

Hồ Đức Vũ

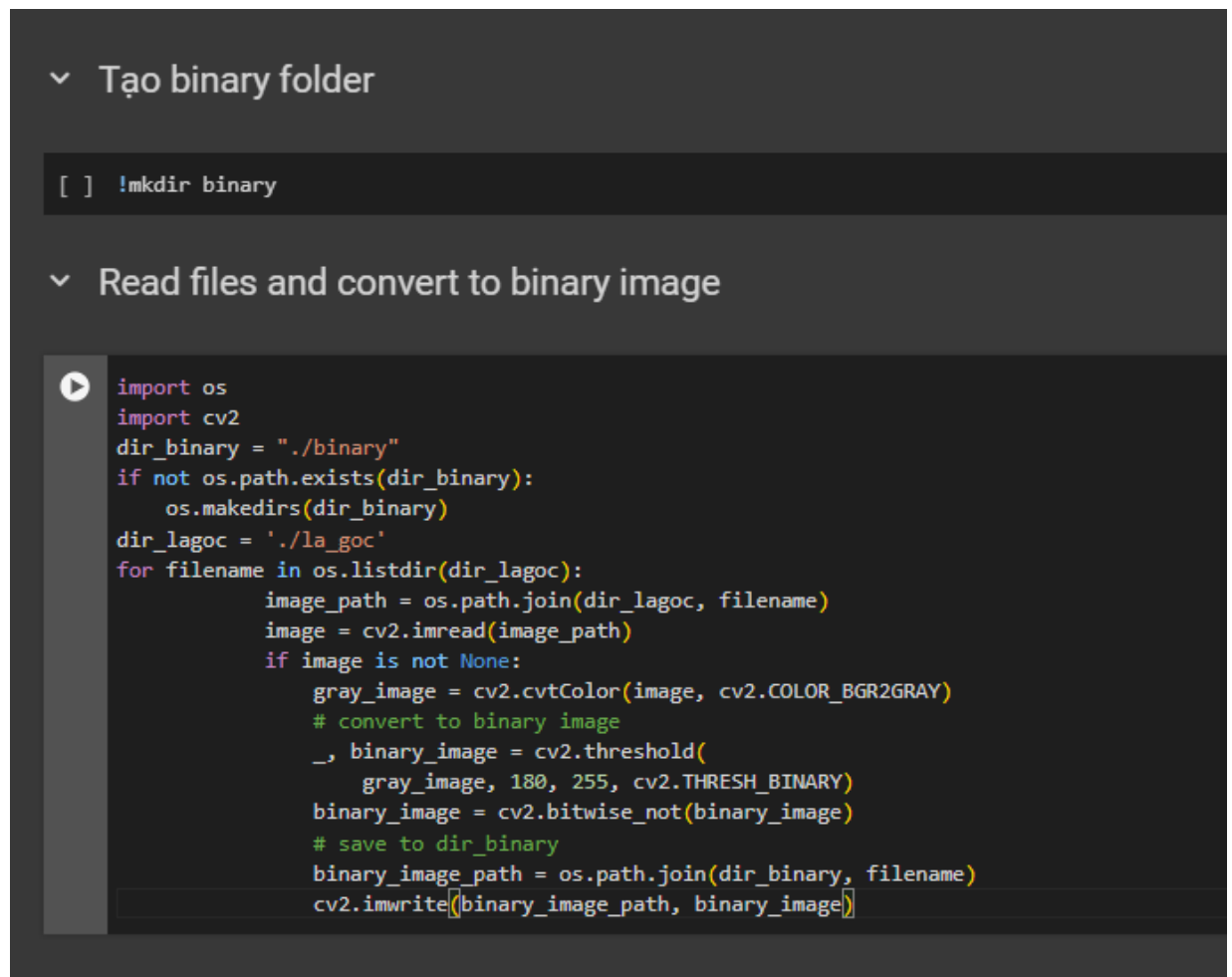
BÁO CÁO BÀI TẬP CODING CUỐI KỲ

MÔN HỌC: TRÍ TUỆ NHÂN TẠO

STT: 1 - Nhóm 3

Các bước tiến hành:

1. Tạo và tải dữ liệu đến thư mục la_goc .
2. Tạo thư mục binary.
3. Đọc ảnh từ thư mục la_goc và chuyển sang dạng nhị phân sau đó lưu vào thư mục binary có cùng tên file ảnh với la_goc.



```
▼ Tạo binary folder

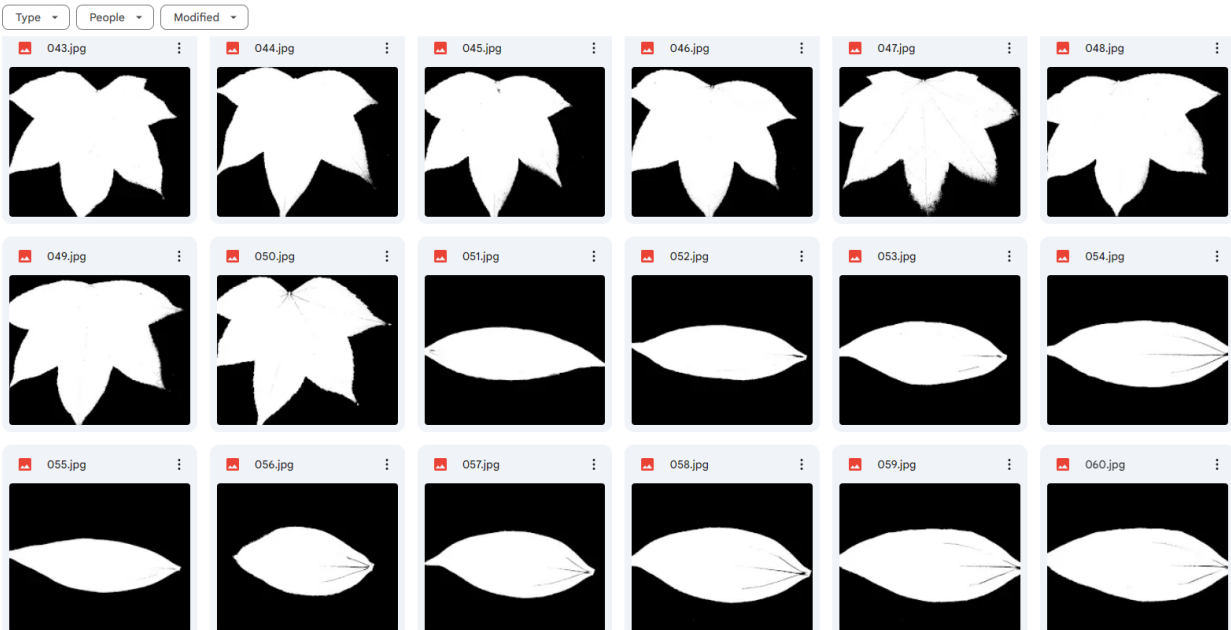
[ ] !mkdir binary

▼ Read files and convert to binary image

import os
import cv2
dir_binary = "./binary"
if not os.path.exists(dir_binary):
    os.makedirs(dir_binary)
dir_lagoc = './la_goc'
for filename in os.listdir(dir_lagoc):
    image_path = os.path.join(dir_lagoc, filename)
    image = cv2.imread(image_path)
    if image is not None:
        gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        # convert to binary image
        _, binary_image = cv2.threshold(
            gray_image, 180, 255, cv2.THRESH_BINARY)
        binary_image = cv2.bitwise_not(binary_image)
        # save to dir_binary
        binary_image_path = os.path.join(dir_binary, filename)
        cv2.imwrite(binary_image_path, binary_image)
```

Hình 1: Code chuyển ảnh sang dạng nhị phân và lưu vào binary folder.

Kết quả của bước này được thể hiện tại hình 2.



Hình 2: Kết quả chuyển bộ dữ liệu ảnh sang nhị phân.

4. Tính Hu's moment và lưu vào file Excel

Tại bước này ta sẽ đọc ảnh từ thư mục binary, gán nhãn cho từng ảnh với quy tắc: ảnh từ 1 đến 25 được gán nhãn 'Nanmu', ảnh từ 26 đến 50 được gán nhãn là 'castor_aralia', ảnh từ 51 đến 75 được gán nhãn là 'Chinese_cinamon', và 25 ảnh còn lại được gán nhãn 'southern_magnolia'.

Sau khi gán nhãn cho các ảnh, bước tiếp theo ta tính Hu's moment cho từng ảnh, sử dụng thư viện open-cv với hàm **cv2.moment()** và **cv2.HuMoments()** để lấy giá 7 giá trị hu's moment cho từng ảnh, sau khi có được hu's moment và các nhãn thì ta sẽ gán nó cho mảng **data** và **labels**.

Bước tiếp theo sử dụng thư viện panda để cấu trúc dữ liệu và lưu vào file hu_moments.csv.

Coding của phần này được biểu diễn tại hình 3.

```
Trích đặc trưng Hu's moments

import os
import cv2
import pandas as pd

def extract_hu_moments(image):
    moments = cv2.moments(image)
    hu_moments = cv2.HuMoments(moments).flatten().tolist()
    return hu_moments

data = []
labels = []
for filename in os.listdir(dir_binary):
    file_number = int(filename.split('.')[0])
    if 1 <= file_number <= 25:
        label = 'Nanmu'
    elif 26 <= file_number <= 50:
        label = 'castor_aralia'
    elif 51 <= file_number <= 75:
        label = 'Chinese_cinnamon'
    else:
        label = 'southern_magnolia'

    img_path = os.path.join(dir_binary, filename)
    img = cv2.imread(img_path, cv2.IMREAD_GRAYSCALE)
    hu_moments = extract_hu_moments(img)
    data.append([filename] + hu_moments)
    labels.append(label)

hu_moments_df = pd.DataFrame(data, columns=['image_name', 'hu_moment_1', 'hu_moment_2', 'hu_moment_3', 'hu_moment_4', 'hu_moment_5', 'hu_moment_6', 'hu_moment_7'])
hu_moments_df["label"] = labels

hu_moments_df.to_csv('hu_moments.csv', index=False)

hu_moments_df.head()
```

Hình 3: Code tính đặc trưng Hu's moments và gán nhãn cho ảnh.

Kết quả của phần này là một file .csv chứa các đặc trưng hu's moment của 100 ảnh. Ta có thể sử dụng **hu_moments_df.head()** để mô tả cấu trúc của file hu_moments.csv này. Kết quả được biểu diễn tại hình 4.

	image_name	hu_moment_1	hu_moment_2	hu_moment_3	hu_moment_4	hu_moment_5	hu_moment_6	hu_moment_7	label
0	042.jpg	0.000713	3.608962e-08	8.169893e-12	1.938231e-13	-1.036471e-25	-1.057900e-17	-2.207845e-25	castor_aralia
1	013.jpg	0.001161	9.356478e-07	8.184981e-11	4.658148e-11	2.875634e-21	4.477681e-14	6.016473e-23	Nanmu
2	011.jpg	0.001111	8.231197e-07	3.650098e-11	1.776342e-11	4.505324e-22	1.506277e-14	-4.013196e-23	Nanmu
3	024.jpg	0.001035	6.600958e-07	3.637574e-11	1.942419e-11	5.163139e-22	1.577219e-14	-2.838050e-24	Nanmu
4	060.jpg	0.000931	4.626538e-07	9.691558e-12	3.174879e-12	1.761019e-23	2.156575e-15	-1.835196e-25	Chinese_cinnamon

Hình 4: Mô tả hu_moments_df.

- Bước tiếp theo ta tách bộ dữ liệu thành 2 phần: tập train và tập test. Với tập train và tập test được chia tỉ lệ 4/1, mỗi class sẽ được chứa làm 20 ảnh train và 5 ảnh test, ta có thể xác định ảnh train là những ảnh có thứ tự nằm trong các khoảng từ 0 đến 20, 26 đến 45, 51 đến 70 và 76 đến 95. Nhưng ảnh nằm ngoài phạm vi trên sẽ được xếp vào tập test. Code phần này được thể hiện tại hình 5.

▼ Tách dữ liệu thành 2 phần: train và test

```
import pandas as pd

hu_moments_df = pd.read_csv('hu_moments.csv')

hu_moments_df = hu_moments_df.sort_values('image_name')

def split(image_name):
    number = int(image_name.split('.')[0])
    if (0 <= number <= 20) or (26 <= number <= 45) or (51 <= number <= 70) or (76 <= number <= 95):
        return 'train'
    else:
        return 'test'

hu_moments_df['set'] = hu_moments_df['image_name'].apply(split)
# lưu vào file hu_moments.csv
hu_moments_df.to_csv('hu_moments.csv', index=False)

train_count = len(hu_moments_df[hu_moments_df['set'] == 'train'])
test_count = len(hu_moments_df[hu_moments_df['set'] == 'test'])

pd.Series({
    "Training Samples:": train_count,
    "Testing Samples:": test_count
}).to_frame().style.hide(axis='columns')
```



Training Samples: 80

Testing Samples: 20

Hình 5: Coding chia dữ liệu thành 2 phần.

Kết quả của bước này là ta sẽ tạo ra một cột trong hu_moments.csv có tên là **set**, với tập train được gán 'train', và tập test được gán là 'test'. Hình 6 mô tả hu_moments_df (kết quả của bước này).

hu_moments_df.head()

	image_name	hu_moment_1	hu_moment_2	hu_moment_3	hu_moment_4	hu_moment_5	hu_moment_6	hu_moment_7	label	set
0	001.jpg	0.001078	7.458491e-07	5.277888e-11	2.657060e-11	9.945767e-22	2.236209e-14	-2.972548e-23	Nanmu	train
1	002.jpg	0.001156	9.253158e-07	6.339329e-11	3.668878e-11	1.769099e-21	3.518973e-14	-3.163390e-23	Nanmu	train
2	003.jpg	0.001018	6.299770e-07	1.459737e-11	6.761341e-12	6.695004e-23	5.126663e-15	-5.453013e-24	Nanmu	train
3	004.jpg	0.000990	5.671810e-07	4.217875e-11	1.580084e-11	4.022112e-22	1.049211e-14	6.796220e-23	Nanmu	train
4	005.jpg	0.001053	7.025325e-07	4.138894e-11	1.873803e-11	5.199723e-22	1.518703e-14	4.398098e-23	Nanmu	train

Next steps: [Generate code with hu_moments_df](#) [View recommended plots](#)

Hình 6: Mô tả hu_moments_df.

Hình 7 mô tả kết quả file hu_moments.csv hoàn chỉnh.

hu_moments.csv										
	A	B	C	D	E	F	G	H	I	J
1	image_name	hu_moment_1	hu_moment_2	hu_moment_3	hu_moment_4	hu_moment_5	hu_moment_6	hu_moment_7	label	set
2	001.jpg	0.001078257334	7.46E-07	5.28E-11	2.66E-11	9.95E-22	2.24E-14	-2.97E-23	Nanmu	train
3	002.jpg	0.001156335549	9.25E-07	6.34E-11	3.67E-11	1.77E-21	3.52E-14	-3.16E-23	Nanmu	train
4	003.jpg	0.001017733259	6.30E-07	1.46E-11	6.76E-12	6.70E-23	5.13E-15	-5.45E-24	Nanmu	train
5	004.jpg	0.000989666085	5.67E-07	4.22E-11	1.58E-11	4.02E-22	1.05E-14	6.80E-23	Nanmu	train
6	005.jpg	0.001053290489	7.03E-07	4.14E-11	1.87E-11	5.20E-22	1.52E-14	4.40E-23	Nanmu	train
7	006.jpg	0.001061090362	7.09E-07	5.82E-11	2.01E-11	6.62E-22	1.30E-14	-1.87E-22	Nanmu	train
8	007.jpg	0.0009431306951	4.53E-07	8.27E-11	3.82E-11	2.14E-21	2.53E-14	-8.83E-23	Nanmu	train
9	008.jpg	0.001004704966	5.97E-07	1.99E-11	4.96E-12	4.41E-23	2.06E-15	2.18E-23	Nanmu	train
10	009.jpg	0.0009968632375	5.89E-07	2.48E-11	1.10E-11	1.82E-22	8.17E-15	1.39E-23	Nanmu	train
11	010.jpg	0.0009971717947	5.51E-07	1.27E-10	6.02E-11	5.24E-21	4.21E-14	6.07E-22	Nanmu	train
12	011.jpg	0.00111090949	8.23E-07	3.65E-11	1.78E-11	4.51E-22	1.51E-14	-4.01E-23	Nanmu	train
13	012.jpg	0.001093550462	7.88E-07	3.69E-11	1.97E-11	5.32E-22	1.74E-14	-1.53E-23	Nanmu	train
14	013.jpg	0.001160748782	9.36E-07	8.18E-11	4.66E-11	2.88E-21	4.48E-14	6.02E-23	Nanmu	train
15	014.jpg	0.001082187241	7.68E-07	1.05E-11	6.06E-12	4.84E-23	5.31E-15	-5.55E-26	Nanmu	train
16	015.jpg	0.0009929311642	5.64E-07	5.04E-11	1.99E-11	6.23E-22	1.29E-14	-1.04E-22	Nanmu	train
17	016.jpg	0.001214246308	1.06E-06	5.51E-11	3.41E-11	1.48E-21	3.49E-14	3.94E-23	Nanmu	train
18	017.jpg	0.00102615173	6.37E-07	5.84E-11	3.01E-11	1.26E-21	2.39E-14	-2.76E-24	Nanmu	train
19	018.jpg	0.001011697928	6.08E-07	4.54E-11	2.29E-11	7.40E-22	1.78E-14	1.83E-23	Nanmu	train
20	019.jpg	0.0009680163041	5.04E-07	8.81E-11	4.39E-11	2.73E-21	3.11E-14	7.83E-24	Nanmu	train
21	020.jpg	0.001026011121	6.42E-07	1.91E-11	9.60E-12	1.30E-22	7.66E-15	-1.99E-24	Nanmu	train
22	021.jpg	0.001039974043	6.65E-07	4.16E-11	2.14E-11	6.38E-22	1.70E-14	3.79E-23	Nanmu	test
23	022.jpg	0.001025693184	6.42E-07	1.93E-11	9.67E-12	1.32E-22	7.72E-15	-2.15E-24	Nanmu	test
24	023.jpg	0.0009317797613	4.46E-07	3.22E-11	1.41E-11	3.00E-22	9.16E-15	-2.14E-23	Nanmu	test
25	024.jpg	0.001035260394	6.60E-07	3.64E-11	1.94E-11	5.16E-22	1.58E-14	-2.84E-24	Nanmu	test
26	025.jpg	0.001066826274	7.29E-07	3.35E-11	1.78E-11	4.34E-22	1.52E-14	-6.33E-24	Nanmu	test
27	026.jpg	0.0007141433121	1.98E-08	2.68E-11	2.20E-13	-2.28E-25	-2.20E-17	-4.81E-25	castor_aralia	train
28	027.jpg	0.0006987573276	1.20E-08	1.66E-11	1.21E-13	-6.68E-26	-4.84E-19	-1.57E-25	castor_aralia	train
29	028.jpg	0.0007124470183	1.65E-08	4.37E-12	7.19E-13	1.27E-24	-9.21E-17	1.41E-25	castor_aralia	train
30	029.jpg	0.0007033166798	3.20E-08	7.19E-12	7.87E-14	-3.00E-27	-2.98E-16	-5.91E-26	castor_aralia	train
31	030.jpg	0.0007031320383	1.47E-08	2.70E-11	2.39E-13	4.89E-25	-2.74E-17	3.64E-25	castor_aralia	train
32	031.jpg	0.000687111351	1.54E-08	3.97E-12	1.75E-13	1.39E-25	-2.17E-17	4.42E-26	castor_aralia	train
33	032.jpg	0.0007029343486	4.90E-09	1.60E-11	6.07E-13	1.80E-24	-3.90E-17	-5.73E-25	castor_aralia	train
34	033.jpg	0.0006866458334	1.23E-09	8.76E-12	1.03E-12	3.08E-24	-3.62E-17	3.53E-25	castor_aralia	train
35	034.jpg	0.0006983770347	1.22E-08	8.83E-12	3.85E-13	6.72E-25	-4.21E-17	2.27E-25	castor_aralia	train
36	035.jpg	0.0007067449157	9.14E-09	2.04E-11	3.77E-13	8.14E-25	-3.25E-17	6.58E-25	castor_aralia	train
37	036.jpg	0.0006977367056	9.44E-09	9.31E-12	3.51E-13	5.33E-25	-3.35E-17	-3.45E-25	castor_aralia	train
38	037.jpg	0.000703232351	1.75E-08	7.16E-12	2.44E-13	-2.35E-25	-1.21E-17	2.21E-25	castor_aralia	train

Hình 7: Mô tả file hu_moments.csv.

6. Bước tiếp theo là nhận diện loại lá cây thứ i trong tập test, với số thứ của em là 1, loại lá cây cần phát hiện là loại lá có class ‘Nanmu’,

Ta chuẩn bị đầu vào với Tập X và Y, trong đó X chứa 7 đặc trưng hu’s moment từ hu_moment_1 đến hu_moment_7. Y chứa các đặc trưng của label ‘Nanmu’. Tiếp theo ta sẽ chia X thành 2 tập gồm tập train và tập test dựa trên cột **set**, Y cũng được triển khai tương tự.

Tiếp theo ta sẽ chuẩn hóa dữ liệu trên tập X_train (tính trung bình và độ lệch chuẩn) và sử dụng các thống kê đã được tính toán trên tập X_train đã được chuẩn hóa để chuẩn hóa các giá trị trong X_test.

Sử dụng model SVC với kernel=’rbf’ để train tập X_train đã được chuẩn hóa và label của Y_train.

Cuối cùng đưa ra dự đoán trên tập X_test.

Hình 8 mô phỏng phần coding của bước này.

```
import pandas as pd
from sklearn.svm import SVC
from sklearn.metrics import classification_report
from sklearn.preprocessing import StandardScaler

hu_moments_df = pd.read_csv('hu_moments.csv')
X = hu_moments_df[['hu_moment_1', 'hu_moment_2', 'hu_moment_3', 'hu_moment_4', 'hu_moment_5', 'hu_moment_6', 'hu_moment_7']]
Y = hu_moments_df['label'].apply(lambda x: 1 if x == "Nanmu" else 0)

X_train = X[hu_moments_df['set'] == 'train']
X_test = X[hu_moments_df['set'] == 'test']
Y_train = Y[hu_moments_df['set'] == 'train']
Y_test = Y[hu_moments_df['set'] == 'test']
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
svm_model = SVC(kernel='rbf', C=1, gamma='scale', probability=True)

svm_model.fit(X_train_scaled, Y_train)

Y_pred = svm_model.predict(X_test_scaled)
```

Hình 8: Coding train SVM.

7. Tính Precision và Recall.

Sử dụng thư viện `classification_report`, `confusion_matrix` để tính ma trận nhầm lẫn và các giá trị Precision và Recall.

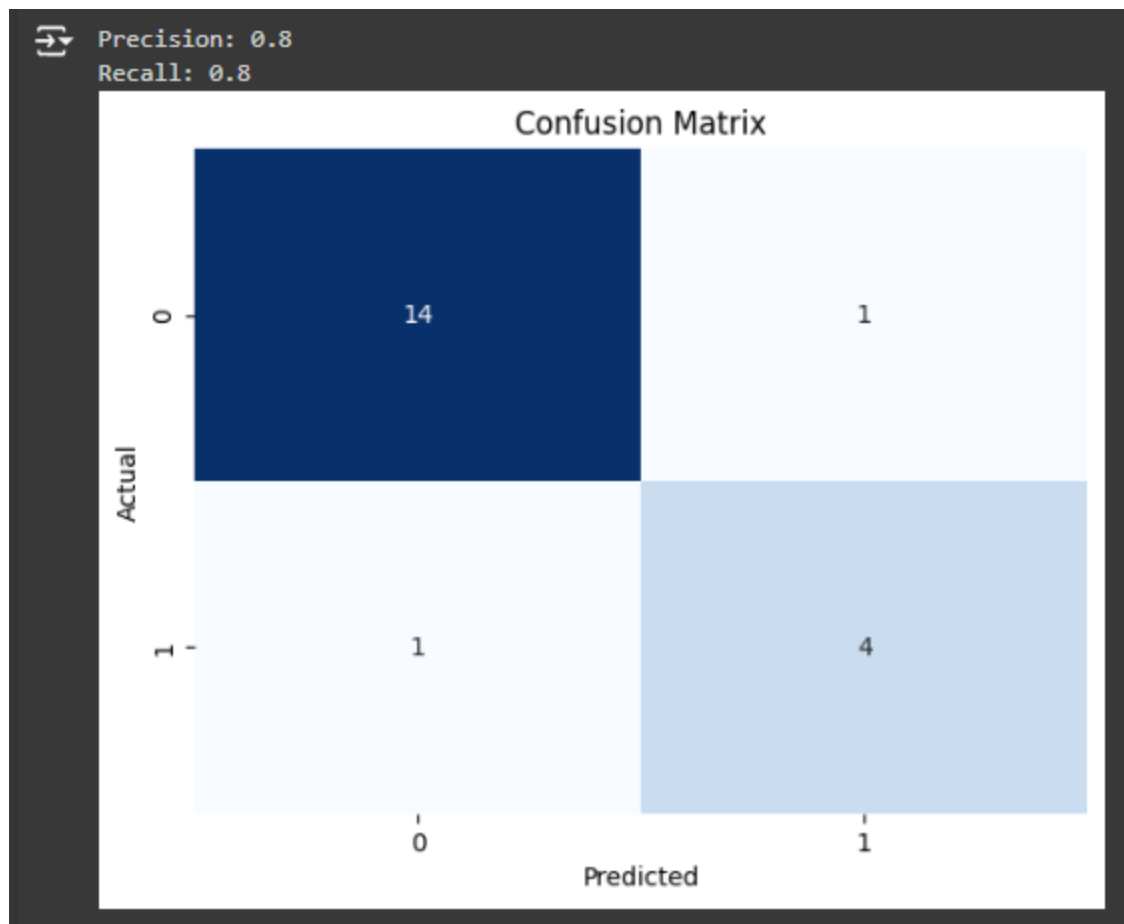
Hình 9 mô tả phần coding của bước này.

Hình 10 biểu diễn ma trận nhầm lẫn và giá trị Precision và Recall.

```
from sklearn.metrics import classification_report, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns

cm = confusion_matrix(Y_test, Y_pred)
sns.heatmap(cm, annot=True, cmap='Blues', fmt='g', cbar=False)
report=classification_report(Y_test, Y_pred, output_dict=True)
print(f"Precision: {report['1']['precision']}")
print(f"Recall: {report['1']['recall']}")
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```

Hình 9: Coding Vẽ ma trận nhầm lẫn và tính Precision và Recall.



Hình 10: Ma trận nhầm lẫn.

Đánh giá kết quả: Ta nhận thấy rằng tỉ lệ Precision và Recall đến đạt 0.8, một ngưỡng giá trị cao, điều này cho thấy mô hình có độ hiệu quả khá cao cho tập dữ liệu được sử dụng trong báo cáo này. Với Precision 0.8 cho thấy khả năng cao trong việc tránh dự đoán nhầm các mẫu không phải lớp 'Nanmu'. Với Recall 0.8, mô hình có thể phát hiện được 80% các mẫu là 'Nanmu' trong tổng số lớp lá này trong tập dữ liệu.

Kết luận: Mô hình cho ra kết quả phân loại và nhận diện tốt với lớp lá 'Nanmu' trong tập dữ liệu. Tuy vậy chưa đủ để đưa ra nhận định rằng mô hình này là mô hình tối ưu. Việc lựa chọn mô hình phụ thuộc nhiều và các tham số đầu vào cho SVC như kernel, C và gamma, nếu ta thay đổi các tham số này thì tỉ lệ dự đoán của mô hình cũng sẽ bị ảnh hưởng theo, dẫn đến tính không ổn định cho mô hình.

Hướng phát triển: Có 2 hướng phát triển cho bài toán này, hướng thứ nhất là ta có thể nghiên cứu và train mô hình trên nhiều tham số và kernel khác nhau từ đó đưa ra mô hình với các tham số tối ưu, hướng thứ 2 là cấu hình và tăng dữ liệu cho bộ dataset phù hợp với mô hình nhất có thể, ta có thể kết hợp cả 2 hướng trên nhằm nâng cao hơn nữa tính hiệu quả của mô hình.