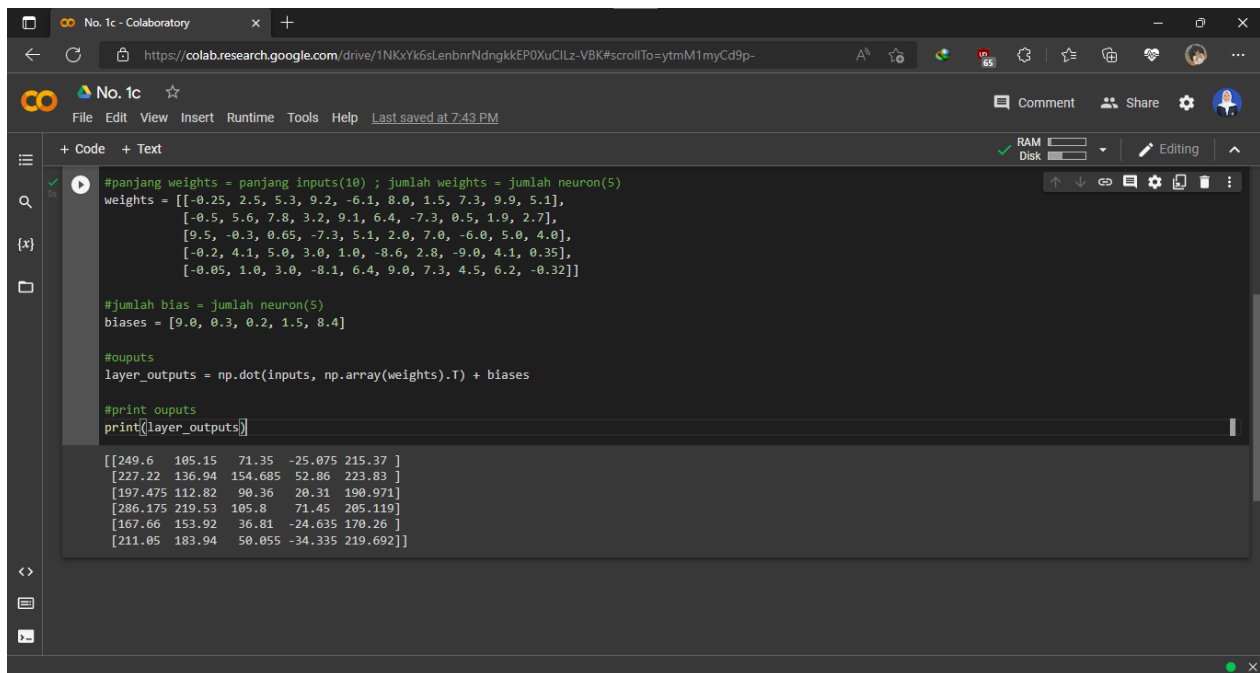
#inisialisasi numpy
import numpy as np

#inisialisasi variabel (input 10 ; batch 6 = matrix 6*10)
inputs = [[1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 1.5],
          [3.0, 5.0, 5.1, 4.6, 6.5, 7.0, 9.9, 1.7, 7.9, 3.0],
          [0.1, 0.3, 5.0, 6.1, 7.8, 5.1, 9.8, 4.3, 5.0, 3.2],
          [0.3, 5.3, 9.0, 6.3, 7.8, 5.5, 6.1, 3.7, 9.2, 6.8],
          [2.0, 9.6, 4.2, 2.9, 5.3, 6.4, 5.0, 6.9, 0.7, 0.5],
          [0.2, 5.1, 4.3, 4.9, 9.6, 7.0, 6.0, 8.6, 4.3, 1.9]]

#panjang weights = panjang inputs(10) ; jumlah weights = jumlah neuron(5)
weights = [[-0.25, 2.5, 5.3, 9.2, -6.1, 8.0, 1.5, 7.3, 9.9, 5.1],
           [-0.5, 5.6, 7.8, 3.2, 9.1, 6.4, -7.3, 0.5, 1.0, 2.7],
           [9.5, -0.3, 0.65, -7.3, 5.1, 2.0, 7.0, -6.0, 5.0, 4.0],
           [-0.2, 4.1, 5.0, 3.0, 1.0, -8.6, 2.8, -9.0, 4.1, 0.35],
           [-0.05, 1.0, 3.0, -8.1, 6.4, 9.0, 7.3, 4.5, 6.2, -0.32]]

#jumlah bias = jumlah neuron(5)
biases = [9.0, 0.3, 0.2, 1.5, 8.4]
```

Code c first part



The screenshot shows the second part of the source code, which calculates the layer outputs and prints them. The output is a 6x5 matrix.

```
#panjang weights = panjang inputs(10) ; jumlah weights = jumlah neuron(5)
weights = [[-0.25, 2.5, 5.3, 9.2, -6.1, 8.0, 1.5, 7.3, 9.9, 5.1],
           [-0.5, 5.6, 7.8, 3.2, 9.1, 6.4, -7.3, 0.5, 1.0, 2.7],
           [9.5, -0.3, 0.65, -7.3, 5.1, 2.0, 7.0, -6.0, 5.0, 4.0],
           [-0.2, 4.1, 5.0, 3.0, 1.0, -8.6, 2.8, -9.0, 4.1, 0.35],
           [-0.05, 1.0, 3.0, -8.1, 6.4, 9.0, 7.3, 4.5, 6.2, -0.32]]

#jumlah bias = jumlah neuron(5)
biases = [9.0, 0.3, 0.2, 1.5, 8.4]

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

#print outputs
print(layer_outputs)]
```

```
[[249.6  105.15  71.35 -25.075 215.37 ]
 [227.22 136.94 154.685 52.86 223.83 ]
 [197.475 112.82 90.36 20.31 190.071]
 [286.175 219.53 105.8 71.45 205.119]
 [167.66 153.92 36.81 -24.635 170.26 ]
 [211.05 183.94 50.055 -34.335 219.692]]
```

Code c second part

Output:

```
[[249.6  105.15  71.35  -25.075 215.37 ]
 [227.22 136.94 154.685 52.86  223.83 ]
 [197.475 112.82 90.36   20.31  190.971]
 [286.175 219.53 105.8   71.45  205.119]
 [167.66  153.92  36.81  -24.635 170.26 ]
 [211.05  183.94  50.055 -34.335 219.692]]
```

Output Result:

```
[[249.6  105.15  71.35  -25.075 215.37 ]
 [227.22 136.94 154.685 52.86  223.83 ]
 [197.475 112.82 90.36   20.31  190.971]
 [286.175 219.53 105.8   71.45  205.119]
 [167.66  153.92  36.81  -24.635 170.26 ]
 [211.05  183.94  50.055 -34.335 219.692]]
```

Analisis:

1. Inisialisasi numpy sebagai method perhitungan
2. Masukkan variabel untuk inputs, weights, dan bias sesuai dengan ketentuan
Inputs = 10
Batch = 6 } Inputs menjadi matric 6*10
Weights = 5*10
Neuron = 5
Biases = 5
3. Buat output untuk menghitung variabel yang telah kita masukkan/buat
np.dot = untuk penghitungan vektor weight dan input
kemudian hasil penghitungan vektor ditambahkan dengan biases
4. Buat command print untuk menampilkan hasil perhitungan output

perhitungan dot product

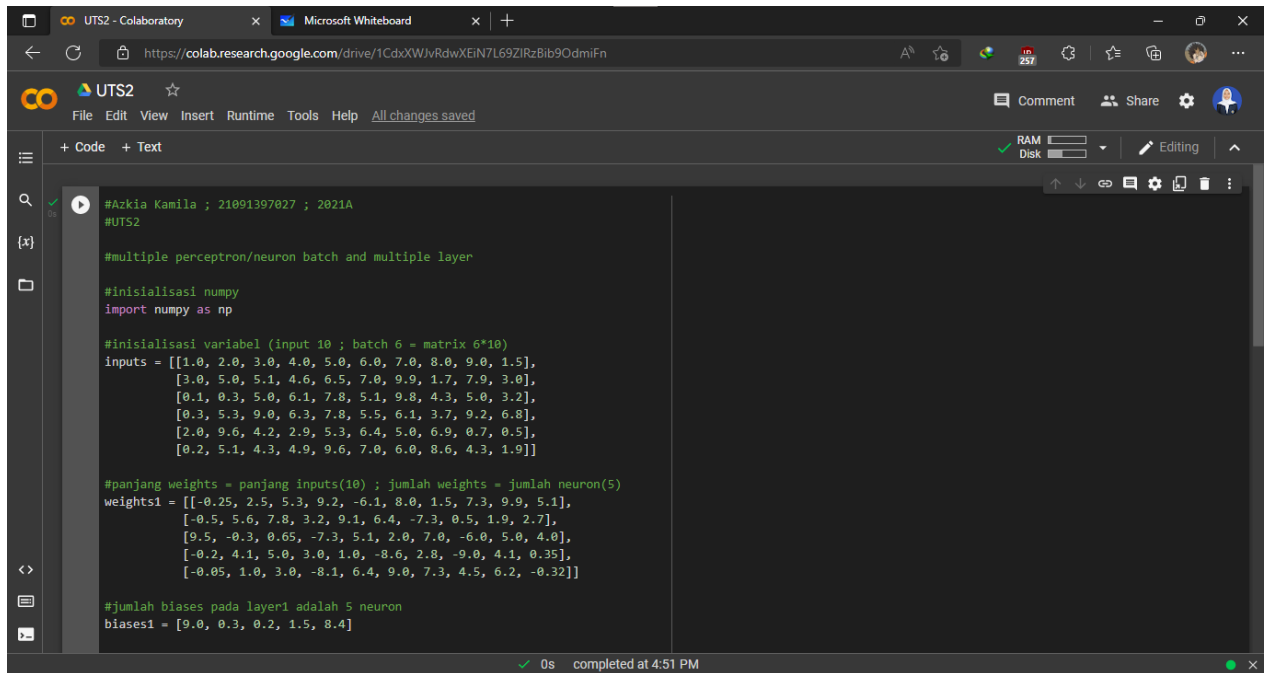
$$\begin{array}{c} \text{weights} \\ 10 \times 5 \end{array} \begin{bmatrix} -0.25 & 2.5 & 5.3 & 9.2 & -6.1 & 8.0 & 1.5 & 7.3 & 9.9 & 5.1 \\ -0.5 & 5.6 & 7.8 & 3.2 & 9.1 & 6.4 & -7.3 & 0.5 & 1.9 & 2.7 \\ 9.5 & 0.3 & 0.65 & -7.3 & 5.1 & 2.0 & 7.0 & -6.0 & 5.0 & 4.0 \\ -0.2 & 4.1 & 5.0 & 3.0 & 1.0 & -8.6 & 2.8 & -9.0 & 4.1 & 0.35 \\ -0.05 & 1.0 & 3.0 & -8.1 & 6.4 & 9.0 & 7.3 & 4.5 & 6.2 & -0.32 \end{bmatrix} * \begin{array}{c} \text{inputs} \\ 6 \times 10 \end{array} \begin{bmatrix} 1.0 & 2.0 & 3.0 & 4.0 & 5.0 & 6.0 & 7.0 & 8.0 & 9.0 & 1.5 \\ 3.0 & 5.0 & 5.1 & 4.6 & 6.5 & 7.0 & 9.9 & 1.7 & 7.9 & 3.0 \\ 0.1 & 0.3 & 5.0 & 6.1 & 7.8 & 5.1 & 9.8 & 4.3 & 5.0 & 3.2 \\ 0.3 & 5.3 & 9.0 & 6.3 & 7.8 & 5.5 & 6.1 & 3.7 & 9.2 & 6.8 \\ 2.0 & 9.6 & 4.2 & 2.9 & 5.3 & 6.4 & 5.0 & 6.9 & 0.7 & 0.5 \\ 0.2 & 5.1 & 4.3 & 4.9 & 9.6 & 7.0 & 6.0 & 8.6 & 4.3 & 1.9 \end{bmatrix}$$
$$\text{weights} * \text{inputs} = \begin{bmatrix} 240.6 & 104.85 & 71.15 & -26.575 & 206.97 \\ 218.22 & 136.64 & 154.485 & 51.36 & 217.43 \\ 188.475 & 112.52 & 90.16 & 18.81 & 182.571 \\ 277.175 & 219.23 & 105.6 & 69.95 & 196.719 \\ 158.66 & 153.62 & 36.61 & -26.135 & 161.86 \\ 202.05 & 183.64 & 49.855 & -35.835 & 211.292 \end{bmatrix}$$

kemudian np.dot + biases

$$\begin{bmatrix} 240.6 & 104.85 & 71.15 & -26.575 & 206.97 \\ 218.22 & 136.64 & 154.485 & 51.36 & 217.43 \\ 188.475 & 112.52 & 90.16 & 18.81 & 182.571 \\ 277.175 & 219.23 & 105.6 & 69.95 & 196.719 \\ 158.66 & 153.62 & 36.61 & -26.135 & 161.86 \\ 202.05 & 183.64 & 49.855 & -35.835 & 211.292 \end{bmatrix} + [9.0, 0.3, 0.2, 1.5, 8.4]$$
$$= \begin{bmatrix} 249.6 & 105.15 & 71.35 & -25.075 & 215.37 \\ 227.22 & 136.94 & 154.685 & 52.86 & 223.83 \\ 197.475 & 112.82 & 90.36 & 20.31 & 190.971 \\ 286.175 & 219.53 & 105.8 & 71.45 & 205.119 \\ 167.66 & 153.92 & 36.81 & -24.635 & 170.26 \\ 211.05 & 183.94 & 50.055 & -34.335 & 219.692 \end{bmatrix}$$

UTS 2

a. Source Code:



```
#Azkia Kamila ; 21091397027 ; 2021A
#UTS2

#multiple perceptron/neuron batch and multiple layer

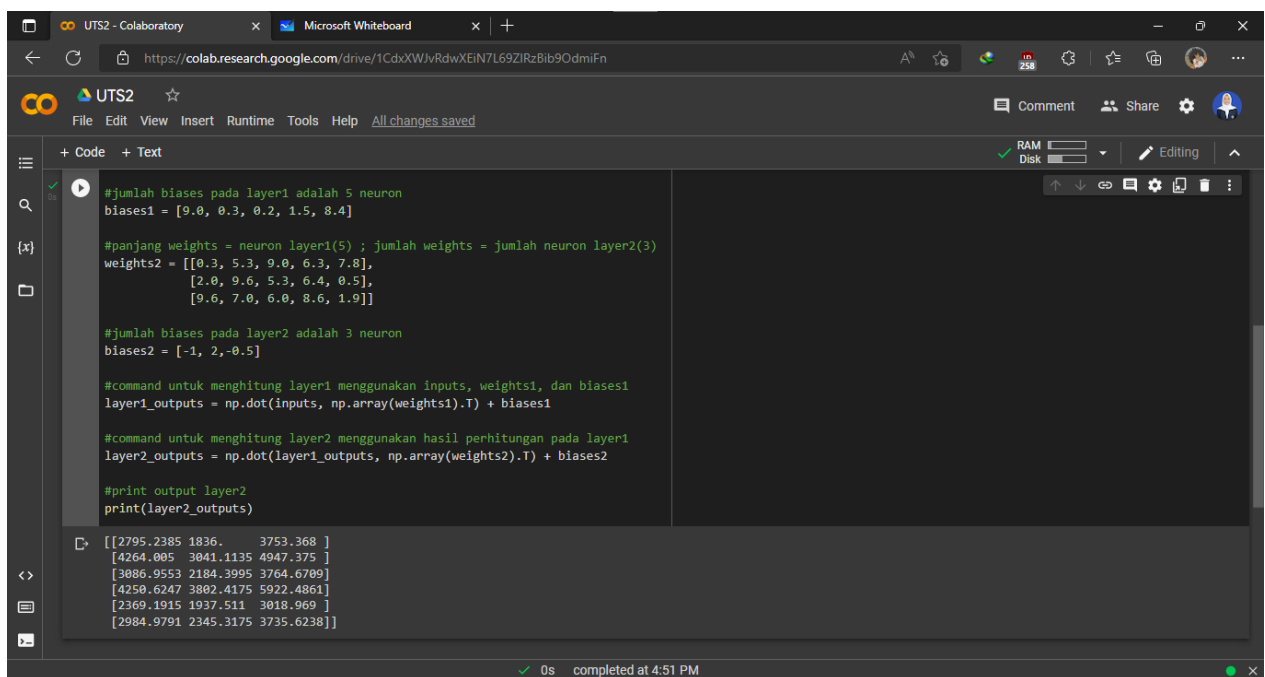
#inisialisasi numpy
import numpy as np

#inisialisasi variabel (input 10 ; batch 6 = matrix 6*10)
inputs = [[1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 1.5],
          [3.0, 5.0, 5.1, 4.6, 6.5, 7.0, 9.9, 1.7, 7.9, 3.0],
          [0.1, 0.3, 5.0, 6.1, 7.8, 5.1, 9.8, 4.3, 5.0, 3.2],
          [0.3, 5.3, 9.0, 6.3, 7.8, 5.5, 6.1, 3.7, 9.2, 6.8],
          [2.0, 9.6, 4.2, 2.9, 5.3, 6.4, 5.0, 6.9, 0.7, 0.5],
          [0.2, 5.1, 4.3, 4.9, 9.6, 7.0, 6.0, 8.6, 4.3, 1.9]]

#panjang weights = panjang inputs(10) ; jumlah weights = jumlah neuron(5)
weights1 = [[-0.25, 2.5, 5.3, 9.2, -6.1, 8.0, 1.5, 7.3, 9.9, 5.1],
            [-0.5, 5.6, 7.8, 3.2, 9.1, 6.4, -7.3, 0.5, 1.9, 2.7],
            [9.5, -0.3, 0.65, -7.3, 5.1, 2.0, 7.0, -6.0, 5.0, 4.0],
            [-0.2, 4.1, 5.0, 3.0, 1.0, -8.6, 2.8, -9.0, 4.1, 0.35],
            [-0.05, 1.0, 3.0, -8.1, 6.4, 9.0, 7.3, 4.5, 6.2, -0.32]]

#jumlah biases pada layer1 adalah 5 neuron
biases1 = [9.0, 0.3, 0.2, 1.5, 8.4]
```

0s completed at 4:51 PM



```
#jumlah biases pada layer1 adalah 5 neuron
biases1 = [9.0, 0.3, 0.2, 1.5, 8.4]

#panjang weights = neuron layer1(5) ; jumlah weights = jumlah neuron layer2(3)
weights2 = [[0.3, 5.3, 9.0, 6.3, 7.8],
            [2.0, 9.6, 5.3, 6.4, 0.5],
            [9.6, 7.0, 6.0, 8.6, 1.9]]

#jumlah biases pada layer2 adalah 3 neuron
biases2 = [-1, 2, -0.5]

#command untuk menghitung layer1 menggunakan inputs, weights1, dan biases1
layer1_outputs = np.dot(inputs, np.array(weights1).T) + biases1

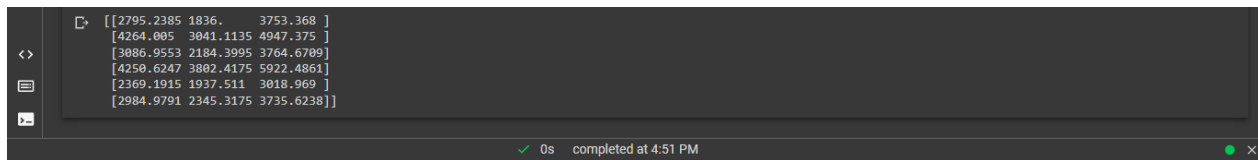
#command untuk menghitung layer2 menggunakan hasil perhitungan pada layer1
layer2_outputs = np.dot(layer1_outputs, np.array(weights2).T) + biases2

#print output layer2
print(layer2_outputs)
```

```
[[2795.2385 1836.        3753.368 ]
 [4264.005  3041.1135 4947.375 ]
 [3086.9553 2184.3995 3764.6709]
 [4250.6247 3802.4175 5922.4861]
 [2369.1915 1937.511  3018.960 ]
 [2984.9791 2345.3175 3735.6238]]
```

0s completed at 4:51 PM

Output:



```
[[2795.2385 1836.    3753.368 ]
 [4264.005  3041.1135 4947.375 ]
 [3086.9553 2184.3995 3764.6709]
 [4250.6247 3802.4175 5922.4861]
 [2369.1915 1937.511  3018.969 ]
 [2984.9791 2345.3175 3735.6238]]
```

0s completed at 4:51 PM

Output Result:

```
[[2795.2385 1836.    3753.368 ]
 [4264.005  3041.1135 4947.375 ]
 [3086.9553 2184.3995 3764.6709]
 [4250.6247 3802.4175 5922.4861]
 [2369.1915 1937.511  3018.969 ]
 [2984.9791 2345.3175 3735.6238]]
```

Analisis:

1. Inisialisasi numpy sebagai method perhitungan
2. Masukkan variabel untuk inputs, weights, dan bias sesuai dengan ketentuan
Inputs = 10
Batch = 6 } Inputs menjadi matric 6*10
Weights1 = 5*10
Biases1 = 5
Weights2 = 3*5
Biases2 = 3
3. Buat output untuk menghitung variabel yang telah kita masukkan/buat
np.dot = untuk penghitungan vektor weight dan input
kemudian hasil penghitungan vektor ditambahkan dengan biases
4. Untuk output yang diinginkan adalah output layer2 yang berasal dari hasil perhitungan layer1 lalu dihitung kembali dengan weights2 dan biases2
5. Buat command print untuk menampilkan hasil perhitungan output

Penghitungan Layer1:

perhitungan dot product

$$\begin{array}{c} \text{weights} \\ 10 \times 5 \end{array} \begin{bmatrix} -0.25 & 2.5 & 5.3 & 9.2 & -6.1 & 8.0 & 1.5 & 7.3 & 9.9 & 5.1 \\ -0.5 & 5.6 & 7.8 & 3.2 & 9.1 & 6.4 & -7.3 & 0.5 & 1.9 & 2.7 \\ 9.5 & 0.3 & 0.65 & -7.3 & 5.1 & 2.0 & 7.0 & -6.0 & 5.0 & 4.0 \\ -0.2 & 4.1 & 5.0 & 3.0 & 1.0 & -8.6 & 2.8 & -9.0 & 4.1 & 0.35 \\ -0.05 & 1.0 & 3.0 & -8.1 & 6.4 & 9.0 & 7.3 & 4.5 & 6.2 & -0.32 \end{bmatrix} * \begin{array}{c} \text{inputs} \\ 6 \times 10 \end{array} \begin{bmatrix} 1.0 & 2.0 & 3.0 & 4.0 & 5.0 & 6.0 & 7.0 & 8.0 & 9.0 & 1.5 \\ 3.0 & 5.0 & 5.1 & 4.6 & 6.5 & 7.0 & 9.9 & 1.7 & 7.9 & 3.0 \\ 0.1 & 0.3 & 5.0 & 6.1 & 7.8 & 5.1 & 9.8 & 4.3 & 5.0 & 3.2 \\ 0.3 & 5.3 & 9.0 & 6.3 & 7.8 & 5.5 & 6.1 & 3.7 & 9.2 & 6.8 \\ 2.0 & 9.6 & 4.2 & 2.9 & 5.3 & 6.4 & 5.0 & 6.9 & 0.7 & 0.5 \\ 0.2 & 5.1 & 4.3 & 4.9 & 9.6 & 7.0 & 6.0 & 8.6 & 4.3 & 1.9 \end{bmatrix}$$

$$\text{weights} * \text{inputs} = \begin{bmatrix} 240.6 & 104.85 & 71.15 & -26.575 & 206.97 \\ 218.22 & 136.64 & 154.485 & 51.36 & 217.43 \\ 188.475 & 112.52 & 90.16 & 18.81 & 182.571 \\ 277.175 & 219.23 & 105.6 & 69.95 & 196.719 \\ 158.66 & 153.62 & 36.61 & -26.135 & 161.86 \\ 202.05 & 183.64 & 49.855 & -35.835 & 211.292 \end{bmatrix}$$

kemudian np.dot + biases

$$\begin{bmatrix} 240.6 & 104.85 & 71.15 & -26.575 & 206.97 \\ 218.22 & 136.64 & 154.485 & 51.36 & 217.43 \\ 188.475 & 112.52 & 90.16 & 18.81 & 182.571 \\ 277.175 & 219.23 & 105.6 & 69.95 & 196.719 \\ 158.66 & 153.62 & 36.61 & -26.135 & 161.86 \\ 202.05 & 183.64 & 49.855 & -35.835 & 211.292 \end{bmatrix} + [9.0, 0.3, 0.2, 1.5, 8.4]$$

$$= \begin{bmatrix} 249.6 & 105.15 & 71.35 & -25.075 & 215.37 \\ 227.22 & 136.94 & 154.685 & 52.86 & 223.83 \\ 197.475 & 112.82 & 90.36 & 20.31 & 190.971 \\ 286.175 & 219.53 & 105.8 & 71.45 & 205.119 \\ 167.66 & 153.92 & 36.81 & -24.635 & 170.26 \\ 211.05 & 183.94 & 50.055 & -34.335 & 219.692 \end{bmatrix}$$

Penghitungan Layer2:

perhitungan dot product

$$\begin{array}{c} \text{weights2} \\ 5 \times 3 \end{array} \begin{bmatrix} 0.3 & 5.3 & 9.0 & 6.3 & 7.8 \\ 2.0 & 9.6 & 5.3 & 6.4 & 0.5 \\ 9.6 & 7.0 & 6.0 & 8.6 & 1.9 \end{bmatrix} * \begin{array}{c} \text{Output Layer1} \\ 5 \times 6 \end{array} \begin{bmatrix} 249.6 & 105.15 & 71.35 & -25.075 & 215.37 & \\ 227.22 & 136.94 & 154.685 & 52.86 & 223.83 & \\ 197.475 & 112.82 & 90.36 & 20.31 & 190.971 & \\ 286.175 & 219.53 & 105.8 & 71.45 & 205.119 & \\ 167.66 & 153.92 & 36.81 & -24.635 & 170.26 & \\ 211.05 & 183.94 & 50.055 & -34.335 & 219.692 & \end{bmatrix}$$

$$\text{weights2} * \text{Output Layer1} = \begin{bmatrix} 2796.2385 & 1834. & 3752.868 \\ 4265.005 & 3039.1135 & 4946.875 \\ 3087.9553 & 2182.3995 & 3764.1709 \\ 4251.6247 & 3800.4175 & 5921.9861 \\ 2370.1915 & 1935.511 & 3018.469 \\ 2985.9791 & 2343.3175 & 3735.1238 \end{bmatrix}$$

kemudian np.dot + biases2

$$\begin{bmatrix} 2796.2385 & 1834. & 3752.868 \\ 4265.005 & 3039.1135 & 4946.875 \\ 3087.9553 & 2182.3995 & 3764.1709 \\ 4251.6247 & 3800.4175 & 5921.9861 \\ 2370.1915 & 1935.511 & 3018.469 \\ 2985.9791 & 2343.3175 & 3735.1238 \end{bmatrix} + [-1, 2, -0.5]$$

$$= \begin{bmatrix} 2795.2385 & 1836. & 3753.368 \\ 4264.005 & 3041.1135 & 4947.375 \\ 3086.9553 & 2184.3995 & 3764.6709 \\ 4250.6247 & 3802.4175 & 5922.4861 \\ 2369.1915 & 1937.511 & 3018.969 \\ 2984.9791 & 2345.3175 & 3735.6238 \end{bmatrix} //$$