Soft Computing Praktik

~ ~ Meet 08 ~ ~

Program Studi Informatika
Universitas Teknologi Yogyakarta

Donny Avianto, S.T., M.T.

Neural Network
Klasifikasi Email Spam

Spam Email Dataset

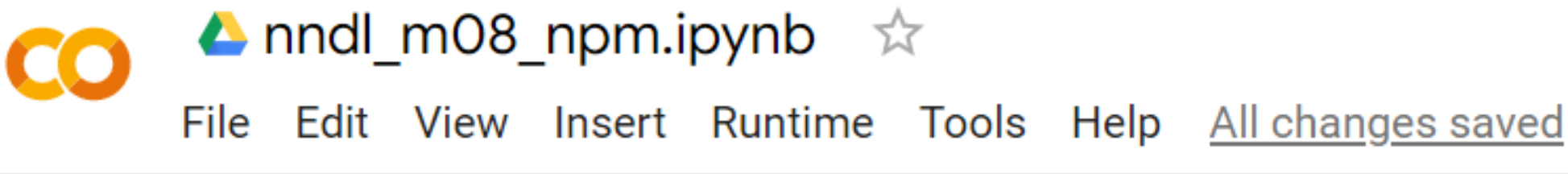
SPAM E-MAIL DATABASE ATTRIBUTES (in .names format)

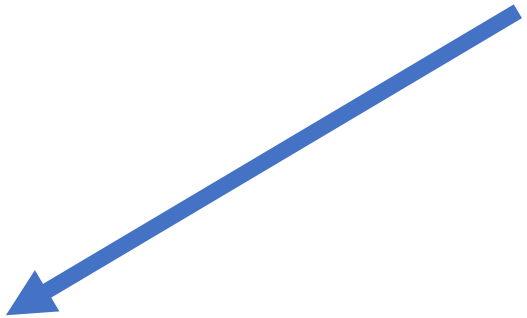
- 48 continuous real [0,100] attributes of type word_freq_WORD
  - = percentage of words in the e-mail that match WORD, i.e. 100 * (number of times the WORD appears in the e-mail) / total number of words in e-mail. A "word" in this case is any string of alphanumeric characters bounded by non-alphanumeric characters or end-of-string.
- 6 continuous real [0,100] attributes of type char_freq_CHAR
  - = percentage of characters in the e-mail that match CHAR, i.e. 100 * (number of CHAR occurences) / total characters in e-mail
- 1 continuous real [1,...] attribute of type capital_run_length_average
  - = average length of uninterrupted sequences of capital letters
- 1 continuous integer [1,...] attribute of type capital_run_length_longest
  - = length of longest uninterrupted sequence of capital letters
- 1 continuous integer [1,...] attribute of type capital_run_length_total
  - = sum of length of uninterrupted sequences of capital letters = total number of capital letters in the e-mail
- 1 nominal {0,1} class attribute of type spam
  - = denotes whether the e-mail was considered spam (1) or not (0), i.e. unsolicited commercial e-mail.

Spam Email Dataset

1 word_freq_make continuous.
2 word_freq_address continuous.
3 word_freq_all continuous.
4 word_freq_3d continuous.
5 word_freq_our continuous.
6 word_freq_over continuous.
7 word_freq_remove continuous.
8 word_freq_internet continuous.
9 word_freq_order continuous.
10 word_freq_mail continuous.
11 word_freq_receive continuous.
12 word_freq_will continuous.
13 word_freq_people continuous.
14 word_freq_report continuous.
15 word_freq_addresses continuous.
16 word_freq_free continuous.
17 word_freq_business continuous.
18 word_freq_email continuous.
19 word_freq_you continuous.
20 word_freq_credit continuous.
21 word_freq_your continuous.
22 word_freq_font continuous.
23 word_freq_000 continuous.
24 word_freq_money continuous.
25 word_freq_hp continuous.
26 word_freq_hpl continuous.
27 word_freq_george continuous.
28 word_freq_650 continuous.
29 word_freq_lab continuous.
30 word_freq_labs continuous.
31 word_freq_telnet continuous.
32 word_freq_857 continuous.
33 word_freq_data continuous.
34 word_freq_415 continuous.
35 word_freq_85 continuous.
36 word_freq_technology continuous.
37 word_freq_1999 continuous.
38 word_freq_parts continuous.
39 word_freq_pm continuous.
40 word_freq_direct continuous.
41 word_freq_cs continuous.
42 word_freq_meeting continuous.
43 word_freq_original continuous.
44 word_freq_project continuous.
45 word_freq_re continuous.
46 word_freq_edu continuous.
47 word_freq_table continuous.
48 word_freq_conference continuous.
1 char_freq_; continuous.
2 char_freq_(continuous.
3 char_freq_[continuous.
4 char_freq_! continuous.
5 char_freq_$ continuous.
6 char_freq_# continuous.
1 capital_run_length_average continuous.
1 capital_run_length_longest continuous.
1 capital_run_length_total continuous.
1 spam (1) or not (0) nominal

Nama file ipynb





+ Code + Text

RAM
Disk

Editing

NPM:

Nama:

```python
import pandas as pd
import numpy as np
import random
import math
import matplotlib.pyplot as plt
```

```python
import io

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

df = pd.read_csv(io.StringIO(filenya['emailspam.csv'].decode('utf-8')))
print(df)
```

```python
#baca data csv
# csv_data = pd.read_csv("emailspam.csv", delimiter=';', header=0)
data = np.array(df) #konversi data csv menjadi array
data = data.astype(float) #konversi data menjadi tipe float
n_data = len(data[:,0]) #menghitung banyaknya data

print('jumlah data:',n_data)

#membaca jumlah feature
n_feature = len(data[0,:]) - 1

print('jumlah feature:',n_feature)
```

```
[ ]
#membagi data: data latih dan uji
rasio_data_latih = 0.7
n_data_latih = int(n_data * rasio_data_latih)
data_latih = data[:n_data_latih,:]
data_uji = data[n_data_latih:,:]
n_data_uji = len(data_uji[:,0])

print('jumlah data latih',n_data_latih)
print('jumlah data uji',n_data_uji)

np.seterr(invalid='ignore')

#normalisasi data latih dalam rentang [0.1, 0.9]
for i in range(1, n_feature + 1):
  data_latih[:,i] = 0.1 + ((data_latih[:,i] - min(data_latih[:,i]))/(max(data_latih[:,i])-min(data_latih[:,i]))) * 0.8

print(data_latih)

#normalisasi data uji dalam rentang [0.1, 0.9]
for i in range(1, n_feature + 1):
  data_uji[:,i] = 0.1 + ((data_uji[:,i] - min(data_uji[:,i]))/(max(data_uji[:,i])-min(data_uji[:,i]))) * 0.8
```

```
[ ]
#inisialisasi parameter jst
n_input = n_feature #jumlah neuron pada input layer
n_hidden = 1 #jumlah neuron pada hidden layer
n_output = 1 #jumlah neuron pada output layer
n_epoch = 1 #jumlah epoch/ iterasi maksimal
alfa = 0 #learning rate

np.random.seed(seed=716)

#inisialisasi bobot MLP dalam rentang [-1, 1]
w = np.random.rand(n_hidden,n_input) * 2 - 1
b1 = np.random.rand(n_hidden) * 2 - 1
v = np.random.rand(n_output, n_hidden) #* 2 - 1
b2 = np.random.rand(n_output) * 2 - 1
```

```
#learning
itr = 0
MSE = np.zeros(n_epoch + 1)
while(itr <= n_epoch):
  print("Epoch ke-" + str(itr))

  for idx_data in range(0, n_data_latih):
    label = data[idx_data,n_feature]
    feature = data_latih[idx_data,0:n_feature]

    #hitung nilai pada hidden layer
    z = np.zeros(n_hidden)
    for i in range(0,n_hidden):
      net = np.sum(feature * w[i]) + b1[i]
      z[i] = 1/(1 + math.exp(-net))

    #hitung nilai pada output layer
    y = np.zeros(n_output)
    f_output = np.zeros(n_output)
    for i in range(0,n_output):
      net = np.sum(z * v[i]) + b2[i]
      y[i] = 1/(1 + math.exp(-net))

    #hitung error pada output layer
    error = label - y

    #hitung Jumlah error
    sum_squared_error = sum(error**2)

```

```
[ ]
30  #hitung faktor koreksi pada output layer
31  for i in range(0, n_output):
32    f_output[i] = error * y[i] * (1 - y[i])
33
34  #hitung perbaikan bobot antara output dan hidden layer
35  delta_v = np.zeros(shape=(n_output, n_hidden))
36  for i in range(0,n_output):
37    delta_v[i,:] = alfa * f_output[i] * z
38
39  #hitung perbaikan bobot BIAS (b2) antara output dan hidden layer
40  delta_b2 = np.zeros(n_output)
41  for i in range(0,n_output):
42    delta_b2[i] = alfa * f_output[i] * 1
43
44  #hitung faktor koreksi pada hidden layer
45  f_hidden = np.zeros(n_hidden)
46  for i in range (0, n_hidden):
47    #langkah 1 - hitung f_hidden_net
48    f_hidden_net = sum(f_output * v[:,i])
49    #langkah 2 - hitung f_hidden
50    f_hidden[i] = f_hidden_net * z[i] * (1 - z[i])
51
52  #hitung perbaikan bobot antara hidden dan input layer
53  delta_w = np.zeros(shape=(n_hidden, n_input))
54  for i in range(n_hidden):
55    delta_w = alfa * f_hidden[i] * feature
56
57  #hitung perbaikan bobot antara hidden dan input layer
58  delta_b1 = np.zeros(n_hidden)
59  for i in range(n_hidden):
60    delta_b1 = alfa * f_hidden[i] * 1
```

```
61
62      #update semua bobot
63      w = w + delta_w
64      b1 = b1 + delta_b1
65      v = v + delta_v
66      b2 = b2 + delta_b2
67    #end for
68    #hitung Mean Squared Error (MSE)
69    MSE[itr] = sum_squared_error / n_data_latih
70    itr += 1
71  #end while
```

[ ]

```
1 print("--------------RESULT---------------")
2 print("Mean Squared Error: " +str(MSE[n_epoch]))
3 #print grafik MSE hasil training
4 plt.title("Mean Squared Error hasil training")
5 plt.plot(MSE)
6 plt.autoscale(enable=True, axis='both', tight=None)
7 plt.show(block=False)
```

Coba jalankan kode program yang barusan Anda ketikkan