Nguyễn Khắc Sơn – MSSV: 21085691
Bài tập về nhà – Buổi số 5

Bài tập 3: Các em có thể dùng các phương pháp phân lớp khác nhau để so sánh các kết quả.

Câu 1:

Khi sử dụng thuật toán để xây dựng cây quyết định, giả sử em tiến hành phân chia các nút cho tới khi có một cây quyết định rất phức tạp với nhiều nút, nhiều nhánh và nhiều nút lá với chỉ rất ít các điểm dữ liệu (vấn đề quá khớp xảy ra).

Dựa trên những kiến thức đã học trên lớp và đọc thêm tài liệu ở nhà, em hãy trình bày và giải thích những cách có thể giải quyết được vấn đề quá khớp (overfitting) gặp phải khi xây dựng các cây quyết định.

Các giải pháp	
1. Giới hạn độ sâu (Pruning)	- Một trong những cách trực tiếp giảm overfitting - Quy định độ sâu tối đa của cây khi cây đang được xây dựng -> ngăn cây tiếp tục phân tách dữ liệu dựa trên các chi tiết nhỏ, không cần thiết.
2. Sử dụng số lượng tối thiểu các mẫu trong một lá (Minimum Samples per Leaf)	- Là việc quy định số lượng các mẫu nhỏ nhất mà một lá trong cây quyết định có thể chứa. Nếu số mẫu quá ít có thể dẫn đến overfitting - Tăng số lượng mẫu tối thiểu cho mỗi lá để giảm bớt sự phân tách không cần thiết
3. Kỹ thuật 'Cost Complexity Pruning'	- Kỹ thuật cắt tỉa cây sau khi được xây dựng hoàn chỉnh. Cây sẽ được cắt tỉa tùy theo mức độ phức tạp của nó, chỉ giữ lại các nhánh quan trọng cho việc dự đoán
4. Phương pháp 'Ensemble Learning'	- Bao gồm phương pháp như Random Forest hoặc Gradient Boosting, giúp giảm overfitting bằng cách kết hợp nhiều cây quyết định
5. Tăng cường kích thước tập huấn luyện (Data Augmentation)	- Tăng cường kích thước và độ đa dạng của tập huấn luyện bằng cách thu thập thêm dữ liệu hoặc sử dụng các kỹ thuật tăng cường dữ liệu, giúp mô hình có thêm thông tin để học.
6. Phương pháp 'Cross-Validation'	- Đánh giá độ chính xác của mô hình bằng cách chia dữ liệu thành nhiều phần và kiểm tra mô hình trên các phần này. - Kỹ thuật như k-fold cross-validation giúp kiểm tra độ ổn định của mô hình trên các tập dữ liệu khác nhau

Gradient Boosting (Boosting)

Xây dựng các mô hình liên tiếp, mỗi mô hình mới cố gắng sửa chữa lỗi của mô hình trước. Kết quả cuối cùng là sự kết hợp của nhiều mô hình yếu được huấn luyện tuần tự. Hiệu quả cao, nhưng dễ overfitting nếu không kiểm soát cẩn thận.

Random Forest (Bagging)

Sử dụng nhiều cây quyết định độc lập, mỗi cây được huấn luyện trên các tập dữ liệu và đặc trưng ngẫu nhiên. Kết quả dự đoán được lấy bằng cách bỏ phiếu số đông (phân loại) hoặc trung bình (hồi quy). Giảm overfitting nhờ tính ngẫu nhiên và mạnh mẽ trong các tập dữ liệu lớn.

Bài 21: (Giải bài toán bằng bút và máy tính cầm tay)

Xét lại bộ dữ liệu huấn luyện như trình bày trong Bảng 1. Dựa vào ví dụ đã trình bày trên lớp, hãy thực hiện đầy đủ các bước giúp lựa chọn được đặc trưng để xây dựng nút gốc trong cây quyết định sử dụng thuật toán ID3.

Day	*Outlook*	*Temperature*	*Humidity*	*Wind*	*PlayTennis*
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Bảng 1: Bộ dữ liệu huấn luyện gồm 14 mẫu; mỗi mẫu có 04 đặc trưng (Outlook, Temperature, Humidity, Wind) và nhãn (PlayTennis).

Ngày: 16-9

Học máy - ID3 -

Xét tập dữ liệu

Day	Outlook	Temp	Humi	Wind	PlayTennis (root nodes)
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	W	Yes
4	Rain	Mild	High	W	Yes
5	Rain	Cool	Normal	W	Yes
6	Rain	Cool	Normal	S	No
7	Overcast	Cool	N	S	Yes
8	Sunny	Mild	High	W	No
9	Sunny	Cool	N	W	Yes
10	Rain	Mild	N	W	Yes
11	Sunny	Mild	N	S	Yes
12	Overcast	Mild	High	S	Yes
13	Overcast	Hot	N	W	Yes
14	Rain	Mild	High	S	No

Có 4 thuộc tính:
- Outlook : Sunny, Overcast, Rain
- Temp : Hot, Mild, Cold
- Humi : High, Normal
- Wind : Weak, Strong

Ngày:

Entropy tại root node: (5 No - 9 Yes)

$$E(S) = -\left(\frac{5}{14}\right)\log_2\left(\frac{5}{14}\right) - \left(\frac{9}{14}\right)\log_2\left(\frac{9}{14}\right)$$

$$\approx 0,94$$

Tính trong nó entropy của các childnode

① Đầu tiên: Outlook

1	Sunny	No
2	Sunny	No
8	Sunny	No
9	Sunny	Yes
11	Sunny	Yes

$m_s = 5$, $S_s = 0,94$

3	Overcast	Yes
7	Overcast	Yes
12	Overcast	Yes
13	Overcast	Yes

$m_o = 4$; $S_o = 0$

4	Rain	Yes
5	Rain	Yes
6	Rain	No
10	Rain	Yes
14	Rain	No

$m_r = 5$; $S_r = 0,94$

[Tập hợp các điểm trong mỗi child node: S_s, S_o, S_r tương ứng: m_s, m_o, m_r

$$\begin{cases} E(S_o) = 0 \\ E(S_s) = -\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5} = 0,971 \\ E(S_r) = -\frac{3}{5}\log_2\frac{3}{5} - \frac{2}{5}\log_2\frac{2}{5} = 0,971 \end{cases}$$

$$\Rightarrow Gain(Outlook, S) = 0,94 - \frac{5}{14}.0,97 - \frac{5}{14}.0,971 = 0,246$$

② Thuộc tính: Temperature

1	Hot	No
2	Hot	No
3	Hot	Yes
13	Hot	Yes

$m_H = 4$;

4	Mild	Yes
8	Mild	No
10	Mild	Yes
11	Mild	Yes
12	Mild	Yes
14	Mild	No

$m_M = 6$;

5	cool	Yes
6	cool	No
7	cool	Yes
9	cool	Yes

$m_C = 4$;

$$\begin{cases} E(S_H) = -(2/4)\log_2(2/4) - (2/4)\log_2(2/4) = 1 \\ E(S_M) = -4/6\log_2(4/6) - 2/6\log_2(2/6) = 0,918 \\ E(S_C) = -3/4\log_2(3/4) - 1/4\log_2(1/4) = 0,811 \end{cases}$$

$$\Rightarrow Gain(temp;S) = 0,97 - \frac{4}{14}(1) - \frac{6}{14}(0,918) - \frac{4}{14}(0,811) = 0,029$$

③ Thuộc tính: Humi

1	High	No
2	High	No
3	High	Yes
4	High	Yes
8	High	No
12	High	Yes
14	High	No

$m_{High} = 7$

5	Normal	Yes
6	Normal	No
7	Normal	Yes
9	Normal	Yes
10	Normal	Yes
11	Normal	Yes
13	Normal	Yes

$m_{Nor} = 7$

Ngày:

$$\begin{cases} E(S_{High}) = -\frac{3}{7}\log_2\left(\frac{3}{7}\right) - \frac{4}{7}\log_2\left(\frac{4}{7}\right) = 0,985 \\ E(S_{Nor}) = -\frac{6}{7}\log_2\left(\frac{6}{7}\right) - \frac{1}{7}\log_2\left(\frac{1}{7}\right) = 0,592 \end{cases}$$

$\Rightarrow$ Gain(humi, S) $= 0,94 - \frac{7}{14}.0,985 - \frac{7}{14}.0,592 = 0,1515$

④ Thuộc tính [Wind]

1	Weak	No
3	Weak	Yes
4	Weak	Yes
5	Weak	Yes
8	Weak	No
9	Weak	Yes
10	Weak	Yes
13	Weak	Yes

$m_w = 8$

2	Strong	No
6	Strong	No
7	Strong	Yes
11	Strong	Yes
12	Strong	Yes
14	Strong	No

$m_s = 6$

$$\begin{cases} E(S_w) = -(2/8)\log_2(2/8) - 6/8\log_2(6/8) = 0,811 \\ E(S_s) = -(3/6)\log_2(3/6) - 3/6\log_2(3/6) = 1 \end{cases}$$

$\Rightarrow$ Gain(Wind, S) $= 0,94 - \frac{8}{14}.0,811 - \frac{6}{14}.1 = 0,048$

Tổng E

Thuộc tính chọn ở bước đầu tiên: Outlook (≈ 0,67) (0,246)

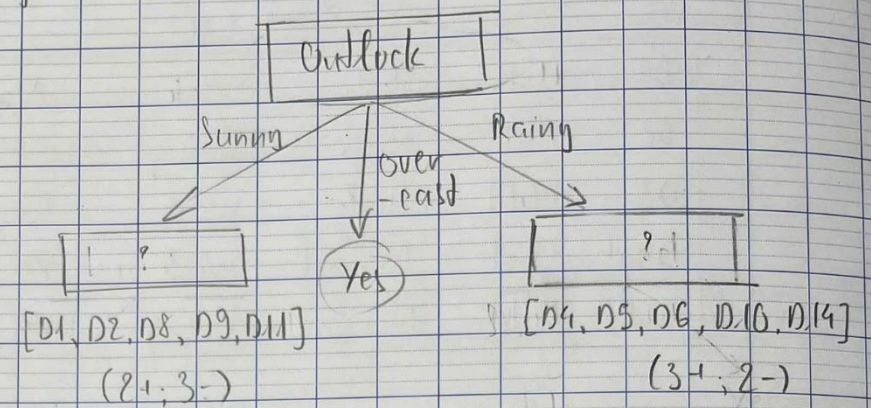
(gọi nhánh có ≈ infor gain lớn nhất)

S

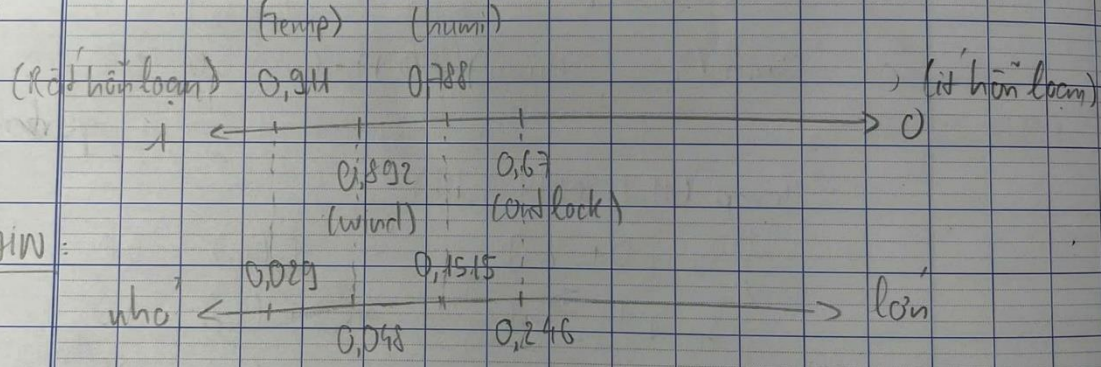
Infrgain

Ngày:

"Play or Not"



*Entropy:



*INFO GAIN:

Ngày:

- Xét tập dữ liệu S_{sunny} (Hai phép chọn Humi cho nhánh Sunny)

Day	Temp	Humi	Wild		
D1	Hot	High	Weak	No	$S_{sunny} = 0,970$
D2	Hot	High	Strong	No	
D8	Mild	High	Weak	No	
D9	Cool	Normal	Weak	Yes	
D11	Mild	Normal	Strong	Yes	

- Humi (High, Normal)

S_{High} [0+; 3-] $\Rightarrow$ $E(S_H) = 0.0$

S_{Normal} [2+; 0-] $\Rightarrow$ $E(S_N) = 0.0$

Gain (S_{sunny}, Humi) = 0,97 - (3/5).0.0 - (2/5) 0.0 = 0,97

- Temp (Hot, Mild, Cold)

S_H [0+; 2-] $\Rightarrow$ $E(S_H) = 0.0$

S_M [1+, 1-] $\Rightarrow$ $E(S_M) = 1.0$

S_C [1+, 0-] $\Rightarrow$ $E(S_C) = 0.0$

Gain (S_{sunny}, Temp) = 0,97 - (2/5).0.0 - (2/5).1 - (1/5).0.0

= 0,57

- Wind (Weak, Strong)

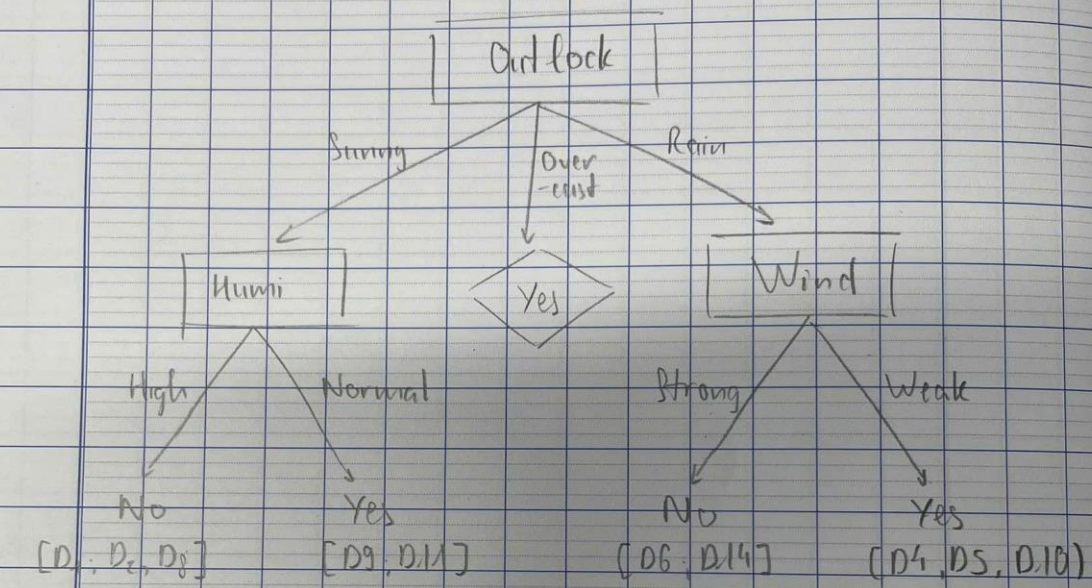
S_W [1+; 2-] $\Rightarrow$ $E(S_W) = 0,9183$

S_S [1+; 1-] $\Rightarrow$ $E(S_S) = 1$

Gain (S_{sunny}, Wind) = $0,97 - \frac{3}{5} \cdot 0,9183 - \frac{2}{5} \cdot 1 = 0,019$

Ngày:

- Vì gain của Humi cao nhất (=0,97)
  => Chọn Humi cho nhánh Outlook : Sunny
- Tập overcast kéo cần xét, tương thì với tập Rain
- Nhánh còn lại chọn thuộc tính Wind.



Bài 3: (Thực hành với Python)

Trong ví dụ về phân lớp các loài hoa diên vĩ của mình, tác giả Andreas Mueller đã xây dựng một mô hình ML sử dụng k-Nearest Neighbors với k = 1 (KNeighborsClassifier(n_neighbors=1)). Ví dụ tham khảo có thể xem và download tại:

https://github.com/amueller/introduction_to_ml_with_python/blob/main/01-introduction.ipynb

Dựa trên ví dụ của tác giả, em hãy sử dụng một cách tiếp cận khác (ví dụ như Support Vector Machine, Decision Trees, Random Forests, ...) để phân lớp các loài hoa diên vĩ. So sánh kết quả em đạt được với kết quả của tác giả.

```
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import ConfusionMatrixDisplay, accuracy_score
import matplotlib.pyplot as plt
import numpy as np
from matplotlib.colors import ListedColormap

# Bước 1: Tải dữ liệu hoa diên vĩ
iris = load_iris()
X, y = iris.data, iris.target

# Chia dữ liệu thành tập huấn luyện và tập kiểm tra
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25,
random_state=42)

# Khởi tạo mô hình k-NN với k=1 và huấn luyện
knn = KNeighborsClassifier(n_neighbors=1)
knn.fit(X_train, y_train)

# Khởi tạo mô hình Decision Tree và huấn luyện
decision_tree = DecisionTreeClassifier(random_state=42)
decision_tree.fit(X_train, y_train)

# Bước 2: Tính toán độ chính xác của cả hai mô hình
knn_predictions = knn.predict(X_test)
decision_tree_predictions = decision_tree.predict(X_test)

knn_accuracy = accuracy_score(y_test, knn_predictions)
decision_tree_accuracy = accuracy_score(y_test, decision_tree_predictions)
```

```python
# Hiển thị độ chính xác
print(f"Độ chính xác của mô hình k-NN (k=1): {knn_accuracy:.2f}")
print(f"Độ chính xác của mô hình Decision Tree: {decision_tree_accuracy:.2f}")

# So sánh độ chính xác giữa hai mô hình
if decision_tree_accuracy > knn_accuracy:
    print("Mô hình Decision Tree có độ chính xác cao hơn.")
elif decision_tree_accuracy < knn_accuracy:
    print("Mô hình k-NN có độ chính xác cao hơn.")
else:
    print("Hai mô hình có độ chính xác tương đương.")

# Bước 3: Hiển thị ma trận nhầm lẫn cho k-NN và Decision Tree
print("\nMa trận nhầm lẫn cho mô hình k-NN:")
ConfusionMatrixDisplay.from_estimator(knn, X_test, y_test)
plt.show()

print("\nMa trận nhầm lẫn cho mô hình Decision Tree:")
ConfusionMatrixDisplay.from_estimator(decision_tree, X_test, y_test)
plt.show()

# Bước 4: Vẽ biên giới quyết định (decision boundary) cho k-NN và Decision Tree
# Sử dụng 2 thuộc tính đầu tiên để dễ hiển thị
X_train_2D = X_train[:, :2]  # Lấy 2 thuộc tính đầu tiên
X_test_2D = X_test[:, :2]

# Huấn luyện lại các mô hình chỉ với 2 thuộc tính
knn_2D = KNeighborsClassifier(n_neighbors=1)
knn_2D.fit(X_train_2D, y_train)

decision_tree_2D = DecisionTreeClassifier(random_state=42)
decision_tree_2D.fit(X_train_2D, y_train)

# Hàm vẽ biên giới quyết định
def plot_decision_boundaries(X, y, model, model_name):
    cmap_light = ListedColormap(['#FFAAAA', '#AAFFAA', '#AAAAFF'])
    cmap_bold = ListedColormap(['#FF0000', '#00FF00', '#0000FF'])

    h = .02  # kích thước lưới
    x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
    y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1
    xx, yy = np.meshgrid(np.arange(x_min, x_max, h),
                         np.arange(y_min, y_max, h))

    # Dự đoán nhãn cho mỗi điểm trong lưới
    Z = model.predict(np.c_[xx.ravel(), yy.ravel()])
    Z = Z.reshape(xx.shape)
```

```
    # Vẽ biên giới quyết định và các điểm dữ liệu
    plt.figure(figsize=(8, 6))
    plt.contourf(xx, yy, Z, cmap=cmap_light)
    plt.scatter(X[:, 0], X[:, 1], c=y, cmap=cmap_bold, edgecolor='k', s=20)
    plt.title(f"Decision Boundary for {model_name}")
    plt.xlabel(iris.feature_names[0])
    plt.ylabel(iris.feature_names[1])
    plt.show()

# Vẽ biên giới quyết định cho k-NN
plot_decision_boundaries(X_test_2D, y_test, knn_2D, "k-NN")

# Vẽ biên giới quyết định cho Decision Tree
plot_decision_boundaries(X_test_2D, y_test, decision_tree_2D, "Decision Tree")
```