



LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB : MADALINE

NAMA : RAIKHAN GEZA ALBURAMA
NIM : 225150207111040
TANGGAL : 1/10/2024
ASISTEN : ALIFAH KHAIRUNNISA
ANDHIKA IHSAN CENDEKIA

A. Praktikum

1. Buka Google Collaboratory melalui [tautan ini](#)
2. Tulis kode berikut ke dalam setiap *cell* pada *notebook* tersebut.
 - a. Import modul

```
import numpy as np
```

b. Fungsi aktivasi

```
def
    aktivasi(x):
        if x < 0:
            return -1
        else:
            return 1
```

c. Fungsi Training Madaline

```
def
    train(train_data,train_target,alpha=0.1,max_epoch=10):
        w = np.random.random((2,2))
        v =
        np.array([0.5,0.5]) b
        = np.random.random(2)
        b = np.append(b,0.5)
        epoch = 0
        v_aktivasi = np.vectorize(aktivasi)
        weight_updated = True
        while weight_updated == True and epoch < max_epoch:
            print("Epoch :", epoch+1)

            weight_updated = False

            for data,target in zip(train_data,train_target):
                z_in = np.dot(data,w)+ b[:-1]
                print("Z_in
                :",z_in) z =
```

```

v_aktivasi(z_in)
print("Z :",z)

y_in = np.dot(z,v) + b[-1]

print("Y_in
:",y_in)      y      =
v_aktivasi(y_in)
print("y :",z)

print("Target :",target)

if      y!=      target:
    weight_updated      =
    True if target ==
    1:
    index      =      np.argmin(np.abs(z_in))
    print("index :",index)
    b[index] = b[index] + alpha * (1 - z_in[index])
    w[:, index] = w[:, index] + alpha * (1 -
z_in[index])*data

    elif target == -1:

        index      =      np.where(z_in>0)[0]
        print("index :",index)
        if len(index)==1:
            index      =
            index[0]
        b[index] = b[index] + alpha * (-1 - z_in[index])

        w[:, index] = w[:, index] + alpha * (-1 - z_in[index]) *

data
    epoch      =      epoch      +1
    return (w,v,b,epoch)

```

d. Fungsi Testing Madaline

```

def test(w,v,b,test_data):
    v_aktivasi
    =

```

```

np.vectorize(aktivasi)      z_in    =
np.dot(test_data, w)+ b[:-1] z    =
v_aktivasi(z_in)
y_in = np.dot(z, v) +
b[-1]      y      =
v_aktivasi(y_in)

return y

```

e. 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)

```

f. Logika AND

```

data = np.array([[1,1],[1,-1],[-1,1],[-1,-1]])
target = np.array([1,-1,-1,-1])
(w,v,b,epoch) = train(data,target,alpha=0.8,max_epoch=10)
output = test(w,v,b,data)
accuracy = calc_accuracy(output, target)

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

```

g. Logika OR

```

data = np.array([[1,1],[1,-1],[-1,1],[-1,-1]])
target = np.array([1,1,1,-1])
(w,v,b,epoch) = train(data,target,alpha=0.2,max_epoch=10)
output = test(w,v,b,data)
accuracy = calc_accuracy(output, target)
print('Epoch:', epoch)
print('Output:', output)
print('Target:', target)
print('Accuracy:', accuracy)

```

B. Screenshot

a. Import Modul

▼ a) Import modul

Tulis kode ke dalam *cell* di bawah ini:

```
[1] import numpy as np
```

0s

+

...

×

225150207111040
Raikhan Geza Alburama

b. Fungsi Aktivasi

▼ b) Fungsi Aktivasi

Tulis kode ke dalam *cell* di bawah ini:

```
def aktivasi(x):  
    if x < 0:  
        return -1  
    else:  
        return 1
```

0s

▶

225150207111040
Raikhan Geza Alburama

c. Fungsi Training Madaline

▼ c) Fungsi *Training* Madaline

Tulis kode ke dalam *cell* di bawah ini:

```
[3] def train(train_data, train_target, alpha=0.1, max_epoch=10):  
    w = np.random.random((2, 2))  
    v = np.array([0.5, 0.5])  
    b = np.random.random(2)  
    b = np.append(b, 0.5)  
    epoch = 0  
    v_aktivasi = np.vectorize(aktivasi)  
    weight_updated = True  
  
    while weight_updated and epoch < max_epoch:  
        print("Epoch :", epoch + 1)  
        weight_updated = False  
  
        for data, target in zip(train_data, train_target):  
            z_in = np.dot(data, w) + b[:-1]  
            print("Z_in :", z_in)  
            z = v_aktivasi(z_in)  
            print("Z :", z)  
  
            y_in = np.dot(z, v) + b[-1]  
            print("Y_in :", y_in)  
            y = v_aktivasi(y_in)  
            print("y :", y)  
  
            print("Target :", target)  
  
            if y != target:  
                weight_updated = True  
  
            if target == 1:  
                index = np.argmin(np.abs(z_in))  
                print("index :", index)  
                b[index] = b[index] + alpha * (1 - z_in[index])  
                w[:, index] = w[:, index] + alpha * (1 - z_in[index]) * data
```

+

...

×

225150207111040
Raikhan Geza Alburama

d. Fungsi *Testing* Madaline

▼ d) Fungsi *Testing* Adaline

Tulis kode ke dalam *cell* di bawah ini:

```
0s ✓ ▶ def test(w, v, b, test_data):  
    v_aktivasi = np.vectorize(aktivasi)  
  
    z_in = np.dot(test_data, w) + b[:-1]  
    z = v_aktivasi(z_in)  
  
    y_in = np.dot(z, v) + b[-1]  
    y = v_aktivasi(y_in)  
  
    return y
```

225150207111040
Raikhan Geza Alburama

e. Fungsi Hitung Akurasi

▼ e) Fungsi Hitung Akurasi

Tulis kode ke dalam *cell* di bawah ini:

```
0s ✓ [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)
```

+

...

225150207111040
Raikhan Geza Alburama

f. Logika AND

▼ f) Logika AND

Tulis kode ke dalam *cell* di bawah ini:

```
data = np.array([[1, 1], [1, -1], [-1, 1], [-1, -1]])
target = np.array([1, -1, -1, -1])

(w, v, b, epoch) = train(data, target, alpha=0.8, max_epoch=10)

output = test(w, v, b, data)

accuracy = calc_accuracy(output, target)

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

```
Epoch : 1
Z_in : [2.53566373 0.95051461]
Z : [1 1]
Y_in : 1.5
y : 1
Target : 1
Z_in : [0.8977789 0.83936714]
Z : [1 1]
Y_in : 1.5
y : 1
Target : -1
index : [0 1]
Z_in : [-0.77617713 -1.88465215]
Z : [-1 -1]
Y_in : -0.5
y : -1
Target : -1
Z_in : [ 0.6223843 -4.93878704]
Z : [ 1 -1]
Y_in : 0.5
y : 1
Target : -1
```

+ ... ✕

225150207111040
Raikhan Geza Alburama

```
y : 1
Target : -1
index : [0 1]
Z_in : [-0.77617713 -1.88465215]
Z : [-1 -1]
Y_in : -0.5
y : -1
Target : -1
Z_in : [ 0.6223843 -4.93878704]
Z : [ 1 -1]
Y_in : 0.5
y : 1
Target : -1
index : [0]
Epoch : 2
Z_in : [-0.7210982  2.42200832]
Z : [-1  1]
Y_in : 0.5
y : 1
Target : 1
Z_in : [-1.91835166 -0.63212657]
Z : [-1 -1]
Y_in : -0.5
y : -1
Target : -1
Z_in : [-2.07408456 -1.88465215]
Z : [-1 -1]
Y_in : -0.5
y : -1
Target : -1
Z_in : [-3.27133802 -4.93878704]
Z : [-1 -1]
Y_in : -0.5
y : -1
Target : -1
Epoch: 2
Output: [ 1 -1 -1 -1]
Target: [ 1 -1 -1 -1]
Accuracy: 1.0
```

225150207111040
Raikhan Geza Alburama

g. Logika OR

Tulis kode ke dalam *cell* di bawah ini:

```
data = np.array([[1, 1], [1, -1], [-1, 1], [-1, -1]])
target = np.array([1, 1, 1, -1])

(w, v, b, epoch) = train(data, target, alpha=0.2, max_epoch=10)

output = test(w, v, b, data)

accuracy = calc_accuracy(output, target)

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

```
Epoch : 1
Z_in : [1.92558832 1.56149115]
Z : [1 1]
Y_in : 1.5
y : 1
Target : 1
Z_in : [ 0.84776887 -0.33403455]
Z : [ 1 -1]
Y_in : 0.5
y : 1
Target : 1
Z_in : [0.36055683 0.6206208 ]
Z : [1 1]
Y_in : 1.5
y : 1
Target : 1
Z_in : [-0.71726262 -1.2749049 ]
Z : [-1 -1]
Y_in : -0.5
y : -1
Target : -1
Epoch: 1
Output: [ 1  1  1 -1]
Target: [ 1  1  1 -1]
Accuracy: 1.0
```

225150207111040
Raikhan Geza Alburama

C. Analisis

1. Berdasarkan source code yang ada, kapan proses training pada Madaline akan berhenti?

Jawaban :

Berdasarkan source code yang ada, proses training pada Madaline akan berhenti apabila nilai dari delta w tertinggi bernilai kurang dari nilai batas (threshold).

2. Cobalah mengganti nilai alpha pada logika AND dengan rentang nilai 0,1 – 1,0. Apakah ada pengaruh alpha terhadap akurasi?

Jawaban :

Pada percobaan ini, ketika terjadi pergantian nilai alpha pada logika AND dengan rentang nilai 0,1 – 1,0 maka akan terjadi ketetapan akurasi yang berada pada persentase 100%. Pada hal ini artinya learning rate atau alpha tidak berpengaruh terhadap akurasi.

3. Apakah jaringan Madaline dapat mengenali logika XOR dengan tepat? Mengapa demikian?

Jawaban :

Jaringan Madaline dapat mengenali logika XOR dengan tepat karena pada Madaline terdapat neuron tambahan dalam hidden layer nya yang membuat Madaline dapat mempelajari logika yang sifatnya non-linear seperable.

4. Ubahlah data dan target pada logika OR menggunakan bilangan biner (bukan bipolar). Jangan lupa mengubah pula fungsi aktivasi agar menghasilkan bilangan biner. Apakah Madaline dapat mengenali logika OR dengan data dan target biner? Mengapa demikian?

Jawaban :

Ketika data dan target pada logika OR diubah menggunakan bilangan biner maka yang akan terjadi adalah akurasi yang didapat hanya mencapai titik akurasi persentase bernilai 75% saja. Hal tersebut membuktikan bahwa dari 4 data yang ada, terdapat 1 data yang tidak sesuai dengan target nya.

D. Kesimpulan

Perbedaan yang dapat dilihat diantara Adaline dan Madaline yaitu, Adaline memiliki arsitektur yang hampir mirip dengan perceptron, dimana beberapa inputan beserta bias dihubungkan langsung dengan neuron output-nya. Pada Adaline, bobot dimodifikasi dengan aturan delta (LMS) dan fungsi aktivasinya adalah fungsi identitas. Sedangkan Madaline merupakan gabungan dari beberapa jaringan Adaline. Dalam Madaline terdapat hidden layer. Arsitektur madaline untuk lebih dari 2 unit masukan dapat dibentuk secara analog. Proses training pada Madaline juga lebih kompleks akibat adanya hidden layer, tetapi hal tersebut juga meningkatkan perhitungan keakurasian dari Adaline.

Contoh-contoh kasus yang bisa dipelajari Madaline dengan baik yaitu, kasus yang decision boundary nya diharapkan untuk menghasilkan sifat non-linear karena ada nya hidden layer. Lalu, kasus yang peng-update-an bobot ataupun bias nya hanya dapat dilakukan pada hidden layer saja. Selain itu, dapat juga kasus yang terdapat dua buah data untuk diklasifikasikan dan digambarkan dengan X dan dots yang dimana data objek tersebut tersebar secara non-linear