

Thực hành Nguyên Lý Máy Học

Buổi 2: Bayes thơ ngây

Mục tiêu:

- củng cố lý thuyết và cài đặt giải thuật Bayes thơ ngây
- Sử dụng hàm có sẵn của Python để cài đặt giải thuật Bayes thơ ngây.
- Kiểm thử và đánh giá

A. HƯỚNG DẪN THỰC HÀNH

1. Hiểu nguyên lý hoạt động của giải thuật xây dựng mô hình Bayes ngây thơ

- Đọc dữ liệu từ file: "play_tennis.csv"

```
import pandas as pd
dt2 = pd.read_csv("play_tennis.csv")
dt2
```

	Outlook	Temp	Humidity	Windy	Play
0	Sunny	Hot	High	False	No
1	Sunny	Hot	High	True	No
2	Overcast	Hot	High	False	Yes
3	Rainy	Mild	High	False	Yes
4	Rainy	Cool	Normal	False	Yes
5	Rainy	Cool	Normal	True	No
6	Overcast	Cool	Normal	True	Yes
7	Sunny	Mild	High	False	No
8	Sunny	Cool	Normal	False	Yes
9	Rainy	Mild	Normal	False	Yes
10	Sunny	Mild	Normal	True	Yes
11	Overcast	Mild	High	True	Yes
12	Overcast	Hot	Normal	False	Yes
13	Rainy	Mild	High	True	No

- Tính xác suất có điều kiện $P(\text{Outlook}/\text{Play} = \text{"yes"})$. Bảng phân phối xác suất có dạng

Outlook/Play = Yes	sunny	overcast	rainny
P			

Gợi ý :

- Tìm giá trị Outlook của các phần tử có Play = "Yes"

dt.Outlook[dt.Play=="Yes"]

Hoặc

PlayYes = dt[dt.Play=="Yes"]

PlayYes.Outlook

- Thống kê tần số xuất hiện của các giá trị, chuẩn hoá cho tổng bằng 1 => bảng phân phối xác suất.

dtO = dt.Outlook[dt.Play=="Yes"]

P1_1 = dtO.value_counts() # số lần xuất hiện của mỗi giá trị của thuộc tính Outlook

P1_1 = P1_1/dtO.count()

P1.1 sẽ lưu giữ bảng phân phối xác suất có điều kiện của Outlook khi biết giá trị Play = yes

```
>>> P1_1
Overcast    0.444444
Rainy       0.333333
Sunny       0.222222
Name: Outlook, dtype: float64
```

- c. Tương tự như trên tính, xác suất $P(\text{Outlook}/\text{Play} = \text{"no"})$ lưu vào biến P1.2
- d. Lặp lại các bước a,b,c cho cột Temperature gán vào các biến: P2.1, P2.2

```
dtTy = dt.Temp[dt.Play=="Yes"]
P2_1 = dtTy.value_counts()/dtTy.count()
dtTn = dt.Temp[dt.Play=="No"]
P2_2 = dtTn.value_counts()/dtTn.count()
P2_1
P2_2
```

- e. Lặp lại các bước a,b,c cho cột Humidity gán vào các biến: P3.1, P3.2
- f. Lặp lại các bước a,b,c cho cột Windy gán vào các biến: P4.1, P4.2
- g. Tính xác suất $P(\text{Play})$, lưu vào biến P

```
Play = dt.Play.value_counts()/dt.Play.count()
Play
```

Bước 2: Dự báo nhãn cho dữ liệu

Sử dụng các bảng xác suất trên dự báo nhãn cho dữ liệu

X = ("rainy", "cool", "high", "false")

```
P_yes = P1_1[1]*P2_1[1]*P3_1[1]*P4_1[0]*Play[0]
P_no = P1_2[1]*P2_2[1]*P3_2[1]*P4_2[0]*Play[1]

Hoac

P_yes = P1_1.Rainy*P2_1.Cool*P3_1.High*P4_1.F*Play.Yes
P_no = P1_2.Rainy*P2_2.Cool*P3_2.High*P4_2.F*Play.No

P_yes
P_no
PY = P_yes/(P_no+P_yes)
PN = P_no/(P_no+P_yes)
PY
PN
```

2. Sử dụng thư viện scikit-learn của Python

a. Dữ liệu Iris

Xét bài toán phân loại hoa IRIS dựa trên thông tin về kích thước của cánh hoa và đài hoa. Tập dữ liệu này có 150 phần tử, mỗi loại hoa có 50 phần tử.



Tập dữ liệu này có thể download từ trang UCI (<https://archive.ics.uci.edu/ml/datasets/iris>) rồi đọc dữ liệu bằng lệnh `read_csv` của thư viện **Pandas** như đã hướng dẫn ở buổi thực hành 2 hoặc có thể nạp dữ liệu có sẵn bởi thư viện **Sklearn**.

Trực tiếp bởi thư viện

```
from sklearn import datasets
iris = datasets.load_iris()
X = iris.data
y = iris.target
```

- b. Sử dụng sklearn để xây dựng mô hình bayes thơ ngây
- Dữ liệu kiểu số => giả sử các thuộc tính có phân phối *Gaussian*
Hàm mật độ xác suất được tính bởi công thức

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

Trong đó:

$$\mu = \frac{1}{n} \sum_{i=1}^n x_i \quad \sigma^2 = \frac{1}{n-1} \sum_{i=1}^n (x_i - \mu)^2$$

- Sklearn cung cấp sẵn hàm để tính mật độ xác suất theo phân phối Gaussian cho dữ liệu kiểu liên tục cũng như Multinomial (phân loại văn bản), BernoulliNB,...

```
from sklearn.naive_bayes import GaussianNB
from sklearn.naive_bayes import MultinomialNB
```

- Sử dụng hàm `train_test_split()` để phân chia dữ liệu theo nghi thức hold-out, xây dựng mô hình theo phân phối Gaussian và so sánh kết quả dự đoán so với kết quả thực tế.

```
# Phân chia dữ liệu thành tập test và train
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
#Xây dựng mô hình dựa trên phân phối xác suất tuân theo Gaussian
model = GaussianNB()
model.fit(X_train, y_train)
print(model)
# dự đoán
thucte = y_test
dubao = model.predict(X_test)
thucte
dubao
```

- Sử dụng hàm `confusion_matrix()` để đánh giá giải thuật

```
from sklearn.metrics import confusion_matrix
cnf_matrix_gnb = confusion_matrix(thucte, dubao)
print(cnf_matrix_gnb)
[[16  0  0]
 [ 0 18  0]
 [ 0  0 11]]
```

- c. Sử dụng nghi thức k-fold để phân chia tập dữ liệu “iris” với k=5 với hàm Kfold

```
class sklearn.model_selection
    KFold(n_splits=3, shuffle=False, random_state=None)
```

- ***n_splits*** : int, default=3
Number of folds. Must be at least 2.
- ***shuffle*** : boolean, optional
Whether to shuffle the data before splitting into batches.
- ***random_state*** : int, RandomState instance or None, optional, default=None
If int, random_state is the seed used by the random number generator; If RandomState instance, random_state is the random number generator; If None, the random number generator is the RandomState instance used by `np.random`. Used when `shuffle == True`.

```
from sklearn.model_selection import KFold
kf= KFold(n_splits=15) # chia tập dữ liệu thành 15 phần
```

```
for train_index, test_index in kf.split(X):
    # print("TRAIN:", train_index, "TEST:", test_index)
    X_train, X_test = X.iloc[train_index,], X.iloc[test_index,]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]
    # print ("X_test", X_test) # In thuộc tính của dữ liệu kiểm tra
    print("=====")
```