

LAPORAN
UTS KECERDASAN BUATAN PART II



Disusun oleh :

Ananda Putri Rahmadani 21091397046

UNIVERSITAS NEGERI SURABAYA
MANAJEMEN INFORMATIKA

2022

1. Buat kodingan
 - 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

- **Kodingan**

```
File Edit Selection View Go Run Terminal Help uts part 2_046_Ananda.py - kecerdasan buatan - Visual Studio Code
Get Started uts part 2_046_Ananda.py X Nomer 1C_UTC_046_Ananda Putri R.py
C:\Users\user> Downloads > uts part 2_046_Ananda.py > ...
1 #Ananda Putri Rahmadani_21091397046
2 #Multiple perceptron / Neuron batch and multiple layer 2
3
4 #inisialisasi numpy
5 import numpy as np
6
7 # inisialisasi variabel
8 # memasukkan nilai variabel layer feature 10 dengan batch sejumlah 6
9 inputs = [
10     [2.0, 3.7, 8.0, 2.7, 0.0, 31.5, 5.9, 3.5, 5.0, 5.5],
11     [0.3, 0.1, 2.2, 2.6, 3.2, 3.4, 0.2, 2.4, 9.2, 7.4],
12     [1.4, 9.5, 18.0, 20.5, 32.1, 60.12, 33.7, 67.1, 76.0, 50.5],
13     [6.0, 3.4, 2.6, 7.8, 3.6, 3.8, 4.6, 4.8, 5.6, 5.8],
14     [1.4, 0.3, 7.2, 5.0, 8.2, 6.1, 9.2, 9.4, 27.3, 0.4],
15     [13.2, 17.3, 14.5, 10.5, 38.1, 12.6, 11.7, 3.23, 59.2, 82.4]]
16
17 # memberikan nilai bobot pada variabel sesuai dengan jumlah input
18 # memasukkan jumlah weight sesuai dengan jumlah neuron yaitu sejumlah 5
19 weights1 = [
20     [6.0, 4.8, 8.4, 2.5, 0.1, 3.5, 9.7, 4.5, 6.2, 15.5],
21     [7.4, 9.7, 4.10, 2.84, 3.52, 38.4, 45.2, 4.4, 5.2, 5.4],
22     [3.3, 6.1, 2.3, 10.9, 31.6, 3.82, 4.26, 4.8, 56.6, 55.8],
23     [5.8, 4.3, 4.2, 7.8, 0.2, 7.4, 3.5, 0.7, 40.3, 71.1],
24     [5.1, 13.7, 30.6, 42.7, 95.1, 12.3, 29.0, 40.7, 28.1, 93.11]]
25
26 # inisialisasi biases pada layer1 sesuai dengan neuron yang ditentukan yaitu layer 1 = 5 neuron
27 biases1 = [4.7, 2.8, 1.0, 9.6, 3.1]
28
29 # inisialisasi jumlah weight 2, weight layer 2 = neuron layer 1 yaitu 5
30 # memasukkan jumlah weight sesuai dengan neuron layer 2 yaitu 3 neuron
31 weights2 = [
32     [10.3, 4.4, 2.9, 3.2, 11.2],
33     [5.0, 1.3, 4.2, 7.5, 9.9],
34     [0.1, 6.6, 3.0, 0.0, 3.7]]
35
36 # inisialisasi biases pada layer2 dengan neuron yang ditentukan yaitu 3
37 biases2 = [8.2, 4.2, 5.6]
38
39
40 # output
41 # menghitung layer1 dengan (inputs*weight1) dan biases1
42 layer1_outputs = np.dot(inputs, np.array(weights1).T) + biases1
43
44 # menghitung layer2 dengan hasil perhitungan pada layer1
45 layer2_outputs = np.dot(layer1_outputs, np.array(weights2).T) + biases2
46
47 #print output layer2
48 print(layer2_outputs)
```

- Penjelasan

<pre> 4 #inisialisasi numpy 5 import numpy as np </pre>	<p>Line ke 4 terdapat tanda # ini menunjukan komentar</p> <p>Line ke5 menginisialisasi numpy ke np untuk mempermudah dalam mengoprasionalkan kodingan</p>
<pre> 8 # memasukan nilai variabel layer feature 10 dengan batch sejumlah 6 9 inputs = [10 [2.0, 3.7, 8.0, 2.7, 0.0, 31.5, 5.9, 3.5, 5.0, 5.5], 11 [0.3, 0.1, 2.2, 2.6, 3.2, 3.4, 0.2, 2.4, 9.2, 7.4], 12 [1.4, 9.5, 18.0, 20.5, 32.1, 60.12, 33.7, 67.1, 76.0, 50.5], 13 [6.0, 3.4, 2.6, 7.8, 3.6, 3.8, 4.6, 4.8, 5.6, 5.8], 14 [1.4, 0.3, 7.2, 5.0, 8.2, 6.1, 9.2, 9.4, 27.3, 0.4], 15 [13.2, 17.3, 14.5, 10.5, 38.1, 12.6, 11.7, 3.23, 59.2, 82.4]] </pre>	<p>Pada line ke 9 variabel diinisialisasikan , lalu memasukan nilai input dengan jumlah 10 baris angka sesuai dengan yang ditentukan yakni feature layer 10 dan 6 kolom angka sesuai yang diminta yaitu batch = 6</p>
<pre> 18 # memasukan jumlah weight sesuai dengan jumlah neuron yaitu sejumlah 5 19 weights1 = [20 [6.0, 4.8, 8.4, 2.5, 0.1, 3.5, 9.7, 4.5, 6.2, 15.5], 21 [7.4, 9.7, 4.10, 2.84, 3.52, 38.4, 45.2, 4.4, 5.2, 5.4], 22 [3.3, 6.1, 2.3, 10.9, 31.6, 3.82, 4.26, 4.8, 56.6, 55.8], 23 [5.8, 4.3, 4.2, 7.8, 0.2, 7.4, 3.5, 0.7, 40.3, 71.1], 24 [5.1, 13.7, 30.6, 42.7, 95.1, 12.3, 29.0, 40.7, 28.1, 93.11]] 25 </pre>	<p>Line ke 19 memasukan nilai weight dengan 5 kolom , hal ini sesuai dengan neuron yang diperintahkan yaitu 5 dan 10 baris angka</p>
<pre> 30 # memasukkan jumlah weight sesuai dengan neuron layer 2 yaitu 3 neuron 31 weights2 = [32 [10.3, 4.4, 2.9, 3.2, 11.2], 33 [5.0, 1.3, 4.2, 7.5, 9.9], 34 [0.1, 6.6, 3.0, 0.0, 3.7]] 35 </pre>	<p>Line 31 memasukan nilai weight2 dengan jumlah weight sama dengan neuron yang ditentukan yakni 5</p>
<pre> 42 layer1_outputs = np.dot(inputs, np.array(weights1).T) + biases1 </pre>	<p>Pada line 42 terdapat rumus untuk menghitung layer 1 dengan weight lalu mentranspose lebih dulu</p>
<pre> 45 layer2_outputs = np.dot(layer1_outputs, np.array(weights2).T) + biases2 </pre>	<p>Line 45 terdapat rumus untuk menghitung dari hasil layer 1 dikalikan dengan weight transpose lalu dijumlahkan dengan biases 2</p>
<pre> 48 print(layer2_outputs) </pre>	<p>Line 48 ini berfungsi untuk menampilkan hasil output dari kodingan</p>

- Output

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER

Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS D:\Ananda (semester 3)\kecerdasan buatan> & C:/Users/user/AppData/Local/Programs/Python/Python310/p
.py"
[[ 36758.5658  32317.0597  19947.1463 ]
 [ 27542.382   29008.873   10961.0326 ]
 [275823.28296 271126.19558 120520.3529 ]
 [ 32320.308   31232.2158   13631.249  ]
 [ 49247.283   48803.0156   22385.3772 ]
 [253128.0635  265354.8398   98603.9884 ]]
PS D:\Ananda (semester 3)\kecerdasan buatan>

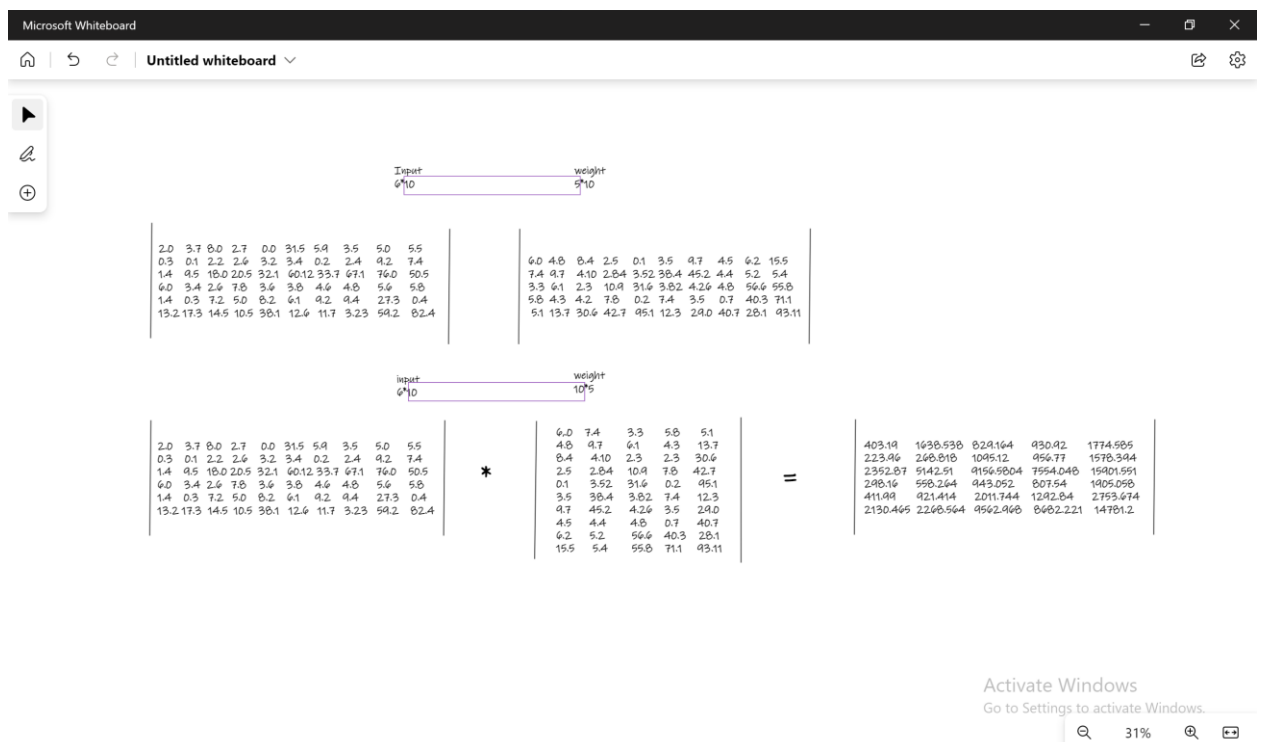
```

- Perhitungan Output

```

41 # menghitung layer1 dengan (inputs*weight1) dan biases1
42 layer1_outputs = np.dot(inputs, np.array(weights1).T) + biases1

```



Ketika layer 1 sudah ditemukan dengan melakukan perkalian input dan transpose weight1 hasil np.dot ditambah dengan biases 1

<table border="1"> <tr><td>403.19</td><td>1638.538</td><td>829.164</td><td>930.92</td><td>1774.585</td></tr> <tr><td>223.96</td><td>268.818</td><td>1095.12</td><td>956.77</td><td>1578.394</td></tr> <tr><td>2352.87</td><td>5142.51</td><td>9156.5804</td><td>7554.048</td><td>15901.551</td></tr> <tr><td>298.16</td><td>558.264</td><td>943.052</td><td>807.54</td><td>1905.058</td></tr> <tr><td>411.99</td><td>921.414</td><td>2011.744</td><td>1292.84</td><td>2753.674</td></tr> <tr><td>2130.465</td><td>2268.564</td><td>9562.968</td><td>8682.221</td><td>14781.2</td></tr> </table>	403.19	1638.538	829.164	930.92	1774.585	223.96	268.818	1095.12	956.77	1578.394	2352.87	5142.51	9156.5804	7554.048	15901.551	298.16	558.264	943.052	807.54	1905.058	411.99	921.414	2011.744	1292.84	2753.674	2130.465	2268.564	9562.968	8682.221	14781.2	+	<table border="1"> <tr><td>4.7</td><td>2.8</td><td>1.0</td><td>9.6</td><td>3.1</td></tr> </table>	4.7	2.8	1.0	9.6	3.1	<table border="1"> <tr><td>407.89</td><td>1641.338</td><td>830.164</td><td>940.52</td><td>1777.685</td></tr> <tr><td>228.66</td><td>271.618</td><td>1096.12</td><td>966.37</td><td>1581.494</td></tr> <tr><td>2357.57</td><td>5145.31</td><td>9157.5804</td><td>7563.648</td><td>15904.651</td></tr> <tr><td>302.86</td><td>561.064</td><td>944.052</td><td>817.14</td><td>1908.158</td></tr> <tr><td>416.69</td><td>924.214</td><td>2012.744</td><td>1302.44</td><td>2756.774</td></tr> <tr><td>2135.165</td><td>2271.364</td><td>9563.968</td><td>8691.821</td><td>14784.315</td></tr> </table>	407.89	1641.338	830.164	940.52	1777.685	228.66	271.618	1096.12	966.37	1581.494	2357.57	5145.31	9157.5804	7563.648	15904.651	302.86	561.064	944.052	817.14	1908.158	416.69	924.214	2012.744	1302.44	2756.774	2135.165	2271.364	9563.968	8691.821	14784.315
403.19	1638.538	829.164	930.92	1774.585																																																																
223.96	268.818	1095.12	956.77	1578.394																																																																
2352.87	5142.51	9156.5804	7554.048	15901.551																																																																
298.16	558.264	943.052	807.54	1905.058																																																																
411.99	921.414	2011.744	1292.84	2753.674																																																																
2130.465	2268.564	9562.968	8682.221	14781.2																																																																
4.7	2.8	1.0	9.6	3.1																																																																
407.89	1641.338	830.164	940.52	1777.685																																																																
228.66	271.618	1096.12	966.37	1581.494																																																																
2357.57	5145.31	9157.5804	7563.648	15904.651																																																																
302.86	561.064	944.052	817.14	1908.158																																																																
416.69	924.214	2012.744	1302.44	2756.774																																																																
2135.165	2271.364	9563.968	8691.821	14784.315																																																																

Menghitung hidden layer 2

```

43
44 # menghitung layer2 dengan hasil perhitungan pada layer1
45 layer2_outputs = np.dot(layer1_outputs, np.array(weights2).T) + biases2
46

```

Saat hasil dari layer1 didapatkan lalu lanjut masuk ke penghitungan layer 2, mencari nilai np.dot layer 2 dengan cara hasil layer 1 dikali dengan weight transpose 2. Weights 2 di transpose untuk menyamakan ordo output hidden layer 1

<p>output layer 1</p> <p>6 * 5</p> <table border="1"> <tr><td>407.89</td><td>1641.338</td><td>830.164</td><td>940.52</td><td>1777.685</td></tr> <tr><td>228.66</td><td>271.618</td><td>1096.12</td><td>966.37</td><td>1581.494</td></tr> <tr><td>2357.57</td><td>5145.31</td><td>9157.5804</td><td>7563.648</td><td>15904.651</td></tr> <tr><td>302.86</td><td>561.064</td><td>944.052</td><td>817.14</td><td>1908.158</td></tr> <tr><td>416.69</td><td>924.214</td><td>2012.744</td><td>1302.44</td><td>2756.774</td></tr> <tr><td>2135.165</td><td>2271.364</td><td>9563.968</td><td>8691.821</td><td>14784.315</td></tr> </table>	407.89	1641.338	830.164	940.52	1777.685	228.66	271.618	1096.12	966.37	1581.494	2357.57	5145.31	9157.5804	7563.648	15904.651	302.86	561.064	944.052	817.14	1908.158	416.69	924.214	2012.744	1302.44	2756.774	2135.165	2271.364	9563.968	8691.821	14784.315		<p>weights</p> <p>3 * 5</p> <table border="1"> <tr><td>10.3</td><td>4.4</td><td>2.9</td><td>3.2</td><td>11.2</td></tr> <tr><td>5.0</td><td>1.3</td><td>4.2</td><td>7.5</td><td>9.9</td></tr> <tr><td>0.1</td><td>6.6</td><td>3.0</td><td>0.0</td><td>3.7</td></tr> </table>	10.3	4.4	2.9	3.2	11.2	5.0	1.3	4.2	7.5	9.9	0.1	6.6	3.0	0.0	3.7																				
407.89	1641.338	830.164	940.52	1777.685																																																															
228.66	271.618	1096.12	966.37	1581.494																																																															
2357.57	5145.31	9157.5804	7563.648	15904.651																																																															
302.86	561.064	944.052	817.14	1908.158																																																															
416.69	924.214	2012.744	1302.44	2756.774																																																															
2135.165	2271.364	9563.968	8691.821	14784.315																																																															
10.3	4.4	2.9	3.2	11.2																																																															
5.0	1.3	4.2	7.5	9.9																																																															
0.1	6.6	3.0	0.0	3.7																																																															
<p>output layer 1</p> <p>6 * 5</p> <table border="1"> <tr><td>407.89</td><td>1641.338</td><td>830.164</td><td>940.52</td><td>1777.685</td></tr> <tr><td>228.66</td><td>271.618</td><td>1096.12</td><td>966.37</td><td>1581.494</td></tr> <tr><td>2357.57</td><td>5145.31</td><td>9157.5804</td><td>7563.648</td><td>15904.651</td></tr> <tr><td>302.86</td><td>561.064</td><td>944.052</td><td>817.14</td><td>1908.158</td></tr> <tr><td>416.69</td><td>924.214</td><td>2012.744</td><td>1302.44</td><td>2756.774</td></tr> <tr><td>2135.165</td><td>2271.364</td><td>9563.968</td><td>8691.821</td><td>14784.315</td></tr> </table>	407.89	1641.338	830.164	940.52	1777.685	228.66	271.618	1096.12	966.37	1581.494	2357.57	5145.31	9157.5804	7563.648	15904.651	302.86	561.064	944.052	817.14	1908.158	416.69	924.214	2012.744	1302.44	2756.774	2135.165	2271.364	9563.968	8691.821	14784.315	*	<p>weight 2 transpose</p> <p>5 * 3</p> <table border="1"> <tr><td>10.3</td><td>5.0</td><td>0.1</td></tr> <tr><td>4.4</td><td>1.3</td><td>6.6</td></tr> <tr><td>2.9</td><td>4.2</td><td>3.0</td></tr> <tr><td>3.2</td><td>7.5</td><td>0.0</td></tr> <tr><td>11.2</td><td>9.9</td><td>3.7</td></tr> </table>	10.3	5.0	0.1	4.4	1.3	6.6	2.9	4.2	3.0	3.2	7.5	0.0	11.2	9.9	3.7	=	<table border="1"> <tr><td>3670.3658</td><td>32312.8597</td><td>19941.5463</td></tr> <tr><td>27534.182</td><td>29004.673</td><td>10955.4326</td></tr> <tr><td>275815.083</td><td>271121.996</td><td>120154.752</td></tr> <tr><td>32312.108</td><td>31228.0158</td><td>13625.649</td></tr> <tr><td>49239.083</td><td>48798.8156</td><td>22379.7772</td></tr> <tr><td>253119.863</td><td>265350.64</td><td>98598.3884</td></tr> </table>	3670.3658	32312.8597	19941.5463	27534.182	29004.673	10955.4326	275815.083	271121.996	120154.752	32312.108	31228.0158	13625.649	49239.083	48798.8156	22379.7772	253119.863	265350.64	98598.3884
407.89	1641.338	830.164	940.52	1777.685																																																															
228.66	271.618	1096.12	966.37	1581.494																																																															
2357.57	5145.31	9157.5804	7563.648	15904.651																																																															
302.86	561.064	944.052	817.14	1908.158																																																															
416.69	924.214	2012.744	1302.44	2756.774																																																															
2135.165	2271.364	9563.968	8691.821	14784.315																																																															
10.3	5.0	0.1																																																																	
4.4	1.3	6.6																																																																	
2.9	4.2	3.0																																																																	
3.2	7.5	0.0																																																																	
11.2	9.9	3.7																																																																	
3670.3658	32312.8597	19941.5463																																																																	
27534.182	29004.673	10955.4326																																																																	
275815.083	271121.996	120154.752																																																																	
32312.108	31228.0158	13625.649																																																																	
49239.083	48798.8156	22379.7772																																																																	
253119.863	265350.64	98598.3884																																																																	

Setelah didapatkan hasil np.dot layer 2 lalu ditambah dengan biases 2 dan hasil layer 2 didapatkan

<p>np.dot layer</p> <table border="1"> <tr><td>3670.3658</td><td>32312.8597</td><td>19941.5463</td></tr> <tr><td>27534.182</td><td>29004.673</td><td>10955.4326</td></tr> <tr><td>275815.083</td><td>271121.996</td><td>120154.752</td></tr> <tr><td>32312.108</td><td>31228.0158</td><td>13625.649</td></tr> <tr><td>49239.083</td><td>48798.8156</td><td>22379.7772</td></tr> <tr><td>253119.863</td><td>265350.64</td><td>98598.3884</td></tr> </table>	3670.3658	32312.8597	19941.5463	27534.182	29004.673	10955.4326	275815.083	271121.996	120154.752	32312.108	31228.0158	13625.649	49239.083	48798.8156	22379.7772	253119.863	265350.64	98598.3884	+	<p>biases 2</p> <table border="1"> <tr><td>8.2</td><td>4.2</td><td>5.6</td></tr> </table>	8.2	4.2	5.6	=	<table border="1"> <tr><td>36758.5658</td><td>32317.0597</td><td>19947.1463</td></tr> <tr><td>27542.382</td><td>29008.873</td><td>10961.0326</td></tr> <tr><td>275823.28296</td><td>271126.1958</td><td>120520.3529</td></tr> <tr><td>32320.308</td><td>31232.2158</td><td>13631.249</td></tr> <tr><td>49247.283</td><td>48803.0156</td><td>22385.3772</td></tr> <tr><td>253128.0635</td><td>265354.8398</td><td>98603.9884</td></tr> </table>	36758.5658	32317.0597	19947.1463	27542.382	29008.873	10961.0326	275823.28296	271126.1958	120520.3529	32320.308	31232.2158	13631.249	49247.283	48803.0156	22385.3772	253128.0635	265354.8398	98603.9884
3670.3658	32312.8597	19941.5463																																									
27534.182	29004.673	10955.4326																																									
275815.083	271121.996	120154.752																																									
32312.108	31228.0158	13625.649																																									
49239.083	48798.8156	22379.7772																																									
253119.863	265350.64	98598.3884																																									
8.2	4.2	5.6																																									
36758.5658	32317.0597	19947.1463																																									
27542.382	29008.873	10961.0326																																									
275823.28296	271126.1958	120520.3529																																									
32320.308	31232.2158	13631.249																																									
49247.283	48803.0156	22385.3772																																									
253128.0635	265354.8398	98603.9884																																									