TUGAS BESAR A IF3270 PEMBELAJARAN MESIN SEMESTER 2 2021/2022

Anggota:

Kahfi Soobhan Zulkifli	13519012
Jose Galbraith Hasintongan	13519022
Muhammad Fahkry Malta	13519032
Muhammad Akram Al Bari	13519142

Tanggal Pengumpulan: 5 Maret 2022

TEKNIK INFORMATIKA
SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA
INSTITUT TEKNOLOGI BANDUNG

2022

IMPLEMENTASI

Dalam melakukan implementasi forward propagation untuk FFNN (Feed Forward Neural Network), kami mendesain model FFNN yang direpresentasikan sebagai object dalam bahasa python. Terdapat tiga buah objek yang kami definisikan yakni: Model, Layer, dan Neuron. Pada objek Model, terdapat atribut yang menyimpan jumlah nilai dari total neuron input layer. Kemudian disimpan pula sebuah array of Layer, yang digunakan untuk menyimpan setiap hidden Layer (termasuk output layer) dari Model FFNN yang dibuat.

```
class Model:
    def __init__(self,size_of_input_layer):
        self.layers = []
        self.size_of_input_layer = size_of_input_layer
```

Pada objek Model terdapat fungsi Add dan juga Predict. Fungsi Add digunakan untuk menambahkan layer baru pada Model dengan cara memasukkan array of weights untuk setiap neuron pada layer tersebut, dan juga activation function yang ingin digunakan. Fungsi Predict digunakan untuk memberikan input, baik single maupun batch, untuk diprediksi oleh model. Kemudian, untuk memfasilitasi pembacaan secara batch, fungsi predict ini juga akan mereset isi array hasil agar tidak terbawa ke iterasi selanjutnya.

```
def predict(self, input_to_process):
    results = []
    for batch in range(len(input_to_process)):

        input = input_to_process[batch]
        input.insert(0,1)

        for layer in range(len(self.layers)):
            self.layers[layer].layer_output = []

        self.layers[0].calculate(input)
        input_layer_output = self.layers[0].layer_output
        for i in range(1, len(self.layers)):
            self.layers[i].calculate(input_layer_output)
            input_layer_output = self.layers[i].layer_output

        if self.layers[-1].activation_function != "softmax":
            results.append(input_layer_output[1]) #y_hat
        else:
            results.append(input_layer_output.index(max(input_layer_output[1:])))
    return results
```

Kemudian objek Layer digunakan untuk merepresentasikan hidden dan juga output layer dari FFNN. Pada layer terdapat array of Neuron yang digunakan untuk menyimpan setiap Neuron yang ada

pada layer yang bersangkutan. Kemudian disimpan juga nilai dari activation function yang digunakan untuk satu layer tersebut. Activation function disimpan pada object layer karena setiap Neuron pada Layer yang sama memiliki activation function yang sama. Kemudian, terdapat juga sebuah atribut yang digunakan untuk menyimpan semua hasil kalkulasi dari setiap Neuron pada Layer terkait.

```
42
43  class Layer:
44      def __init__(self, array_of_weights, activation_function):
45          self.neurons = []
46          self.layer_output = []
47          self.total_softmax_exponents = 0
48          self.activation_function = activation_function
49          for weights in array_of_weights:
50              self.neurons.append(Neuron(activation_function, weights))
51
```

Terakhir adalah objek Neuron, yang digunakan untuk merepresentasikan setiap Neuron yang ada pada suatu layer. Pada objek neuron ini tetap disimpan activation function sebagai sebuah placeholder karena barangkali di kemudian hari akan ada case di mana pada layer yang sama, terdapat neuron dengan activation function yang berbeda. Pada objek ini juga terdapat array of weights yang digunakan untuk menyimpan nilai setiap weight yang ada pada Neuron terkait.

```
class Neuron:
    def __init__(self, activation_function, weights):
        self.activation_function = activation_function
        self.weights = weights
```

Untuk memberikan summary terhadap model yang telah dibuat, kami membuat juga fungsi summary yang menerima input sebuah Model. Fungsi ini nantinya akan memberikan data summary

terhadap model yang diberikan sebagai argumen.

```
def summary(model):
    paramList = []
    param = model.size_of_input_layer
    for layer in range(len(model.layers)):
        param = (param + 1) * len(model.layers[layer].neurons)
        print("Coefficient:")
        for neuron in model.layers[layer].neurons:
            print("w{}{} : {}".format(layer,model.layers[layer].neurons.index(
        print("")
                print("Output Shape:")
                print(len(model.layers[layer].neurons))
                print("")
                print("Activation Function:")
                print(neuron.activation_function)
                print("")
                print("Param")
                print(param)
                paramList.append(param)
                param = len(model.layers[layer].neurons)
                print("=================================================")
            print("Total params: {}".format(sum(paramList)))
            print("Trainable params: {}".format(sum(paramList)))
```

Terakhir, untuk membaca file definisi model dari sebuah file txt, kami mendefinisikan fungsi parse_model_from_file yang menerima input sebuah nama file txt. Untuk format input model, kami mendefinisikannya berupa line pertama menandakan jumlah layer (hidden termasuk juga output) dari model yang kami bangun. Kemudian line kedua adalah jumlah neuron untuk input layer. Dan line-line berikutnya adalah definisi dari setiap layer yang ada.

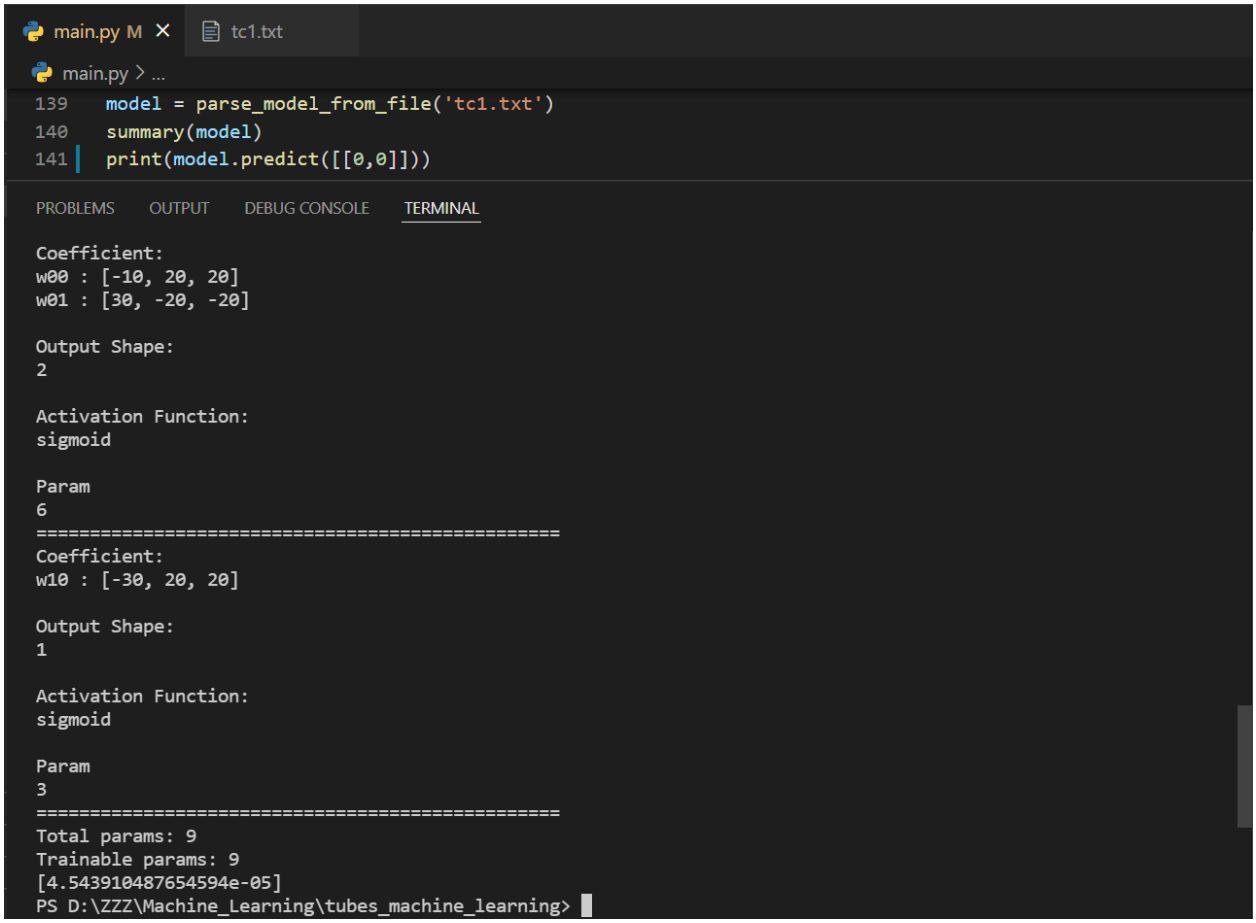
```python
def parse_model_from_file(file_name):
    model = []
    with open(file_name, encoding = 'utf-8') as f:
        hidden_layer_num = int(f.readline().rstrip("\n"))
        input_layer_neuron_num = int(f.readline().rstrip("\n"))

        model = Model(input_layer_neuron_num)
        for i in range(hidden_layer_num):
            hidden_layer_attributes = f.readline().rstrip("\n").split(",")
            hidden_layer_neuron_count = int(hidden_layer_attributes[0])
            hidden_layer_act_func = hidden_layer_attributes[1]

            array_of_weights = []
            for neuron in range(2, 2 + hidden_layer_neuron_count):
                weights = (list(map(int, hidden_layer_attributes[neuron].split(";"))))
                array_of_weights.append(weights)

            model.add(array_of_weights, hidden_layer_act_func)
    return model
```

HASIL PENGUJIAN DAN PERBANDINGAN

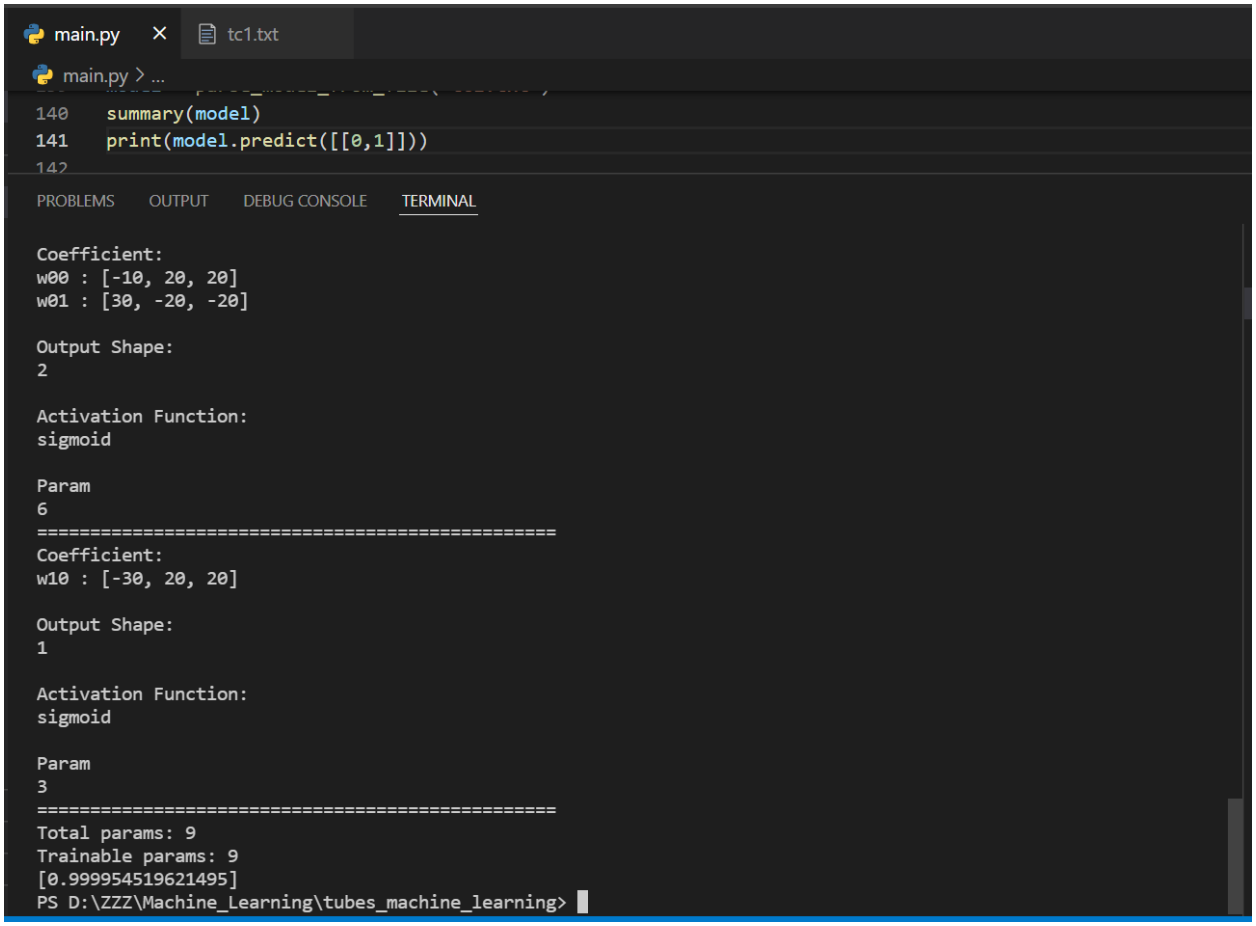
Sigmoid dengan perhitungan manual

X0	X1	X2	f	Σh1	Σh2	h1	h2	Σy	y
1	0	0	0	-10	30	0	1	-10	0
1	0	1	1	10	10	1	1	10	1
1	1	0	1	10	10	1	1	10	1
1	1	1	0	30	-10	1	0	-10	0

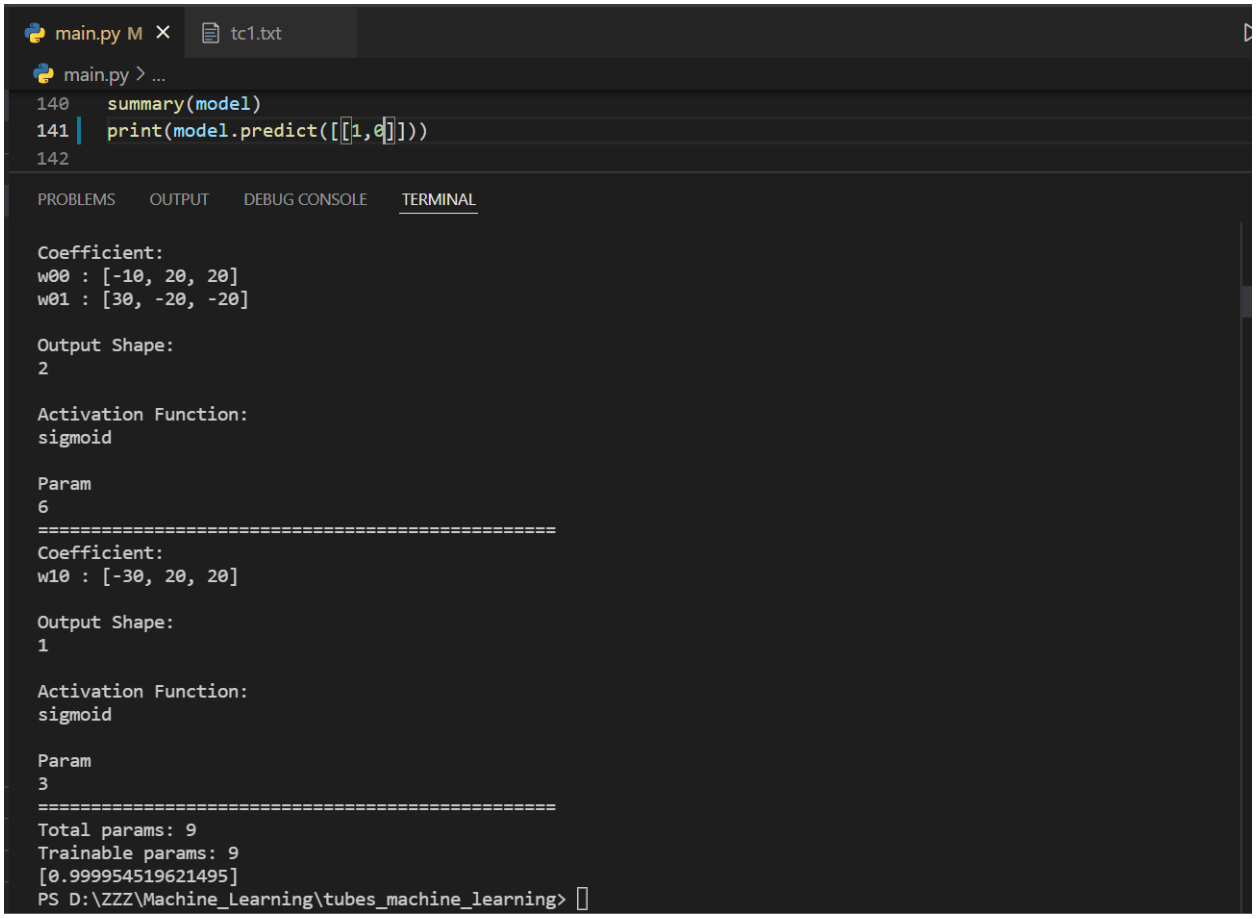
Sigmoid dengan perhitungan program



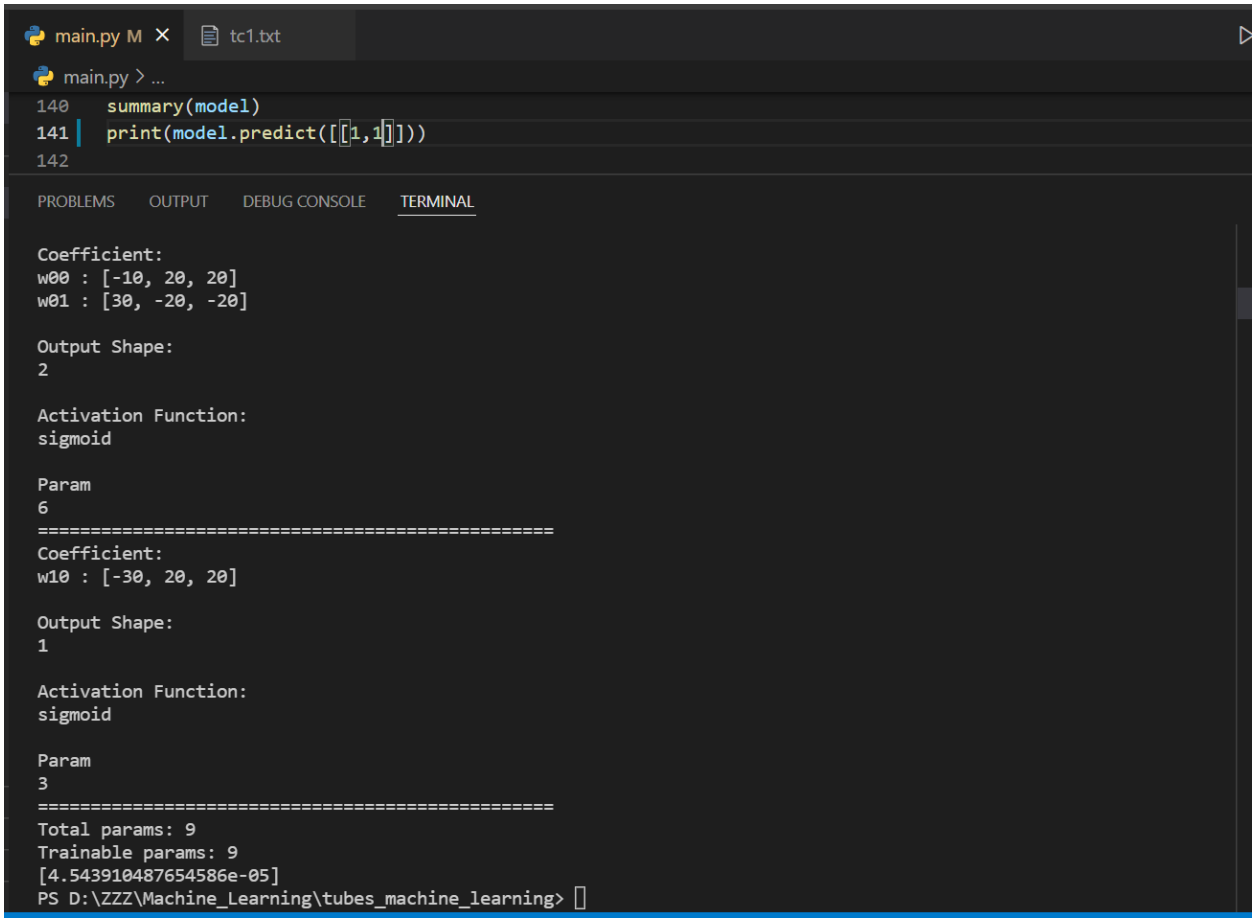
Tes Case Sigmoid dengan input (0,0)



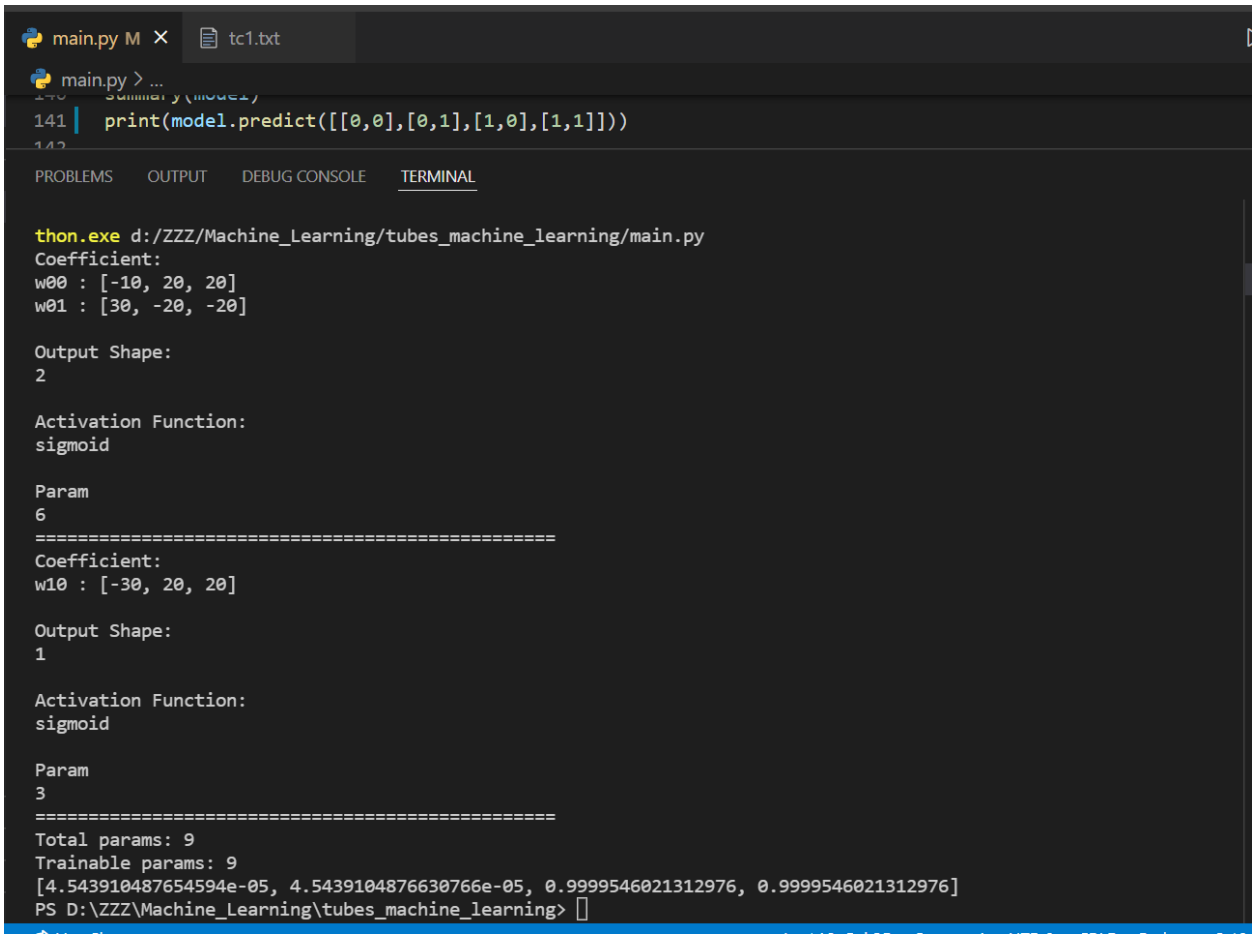
Tes Case Sigmoid dengan input (0,1)



Tes Case Sigmoid dengan input (1,0)



Tes Case Sigmoid dengan input (1,1)



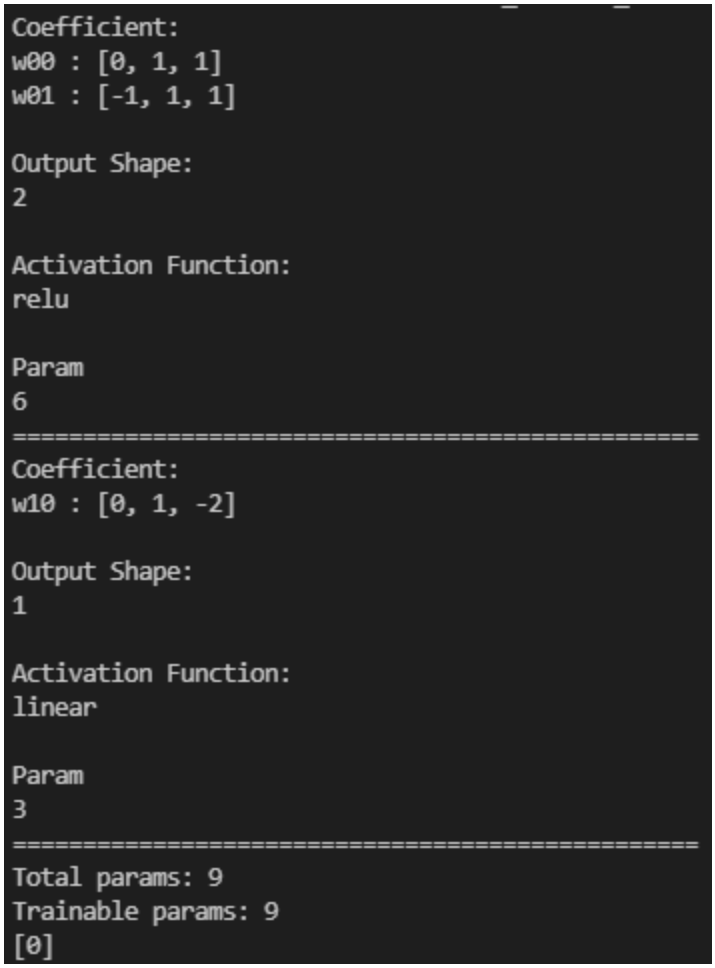
Tes Case Sigmoid dengan input ((0,0),(0,1),(1,0),(1,1)) atau batch

ReLU dengan perhitungan manual

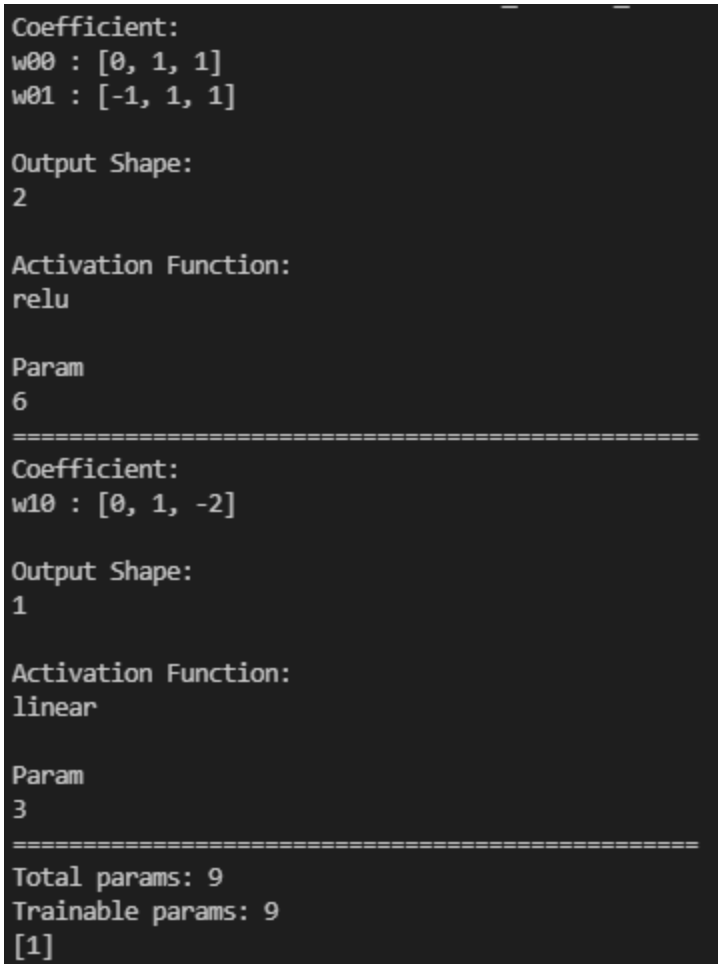
X0	X1	X2	Σh1	Σh2	h1	h2	Σy	y
1	0	0	0	-1	0.00	0.00	0.00	0
1	0	1	1	0	1.00	0.00	1.00	1.00
1	1	0	1	0	1.00	0.00	1.00	1.00
1	1	1	2	1	2.00	1.00	0.00	0

ReLU dengan perhitungan program

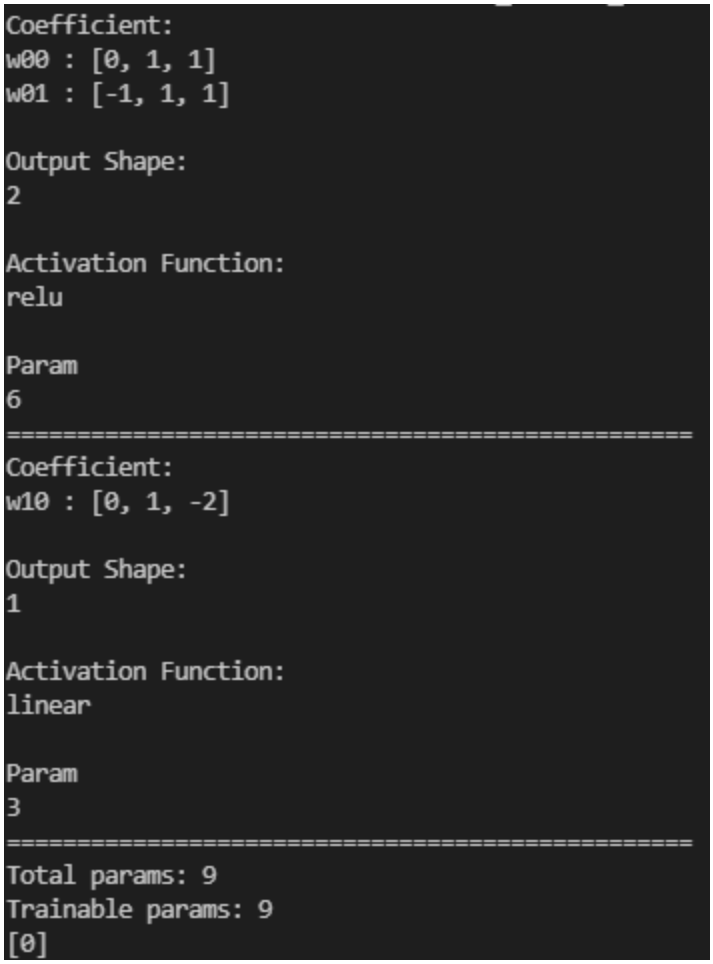
TestCase ReLU dengan input (0,0)



Test Case ReLU dengan input (0,1)



Test Case ReLU dengan input (1,1)



Test Case ReLU dengan input (1,0)

```
Coefficient:
w00 : [0, 1, 1]
w01 : [-1, 1, 1]

Output Shape:
2

Activation Function:
relu

Param
6
=================================================
Coefficient:
w10 : [0, 1, -2]

Output Shape:
1

Activation Function:
linear

Param
3
=================================================
Total params: 9
Trainable params: 9
[1]
```

Test Case Relu dengan input batch [(0,0),(1,0),(0,1),(1,1)]

```
Coefficient:
w00 : [0, 1, 1]
w01 : [-1, 1, 1]
Output Shape:
1

Activation Function:
linear

Param
3
=================================================
Total params: 9
Trainable params: 9
[0, 1, 1, 0]
```

PEMBAGIAN TUGAS

13519012 Kahfi Soobhan Zulkifli	Membuat operasi matriks dalam FFNN dan definisi aktivasi fungsi
13519022 Jose Galbraith Hasintongan	Membuat fungsi summary dan perhitungan manual fungsi aktivasi di dokumen
13519032 Muhammad Fahkry Malta	Membuat dan membantu pembuatan fungsi parser serta mengisi hasil pengujian di laporan
13519142 Muhammad Akram Al Bari	Membuat operasi matriks dalam FFNN dan struktur umum model FFNN, membuat format pembacaan model dari file .txt