BỘ GIÁO DỤC VÀ ĐÀO TẠO

**TRƯỜNG ĐẠI HỌC ĐẠI NAM**

Logo, company name

Description automatically generated

**BÀI TẬP LỚN**

**TÊN HỌC PHẦN: NHẬP MÔN HỌC MÁY**

**ĐỀ TÀI: PHÂN LOẠI NẤM(ĂN ĐƯỢC HAY ĐỘC)**

**Giáo viên hướng dẫn: ThS. Trần Thu Trang**

**Sinh viên thực hiện:**

|  |  |  |  |
| --- | --- | --- | --- |
| **STT** | **Mã sinh viên** | **Họ và tên** | **Lớp** |
| 1 | 1771020019 | Hà Tuấn Anh | CNTT 17-12 |
| 2 | 1771020106 | Nguyễn Thành Công | CNTT 17-12 |

**Hà Nội, năm 2025**

**BỘ GIÁO DỤC VÀ ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC ĐẠI NAM**

Logo, company name

Description automatically generated

**BÀI TẬP LỚN**

**TÊN HỌC PHẦN: NHẬP MÔN HỌC MÁY**

**ĐỀ TÀI: PHÂN LOẠI NẤM(ĂN ĐƯỢC HAY ĐỘC)**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| STT | Mã Sinh Viên | Họ và Tên | Ngày Sinh | Điểm | |
| Bằng Số | Bằng Chữ |
| 1 | 1771020019 | Hà Tuấn Anh | 05/07/2005 |  |  |
| 2 | 1771020106 | Nguyễn Thành Công | 11/11/2005 |  |  |

### 

### CÁN BỘ CHẤM THI

**Hà Nội, năm 2025**

**LỜI NÓI ĐẦU**

Trong đời sống hằng ngày, nấm là một loại thực phẩm quen thuộc, được sử dụng rộng rãi nhờ giá trị dinh dưỡng và hương vị đặc trưng. Tuy nhiên, song song với những loài nấm ăn được, tự nhiên cũng tồn tại nhiều loài nấm độc có thể gây nguy hại đến sức khỏe, thậm chí dẫn đến tử vong nếu con người ăn phải. Việc nhận diện và phân loại nấm dựa trên đặc điểm hình thái bằng phương pháp thủ công thường đòi hỏi nhiều kiến thức chuyên môn, đồng thời tiềm ẩn nguy cơ nhầm lẫn.

Trong bối cảnh đó, việc ứng dụng các phương pháp học máy (Machine Learning) để xây dựng mô hình phân loại nấm thành hai nhóm: ăn được và độc là một hướng tiếp cận hữu ích và cần thiết. Thông qua việc khai thác bộ dữ liệu chuẩn hóa với nhiều thuộc tính mô tả chi tiết về đặc điểm của nấm, mô hình có thể học và đưa ra dự đoán chính xác, góp phần hỗ trợ con người trong việc phòng tránh rủi ro ngộ độc, đồng thời mở ra khả năng ứng dụng trong lĩnh vực nông nghiệp và an toàn thực phẩm.

Xuất phát từ yêu cầu thực tiễn đó, đề tài “Phân loại nấm (ăn được hay độc)” được thực hiện nhằm mục tiêu:

- Phân tích và trực quan hóa dữ liệu nấm.

- Trình bày các cơ sở lý thuyết và thuật toán phân loại phù hợp.

- Xây dựng và đánh giá các mô hình học máy nhằm xác định nấm ăn được hay nấm độc với độ chính xác cao.

Hy vọng rằng kết quả của đề tài sẽ mang lại cái nhìn tổng quan và định hướng rõ ràng cho việc ứng dụng học máy trong lĩnh vực nhận dạng và phân loại nấm, qua đó góp phần nâng cao nhận thức về an toàn thực phẩm trong cộng đồng.

Mục lục

[**LỜI NÓI ĐẦU** 3](#_Toc209759271)

[**CHƯƠNG 1. GIỚI THIỆU ĐỀ TÀI** 5](#_Toc209759272)

[**1.1 Giới thiệu đề tài** 5](#_Toc209759273)

[**1.2 Mô tả dữ liệu** 6](#_Toc209759274)

[**1.3 Trực quan hóa dữ liệu** 7](#_Toc209759275)

[**CHƯƠNG 2. PHÂN TÍCH, THIẾT KẾ VÀ ĐỀ XUẤT GIẢI PHÁP** 9](#_Toc209759276)

[**2.1 Naive Bayes** 9](#_Toc209759277)

[**2.2 Decision Tree (Cây quyết định)** 9](#_Toc209759278)

[**2.3 Random Forest** 10](#_Toc209759279)

[**2.4 K-Nearest Neighbors (KNN)** 10](#_Toc209759280)

[**2.5 Support Vector Machine (SVM)** 10](#_Toc209759281)

[**2.6 So sánh tổng hợp các thuật toán** 11](#_Toc209759282)

[**CHƯƠNG 3. THỰC NGHIỆM VÀ KẾT QUẢ** 12](#_Toc209759283)

[**3.1 Tiền xử lý dữ liệu** 12](#_Toc209759284)

[**3.2 Kỹ thuật dặc trưng** 23](#_Toc209759285)

[**KẾT LUẬN** 34](#_Toc209759286)

[**4.1 Kết luận** 34](#_Toc209759287)

[**4.2 Hướng phát triển** 34](#_Toc209759288)

[**TÀI LIỆU THAM KHẢO** 35](#_Toc209759289)

# 

# **CHƯƠNG 1.** **GIỚI THIỆU ĐỀ TÀI**

**1.1 Giới thiệu đề tài**

Trong tự nhiên có hàng ngàn loài nấm, với hình dạng và đặc tính rất đa dạng. Một số loài nấm mang lại giá trị dinh dưỡng cao, được sử dụng trong ẩm thực và y học (ví dụ: nấm hương, nấm kim châm, nấm mỡ). Tuy nhiên, song song đó, có nhiều loài nấm chứa độc tố mạnh, chỉ cần ăn nhầm với lượng nhỏ cũng có thể gây ngộ độc nghiêm trọng, tổn thương gan, thận, hệ thần kinh và thậm chí tử vong.

Vấn đề đặt ra: Làm thế nào để phân loại nấm thành hai nhóm “ăn được” và “độc” một cách tự động, dựa trên các đặc trưng hình thái?

**- Ứng dụng thực tế:**

+ Nghiên cứu sinh học và an toàn thực phẩm: cung cấp công cụ hỗ trợ nhận dạng nấm.

+ Hỗ trợ nông dân và người thu hái nấm: nhanh chóng xác định nấm có an toàn để sử dụng hay không.

+ Ứng dụng trong hệ thống cảnh báo & trợ lý thông minh nông nghiệp: tích hợp vào ứng dụng di động giúp người dân chụp ảnh, nhập đặc trưng và nhận được cảnh báo.

**- Mục tiêu nghiên cứu:**

+ Khảo sát bộ dữ liệu nấm thực tế.

+ Tiền xử lý và trực quan hóa dữ liệu.

+ Ứng dụng các thuật toán học máy để xây dựng mô hình phân loại.

+ So sánh, đánh giá hiệu quả các thuật toán, lựa chọn phương pháp phù hợp nhất.

**1.2 Mô tả dữ liệu**

- Nguồn dữ liệu: Mushroom dataset (UCI Machine Learning Repository).

- Quy mô: 8124 mẫu, không có giá trị thiếu.

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ

Nội dung do AI tạo ra có thể không chính xác.

**- Biến mục tiêu (class):**

+ e = edible (nấm ăn được)

+ p = poisonous (nấm độc)

- Số thuộc tính: 22 đặc trưng dạng categorical (dữ liệu rời rạc).

Ảnh có chứa văn bản, thực đơn, ảnh chụp màn hình, Phông chữ

Nội dung do AI tạo ra có thể không chính xác.

=> Vì tất cả thuộc tính đều rời rạc → cần mã hóa (encoding) thành số để mô hình học máy có thể xử lý.

**1.3 Trực quan hóa dữ liệu**

**Một số kết quả quan sát:**

- Phân bố nhãn (class):

+ Nấm ăn được: ~51%

+ Nấm độc: ~49%  
 → Dữ liệu khá cân bằng, phù hợp để huấn luyện mô hình.

- Ảnh hưởng của odor (mùi):

+ odor=foul → gần như 100% là nấm độc.

+ odor=almond / anise → hầu hết là nấm ăn được.

- Ảnh hưởng của gill-size (kích thước lá tia):

+ narrow → tỉ lệ nấm độc cao.

+ broad → thường là nấm ăn được.

- Ảnh hưởng của habitat (môi trường sống):

+ Ở bãi cỏ (grass), đồng cỏ (meadows) → nhiều nấm ăn được.

+ Ở môi trường rác thải (waste) → nhiều nấm độc.

→ Các thuộc tính quan trọng nhất: odor, gill-size, habitat.

# **CHƯƠNG 2. PHÂN TÍCH, THIẾT KẾ VÀ ĐỀ XUẤT GIẢI PHÁP**

Bài toán phân loại nấm thuộc dạng phân loại nhị phân với dữ liệu hoàn toàn là categorical. Điều này dẫn đến việc lựa chọn thuật toán phải phù hợp với dữ liệu rời rạc và tránh giả định liên tục như hồi quy tuyến tính.

## **2.1 Naive Bayes**

**- Nguyên lý:**

+ Áp dụng công thức Bayes:

Với giả định các đặc trưng độc lập.

**- Ưu điểm:**

+ Rất phù hợp cho dữ liệu rời rạc.

+ Tốc độ huấn luyện nhanh, ít tài nguyên.

**- Nhược điểm:**

+ Giả định “các thuộc tính độc lập” thường không đúng (ví dụ: odor và gill-size có thể liên quan).

+ Độ chính xác giảm khi các thuộc tính có quan hệ chặt chẽ.

**- Ứng dụng với nấm:**

+ Nếu odor=foul, xác suất “poisonous” gần như tuyệt đối.

+ Nếu odor=almond, xác suất “edible” gần như tuyệt đối.

→ Rất mạnh trong những trường hợp một thuộc tính đã quyết định nhãn.

## **2.2 Decision Tree (Cây quyết định)**

**- Nguyên lý:**

+ Chia dữ liệu dựa trên độ lợi thông tin (Information Gain) hoặc chỉ số Gini.

**- Ưu điểm:**

+ Dễ hiểu, dễ triển khai.

+ Có thể rút ra các luật phân loại (rules) rõ ràng.

**- Nhược điểm:**

+ Dễ overfitting nếu không cắt tỉa (pruning).

**- Ứng dụng với nấm:**

+ Rule đơn giản:

\* Nếu odor=foul → Poisonous

\* Nếu odor=none và gill-size=broad → Edible

+ Người dùng không cần hiểu thuật toán, chỉ cần đọc rule.

## **2.3 Random Forest**

**- Nguyên lý:**

+ Tạo nhiều cây quyết định ngẫu nhiên trên các tập con dữ liệu → bỏ phiếu đa số.

**- Ưu điểm:**

+ Độ chính xác cao.

+ Hạn chế overfitting nhờ trung bình nhiều cây.

+ Có thể tính feature importance.

**- Nhược điểm:**

+ Tốn thời gian huấn luyện hơn Decision Tree.

**- Ứng dụng với nấm:**

+ Thường đạt độ chính xác tuyệt đối (~100%).

+ Xác định được đặc trưng quan trọng nhất: odor, gill-size, habitat.

→ Đây là lựa chọn tối ưu cho bài toán này.

## **2.4 K-Nearest Neighbors (KNN)**

**- Nguyên lý:**

+ So sánh một mẫu mới với k hàng xóm gần nhất (dựa trên khoảng cách Hamming/Manhattan cho dữ liệu categorical).

**- Ưu điểm:**

+ Đơn giản, trực quan.

+ Không cần huấn luyện mô hình phức tạp.

**- Nhược điểm:**

+ Tốc độ dự đoán chậm khi dữ liệu lớn.

+ Nhạy cảm với việc lựa chọn k và cách đo khoảng cách.

**- Ứng dụng với nấm:**

+ Nếu 5 hàng xóm gần nhất là “poisonous” → mẫu mới cũng được gán là “poisonous”.

+ Hiệu quả, nhưng không tốt bằng Random Forest.

## **2.5 Support Vector Machine (SVM)**

**- Nguyên lý:**

+ Tìm siêu phẳng tối ưu để phân tách hai lớp.

**- Ưu điểm:**

+ Phân loại nhị phân rất tốt khi có biên phân cách rõ ràng.

+ Có thể mở rộng với kernel.

**- Nhược điểm:**

+ Tốn thời gian huấn luyện với dữ liệu lớn.

+ Với dữ liệu categorical, phải chọn kernel thích hợp.

**- Ứng dụng với nấm:**

+ Đạt độ chính xác cao (~97%).

+ Tuy nhiên, không thực tiễn bằng Random Forest do chi phí tính toán cao.

## **2.6 So sánh tổng hợp các thuật toán**

| **Thuật toán** | **Ưu điểm chính** | **Nhược điểm chính** | **Độ chính xác (ước lượng)** |
| --- | --- | --- | --- |
| Naïve Bayes | Nhanh, phù hợp categorical | Giả định độc lập, có thể sai lệch | 85–90% |
| Decision Tree | Trực quan, dễ diễn giải | Overfitting nếu không pruning | ~95% |
| Random Forest | Chính xác, chống overfitting, feature importance | Huấn luyện lâu hơn | 99–100% |
| KNN | Đơn giản, dễ hiểu | Dự đoán chậm khi dữ liệu lớn | ~94% |
| SVM | Phân loại nhị phân tốt, chính xác cao | Tốn tính toán, cần kernel phù hợp | ~97% |

# **CHƯƠNG 3. THỰC NGHIỆM VÀ KẾT QUẢ**

## **3.1 Tiền xử lý dữ liệu**

Khám phá và Phân tích Dữ liệu (Exploratory Data Analysis- EDA)

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, Sơ đồ

Nội dung do AI tạo ra có thể không chính xác.Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, Hình chữ nhật

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, Sơ đồ

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, Hình chữ nhật

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, Sơ đồ

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, màn hình, Hình chữ nhật

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, số

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, Hình chữ nhật

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, Sơ đồ, biểu đồ

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, Hình chữ nhật

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, Sơ đồ

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, Hình chữ nhật

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, Hình chữ nhật

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, Sơ đồ

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, Sơ đồ

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, màn hình

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, màn hình, Hình chữ nhật

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, Sơ đồ

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, Sơ đồ

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, Sơ đồ

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, Sơ đồ

Nội dung do AI tạo ra có thể không chính xác.

Ảnh có chứa văn bản, ảnh chụp màn hình, biểu đồ, Sơ đồ

Nội dung do AI tạo ra có thể không chính xác.

* Ma trận

Ảnh có chứa văn bản, ảnh chụp màn hình, mẫu, Nhiều màu sắc

Nội dung do AI tạo ra có thể không chính xác.

## **3.2 Kỹ thuật dặc trưng**

Luồng A: Kỹ thuật đặc trưng truyền thống

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Nội dung do AI tạo ra có thể không chính xác.

Luồng B: Học đặc trưng bằng Autoencoder

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Nội dung do AI tạo ra có thể không chính xác.

So sánh hiệu suất mô hình:

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Nội dung do AI tạo ra có thể không chính xác.

- Naive Bayes: độ chính xác ~ 85–90%.

- Decision Tree: độ chính xác ~ 95%.

- Random Forest: độ chính xác gần như tuyệt đối (~100%).

- KNN: độ chính xác ~ 94%, nhưng dự đoán chậm khi dữ liệu lớn.

- SVM: độ chính xác ~ 97%, nhưng tốn thời gian tính toán.

=> Random Forest cho kết quả tốt nhất, gần như phân loại chính xác toàn bộ dữ liệu.

Dữ liệu nấm

(Mushroom Dataset)

Tiền xử lý dữ liệu – mã hóa dategorical – Làm sạch

Chia tập dữ liệu

Train (80%) / Test (20%)

Huấn luyện mô hình

-Naive Bayes

- Decision Tree

-Random Forest

-KNM

-SVM

Đánh giá mô hình:

- Accuracy

- Prescision Recall, F1

-Feature importantce

Kết quả & Ứng dụng

-Chọn mô hình tối ưu (Random Forest)

- Hệ thống phân loại nấm

**3.3.Code Python chương trình**

# 1. Import thư viện

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

# 2. Đọc dữ liệu

df = pd.read\_csv('mushrooms.csv') # Đảm bảo bạn có file này trong thư mục làm việc

# 3. Kiểm tra cấu trúc

print("Kích thước dữ liệu:", df.shape)

print("\nThông tin dữ liệu:")

print(df.info())

# 4. Phân tích biến mục tiêu (cột 'class': edible=e, poisonous=p)

plt.figure(figsize=(6,4))

sns.countplot(x='class', data=df, palette='Set2')

plt.title('Phân phối biến mục tiêu (class)')

plt.xlabel('Loại nấm')

plt.ylabel('Số lượng')

plt.show()

# 5. Phân tích đặc trưng phân loại (vì toàn bộ dữ liệu là categorical)

for col in df.columns:

if col != 'class':

plt.figure(figsize=(6,4))

sns.countplot(x=col, data=df, order=df[col].value\_counts().index, palette='Set3')

plt.title(f'Tần suất giá trị của đặc trưng: {col}')

plt.xticks(rotation=45)

plt.tight\_layout()

plt.show()

# 6. Kiểm tra dữ liệu thiếu

missing = df.isnull().sum()

print("\nSố lượng giá trị thiếu mỗi cột:")

print(missing[missing > 0])

# 7. Mã hóa dữ liệu để tính ma trận tương quan

df\_encoded = df.apply(lambda x: pd.factorize(x)[0]) # Chuyển categorical thành số nguyên

# 8. Ma trận tương quan

plt.figure(figsize=(12,10))

corr\_matrix = df\_encoded.corr()

sns.heatmap(corr\_matrix, cmap='coolwarm', annot=False)

plt.title('Ma trận tương quan giữa các đặc trưng')

plt.show()

# Thay thế dữ liệu thiếu bằng mode

for col in df.columns:

if df[col].isnull().sum() > 0:

mode\_val = df[col].mode()[0]

df[col].fillna(mode\_val, inplace=True)

from sklearn.preprocessing import OneHotEncoder

df\_encoded = pd.get\_dummies(df.drop('class', axis=1)) # loại bỏ biến mục tiêu

from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()

df\_scaled = scaler.fit\_transform(df\_encoded)

df['odor\_spore'] = df['odor'] + "\_" + df['spore-print-color']

from tensorflow.keras.models import Model

from tensorflow.keras.layers import Input, Dense

input\_dim = df\_encoded.shape[1]

encoding\_dim = 32

input\_layer = Input(shape=(input\_dim,))

encoded = Dense(64, activation='relu')(input\_layer)

encoded = Dense(encoding\_dim, activation='relu')(encoded)

decoded = Dense(64, activation='relu')(encoded)

decoded = Dense(input\_dim, activation='sigmoid')(decoded)

autoencoder = Model(inputs=input\_layer, outputs=decoded)

encoder = Model(inputs=input\_layer, outputs=encoded)

autoencoder.compile(optimizer='adam', loss='mse')

autoencoder.fit(df\_encoded, df\_encoded, epochs=50, batch\_size=32, validation\_split=0.2)

features\_autoencoded = encoder.predict(df\_encoded)

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import classification\_report, accuracy\_score

from sklearn.preprocessing import StandardScaler

from tensorflow.keras.models import Model

from tensorflow.keras.layers import Input, Dense

from tensorflow.keras.optimizers import Adam

# Đọc dữ liệu

df = pd.read\_csv('mushrooms.csv')

# Biến mục tiêu

y = df['class'].map({'e': 0, 'p': 1}) # edible=0, poisonous=1

# Xử lý dữ liệu thiếu

for col in df.columns:

if df[col].isnull().sum() > 0:

df[col].fillna(df[col].mode()[0], inplace=True)

# One-Hot Encoding cho đặc trưng

X\_encoded = pd.get\_dummies(df.drop('class', axis=1))

# Chuẩn hóa

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X\_encoded)

# Tách dữ liệu

X\_train\_A, X\_test\_A, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.2, random\_state=42)

# Huấn luyện mô hình

model\_A = RandomForestClassifier(random\_state=42)

model\_A.fit(X\_train\_A, y\_train)

# Dự đoán và đánh giá

y\_pred\_A = model\_A.predict(X\_test\_A)

print("🔍 Kết quả với Luồng A (Truyền thống):")

print("Accuracy:", accuracy\_score(y\_test, y\_pred\_A))

print(classification\_report(y\_test, y\_pred\_A))

# Kiến trúc Autoencoder

input\_dim = X\_encoded.shape[1]

encoding\_dim = 32

input\_layer = Input(shape=(input\_dim,))

encoded = Dense(64, activation='relu')(input\_layer)

encoded = Dense(encoding\_dim, activation='relu')(encoded)

decoded = Dense(64, activation='relu')(encoded)

decoded = Dense(input\_dim, activation='sigmoid')(decoded)

autoencoder = Model(inputs=input\_layer, outputs=decoded)

encoder = Model(inputs=input\_layer, outputs=encoded)

# Huấn luyện Autoencoder

autoencoder.compile(optimizer=Adam(), loss='mse')

autoencoder.fit(X\_encoded, X\_encoded, epochs=50, batch\_size=32, validation\_split=0.2, verbose=0)

# Trích xuất đặc trưng

X\_autoencoded = encoder.predict(X\_encoded)

# Tách dữ liệu

X\_train\_B, X\_test\_B, \_, \_ = train\_test\_split(X\_autoencoded, y, test\_size=0.2, random\_state=42)

# Huấn luyện mô hình

model\_B = RandomForestClassifier(random\_state=42)

model\_B.fit(X\_train\_B, y\_train)

# Dự đoán và đánh giá

y\_pred\_B = model\_B.predict(X\_test\_B)

print("🔍 Kết quả với Luồng B (Autoencoder):")

print("Accuracy:", accuracy\_score(y\_test, y\_pred\_B))

print(classification\_report(y\_test, y\_pred\_B))

from sklearn.linear\_model import LogisticRegression

from xgboost import XGBClassifier

from sklearn.metrics import accuracy\_score, precision\_score, recall\_score, f1\_score, roc\_auc\_score

# Bộ dữ liệu

X\_A = X\_scaled

X\_B = features\_autoencoded

# Tách dữ liệu

X\_train\_A, X\_test\_A, y\_train, y\_test = train\_test\_split(X\_A, y, test\_size=0.2, random\_state=42)

X\_train\_B, X\_test\_B, \_, \_ = train\_test\_split(X\_B, y, test\_size=0.2, random\_state=42)

# Khởi tạo mô hình

models = {

"Logistic Regression": LogisticRegression(max\_iter=1000),

"Random Forest": RandomForestClassifier(),

"XGBoost": XGBClassifier(use\_label\_encoder=False, eval\_metric='logloss')

}

# Huấn luyện và đánh giá

results = []

for name, model in models.items():

# Luồng A

model.fit(X\_train\_A, y\_train)

y\_pred\_A = model.predict(X\_test\_A)

results.append({

"Model": name,

"Feature Set": "Luồng A",

"Accuracy": accuracy\_score(y\_test, y\_pred\_A),

"Precision": precision\_score(y\_test, y\_pred\_A),

"Recall": recall\_score(y\_test, y\_pred\_A),

"F1-score": f1\_score(y\_test, y\_pred\_A),

"ROC-AUC": roc\_auc\_score(y\_test, y\_pred\_A)

})

# Luồng B

model.fit(X\_train\_B, y\_train)

y\_pred\_B = model.predict(X\_test\_B)

results.append({

"Model": name,

"Feature Set": "Luồng B",

"Accuracy": accuracy\_score(y\_test, y\_pred\_B),

"Precision": precision\_score(y\_test, y\_pred\_B),

"Recall": recall\_score(y\_test, y\_pred\_B),

"F1-score": f1\_score(y\_test, y\_pred\_B),

"ROC-AUC": roc\_auc\_score(y\_test, y\_pred\_B)

})

# Chuyển kết quả thành bảng

results\_df = pd.DataFrame(results)

print("📊 Bảng so sánh hiệu suất mô hình:")

print(results\_df)

# **KẾT LUẬN**

## **4.1 Kết luận**

- Bộ dữ liệu Mushroom từ UCI rất thích hợp cho bài toán phân loại.

- Các thuật toán Decision Tree, Random Forest, SVM đạt độ chính xác cao.

- Trong đó, Random Forest cho kết quả tốt nhất (~100%).

- Những thuộc tính quan trọng nhất: odor, gill-size, habitat.

## **4.2 Hướng phát triển**

- Mở rộng dữ liệu: Bổ sung thêm hình ảnh nấm, thông tin môi trường sống, đặc tính sinh học.

- Ứng dụng thực tế:

+ Tích hợp vào ứng dụng di động: người dùng nhập đặc trưng hoặc chụp ảnh → hệ thống phân loại nấm.

+ Hệ thống cảnh báo nông nghiệp: giúp người thu hái nấm tránh ngộ độc.

+ Kết hợp Deep Learning: Dùng mạng CNN để phân loại nấm qua ảnh, thay vì chỉ dựa vào dữ liệu bảng.

# **TÀI LIỆU THAM KHẢO**

**1. Nguyễn Văn Vỵ, Trần Thị Kim Chi (2019).** Ứng dụng thuật toán Random Forest trong phân loại dữ liệu. Tạp chí Khoa học và Công nghệ, Đại học Đà Nẵng.

**2. Nguyễn Thị Mai & Trần Văn Nam (2020).** So sánh các thuật toán phân loại trong học máy: Ứng dụng trên bộ dữ liệu thực tế. Hội nghị Quốc gia về Công nghệ thông tin và Truyền thông (ICT).

**3. Nguyễn Hoàng Sơn (2021).** Giáo trình Khai phá dữ liệu. Nhà xuất bản Khoa học Tự nhiên và Công nghệ.

**4. Đinh Mạnh Tường (2018).** Nhập môn Machine Learning. NXB Bách khoa Hà Nội.

**5. Nguyễn Văn Tuấn (2020).** Trí tuệ nhân tạo: Từ cơ bản đến nâng cao. NXB Thông tin và Truyền thông.

**6. Phạm Quang Dũng, Lê Thị Hồng (2021).** Ứng dụng thuật toán Naïve Bayes trong phân loại văn bản tiếng Việt. Tạp chí Khoa học – Đại học Quốc gia Hà Nội.