



Phân loại ảnh (Image Classification)



1. Giới thiệu bài toán

- Input:

- Tập dữ liệu: $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$
 - $x_i \in \mathcal{F} = \mathbb{R}^d$: mỗi ảnh được mô tả trong không gian d chiều
 - $y_i \in \mathcal{L} = \{l_1, l_2, \dots, l_M\}$, là nhãn của các lớp/class

- Ảnh mới $x \in \mathcal{F}$

- Output:

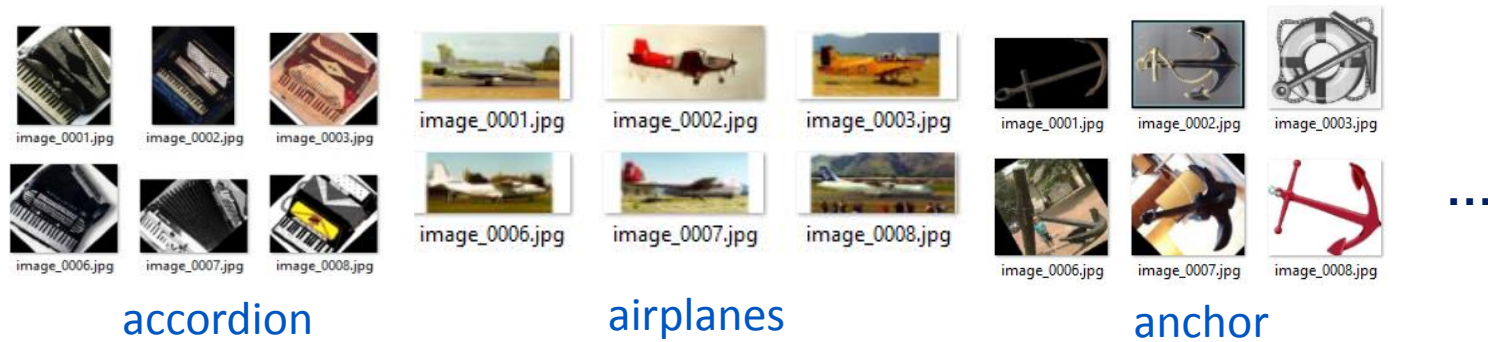
- Ảnh x được gán nhãn $y \in \mathcal{L}$



1. Giới thiệu bài toán



Tập các lớp được cho trước.
(ImageNet: 1K lớp)



Input



Mô hình phân loại



'airplanes'

Output

→ Xác định ảnh x thuộc vào một trong C lớp cho trước.



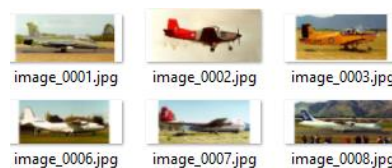
1. Giới thiệu bài toán



Tập các lớp được cho trước.



accordion



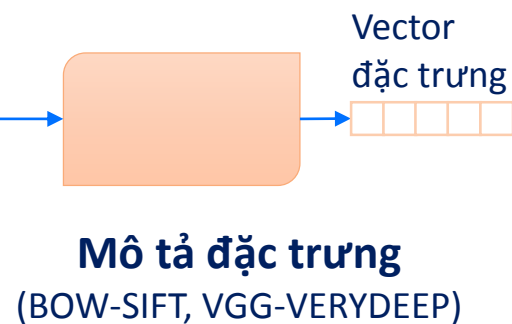
airplanes



anchor

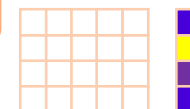


Input



Mô hình phân loại

Mô tả đặc trưng



'airplanes'

Output

1. Giới thiệu bài toán

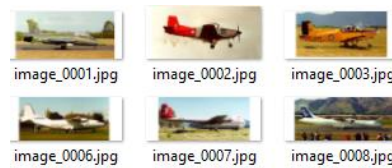


- Hai giai đoạn chính

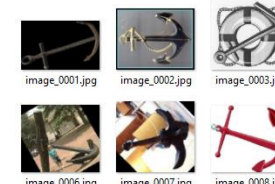
Tập các lớp được cho trước.



accordion



airplanes



anchor

Giai đoạn huấn luyện
(Offline)

Xây dựng mô hình phân loại



Input

Mô hình phân loại

Output

'airplanes'

Giai đoạn phân lớp
(Online)

2. Tầm quan trọng của bài toán



- Có nhiều ứng dụng trong thực tiễn:

- Gán nhãn cho ảnh-video

- Diễn đạt nội dung ảnh

- Tổ chức quản lý và phân loại tự động ảnh mới

- Tìm kiếm ảnh-video,...



sacre coeur;
Paris;
location vacances



"a large airplane sitting on top
of an airport runway"

- Có thể mở rộng cho các loại dữ liệu khác: video, văn bản,...



3. Độ đo đánh giá kết quả

- Giả sử tập dữ liệu test gồm có:
 - $X = \{x_1, \dots, x_T\}$ gồm T ảnh cần phân lớp
 - Tập $G_{T \times 1} \in \mathcal{L}$ giá trị nhãn **đã biết** của T ảnh.
- Kết quả phân lớp T ảnh ta được:
 - Tập $R_{T \times 1} \in \mathcal{L}$ giá trị nhãn **dự đoán** của T ảnh.



3. Độ đo đánh giá kết quả

3.1 Độ chính xác phân lớp (accuracy):

- Để trả lời cho câu hỏi “What proportion of photos — both Positive and Negative — were correctly classified?”
- Độ chính xác phân lớp được tính:

$$\text{Accuracy} = \frac{\text{number of correct predictions}}{\text{Total number of predictions}}$$



3. Độ đo đánh giá kết quả

3.1 Độ chính xác phân lớp (accuracy):

Độ chính xác phân lớp được tính:

$$Accuracy = \frac{\sum_{i=1}^T f(G_{i,1}, R_{i,1})}{T}$$

Trong đó: f là một hàm indicator

$$f(a, b) = \begin{cases} 1, & \text{nếu } a = b \\ 0, & \text{nếu } a \neq b \end{cases}$$



3. Độ đo đánh giá kết quả

3.1 Độ chính xác phân lớp (accuracy):

Trong trường hợp **phân lớp nhị phân**, Acc có thể tính theo kết quả dự đoán các mẫu dương và các mẫu âm:

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

Trong đó:

- TP = True Positives,
- TN = True Negatives,
- FP = False Positives, and
- FN = False Negatives.

		Predicted class	
		P	N
Actual Class	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)



3. Độ đo đánh giá kết quả

3.1 Độ chính xác phân lớp (accuracy):

Ví dụ:

		Predicted class		
		Positive	Negative	
Actual class	n=165			
	Positive	TP=100	FN=5	105
	Negative	FP=10	TN=50	60
		110	55	

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

$$Accuracy = \frac{100 + 50}{100 + 50 + 10 + 5} = 0.9091 = 90.91\%$$



3. Độ đo đánh giá kết quả

3.2 Precision và Recall:

Precision

- Để trả lời cho câu hỏi “What proportion of positive identifications was actually correct? “

- Công thức:

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

The precision is also called the positive predictive value (PPV)

		Predicted class	
		P	N
Actual Class	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)



3. Độ đo đánh giá kết quả

3.2 Precision và Recall:

Recall

- Để trả lời cho câu hỏi “What proportion of actual positives was identified correctly? “
- Công thức:

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

		Predicted class	
		<i>P</i>	<i>N</i>
Actual Class	<i>P</i>	True Positives (TP)	False Negatives (FN)
	<i>N</i>	False Positives (FP)	True Negatives (TN)



3. Độ đo đánh giá kết quả

3.2 Precision và Recall:

Specificity:

Specificity determines the proportion of actual negatives that are correctly identified.

$$Specificity = \frac{TN}{TN + FP}$$



3. Độ đo đánh giá kết quả

3.2 Precision và Recall:

Ví dụ:

		Predicted class		
		n=165		
Actual class		Positive	Negative	
	Positive	TP=100	FN=5	105
	Negative	FP=10	TN=50	60
		110	55	

$$Precision = \frac{TP}{TP+FP} = \frac{100}{100+10} = 0.9091 = 90.91\%$$

$$Recall = \frac{TP}{TP + FN} = \frac{100}{100 + 5} = 0.9524 = 95.24\%$$



3. Độ đo đánh giá kết quả

3.2 Precision và Recall:

- What is more important, precision or recall?
→ This really **depends on** your **specific** classification problem.

For example:

- Imagine that your classifier needs to detect diabetes in human patients. “**Positive**” means the patient has **diabetes**. “Negative” means that the patient is healthy.
- In this case, you probably want to make sure that your **classifier has high recall**, so that as many diabetics as possible are correctly detected.



3. Độ đo đánh giá kết quả

3.2 Precision và Recall:

- What is more important, precision or recall?

→ This really depends on your specific classification problem.

For example:

- You are building a video recommendation system, and your classifier predicts **Positive** for **a relevant video** and Negative for non-relevant video. You want to make sure that almost all of the recommended **videos are relevant** to the user, so you want **high precision**.



3. Độ đo đánh giá kết quả

3.2 Precision và Recall:

- What is more important, precision or recall?
→ This really depends on your specific classification problem.
- Life is full of trade-offs, and that's also true of classifiers. There's usually a **trade-off between** good **precision** and good **recall**. You usually can't have both.



3. Độ đo đánh giá kết quả

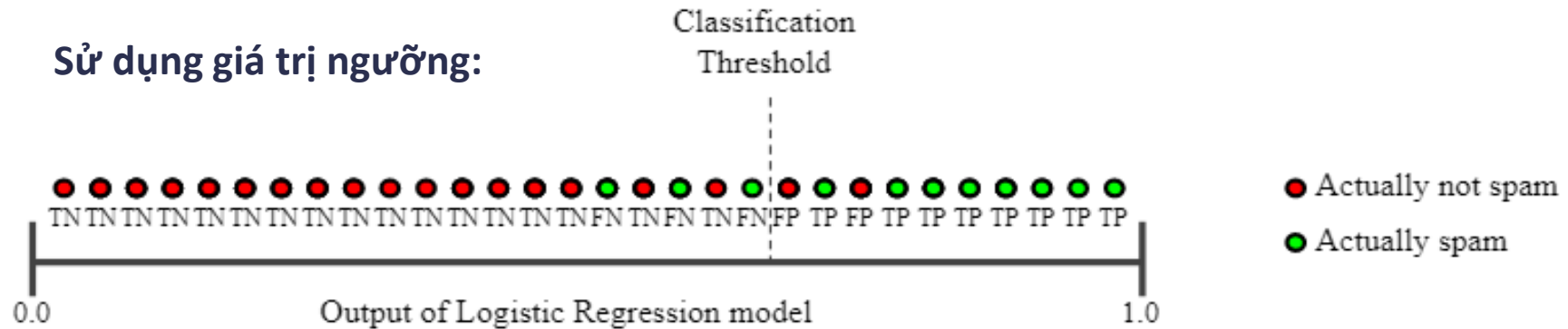
3.2 Precision và Recall:

- Thông thường: precision tăng sẽ làm giảm recall và ngược lại.
- Ví dụ: trong bài toán phân loại email: “spam” hay “not spam” → để xác định kết quả ta so sánh giá trị dự đoán của mô hình so với một giá trị ngưỡng.



3. Độ đo đánh giá kết quả

3.2 Precision và Recall.



True Positives (TP): 8	False Positives (FP): 2
False Negatives (FN): 3	True Negatives (TN): 17

$$Precision = \frac{TP}{TP + FP} = \frac{8}{8 + 2} = 0.80$$

$$Recall = \frac{TP}{TP + FN} = \frac{8}{8 + 3} = 0.73$$

Hình ảnh minh họa được tham khảo từ website:

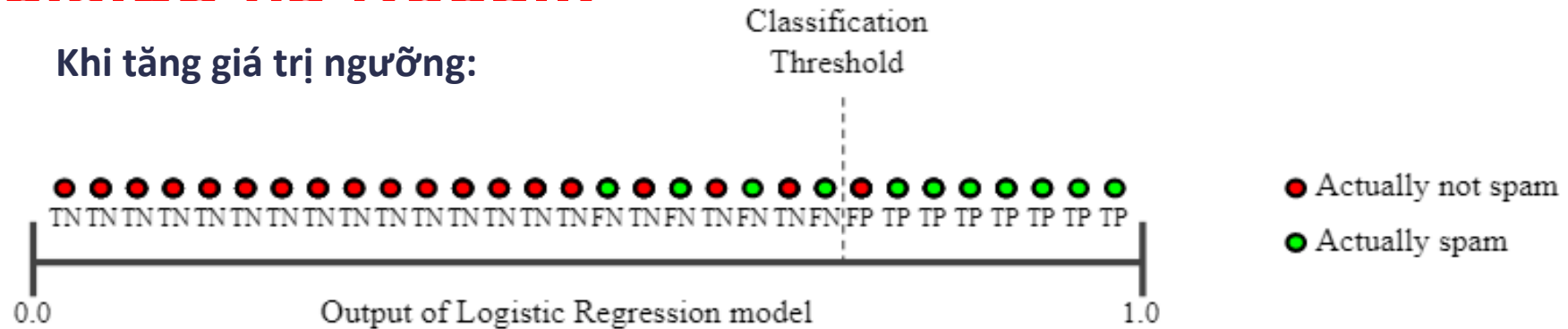
<https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall>



3. Độ đo đánh giá kết quả

3.2 Precision và Recall.

Khi tăng giá trị ngưỡng:



True Positives (TP): 9	False Positives (FP): 3
False Negatives (FN): 2	True Negatives (TN): 16

$$Precision = \frac{TP}{TP + FP} = \frac{9}{9 + 3} = 0.75$$

$$Recall = \frac{TP}{TP + FN} = \frac{9}{9 + 2} = 0.82$$

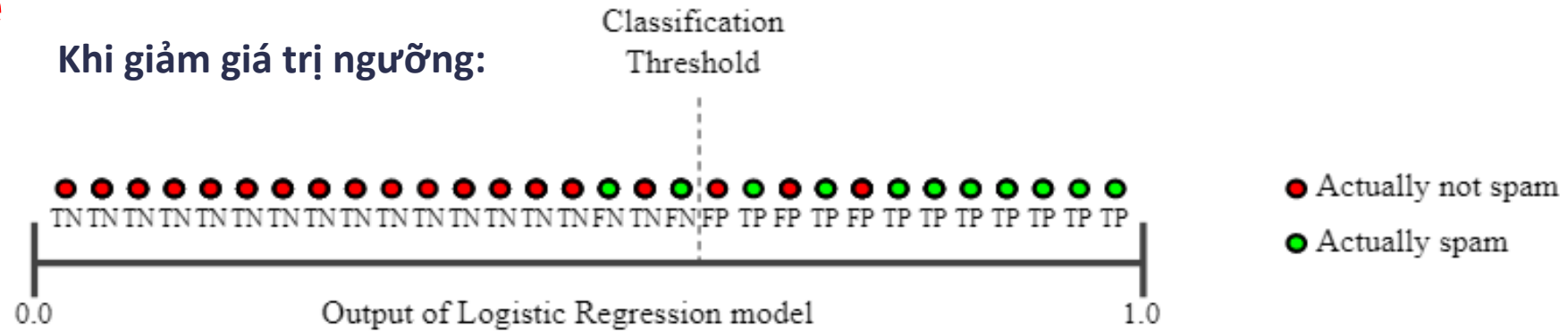
Hình ảnh minh họa được tham khảo từ website:

<https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall>



3. Độ đo đánh giá kết quả

3.2 Precision và Recall.



True Positives (TP): 7	False Positives (FP): 1
False Negatives (FN): 4	True Negatives (TN): 18

$$Precision = \frac{TP}{TP + FP} = \frac{7}{7 + 1} = 0.88$$

$$Recall = \frac{TP}{TP + FN} = \frac{7}{7 + 4} = 0.64$$

Hình ảnh minh họa được tham khảo từ website:

<https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall>



3. Độ đo đánh giá kết quả

3.3 F-score:

- Kết hợp cả precision và recall.
- F1-score is computed using a *harmonic mean*:

$$F_1 - score = \frac{2 * Recall * Precision}{Recall + Precision}$$



3. Độ đo đánh giá kết quả

3.3 F-score:

- Similar to arithmetic mean, the F1-score will always be somewhere in between precision and recall.
- But it behaves differently: the F1-score gives a larger weight to lower numbers.
- For example:
 - When Precision is 100% and Recall is 0%, the F1-score will be 0%, not 50%.



3. Độ đo đánh giá kết quả

3.3 F-score:

- For example:
 - Say that **Classifier A** has precision=recall=80%, and **Classifier B** has precision=60%, recall=100%.
 - Arithmetically, the mean of the precision and recall is the same for both models.
 - But when we use F1's harmonic mean formula, the score for **Classifier A** will be 80%, and for **Classifier B** it will be only 75%.
 - Model B's low precision score pulled down its F1-score.



3. Độ đo đánh giá kết quả

3.4 ROC curve và AUC:

ROC curve:

- ROC curve (receiver operating characteristic curve) là một đồ thị biểu diễn sự hiệu quả (performance) của mô hình phân lớp theo giá trị ngưỡng phân lớp.
- Đồ thị được xác định dựa trên 2 tham số:
 - True Positive Rate
 - False Positive Rate



3. Độ đo đánh giá kết quả

3.4 ROC curve và AUC:

ROC curve:

- True Positive Rate (TP Rate, TPR, tần suất dương tính thật)

$$TPR \equiv Recall = \frac{\text{True Positives}}{\text{All Positives}} = \frac{TP}{TP + FP}$$

- False Positive Rate (FP Rate, FPR
suất dương tính giả)

$$FPR = \frac{\text{False Positives}}{\text{All Negatives}} = \frac{FP}{FP + TN}$$

		Predicted class	
		P	N
Actual Class	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

The true positive rate is also called sensitivity, recall, or hit rate

3. Độ đo đánh giá kết quả

3.4 ROC curve và AUC:

ROC curve:

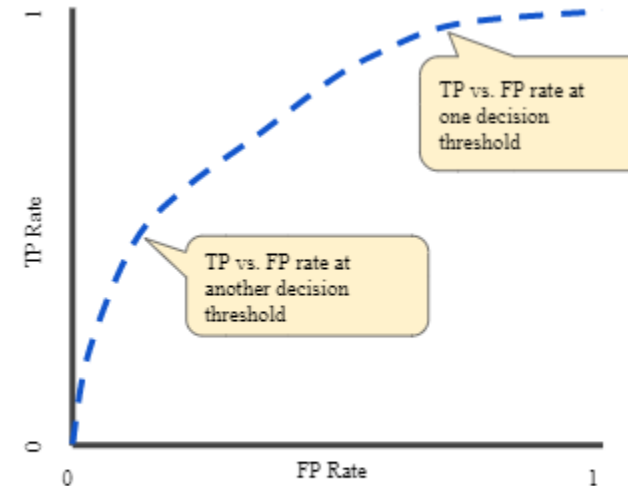


Figure 4. TP vs. FP rate at different classification thresholds.

- Mỗi điểm trên đường cong ROC là tọa độ tương ứng với tần suất dương tính thật (độ nhạy) trên trục tung và tần suất dương tính giả (1-độ đặc hiệu) trên trục hoành.



3. Độ đo đánh giá kết quả

3.4 ROC curve và AUC:

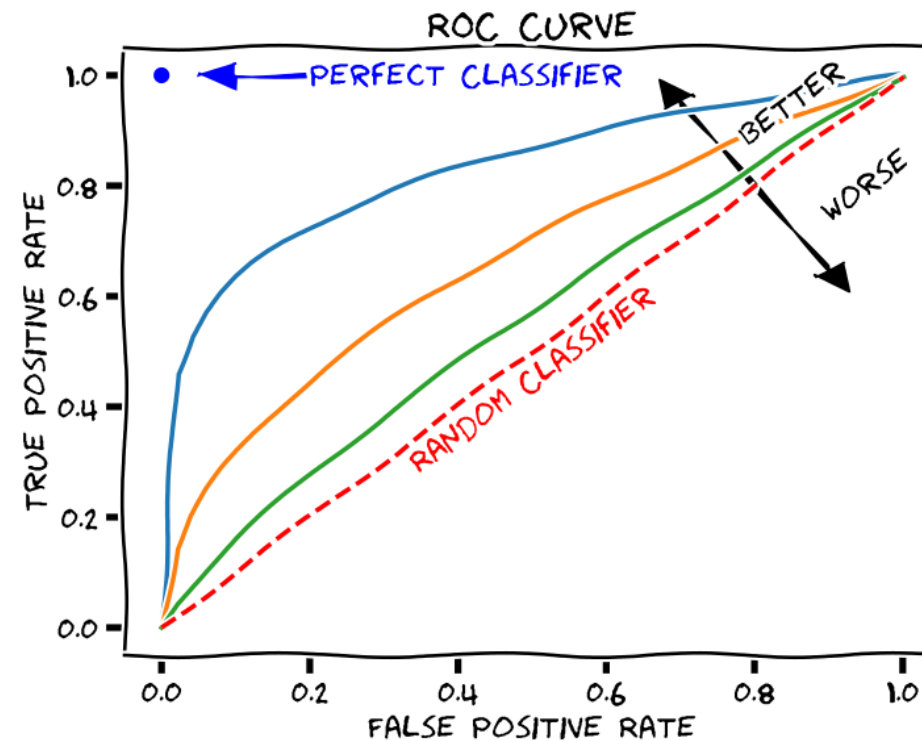
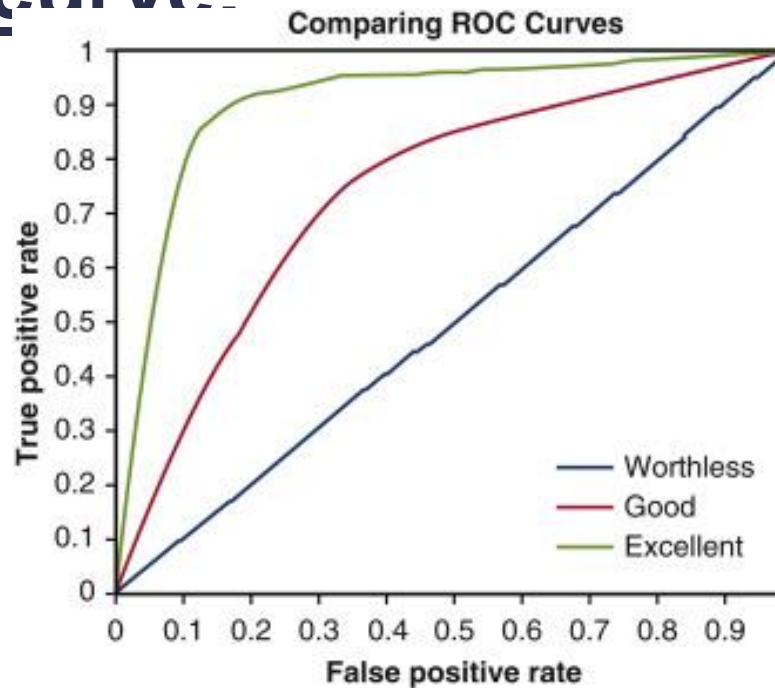
ROC curve:

- Đường biểu diễn càng lệch về phía bên trên và bên trái thì sự phân biệt giữa 2 trạng thái (ví dụ có bệnh hoặc không bệnh) càng rõ.
- Đường cong càng đi dọc theo biên trái và rồi đi dọc theo biên phía trên của không gian ROC, thì chứng tỏ kết quả kiểm tra càng chính xác.
- Đường cong càng tiến tới thành đường chéo 45 độ trong không gian ROC, thì độ chính xác của kiểm tra càng kém.

3. Độ đo đánh giá kết quả

3.4 ROC curve và AUC:

ROC curve:

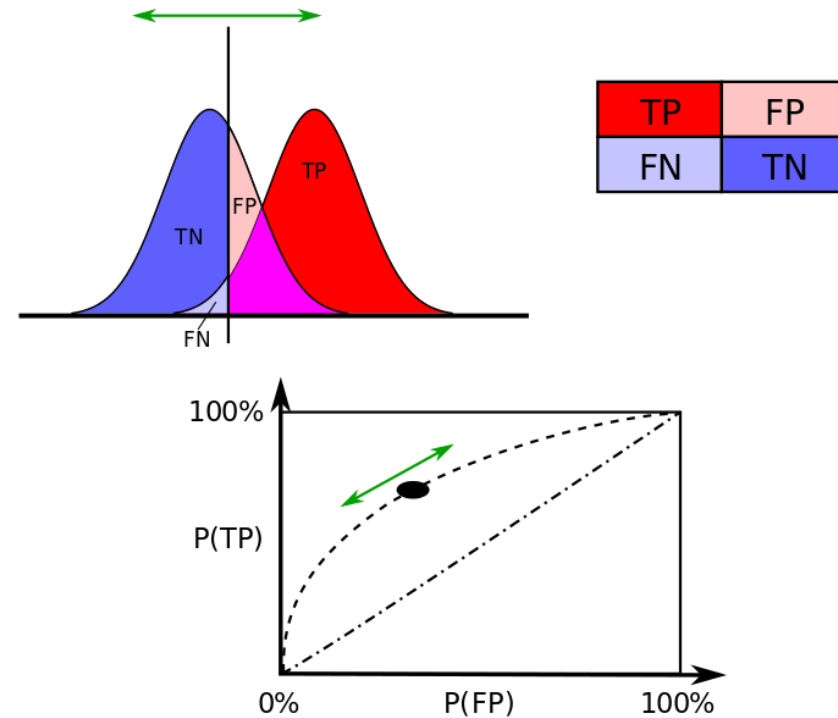


3. Độ đo đánh giá kết quả

3.4 ROC curve và AUC:

ROC curve:

- A ROC curve always starts at the **lower left-hand corner**, i.e. the point ($FPR = 0$, $TPR = 0$) which corresponds to a decision **threshold of 1** (where every example is classified as negative, because all predicted probabilities are less than 1.)

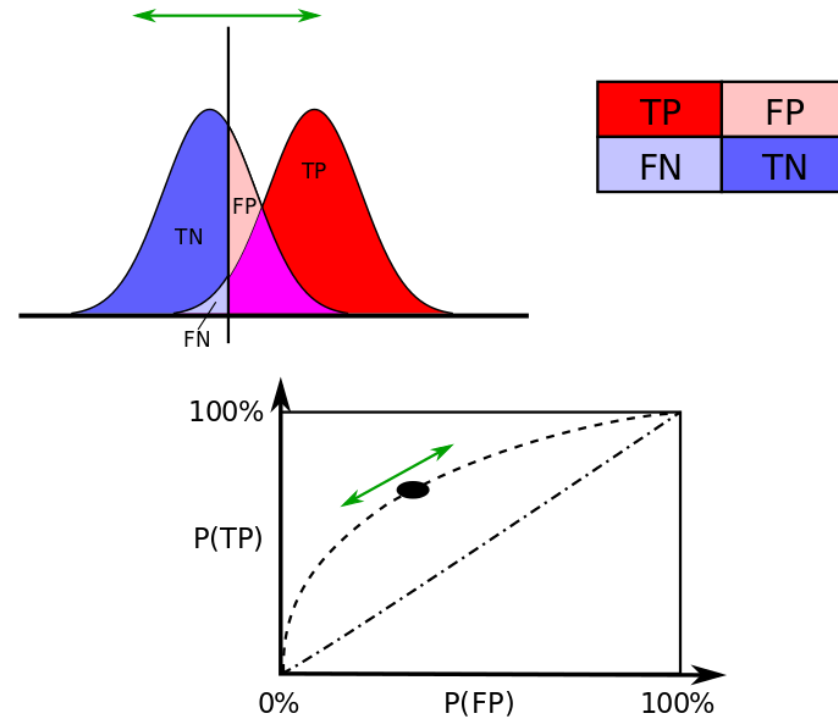


3. Độ đo đánh giá kết quả

3.4 ROC curve và AUC:

ROC curve:

- A ROC curve always ends at the **upper right-hand corner**, i.e. the point (FPR = 1, TPR = 1) which corresponds to a decision **threshold of 0** (where every example is classified as positive, because all predicted probabilities are greater than 0.)



3. Độ đo đánh giá kết quả

3.4 ROC curve và AUC:

ROC curve:

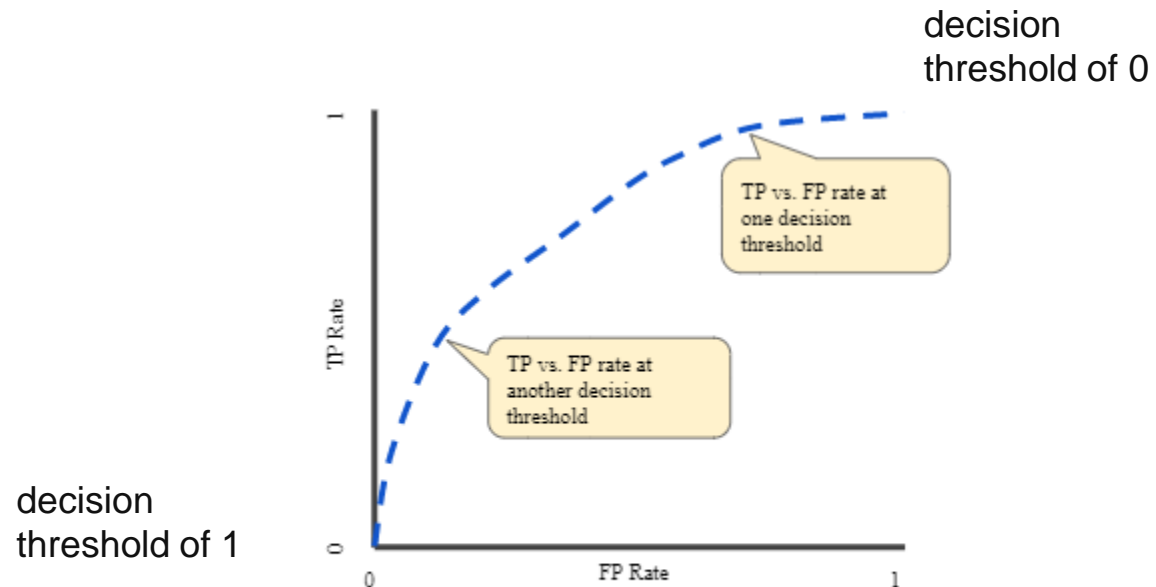


Figure 4. TP vs. FP rate at different classification thresholds.

- Ngưỡng phân lớp càng thấp sẽ có nhiều phần tử được gán nhãn positive hơn → **False Positives** và **True Positives** càng tăng.
- Để tính các điểm trên đường cong ROC, chúng ta có thể sử dụng mô hình logistic regression tại các giá trị ngưỡng khác nhau → không hiệu quả.

3. Độ đo đánh giá kết quả

3.4 ROC curve và AUC:

AUC:

- AUC stands for "Area under the ROC Curve."
- AUC đo diện tích của phần nằm dưới đường cong ROC.

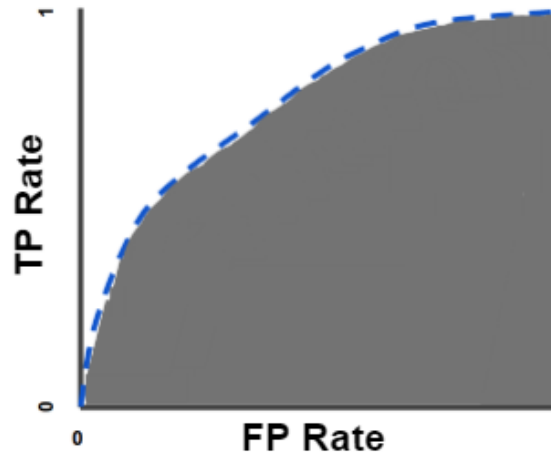


Figure 5. AUC (Area under the ROC Curve).



3. Độ đo đánh giá kết quả

3.4 ROC curve và AUC:

AUC:

- AUC là một thước đo tổng hợp về hiệu suất trên tất cả các ngưỡng phân loại có thể có.
 - One way of interpreting AUC is as the probability that the model ranks a random **positive example** more **highly than** a random **negative example**.
- AUROC is thus a performance metric for “discrimination”

AUC:

0.0 Output of Log. Reg. model 1.0

● Actual Negative
● Actual Positive

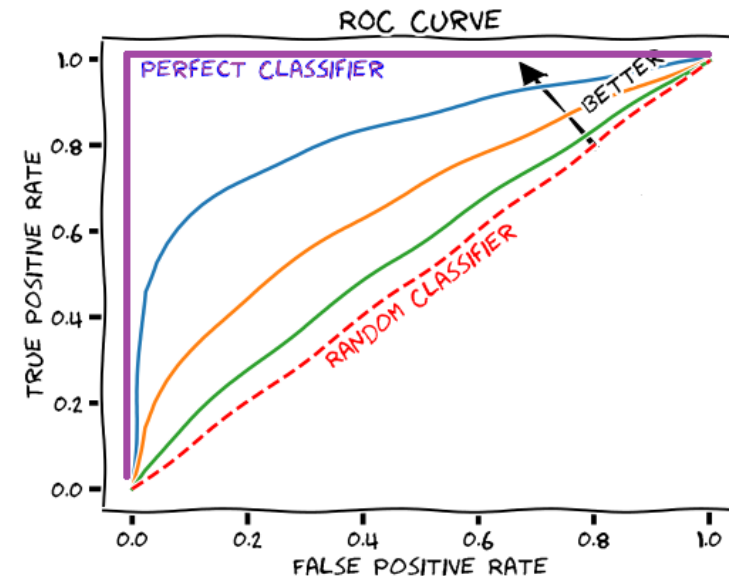
AUC biểu diễn xác suất mà một mẫu ngẫu nhiên positive (green) được đặt phía bên phải một mẫu ngẫu nhiên negative (red).

3. Độ đo đánh giá kết quả

3.4 ROC curve và AUC:

AUC:

- AUC ranges in value from 0 to 1.
- A model whose predictions are 100% wrong has an AUC of 0.0; one whose predictions are 100% correct has an AUC of 1.0.





3. Độ đo đánh giá kết quả

3.4 ROC curve và AUC:

AUC:

- Code trong Python:



```
import sklearn.metrics
```

```
fpr, tpr, thresholds = sklearn.metrics.roc_curve(y_true = true_labels,  
y_score = pred_probs, pos_label = 1) #positive class is 1; negative class  
is 0  
auroc = sklearn.metrics.auc(fpr, tpr)
```



3. Độ đo đánh giá kết quả

3.4 ROC curve và AUC:

AUC:

- (Examples

```
>>> import numpy as np
>>> from sklearn import metrics
>>> y = np.array([1, 1, 2, 2])
>>> scores = np.array([0.1, 0.4, 0.35, 0.8])
>>> fpr, tpr, thresholds = metrics.roc_curve(y, scores, pos_label=2)
>>> fpr
array([0. , 0. , 0.5, 0.5, 1. ])
>>> tpr
array([0. , 0.5, 0.5, 1. , 1. ])
>>> thresholds
array([1.8 , 0.8 , 0.4 , 0.35, 0.1 ])
```

Hình ảnh minh họa được tham khảo từ website:

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html



3. Độ đo đánh giá kết quả

3.5 Ma trận nhập nhằng (Confusion matrix):

- confusion matrix = error matrix = matching matrix
- The general idea is to count the number of times instances of class A are classified as class B

Confusion Matrix

	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)



3. Độ đo đánh giá kết quả

3.5 Ma trận nhập nhằng (Confusion matrix):

- Each row of the matrix represents the instances in a predicted class, while each column represents the instances in an actual class (or vice versa)

Giúp chúng ta dễ dàng nhận diện điểm yếu và điểm mạnh của mô hình phân lớp.

Confusion Matrix

	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)



3. Độ đo đánh giá kết quả

3.5 Ma trận nhập nhằng (Confusion matrix):

Ví dụ:





- Tập test:
 - 13 ảnh = 8 of cats (class 1) + 5 of dogs (class 0),
 - actual = [1,1,1,1,1,1,1,1,0,0,0,0,0]
- Kết quả dự đoán:
 - prediction = [0,0,0,1,1,1,1,1,0]
- Ma trận confusion

Pre- dicted class \ Actual class	Actual class	
	Cat	Dog
Cat	5	2
Dog	3	3

3. Độ đo đánh giá kết quả

3.5 Ma trận nhập nhằng (Confusion matrix):

Ví dụ:

		PREDICTIVE VALUES	
		POSITIVE (CAT)	NEGATIVE (DOG)
ACTUAL VALUES	POSITIVE (CAT)	<p>TRUE POSITIVE</p>  <p>3</p>	<p>FALSE NEGATIVE</p>  <p>1</p> <p>TYPE II ERROR</p>
	NEGATIVE (DOG)	<p>FALSE POSITIVE</p>  <p>2</p> <p>TYPE I ERROR</p>	<p>TRUE NEGATIVE</p>  <p>4</p>

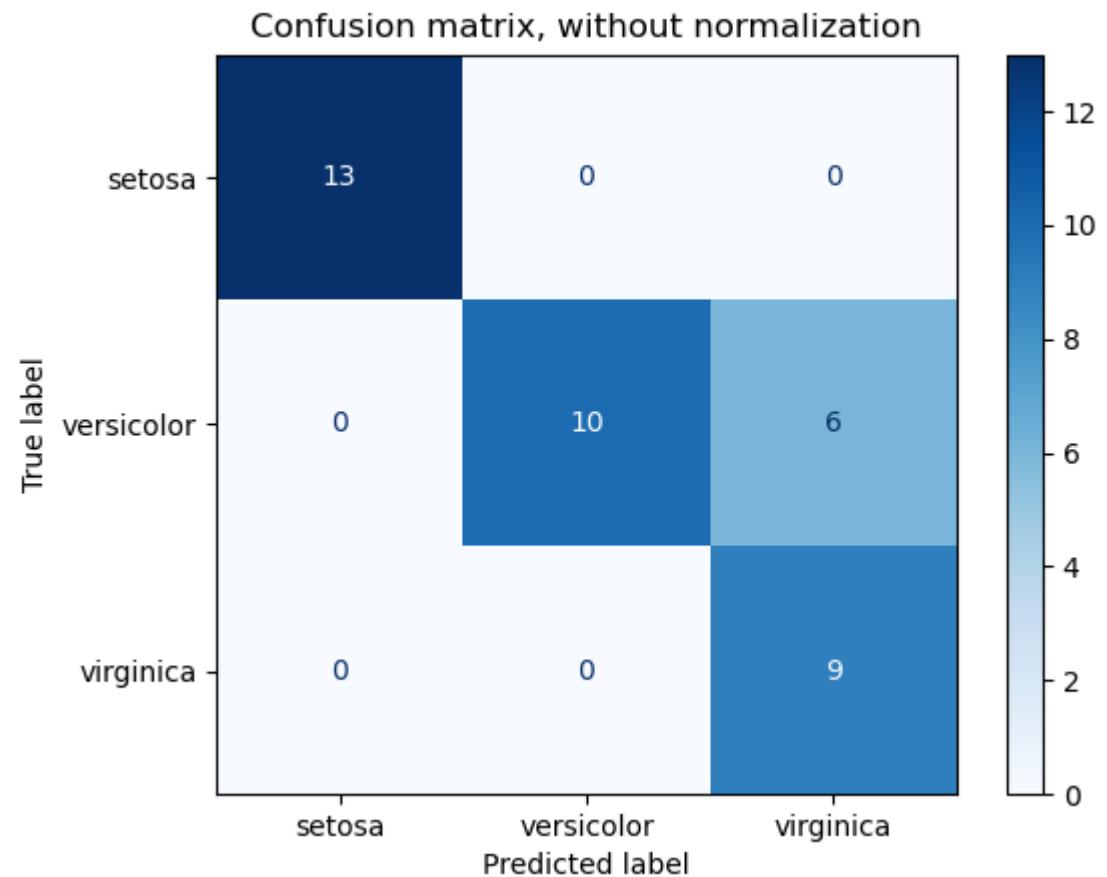
Ví dụ minh họa được tham khảo từ website:

<https://towardsdatascience.com/decoding-the-confusion-matrix-bb4801decbb>



3. Độ đo đánh giá kết quả

3.5 Ma trận nhập nhằng (Confusion matrix):



Hình minh họa được tham khảo từ website:

https://scikit-learn.org/stable/auto_examples/model_selection/plot_confusion_matrix.html



3. Độ đo đánh giá kết quả

3.5 Ma trận nhập nhằng (Confusion matrix):

Code Examples

```
>>> from sklearn.metrics import confusion_matrix
>>> y_true = [2, 0, 2, 2, 0, 1]
>>> y_pred = [0, 0, 2, 2, 0, 2]
>>> confusion_matrix(y_true, y_pred)
array([[2, 0, 0],
       [0, 0, 1],
       [1, 0, 2]])
```

```
>>> y_true = ["cat", "ant", "cat", "cat", "ant", "bird"]
>>> y_pred = ["ant", "ant", "cat", "cat", "ant", "cat"]
>>> confusion_matrix(y_true, y_pred, labels=["ant", "bird", "cat"])
array([[2, 0, 0],
       [0, 0, 1],
       [1, 0, 2]])
```

In the binary case, we can extract true positives, etc as follows:

```
>>> tn, fp, fn, tp = confusion_matrix([0, 1, 0, 1], [1, 1, 1, 0]).ravel()
>>> (tn, fp, fn, tp)
(0, 2, 1, 1)
```

Ví dụ minh họa được tham khảo từ website:

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html



3. Độ đo đánh giá kết quả

We can conclude that:



- **Accuracy** value of **70%** means that identification of 3 of every **10 cats** is incorrect, and **7 is correct**.
- **Precision** value of **60%** means that label of 4 of every **10 cats** is a not a cat (i.e. a dog), and **6 are cats**.
- **Recall** value is **70%** means that 3 of every 10 **cats**, in reality, are missed by our model and **7 are correctly identified as cats**.
- **Specificity** value is **60%** means that 4 of every 10 dogs (i.e. not cat) in reality are miss-labeled as cats and **6 are correctly labeled as dogs**.



3. Độ đo đánh giá kết quả

Bài tập

- Tính Accuracy, Precision, Recall, Specificity, F1-score?

		True/Actual	
		Positive ()	Negative
Predicted	Positive ()	5 (TP)	1 (FP)
	Negative	2 (FN)	2 (TN)

Ví dụ minh họa được tham khảo từ website:

<https://towardsdatascience.com/multi-class-metrics-made-simple-part-ii-the-f1-score-ebe8b2c2ca1>



3. Độ đo đánh giá kết quả

Trường hợp có nhiều hơn 2 lớp

- Accuracy
- Precision, Recall
- F1-score
- macro-average-?
- weighted-average-?



3. Độ đo đánh giá kết quả

Giả sử: sau khi thực hiện phân lớp 25 ảnh vào 3 lớp Cat, Fish và Hen, ta được ma trận confusion:

		True/Actual		
		Cat (🐱)	Fish (🐟)	Hen (🐔)
Predicted	Cat (🐱)	4	6	3
	Fish (🐟)	1	2	0
	Hen (🐔)	1	2	6

$$Precision(Cat) = \frac{4}{4 + 6 + 3} = 30.8\%$$

Ví dụ minh họa được tham khảo từ website:

<https://towardsdatascience.com/multi-class-metrics-made-simple-part-i-precision-and-recall-9250280bddc2>



3. Độ đo đánh giá kết quả

Trường hợp có nhiều hơn 2 lớp

		True/Actual		
		Cat (🐱)	Fish (🐟)	Hen (🐔)
Predicted	Cat (🐱)	4	6	3
	Fish (🐟)	1	2	0
	Hen (🐔)	1	2	6

$$Precision(Cat) = \frac{4}{4 + 6 + 3} = 30.8\%$$

$$Recall(Cat) = \frac{4}{4 + 1 + 1} = 66.7\%$$



3. Độ đo đánh giá kết quả

Trường hợp có nhiều hơn 2 lớp

		True/Actual		
		Cat (🐱)	Fish (🐟)	Hen (🐔)
Predicted	Cat (🐱)	4	6	3
	Fish (🐟)	1	2	0
	Hen (🐔)	1	2	6

$$\text{Precision}(\text{Fish}) = 66.7\%$$

$$\text{Recall}(\text{Fish}) = 20.0\%$$

Ví dụ minh họa được tham khảo từ website:

<https://towardsdatascience.com/multi-class-metrics-made-simple-part-i-precision-and-recall-9250280bddc2>



3. Độ đo đánh giá kết quả

Trường hợp có nhiều hơn 2 lớp

		True/Actual		
		Cat (🐱)	Fish (🐟)	Hen (🐔)
Predicted	Cat (🐱)	4	6	3
	Fish (🐟)	1	2	0
	Hen (🐔)	1	2	6

$$\text{Precision}(\text{Fish}) = 66.7\%$$

$$\text{Recall}(\text{Fish}) = 20.0\%$$

$$\text{Precision}(\text{Hen}) = 66.7\%$$

$$\text{Recall}(\text{Hen}) = 66.7\%$$

Ví dụ minh họa được tham khảo từ website:

<https://towardsdatascience.com/multi-class-metrics-made-simple-part-i-precision-and-recall-9250280bddc2>



3. Độ đo đánh giá kết quả

Giả sử: sau khi thực hiện phân lớp 25 ảnh vào 3 lớp Cat, Fish và Hen, ta được ma trận confusion:

		True/Actual		
		Cat (🐱)	Fish (🐟)	Hen (🐔)
Predicted	Cat (🐱)	4	6	3
	Fish (🐟)	1	2	0
	Hen (🐔)	1	2	6

the precision and recall
for our three classes

Class	Precision	Recall
Cat	30.8%	66.7%
Fish	66.7%	20.0%
Hen	66.7%	66.7%

Ví dụ minh họa được tham khảo từ website:

<https://towardsdatascience.com/multi-class-metrics-made-simple-part-ii-the-f1-score-ebe8b2c2ca1>



3. Độ đo đánh giá kết quả

Giả sử: sau khi thực hiện phân lớp 25 ảnh vào 3 lớp Cat, Fish và Hen, ta được ma trận confusion:

		True/Actual		
		Cat (🐱)	Fish (🐟)	Hen (🐔)
Predicted	Cat (🐱)	4	6	3
	Fish (🐟)	1	2	0
	Hen (🐔)	1	2	6

Class	Precision	Recall	F1-score
Cat	30.8%	66.7%	42.1%
Fish	66.7%	20.0%	30.8%
Hen	66.7%	66.7%	66.7%

Ví dụ minh họa được tham khảo từ website:

<https://towardsdatascience.com/multi-class-metrics-made-simple-part-ii-the-f1-score-ebe8b2c2ca1>



3. Độ đo đánh giá kết quả

Trường hợp có nhiều hơn 2 lớp

```
1  from sklearn import metrics
2
3  # Constants
4  C="Cat"
5  F="Fish"
6  H="Hen"
7
8  # True values
9  y_true = [C,C,C,C,C,C, F,F,F,F,F,F,F,F,F,F, H,H,H,H,H,H,H,H,H]
10 # Predicted values
11 y_pred = [C,C,C,C,H,F, C,C,C,C,C,C,H,H,F,F, C,C,C,H,H,H,H,H,H]
12
13 # Print the confusion matrix
14 print(metrics.confusion_matrix(y_true, y_pred))
15
16 # Print the precision and recall, among other metrics
17 print(metrics.classification_report(y_true, y_pred, digits=3))
```

Ví dụ minh họa được tham khảo từ website:

<https://towardsdatascience.com/multi-class-metrics-made-simple-part-i-precision-and-recall-9250280bddc2>



3. Độ đo đánh giá kết quả

Trường hợp có nhiều hơn 2 lớp

And here is the output. Note the confusion matrix is transposed here — that's just the way sklearn works. Notice the **support** column: it lists the number of samples for each class (6 for Cat, 10 for Fish, etc).

```
[[4 1 1]
 [6 2 2]
 [3 0 6]]
```

	precision	recall	f1-score	support
Cat	0.308	0.667	0.421	6
Fish	0.667	0.200	0.308	10
Hen	0.667	0.667	0.667	9
accuracy			0.480	25
macro avg	0.547	0.511	0.465	25
weighted avg	0.581	0.480	0.464	25

Ví dụ minh họa được tham khảo từ website:

<https://towardsdatascience.com/multi-class-metrics-made-simple-part-i-precision-and-recall-9250280bddc2>



3. Độ đo đánh giá kết quả

Trường hợp có nhiều hơn 2 lớp

- macro-averaged F1-score, or the macro-F1

$$\text{Macro} - F1 = \frac{1}{C} \sum_{i=1}^C \frac{2 * \text{Precision}_i * \text{Recall}_i}{\text{Precision}_i + \text{Recall}_i} = \frac{1}{C} \sum_{i=1}^C F1 - \text{score}_i$$

trong đó:

- C : là số lượng lớp
- $F1 - \text{score}_i$ là giá trị độ đo F1-score của lớp thứ i



3. Độ đo đánh giá kết quả

Trường hợp có nhiều hơn 2 lớp

- macro-averaged Precision/Recall, or the macro-Precision/Recall

$$Macro - Precision = \frac{1}{C} \sum_{i=1}^C Precision_i$$

$$Macro - Recall = \frac{1}{C} \sum_{i=1}^C Recall_i$$



3. Độ đo đánh giá kết quả

Trường hợp có nhiều hơn 2 lớp

- weight-averaged F1-score, or the weight-F1
- When averaging the macro-F1, we gave equal weights to each class.
- We don't have to do that: in weighted-average F1-score, or weighted-F1, we weight the F1-score of each class by the number of samples from that class.



3. Độ đo đánh giá kết quả

- **weight-averaged F1-score, or the weight-F1**
- We have a total of 25 samples: 6 Cat, 10 Fish, and 9 Hen. The weighted-F1 score is thus computed as follows:

$$\text{Weighted-F1} = (6 \times 42.1\% + 10 \times 30.8\% + 9 \times 66.7\%) / 25 = 46.4\%$$

Similarly, we can compute weighted precision and weighted recall:

$$\text{Weighted-precision} = (6 \times 30.8\% + 10 \times 66.7\% + 9 \times 66.7\%) / 25 = 58.1\%$$

$$\text{Weighted-recall} = (6 \times 66.7\% + 10 \times 20.0\% + 9 \times 66.7\%) / 25 = 48.0\%$$

Ví dụ minh họa được tham khảo từ website:

<https://towardsdatascience.com/multi-class-metrics-made-simple-part-ii-the-f1-score-ebe8b2c2ca1>



3. Độ đo đánh giá kết quả

Trường hợp có nhiều hơn 2 lớp

Độ chính xác

- Thông thường, kết quả phân lớp là một ma trận $R_{T \times M}$, với T là số lượng ảnh cần phân lớp và M là số lượng lớp.
- Mỗi giá trị $R_{i,j} \in \mathbb{R}$ cho biết xác suất ảnh i thuộc vào lớp j .



3. Độ đo đánh giá kết quả

Trường hợp có nhiều hơn 2 lớp

- Độ chính xác theo **top-1**: dùng để đánh giá kết quả gán nhãn của các ảnh có đúng hay không, mỗi ảnh chỉ được gán vào một lớp.



3. Độ đo đánh giá kết quả

Trường hợp có nhiều hơn 2 lớp

- Độ chính xác theo **top-5**: dùng để xác định xem **trong 5 kết quả** phân lớp được gán cho một ảnh, **có kết quả đúng** nào không. Việc sử dụng độ đo này là do trong một ảnh có thể có nhiều đối tượng khác nhau, nếu chỉ gán ảnh vào một lớp có thể không đánh giá được hiệu quả của phương pháp.



3. Độ đo đánh giá kết quả

Trường hợp có nhiều hơn 2 lớp

- Một số độ đo khác:
 - Độ lỗi (top-1, top-5,...)
 - Thời gian thực thi (phụ thuộc vào phần cứng)



3. Độ đo đánh giá kết quả

Trường hợp có nhiều hơn 2 lớp

- Bài tập

		True Class		
		Apple	Orange	Mango
Predicted Class	Apple	7	8	9
	Orange	1	2	3
	Mango	3	2	1



3. Độ đo đánh giá kết quả

Trường hợp có nhiều hơn 2 lớp

- Bài tập

	Target	Selected							
	1	2	3	4	5	6	7	8	9
1	137	13	3	0	0	1	1	0	0
2	1	55	1	0	0	0	0	6	1
3	2	4	84	0	0	0	1	1	2
4	3	0	1	153	5	2	1	1	1
5	0	0	3	0	44	2	2	1	2
6	0	0	2	1	4	35	0	0	1
7	0	0	0	0	0	0	61	2	2
8	0	0	0	1	0	0	0	69	3
9	0	0	0	0	0	0	0	2	26



#importing a 3-class dataset from sklearn's toy dataset

```
from sklearn.datasets import load_wine
```

```
dataset = load_wine()
```

```
X = dataset.data
```

```
y = dataset.target
```

```
from sklearn.model_selection import train_test_split
```

```
from sklearn.svm import SVC
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y,  
random_state=0)
```

```
svc = SVC(kernel='rbf', C=1).fit(X_train, y_train)
```

```
y_pred = svc.predict(X_test)
```



#importing confusion matrix

```
from sklearn.metrics import confusion_matrix
```

```
confusion = confusion_matrix(y_test, y_pred)
```

```
print('Confusion Matrix\n')
```

```
print(confusion)
```



```
#importing accuracy_score, precision_score, recall_score, f1_score
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

print('\nAccuracy: {:.2f}\n'.format(accuracy_score(y_test, y_pred)))

print('Micro Precision: {:.2f}'.format(precision_score(y_test, y_pred, average='micro')))
print('Micro Recall: {:.2f}'.format(recall_score(y_test, y_pred, average='micro')))
print('Micro F1-score: {:.2f}\n'.format(f1_score(y_test, y_pred, average='micro')))

print('Macro Precision: {:.2f}'.format(precision_score(y_test, y_pred, average='macro')))
print('Macro Recall: {:.2f}'.format(recall_score(y_test, y_pred, average='macro')))
print('Macro F1-score: {:.2f}\n'.format(f1_score(y_test, y_pred, average='macro')))

print('Weighted Precision: {:.2f}'.format(precision_score(y_test, y_pred,
average='weighted')))
print('Weighted Recall: {:.2f}'.format(recall_score(y_test, y_pred, average='weighted')))
print('Weighted F1-score: {:.2f}'.format(f1_score(y_test, y_pred, average='weighted')))
```



```
from sklearn.metrics import classification_report
```

```
print('\nClassification Report\n')
```

```
print(classification_report(y_test, y_pred, target_names=['Class 1',  
'Class 2', 'Class 3']))
```



CONFUSION MATRIX:

```
[[15  0  1]
 [ 0 17  4]
 [ 0  3  5]]
```

Accuracy: 0.82

Micro Precision: 0.82

Micro Recall: 0.82

Micro F1-score: 0.82

Macro Precision: 0.78

Macro Recall: 0.79

Macro F1-score: 0.78

Weighted Precision: 0.84

Weighted Recall: 0.82

Weighted F1-score: 0.83

CLASSIFICATION REPORT:

	precision	recall	f1-score	support
Class 1	1.00	0.94	0.97	16
Class 2	0.85	0.81	0.83	21
Class 3	0.50	0.62	0.56	8
accuracy			0.82	45
macro avg	0.78	0.79	0.78	45
weighted avg	0.84	0.82	0.83	45





3. Độ đo đánh giá kết quả

Độ mất mát (loss):

- Đánh giá sự sai khác giá trị

$$Loss = - \sum_{i=1}^N y_i \cdot \log(\hat{y}_i)$$

Kết quả dự đoán \hat{y}_i là một giá trị xác suất (0-1),