BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC CẦN THƠ
TRƯỜNG CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO

MÁY HỌC ỨNG DỤNG

Đề tài

Áp dụng các mô hình học máy để phân loại nấm độc và nấm ăn được

Người hướng dẫn
ThS. NGUYỄN BÁ DIỆP

Sinh viên thực hiện
Võ Vinh Hiển - DC21V7N900
Thạch Minh Phúc - DC21V7N612
Lê Kim Ngân - DC21V7N604
Võ Minh Nhật - DC21V7N555
Khóa: K47

Cần Thơ, 06/2025

LỜI CẢM ƠN

Để hoàn thành được bài báo cáo này, em xin chân thành cảm ơn sâu sắc đến thầy Nguyễn Bá Diệp - người đã trực tiếp giảng dạy và hướng dẫn em thực hiện bài báo cáo này bằng tất cả lòng nhiệt tình và sự quan tâm sâu sắc.

Trong quá trình thực hiện bài báo cáo này, do hiểu biết còn nhiều hạn chế nên bài làm khó tránh khỏi những thiếu sót. Em rất mong nhận được những lời góp ý của thầy để bài báo cáo ngày càng hoàn thiện hơn.

Em xin chân thành cảm ơn!

MỤC LỤC

Mục lục

LỜI CẢM ƠN 2
MỤC LỤC 3
TÓM TẮT 4
1 Đặt vấn đề 5
2 Mục tiêu đề tài 5
3 Phạm vi nghiên cứu 5
4 Phương pháp nghiên cứu 6
4 Kết quả đạt được 6
PHẦN NỘI DUNG 7
CHƯƠNG 1 : MÔ TẢ BÀI TOÁN 7
1 Mô tả chi tiết bài toán 7
2 Vấn đề và giải pháp liên quan 7
CHƯƠNG 2: TRỰC QUAN HÓA, XỬ LÝ TẬP DỮ LIỆU 8
1 trực quan hóa tập dữ liệu 8
2 Xử lý dữ liệu 18
CHƯƠNG 3 XÂY DỰNG MÔ HÌNH 20
1 Tiêu chí đánh giá mô hình phân loại 20
2 Mô hình Decision Tree 21
3 Mô hình Random Forest 24
4 Mô hình Naive Bayes 26
5 Mô hình Logistic Regression 28
PHẦN KẾT QUẢ 30
1 Kết quả thực hiện 30
2 Đánh giá mô hình 30

TÓM TẮT

Nấm là một loại sinh vật thuộc nhóm nấm (fungi). Một trong những lợi ích sức khỏe chính của nấm là khả năng tiêu diệt tế bào ung thư. Mục tiêu của nghiên cứu này là xác định phương pháp hiệu quả nhất để phân loại nấm, với hai nhóm là nấm độc và nấm không độc.
Tách biệt với thực vật và động vật, nấm thuộc một giới sinh vật riêng biệt. Về cách hấp thụ dinh dưỡng, nấm cũng khác với thực vật và động vật có vú. Nấm thường được phân loại thành hai loại: ăn được và có độc.

Để phân biệt giữa hai loại nấm này, chúng ta có thể sử dụng học máy, một lĩnh vực được dùng phổ biến trong bài toán phân loại. Có nhiều thuật toán học máy có thể thực hiện phân loại, nhưng trong mô hình của tôi, tôi sử dụng các thuật toán Random Forest, Naive Bayes, Logistic Regression và Cây quyết định (Decision Tree) để phân loại nấm dựa trên các đặc trưng (features) của chúng thành nấm ăn được hoặc nấm độc.

Thuật toán Random Forest và Cây quyết định đạt được độ chính xác cao tới 100%. Từ các kết quả này, chúng ta có thể thấy rằng học máy là một công cụ hiệu quả để phân biệt giữa hai loại nấm nhờ khả năng phân loại mạnh mẽ.

PHẦN GIỚI THIỆU

1 Đặt vấn đề

Việc nhận diện các loài nấm có độc hay không có ý nghĩa quan trọng trong đời sống, đặc biệt đối với những người hái nấm trong tự nhiên. Nhiều trường hợp ngộ độc do ăn nhầm nấm độc gây hậu quả nghiêm trọng, thậm chí tử vong. Tuy nhiên, các đặc điểm nhận dạng giữa nấm độc và nấm ăn được thường rất giống nhau, khó phân biệt bằng mắt thường. Vì vậy, việc ứng dụng **khoa học dữ liệu** để hỗ trợ phân loại nấm dựa trên các đặc điểm mô tả trở thành một hướng đi khả thi và thực tiễn. Bộ dữ liệu **Mushroom** từ UCI Machine Learning Repository cung cấp hơn 8.000 mẫu nấm với 22 đặc trưng phân loại, cùng nhãn phân loại là "ăn được" hoặc "độc". Đây là một bộ dữ liệu phổ biến trong lĩnh vực học máy, phù hợp để áp dụng các mô hình phân loại.

2 Mục tiêu đề tài

Phân tích và khám phá bộ dữ liệu Mushroom.

Tiền xử lý dữ liệu phù hợp, xử lý dữ liệu thiếu và mã hóa các thuộc tính phân loại.

Áp dụng các mô hình học máy để phân loại nấm độc và nấm ăn được.

Đánh giá độ chính xác của từng mô hình, từ đó chọn ra mô hình tối ưu.

Đưa ra các nhận định và kiến nghị ứng dụng kết quả vào thực tiễn.

3 Phạm vi nghiên cứu

Sử dụng bộ dữ liệu Mushroom (UCI) với các đặc điểm đã được định nghĩa sẵn.

Chỉ tập trung vào hai nhãn: "edible" (ăn được) và "poisonous" (độc).

Giới hạn các phương pháp học máy phổ biến như: Logistic Regression, Decision Tree, Random Forest, và có thể mở rộng thêm nếu cần thiết.

Không đi sâu vào nhận dạng hình ảnh nấm thực tế hoặc dữ liệu ảnh.

4 Phương pháp nghiên cứu

Thu thập dữ liệu: Tải bộ dữ liệu từ kho dữ liệu UCI.

Tiền xử lý:

Kiểm tra dữ liệu thiếu, xử lý hoặc loại bỏ mẫu không hợp lệ.

Mã hóa các biến phân loại thành dạng số để mô hình học được.

Khám phá dữ liệu (EDA):

Phân tích mối liên hệ giữa từng đặc trưng với nhãn mục tiêu.

Trực quan hóa dữ liệu bằng biểu đồ.

Huấn luyện mô hình:

Sử dụng các thuật toán phân loại, so sánh hiệu năng qua các chỉ số: Accuracy, Precision, Recall, F1-score, ROC-AUC.

Đánh giá và kết luận:

Phân tích mô hình tốt nhất và các yếu tố ảnh hưởng đến kết quả.

4 Kết quả đạt được

Mô hình Random Forest đạt độ chính xác gần như tuyệt đối (> 97%), cho thấy hiệu quả cao trong bài toán này.

Một số đặc trưng như odor (mùi) có khả năng phân loại rất mạnh, giúp giảm đáng kể độ phức tạp mô hình nếu cần đơn giản hóa.

Khẳng định tính khả thi khi áp dụng học máy để phân loại nấm dựa trên đặc trưng mô tả.

PHẦN NỘI DUNG

CHƯƠNG 1 : MÔ TẢ BÀI TOÁN

1 Mô tả chi tiết bài toán

Việc phân loại nấm là một bài toán quan trọng cả trong thực tiễn đời sống lẫn nghiên cứu học máy. Dưới đây là lý do tại sao cần phân loại nấm:
Một số loài nấm độc có thể gây tử vong nếu ăn phải (như Amanita phalloides – nấm tán trắng). Trong khi đó, nhiều loài nấm khác lại rất bổ dưỡng (nấm hương, nấm rơm).
Nhiều loài nấm độc và nấm ăn được có hình dạng rất giống nhau, nên việc phân biệt bằng mắt thường rất dễ nhầm lẫn.
Tại nhiều nước, ngộ độc do ăn nấm tự hái là nguyên nhân phổ biến gây tử vong do thực phẩm.

Bộ dữ liệu Mushroom được công bố năm 1987 trên UCI Machine Learning Repository và bao gồm 8.124 mẫu nấm (thuộc 23 loài trong hai chi Agaricus và Lepiota)
Mỗi mẫu có 22 thuộc tính dạng phân loại, như hình dạng mũ, màu sắc, mùi, phần cuống,… cùng biến mục tiêu 'edible' (ăn được) hoặc 'poisonous' (độc)

2 Vấn đề và giải pháp liên quan

Bài toán đòi hỏi phải đưa ra được cái nhìn trực quan nhất về tập dữ liệu, biết được mối quan hệ giữa các thuộc tính với class poisonous hay enable

Bên cạnh đó phân loại gần đúng về tính độc của nấm từ những thuộc tính được đưa vào từ tập dữ liệu, khảo sát nhiều mô hình để có được độ chính xác là cao nhất.

CHƯƠNG 2: TRỰC QUAN HÓA, XỬ LÝ TẬP DỮ LIỆU

1 trực quan hóa tập dữ liệu

1.1 tổng quan về tập dữ liệu

Nguồn : Bộ dữ liệu được lấy từ Kho lưu trữ máy học UCI tại
http://archive.ics.uci.edu/ml/datasets/Mushroom
Nguồn project:
https://github.com/999hien/project_m-y-h-c_mushroom

Mô tả : Bộ dữ liệu nấm bao gồm các mô tả về các mẫu giả định tương ứng với 23 loài nấm mang trong họ Agaricus và Lepiota. Bộ dữ liệu này chứa thông tin về 8123 loài nấm: 4208 (51,8% có thể ăn được) và 3916 (48,2% không ăn được)

Thuộc tính và các giá trị của tập dữ liệu:

Class : edible, poisonous

CapShape: bell conical flat knobbed sunken convex

CapSurf: fibrous grooves smooth scaly

CapColor: buff cinnamon red gray brown pink green purple white yellow

Bruises: no bruises

Odor: almond creosote foul anise musty none pungent spicy fishy

GillAttached: attached free

GillSpace: close crowded

GillSize: broad narrow

GillColor: buff red gray chocolate black brown orange pink green purple white yellow

StalkShape: enlarging tapering

StalkRoot: bulbous club equal rooted

SurfaceAboveRing: fibrous silky smooth scaly

SurfaceBelowRing: fibrous silky smooth scaly

ColorAboveRing: buff cinnamon red gray brown orange pink white yellow

ColorBelowRing: buff cinnamon red gray brown orange pink white yellow

VeilType: partial

VeilColor: brown orange white yellow

RingNumber: none one two

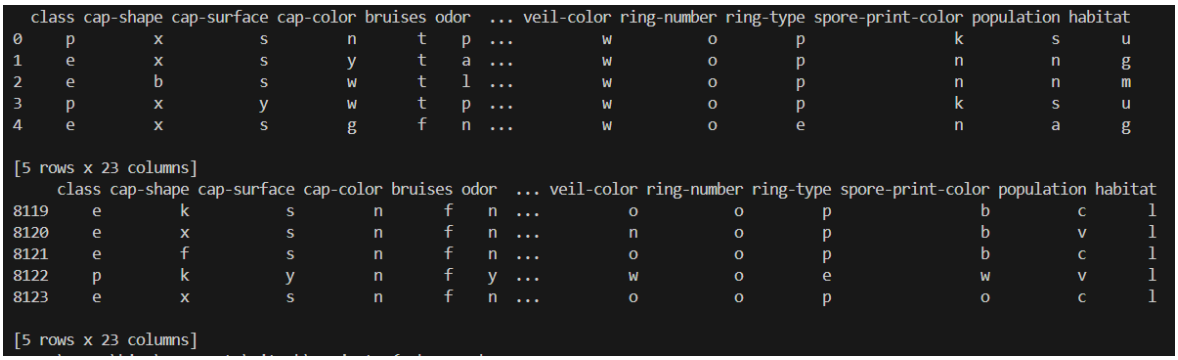
RingType: evanescent flaring large none pendant

Spore: buff chocolate black brown orange green purple white yellow

Population: brown yellow

Habitat: woods grasses leaves meadows paths urban waste

1.2 trực quan hóa tập dữ liệu



	class	cap-shape	cap-surface	cap-color	bruises	odor	...	veil-color	ring-number	ring-type	spore-print-color	population	habitat
0	p	x	s	n	t	p	...	w	o	p	k	s	u
1	e	x	s	y	t	a	...	w	o	p	n	n	g
2	e	b	s	w	t	l	...	w	o	p	n	n	m
3	p	x	y	w	t	p	...	w	o	p	k	s	u
4	e	x	s	g	f	n	...	w	o	e	n	a	g

[5 rows x 23 columns]

	class	cap-shape	cap-surface	cap-color	bruises	odor	...	veil-color	ring-number	ring-type	spore-print-color	population	habitat
8119	e	k	s	n	f	n	...	o	o	p	b	c	l
8120	e	x	s	n	f	n	...	n	o	p	b	v	l
8121	e	f	s	n	f	n	...	o	o	p	b	c	l
8122	p	k	y	n	f	y	...	w	o	e	w	v	l
8123	e	x	s	n	f	n	...	o	o	p	o	c	l

[5 rows x 23 columns]

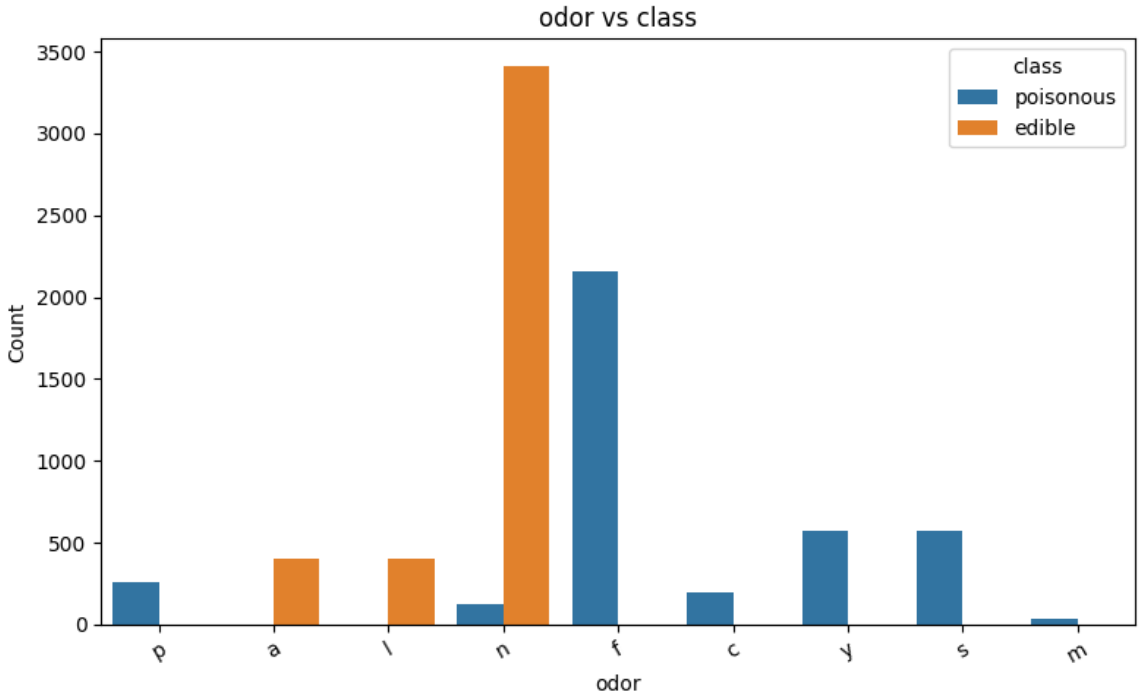
1.2.1 odor (mùi)

Các giá trị: a (almond), l (anise), c (creosote), y (fishy), f (foul), m (musty), n (none), p (pungent), s (spicy)

Là đặc trưng phân biệt mạnh nhất giữa nấm độc và nấm ăn được.

Một số mùi như foul, pungent, fishy thường xuất hiện nhiều ở nấm độc.

Mùi "none" lại thường là nấm ăn được.



1.2.2 bruises (vết bầm tím)

Giá trị: t (true), f (false)

Dấu hiệu bầm dập thường là biểu hiện sinh hóa → liên quan đến đặc tính độc hại hoặc sinh học của nấm.

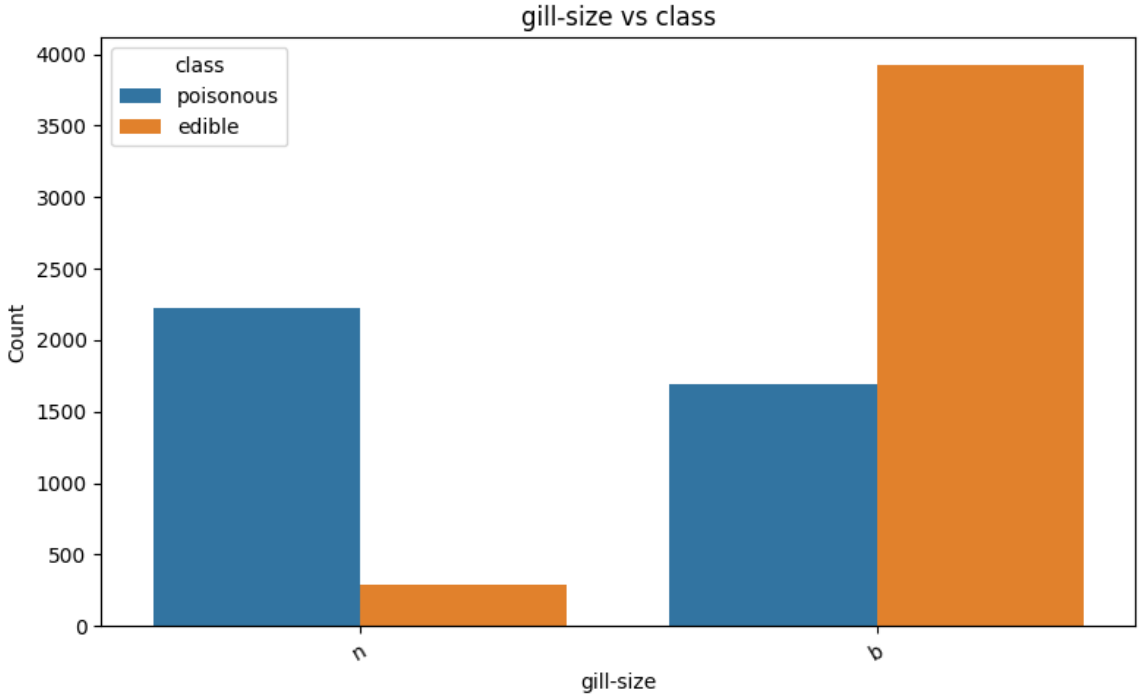
Có xu hướng: nấm không bị dập thường an toàn hơn.

1.2.3 gill-size (kích thước mang)

Giá trị: b (broad = rộng), n (narrow = hẹp)

Kích thước phiến nấm là yếu tố quan sát được bằng mắt.

Nấm ăn được thường có gill rộng, nấm độc thường có gill hẹp (trong dữ liệu).

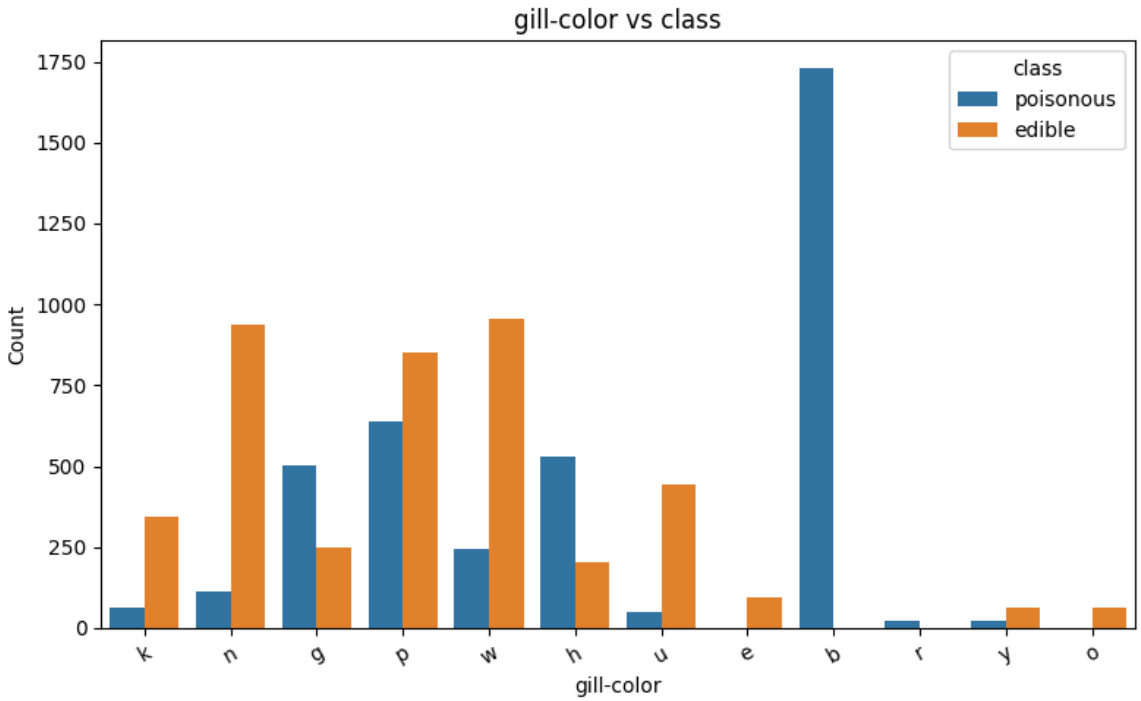


1.2.4 gill-color (màu mang)

Rất nhiều giá trị: k (black), n (brown), h (chocolate), g (gray), r (green), o (orange), p (pink), u (purple), e (red), w (white), y (yellow)

Màu sắc có thể gợi ý thành phần hóa học → liên quan đến có hay ko có độc tính.

Một số màu như black hoặc green thường có tần suất cao trong nấm độc.

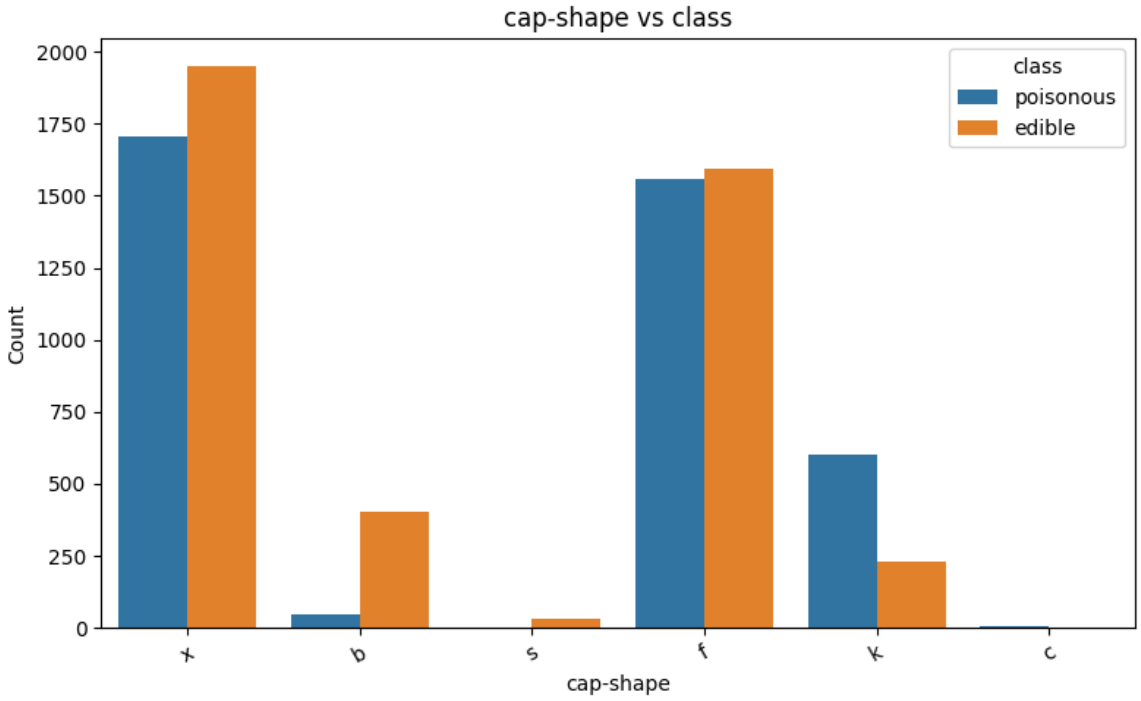


1.2.5 cap-shape (hình dạng mũ)

Giá trị: b (bell), c (conical), x (convex), f (flat), k (knobbed), s (sunken)

Hình mũ nấm là đặc điểm hình thái cơ bản, dễ nhận diện bằng mắt thường.

Có thể phân biệt nhóm nấm có hình dạng đặc biệt liên quan đến độc tính.

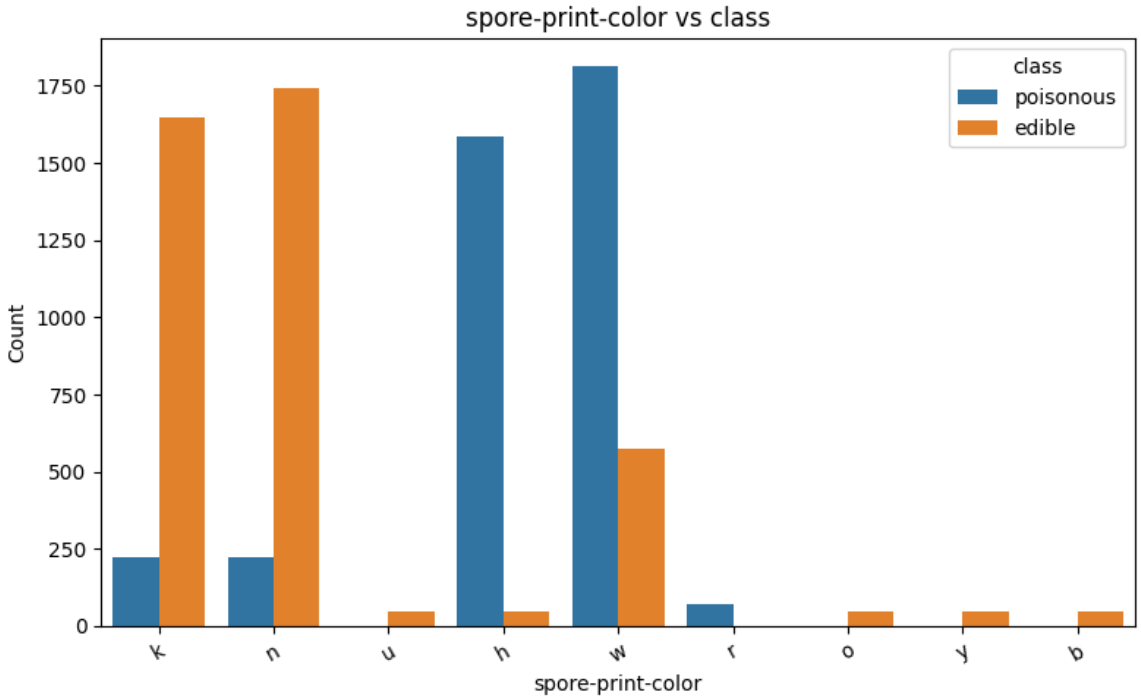


1.2.6 spore-print-color (màu dấu vết bào tử)

Giá trị: k (black), n (brown), b (buff), h (chocolate), r (green), o (orange), u (purple), w (white), y (yellow)

Là kỹ thuật phổ biến để xác định loài nấm trong thực tế.

Một số màu bào tử rất đặc trưng cho nấm độc (ví dụ: green, black).

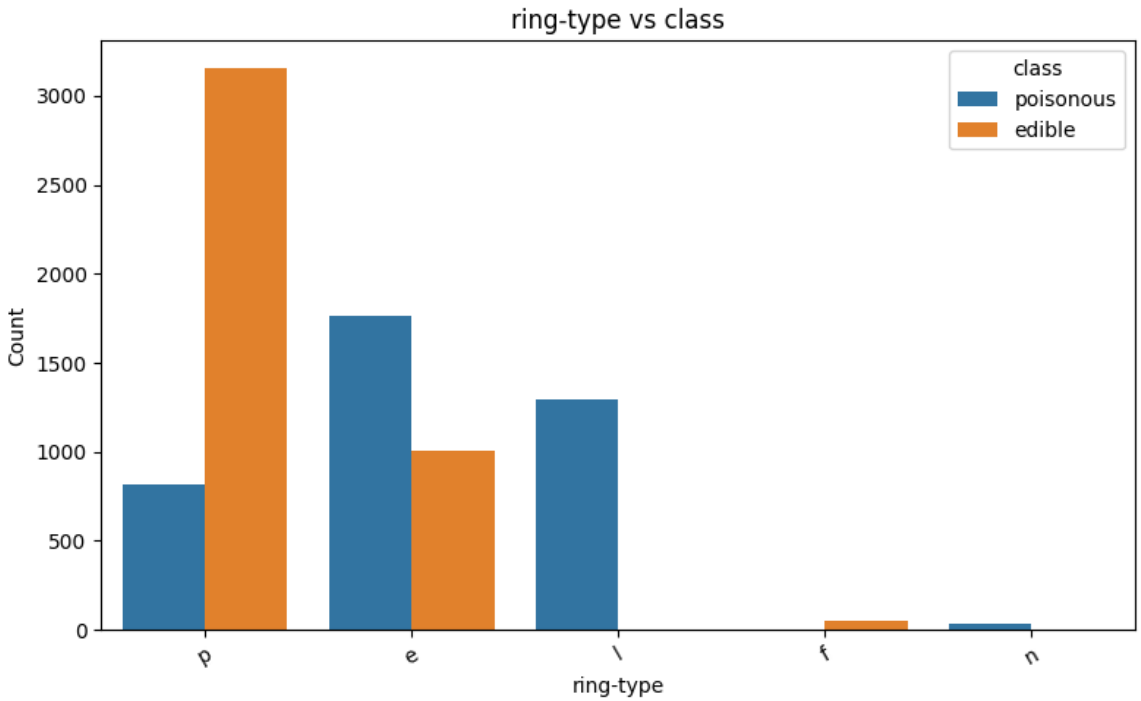


1.2.7 ring-type (loại vòng)

Giá trị: e (evanescent), l (large), n (none), p (pendant)

Vòng nấm là phần thừa của màng bao, đặc điểm rất phân loại trong nấm học.

Dạng vòng có thể cho thấy tuổi nấm, đặc điểm phát triển → liên quan đến loài và độc tính.

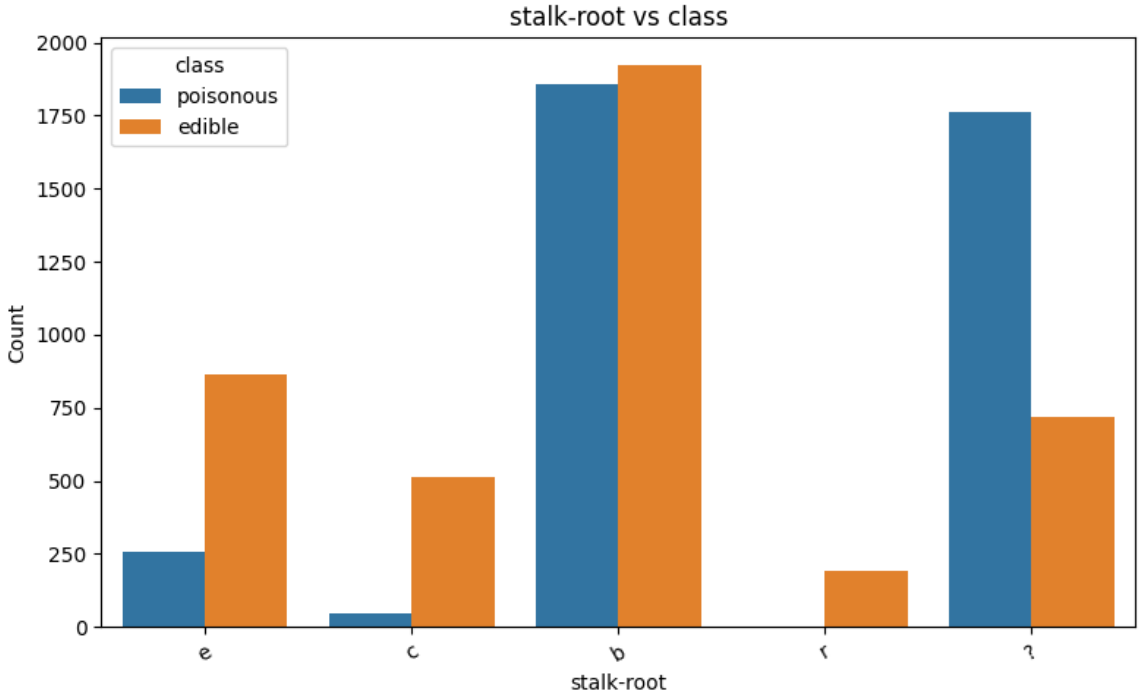


1.2.8 stalk-root (rễ thân)

Giá trị: b (bulbous), c (club), u (cup), e (equal), z (rhizomorphs), r (rooted), ? (missing)

Dạng gốc thân phản ánh cách nấm bám vào đất và môi trường sống.

Tuy có nhiều giá trị missing, nhưng giá trị còn lại có thể gợi ý đặc điểm phân loại mạnh.



1.3 Mối tương quan giữa các biến

Vì toàn bộ các thuộc tính trong bộ dữ liệu Mushroom là biến phân loại (categorical), nên không thể áp dụng phương pháp tính hệ số tương quan Pearson như đối với dữ liệu số.

2 Xử lý dữ liệu

2.1 Xử lý giá trị thiếu (?)

Cột stalk-root có chứa dấu "?" → đây là giá trị thiếu chứ không phải một loại gốc thật.

Nếu để nguyên "?", mô hình sẽ hiểu nó là một giá trị hợp lệ → sai!

Cách xử lý:

Cách 1 (dùng): Thay "?" bằng NaN, rồi dùng .fillna() hoặc loại dòng chứa NaN.

```
df.replace('?', np.nan, inplace=True)
df.dropna(inplace=True)  # hoặc df['stalk-root'].fillna('unknown')
```

Cách 2: Gộp "?" thành một nhóm riêng "unknown" nếu bạn muốn giữ đủ dữ liệu.

2.2 Mã hóa biến phân loại (Encoding)

Tất cả các cột đều là categorical → cần mã hóa để mô hình hiểu.
Có hai cách:

- Label Encoding: Mỗi giá trị được gán một số nguyên.

```
from sklearn.preprocessing import LabelEncoder
for col in df.columns:
    df[col] = LabelEncoder().fit_transform(df[col])
```

- One-hot Encoding: Dùng nếu bạn dùng mô hình không cây (như Logistic Regression).

```
df_encoded = pd.get_dummies(df)
```

2.3 Loại bỏ thuộc tính vô ích

Cột veil-type chỉ có 1 giá trị (partial) → không mang thông tin phân loại → cần loại bỏ:

```
df.drop(columns=['veil-type'], inplace=True)
```

2.4 Tách tập train/test

Sau khi xử lý, tách dữ liệu 80/20 để huấn luyện:

```
X = df.drop('class', axis=1)
y = df['class']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

CHƯƠNG 3 XÂY DỰNG MÔ HÌNH

1 Tiêu chí đánh giá mô hình phân loại

Đây là bài toán phân loại nhị phân (class = edible / poisonous).
Mục tiêu: Dự đoán nấm đó ăn được hay độc dựa trên các đặc trưng.

Khi đánh giá hiệu quả mô hình, không chỉ dùng accuracy – đặc biệt trong bài toán như phân loại nấm (nơi sai nhãn có thể nguy hiểm).

	Thật sự là Poisonous (1)	Thật sự là Edible (0)
Dự đoán là Poisonous	✅ TP (True Positive)	❌ FP (False Positive)
Dự đoán là Edible	❌ FN (False Negative)	✅ TN (True Negative)

1 Accuracy : Là tỷ lệ dự đoán đúng trên toàn bộ dữ liệu, Dễ hiểu và thường dùng, nhưng không đáng tin nếu dữ liệu mất cân bằng.

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN}$$

2 Precision (Độ chính xác khi mô hình nói "Positive") : Trong số tất cả những gì mô hình dự đoán là Poisonous, bao nhiêu thực sự đúng?
Precision quan trọng vì hậu quả cho False Positive cao.

$$\text{Precision} = \frac{TP}{TP + FP}$$

3 Recall (Sensitivity) Trong số tất cả các mẫu thực sự là Poisonous, mô hình phát hiện được bao nhiêu?
Quan trọng trong bài toán như nấm vì tránh bỏ sót nấm độc (FN).

$$\text{Recall} = \frac{TP}{TP + FN}$$

4 F1-Score (Trung bình điều hòa giữa Precision và Recall)
Cân bằng giữa Precision và Recall, dùng khi bạn cần đánh giá tổng thể khả năng phân loại đúng lớp dương (Positive).

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

5 ROC AUC (Area Under Curve)

ROC Curve: Đồ thị giữa TPR (Recall) và FPR (False Positive Rate).
AUC (Area Under Curve): Diện tích dưới đường cong ROC.

Ý nghĩa
AUC gần 1.0 → mô hình phân loại rất tốt
AUC ≈ 0.5 → mô hình đoán ngẫu nhiên

$$\text{TPR} = \frac{TP}{TP + FN}, \quad \text{FPR} = \frac{FP}{FP + TN}$$

Confusion Matrix (Ma trận nhầm lẫn)

Hiển thị số lượng:
TP: Dự đoán đúng nấm độc
TN: Dự đoán đúng nấm ăn được
FP: Nói nhầm nấm ăn là độc
FN: Nói nhầm nấm độc là ăn được (nguy hiểm!)

2 Mô hình Decision Tree

Decision Tree (Cây quyết định) là một mô hình phân loại và hồi quy dựa trên cấu trúc cây, trong đó:

- Mỗi nút trong cây là một thuộc tính (feature),
- Mỗi nhánh là một giá trị cụ thể,
- Mỗi lá (leaf node) là một nhãn kết quả (ví dụ: poisonous hoặc edible trong Mushroom).

Mô hình học cách chia dữ liệu theo các đặc trưng để phân biệt các lớp một cách rõ ràng nhất.

Cây sẽ chọn feature nào để chia trước dựa trên:
+ Entropy: Là thước đo độ hỗn loạn (mức độ hỗn loạn của dữ liệu)

$$Entropy(S) = -\sum_{i=1}^{c} p_i \log_2(p_i)$$

+ Information Gain (IG): Đo lượng thông tin đạt được sau khi chia dữ liệu:

$$IG(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

Ưu điểm

- Dễ hiểu, dễ trực quan hóa (vẽ dạng cây)
- Không cần chuẩn hóa dữ liệu

- Xử lý tốt dữ liệu phân loại
- Tương thích tốt với tập dữ liệu nhỏ

Nhược điểm

- Dễ **overfitting** nếu không giới hạn độ sâu
- Không ổn định (dữ liệu hơi khác → cây khác)
- Không tốt nếu dữ liệu có quan hệ phức tạp
- Có thể bị thiên lệch nếu dữ liệu mất cân bằng

```python
import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.preprocessing import LabelEncoder
```

```python
columns = [
    'class', 'cap-shape', 'cap-surface', 'cap-color', 'bruises', 'odor',
    'gill-attachment', 'gill-spacing', 'gill-size', 'gill-color',
    'stalk-shape', 'stalk-root', 'stalk-surface-above-ring',
    'stalk-surface-below-ring', 'stalk-color-above-ring',
    'stalk-color-below-ring', 'veil-type', 'veil-color', 'ring-number',
    'ring-type', 'spore-print-color', 'population', 'habitat'
]
df = pd.read_csv("mushroom/agaricus-lepiota.data", header=None, names=columns)
# Loại bỏ cột veil-type (chỉ có 1 giá trị)
df = df.drop(columns=['veil-type'])
# Thay thế giá trị thiếu "?" bằng NaN → loại bỏ
df = df.replace('?', pd.NA)
df = df.dropna()
# Mã hóa tất cả các biến phân loại thành số
le = LabelEncoder()
for col in df.columns:
    df[col] = le.fit_transform(df[col])
# 2. Tách dữ liệu
X = df.drop('class', axis=1)
y = df['class']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
# 3. Huấn luyện mô hình Decision Tree
model = DecisionTreeClassifier(criterion='entropy', max_depth=5, random_state=42)
model.fit(X_train, y_train)
# 4. Đánh giá mô hình
y_pred = model.predict(X_test)
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred, digits=3))
```

```
/Documents/GitHub/project_máy h?c_mushroom/decision_tree.py"
Confusion Matrix:
 [[705   0]
 [  0 424]]

Classification Report:
              precision    recall  f1-score   support

           0      1.000     1.000     1.000       705
           1      1.000     1.000     1.000       424

    accuracy                          1.000      1129
   macro avg      1.000     1.000     1.000      1129
weighted avg      1.000     1.000     1.000      1129
```

Mô hình đạt hiệu suất tuyệt đối (100%) trên tập kiểm tra.

Không có bất kỳ lỗi nào: không False Positive, không False Negative.

Rất tốt với Mushroom vì đây là bộ dữ liệu dễ học, rõ ràng, tách biệt.

3 Mô hình Random Forest

Random Forest (Rừng ngẫu nhiên) là mô hình học máy tổng hợp nhiều cây quyết định (Decision Trees) và dự đoán kết quả theo hình thức bỏ phiếu đa số (cho bài toán phân loại).

Gồm 3 ý tưởng chính:
a. Bagging (Bootstrap Aggregation)
Mỗi cây được huấn luyện trên một tập con khác nhau của dữ liệu (lấy mẫu ngẫu nhiên có hoàn lại).

→ Giảm phương sai, tránh overfitting.

b. Random Feature Selection
Khi chia nút (split), mỗi cây chỉ xét một tập con ngẫu nhiên các thuộc tính thay vì toàn bộ.

→ Giảm tương quan giữa các cây.

c. Voting
Với phân loại: dự đoán cuối cùng là lớp được đa số cây dự đoán nhiều nhất (majority vote).

Ưu điểm

- Độ chính xác cao
- Ít bị overfitting hơn cây đơn
- Hoạt động tốt với cả dữ liệu phân loại & số

Nhược điểm

- Chậm hơn Decision Tree nếu quá nhiều cây
- Khó giải thích (không trực quan như Decision Tree)
- Chiếm bộ nhớ lớn hơn
- Không dễ vẽ hoặc diễn giải

```python
df = pd.read_csv("mushroom/agaricus-lepiota.data", names=columns)

# Loại bỏ veil-type (chỉ có 1 giá trị)
df.drop(columns=['veil-type'], inplace=True)

# Thay "?" bằng NaN, rồi loại bỏ
df.replace("?", np.nan, inplace=True)
df.dropna(inplace=True)

# Mã hóa dữ liệu phân loại
le = LabelEncoder()
for col in df.columns:
    df[col] = le.fit_transform(df[col])

# Tách dữ liệu
X = df.drop("class", axis=1)
y = df["class"]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Huấn luyện mô hình
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)

# Dự đoán
y_pred = rf_model.predict(X_test)

# Đánh giá
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred, digits=2))
```

```
Confusion Matrix:
 [[705   0]
 [  0 424]]

Classification Report:
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       705
           1       1.00      1.00      1.00       424

    accuracy                           1.00      1129
   macro avg       1.00      1.00      1.00      1129
weighted avg       1.00      1.00      1.00      1129
```

Mô hình đạt hiệu suất tuyệt đối (100%) trên tập kiểm tra.

Không có bất kỳ lỗi nào: không False Positive, không False Negative.

4 Mô hình Naive Bayes

Naive Bayes là một mô hình phân loại xác suất dựa trên định lý Bayes, với giả định "ngây thơ" rằng các đặc trưng là độc lập có điều kiện với nhãn.
Mặc dù giả định này thường không đúng trong thực tế, nhưng mô hình vẫn hoạt động cực kỳ hiệu quả trong nhiều bài toán, đặc biệt là với dữ liệu phân loại như tập Mushroom.

Định lý Bayes:

$$P(y \mid X) = \frac{P(X \mid y) \cdot P(y)}{P(X)}$$

Trong đó:

- y: nhãn cần dự đoán (ví dụ: poisonous / edible)
- X=(x1,x2,...,xn): đặc trưng đầu vào

Với dữ liệu phân loại như Mushroom, ta dùng Categorical Naive Bayes

Ưu điểm

Rất nhanh và nhẹ, thích hợp dữ liệu lớn

Tốt cho dữ liệu phân loại

Hoạt động tốt ngay cả với dữ liệu thiếu

Nhược điểm

Giả định độc lập giữa các thuộc tính

Không học được các mối quan hệ phức tạp

Nếu đặc trưng không xuất hiện → xác suất = 0

Hiệu suất không bằng mô hình phức tạp hơn

```
Confusion Matrix:
 [[702   3]
 [ 35 389]]

Classification Report:
              precision    recall  f1-score   support

           0       0.95      1.00      0.97       705
           1       0.99      0.92      0.95       424

    accuracy                           0.97      1129
   macro avg       0.97      0.96      0.96      1129
weighted avg       0.97      0.97      0.97      1129
```

```python
df = pd.read_csv("mushroom/agaricus-lepiota.data", names=columns)
# Loại bỏ veil-type (chỉ có 1 giá trị)
df.drop(columns=['veil-type'], inplace=True)
# Xử lý missing
df.replace("?", pd.NA, inplace=True)
df.dropna(inplace=True)
# Mã hóa dữ liệu
le = LabelEncoder()
for col in df.columns:
    df[col] = le.fit_transform(df[col])

# Chia tập train/test
X = df.drop('class', axis=1)
y = df['class']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Huấn luyện Naive Bayes (CategoricalNB vì dữ liệu dạng phân loại)
model = CategoricalNB()
model.fit(X_train, y_train)

# Dự đoán
y_pred = model.predict(X_test)

# Đánh giá kết quả
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred, digits=2))
```

3 mẫu nấm ăn được bị phân loại nhầm là nấm độc → không nguy hiểm.

35 mẫu nấm độc bị phân loại nhầm là ăn được → nguy hiểm (False Negative cao hơn mong muốn).

Precision lớp 1 (nấm độc) = 0.99
99% các mẫu dự đoán là độc thực sự là độc.

Recall lớp 1 (nấm độc) = 0.92
Mô hình nhận diện được 92% nấm độc thật, còn 8% bị bỏ sót (nguy hiểm trong thực tế).

F1-score lớp 1 = 0.95
Mức cân bằng giữa precision và recall cho nấm độc là rất cao, nhưng chưa hoàn hảo.

Accuracy = 97%
Tổng thể rất tốt, nhưng không an toàn tuyệt đối (ví dụ: chọn sai nấm độc → nguy hiểm).

5 Mô hình Logistic Regression

Logistic Regression là một mô hình học máy dùng để phân loại nhị phân, dựa trên việc ước lượng xác suất đầu ra thuộc về một lớp bằng hàm sigmoid.
Mặc dù có tên là "Regression", Logistic Regression là mô hình phân loại, không phải hồi quy.

a. Hàm dự đoán xác suất:

Cho đầu vào đặc trưng X = (x1, x2 ,x3, …xn) và trọng số β

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n = \beta^T X$$

Dự đoán xác suất bằng hàm sigmoid:

$$\hat{p} = \frac{1}{1 + e^{-z}} = \frac{1}{1 + e^{-(\beta^T X)}}$$

b. Hàm mất mát – Binary Cross-Entropy (Log Loss):

Mục tiêu của mô hình là tối thiểu hóa tổng hàm mất mát qua toàn bộ tập dữ liệu.

$$L = -\left[y \cdot \log(\hat{p}) + (1 - y) \cdot \log(1 - \hat{p})\right]$$

Ưu điểm

- Dễ hiểu, diễn giải rõ ràng
- Huấn luyện nhanh
- Dự đoán xác suất tốt

Nhược điểm

- Giả định tuyến tính giữa đặc trưng và logit (log-odds)
- Không tốt với dữ liệu phức tạp, phi tuyến
- Nhạy với dữ liệu nhiễu, mất cân bằng
- Có thể underfit nếu mô hình quá đơn giản

```
Confusion Matrix:
 [[689  16]
 [ 31 393]]

Classification Report:
              precision    recall  f1-score   support

           0       0.96      0.98      0.97       705
           1       0.96      0.93      0.94       424

    accuracy                           0.96      1129
   macro avg       0.96      0.95      0.96      1129
weighted avg       0.96      0.96      0.96      1129
```

```python
# Đọc dữ liệu
columns = [
    'class', 'cap-shape', 'cap-surface', 'cap-color', 'bruises', 'odor',
    'gill-attachment', 'gill-spacing', 'gill-size', 'gill-color',
    'stalk-shape', 'stalk-root', 'stalk-surface-above-ring',
    'stalk-surface-below-ring', 'stalk-color-above-ring',
    'stalk-color-below-ring', 'veil-type', 'veil-color', 'ring-number',
    'ring-type', 'spore-print-color', 'population', 'habitat'
]

df = pd.read_csv("mushroom/agaricus-lepiota.data", names=columns)
df.drop(columns=["veil-type"], inplace=True)
df.replace("?", pd.NA, inplace=True)
df.dropna(inplace=True)

# Mã hóa dữ liệu
le = LabelEncoder()
for col in df.columns:
    df[col] = le.fit_transform(df[col])
# Tách tập train/test
X = df.drop("class", axis=1)
y = df["class"]

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Huấn luyện Logistic Regression
model = LogisticRegression(max_iter=1000)
model.fit(X_train, y_train)

# Dự đoán và đánh giá
y_pred = model.predict(X_test)

print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred, digits=2))
```

Precision = 0.96 cho cả hai lớp → Mô hình đoán chính xác cao với cả nấm độc và nấm ăn được.

Recall lớp 1 (nấm độc) = 0.93 → Mô hình bỏ sót 7% nấm độc, có 31 mẫu nấm độc bị phân loại sai là ăn được → có thể gây nguy hiểm nếu áp dụng thực tế.

Accuracy tổng thể = 96% → Mô hình hoạt động tốt, nhưng thấp hơn một chút so với Random Forest hay Naive Bayes (nếu bạn đã chạy trước đó).

PHẦN KẾT QUẢ

1 Kết quả thực hiện

	precision	recall	accuracy
Decision tree	100%	100%	100%
Random forest	100%	100%	100%
Naïve Bayes	99%	92%	97%
Logistic Regression	96%	93%	96%

2 Đánh giá mô hình

Tree-based models như Decision Tree, Random Forest hoạt động rất hiệu quả, đạt accuracy tuyệt đối trên tập Mushroom.

Naive Bayes tuy đơn giản nhưng vẫn có kết quả tốt (97%), phù hợp khi cần tốc độ và ít tài nguyên.

Logistic Regression có độ chính xác cao (96%), nhưng bị giới hạn khi xử lý quan hệ phi tuyến.