Exercise 5: Diabetes.csv ¶

Cho dữ liệu Diabetes.csv

Yêu cầu: đọc dữ liệu về, chuẩn hóa dữ liệu (nếu cần) và áp dụng thuật toán Naive Bayes để thực hiện việc dự đoán khả năng dương tính với bệnh tiểu đường (positive diabete - outputs) dựa trên các biến lâm sàng khác (clinical variables - inputs)

Tạo X_train, X_test, y_train, y_test từ dữ liệu đọc được với tỷ lệ dữ liệu test là 0.2

Áp dụng thuật toán Naive Bayer

Tìm kết quả

Hãy cho biết với những người có pregnant, glucose, pressure, triceps, insulin, mass, pedigree, age lần lượt như sau thì ai có khả năng dương tính với bệnh tiểu đường, ai không?

1. 8, 176, 90, 34, 300, 33.7, 0.467, 58
2. 1, 100, 66, 15, 56, 23.6, 0.666, 26
3. 12, 88, 74, 40, 54, 35.3, 0.378, 48

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
```

In [2]:
```
# import some data to play with
data = pd.read_csv("Diabetes.csv")
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 10 columns):
ID          768 non-null int64
pregnant    768 non-null int64
glucose     763 non-null float64
pressure    733 non-null float64
triceps     541 non-null float64
insulin     394 non-null float64
mass        757 non-null float64
pedigree    768 non-null float64
age         768 non-null int64
diabetes    768 non-null object
dtypes: float64(6), int64(3), object(1)
memory usage: 60.1+ KB
```

In [3]:
```
data = data.interpolate()
```

In [4]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 10 columns):
ID          768 non-null int64
pregnant    768 non-null int64
glucose     768 non-null float64
pressure    768 non-null float64
triceps     768 non-null float64
insulin     765 non-null float64
mass        768 non-null float64
pedigree    768 non-null float64
age         768 non-null int64
diabetes    768 non-null object
dtypes: float64(6), int64(3), object(1)
memory usage: 60.1+ KB
```

In [5]:
```
data = data.dropna(axis=0)
```

In [6]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 765 entries, 3 to 767
Data columns (total 10 columns):
ID          765 non-null int64
pregnant    765 non-null int64
glucose     765 non-null float64
pressure    765 non-null float64
triceps     765 non-null float64
insulin     765 non-null float64
mass        765 non-null float64
pedigree    765 non-null float64
age         765 non-null int64
diabetes    765 non-null object
dtypes: float64(6), int64(3), object(1)
memory usage: 65.7+ KB
```

In [7]:
```
data.head()
```

Out[7]:

	ID	pregnant	glucose	pressure	triceps	insulin	mass	pedigree	age	diabetes
3	4	1	89.0	66.0	23.0	94.0	28.1	0.167	21	neg
4	5	0	137.0	40.0	35.0	168.0	43.1	2.288	33	pos
5	6	5	116.0	74.0	33.5	128.0	25.6	0.201	30	neg
6	7	3	78.0	50.0	32.0	88.0	31.0	0.248	26	pos
7	8	10	115.0	60.0	38.5	315.5	35.3	0.134	29	neg

In [8]:
```
data.tail()
```

Out[8]:

	ID	pregnant	glucose	pressure	triceps	insulin	mass	pedigree	age	diabetes
763	764	10	101.0	76.0	48.0	180.0	32.9	0.171	63	neg
764	765	2	122.0	70.0	27.0	146.0	36.8	0.340	27	neg
765	766	5	121.0	72.0	23.0	112.0	26.2	0.245	30	neg
766	767	1	126.0	60.0	27.0	112.0	30.1	0.349	47	pos
767	768	1	93.0	70.0	31.0	112.0	30.4	0.315	23	neg

In [9]:
```
# chuẩn hóa dữ liệu
data_class = {'neg':0, 'pos':1}
data['diabetes'] = [data_class[i] for i in data.diabetes]
# hoặc dùng
# df.replace("neg", 0)
# df.replace("pos", 1)
data.head()
```

Out[9]:

	ID	pregnant	glucose	pressure	triceps	insulin	mass	pedigree	age	diabetes
3	4	1	89.0	66.0	23.0	94.0	28.1	0.167	21	0
4	5	0	137.0	40.0	35.0	168.0	43.1	2.288	33	1
5	6	5	116.0	74.0	33.5	128.0	25.6	0.201	30	0
6	7	3	78.0	50.0	32.0	88.0	31.0	0.248	26	1
7	8	10	115.0	60.0	38.5	315.5	35.3	0.134	29	0

In [10]:
```
X = data.drop(['ID', 'diabetes'], axis=1)
y = data.diabetes
```

In [11]:
```
X.tail()
```

Out[11]:

	pregnant	glucose	pressure	triceps	insulin	mass	pedigree	age
763	10	101.0	76.0	48.0	180.0	32.9	0.171	63
764	2	122.0	70.0	27.0	146.0	36.8	0.340	27
765	5	121.0	72.0	23.0	112.0	26.2	0.245	30
766	1	126.0	60.0	27.0	112.0	30.1	0.349	47
767	1	93.0	70.0	31.0	112.0	30.4	0.315	23

In [12]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)
```

In [13]:
```
from sklearn.naive_bayes import GaussianNB
```

In [14]:
```
#Create a Gaussian Classifier
model = GaussianNB()
# Train the model using the training sets
model.fit(X_train, y_train)
```

Out[14]:
```
GaussianNB(priors=None)
```

In [15]:
```
print('score Scikit learn: ', model.score(X_test,y_test))
```

```
score Scikit learn:  0.7647058823529411
```

In [16]:
```
y_pred = model.predict(X_test)
```

In [17]:
```
from sklearn.metrics import accuracy_score
# Kiểm tra độ chính xác
print("Accuracy is ", accuracy_score(y_test,y_pred)*100,"%")
```

```
Accuracy is  76.47058823529412 %
```

In [18]:
```
X_test_new = [[8, 176, 90, 34, 300, 33.7, 0.467, 58], [1, 100, 66, 15, 56, 23.6, 0
y_pred_new = model.predict(X_test_new)
y_pred_new
```

Out[18]:
```
array([1, 0, 1], dtype=int64)
```

In [ ]: