BAB : BACKPROPOGATION (1)

NAMA : RAIKHAN GEZA ALBURAMA
NIM : 225150207111040
TANGGAL : 28/10/2024
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. Fungsi Binary Encoding dan Decoding

```
def bin_enc(lbl):
    mi = min(lbl)
    length = len(bin(max(lbl) - mi + 1)[2:])
    enc = []

    for i in lbl:
        b = bin(i - mi)[2:].zfill(length)
        enc.append([int(n) for n in b])

    return enc


def bin_dec(enc, mi=0):
    lbl = []

    for e in enc:
        rounded = [int(round(x)) for x in e]
        string = ''.join(str(x) for x in rounded)
        num = int(string, 2) + mi

        lbl.append(num)

    return lbl
```

   b. Percobaan Binary Encoding dan Decoding

```
labels = 1, 2, 3, 4
enc = bin_enc(labels)
dec = bin_dec(enc, min(labels))
```

```
print(enc)
print(dec)
```

c. Fungsi One-hot Encoding dan Decoding

```
  import numpy as
np
  def
onehot_enc(lbl,
min_val=0):
  mi = min(lbl)
  enc            =
np.full((len(lbl),
max(lbl) - mi +
1),       min_val,
np.int8)
  for  i,  x  in
enumerate(lbl):
  enc[i, x - mi] =
1
  return enc

  def
onehot_dec(enc,
mi=0):
  return
[np.argmax(e) + mi
for e in enc]
```

d. Percobaan Binary Encoding dan Decoding

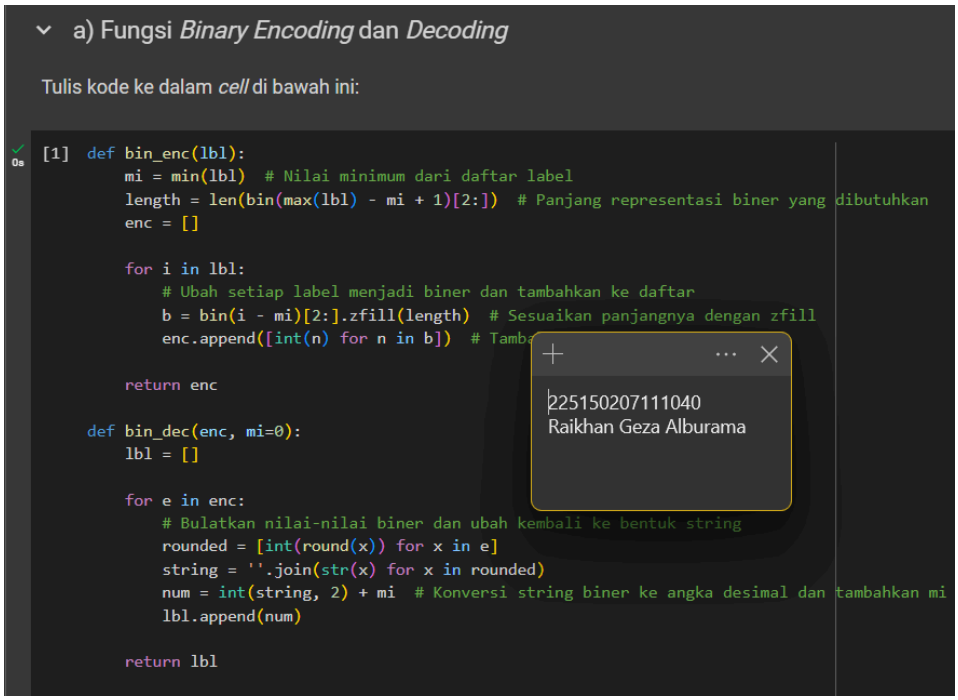
```
labels = 1, 2, 3, 4
enc = onehot_enc(labels)
dec = onehot_dec(enc, min(labels))

print(enc)
print(dec)
```

e. Fungsi Aktivasi Sigmoid dan Derivatifnya

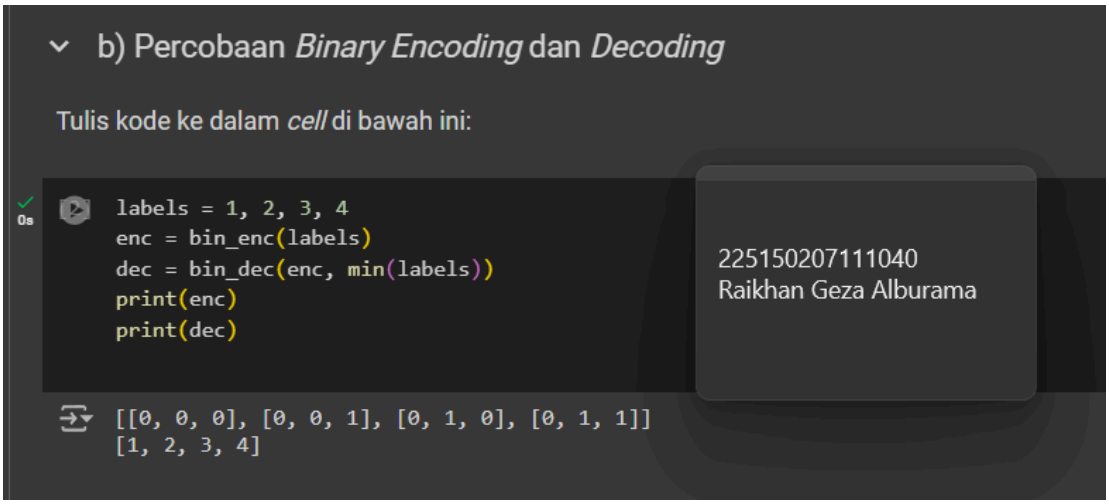
```
def sig(X):
    return [1 / (1 + np.exp(-x)) for x in X]

def sigd(X):
    output = []

    for i, x in enumerate(X):
        s = sig([x])[0]

        output.append(s * (1 - s))

    return output
```

B. Screenshot

a. Fungsi Binary Encoding dan Decoding



b. Percobaan Binary Encoding dan Decoding



c. Fungsi One-hot Encoding dan Decoding

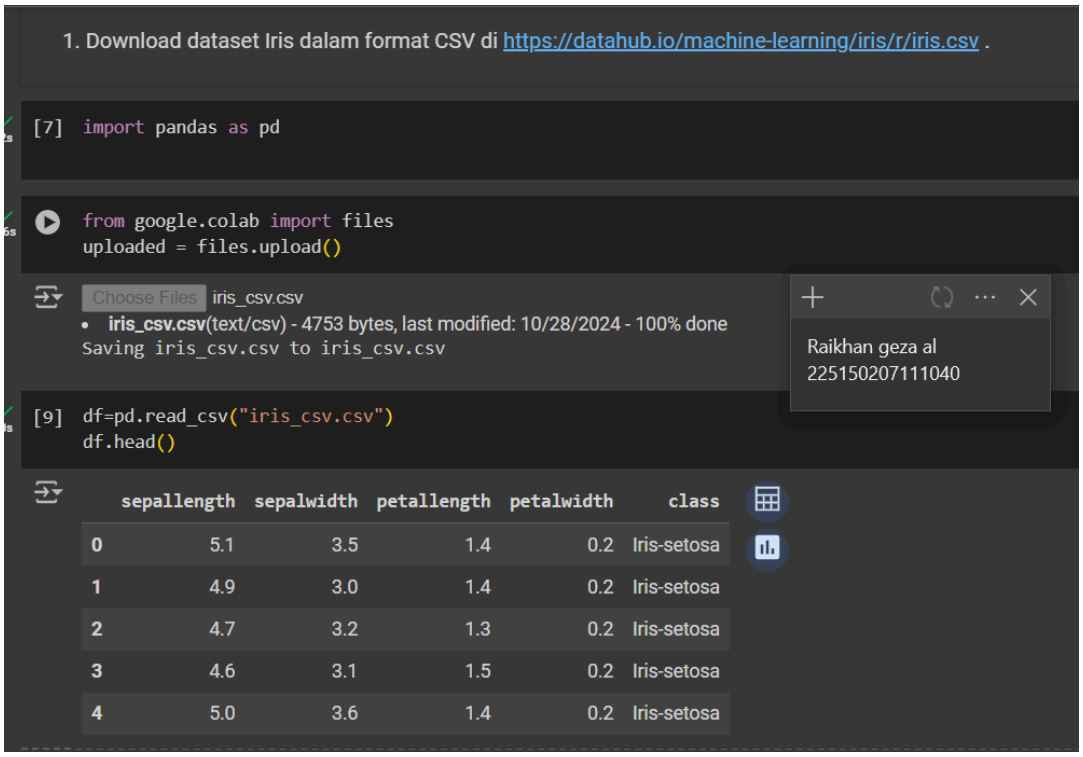
d. Percobaan Binary Encoding dan Decoding

e. Fungsi Aktivasi Sigmoid dan Derivatifnya

C. Analisis

1.Download dataset Iris dalam format CSV di https://datahub.io/machine-learning/iris/r/iris.csv .n berhenti?

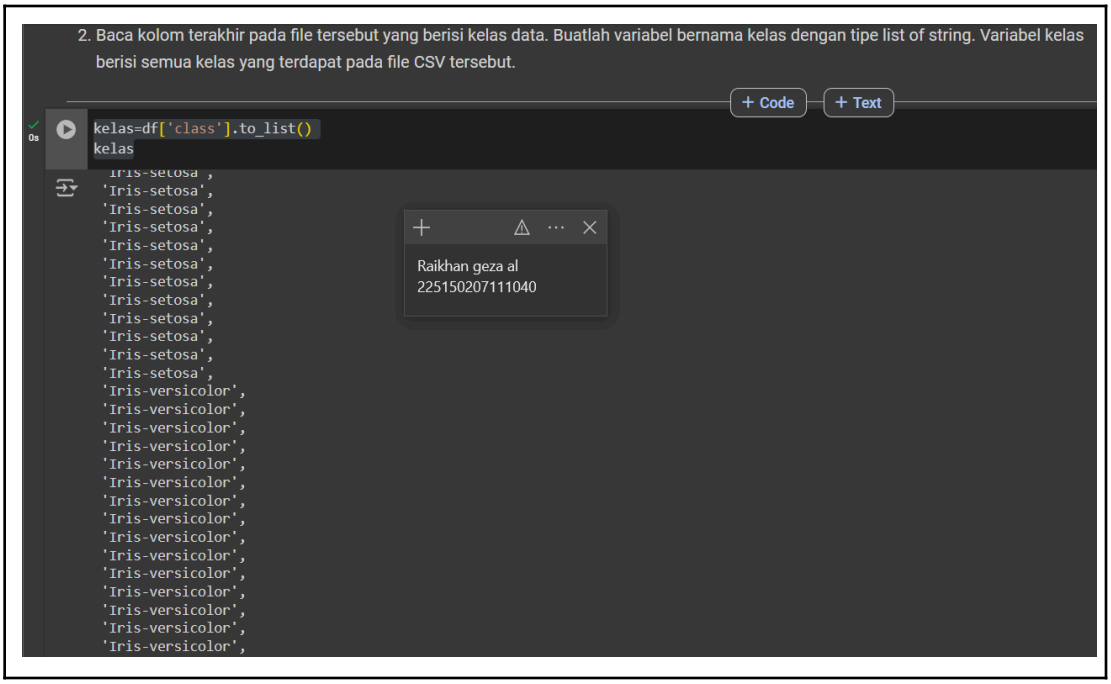
```
import pandas as pd

from google.colab import files
uploaded = files.upload()

df = pd.read_csv("iris_csv.csv")
df.head()
```



	sepallength	sepalwidth	petallength	petalwidth	class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

2. Baca kolom terakhir pada file tersebut yang berisi kelas data. Buatlah variabel bernama kelas dengan tipe list of string. Variabel kelas berisi semua kelas yang terdapat pada file CSV tersebut.

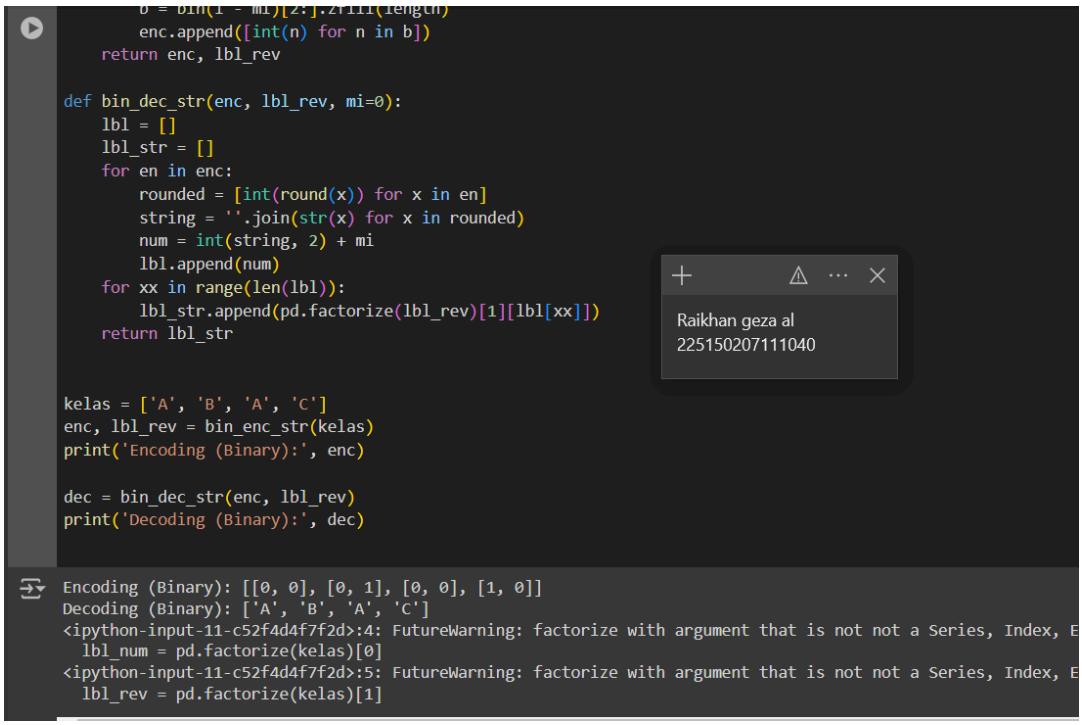
```
kelas=df['class'].to_list()
kelas
```



3. Buatlah fungsi bernama bin_enc_str yang berfungsi untuk melakukan binary encoding pada string. Fungsi ini menerima input berupa list of string dan menghasilkan output berupa representasi binary encoding dari list tersebut. Jangan lupa membuat fungsi decodernya juga dengan nama bin_dec_str

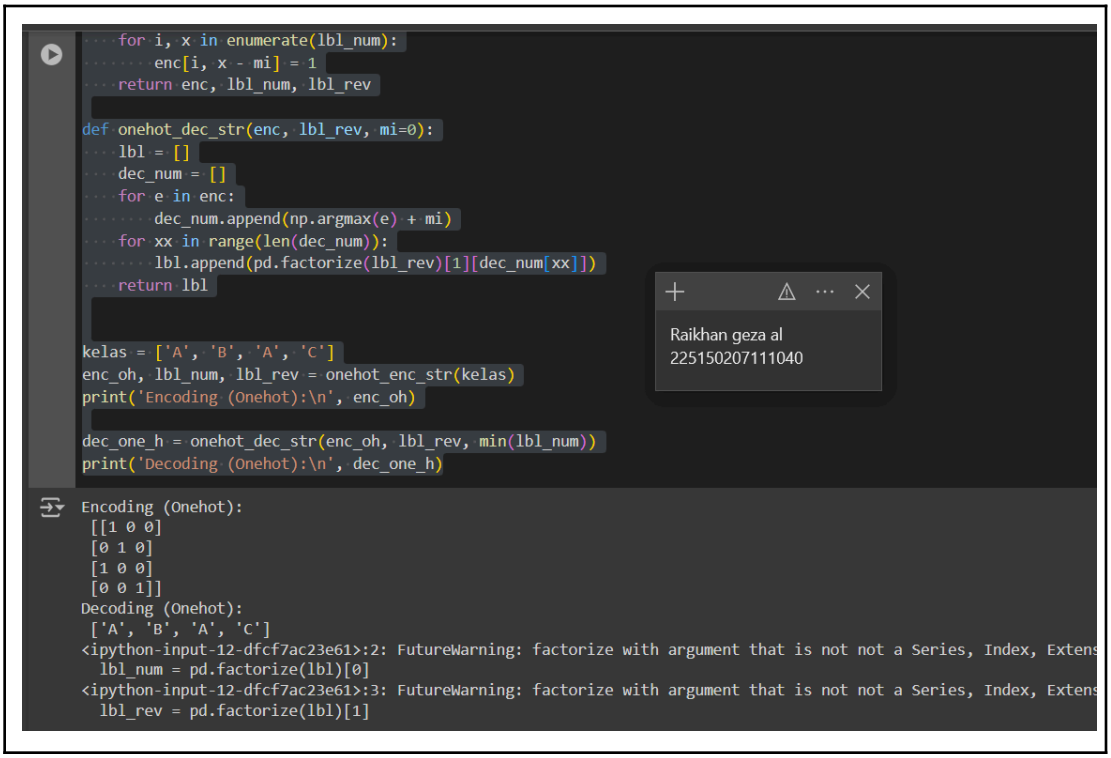
```
import pandas as pd

def bin_enc_str(kelas):
    lbl_num = pd.factorize(kelas)[0]
    lbl_rev = pd.factorize(kelas)[1]
    mi = min(lbl_num)
    length = len(bin(max(lbl_num) - mi + 1)[2:])
    enc = []
    for i in lbl_num:
        b = bin(i - mi)[2:].zfill(length)
        enc.append([int(n) for n in b])
    return enc, lbl_rev

def bin_dec_str(enc, lbl_rev, mi=0):
    lbl = []
    lbl_str = []
    for en in enc:
        rounded = [int(round(x)) for x in en]
        string = ''.join(str(x) for x in rounded)
        num = int(string, 2) + mi
```

```
        lbl.append(num)
    for xx in range(len(lbl)):

lbl_str.append(pd.factorize(lbl_rev)[1][lbl[xx]])
    return lbl_str


kelas = ['A', 'B', 'A', 'C']
enc, lbl_rev = bin_enc_str(kelas)
print('Encoding (Binary):', enc)

dec = bin_dec_str(enc, lbl_rev)
print('Decoding (Binary):', dec)
```



4. Buatlah fungsi bernama onehot_enc_str yang berfungsi untuk melakukan one-hot encoding pada string. Fungsi ini menerima input berupa list of string dan menghasilkan output berupa representasi one-hot encoding dari list tersebut. Jangan lupa membuat fungsi decodernya juga dengan nama onehot_dec_str

```
def onehot_enc_str(lbl, min_val=0):
    lbl_num = pd.factorize(lbl)[0]
    lbl_rev = pd.factorize(lbl)[1]
    mi = min(lbl_num)
    enc = np.full((len(lbl_num), max(lbl_num) - mi +
1), min_val, np.int8)
    for i, x in enumerate(lbl_num):
        enc[i, x - mi] = 1
    return enc, lbl_num, lbl_rev

def onehot_dec_str(enc, lbl_rev, mi=0):
    lbl = []
    dec_num = []
    for e in enc:
        dec_num.append(np.argmax(e) + mi)
    for xx in range(len(dec_num)):

lbl.append(pd.factorize(lbl_rev)[1][dec_num[xx]])
    return lbl


kelas = ['A', 'B', 'A', 'C']
enc_oh, lbl_num, lbl_rev = onehot_enc_str(kelas)
print('Encoding (Onehot):\n', enc_oh)

dec_one_h = onehot_dec_str(enc_oh, lbl_rev,
min(lbl_num))
print('Decoding (Onehot):\n', dec_one_h)
```



D. Kesimpulan

Backpropagation adalah algoritme pelatihan untuk jaringan saraf yang bekerja dengan memperbarui bobot melalui propagasi balik dari kesalahan. Dalam proses ini, kesalahan dihitung di lapisan output dan didistribusikan kembali ke lapisan-lapisan sebelumnya untuk memperbaiki bobot berdasarkan gradien. Agar backpropagation dapat berfungsi dengan benar, encoding diperlukan karena jaringan saraf hanya bekerja dengan data numerik. Label atau data kategorikal harus diubah ke dalam format angka agar jaringan dapat memprosesnya dan menghitung kesalahan dengan tepat.

Beberapa jenis encoding yang umum digunakan meliputi one-hot encoding, di mana setiap kelas diwakili oleh vektor biner dengan satu nilai 1 dan sisanya 0, binary encoding yang mengubah kategori menjadi representasi biner, serta label encoding yang menggantikan setiap kategori dengan angka urut. One-hot encoding sering digunakan untuk klasifikasi multikategori, sementara binary encoding lebih efisien untuk data dengan banyak kategori dan lebih sedikit dimensi.