NaiveBayes_Homework_A

August 31, 2021

Bài tập: Cho tập dữ liệu về bệnh ung thư: https://www.kaggle.com/uciml/breast-cancer-wisconsin-data

Trong đó là các thông tin về đặc điểm của tế bào đã được ghi nhận thành các thuộc tính (radius_mean, texture_mean, ...), thể hiện dưới dạng bảng. Cùng với đó là các chẩn đoán (diagnosis) liệu tế bào đó có phải là tế bào ung thư hay không.

Hãy xây dựng mô hình Naive Bayes để dự đoán liệu một tế bào với các đặc điểm cho trước có phải là một tế bào ung thư hay không?

- Nếu dùng GColab, cần kết nối với server và drive (nếu đọc/ ghi dữ liệu từ drive)

[4]:
```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

[5]:
```
cd /content/drive/MyDrive/NB_practice/2. Assignment
```

```
/content/drive/MyDrive/NB_practice/2. Assignment
```

- Import các thư viện cần thiết và load dữ liệu từ file

[2]:
```
#Import các thư viện cần thiết
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

[6]:
```
#Load data
data = pd.read_csv("cancer_data.csv")
data.head() #Quan sát dữ liệu cho thấy, tập dữ liệu đã được xử lí khá "sạch"
```

[6]:

	id	diagnosis	…	fractal_dimension_worst	Unnamed: 32
0	842302	M	…	0.11890	NaN
1	842517	M	…	0.08902	NaN
2	84300903	M	…	0.08758	NaN
3	84348301	M	…	0.17300	NaN
4	84358402	M	…	0.07678	NaN

[5 rows x 33 columns]

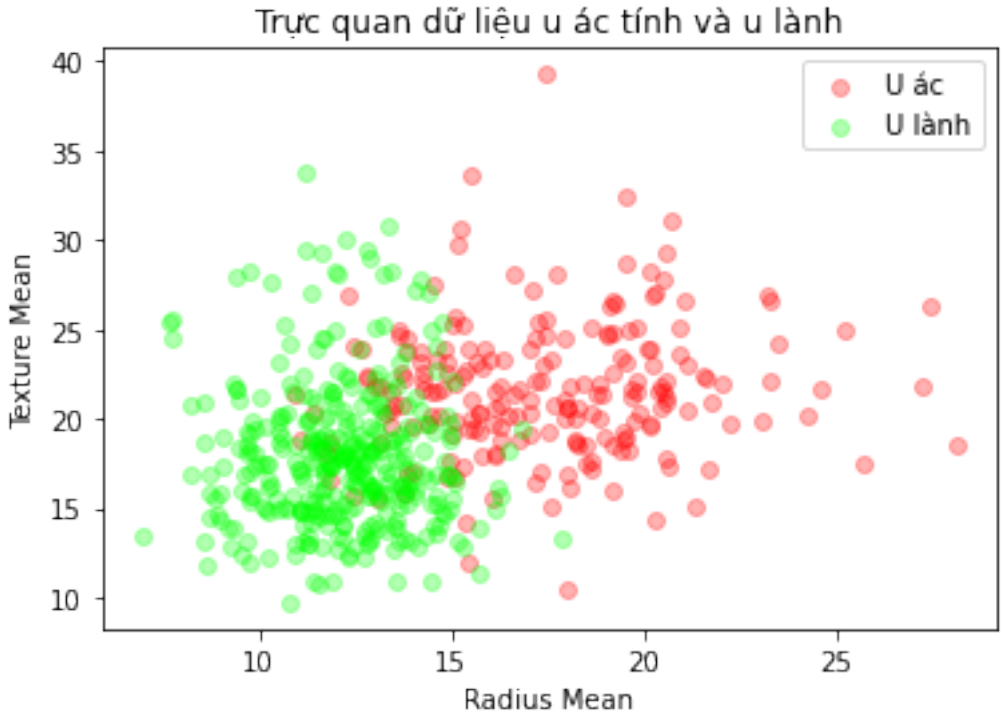
[7]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 569 entries, 0 to 568
Data columns (total 33 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   id                       569 non-null    int64  
 1   diagnosis                569 non-null    object 
 2   radius_mean              569 non-null    float64
 3   texture_mean             569 non-null    float64
 4   perimeter_mean           569 non-null    float64
 5   area_mean                569 non-null    float64
 6   smoothness_mean          569 non-null    float64
 7   compactness_mean         569 non-null    float64
 8   concavity_mean           569 non-null    float64
 9   concave points_mean      569 non-null    float64
 10  symmetry_mean            569 non-null    float64
 11  fractal_dimension_mean   569 non-null    float64
 12  radius_se                569 non-null    float64
 13  texture_se               569 non-null    float64
 14  perimeter_se             569 non-null    float64
 15  area_se                  569 non-null    float64
 16  smoothness_se            569 non-null    float64
 17  compactness_se           569 non-null    float64
 18  concavity_se             569 non-null    float64
 19  concave points_se        569 non-null    float64
 20  symmetry_se              569 non-null    float64
 21  fractal_dimension_se     569 non-null    float64
 22  radius_worst             569 non-null    float64
 23  texture_worst            569 non-null    float64
 24  perimeter_worst          569 non-null    float64
 25  area_worst               569 non-null    float64
 26  smoothness_worst         569 non-null    float64
 27  compactness_worst        569 non-null    float64
 28  concavity_worst          569 non-null    float64
 29  concave points_worst     569 non-null    float64
 30  symmetry_worst           569 non-null    float64
 31  fractal_dimension_worst  569 non-null    float64
 32  Unnamed: 32              0 non-null      float64
dtypes: float64(31), int64(1), object(1)
memory usage: 146.8+ KB
```

[9]:
```
# Loại trường không cần thiết: id, unnamed
data = data.drop(["id", "Unnamed: 32"], axis = 1)
```

[11]:
```
#Trực quan hóa về dữ liệu: u ác tính và u lành:
M = data[data.diagnosis == "M"] # dữ liệu ứng với u ác tính
B = data[data.diagnosis == "B"] # dữ liệu ứng với u lành

plt.title("Trực quan dữ liệu u ác tính và u lành")
plt.xlabel("Radius Mean")
plt.ylabel("Texture Mean")
plt.scatter(M.radius_mean, M.texture_mean, color = "red", label = "U ác", alpha␣
↪= 0.3)
plt.scatter(B.radius_mean, B.texture_mean, color = "lime", label = "U lành",␣
↪alpha = 0.3)
plt.legend()
plt.show()
```



- Tiền xử lí trước khi huấn luyện mô hình:

[13]:
```
# Câu hỏi: Đưa trường chẩn đoán từ dạng chữ thành dạng số
# Code ############
data.diagnosis = [1 if i == "M" else 0 for i in data.diagnosis]
data.head()
#########################################################
```

[13]:
```
   diagnosis  radius_mean  …  symmetry_worst  fractal_dimension_worst
0          0        17.99  …          0.4601                  0.11890
1          0        20.57  …          0.2750                  0.08902
2          0        19.69  …          0.3613                  0.08758
3          0        11.42  …          0.6638                  0.17300
4          0        20.29  …          0.2364                  0.07678

[5 rows x 31 columns]
```

[15]:
```
# Câu hỏi: Tách thuộc tính và nhãn:

# Code ###############
X = data.drop(["diagnosis"], axis=1)
y = data.diagnosis.values
X

##################
```

[15]:
```
     radius_mean  texture_mean  …  symmetry_worst  fractal_dimension_worst
0          17.99         10.38  …          0.4601                  0.11890
1          20.57         17.77  …          0.2750                  0.08902
2          19.69         21.25  …          0.3613                  0.08758
3          11.42         20.38  …          0.6638                  0.17300
4          20.29         14.34  …          0.2364                  0.07678
..           …             …  …             …                        …
564        21.56         22.39  …          0.2060                  0.07115
565        20.13         28.25  …          0.2572                  0.06637
566        16.60         28.08  …          0.2218                  0.07820
567        20.60         29.33  …          0.4087                  0.12400
568         7.76         24.54  …          0.2871                  0.07039

[569 rows x 30 columns]
```

[18]:
```
# Chuẩn hóa các trường của X
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
sc.fit(X)
```

[18]:
```
StandardScaler(copy=True, with_mean=True, with_std=True)
```

[21]:
```
# Câu hỏi: Chia dữ liệu train/test:
from sklearn.model_selection import train_test_split
# Code ###################
x_train, x_test, y_train, y_test = \
  train_test_split(X, y, test_size = 0.3, random_state = 4)
##############################
```

- Xây dựng và đánh giá mô hình Naive Bayes:

[23]:
```
# Câu hỏi: Multinormial NB:

# Code ###################################
from sklearn.naive_bayes import MultinomialNB
model_MNB = MultinomialNB()
model_MNB.fit(x_train, y_train)

print("Multinormial Naive Bayes score: ", model_MNB.score(x_test, y_test))

########################################
```

```
Multinormial Naive Bayes score:  1.0
```

[25]:
```
# Câu hỏi: Gaussian NB:

# Code ########################################
from sklearn.naive_bayes import GaussianNB
model_GNB = GaussianNB(var_smoothing=1e-3)
model_GNB.fit(x_train, y_train)

print("Gaussian Naive Bayes score: ", model_GNB.score(x_test, y_test))

############################################
```

```
Gaussian Naive Bayes score:  1.0
```

[43]:
```
# Câu hỏi: Dùng GNB để dự đoán thử một sample = phần tử đầu tiên của x_test:

# Code ########################################3

X_test = x_test.reset_index(drop=True) #Đánh lại chỉ số để dễ gọi

sample = X_test.iloc[0:1]

print('Label: ', y_test[0])
print('Sample: \n', sample)
print('--------------------------------\n Dự đoán nhãn: ', end='')
model_GNB.predict(sample)[0]

################################################
```

```
Label:  0
Sample:
    radius_mean  texture_mean  …  symmetry_worst  fractal_dimension_worst
0         14.42         16.54  …          0.3053                  0.08764
```

```
[1 rows x 30 columns]
---------------------------------
 Dự đoán nhãn:
```

[43]: 0