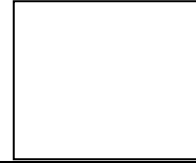




LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : KONSEP JST
NAMA : DIMAS TRI MUSTAKIM
NIM : 205150200111049
TANGGAL : 12/09/2022
ASISTEN : ANDIKA IRZA PRADANA



A. Praktikum

1. Buka Google Colaboratory melalui [tautan ini](#).
2. Tulis kode berikut ke dalam setiap *cell* pada *notebook* tersebut.
 - a. Fungsi *Step Biner*

```
def binstep(x, th=0):  
    return 1 if x >= th else 0
```

- b. Fungsi McCulloch-Pitts (MCP) Neuron

```
import numpy as np  
  
def MCP(x, w, th):  
    y_in = np.dot(x, w)  
    y_out = binstep(y_in, th)  
  
    return y_out
```

- c. Fungsi Hitung Akurasi

```
def calc_accuracy(a, b):  
    s = [1 if a[i] == b[i] else 0 for i in range(len(a))]  
  
    return sum(s) / len(a)
```

- d. Fungsi Logika AND

```
def AND(X):  
    w = 1, 1  
    th = 2  
    y = [MCP(i, w, th) for i in X]  
  
    return y
```

- e. Eksekusi Logika AND

```

data = (0, 0), (0, 1), (1, 0), (1, 1)
output = AND(data)
true = 0, 0, 0, 1
accuracy = calc_accuracy(output, true)

print('Output:', output)
print('True:', true)
print('Accuracy:', accuracy)

```

f. Fungsi Logika OR

Buat fungsi baru untuk logika OR berdasarkan fungsi logika AND dengan mengubah nilai bobot menjadi [2, 2].

```

def OR(X):
    w = 2, 2
    th = 2
    y = [MCP(i, w, th) for i in X]

    return y

```

g. Eksekusi Logika OR

```

data = (0, 0), (0, 1), (1, 0), (1, 1)
output = OR(data)
true = 0, 1, 1, 1
accuracy = calc_accuracy(output, true)

print('Output:', output)
print('True:', true)
print('Accuracy:', accuracy)

```

h. Logika AND NOT

Buat fungsi baru untuk logika OR berdasarkan fungsi logika AND dengan mengubah nilai bobot menjadi [2, -1].

```

def ANDNOT(X):
    w = 2, -1
    th = 2
    y = [MCP(i, w, th) for i in X]

    return y

```

i. Eksekusi Logika AND NOT

```

data = (0, 0), (0, 1), (1, 0), (1, 1)

```

```

output = ANDNOT(data)
true = 0, 0, 1, 0
accuracy = calc_accuracy(output, true)

print('Output:', output)
print('True:', true)
print('Accuracy:', accuracy)

```

j. Logika XOR

```

def XOR(X):
    X_flip = [(i[1], i[0]) for i in X]
    y1 = ANDNOT(X)
    y2 = ANDNOT(X_flip)
    y = zip(y1, y2)
    z = OR(y)

    return z

```

k. Eksekusi Logika XOR

```

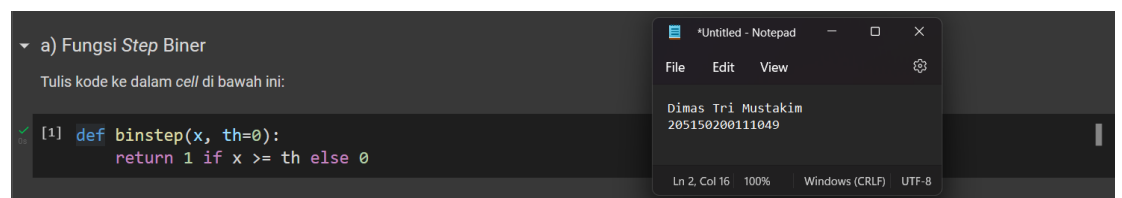
data = (0, 0), (0, 1), (1, 0), (1, 1)
output = XOR(data)
true = 0, 1, 1, 0
accuracy = calc_accuracy(output, true)

print('Output:', output)
print('True:', true)
print('Accuracy:', accuracy)

```

B. Screenshot

a. Fungsi *Step* Biner



b. Fungsi McCulloch-Pitts (MCP) Neuron

▼ b) Fungsi McCulloch-Pitts (MCP) Neuron

Tulis kode ke dalam cell di bawah ini:

```
[2] import numpy as np

def MCP(x, w, th):
    y_in = np.dot(x, w)
    y_out = binstep(y_in, th)

    return y_out
```

Untitled - Notepad

File Edit View

Dimas Tri Mustakim
205150200111049

Ln 2, Col 16 100% Windows (CRLF) UTF-8

c. Fungsi Hitung Akurasi

▼ c) Fungsi Hitung Akurasi

```
[3] def calc_accuracy(a, b):
    s = [1 if a[i] == b[i] else 0 for i in range(len(a))]

    return sum(s) / len(a)
```

Untitled - Notepad

File Edit View

Dimas Tri Mustakim
205150200111049

Ln 2, Col 16 100% Windows (CRLF) UTF-8

d. Fungsi Logika AND

▼ d) Fungsi Logika AND

Tulis kode ke dalam cell di bawah ini:

```
[4] def AND(X):
    w = 1, 1
    th = 2
    y = [MCP(i, w, th) for i in X]

    return y
```

Untitled - Notepad

File Edit View

Dimas Tri Mustakim
205150200111049

Ln 2, Col 16 100% Windows (CRLF) UTF-8

e. Eksekusi Logika AND

▼ e) Eksekusi Logika AND

Tulis kode ke dalam cell di bawah ini:

```
[5] data = (0, 0), (0, 1), (1, 0), (1, 1)
output = AND(data)
true = 0, 0, 0, 1
accuracy = calc_accuracy(output, true)

print('Output:', output)
print('True:', true)
print('Accuracy:', accuracy)
```

Output: [0, 0, 0, 1]
True: (0, 0, 0, 1)
Accuracy: 1.0

Untitled - Notepad

File Edit View

Dimas Tri Mustakim
205150200111049

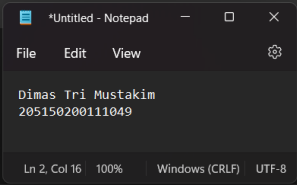
Ln 2, Col 16 100% Windows (CRLF) UTF-8

f. Fungsi Logika OR

▼ f) Fungsi Logika OR

Tulis kode ke dalam *cell* di bawah ini:

```
[6] def OR(X):  
    w = 2, 2  
    th = 2  
    y = [MCP(i, w, th) for i in X]  
  
    return y
```



g. Eksekusi Logika OR

▼ g) Eksekusi Logika OR

Tulis kode ke dalam *cell* di bawah ini:

```
[7] data = (0, 0), (0, 1), (1, 0), (1, 1)  
output = OR(data)  
true = 0, 1, 1, 1  
accuracy = calc_accuracy(output, true)  
  
print('Output:', output)  
print('True:', true)  
print('Accuracy:', accuracy)
```

Output: [0, 1, 1, 1]
True: (0, 1, 1, 1)
Accuracy: 1.0

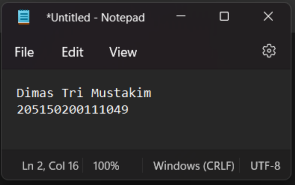


h. Logika AND NOT

▼ h) Logika AND NOT

Tulis kode ke dalam *cell* di bawah ini:

```
[10] def ANDNOT(X):  
    w = 2, -1  
    th = 2  
    y = [MCP(i, w, th) for i in X]  
  
    return y
```



i. Eksekusi Logika AND NOT

▼ i) Eksekusi Logika AND NOT

Tulis kode ke dalam *cell* di bawah ini:

```
[11] data = (0, 0), (0, 1), (1, 0), (1, 1)  
output = ANDNOT(data)  
true = 0, 0, 1, 0  
accuracy = calc_accuracy(output, true)  
  
print('Output:', output)  
print('True:', true)  
print('Accuracy:', accuracy)
```

Output: [0, 0, 1, 0]
True: (0, 0, 1, 0)
Accuracy: 1.0



j. Logika XOR

```
▼ j) Fungsi Logika XOR
Tulis kode ke dalam cell di bawah ini:

[12] def XOR(X):
    X_flip = [(i[1], i[0]) for i in X]
    y1 = ANDNOT(X)
    y2 = ANDNOT(X_flip)
    y = zip(y1, y2)
    z = OR(y)

    return z
```

k. Eksekusi Logika XOR

```
▼ k) Eksekusi Logika XOR
Tulis kode ke dalam cell di bawah ini:

data = (0, 0), (0, 1), (1, 0), (1, 1)
output = XOR(data)
true = 0, 1, 1, 0
accuracy = calc_accuracy(output, true)

print('Output:', output)
print('True:', true)
print('Accuracy:', accuracy)

Output: [0, 1, 1, 0]
True: (0, 1, 1, 0)
Accuracy: 1.0
```

C. Analisis

1. Jalankan semua *cell* (Ctrl+F9). Amati *output* dan akurasi yang dihasilkan.

Jawab:

Ketika semua *cell* dijalankan, tidak ada yang *error* yang terjadi. Untuk *cell* yang mengeksekusi fungsi logika di atasnya, akan keluar *output* bahwa dengan nilai yang dihasilkan oleh fungsi dan nilai yang di ekspektasikan adalah sama, dan nilai akurasi bernilai 1.0 (sempurnya).

2. Pada kode a, apa maksud dari *statement* `return 1 if x >= th else 0`?

Jawab:

Merupakan kode yang berperan sebagai fungsi aktivasi yaitu fungsi step biner. *Statement* tersebut akan mengembalikan nilai 1 jika nilai x (input) bernilai lebih dari sama dengan *threshold*, dan jika tidak maka akan mengembalikan nilai 0. Fungsi yang digunakan yaitu step biner dengan notasi matematika seperti berikut.

$$f(x) = \begin{cases} 1 & \text{if } x \geq \theta \\ 0 & \text{if } x < \theta \end{cases}$$

3. Pada kode b:

- a. Apakah perbedaan antara variabel *y_in* dan *y_out*?

Jawab:

Variabel y_{in} merupakan hasil dari dot product dari input x dan *weight*, sedangkan y_{out} merupakan hasil y_{in} yang telah melewati fungsi aktivasi yaitu fungsi step biner.

- b. Apakah yang dilakukan oleh fungsi `np.dot()`?

Jawab:

Fungsi tersebut akan menghasilkan hasil perkalian (*dot product*) dari *array* yang masuk. Fungsi tersebut digunakan untuk perkalian skalar antara *array input* dan *array weight* (bobot).

4. Pada kode c, apakah kegunaan dari variabel-variabel w dan th ?

Jawab:

Variable w merupakan *weight* (bobot) dan th merupakan threshold. Variable w digunakan dalam perhitungan weighted sum, dimana w dikalikan dengan input x . Weight sendiri merepresentasikan kekuatan koneksi antar unit. Variable th merupakan *threshold*, yang akan digunakan di dalam fungsi aktivasi. Fungsi aktivasi yang digunakan adalah fungsi step biner.

5. Pada kode j, apakah tujuan dari variabel X_{flip} ?

Jawab:

Variabel x_{flip} akan terisi nilai variabel X (input) yang telah dibalik posisi masing-masing pasang isinya.

6. Pada hal apa saja terdapat kesamaan antara neuron biologis dan neuron tiruan?

Jawab:

Kesamaannya yaitu bahwa neuron di keduanya merupakan unit paling sederhana di dalam sistem jaringan saraf. Keduanya juga merupakan unit yang digunakan sebagai pemrosesan informasi, dan dapat meneruskan informasi ke neuron lain.

7. Jelaskan kelebihan dan kekurangan yang dimiliki McCulloch-Pitts neuron.

Jawab:

Kelebihan dari McCulloch-Pitts neuron adalah bahwa modelnya yang sederhana, sehingga kebutuhan komputasinya rendah. Sedangkan kekurangannya yaitu seperti:

- McCulloch-Pitts neuron hanya dapat memproses input dan output dalam bentuk Boolean. Hal tersebut dikarenakan model ini hanya menggunakan fungsi aktivasi step biner yang hanya menghasilkan nilai 0 atau 1, dan fungsi pembobotannya juga hanya mendukung input bertipe Boolean.
- Nilai *threshold* yang ditentukan dan di-*hard code* sehingga harus mengetahui *threshold* yang dibutuhkan terlebih dahulu.
- Tidak bisa menyelesaikan permasalahan yang tidak bisa dipisahkan secara linier seperti fungsi XOR.

8. Mengapa logika XOR lebih rumit sehingga membutuhkan kombinasi beberapa logika yang lain?

Jawab:

Karena logika XOR merupakan permasalahan yang tidak bisa dipisahkan secara linier, dan model McCulloch-Pitts hanya bisa menyelesaikan permasalahan yang hanya bisa dipisahkan secara linier.

D. Kesimpulan

Jaringan Saraf Tiruan merupakan sistem pemrosesan informasi yang menggunakan jaringan saraf biologis sebagai model. Jaringan saraf tiruan dikembangkan menggunakan model matematis dengan komponen yaitu:

- Neuron, sebagai pemroses informasi.
- Penghubung, yang menghubungkan output satu neuron ke input neuron yang lain.
- Weight atau bobot, yang mempengaruhi seberapa kuatnya informasi diteruskan.
- Fungsi aktivasi, yang memproses sinyal input menjadi sinyal output.

Pada jaringan saraf tiruan terdapat beberapa variabel dan fungsi yang diperlukan, yaitu variabel x , w , y , dan fungsi aktivasi. Variabel x melambangkan nilai input. Nilai tersebut akan masuk ke dalam neuron input yang berada pada layer pertama yang juga disebut *input layer*. Jumlah neuron pada layer tersebut disesuaikan dengan panjang vektor pada data input yang digunakan. Variabel w merupakan weight (bobot) yang merepresentasikan kekuatan koneksi antar unit. Variabel y merupakan *output* dari jaringan saraf tiruan. Dan fungsi aktivasi merupakan fungsi yang melakukan perhitungan terhadap sinyal input yang masuk ke dalam neuron dan menghasilkan aktivasi dari neuron tersebut.

Contoh jaringan saraf tiruan yang cukup sederhana adalah McCulloch-Pitts Neuron, yang diperkirakan merupakan jaringan saraf tiruan yang pertama kali dibuat. Model ini dirancang pada tahun 1943 oleh McCulloch dan Pitts. Jaringan saraf tiruan ini menggunakan aktivasi *step biner* dan bobot serta *threshold*-nya harus ditentukan secara manual. Contoh penggunaan model ini adalah pada kasus klasifikasi biner.