LABORATORIUM PEMBELAJARAN ILMU KOMPUTER
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA

BAB	: BACKPROPAGATION (2)
NAMA	: DIMAS TRI MUSTAKIM
NIM	: 205150200111049
TANGGAL	: 14/11/2022
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 *Training* Backpropagation

```
import numpy as np

def bp_fit(X, target, layer_conf, max_epoch, max_error=.1,
learn_rate=.1, print_per_epoch=100):
  np.random.seed(1)
  nin = [np.empty(i) for i in layer_conf]
  n = [np.empty(j + 1) if i < len(layer_conf) - 1 else np.empty(j)
for i, j in enumerate(layer_conf)]
  w = np.array([np.random.rand(layer_conf[i] + 1, layer_conf[i +
1]) for i in range(len(layer_conf) - 1)], dtype=object)
  dw = [np.empty((layer_conf[i] + 1, layer_conf[i + 1])) for i in
range(len(layer_conf) - 1)]
  d = [np.empty(s) for s in layer_conf[1:]]
  din = [np.empty(s) for s in layer_conf[1:-1]]
  epoch = 0
  mse = 1

  for i in range(0, len(n)-1):
    n[i][-1] = 1

  while (max_epoch == -1 or epoch < max_epoch) and mse >
max_error:
    epoch += 1
    mse = 0

    for r in range(len(X)):
      n[0][:-1] = X[r]

      for L in range(1, len(layer_conf)):
        nin[L] = np.dot(n[L-1], w[L-1])
        n[L][:len(nin[L])] = sig(nin[L])
      e = target[r] - n[-1]
      mse += sum(e ** 2)
      d[-1] = e * sigd(nin[-1])
      dw[-1] = learn_rate * d[-1] * n[-2].reshape((-1, 1))
      for L in range(len(layer_conf) - 1, 1, -1):
        din[L-2] = np.dot(d[L-1], np.transpose(w[L-1][:-1]))
```

```
        d[L-2] = din[L-2] * np.array(sigd(nin[L-1]))
        dw[L-2] = (learn_rate * d[L-2]) * n[L-2].reshape((-1, 1))
      w += dw
    mse /= len(X)
    if print_per_epoch > -1 and epoch % print_per_epoch == 0:
      print(f'Epoch {epoch}, MSE: {mse}')

  return w, epoch, mse
```

b. Fungsi *Testing* Backpropagation

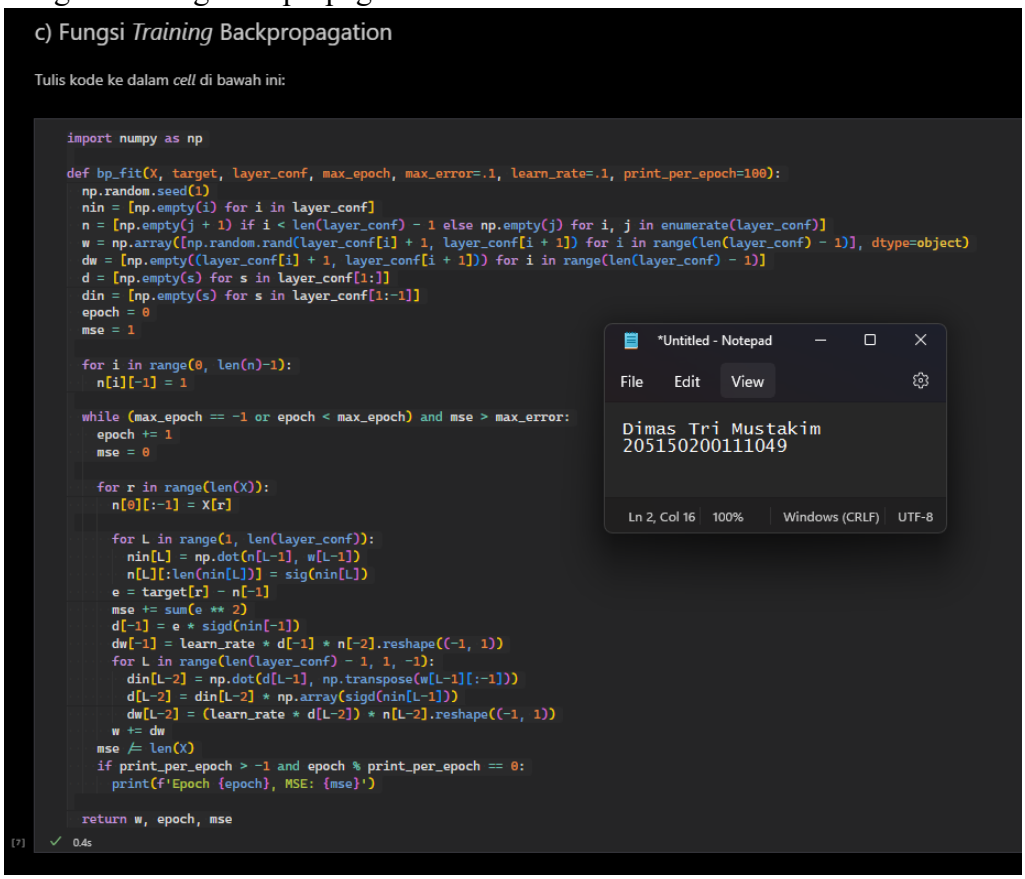
```
def bp_predict(X, w):
  n = [np.empty(len(i)) for i in w]
  nin = [np.empty(len(i[0])) for i in w]
  predict = []
  n.append(np.empty(len(w[-1][0])))
  for x in X:
    n[0][:-1] = x
    for L in range(0, len(w)):
      nin[L] = np.dot(n[L], w[L])
      n[L + 1][:len(nin[L])] = sig(nin[L])
    predict.append(n[-1].copy())
  return predict
```

c. Percobaan Klasifikasi Dataset Iris

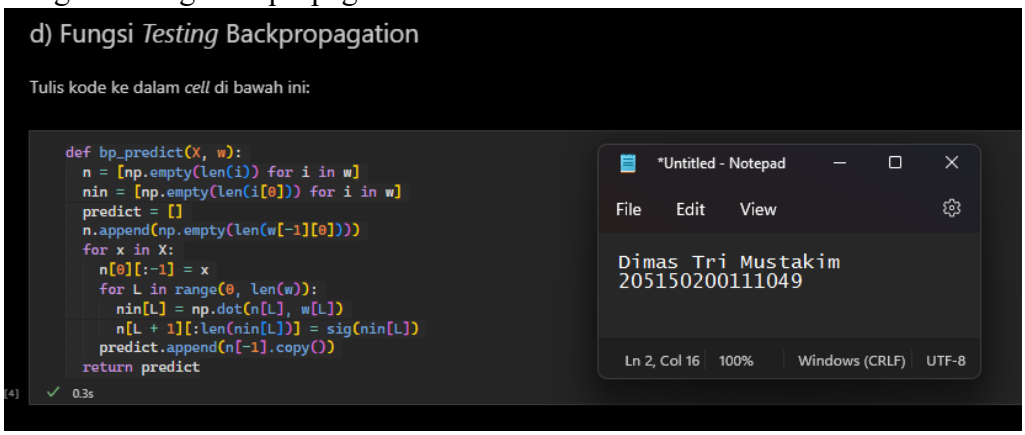
```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import minmax_scale
from sklearn.metrics import accuracy_score

iris = datasets.load_iris()
X = minmax_scale(iris.data)
Y = onehot_enc(iris.target)

X_train, X_test, y_train, y_test = train_test_split(X, Y,
test_size=.3,random_state=1)
w, ep, mse = bp_fit(X_train, y_train, layer_conf=(4, 3, 3),
learn_rate=.1, max_epoch=1000, max_error=.1, print_per_epoch=25)
print(f'Epochs: {ep}, MSE: {mse}')
predict = bp_predict(X_test, w)
predict = onehot_dec(predict)
y_test = onehot_dec(y_test)
accuracy = accuracy_score(predict, y_test)
print('Output:', predict)
print('True :', y_test)
print('Accuracy:', accuracy)
```

B. Screenshot

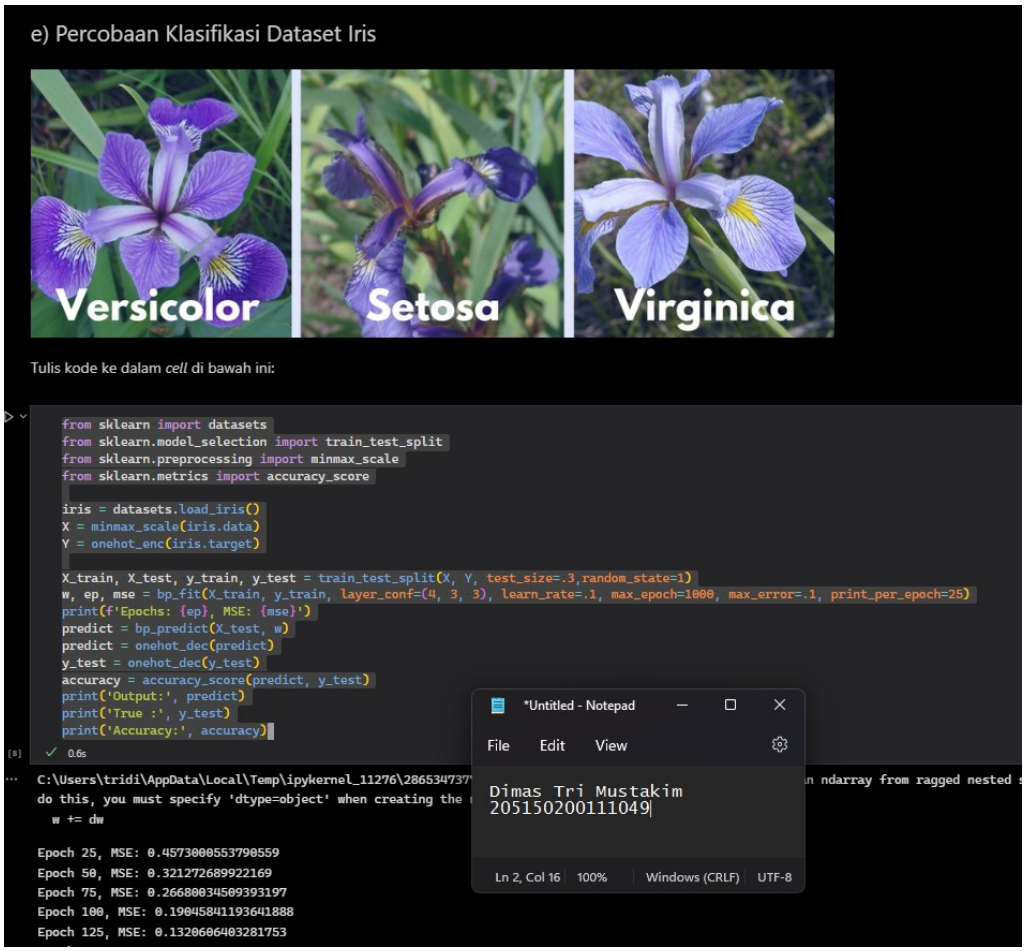
a. Fungsi Training Backpropagation

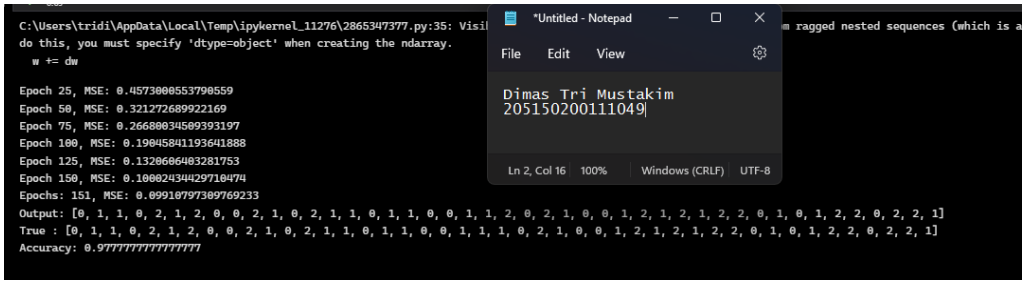


b. Fungsi Testing Backpropagation



c. Percobaan Klasifikasi Dataset Iris





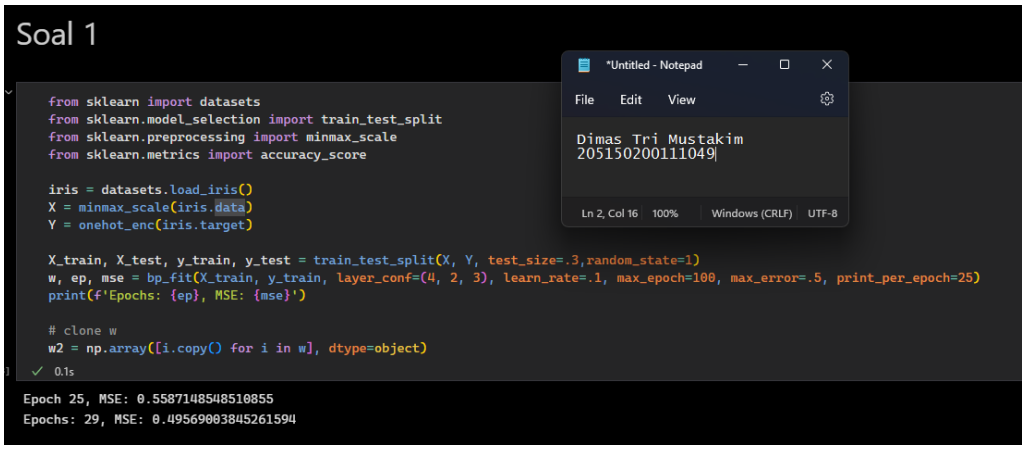
C. Analisis

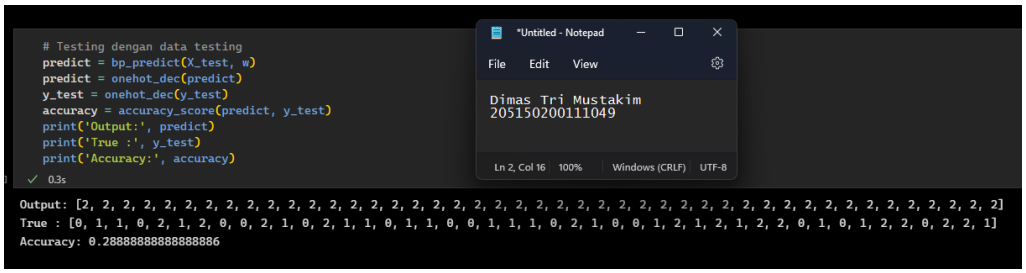
1. Lakukan klasifikasi dengan menggunakan dataset Iris seperti di atas. Ubahlah beberapa pengaturan sebagai berikut:
   - Rasio data latih 70% dan data uji 30%
   - Hidden neuron = 2
   - Max epoch = 100
   - Learning rate = 0,1
   - Max error = 0,5

   Lakukan pengujian (testing) menggunakan data latih dan data uji. Bandingkan nilai akurasi yang didapatkan. Fenomena apa yang terjadi pada pengujian ini? Mengapa hal tersebut terjadi?

   Jawab:

   Akurasi yang didapatkan dari hasil pengujian didapatkan bahwa kinerja model pada pelatihan tersebut buruk. Fenomena tersebut merupakan underfitting dan terjadi karena pemilihan parameter pelatihan yang ditunjukkan tidak optimal. Berikut screenshot proses pelatihan.





```
# Testing dengan data latih
predict = bp_predict(X_train, w2)
predict = onehot_dec(predict)
y_train = onehot_dec(y_train)
accuracy = accuracy_score(predict, y_train)
print('Output:', predict)
print('True :', y_train)
print('Accuracy:', accuracy)
```

✓ 0.6s

*Untitled - Notepad — Dimas Tri Mustakim 205150200111049 — Ln 2, Col 16 | 100% | Windows (CRLF) | UTF-8

```
Output: [2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
True : [2, 0, 0, 0, 1, 0, 0, 2, 2, 2, 2, 2, 1, 2, 1, 0, 2, 2, 0, 0, 2, 0, 2, 2, 1, 1, 2, 2, 0, 1, 1, 2, 1, 2, 1, 0, 0, 0,
0, 2, 0, 2, 2, 0, 2, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 2, 0, 0, 2, 1, 2, 1, 2, 2, 1, 2, 0]
Accuracy: 0.3523809523809524
```

2. Lakukan klasifikasi dengan menggunakan dataset Iris seperti di atas. Ubahlah beberapa pengaturan sebagai berikut:
   - Rasio data latih 70% dan data uji 30%
   - Hidden neuron = 25
   - Max epoch = 10000
   - Learning rate = 0,1
   - Max error = 0,01

   Lakukan pengujian (testing) menggunakan data latih dan data uji. Bandingkan nilai akurasi yang didapatkan. Fenomena apa yang terjadi pada pengujian ini? Mengapa hal tersebut terjadi?

   Jawab:

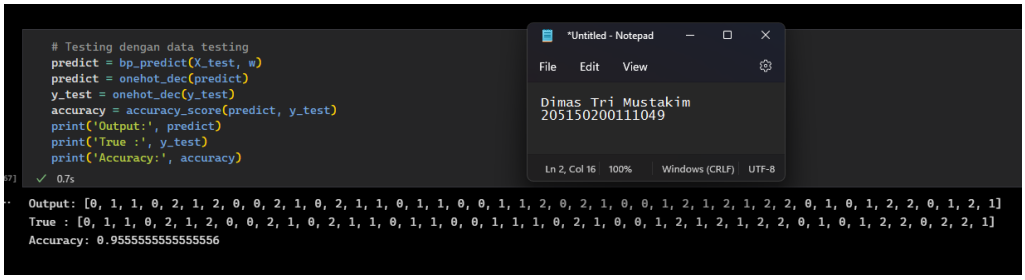
   Nilai akurasi yang didapatkan menunjukkan skor yang cukup tinggi, nilai akurasi ketika testing menggunakan data training menunjukkan nilai yang lebih besar daripada ketika menggunakan data testing. Fenomena tersebut menunjukkan overfitting, dimana model baik dalam mengklasifikasikan data latih, tetapi tidak dengan data testing (tidak bisa men-generalisasi). Berikut screenshot proses pelatihan dan testing.

```
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import minmax_scale
from sklearn.metrics import accuracy_score

iris = datasets.load_iris()
X = minmax_scale(iris.data)
Y = onehot_enc(iris.target)

X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size=.3,random_state=1)
w, ep, mse = bp_fit(X_train, y_train, layer_conf=(4, 25, 3), learn_rate=.1, max_epoch=10000, max_error=.01, print_per_epoch=25)
print(f'Epochs: {ep}, MSE: {mse}')

# clone w
w2 = np.array([i.copy() for i in w], dtype=object)
```

[66] ✓ 2m 19.1s

*Untitled - Notepad — Dimas Tri Mustakim 205150200111049 — Ln 2, Col 16 | 100% | Windows (CRLF) | UTF-8

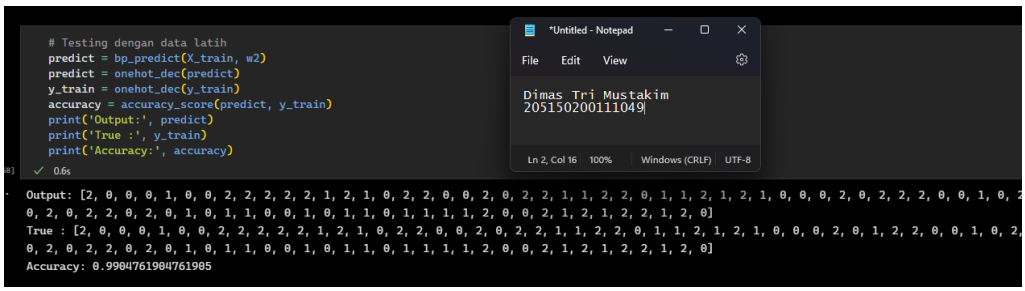
```
Output exceeds the size limit. Open the full output data in a text editor
Epoch 25, MSE: 1.9997475602526067
Epoch 50, MSE: 1.9996839107720699
Epoch 75, MSE: 1.9995729256252905
Epoch 100, MSE: 1.9993234166160632
Epoch 125, MSE: 1.998004842894972
Epoch 150, MSE: 1.5034310711672474
Epoch 175, MSE: 0.9882734055300789
Epoch 200, MSE: 0.9358385629366138
Epoch 225, MSE: 0.8962403847767554
Epoch 250, MSE: 0.572011308818784
Epoch 275, MSE: 0.17189641558830482
Epoch 300, MSE: 0.13551945448567235
Epoch 325, MSE: 0.10976277201695427
```




```
# Testing dengan data testing
predict = bp_predict(X_test, w)
predict = onehot_dec(predict)
y_test = onehot_dec(y_test)
accuracy = accuracy_score(predict, y_test)
print('Output:', predict)
print('True :', y_test)
print('Accuracy:', accuracy)
```

✓ 0.7s

```
Output: [0, 1, 1, 0, 2, 1, 2, 0, 0, 2, 1, 0, 2, 1, 1, 0, 1, 1, 0, 0, 1, 1, 2, 0, 2, 1, 0, 0, 1, 2, 1, 2, 1, 2, 2, 0, 1, 0, 1, 2, 2, 0, 1, 2, 1]
True : [0, 1, 1, 0, 2, 1, 2, 0, 0, 2, 1, 0, 2, 1, 1, 0, 1, 1, 0, 0, 1, 1, 1, 0, 2, 1, 0, 0, 1, 2, 1, 2, 1, 2, 2, 0, 1, 0, 1, 2, 2, 0, 2, 2, 1]
Accuracy: 0.9555555555555556
```




```
# Testing dengan data latih
predict = bp_predict(X_train, w2)
predict = onehot_dec(predict)
y_train = onehot_dec(y_train)
accuracy = accuracy_score(predict, y_train)
print('Output:', predict)
print('True :', y_train)
print('Accuracy:', accuracy)
```

✓ 0.6s

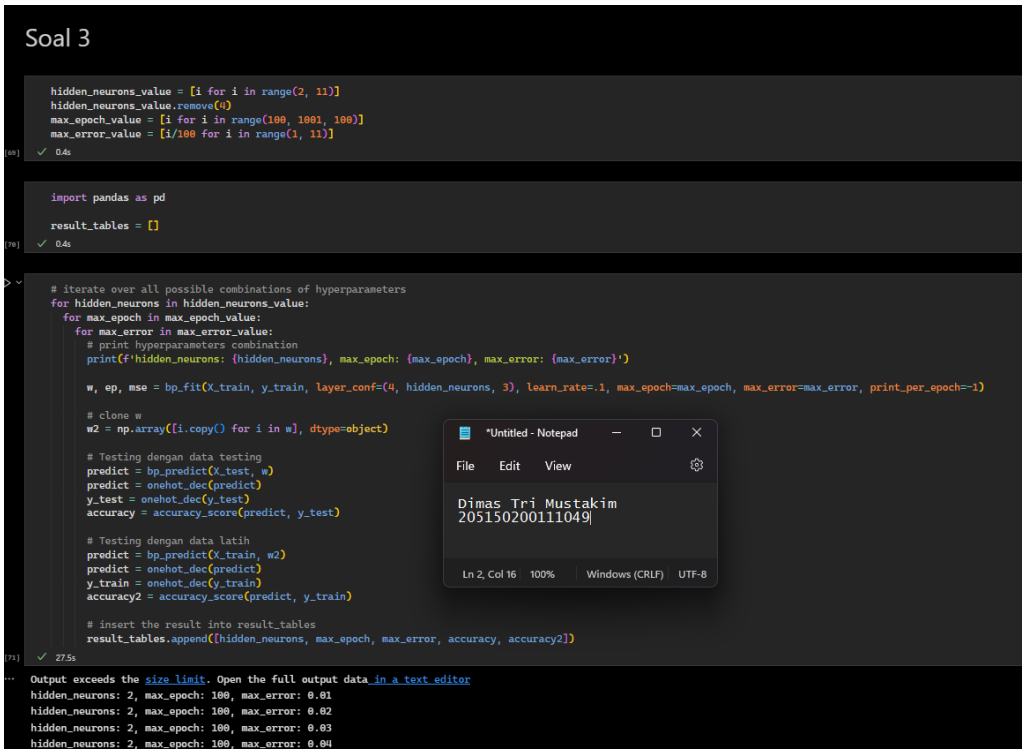
```
Output: [2, 0, 0, 0, 1, 0, 0, 2, 2, 2, 2, 1, 2, 1, 0, 2, 2, 0, 0, 2, 0, 2, 2, 1, 1, 2, 2, 0, 1, 1, 2, 1, 2, 1, 0, 0, 0, 2, 0, 2, 2, 2, 0, 0, 1, 0, 2
0, 2, 0, 2, 2, 0, 2, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 2, 0, 0, 2, 1, 2, 1, 2, 2, 1, 2, 0]
True : [2, 0, 0, 0, 1, 0, 0, 2, 2, 2, 2, 1, 2, 1, 0, 2, 2, 0, 0, 2, 0, 2, 2, 1, 1, 2, 2, 0, 1, 1, 2, 1, 2, 1, 0, 0, 0, 2, 0, 1, 2, 2, 0, 0, 1, 0, 2,
0, 2, 0, 2, 2, 0, 2, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 2, 0, 0, 2, 1, 2, 1, 2, 2, 1, 2, 0]
Accuracy: 0.9904761904761905
```

3. Ubahlah parameter berikut agar mendapatkan akurasi tertinggi saat melakukan testing menggunakan data uji :
   - Hidden neuron
   - Max epoch
   - Max error

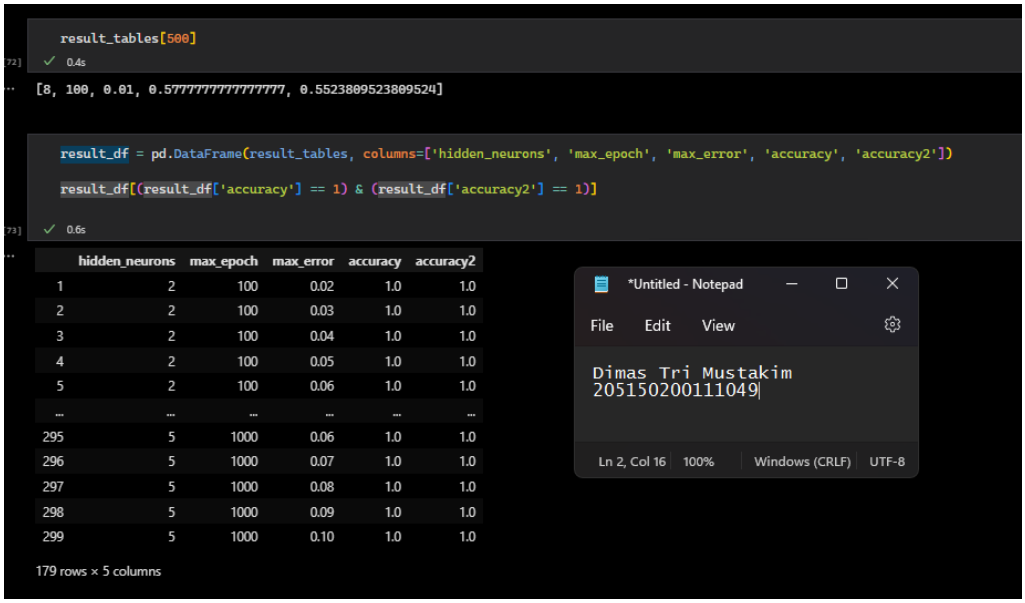
   Berapakah nilai akurasi tertinggi yang dapat Anda peroleh? Berapakah nilai masing-masing parameter tersebut?

   Jawab:

   Untuk memperoleh kombinasi parameter dengan akurasi tertinggi, saya melakukan perulangan untuk beberapa parameter dari range tertentu dan kemudian dicatat akurasi terhadap data latih dan data testing. Untuk range parameter bisa dilihat di screenshot kode program dibawah.



Dari kode program diatas, didapatkan hasil seperti berikut.



Terdapat beberapa kombinasi parameter yang dapat menghasilkan nilai akurasi sempurna ketika testing menggunakan data latih dan data uji.

D. Kesimpulan

Overfitting merupakan fenomena ketika sebuah model machine learning memberikan hasil yang akurat ketika melakukan prediksi terhadap data latih tetapi tidak ketika mendapatkan data baru. Overfitting terjadi ketika model tidak dapat menggeneralisasi dan terlalu cocok dengan dataset pelatihan. Overfitting bisa dicegah dengan cara diversifikasi dan memperbesar data training atau menggunakan beberapa strategi data seperti early stopping, pruning, regularization, dan ensembling. Cara lain adalah dengan menghapus data yang tidak relevan (noisy data), dan mengatur hyperparameter ketika pelatihan.

Underfitting merupakan fenomena ketika model tidak bisa menentukan hubungan yang berarti antara input dan output data. Model yang underfit tidak bisa melakukan klasifikasi yang baik pada data yang diberikan. Underfit terjadi ketika tidak dilatih untuk jangka waktu yang cukup (epoch), kurangnya data latih dan fitur data, serta pemilihan hyperparameter yang tidak tepat. Cara mengatasi overfitting adalah dengan menyelesaikan penyebabnya, mulai dari mencari data dengan jumlah yang cukup banyak, fitur lengkap, dan mencari hyperparameter yang seusai.