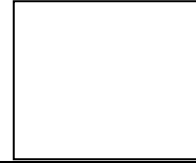




**LABORATORIUM PEMBELAJARAN ILMU KOMPUTER**  
**FAKULTAS ILMU KOMPUTER**  
**UNIVERSITAS BRAWIJAYA**

BAB : MADALINE  
NAMA : DIMAS TRI MUSTAKIM  
NIM : 205150200111049  
TANGGAL : 10/10/2022  
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. Import Modul

```
import numpy as np
```

**b. Fungsi Aktivasi**

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

**c. Fungsi Training Adaline**

```
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:  
        weight_updated = False  
  
        for data, target in zip(train_data, train_target):  
            z_in = np.dot(data, w)  
            z_in = z_in + b[:-1]  
            z = v_aktivasi(z_in)  
            y_in = np.dot(z, v) + b[-1]  
            y = v_aktivasi(y_in)  
  
            if y != target:  
                weight_updated = True
```

```

        if target == 1:
            index = np.argmin(np.abs(z_in))
            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]
            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)

```

#### d. Fungsi Testing Madaline

```

def test(w,v,b,test_data):
    v_aktivasi = np.vectorize(aktivasi)
    z_in = np.dot(test_data, w)
    z_in = z_in + 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) = train(data, target, alpha=0.8, max_epoch=10)
output = test(w,v,b, data)
accuracy = calc_accuracy(output, target)

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

```

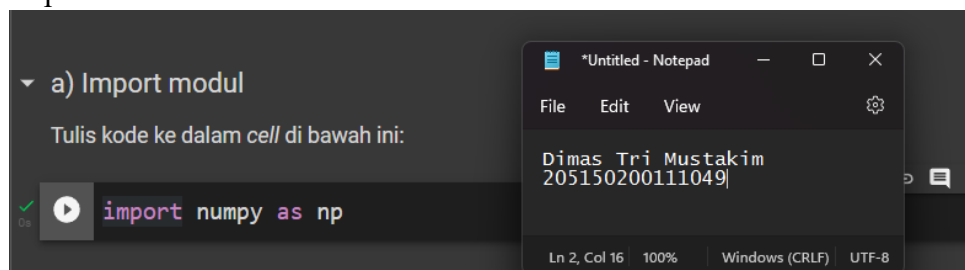
#### g. Logika OR

```
data = np.array([[1,1],[1,-1],[-1,1],[-1,-1]])
target = np.array([1,1,1,-1])
(w,v,b) = train(data,target,alpha=0.2,max_epoch=10)
output = test(w,v,b,data)
accuracy = calc_accuracy(output, target)

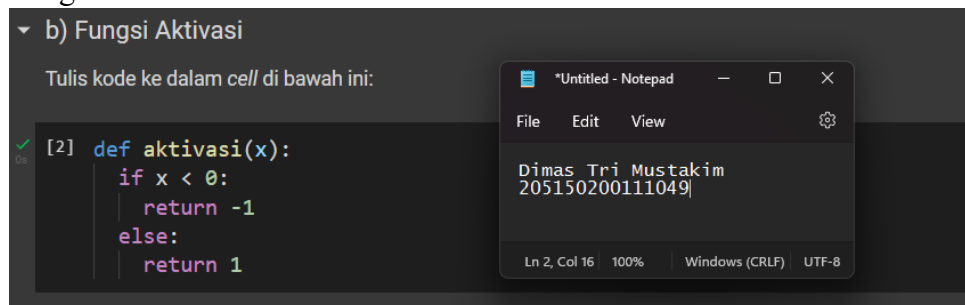
print('Output:', output)
print('Target:', target)
print('Accuracy:', accuracy)
```

## B. Screenshot

### a. Import Modul



### b. Fungsi Aktivasi



### c. Fungsi Training Madaline

▼ c) Fungsi *Training* Madaline

Tulis kode ke dalam *cell* di bawah ini:

```
[17] 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:
        weight_updated = False

        for data, target in zip(train_data, train_target):
            z_in = np.dot(data,w)
            z_in = z_in + b[:-1]
            z = v_aktivasi(z_in)
            y_in = np.dot(z,v) + b[-1]
            y = v_aktivasi(y_in)

            if y != target:
                weight_updated = True

            if target == 1:
                index = np.argmin(np.abs(z_in))
                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]
                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)
```

Untitled - Notepad

File Edit View

Dimas Tri Mustakim  
205150200111049

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

### d. Fungsi Testing Adaline

▼ d) Fungsi *Testing* Madaline

Tulis kode ke dalam *cell* di bawah ini:

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

Untitled - Notepad

File Edit View

Dimas Tri Mustakim  
205150200111049

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

### e. Fungsi Hitung Akurasi

```
e) Fungsi Hitung Akurasi
Tulis kode ke dalam cell di bawah ini:

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

```
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) = train(data, target, alpha=0.8, max_epoch=10)
output = test(w,v,b, data)
accuracy = calc_accuracy(output, target)

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

Output: [ 1 -1 -1 -1]  
Target: [ 1 -1 -1 -1]  
Accuracy: 1.0

### g. Logika OR

```
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) = train(data,target,alpha=0.2,max_epoch=10)
output = test(w,v,b,data)
accuracy = calc_accuracy(output, target)

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

Output: [ 1 1 1 -1]  
Target: [ 1 1 1 -1]  
Accuracy: 1.0

## C. Analisis

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

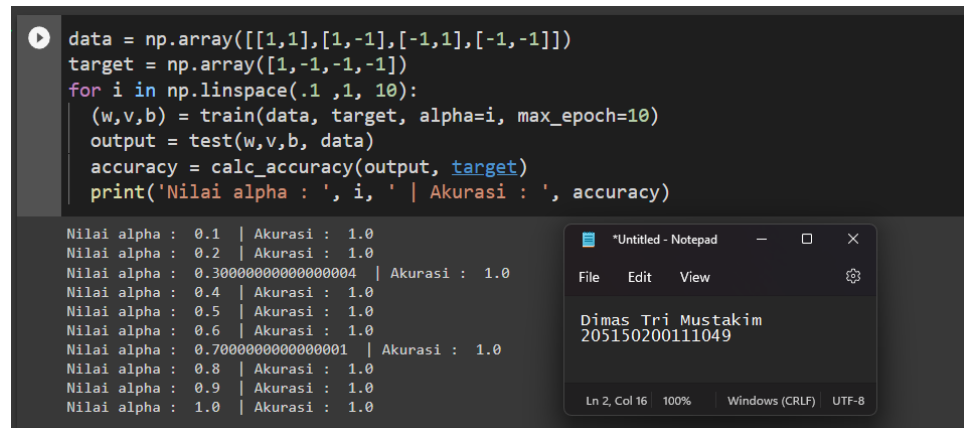
**Jawab:**

Parameter Proses training pada madaline akan berhenti ketika tidak terdapat weight yang diupdate pada satu epoch atau ketika epoch sudah mencapai max\_epoch yang tertera.

2. Cobalah mengganti nilai alpha pada logika AND dengan rentang nilai 0,1 - 1.0. Apakah ada pengaruh alpha terhadap akurasi?

**Jawab:**

Jika Dari percobaan yang saya lakukan, untuk kasus logika AND, nilai alpha di rentang 0.1 – 1.0 tidak berpengaruh terhadap akurasi. Nilai alpha hanya berpengaruh di jumlah epoch. Berikut screenshot hasil.



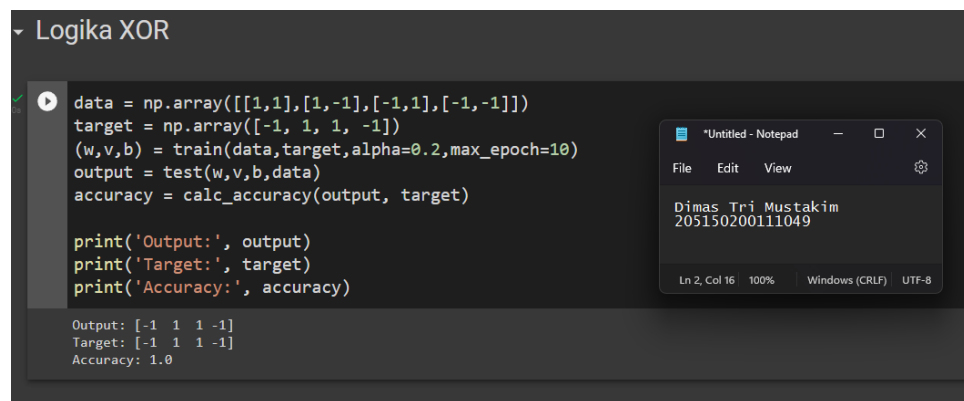
```
data = np.array([[1,1],[1,-1],[-1,1],[-1,-1]])
target = np.array([1,-1,-1,-1])
for i in np.linspace(.1,1,10):
    (w,v,b) = train(data, target, alpha=i, max_epoch=10)
    output = test(w,v,b, data)
    accuracy = calc_accuracy(output, target)
    print('Nilai alpha : ', i, ' | Akurasi : ', accuracy)
```

Nilai alpha : 0.1 | Akurasi : 1.0  
Nilai alpha : 0.2 | Akurasi : 1.0  
Nilai alpha : 0.30000000000000004 | Akurasi : 1.0  
Nilai alpha : 0.4 | Akurasi : 1.0  
Nilai alpha : 0.5 | Akurasi : 1.0  
Nilai alpha : 0.6 | Akurasi : 1.0  
Nilai alpha : 0.7000000000000001 | Akurasi : 1.0  
Nilai alpha : 0.8 | Akurasi : 1.0  
Nilai alpha : 0.9 | Akurasi : 1.0  
Nilai alpha : 1.0 | Akurasi : 1.0

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

**Jawab:**

Jumlah Jaringan Madaline dapat mengenali logika XOR dengan tepat. Hal tersebut karena Madaline merupakan jaringan multilayer yang tersusun dari sekumpulan Adaline. Algoritma neural network multi-layer bisa mengenali permasalahan yang termasuk non-linear. Arsitektur multi-layer tersebut memungkinkan untuk mengklasifikasikan permasalahan dengan decision boundary yang lebih kompleks. Semakin banyak hidden layer yang ada, semakin kompleks pula permasalahan yang dapat diklasifikasikan. Berikut screenshot hasil klasifikasi XOR menggunakan Madaline.



```
data = np.array([[1,1],[1,-1],[-1,1],[-1,-1]])
target = np.array([-1, 1, 1, -1])
(w,v,b) = train(data,target,alpha=0.2,max_epoch=10)
output = test(w,v,b,data)
accuracy = calc_accuracy(output, target)

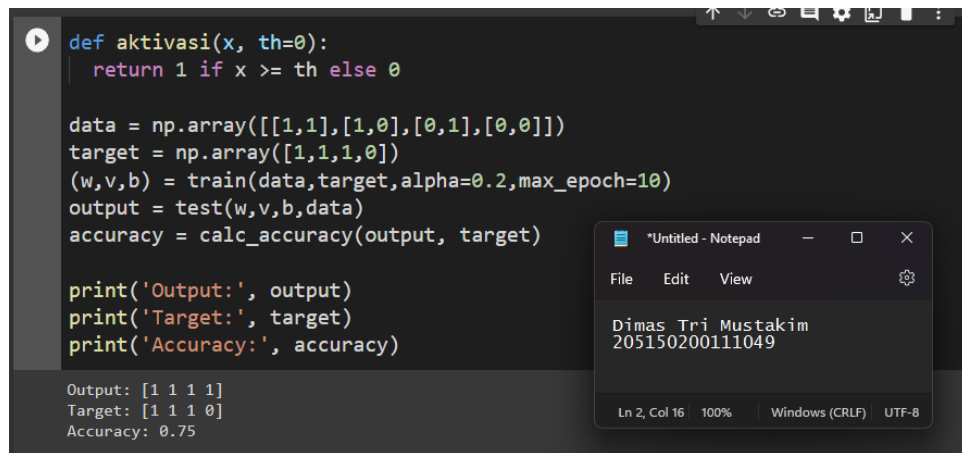
print('Output:', output)
print('Target:', target)
print('Accuracy:', accuracy)
```

Output: [-1 1 1 -1]  
Target: [-1 1 1 -1]  
Accuracy: 1.0

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?

**Jawab:**

Hal Dari percobaan tersebut, Madaline tidak dapat mengenali logika OR dengan data target maupun data latih biner. Hal tersebut karena dengan menggunakan data biner, rumus yang digunakan Madaline tidak dapat mengupdate bobot ketika targetnya adalah 0. Dapat dilihat di kode program terdapat seleksi kondisi yang mengecek apakah target bernilai 1 atau -1 dan bukan 0.



```
def aktivasi(x, th=0):  
    return 1 if x >= th else 0  
  
data = np.array([[1,1],[1,0],[0,1],[0,0]])  
target = np.array([1,1,1,0])  
(w,v,b) = train(data,target,alpha=0.2,max_epoch=10)  
output = test(w,v,b,data)  
accuracy = calc_accuracy(output, target)  
  
print('Output:', output)  
print('Target:', target)  
print('Accuracy:', accuracy)
```

Output: [1 1 1 1]  
Target: [1 1 1 0]  
Accuracy: 0.75

Untitled - Notepad  
File Edit View  
Dimas Tri Mustakim  
205150200111049  
Ln 2, Col 16 100% Windows (CRLF) UTF-8

#### **D. Kesimpulan**

Adaline merupakan salah satu jenis algoritma jaringan saraf tiruan *single layer* yang terdiri dari beberapa input neuron dan satu output neuron, sedangkan Madaline (many adaptive linear neurons) merupakan jenis jaringan saraf tiruan *multilayer* yang terdiri dari beberapa Adaline. Proses pelatihan Madaline lebih kompleks dibandingkan dengan Adaline karena memiliki banyak layer. Adaline menggunakan algoritma pelatihan yang akan meminimalisir error pada proses pelatihan, sedangkan Madaline menggunakan algoritma Madaline Rule I hingga III.

Kasus yang dapat diselesaikan Madaline dengan baik adalah kasus klasifikasi biner baik yang bersifat *linearly separable* maupun *non linearly separable*. Semakin rumit decision boundary dari sebuah permasalahan, maka perlu ditambahkan layer pada algoritma Madaline.