TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH
KHOA ĐÀO TẠO CHẤT LƯỢNG CAO
NGÀNH CÔNG NGHỆ THÔNG TIN

BÁO CÁO ĐỒ ÁN 3

NHẬN DIỆN TRÁI CÂY

SVTH:

VÕ MINH HIẾU	**17110136**
LÊ MINH TIẾN	**17110236**

GVHD: NGUYỄN THIÊN BẢO

TP. Hồ Chí Minh, tháng 12 năm 2020

MỤC LỤC

LỜI MỞ ĐẦU

Nhóm xin chân thành cảm ơn sự hướng dẫn tận tình của thầy …, cả về chuyên môn lẫn tinh thần, đã giúp đỡ nhóm trong quá trình thực hiện project.

Với mỗi sinh viên nói chung, việc tích lũy kiến thức qua giáo trình, bài giảng trên lớp là rất quan trọng và cần thiết, tuy nhiên sinh viên lại chưa có nhiều cơ hội để áp dụng các kiến thức đó vào project. Cho nên, việc thực hiện Đồ án 3 chính là cơ hội cho nhóm được thử sức mình tạo ra những project thực tế, được hoạt động theo nhóm một cách chuyên nghiệp dưới sự hướng dẫn của các giảng viên có dày dặn kinh nghiệm trong lĩnh vực trí tuệ nhân tạo, cụ thể là machine learning.

Vì kiến thức và kỹ thuật còn hạn hẹp, nên trong quá trình thực hiện có sảy ra sai sót, rất mong thầy góp ý để nhóm có thể hoàn thiện project tốt hơn.

Được sự hướng dẫn tận tình của thầy, nhóm xin chân thành cảm ơn!

Danh mục các hình

Danh mục các bảng

CHƯƠNG 1: TỔNG QUAN VỀ ĐỀ TÀI

1.1. Giới thiệu đề tài

Chủ đề của nhóm là xây dựng ứng dụng nhận diện ký tự văn bản với machine learning. Ngày nay, các ứng dụng thông minh xuất hiện rất nhiều, đó là sự bùng nổ của lĩnh vực trí tuệ nhân tạo, nhóm đã quyết định xây dựng một ứng dụng có khả năng nhập diện được các loại trái cây khác nhau, giúp người dùng có thể nhận biết được các loại quả khác nhau mà trước giờ mình chưa từng được biết

1.2.1. Lý do, mục đích chọn đề tài

Với mong muốn tạo một ứng dụng thực tiễn, nhóm đã quyết định chọn đề tài này làm tiền đề để phát triển sau này.

Đây là một chủ đề có thể xem là nền tảng trong việc ứng dụng các thuật toán machine learning để giải quyết các vấn đề trong thực tiễn

1.2.2. Mục tiêu

Mục tiêu nhóm đặt ra là nhóm phải biết được quy trình thực hiện một dự án machine learning, hiểu về cách hoạt động của thuật toán, biết được các kỹ thuật xử lý dữ liệu, lựa chọn model, đánh giá model và phân tích model.

Mục tiêu đề ra là model xây dựng được phải có accuracy score trên 0.85.

CHƯƠNG 2: NỘI DUNG

1. Giới thiệu bộ dữ liệu

- Đặc tả dữ liệu

Nhóm chọn bộ dataset Fruits 360 dataset lấy từ Kaggle. Bộ dataset bao gồm hình ảnh với định dạng jpg của các trái cây và rau củ với 131 loại trái cây và rau củ khác nhau tương đương với 131 classes:

Label	Tên	Label	Tên
0	AppleBraeburn	**66**	Mangostan
1	AppleCrimsonSnow	**67**	Maracuja
2	AppleGolden1	**68**	MelonPieldeSapo
3	AppleGolden2	**69**	Mulberry
4	AppleGolden3	**70**	Nectarine
5	AppleGrannySmith	**71**	NectarineFlat
6	ApplePinkLady	**72**	NutForest
7	AppleRed1	**73**	NutPecan
8	AppleRed2	**74**	OnionRed
9	AppleRed3	**75**	OnionRedPeeled
10	AppleRedDelicious	**76**	OnionWhite
11	AppleRedYellow1	**77**	Orange
12	AppleRedYellow2	**78**	Papaya
13	Apricot	**79**	PassionFruit
14	Avocado	**80**	Peach
15	Avocadoripe	**81**	Peach2
16	Banana	**82**	PeachFlat
17	BananaLadyFinger	**83**	Pear
18	BananaRed	**84**	Pear2
19	Beetroot	**85**	PearAbate

20	Blueberry	**86**	PearForelle
21	Cactusfruit	**87**	PearKaiser
22	Cantaloupe1	**88**	PearMonster
23	Cantaloupe2	**89**	PearRed
24	Carambula	**90**	PearStone
25	Cauliflower	**91**	PearWilliams
26	Cherry1	**92**	Pepino
27	Cherry2	**93**	PepperGreen
28	CherryRainier	**94**	PepperOrange
29	CherryWaxBlack	**95**	PepperRed
30	CherryWaxRed	**96**	PepperYellow
31	CherryWaxYellow	**97**	Physalis
32	Chestnut	**98**	PhysaliswithHusk
33	Clementine	**99**	Pineapple
34	Cocos	**100**	PineappleMini
35	Corn	**101**	PitahayaRed
36	CornHusk	**102**	Plum
37	CucumberRipe	**103**	Plum2
38	CucumberRipe2	**104**	Plum3
39	Dates	**105**	Pomegranate
40	Eggplant	**106**	PomeloSweetie
41	Fig	**107**	PotatoRed
42	GingerRoot	**108**	PotatoRedWashed
43	Granadilla	**109**	PotatoSweet
44	GrapeBlue	**110**	PotatoWhite
45	GrapePink	**111**	Quince
46	GrapeWhite	**112**	Rambutan

47	GrapeWhite2	**113**	Raspberry
48	GrapeWhite3	**114**	Redcurrant
49	GrapeWhite4	**115**	Salak
50	GrapefruitPink	**116**	Strawberry
51	GrapefruitWhite	**117**	StrawberryWedge
52	Guava	**118**	Tamarillo
53	Hazeflnut	**119**	Tangelo
54	Huckleberry	**120**	Tomato1
55	Kaki	**121**	Tomato2
56	Kiwi	**122**	Tomato3
57	Kohlrabi	**123**	Tomato4
58	Kumquats	**124**	TomatoCherryRed
59	Lemon	**125**	TomatoHeart
60	LemonMeyer	**126**	TomatoMaroon
61	Limes	**127**	TomatonotRipened
62	Lychee	**128**	TomatoYellow
63	Mandarine	**129**	Walnut
64	Mango	**130**	Watermelon
65	MangoRed		

Bảng 1. Vị trí và tên các class của dự án

Có tổng cộng 90 483 hình ảnh trong đó có:

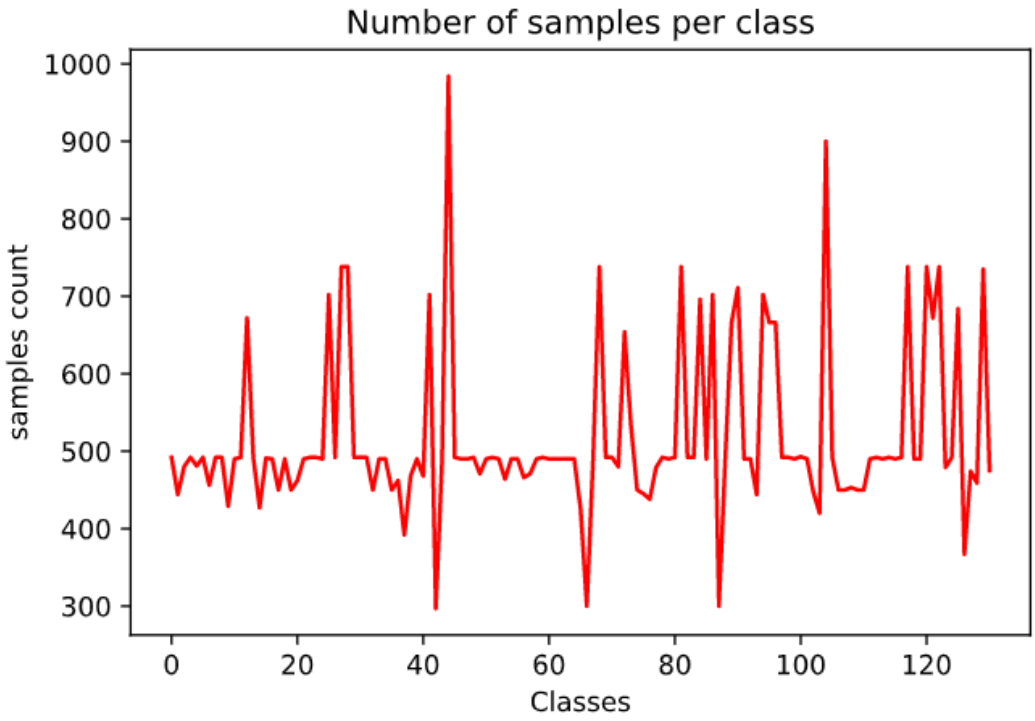
Training set: 67 692 hình ảnh (mỗi hình là một quả một rau củ)

Test set có: 22 688 hình ảnh (mỗi hình ảnh là một quả hoặc một rau củ)

Multi-fruits set: 103 hình ảnh (mỗi hình ảnh có nhiều trái cây or rau củ).

Mỗi hình ảnh có kích thước: 100x100 pixels. [1]

- Biểu đồ phân bố dữ liệu



Hình 1. Số lượng sample trong mỗi class

Mục tiêu

Mục tiêu nhóm đặt ra là nhóm phải biết được quy trình thực hiện một dự án machine learning, hiểu về cách hoạt động của thuật toán, biết được các kỹ thuật xử lý dữ liệu, lựa chọn model, đánh giá model và phân tích model.

Mục tiêu đề ra là model xây dựng được phải có accuracy score trên 0.85.

2. Thư viện sử dụng

2.1. Thư viện Scikit-learn

Scikit-learn (Sklearn) là thư viện mạnh mẽ nhất dành cho các thuật toán học máy được viết trên ngôn ngữ Python. Thư viện cung cấp một tập các công cụ xử lý các bài toán machine learning và statistical modeling gồm: classification, regression, clustering, và dimensionality reduction. [2]

Project lần này của nhóm thuộc về classification.

2.2. Thư viện OpenVC

Thư viện OpenCV thường được biết là một thư viện dùng để nhận diện tốt nhất hiện nay. Nhưng trong project môn học, ta chỉ áp dụng thư viện OpenCV để xử lý ảnh đầu vào như đọc ảnh màu, resize ảnh, convert màu ảnh từ RGB sang Gray.

2.3. Thư viện numpy

Đây là thư viện giúp xử lý tính toán trên mảng có kích thước lớn và nhiều chiều, được sử dụng xuyên suốt project.

2.4. Thư viện matplot

Matplotlib là một thư viện sử dụng để vẽ các đồ thị trong Python, chính vì vậy nó là thư viện cực phổ biến của Python. Nhóm sử dụng matplot để vẽ lên các đồ thị và confusion matrix.

2.5. Thư viện joblib

Đây là thư viện giúp nhóm lưu các model tìm được và các thông số tính toán.

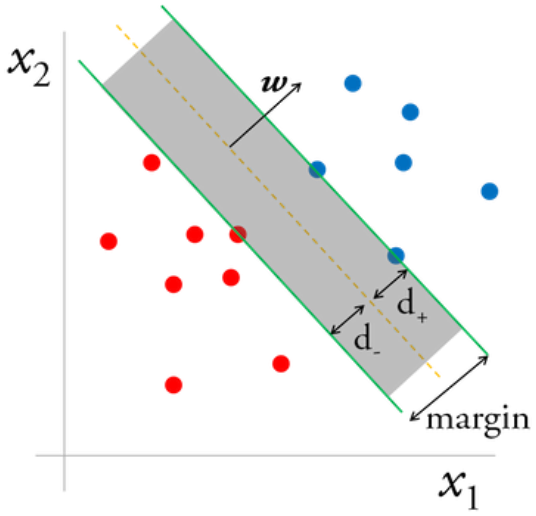
2. Thuật toán Support Vector Machine (SVM)

SVM là một thuật toán thuộc nhóm Supervised Learning (Học có giám sát) dùng để phân chia dữ liệu (Classification) thành các nhóm riêng biệt.

SVM là một thuật toán rất mạnh, chạy rất nhanh, trước khi deep learning ra đời thì nó là một trong những thuật toán tốt nhất. Trong một số trường hợp SVM nó còn tốt hơn deep learning vì deep learnining sinh ra model rất phức tạp và nó cần phải có một lượng lớn dữ liệu nếu không nó chạy không tốt.

2.1. Margin

Xét trong không gian hai chiều, Margin là khoảng cách giữa siêu phẳng đến 2 điểm dữ liệu gần nhất tương ứng với 2 phân lớp.



Hình 2. Mô hình minh hoạ SVM

SVM cố gắng tối ưu thuật toán bằng các tìm cách maximize giá trị margin này, từ đó tìm ra siêu phẳng đẹp nhất để phân 2 lớp dữ liệu.

2.2. Support vectors

Bài toán của chúng ta trở thành tìm ra 2 đường biên của 2 lớp dữ liệu (ở hình bên trên là 2 đường xanh lá cây) sao cho khoảng cách giữa 2 đường này là lớn nhất.

Đường biên của lớp xanh sẽ đi qua một (hoặc một vài) điểm xanh.

Đường biên của lớp đỏ sẽ đi qua một (hoặc một vài) điểm đỏ.

Các điểm xanh, đỏ nằm trên 2 đường biên được gọi là các support vector, vì chúng có nhiệm vụ support để tìm ra siêu phẳng.

Đó cũng là lý do của tên gọi thuật toán Support Vector Machine.

2.3. Soft Margin

Margin được chia thành 2 loại đó là Soft margin và Hard margin.

Để tránh overfitting, nhiều khi để muốn có margin cao, ta chấp nhận việc một vài data có thể không được chia chính xác (ví dụ như 1 bóng xanh bị lọt sang vùng của bóng đỏ). Data này được gọi là nhiễu.

⇨ Margin trong trường hợp này gọi là Soft Margin.

Hard Margin ám chỉ việc tìm được Margin mà không nhiễu (tất cả các data đều thoả mãn sự phân chia). Với các bài toán thực tế, việc tìm được Hard Margin nhiều khi là bất khả thi, vì thế việc chấp nhận sai lệch ở một mức độ chấp nhận được là vô cùng cần thiết.

Trong cài đặt SVM, người ta giới thiệu tham số CC với quy ước:

- C là một số dương vô cùng: đồng nghĩa với việc không cho phép sai lệch, đồng nghĩa với Hard Margin.
- C lớn: cho phép sai lệch nhỏ, thu được margin nhỏ.
- C nhỏ: cho phép sai lệch lớn, thu được margin lớn.

⇨ Tuỳ bài toán cụ thể mà ta cần điều chỉnh tham số C này để thu được kết quả tốt nhất.

3. Xử lý dữ liệu

3.1. Import dữ liệu và tiền xử lý dữ liệu

Bộ dữ liệu có 131 classes, mỗi bức ảnh có kích thước 100x100 pixels. Vì thế nên ta resize thành ảnh có kích thước 45x45. Một phần để giảm nhỏ lại số lượng feature nhằm tránh việc train model nhanh hơn.

Ta sử dụng OpenCV để đọc hình ảnh màu.

```
fruit_images = []
labels = []
for fruit_dir_path in glob.glob(r"D:\TaiLieuHocTap\Fruit360\fruits-360\Training\*"):
    fruit_label = fruit_dir_path.split("/")[-1]
    for image_path in glob.glob(os.path.join(fruit_dir_path, "*.jpg")):
        image = cv2.imread(image_path, cv2.IMREAD_COLOR)
```

Hình 3. Code sử dụng OpenCV để đọc ảnh màu

Sau đó ta resize ảnh có kích thước 100x100 thành 45x45 để giảm số lượng feature. Tiếp đó chuyển hình ảnh RGB thành bảng màu GRAY.

```
image = cv2.resize(image, (45, 45))
image = cv2.cvtColor(image, cv2.COLOR_RGB2GRAY)
```

Hình 4. Code chuyển màu ảnh red, green, blue sang gray

Bộ dữ liệu đầu ra của mỗi ảnh là mảng dữ liệu 2 chiều vector hoá 45x45 với mỗi phần tử có độ dài từ 0-255. Label được đánh số từ 0 đến 130. [4]

Đây là bộ dữ liệu của ảnh thứ 0 của AppleBraeburn (label 0) sau khi được chuyển đổi.

```
array([[254, 254, 254, ..., 254, 255, 255],
       [254, 254, 253, ..., 255, 255, 255],
       [255, 254, 254, ..., 255, 255, 255],
       ...,
       [254, 255, 255, ..., 255, 255, 255],
       [254, 255, 255, ..., 255, 255, 255],
       [255, 255, 255, ..., 255, 255, 255]],
```

Hình 5. Giá trị của các vector của sample sau khi chuyển

3.2. Reshape dữ liệu

Sau đó chúng ta reshape mảng hai chiều đó thành mảng 1 chiều với 2075 phần tử.

```
if first_run:
    X_train = X_train.reshape(67692, 2025)
    X_test = X_test.reshape(22688, 2025)
```

Hình 6. Reshape giá trị

3.3. Feature scaling

Ta scaling dữ liệu từ mảng 1 chiều với 2075 phần tử, mỗi phần tử có độ dài từ 0-255 sang mảng 1 chiều với 2075 phần tử, mỗi phần tử có độ dài từ 0-1.

```
from sklearn.preprocessing import StandardScaler
if 0:
    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train.astype(np.float64))
    joblib.dump(X_train_scaled, 'tam/X_train_scaled')

    X_test_scaled = scaler.fit_transform(X_test.astype(np.float64))
    joblib.dump(X_test_scaled, 'tam/X_test_scaled')
else:
    X_train_scaled = joblib.load('tam/X_train_scaled')
    X_test_scaled = joblib.load('tam/X_test_scaled')
print('done')
```

Hình 7. Scale các phần tử trong tập train

4. Tạo model bằng SVM

4.1. Training data

```
from sklearn import svm

svm_clf = svm.SVC(decision_function_shape='ovo')
if 0:
    svm_clf = svm_clf.fit(X_train_scaled, y_train)
    joblib.dump(svm_clf, 'tam/svm_clf')
else:
    svm_clf = joblib.load('tam/svm_clf')
```

Hình 8. Training data

Các parameters:

- decision_function_shape: Mặc định SVC sẽ train với option là “ova” tức là one vs all, nhưng nhóm chỉ định option “ovo” để cải thiện tốc độ training. Số lượng class là 131, nên tổng số binary class phải dùng là: n * (n – 1)/2 = 8515 class. Tuy số lượng class rất nhiều, nhưng đối với one vs all, ta cần phải nạp toàn bộ training data vào Ram mới thực hiện training, còn đối với one vs one, ta chỉ cần nập từng class vào, và do tính độc lập (one vs one) nên có thể thực hiện tính toán song song hiệu quả hơn.

4.2. Đánh giá model

Các điểm đánh giá:

- Accuracy: đây là điểm số hay được sử dụng nhất, nó được tính bằng tỉ lệ giữa số lượng sample dự đoán đúng với tổng số lượng dự đoán trong bộ dữ liệu dùng để kiểm thử, có giá trị trong khoảng từ [0, 1], càng cao càng tốt.

- Precision: được xem như độ chính xác, có giá trị trong khoảng từ [0, 1], càng cao càng tốt, được tính bằng công thức:

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

Các ký hiệu được mô tả như sau:

	Dự đoán là Positive	**Dự đoán là Negative**
Actual: Positive	True Positive (TP)	False Newgative (FN)
Actual: Negative	False Positive (FP)	True Negative (TN)

- Recall: được xem như độ bao phủ, có giá trị trong khoảng từ [0, 1], càng cao càng tốt, được tính bằng công thức:

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

4.3. Accuracy score với cross validation

```
from sklearn.model_selection import cross_val_predict
from sklearn.model_selection import cross_val_score

if 0:
    svm_acc = cross_val_score(svm_clf, X_train_scaled, y_train, cv=5, n_jobs=-1, scoring="accuracy")
    joblib.dump(svm_acc, 'tam/svm_acc')
else:
    svm_acc = joblib.load('tam/svm_acc')

print('done')
print('Cross validate score:')
print(svm_acc)
```

```
done
Cross validate score:
[0.81505281 0.90819115 0.7834983  0.69197814 0.72957601]
```

Hình 9. Cross validation scrore

Cross validation score với 5 folds cho ra kết quả chưa được tốt cho lắm, 2 folds đầu cho ra tương đối, nhưng 3 folds còn lại cho ra kết quả khá kém.

4.4. Accuracy score trên tập test set

```
if 1:
    svm_test_acc = svm_clf.score(X_test_scaled, y_test)
    joblib.dump(svm_test_acc, 'tam/svm_test_acc')
else:
    svm_test_acc = joblib.load('tam/svm_test_acc')
print('done')
print('test accuracy score:')
print(svm_test_acc)
```

```
done
test accuracy score:
0.8672866713681241
```

Hình 10. Accuracy score

Ta thấy accuracy score trên tập test set cũng khá cao, đã đạt yêu cầu, nhưng vẫn có thể được cải thiện.

4.5. Precision và recall và f1 scores

	precision	recall	f1-score	support
0	0.87	0.73	0.79	164
1	0.80	0.86	0.83	148
2	1.00	0.89	0.94	160
3	0.92	1.00	0.96	164
4	0.72	1.00	0.83	161
5	0.89	0.85	0.87	164
6	0.80	0.72	0.76	152
7	0.97	0.68	0.80	164
8	0.78	0.78	0.78	164
9	0.99	0.94	0.96	144
10	0.96	0.99	0.98	166
11	0.88	0.92	0.90	164
12	0.87	0.95	0.91	219
13	0.78	0.76	0.77	164
14	0.94	0.82	0.87	143
15	0.93	0.98	0.96	166
16	0.66	0.69	0.67	166
17	0.44	0.82	0.57	152
18	0.60	0.55	0.58	166
19	0.78	0.43	0.55	150
20	1.00	0.78	0.88	154
21	0.68	0.83	0.75	166
22	0.99	1.00	1.00	164
120	1.00	1.00	1.00	246
121	0.84	1.00	0.91	225
122	0.95	0.83	0.89	246
123	0.99	1.00	0.99	160
124	0.98	1.00	0.99	164
125	1.00	0.84	0.91	228
126	0.95	1.00	0.97	127
127	0.95	0.95	0.95	158
128	0.99	1.00	0.99	153
129	0.75	1.00	0.86	249
130	0.77	0.71	0.74	157
accuracy			0.87	22688
macro avg	0.88	0.86	0.86	22688
weighted avg	0.88	0.87	0.86	22688

Hình 11. Report precision và recall

5. Tunning tham số C.

Nhóm thực hiện tunning tham số C sao cho model cho ra kết quả tốt nhất. Các parameter candidates là: C = {0.1, 1, 10, 17, 25, 37, 50, 75, 100}. Vì thời gian tunning quá lâu, nên nhóm chia từng part để tunning, mỗi lần chỉ tunning 2 giá trị.

- Lần 1:

```
parameter_candidates_part_1 = [
  {'C': [0.01, 0.1]},
]
if 0:
    tunning_01 = GridSearchCV(estimator=svm.SVC(decision_function_shape='ovo'), param_grid=parameter_candidates_part_1, n_jobs=-1
    tunning_01.fit(X_train_scaled, y_train)
    joblib.dump(tunning_01, 'tunning/clf_tunning_01')
else:
    tunning_01 = joblib.load('tunning/clf_tunning_01')

print('Best score:', tunning_01.best_score_)
print('Best C:',tunning_01.best_estimator_.C)
```

Hình 12. Tunning model

Kết quả cho ra:

```
tunning C part 1
C = {0.1, 1}
Best score: 0.6438091984252667
Best C: 0.1
```

Hình 13. Kết quả tunning lần 1

- Lần 2:

```
tunning C part 2
C = {1, 10}
Best score: 0.8005356913849695
Best C: 10
```

Hình 14. Kết quả tunning lần 2

Ta thấy với C = 10 thì Best score tăng lên rõ rệt, ta tiếp tục tăng để xem kết quả có còn được cải thiện hay không.

- Lần 3:

```
tunning C part 3
C = {50, 100}
Best score: 0.8007277292016471
Best C: 50
```

Hình 15. Kết quả tunning lần 3

Ta thấy kết quả không tăng lên nhiều, có thể ta đã tăng lên quá nhiều.

- Lần 4:

```
tunning C part 4
C = {25, 75}
Best score: 0.8007425024325527
Best C: 24
```

Hình 16. Kết quả tunning lần 4

- Lần 5:

```
tunning C part 5
C = {17, 37}
Best score: 0.8007720467120418
Best C: 17
```

Hình 17. Kết quả tunning lần 5

=> Với C bằng 17 thì model cho ra score tốt nhất, nên ta sẽ trainning với tham số C = 17 để có được model tốt hơn.

5.3.2. Tunning tham số Gama.

5.4. Trainning lại model với best parameters

```python
from sklearn import svm

svm_clf_best_parameter = svm.SVC(decision_function_shape='ovo', kernel='rbf', gamma='scale', C = 17)
if 0:
    svm_clf_best_parameter = svm_clf_best_parameter.fit(X_train_scaled, y_train)
    joblib.dump(svm_clf_best_parameter, 'tam/svm_clf_best_parameter')
else:
    svm_clf_best_parameter = joblib.load('tam/svm_clf_best_parameter')
print('done')
```

Hình 18. Training model với best parameter

Sau khi tìm được tham số tốt nhất cho model thì nhóm tiến hành trainning lại với 100% dữ liệu training, hi vọng rằng nó sẽ cho ra kết quả tốt.

R2Score và Root Mean Square Error

```
Calculate on Test set
R2 score (on testing data, best=1): 0.8945257404795487
Root Mean Square Error:  16.1
==========================================
```

Hình 19. R2 score và RMSE

Ta thấy kết quả cho ra tốt hơn khá nhiều so với lúc đầu, R2 score tăng lên tới 0.89 so với ban đầu là 0.86.

Cross validation accuracy

```
from sklearn.model_selection import cross_val_predict
from sklearn.model_selection import cross_val_score

if 0:
    svm_acc_val_best_parameter = cross_val_score(svm_clf_best_parameter, X_train_scaled, y_train, cv=5, n_jobs=-1, scoring="ac
    joblib.dump(svm_acc_val_best_parameter, 'tam/svm_acc_val_best_parameter')
else:
    svm_acc_val_best_parameter = joblib.load('tam/svm_acc_val_best_parameter')

print('done')
print('Cross validate score with best parameter:')
print(svm_acc_val_best_parameter)
```

Hình 20. Cross validation code

- Kết quả thu được:

```
Cross validate score with best parameter:
[0.82325135 0.91690671 0.80174324 0.71206973 0.7498892 ]
```

Hình 21. Kết quả cross validation

Accuracy trên tập testing data

```
if 0:
    svm_test_acc_best_parameter = svm_clf_best_parameter.score(X_test_scaled, y_test)
    joblib.dump(svm_test_acc_best_parameter, 'tam/svm_test_acc_best_parameter')
else:
    svm_test_acc_best_parameter = joblib.load('tam/svm_test_acc_best_parameter')
print('done')
print('test accuracy score with best parameter:')
print(svm_test_acc_best_parameter)
```

Hình 22. Accuracy trên tập data

- Kết quả thu được:

```
test accuracy score with best parameter:
0.8945257404795487
```

Hình 23. Kết quả accuracy trên tập data

Precison, recall và f1 score

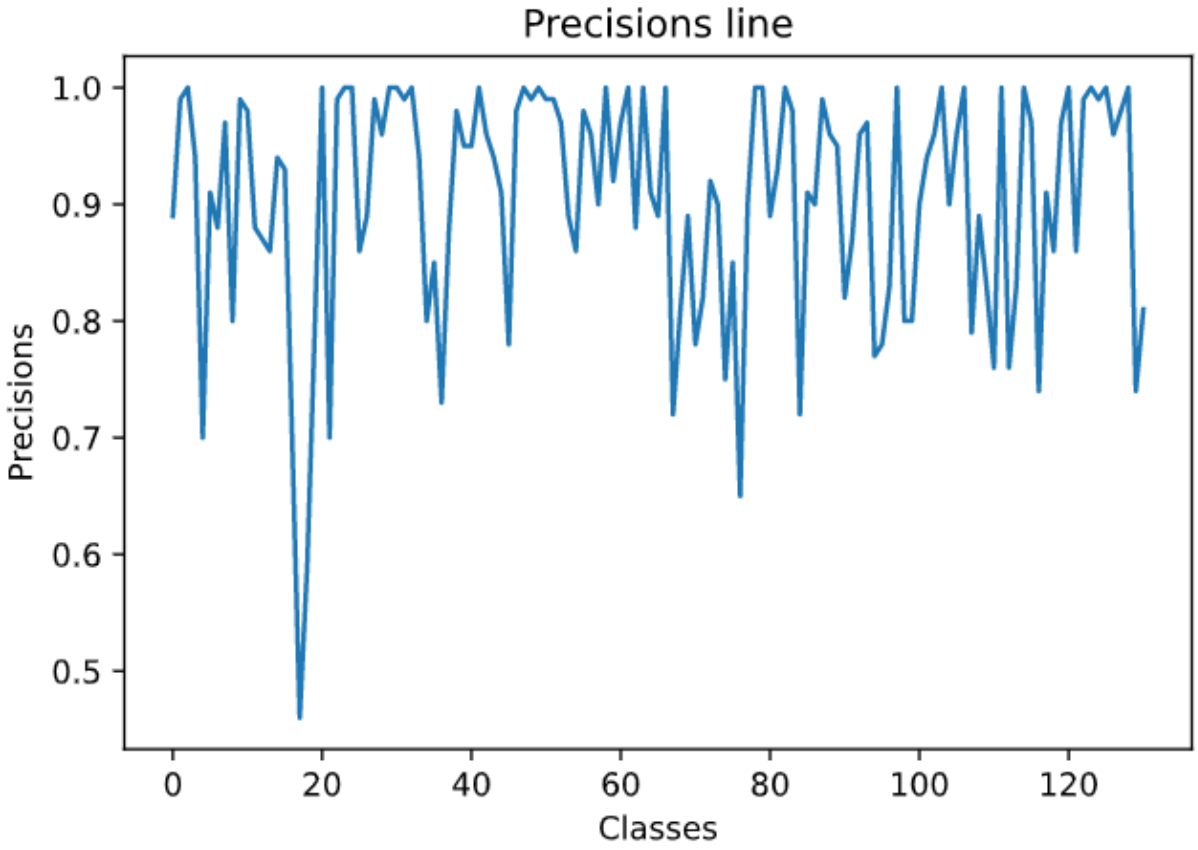
	precision	recall	f1-score	support
0	0.89	0.76	0.82	164
1	0.99	0.90	0.94	148
2	1.00	0.88	0.94	160
3	0.94	1.00	0.97	164
4	0.70	1.00	0.82	161
5	0.91	0.85	0.88	164
6	0.88	0.82	0.85	152
7	0.97	0.70	0.81	164
8	0.80	0.80	0.80	164
9	0.99	0.99	0.99	144
10	0.98	0.99	0.99	166
11	0.88	0.92	0.90	164
12	0.87	0.95	0.91	219
13	0.86	0.77	0.81	164
14	0.94	0.81	0.87	143
15	0.93	0.98	0.96	166
16	0.69	0.71	0.70	166
17	0.46	0.85	0.59	152
18	0.59	0.55	0.57	166
19	0.79	0.50	0.61	150
20	1.00	0.89	0.94	154
21	0.70	0.83	0.76	166
22	0.99	1.00	1.00	164

120	1.00	1.00	1.00	246
121	0.86	1.00	0.93	225
122	0.99	0.89	0.94	246
123	1.00	1.00	1.00	160
124	0.99	1.00	0.99	164
125	1.00	0.84	0.91	228
126	0.96	1.00	0.98	127
127	0.98	0.96	0.97	158
128	1.00	1.00	1.00	153
129	0.74	1.00	0.85	249
130	0.81	0.71	0.76	157
accuracy			0.89	22688
macro avg	0.90	0.89	0.89	22688
weighted avg	0.90	0.89	0.89	22688

Hình 24. Precision và recall report

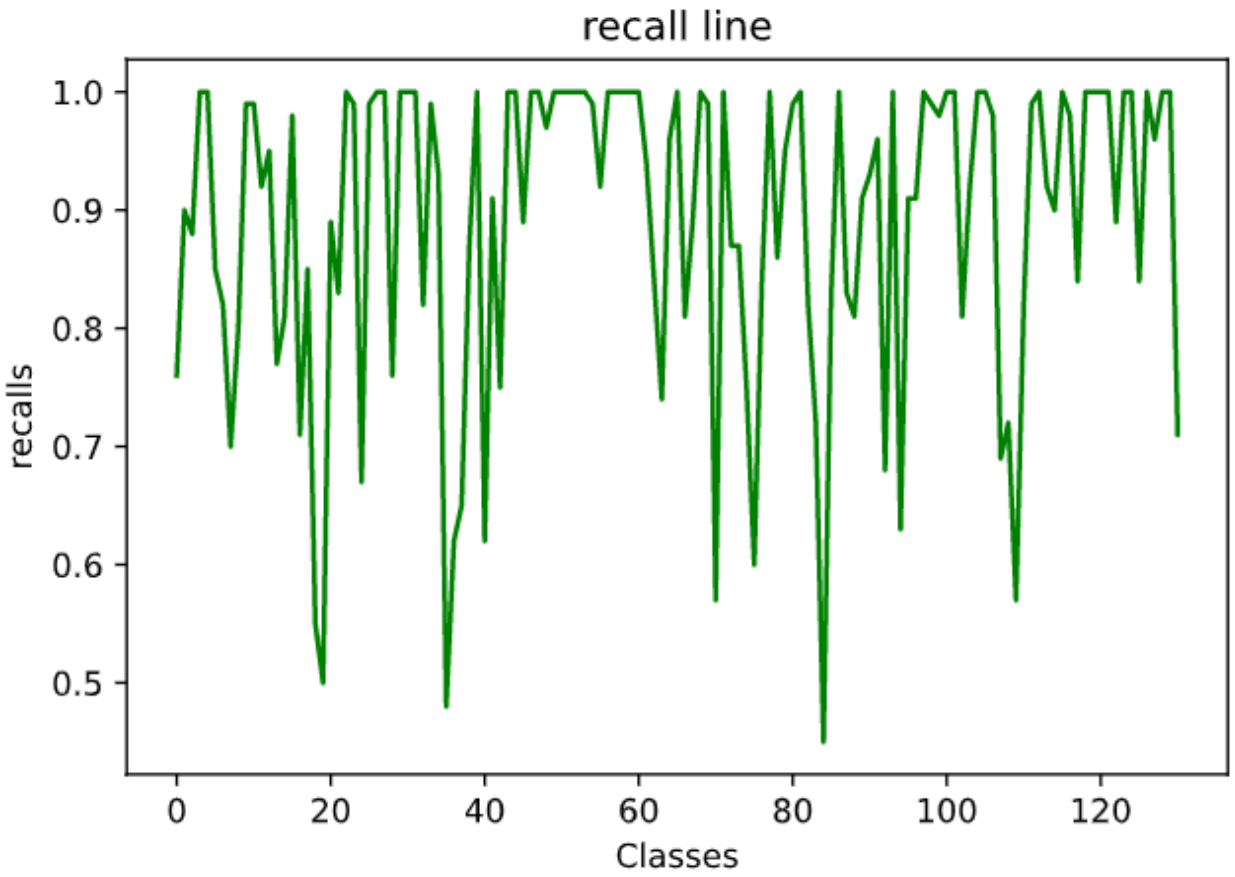
Biểu đồ thể hiện:

Precisons:



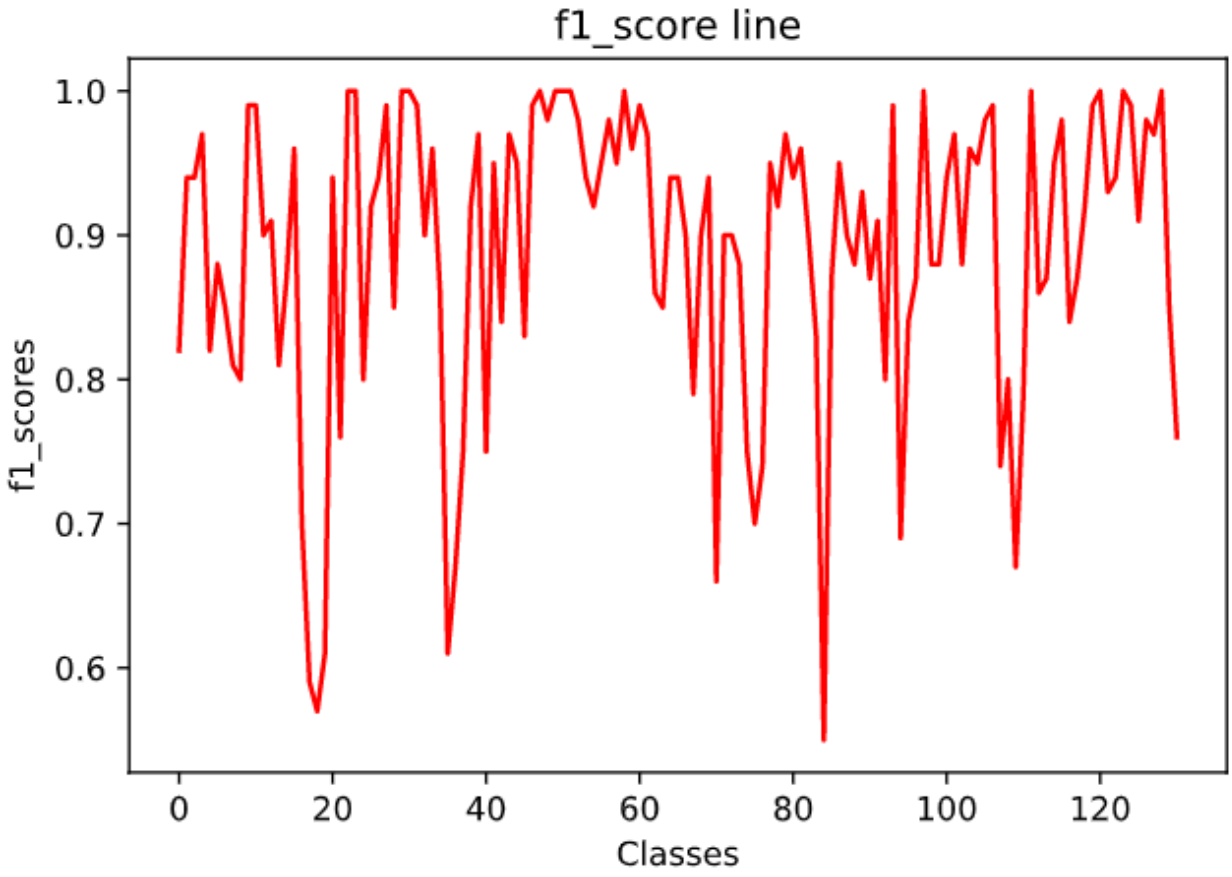
Hình 25. Sơ đồ presicion

Ta thấy
Recalls:



Hình 26. Sơ đồ recall

F1 scores:



Hình 27. Sơ đồ F1 score

Với F1 scores là điểm trung bình hài hòa giữa precision và recall, các class có điểm số ổn định hơn, một số class có điểm số rất thấp như 17, 18, 35, 36, 84 vẫn có điểm số thấp, ta sẽ phải phân tích để xem các sample của class này có vấn đề gì.

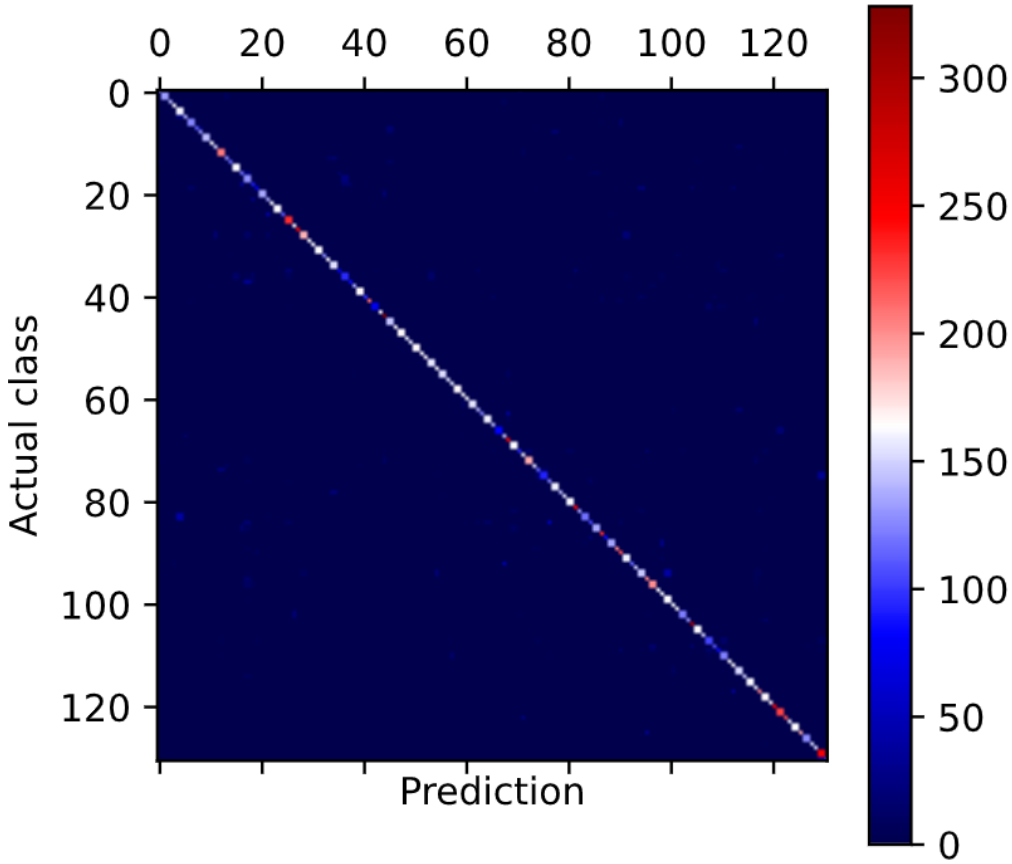
5.1. Phân tích và test

5.1.1. Phân tích bằng confusion matrix

Để phân tích lỗi, ta cần predict hết các samples trong tập test set và kiểm tra với label thật sự, sau đó lưu lại kết quả predict.

Công cụ để chúng ta có thể phân tích lỗi hiệu quả, chi tiết và được dùng phổ biến là Confusion matrix.

Confusion matrix được vẽ trên thể hiện mức độ chính xác với từng class, mức độ chính xác tăng tương tứng từ gang màu xanh lên màu đỏ. Trục X là dự đoán còn trục Y là class thật sự.

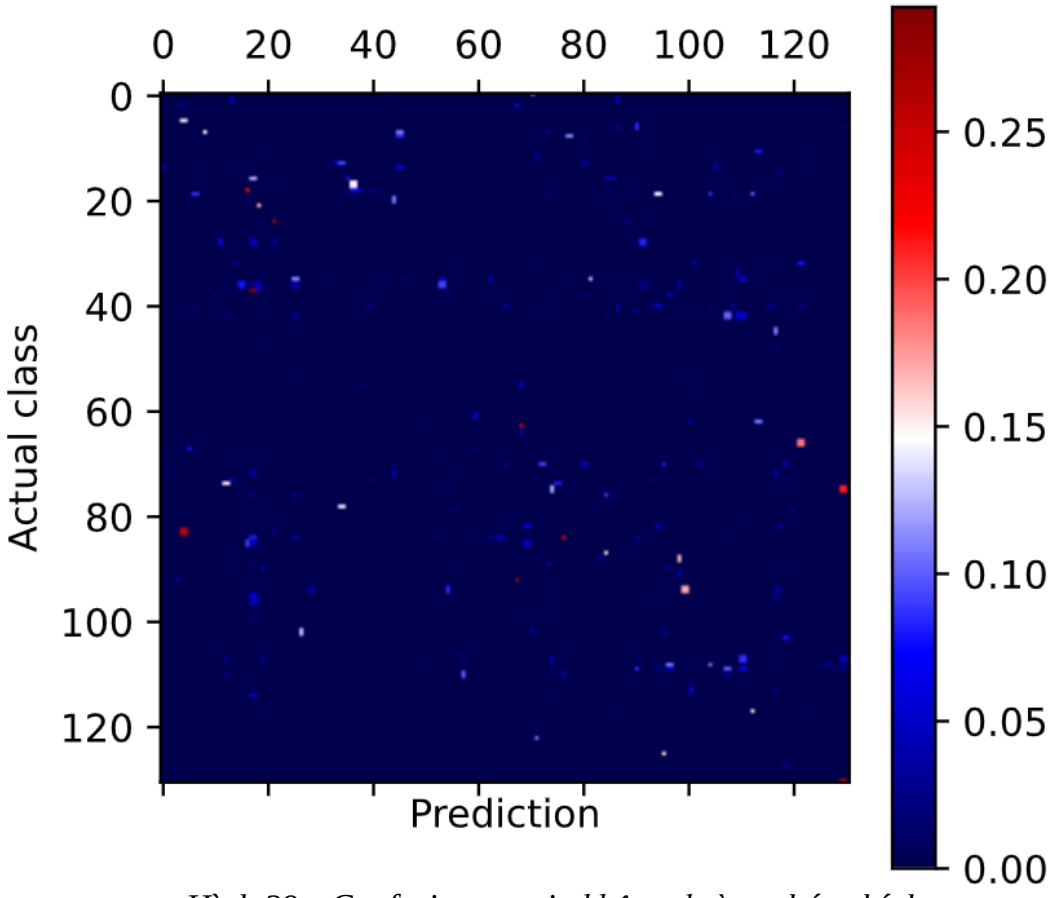


Hình 28. Confusion matrix có đường chéo chính

Ta thấy đường chéo chính được nổi bật lên hẳn, cho thấy model dự đoán khá tốt.

Để cho dễ quan sát và phân tích lỗi hơn thì ta sẽ đưa đuog chéo chính không còn nổi lên nữa, ta sẽ đưa nó về bằng 0. và các lỗi sẽ nổi lên.

Giờ ta sẽ tính theo tỉ lệ để so sánh sai khác giữa các class khác nhau, bỏ qua số lượng phần tử của nó, lấy tỉ lệ từ 0 đến 1.



Hình 29. Confusion matrix không đường chéo chính

Ta thấy, có class lỗi tối đa ở mức 0.3 trong tổng số 131 class, nghĩa là có một số class lỗi rất nhiều.

Một số điểm màu đỏ thể hiện tỉ lệ lỗi cao trên 0.25:

```
Các class có tỉ lệ lỗi trên 0.25:
[{0.2710843373493976, 24, 21} {0.2923076923076923, 17, 37}
 {0.25301204819277107, 68, 63} {0.27710843373493976, 67, 92}
 {0.2611464968152866, 129, 130}]
```

Hình 30. Các class có tỉ lệ lỗi trên 0.25

+ Class 24 bị nhầm sang class 21 có tỉ lệ lỗi lên đến 0.27.

+ Class 17 bị nhầm sang class 37 có tỉ lệ lỗi lên đến 0.29.

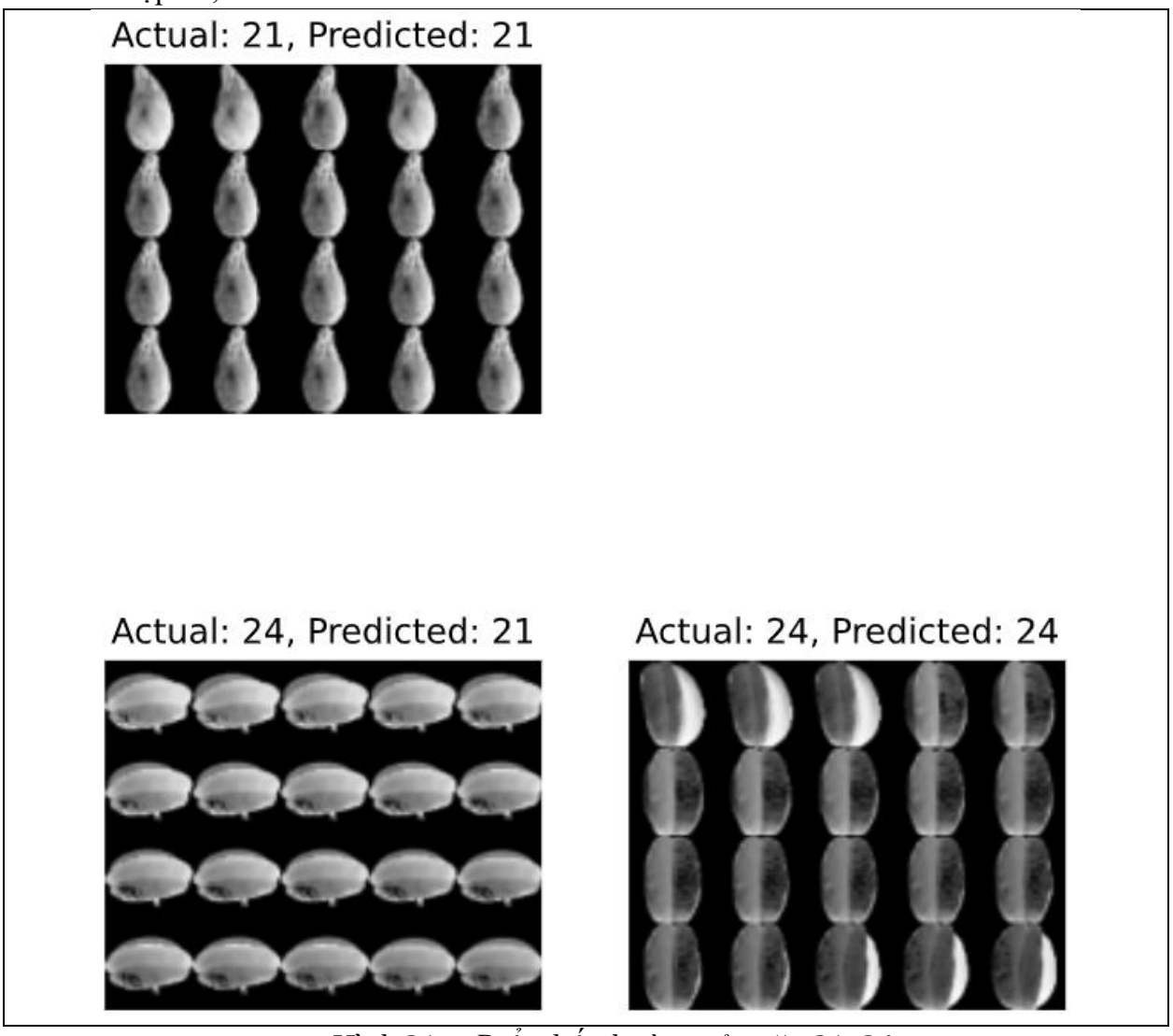
+ Class 68 bị nhầm sang class 63 có tỉ lệ lỗi lên đến 0.25.

+ Class 67 bị nhầm sang class 92 có tỉ lệ lỗi lên đến 0.27.

+ Class 129 bị nhầm sang class 130 có tỉ lệ lỗi lên đến 0.26.

Ta cần xem xét lại dữ liệu xem có điều gì bất thường:

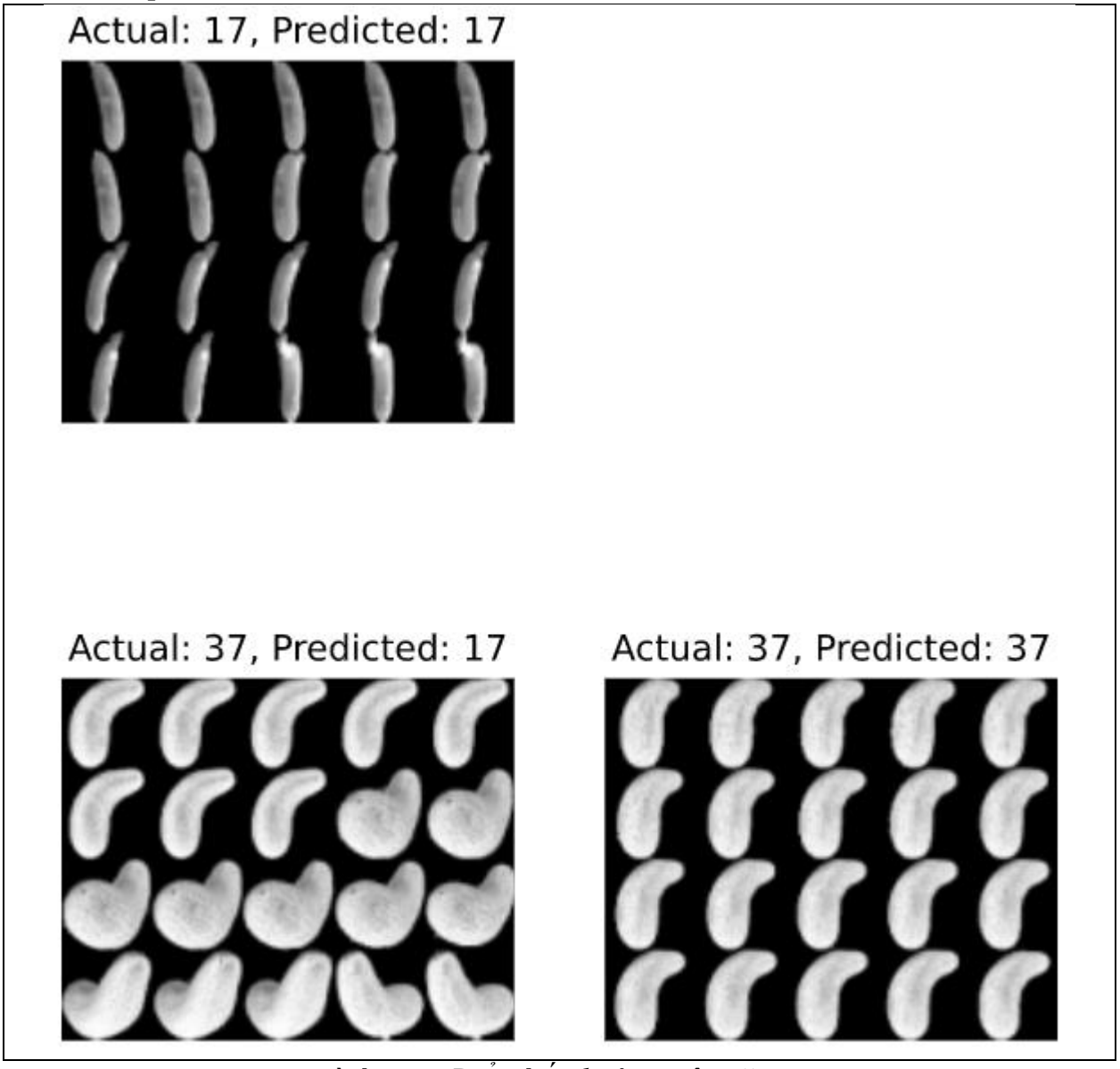
+ Cặp 24, 21:



Hình 31. Điểm bất thường của cặp 21, 24

Hầu hết các quả khế nằm ngang đều dự đoán nhầm sang quả sương rồng, trong khi không có bất kỳ sample nào dự đoán nhầm sang quả khế.

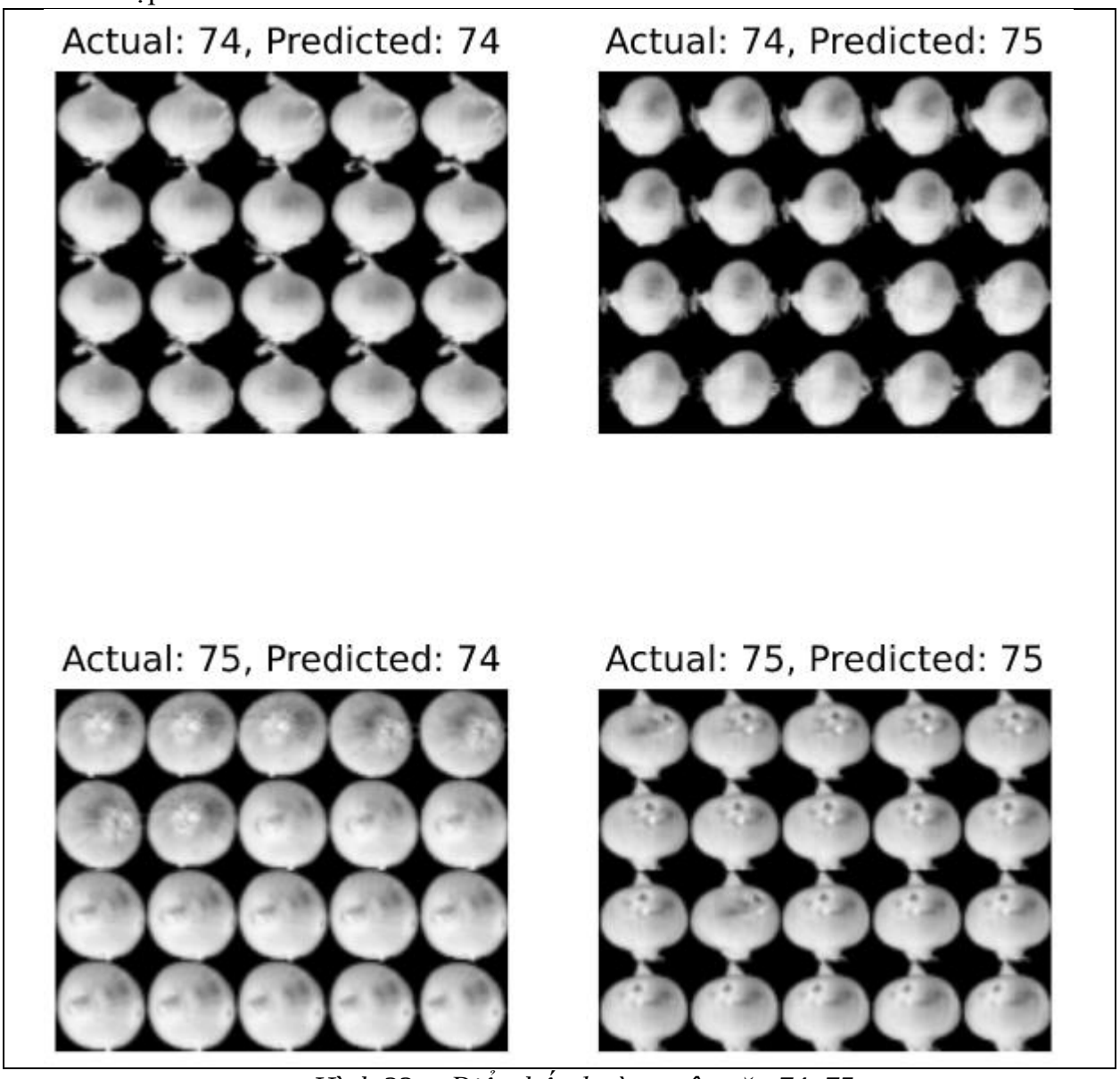
+ Cặp 17 và 37:



Hình 32. Điểm bất thường trên cặp 17, 37

Ta thấy CucumberRipe có hình dáng to mập bị dự đoán nhầm sang BananaLadyFinger khá nhiều. Không có trường hợp BananaLadyFinger bị nhầm sang CucumberRipe.

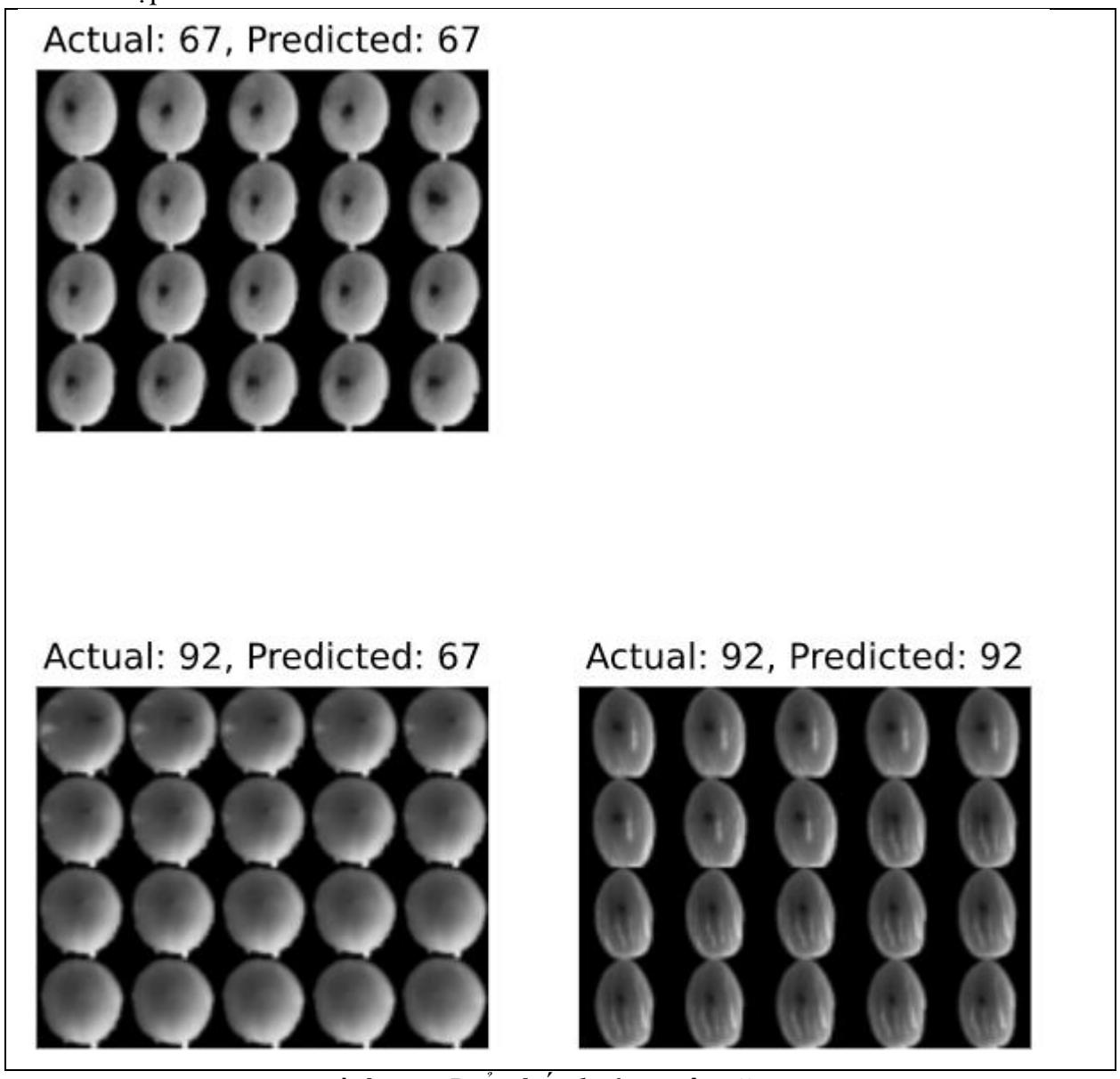
+ Cặp 74 và 75:



Hình 33. Điểm bất thường trên cặp 74, 75

Ta thấy OnionRed khi cho nằm ngang ra thì dự đoán bị sai sang OnionRedPeeled, còn OnionRedPeeled khi dựng đứng lên thì toàn bộ bị dự đoán nhầm sang OnionRed. Nhìn bằng mắt thường ta cũng khó phân biệt được 2 class này, điều này khiến model bị bối rối, khó phân biệt được 2 class.

+ Cặp 67 và 92:



Hình 34. Điểm bất thường trên cặp 67, 92

Ta thấy Pepino bị khi được chụp từ trên xuống thì nó bị nhầm sang Maracuja. Không có trường hợp dự đoán nhầm ngược lại.

CHƯƠNG 3: KẾT LUẬN

Với mục tiêu đề ra từ đầu là xây dựng một ứng dụng dự đoán ký tự sử dụng machine learning, nhóm đã hoàn thành được khoảng 90%. Accuracy của model tìm đạt trên 0.89, đạt được mục tiêu đề ra ban đầu là 0.85.

Sau khi hoàn thành project, nhóm thấy mình đã nắm vững kiến thức về machine learning nhiều hơn, biết cách xử lý dữ liệu sao cho hợp lý, có khả năng đánh giá model qua các điểm đánh giá, biết phân tích lỗi và tối ưu model để hiệu suất của model tăng lên.

TÀI LIỆU THAM KHẢO

[1] Kaggle, Fruit 360 [online] Available at: https://www.kaggle.com/moltean/fruits. [Accessed 09/01/2021]

[2] Tác giả hungdhv97@gmail.com, Thư Viện Scikit-learn Trong Python Là Gì? [online] Available at: https://codelearn.io/sharing/scikit-learn-trong-python-la-gi#:~:text=Scikit%2Dlearn%20(Sklearn)%20l%C3%A0,%2C%20clustering%20%2C%20v%C3%A0%20dimensionality%20reduction%20. [Accessed 09/01/2021]