

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



Disusun oleh:

Azkie Kamila (21091397027)

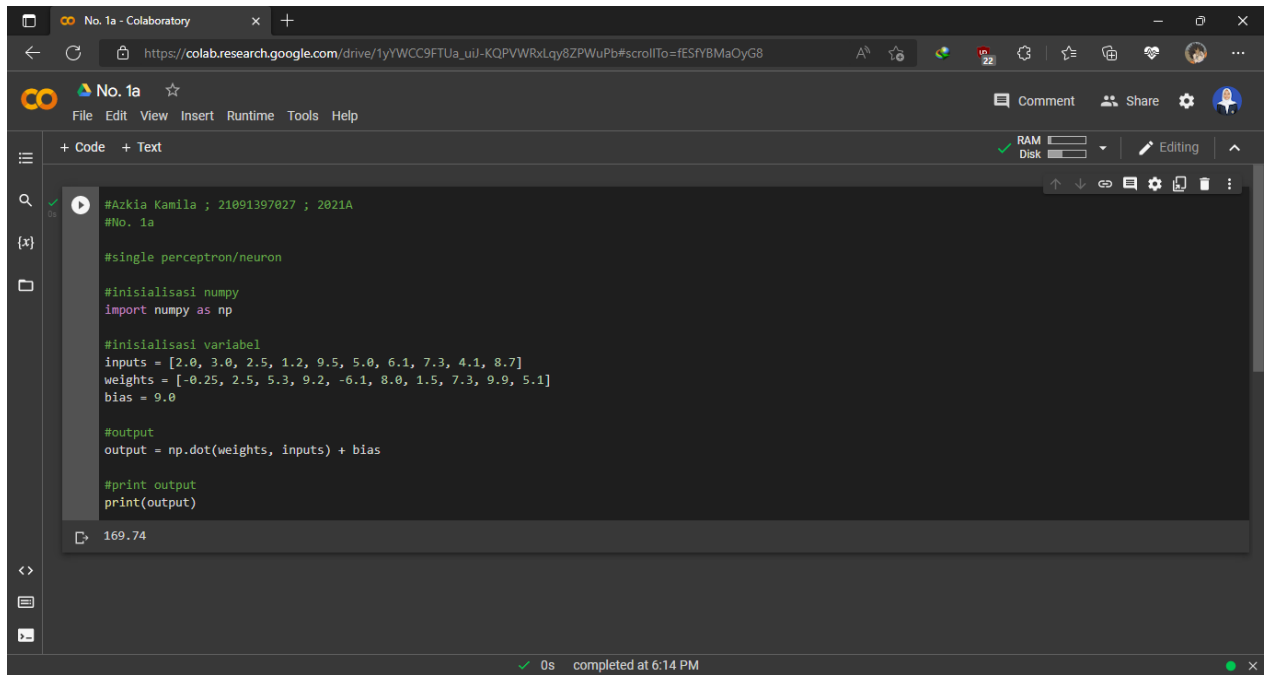
Program Studi D4 Manajemen Informatika

Fakultas Vokasi

Universitas Negeri Surabaya

2022

a. Source Code:



The screenshot shows a Google Colab notebook titled "No. 1a". The code is written in a dark-themed editor and implements a single perceptron/neuron. It includes comments in Indonesian and Python code using NumPy. The code defines inputs, weights, and a bias, then calculates the output using a dot product and adds the bias. The output is printed, showing the value 169.74. The status bar at the bottom indicates the code was completed at 6:14 PM.

```
#Azkia Kamila ; 21091397827 ; 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

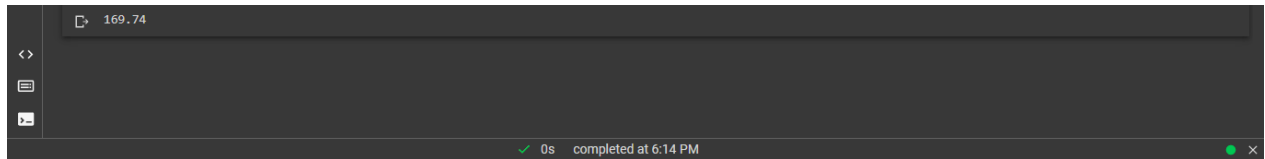
#output
output = np.dot(weights, inputs) + bias

#print output
print(output)
```

169.74

0s completed at 6:14 PM

Output:



The screenshot shows the output of the code execution in a dark-themed editor. The output is the value 169.74. The status bar at the bottom indicates the code was completed at 6:14 PM.

169.74

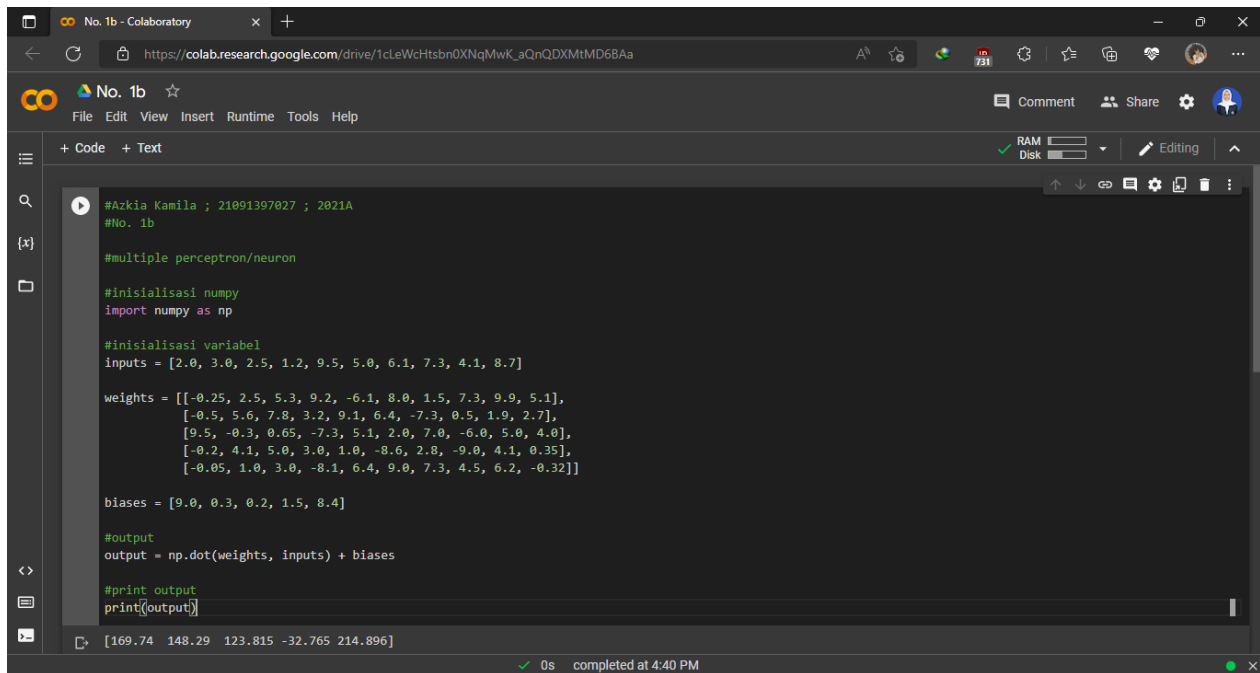
0s completed at 6:14 PM

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

b. Source 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]

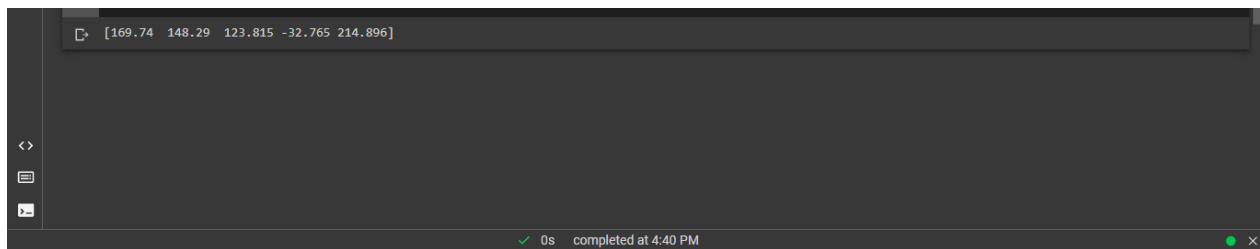
#output
output = np.dot(weights, inputs) + biases

#print output
print(output)
```

[169.74 148.29 123.815 -32.765 214.896]

0s completed at 4:40 PM

Output:



```
[169.74 148.29 123.815 -32.765 214.896]
```

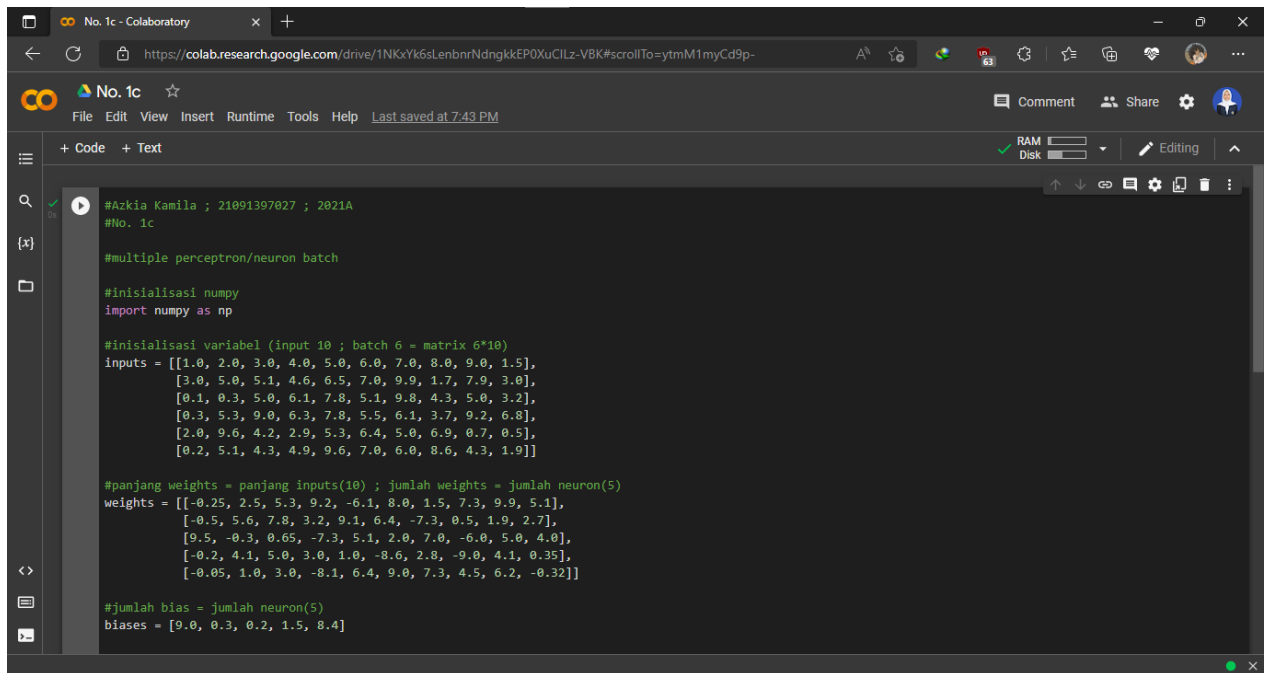
0s completed at 4:40 PM

Output Result: [169.75 148.29 123.815 -32.765 214.896]

Analysis:

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

c. Source Code:



The screenshot shows a Google Colab notebook titled "No. 1c". The code defines the initial parameters of a neural network, including inputs, weights, and biases.

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

#multiple perceptron/neuron batch

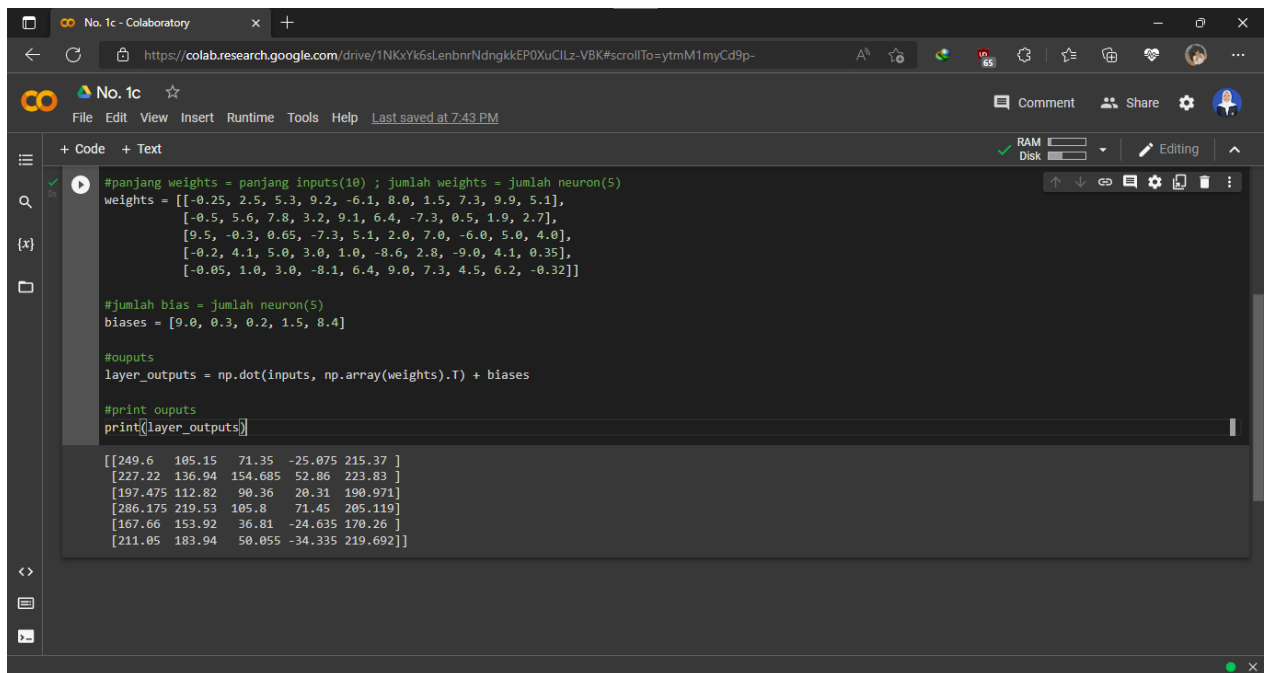
#inisialisasi numpy
import numpy as np

#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.

```
#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)]
```

The output of the code is displayed as follows:

```
[[249.6 105.15 71.35 -25.075 215.37 ]
 [227.22 136.94 154.685 52.06 223.03 ]
 [197.475 112.82 90.36 20.31 190.071]
 [286.175 219.53 105.0 71.45 205.119]
 [167.66 153.92 36.01 -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