

Thực hành Nguyên Lý Máy Học

Buổi 2: Giải thuật cây quyết định

Mục tiêu:

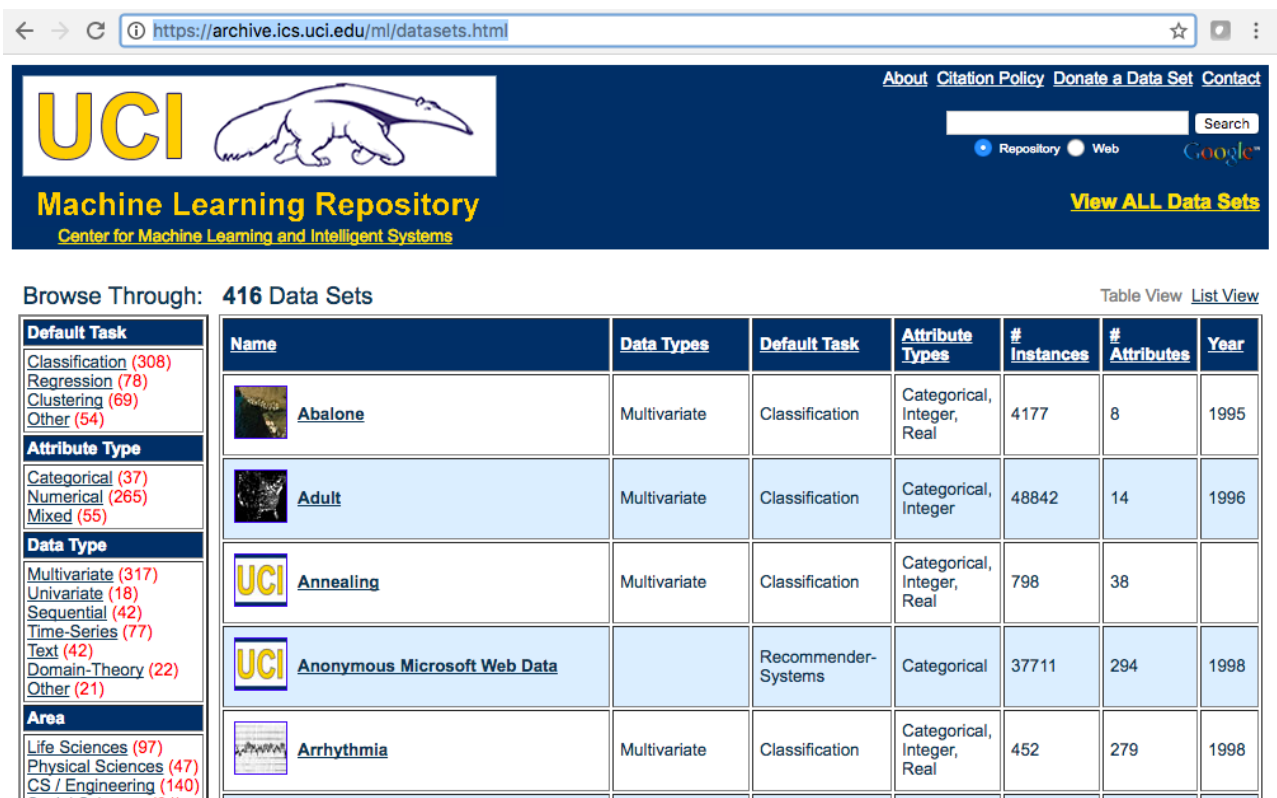
- củng cố lý thuyết và cài đặt giải thuật cây quyết định
- Kiểm thử và đánh giá theo nghi thức hold-out

1. HƯỚNG DẪN THỰC HÀNH






- Cài đặt cây quyết định ID3
- Cách cài đặt một số thư viện cần thiết
 - o Cài đặt một số thư viện phục vụ cho bài thực hành: pandas, sklearn
 - **pip3 install pandas** // đọc file csv
 - **pip3 install sklearn**

Trang web lưu trữ các tập dữ liệu sử dụng trong quá trình thực hành

<https://archive.ics.uci.edu/ml/datasets.html>



The screenshot shows the UCI Machine Learning Repository website. The header includes the UCI logo, a search bar, and navigation links. The main content area displays a table of datasets with columns: Name, Data Types, Default Task, Attribute Types, # Instances, # Attributes, and Year. A sidebar on the left provides filters for Default Task, Attribute Type, Data Type, and Area.

Name	Data Types	Default Task	Attribute Types	# Instances	# Attributes	Year
 Abalone	Multivariate	Classification	Categorical, Integer, Real	4177	8	1995
 Adult	Multivariate	Classification	Categorical, Integer	48842	14	1996
 Annealing	Multivariate	Classification	Categorical, Integer, Real	798	38	
 Anonymous Microsoft Web Data		Recommender-Systems	Categorical	37711	294	1998
 Arrhythmia	Multivariate	Classification	Categorical, Integer, Real	452	279	1998

A. Hướng dẫn cài đặt cây quyết định ID3

Cây quyết định ID3 được xây dựng trên tập dữ liệu play_tennis.csv (file đính kèm)

Các thư viện cần thiết:

```
11
12 import pandas as pd
13 import numpy as np
14 from pprint import pprint
15
```

Load dữ liệu từ file play_tennis.csv

```
23 #Load dữ liệu từ file play_tennis.csv
24 dataset = pd.read_csv('play_tennis.csv',
25                        names=['day','outlook','temp','humidity','wind','play'])
26
27
```

Hàm entropy dùng để tính toán giá trị entropy của cột target_col, được xây dựng như sau

$$\text{Info}(D) = \text{entropy}(p_1, p_2, \dots, p_n) = -p_1 \log p_1 - p_2 \log p_2 - \dots - p_n \log p_n$$

Hàm tính entropy

```
32 #Hàm tính entropy
33 #target_col là cột nhãn
34
35 def entropy(target_col):
36
37     #đếm số lần xuất hiện của các nhãn lưu vào 2 mảng
38     #elements: mảng chứa giá trị lớp
39     #counts: mảng đếm số lần xuất hiện của các phần tử trong elements
40     elements, counts = np.unique(target_col, return_counts = True)
41     en = 0
42     #vòng lặp chạy tuần tự các phần tử trong mảng elements
43     for i in range(len(elements)):
44         #xác suất xuất hiện nhãn i
45         pi = counts[i]/np.sum(counts)
46         #áp dụng công thức tính entropy
47         temp = - pi*np.log2(pi)
48         en = en + temp
49     return en
50
51
```

Thông tin bổ sung về hàm numpy.unique()

- <https://docs.scipy.org/doc/numpy/reference/generated/numpy.unique.html>
- `numpy.unique(ar, return_index=False, return_inverse=False, return_counts=False, axis=None)`
return_counts : *bool, optional*
If True, also return the number of times each unique item appears in *ar*.

Hàm tính độ lợi thông tin InfoGain nhận vào đối số “data” là tập dữ liệu mẫu, “split_attribute_name” là thuộc tính cần đo độ lợi thông tin, “target_name” là tên của cột nhãn.

$$\text{Info}_A(D) = D_1/D * \text{Info}(D_1) + D_2/D * \text{Info}(D_2) + \dots + D_v/D * \text{Info}(D_v)$$

```

#Hàm tính độ lợi thông tin, Information Gain
def InfoGain(data,split_attribute_name,target_name):

    #Tính entropy của dữ liệu trước khi phân hoạch
    total_entropy = entropy(data[target_name])

    #thống kê giá trị và số lượng mẫu tin cho các thuộc tính phân hoạch
    vals,counts= np.unique(data[split_attribute_name],return_counts=True)
    print(vals)
    print(counts)

    # tổng số lượng phần tử của tập dữ liệu
    total_elements = np.sum(counts)
    print(total_elements)

    #Tính entropy cho thuộc tính split_attribute_name
    Weighted_Entropy = 0
    for i in range(len(vals)): # duyệt qua các giá trị khác nhau của thuộc tính đang xét

        # xét giá trị i (vals[i]) của thuộc tính (split_attribute_name) đang xét
        print(vals[i])
        print(split_attribute_name)

        # tính trọng số của giá trị thứ i của thuộc tính đang xét Di/D
        Weighted_Elements = (counts[i]/total_elements)
        print(Weighted_Elements)

        # Trích các dòng dữ liệu chứa giá trị thuộc tính vals[i] => Di
        dt_split_attribute_vals = data[data[split_attribute_name] == vals[i]]
        print(dt_split_attribute_vals)

        # tính entropy của dữ liệu vừa Di vừa trích
        Entropy_Elements = entropy(dt_split_attribute_vals[target_name])

        # tính độ hỗn loạn thông tin sau khi phân hoạch bởi thuộc tính split_attribute_name
        Weighted_Entropy = Weighted_Entropy + Weighted_Elements*Entropy_Elements
        print(Weighted_Entropy)

    #Tính information gain
    Information_Gain = total_entropy - Weighted_Entropy
    print(Information_Gain)
    return Information_Gain

InfoGain(training_data, "outlook", "play")

```

Hàm ID3 dùng xây dựng cây quyết định với các đối số lần lượt là:

- data: chứa dữ liệu tập huấn (dataset), tham số sẽ thay đổi trong quá trình phân hoạch
- originaldata: tập dữ liệu tập huấn
- target_attribute_name: tên thuộc tính nhãn
- parent_node_class: lưu nút cha khi phân hoạch, chọn nhãn có số phần tử xuất hiện nhiều nhất (đối với trường hợp dữ liệu không thuần nhất)
- features: tập thuộc tính được xét trong quá trình phân hoạch

```

84 def ID3(data,originaldata,features,target_attribute_name,parent_node_class):
85     #Nếu dữ liệu thuần nhất thì trả về kết quả là kiểu dữ liệu đó
86     if len(np.unique(data[target_attribute_name])) <= 1:
87         print('Dữ liệu thuần nhất, trả về nút lá')
88         return np.unique(data[target_attribute_name])[0]
89     #nếu tập features rỗng thì trả về parent_node_class
90     #trường hợp dữ liệu không thuần nhất và đã xét hết các thuộc tính
91     #sẽ trả về nhãn có tần số xuất hiện nhiều nhất
92     elif len(features) ==0:
93         #return parent_node_class
94         return 0
95     else:
96         parent_node_class = np.unique(data[target_attribute_name])\
97             [np.argmax(np.unique(data[target_attribute_name],\
98                 return_counts=True)[1])]
99         print('parent_node_class:', parent_node_class)
101     #Chọn thuộc tính phân hoạch dữ liệu tốt nhất, có IG lớn nhất
102     print('tập features: ', features)
103     item_values = [InfoGain(data,feature,target_attribute_name) \
104         for feature in features]#tập các giá trị IG của các thuộc tính
105     print("item_value: ", item_values)
106     best_feature_index = np.argmax(item_values)
107     best_feature = features[best_feature_index]
108     print("best_feature", best_feature)
109     #Tạo cây, nút gốc có IG lớn nhất
110     tree = {best_feature:{}}
111     #xóa thuộc tính có IG lớn nhất trong tập features
112     features = [i for i in features if i != best_feature]
113     #Phát triển nhánh từ nút gốc ứng với mỗi giá trị thuộc tính gốc
114     print('tập thuộc tính sau khi loại là: ', features)
115     for value in np.unique(data[best_feature]):
116         print('Xét giá trị: ', value, '\n')
117         #Lọc dữ liệu theo từng giá trị thuộc tính
118         sub_data = data.where(data[best_feature] == value).dropna()
119         #Gọi đệ quy ID3 cho dataset còn lại
120         print("subdata là: \n", sub_data)
121         print("parent_node_class là: ", parent_node_class , "\n")
122         subtree = ID3(sub_data,dataset,features,target_attribute_name,
123             parent_node_class)
124         #Thêm cây con vào tree
125         tree[best_feature][value] = subtree
126     return(tree)
127

```

np.unique(data[target_attribute_name]) \

[np.argmax(np.unique(data[target_attribute_name]

<https://docs.scipy.org/doc/numpy/reference/generated/numpy.argmax.html>

numpy.argmax(a, axis=None, out=None)

Returns the indices of the maximum values along an axis.

Kết quả cây quyết định ID3:

```

176 #####
177 #bỏ dòng đầu tiên là tên cột, lấy từ dòng 1 đến 14
178 training_data = dataset.iloc[1:15].reset_index(drop=True)
179 #bỏ cột day
180 tree = ID3(training_data, training_data, training_data.columns[1:-1], 'play', None)
181 pprint(tree)
182

```

```

{'outlook': {'Overcast': 'Yes',
             'Rain': {'wind': {'Strong': 'No', 'Weak': 'Yes'}},
             'Sunny': {'humidity': {'High': 'No', 'Normal': 'Yes'}}}}

```

B. Bài toán phân lớp – chỉ số Gini (sử dụng thư viện Sklearn)

• Tập 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. Dữ liệu có 4 thuộc tính (sepal length, sepal width, petal length, petal width) và 3 lớp (3 loại hoa Iris: Setosa, Versicolour, Virginica)



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** hoặc có thể nạp dữ liệu có sẵn từ thư viện **Sklearn**

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5.0	3.4	1.5	0.2	setosa

▪ Sử dụng tập dữ liệu có sẵn "iris"

```

#Lay file iris truc tiep tu sklearn
from sklearn.datasets import load_iris
iris_dt = load_iris()
iris_dt.data[1:5] # thuoc tinh cua tap iris
iris_dt.target[1:5] #gia tri cua nhan /class

```


- *Phân chia tập dữ liệu để xây dựng mô hình và kiểm tra theo nghi thức Hold-out*

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(iris_dt.data, iris_dt.target, test_size=1/3.0,
random_state=5)

X_train[1:6]
X_train[1:6,1:3]
y_train[1:6]
X_test[6:10]
y_test[6:10]
```

- *Xây dựng mô hình cây quyết định dựa trên chỉ số Gini với độ sâu của cây bằng 3, nút nhánh ít nhất có 5 phần tử.*

```
# Xây dựng mô hình cây quyết định dựa trên chỉ số Gini
from sklearn.tree import DecisionTreeClassifier
clf_gini = DecisionTreeClassifier(criterion = "gini", random_state = 100, max_depth=3, min_samples_leaf=5)
clf_gini.fit(X_train, y_train)
```

- *Dự đoán nhãn cho các phần tử trong tập kiểm tra*

```
# dự đoán

y_pred = clf_gini.predict(X_test)
y_test
clf_gini.predict([[4, 4, 3, 3]])
```

- *Tính độ chính xác cho giá trị dự đoán của phần tử trong tập kiểm tra*

```
# tính độ chính xác
from sklearn.metrics import accuracy_score
print ("Accuracy is ", accuracy_score(y_test,y_pred)*100)
```

Kết quả thu được
Accuracy is 96.0

- *Tính độ chính xác cho giá trị dự đoán thông qua ma trận con*

```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test, y_pred, labels=[2,0,1])
```

Kết quả thu được

```
[>>> confusion_matrix(y_test, y_pred, labels=[2,0,1])
array([[15,  0,  2],
       [ 0, 16,  0],
       [ 1,  0, 16]])
```

C. Một số cách đọc dữ liệu đầu vào

1. Đọc dữ liệu từ file bằng thư viện panda

Hướng dẫn đọc dữ liệu từ file bằng thư viện “pandas” và truy xuất dữ liệu theo số lượng dòng cũng như theo chỉ số; xác định độ lớn của tập dữ liệu (số record)

```
import pandas as pd
dt5 = pd.read_csv("iris_data.csv")
dt5[1:5]
len(dt5)
dt5.petalLength[1:5]
```

2. Tạo các biến lưu trữ dữ liệu

Tạo dữ liệu gồm 2 thuộc tính x_1 , x_2 và nhãn đặt ở biến y

STT	X1	X2	Nhãn
1.	0	0	0
2.	1	0	0
3.	1	1	0
4.	2	1	1
5.	2	1	1
6.	2	0	0

```
X = [ [0, 0],
      [1, 0],
      [1, 1],
      [2, 1],
      [2, 1],
      [2, 0] ]
Y = [0, 0, 0, 1, 1, 0]
```

2. BÀI TẬP

Xây dựng cây quyết định sử dụng chỉ số GINI và dự đoán nhãn (không sử dụng thư viện Sklearn)

- Đọc dữ liệu từ tập dữ liệu đánh giá chất lượng rượu vang trắng trên trang UCI <https://archive.ics.uci.edu/ml/datasets/wine+quality>

Wine Quality Data Set

Download: [Data Folder](#), [Data Set Description](#)

Abstract: Two datasets are included, related to red and white vinho verde wine samples, from the north of Portugal. The goal is to model wine quality based on physicochemical tests (see [Cortez et al., 2009], [Web Link]).



Data Set Characteristics:	Multivariate	Number of Instances:	4898	Area:	Business
Attribute Characteristics:	Real	Number of Attributes:	12	Date Donated	2009-10-07
Associated Tasks:	Classification, Regression	Missing Values?	N/A	Number of Web Hits:	578954

Index of /ml/machine-learning-databases/wine-quality

Name	Last modified	Size	Description
 Parent Directory			-
 winequality-red.csv	16-Oct-2009 14:36	82K	
 winequality-white.csv	16-Oct-2009 14:36	258K	
 winequality.names	21-Oct-2009 11:00	3.2K	

Apache/2.2.15 (CentOS) Server at archive.ics.uci.edu Port 443

- b. Tập dữ liệu trên có bao nhiêu phần tử, Có bao nhiêu nhãn? Ghi chú kết quả trong file code sau câu lệnh.
 - c. Sử dụng **8 phần để** xây dựng mô hình và sử dụng **2 phần** để thực hiện đánh giá mô hình. Anh/chị ghi chú lại số lượng phần tử trong tập test và nhãn của các phần tử thuộc tập test trong file code
 - d. Xây dựng mô hình cây quyết định dựa trên tập dữ liệu học tạo ra ở bước c (tự xây dựng cây quyết định sử dụng chỉ số GINI, không dùng thư viện Sklearn)
 - e. Đánh giá độ chính xác tổng thể và độ chính xác của từng lớp cho toàn bộ dữ liệu trong tập test
 - f. Đánh giá độ chính xác tổng thể và độ chính xác của từng lớp cho 6 phần tử đầu tiên trong tập test
2. Cho tập dữ liệu gồm 5 phần tử như bảng bên dưới

STT	Chiều cao	Độ dài mái tóc	Giọng nói	Nhãn
1.	180	15	0	Nam
2.	167	42	1	Nữ
3.	136	35	1	Nữ
4.	174	15	0	Nam
5.	141	28	1	Nữ

- Sử dụng 5 phần tử trong tập dữ liệu trên để xây dựng mô hình cây quyết định dựa vào **chỉ số độ lợi thông tin** với thuộc tính: chiều cao, độ dài tóc và giọng nói để dự đoán nhãn là nam giới hay nữ giới.
- Dự báo phần tử mới tới có thông tin chiều cao=135, độ dài mái tóc = 39 và giọng nói có giá trị là 1 thì người này là nam hay nữ?

Hạn nộp bài: thứ 4 (07/4/21) lúc 23h55'