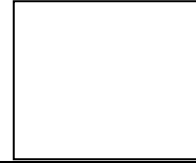




LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : HEBB NET
NAMA : DIMAS TRI MUSTAKIM
NIM : 205150200111049
TANGGAL : 19/09/2022
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 Bipolar*

```
def bipstep(y, th=0):  
    return 1 if y >= th else -1
```

b. Fungsi *Training Hebb*

```
def hebb_fit(train, target, verbose=False, draw=False,  
draw_padding=1):  
    w = np.zeros(len(train[0]) + 1)  
    bias = np.ones((len(train), 1))  
    train = np.hstack((bias, train))  
  
    for r, row in enumerate(train):  
        w = [w[i] + row[i] * target[r] for i in range(len(row))]  
        if verbose:  
            print('Bobot:', w)  
        if draw:  
            plot(line(w, 0), train, target, draw_padding)  
  
    return w
```

c. Fungsi *Testing Hebb*

```
def hebb_predict(X, w):  
    Y = []  
    for x in X:  
        y_in = w[0] + np.dot(x, w[1:])  
        y = bipstep(y_in)  
        Y.append(y)  
    return Y
```

d. 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)
```

e. Logika AND

```
from sklearn.metrics import accuracy_score  
  
train = (1, 1), (1, -1), (-1, 1), (-1, -1)  
target = 1, -1, -1, -1  
model = hebb_fit(train, target, verbose=False, draw=False)  
output = hebb_predict(train, model)  
accuracy = accuracy_score(output, target)  
  
print('Output:', output)  
print('Target:', target)  
print('Accuracy:', accuracy)
```

f. Logika OR

```
train = (1, 1), (1, -1), (-1, 1), (-1, -1)  
target = 1, 1, 1, -1  
model = hebb_fit(train, target, verbose=True, draw=True)  
output = hebb_predict(train, model)  
accuracy = accuracy_score(output, target)  
  
print('Output:', output)  
print('Target:', target)  
print('Accuracy:', accuracy)
```

g. Logika AND NOT

```
train = (1, 1), (1, -1), (-1, 1), (-1, -1)  
target = -1, 1, -1, -1  
  
model = hebb_fit(train, target, verbose=True, draw=True)  
output = hebb_predict(train, model)  
accuracy = accuracy_score(output, target)  
  
print('Output:', output)  
print('Target:', target)  
print('Accuracy:', accuracy)
```

h. Logika XOR

```
train = (1, 1), (1, -1), (-1, 1), (-1, -1)  
target = -1, 1, 1, -1
```

```

model = hebb_fit(train, target, verbose=True, draw=True)
output = hebb_predict(train, model)
accuracy = accuracy_score(output, target)

print('Output:', output)
print('Target:', target)
print('Accuracy:', accuracy)

```

B. Screenshot

a. Fungsi *Step* Bipolar

```

a) Fungsi step Bipolar
Tulis kode ke dalam cell di bawah ini:

[2] def bipstep(y, th=0):
    return 1 if y >= th else -1

```

b. Fungsi *training* Hebb

```

b) Fungsi training Hebb
Tulis kode ke dalam cell di bawah ini:

[3] def hebb_fit(train, target, verbose=False, draw=False, draw_padding=1):
    w = np.zeros(len(train[0]) + 1)
    bias = np.ones((len(train), 1))
    train = np.hstack((bias, train))

    for n, row in enumerate(train):
        w = [w[i] + row[i] * target[r] for i in range(len(row))]
        if verbose:
            print('Robot:', w)
        if draw:
            plot(line(w, 0), train, target, draw_padding)

    return w

```

c. Fungsi *Testing* Hebb

```

c) Fungsi testing Hebb
Tulis kode ke dalam cell di bawah ini:

[4] def hebb_predict(X, w):
    Y = []
    for x in X:
        y_in = w[0] + np.dot(x, w[1:])
        y = bipstep(y_in)
        Y.append(y)
    return Y

```

d. Fungsi *Hitung* Akurasi

```
d) Fungsi Hitung Akurasi

[5] 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)
```

e. Logika AND

```
e) Logika AND

Tulis kode ke dalam cell di bawah ini:

[6] from sklearn.metrics import accuracy_score

train = (1, 1), (1, -1), (-1, 1), (-1, -1)
target = 1, -1, -1, -1
model = hebb_fit(train, target, verbose=False, draw=False)
output = hebb_predict(train, model)
accuracy = accuracy_score(output, target)

print('Output:', output)
print('Target:', target)
print('Accuracy:', accuracy)

Output: [1, -1, -1, -1]
Target: (1, -1, -1, -1)
Accuracy: 1.0
```

f. Logika OR

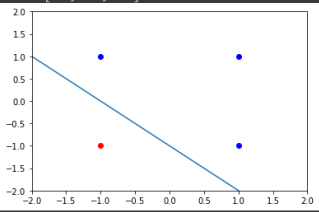
```
f) Logika OR

Tulis kode ke dalam cell di bawah ini:

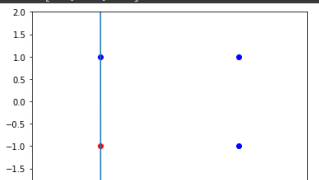
[7] train = (1, 1), (1, -1), (-1, 1), (-1, -1)
target = 1, 1, 1, -1
model = hebb_fit(train, target, verbose=True, draw=True)
output = hebb_predict(train, model)
accuracy = accuracy_score(output, target)

print('Output:', output)
print('Target:', target)
print('Accuracy:', accuracy)
```

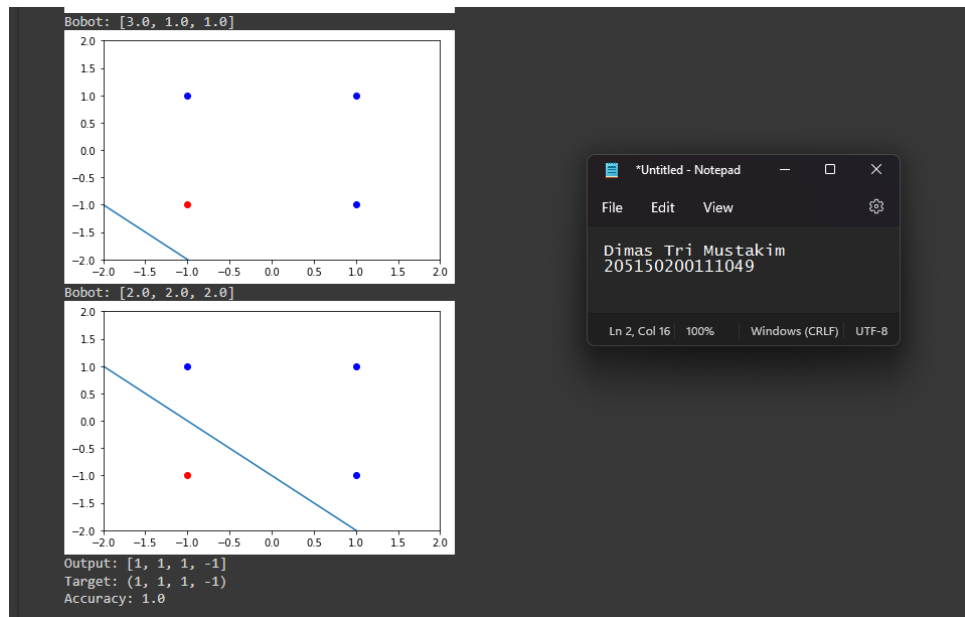
Bobot: [1.0, 1.0, 1.0]



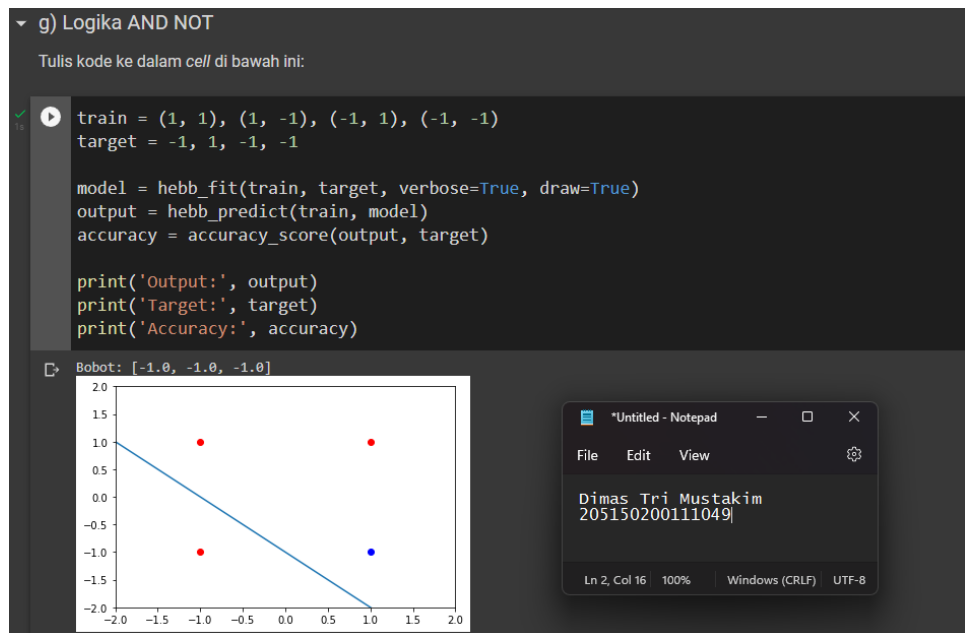
Bobot: [2.0, 2.0, 0.0]

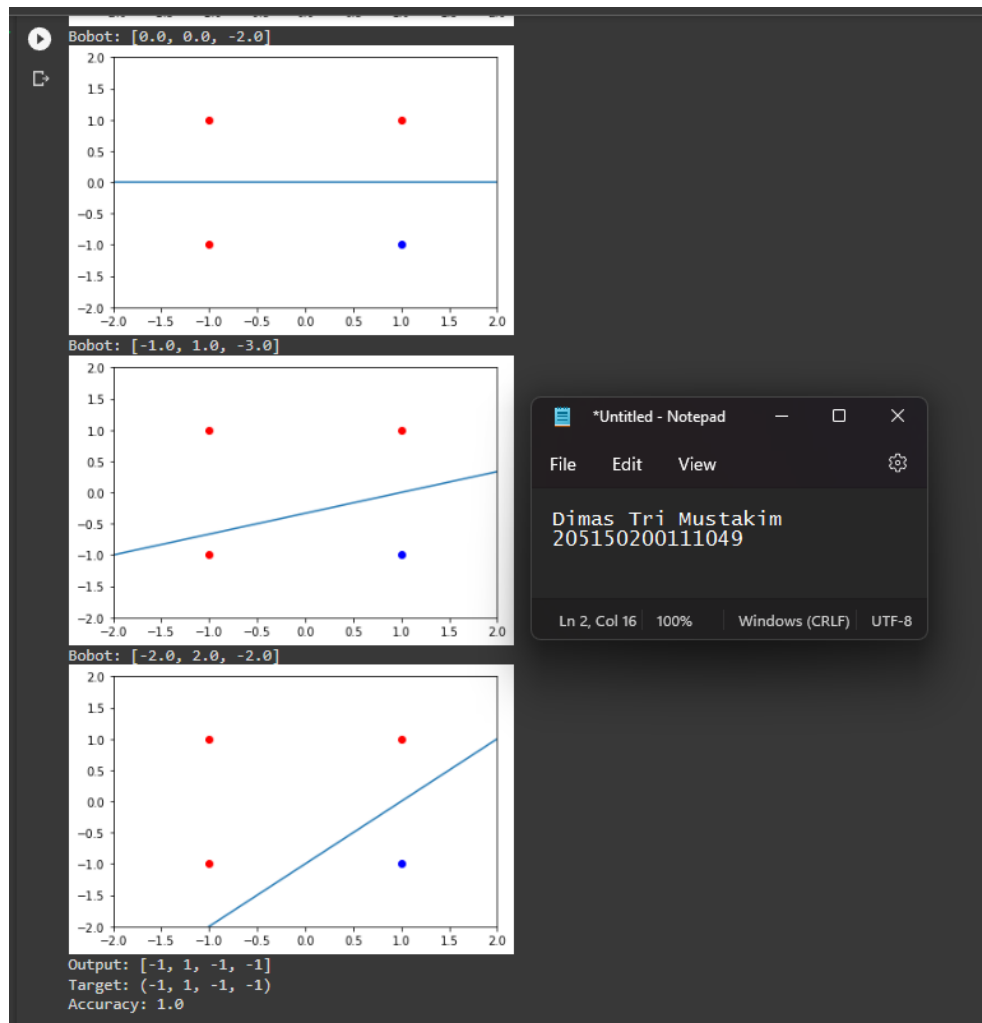


1s completed at 9:04 AM



g. Logika AND NOT





h. Logika XOR

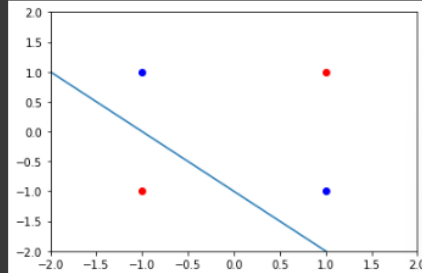
h) Logika XOR

Tulis kode ke dalam cell di bawah ini:

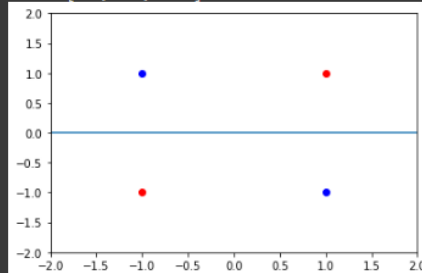
```
train = (1, 1), (1, -1), (-1, 1), (-1, -1)
target = -1, 1, 1, -1
model = hebb_fit(train, target, verbose=True, draw=True)
output = hebb_predict(train, model)
accuracy = accuracy_score(output, target)

print('Output:', output)
print('Target:', target)
print('Accuracy:', accuracy)
```

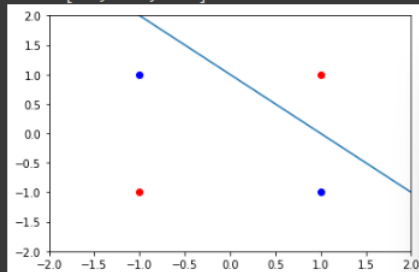
Bobot: [-1.0, -1.0, -1.0]



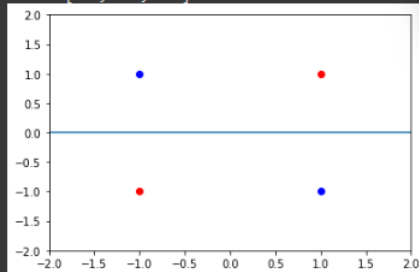
Bobot: [0.0, 0.0, -2.0]



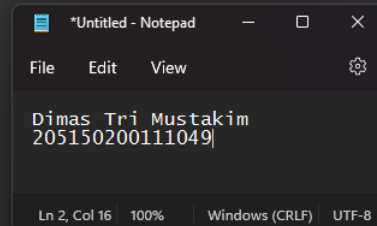
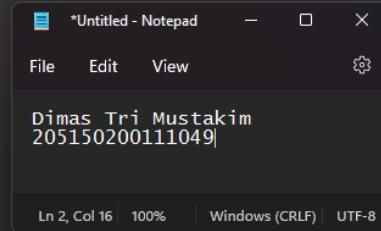
Bobot: [1.0, -1.0, -1.0]



Bobot: [0.0, 0.0, 0.0]



Output: [1, 1, 1, 1]
Target: (-1, 1, 1, -1)
Accuracy: 0.5



C. Analisis

1. Pada klasifikasi menggunakan logika XOR, mengapa akurasi yang didapatkan tidak mencapai 1 (100%)?

Jawab:

Karena logika XOR merupakan permasalahan yang termasuk yang *not linearly-separable*. Dapat dilihat dari hasil plotingan, bahwa hebb net berusaha untuk menemukan garis yang dapat memisahkan klasifikasi XOR menggunakan garis linear, tetapi tidak akan bisa karena termasuk non-linear separable.

2. Lakukan proses training dan testing menggunakan data berikut.

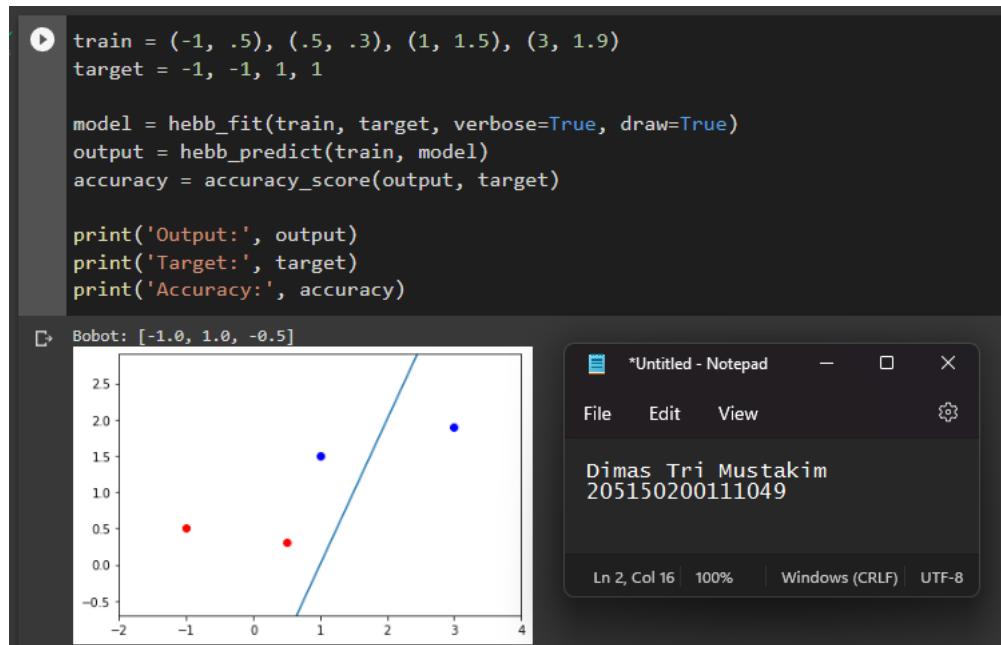
Training: (-1, .5), (.5, .3), (1, 1.5), (3, 1.9)

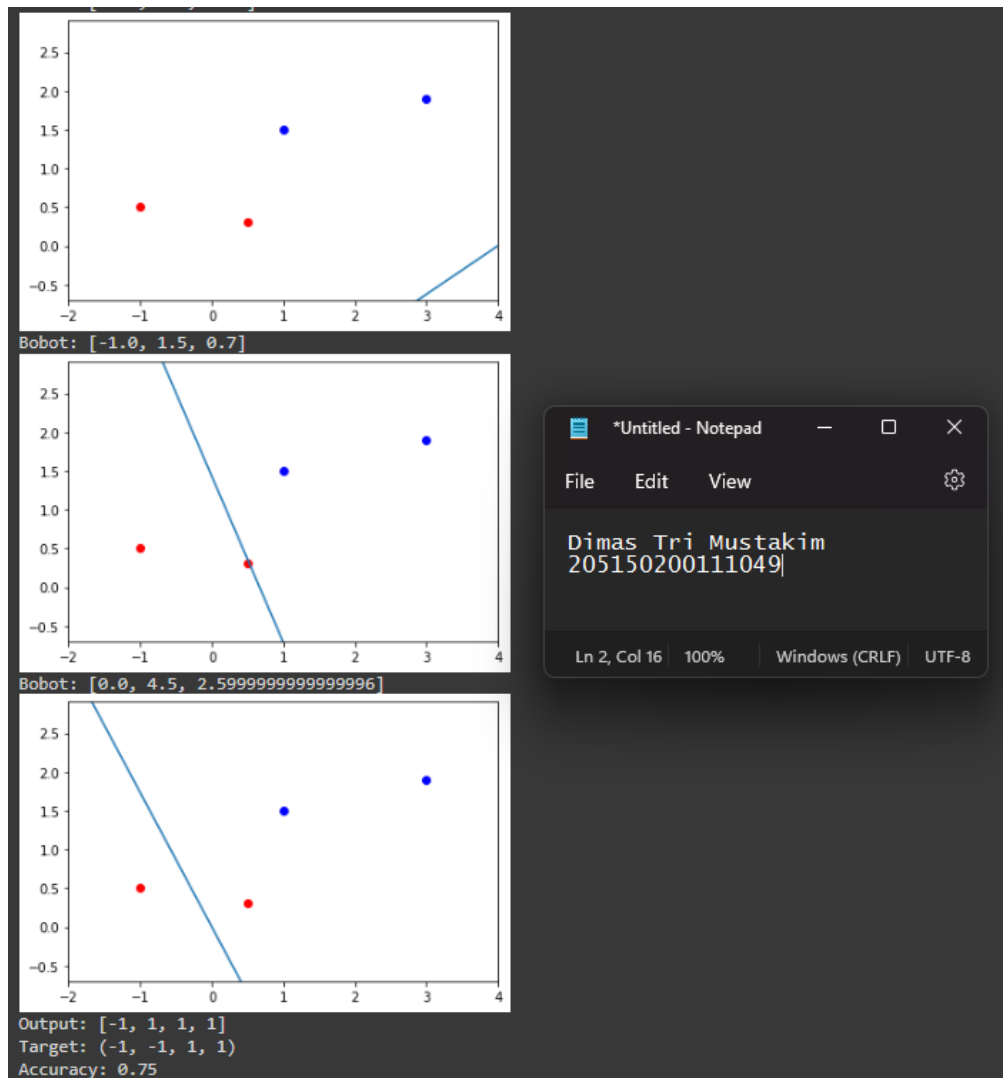
Target: (-1, -1, 1, 1)

Berapa akurasi yang didapatkan? Mengapa tidak dapat mencapai akurasi 1 (100%)?

Jawab:

Screenshot:





Merupakan Dari hasil percobaan tersebut, didapatkan bahwa hebbnet mendapatkan akurasi sebesar 75%. Hal tersebut karena bentuk input dari data yang ingin diklasifikasikan tidak berbentuk bipolar. Hebb net paling bagus digunakan untuk data input dan output bipolar.

D. Kesimpulan

Jaringan Hebb merupakan salah satu ANN yang menggunakan Hebb rule sebagai algoritma pelatihannya. Algoritma tersebut hanya akan menggunakan setiap data latih sebanyak satu kali dalam proses pelatihan (1 epoch). Jika dibandingkan dengan Mcculloch-pitts, salah satu kelebihan dari Hebb Net adalah adanya proses *training*, berbeda dengan Mcculloch-pitts yang membutuhkan kita untuk memasukkan bobot sendiri. Perbedaan lain adalah tipe data yang baik digunakan, hebb net paling bagus menggunakan data input dan output bipolar, sedangkan Mcculloch pitts

menggunakan data binary. Pada Hebb Net terdapat bias, sedangkan Mcculloch-pitts tidak.

Bias adalah bobot tambahan pada suatu jaringan saraf tiruan diluar neuron yang telah ditetapkan. Fungsi dari bias dapat disamakan dengan fungsi konstan pada fungsi linear, yaitu untuk menggeser garis atau hasil agar dapat memberikan *coverage* yang lebih baik. Sama seperti bobot, bias dapat dicari pada saat proses training.

Epoch merupakan istilah dalam machine learning untuk menyebut fase dimana semua data sudah diproses sebanyak satu kali pada proses *training*. Contohnya untuk 10 data latih, ketika semuanya sudah dimasukkan pada proses training, disebut satu epoch. Jika setiap data sudah dimasukkan sebanyak 2 kali (20 pemrosesan data) disebut dua epoch dan seterusnya.