1. **Praktikum**
2. Buka Google Colaboratory melalui [tautan ini](https://colab.research.google.com/drive/1YOR46B8wuZQTJkHSxFUTzENZlO6tJmtP?usp=sharing).
3. Tulis kode berikut ke dalam setiap *cell* pada *notebook* tersebut.
   1. Import Modul

|  |
| --- |
| import numpy as np |

* 1. Fungsi Aktivasi

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

1. 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) |

* 1. 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 |

* 1. 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) |

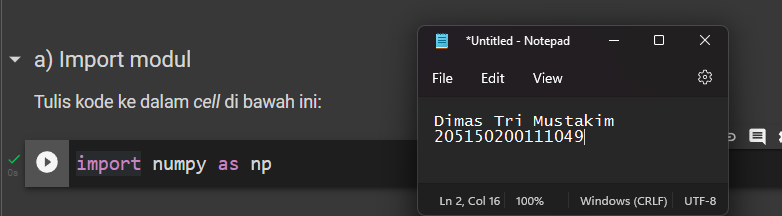
* 1. Logika AND

|  |
| --- |
| 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) |

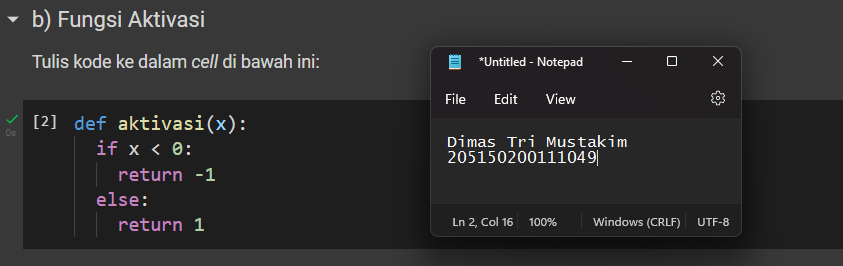
* 1. Logika OR

|  |
| --- |
| 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) |

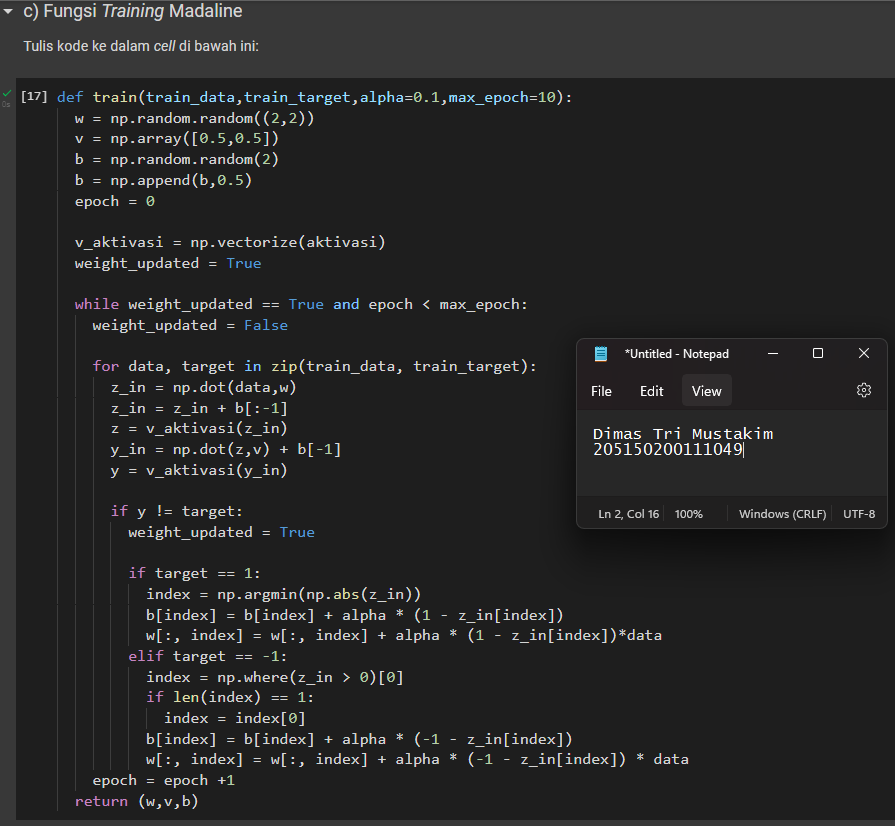
1. **Screenshot**
2. Import Modul



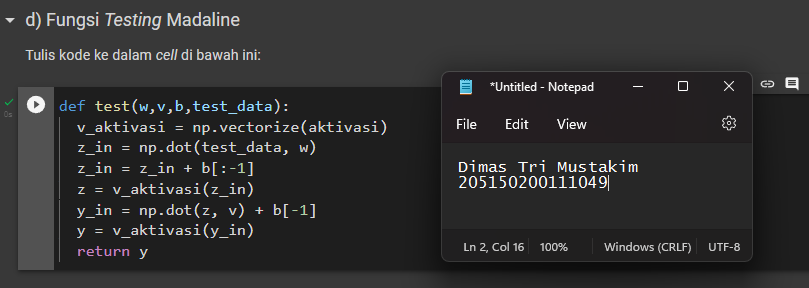
1. Fungsi Aktivasi



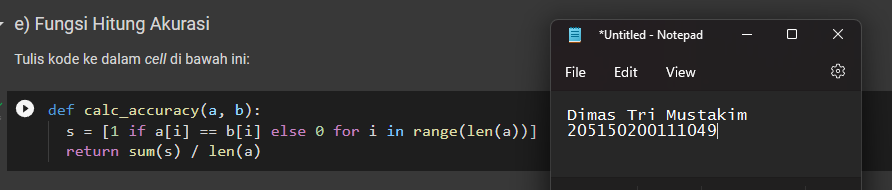
1. Fungsi Training Madaline



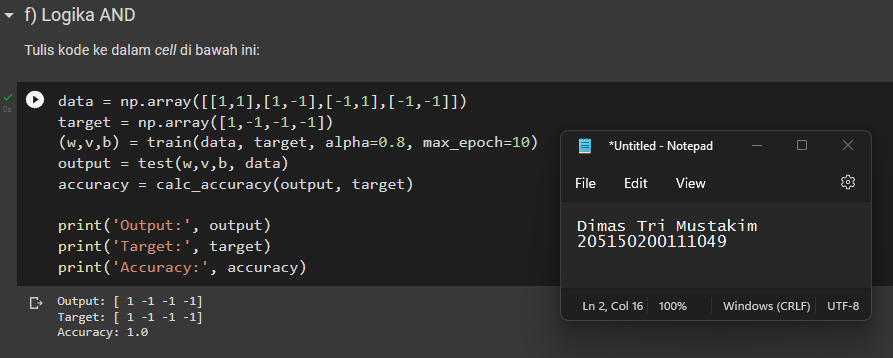
1. Fungsi Testing Adaline



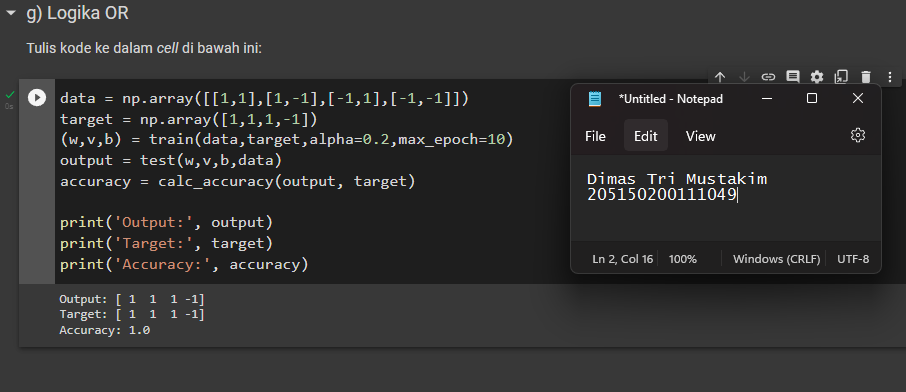
1. Fungsi Hitung Akurasi



1. Logika AND



1. Logika OR



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

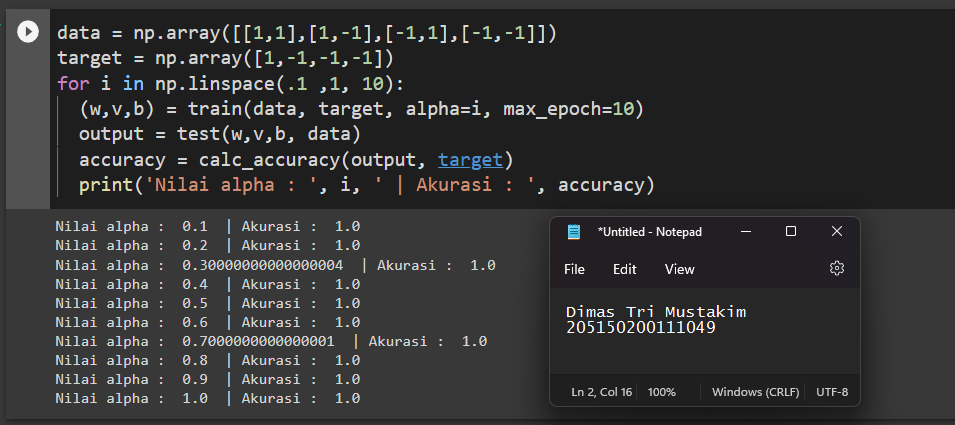
**Jawab:**

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.

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

**Jawab:**

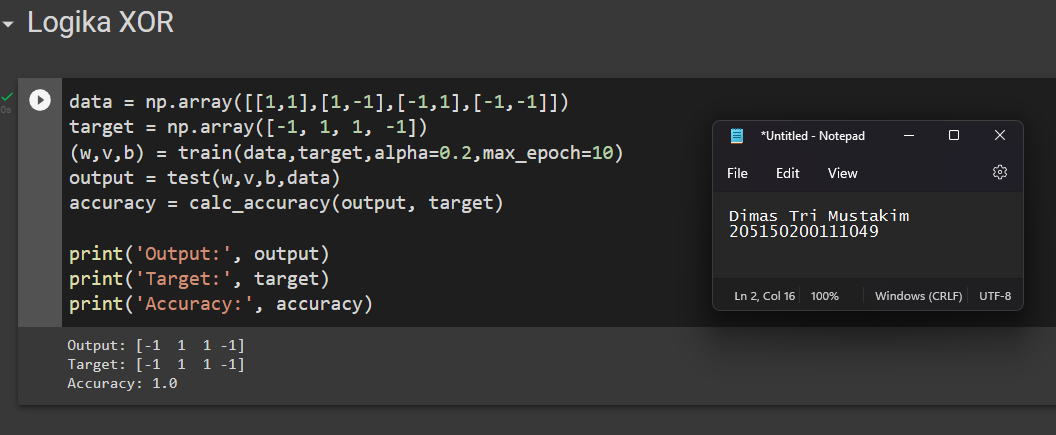
Jika Dari percobaan yang saya lakukan, untuk kasus logika AND, nilai alpha di rentang 0.1 – 1.0 tidak berpengaruh terhadap akurasi. Nilai alpha hanya berpengaruh di jumlah epoch. Berikut screenshot hasil.



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

**Jawab:**

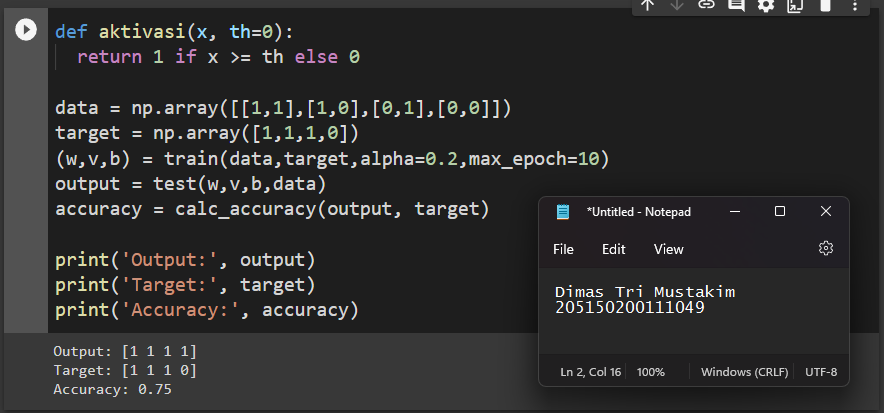
Jumlah Jaringan Madaline dapat mengenali logika XOR dengan tepat. Hal tersebut karena Madaline merupakan jaringan multilayer yang tersusun dari sekumpulan Adaline. Algoritma neural network multi-layer bisa mengenali permasalahan yang termasuk non-linear. Arsitektur multi-layer tersebut memungkinkan untuk 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.



1. 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 perecobaan 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.



1. **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.