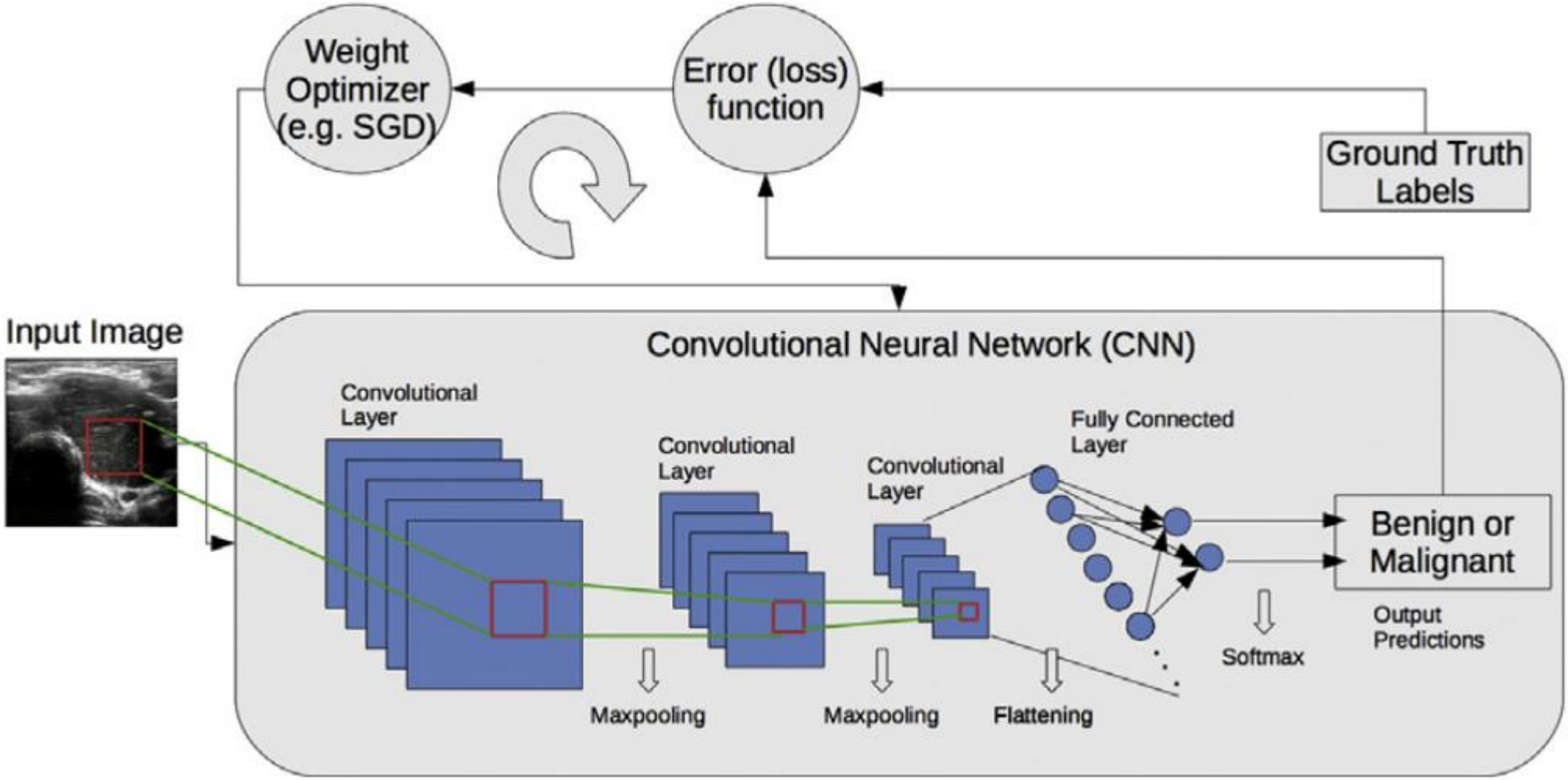
CHƯƠNG 3: HUẤN LUYỆN HỌC SÂU

TỔNG QUAN

SGD = stochastic gradient descent (giảm độ dốc ngẫu nhiên)



TỔNG QUAN

- Để thực hiện dự đoán từ dữ liệu đầu vào, điểm số đầu ra của lớp CNN cuối cùng được kết nối với hàm phi tuyến tính softmax giúp chuẩn hóa điểm số thành phân phối đa thức trên các nhãn.
- Ngoài ra, một trình tối ưu hóa giúp giảm thiểu lỗi giữa dự đoán và nhãn thực tế thông qua hàm mất mát và phương pháp lan truyền ngược độ dốc (cập nhật trọng số ở mỗi lần lặp) được sử dụng để huấn luyện kiến trúc CNN cho đến khi chúng hội tụ về trạng thái ổn định.

TỔNG QUAN

- Bài toán: detect nhiều đối tượng trong 1 ảnh có độ phân giải cao → cần huấn luyện một DNN chứa nhiều lớp, mỗi lớp chứa hàng trăm nơ-ron được kết nối bởi hàng trăm nghìn kết nối?
- Các vấn đề đặt ra:

 Đối mặt với vấn đề Vanishing/Exploding Gradients → ảnh hưởng đến các DNN và khiến các lớp thấp hơn rất khó huấn luyện.

 Có thể không có đủ dữ liệu huấn luyện cho một mạng lớn như vậy hoặc có thể quá tốn kém để gắn nhãn.

 Huấn luyện có thể cực kỳ chậm.

 Mô hình với hàng triệu tham số sẽ có nguy cơ quá khớp với tập huấn luyện (overfitting), đặc biệt nếu không có đủ các trường hợp huấn luyện hoặc quá nhiễu.

TỔNG QUAN

➢ Để giải quyết vấn đề:

 Tìm hiểu về Vanishing/Exploding Gradients và khám phá một số giải pháp phổ biến nhất cho vấn đề này.

 Xem xét phương pháp học chuyển đổi (transfer learning) và huấn luyện trước không giám sát (unsupervised pretraining) → giải quyết các tác vụ phức tạp ngay cả khi có ít dữ liệu được gắn nhãn.

 Tìm hiểu về các trình tối ưu hóa (optimizers) khác nhau có thể tăng tốc độ huấn luyện các mô hình lớn so với Gradient Descent đơn giản.

 Tìm hiểu các kỹ thuật chuẩn hóa phổ biến cho các mạng thần kinh.

 → có thể huấn luyện các mạng rất sâu:

BIAS VÀ VARIANCE

- **Bias**: là sai số giữa giá trị dự đoán trung bình của mô hình và giá trị thực tế
  - High bias: sai số lớn, mô hình đơn giản, tuy nhiên kết quả dự đoán chính xác không cao
  - Low bias: sai số nhỏ, mô hình phức tạp, kết quả dự đoán tốt
- **Variance**: là sai số thể hiện mức độ “nhạy cảm” của mô hình với những biến động trong dữ liệu huấn luyện
  - Low-variance: mô hình ít biến thiên theo sự thay đổi của dữ liệu huấn luyện
  - High-variance: mô hình biến thiên mạnh, bám sát theo sự thay đổi của dữ liệu huấn luyện

OVERFITTING THE TRAINING DATA

- Mô hình hoạt động tốt trên dữ liệu huấn luyện, nhưng nó **không tổng quát hóa tốt**.

- Là hiện tượng mà mô hình có **low bias và high variance**.

- Mô hình cho kết quả rất tốt trên dữ liệu đã được học, nhưng cho kết quả tệ trên dữ liệu chưa từng gặp bao giờ.

- Vấn đề này xảy ra khi mô hình cố gắng fit tất cả các điểm dữ liệu huấn luyện, bao gồm cả nhiễu.

-> Hạn chế overfitting: **Regularization**

 Regularization: penalty đối với độ phức tạp của một mô hình (model) -> giúp ngăn chặn việc overfitting.

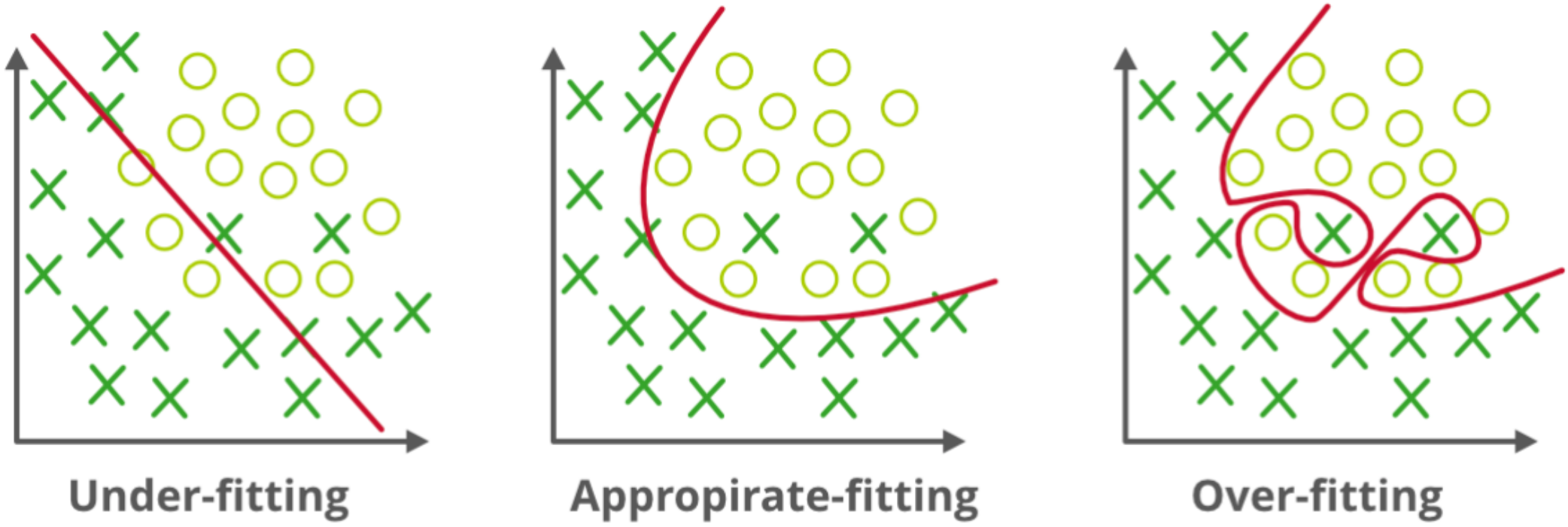
UNDERFITTING THE TRAINING DATA

- Là hiện tượng mà mô hình có **high bias** và **low varriance.**

- Cho kết quả dự đoán không tốt trên cả tập huấn luyện và tập kiểm tra vì mô hình chưa nắm bắt được độ phức tạp của tập dữ liệu huấn luyện (training data).

- Không thể mô hình hóa dữ liệu huấn luyện (training data) cũng như không tổng quát hóa (generalize) dữ liệu mới.

UNDERFITTING THE TRAINING DATA



HYPERPARAMETER

- Là các tham số (parameter) mà giá trị của nó **kiểm soát quá trình học** (learning) và **xác định giá trị của các tham số của mô hình** (model parameter) mà một thuật toán học (learning algorithm) kết thúc việc học.
- "**hyper_**": là các tham số "**cấp cao nhất**" kiểm soát quá trình học và các tham số của mô hình là kết quả.
- Khi thiết kế mô hình (model), **chọn và thiết lập các giá trị của siêu tham số** (hyperparameter) mà thuật toán học sẽ sử dụng **trước khi quá trình huấn luyện** (training) **bắt đầu** → nằm ngoài mô hình vì mô hình không thể thay đổi các giá trị của nó trong quá trình học /huấn luyện.

HYPERPARAMETER

- Một số ví dụ phổ biến về siêu tham số (hyperparameter) như:

  - Tỷ lệ phân chia (split ratio) tập huấn luyện - tập thử nghiệm
  - Tốc độ học (learning rate) trong các thuật toán tối ưu (optimization algorithm) (ví dụ: gradient descent)
  - Sự lựa chọn hàm tổn thất/mất mát (loss function) mà mô hình sẽ sử dụng

VANISHING/ EXPLODING GRADIENT

- **Vanishing gradient**: là vấn đề xảy ra khi huấn luyện các mạng nơ ron nhiều lớp. Khi huấn luyện, giá trị đạo hàm là thông tin phản hồi của quá trình lan truyền ngược. Giá trị này trở nên **vô cùng nhỏ tại các lớp nơ-ron đầu tiên** khiến cho việc cập nhật trọng số mạng không thể xảy ra.
- **Exploding gradient**: ngược với vanishing gradient, giá trị này là **lớn**, khi đó gradient tăng nhanh và sẽ xảy ra hiện tượng tràn số.

Đánh giá hiệu năng hệ thống học máy

- Trong thực tế, một bài toán Machine Learning có thể được giải quyết bởi nhiều phương pháp, cho ra những mô hình khác nhau.

- Đứng trước nhiều sự lựa chọn, làm cách nào để chúng ta có thể chọn ra mô hình phù hợp nhất cho bài toán đang giải quyết?

- → Cần phải đánh giá "hiệu năng" của mô hình trên dữ liệu mới

Đánh giá hiệu năng hệ thống học máy

Đánh giá mô hình giúp ta trả lời những câu hỏi sau:

- Mô hình đã được huấn luyện thành công hay chưa?
- Mức độ thành công của mô hình tốt đến đâu?
- Khi nào nên dừng quá trình huấn luyện?
- Khi nào nên cập nhật mô hình?

Đánh giá hiệu năng hệ thống học máy

- Việc **đánh giá mô hình** thường được thực hiện trên dữ liệu mà mô hình chưa từng được học – trên **validation set và test set**

- Những bài toán khác nhau sẽ có những tiêu chí đánh giá khác nhau, vì vậy cần phải xác định rõ thứ tự ưu tiên của các tiêu chí cho việc đánh giá mô hình.

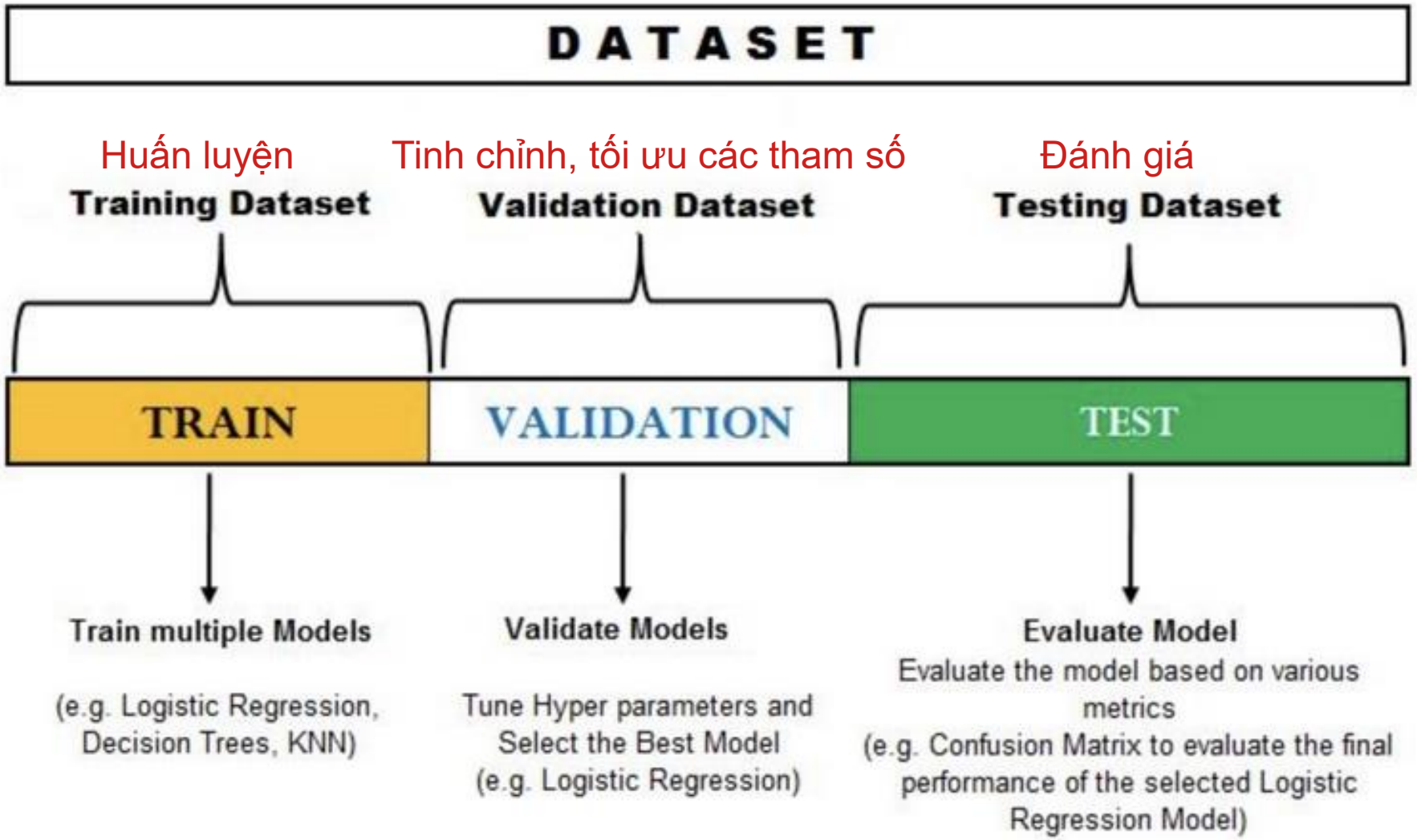
- Bên cạnh đó, đối với những tiêu chí phụ thuộc vào nhau, tức nếu cải thiện "hiệu năng" ở tiêu chí A thì tiêu chí B sẽ có "hiệu năng" thấp, cần phải đánh đổi (tradeoff) giữa các tiêu chí, thì chúng ta cần sử dụng các độ đo (metric) nhằm thuận tiện hơn cho việc đánh giá mô hình

Đánh giá hiệu năng hệ thống học máy

- Các **phương pháp đánh giá (evaluation methods**)
  - Làm sao có được một đánh giá đáng tin cậy về hiệu năng của hệ thống

- Các **tiêu chí đánh giá (evaluation metrics)**
  - Làm sao để đo (tính toán) hiệu năng của hệ thống

Đánh giá hiệu năng hệ thống học máy



Các phương pháp đánh giá

- Làm thế nào để thu được một đánh giá đáng tin cậy về hiệu năng của hệ thống?
  - Tập huấn luyện càng lớn, thì hiệu năng của hệ thống học càng tốt
  - Tập kiểm thử càng lớn, thì việc đánh giá càng chính xác
  - Vấn đề: Rất khó (ít khi) có thể có được các tập dữ liệu (rất) lớn

- Hiệu năng của hệ thống không chỉ phụ thuộc vào giải thuật học máy được sử dụng, mà còn phụ thuộc vào:
  - Phân bố lớp (Class distribution)
  - Chi phí của việc phân lớp sai (Cost of misclassification)
  - Kích thước của tập kiểm thử (Size of the test set)

Các phương pháp đánh giá

- Hold-out
- Stratified sampling
- Repeated hold-out
- Cross-validation
  - K-fold
  - Leave-one-out
- Bootstrap sampling

Hold-out (Splitting)

- Toàn bộ tập dữ liệu D được chia thành 2 tập con **không giao nhau**
  - Tập huấn luyện D_train để huấn luyện hệ thống
  - Tập kiểm thử D_test – để đánh giá hiệu năng của hệ thống đã học

- Các yêu cầu
  - Bất kỳ dữ liệu nào thuộc vào tập kiểm thử D_test đều không được sử dụng trong quá trình huấn luyện hệ thống
  - Bất kỳ dữ liệu nào được sử dụng trong giai đoạn huấn luyện hệ thống đều không được sử dụng trong giai đoạn đánh giá hệ thống
  - Các dữ liệu kiểm thử trong D_test cho phép một đánh giá không thiên vị đối với hiệu năng của hệ thống

Stratified sampling

- Đối với các tập dữ liệu có kích thước nhỏ hoặc không cân xứng (Unbalanced datasets), các dữ liệu trong tập huấn luyện và thử nghiệm có thể không phải là đại diện

- Ví dụ: có (rất) ít, hoặc không có, các dữ liệu đối với một số lớp

- Mục tiêu: phân bố lớp (Class distribution) trong tập huấn luyện và tập kiểm thử phải xấp xỉ như trong toàn bộ tập dữ liệu

Stratified sampling

- Lấy mẫu phân tầng (Stratified sampling)
  - Là một phương pháp để cân xứng (về phân bố lớp)
  - Đảm bảo tỷ lệ phân bố lớp (tỷ lệ dữ liệu giữa các lớp) trong tập huấn luyện và tập kiểm thử là xấp xỉ nhau

- Phương pháp lấy mẫu phân tầng không áp dụng được cho bài toán học máy dự đoán/ hồi quy (vì giá trị đầu ra của hệ thống là một giá trị số, không phải là một nhãn lớp)

Repeated hold-out

- Áp dụng phương pháp đánh giá Hold-out nhiều lần, để sinh ra (sử dụng) các tập huấn luyện và thử nghiệm khác nhau
- Trong mỗi bước lặp, một tỷ lệ nhất định của tập D **được lựa chọn ngẫu nhiên** để tạo nên tập huấn luyện (có thể sử dụng kết hợp với phương pháp lấy mẫu phân tầng - Stratified sampling)
- Các giá trị lỗi (hoặc các giá trị đối với các tiêu chí đánh giá khác) ghi nhận được trong các bước lớp này được lấy trung bình cộng (Average) để xác định giá trị lỗi tổng thể

Repeated hold-out

- Phương pháp này vẫn không hoàn hảo
- Mỗi bước lặp sử dụng một tập kiểm thử khác nhau
- Có một số dữ liệu trùng lặp (được sử dụng lại nhiều lần) trong các tập kiểm thử này

Cross-validation

- Để tránh việc trùng lặp giữa các tập kiểm thử (một số dữ liệu cùng xuất hiện trong các tập kiểm thử khác nhau)

- k-fold cross-validation
- Tập toàn bộ các dữ liệu được chia thành k tập con **không giao nhau** (gọi là “fold”) có kích thước xấp xỉ nhau
- Mỗi lần (trong số k lần) lặp, một tập con được sử dụng làm tập kiểm thử và k-1 tập con còn lại được dùng làm tập huấn luyện
- K giá trị lỗi (mỗi giá trị tương ứng với một fold) được tính trung bình cộng để thu được giá trị lỗi tổng thể

Cross-validation

- Các lựa chọn thông thường của k: 10 hoặc 5

- Thông thường, mỗi tập con (fold) được lấy mẫu phân tầng (xấp xỉ phân bố lớp) trước khi áp dụng quá trình đánh giá Cross-validation

- Phù hợp khi ta có tập dữ liệu vừa và nhỏ

Leave-one-out Cross-validation

- Một trường hợp (kiểu) của phương pháp Cross-validation
  - Số lượng các nhóm (folds) bằng kích thước của tập dữ liệu
  - Mỗi nhóm (fold) chỉ bao gồm một dữ liệu
- Khai thác tối đa tập dữ liệu ban đầu
- Không có bước lấy mẫu ngẫu nhiên (no random sub-sampling)

Leave-one-out Cross-validation

- Áp dụng lấy mẫu phân tầng (Straification) không phù hợp
  - Vì ở mỗi bước lặp, tập thử nghiệm chỉ gồm có một dữ liệu
- Chi phí tính toán (rất) cao
- Phù hợp khi có tập dữ liệu (rất) nhỏ

Boostrap sampling

- Phương pháp Cross-validation sử dụng việc lấy mẫu không lặp lại (Sampling without replacement)
- Đối với mỗi dữ liệu, một khi đã được chọn (được sử dung) thì không thể được chọn (sử dụng) lại cho tập huấn luyện

- Phù hợp với tập dữ liệu có kích thước (rất) nhỏ

- Phương pháp Bootstrap sampling sử dụng việc **lấy mẫu có lặp lại (Sampling with replacement)** để tạo nên tập huấn luyện

Boostrap sampling

- Giả sử tập dữ liệu D gồm n mẫu dữ liệu

- Lấy mẫu có lặp lại n lần đối với tập D, để tạo nên tập huấn luyện D_train gồm n mẫu dữ liệu
  - Từ tập D, lấy ra **ngẫu nhiên** một mẫu dữ liệu X (nhưng không loại bỏ X ra khỏi tập D)
  - Đưa mẫu dữ liệu X vào trong tập huấn luyện D_train
  - Lặp lại 2 bước trên n lần

- Sử dụng tập D_train để huấn luyện hệ thống

- Sử dụng tất cả dữ liệu thuộc D nhưng không thuộc D_train để tạo nên tập thử nghiệm D_test

Các tiêu chí đánh giá

- **Tính chính xác (Accuracy)**

  - Mức độ dự đoán (phân lớp) chính xác của hệ thống (đã được huấn luyện) đối với các dữ liệu kiểm tra (Test instances)

- **Tính hiệu quả (Eficiency)**

  - Chi phí về thời gian và tài nguyên (bộ nhớ) cần thiết cho việc huấn luyện và kiểm thử hệ thống

- **Khả năng xử lý nhiễu (Robustness)**

  - Khả năng xử lý (chịu được) của hệ thống đối với các dữ liệu nhiễu (lỗi) hoặc thiếu giá trị.

Các tiêu chí đánh giá

- **Khả năng mở rộng (Scalability)**
  - Hiệu năng của hệ thống (tốc độ học/ phân loại) thay đổi như thế nào đối với kích thước của tập dữ liệu

- **Khả năng diễn giải (Interpretability)**
  - Mức độ dễ hiểu (đối với người sử dụng) của các kết quả và hoạt động của hệ thống

- **Mức độ phức tạp (Complexity)**
  - Mức độ phức tạp của mô hình hệ thống (hàm mục tiêu) học được

Tính chính xác

- **Đối với bài toán phân loại**
  - Giá trị (kết quả) đầu ra của hệ thống là một giá trị định danh

$$Accuracy = \frac{1}{|D\_test|} \sum_{x \in D\_test} Identical(o(x), c(x));$$

$$Identical(a,b) = \begin{cases} 1, \text{if} & (a = b) \\ 0, \text{if otherwise} \end{cases}$$

  - Trong đó
    - x: dữ liệu trong tập D_test
    - o(x): giá trị đầu ra (phân lớp) bởi hệ thống đối với dữ liệu x
    - c(x): phân lớp thực sự (đúng) của dữ liệu x

Tính chính xác

- **Đối với bài toán hồi quy (dự đoán)**
- Giá trị (kết quả) đầu ra của hệ thống là một giá trị số

$$Error = \frac{1}{|D\_test|} \sum_{x \in D\_test} Error(x);$$

$$Error(x) = |d(x) - o(x)|$$

- Trong đó
  - o(x): giá trị đầu ra (dự đoán) bởi hệ thống đối với dữ liệu x
  - d(x): giá trị đầu ra thực sự (đúng) của dữ liệu x
  - Accuracy là một hàm đảo (inverse function) đối với Error

Ma trận nhầm lẫn (Confusion Matrix)

- Nhược điểm của Accuracy là chỉ cho ta biết độ chính xác khi dự báo của mô hình, nhưng không thể hiện mô hình đang dự đoán sai như thế nào, vì vậy chúng ta cần một phương pháp đánh giá khác – Confusion Matrix

- Còn được gọi là Contingency Table

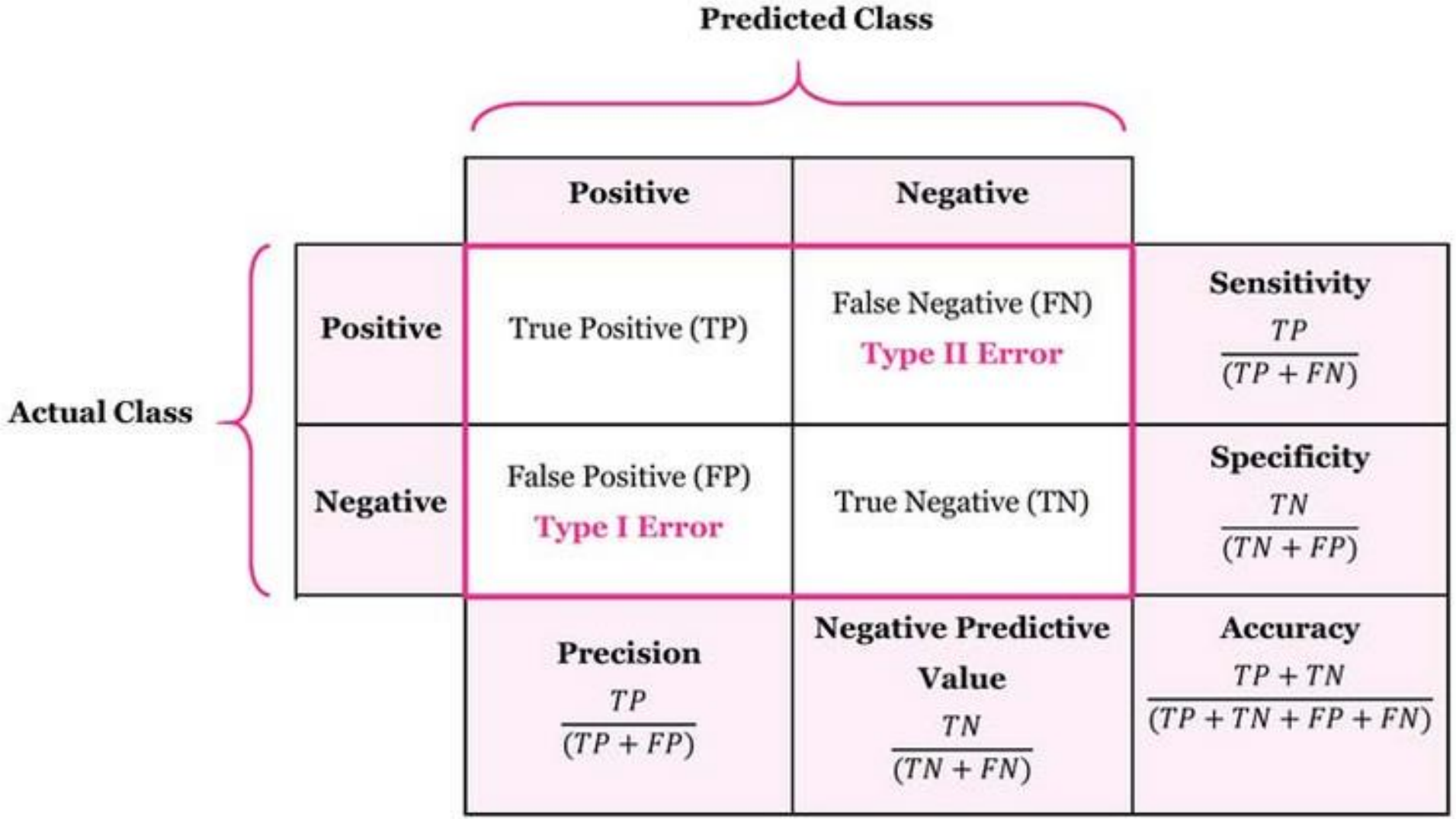
- Chỉ được sử dụng đối với bài toán phân loại (không thể áp dụng cho bài toán hồi quy - dự đoán)

Ma trận nhầm lẫn (Confusion Matrix)

- Là một ma trận thể hiện số lượng điểm dữ liệu thuộc vào một lớp và được dự đoán thuộc vào lớp

- Cung cấp thêm thông tin về tỉ lệ phân lớp đúng giữa các lớp, hay giúp phát hiện các lớp có tỉ lệ phân lớp nhầm cao nhờ vào các khái niệm True (False) Positive (Negative)

Ma trận nhầm lẫn (Confusion Matrix)



Ma trận nhầm lẫn (Confusion Matrix)

- True Positive (TP): đối tượng ở lớp Positive, mô hình phân đối tượng vào lớp Positive (dự đoán đúng)

- True Negative (TN): đối tượng ở lớp Negative, mô hình phân đối tượng vào lớp Negative (dự đoán đúng)

- False Positive (FP): đối tượng ở lớp Negative, mô hình phân đối tượng vào lớp Positive (dự đoán sai) – Type I Error

- False Negative (FN): đối tượng ở lớp Positive, mô hình phân đối tượng vào lớp Negative (dự đoán sai) – Type II Error

Ma trận nhầm lẫn (Confusion Matrix)

- Ví dụ: cần dự đoán kết quả xét nghiệm COVID-19 của 1000 người nghi nhiễm. Dưới đây là kết quả dự đoán của mô hình:
  - Mô hình dự đoán có 30 ca dương tính, trong khi thực tế có 13 người nhiễm COVID-19
  - Mô hình dự đoán có 970 ca âm tính, nhưng thực tế trong 970 ca đó có 20 ca dương tính

Ma trận nhầm lẫn (Confusion Matrix)

- Biểu diễn kết quả dự đoán của mô hình bằng confusion matrix như sau

Predict class

	Positive	Negative
Positive	TP = 13	FN = 20
Negative	FP = 17	TN = 950

Actual class

Ma trận nhầm lẫn (Confusion Matrix)

- Biểu diễn kết quả dự đoán của mô hình bằng confusion matrix như sau
- True Positive TP = 13: có 13 người nhiễm COVID-19 được mô hình dự đoán đúng
- False Positive FP = 17: có 17 người âm tính với COVID-19, nhưng được mô hình dự đoán dương tính
- True Negative TN = 950: 950 trường hợp âm tính được mô hình phân loại chính xác
- False Negative FN = 20: có 20 trường hợp dương tính với COVID-19 nhưng bị mô hình phân loại sai

Ma trận nhầm lẫn (Confusion Matrix)

- Vào ngày 16/04/2021 tại bệnh viện A có 100 bệnh nhân đến khám một loại bệnh, giả sử biết trước trong 100 bệnh nhân có 60 người mắc bệnh, 40 người không có bệnh. Sau khi thăm khám, bệnh viện đưa ra kết quả:
  - Trong 60 người bệnh thật thì có 45 người chuẩn đoán có bệnh, 15 người chuẩn đoán không mắc bệnh.
  - Trong 40 người không mắc bệnh thì có 30 người chuẩn đoán không mắc bệnh, 10 người chuẩn đoán là mắc bệnh.

Ma trận nhầm lẫn (Confusion Matrix)

- Biểu diễn kết quả dự đoán của mô hình bằng confusion matrix như sau

Predict class

	Positive	Negative
Positive	TP = 45	FN = 15
Negative	FP = 10	TN = 30

Actual class

Ma trận nhầm lẫn (Confusion Matrix)

- Bài tập: dự đoán kết quả xét nghiệm của 1005 bệnh nhân xem họ có bị ung thư hay không. Dưới đây là những gì mô hình dự đoán:
- 90 bệnh nhân bị ung thư và tất cả dự đoán đều đúng.
- 915 bệnh nhân không bị ung thư nhưng thật ra có tới 910 người bị ung thư trong thực tế.

Ma trận nhầm lẫn (Confusion Matrix)

- Biểu diễn kết quả dự đoán của mô hình bằng confusion matrix như sau

Predict class

	Positive	Negative
Positive	TP = 90	FN = 910
Negative	FP = 0	TN = 5

Actual class

Precision - Recall

- Với những thông tin có được từ Confusion matrix → có thể định lượng độ hiệu quả của mô hình qua nhiều thang đo khác nhau. Precision và Recall là hai thang đo quan trọng trong số đó.

- **Precision** trả lời cho câu hỏi: trong số các điểm dữ liệu được mô hình phân loại vào lớp Positive, có bao nhiêu điểm dữ liệu thực sự thuộc về lớp Positive.

- **Recall** giúp chúng ta biết được có bao nhiêu điểm dữ liệu thực sự ở lớp Positive được mô hình phân lớp đúng trong mọi điểm dữ liệu thực sự ở lớp Positive.

Precision - Recall

- Precision và Recall **có giá trị trong [0,1**], hai giá trị này càng gần với 1 thì mô hình càng chính xác.

- Precision **càng cao** đồng nghĩa với các điểm được phân loại **càng chính xá**c.

- Recall **càng ca**o cho thể hiện cho việc **ít bỏ sót các điểm dữ liệu đúng**.

$$precision = \frac{TP}{TP + FP} \qquad recall = \frac{TP}{TP + FN}$$

Precision - Recall

- Ví dụ: cần dự đoán kết quả xét nghiệm COVID-19 của 1000 người nghi nhiễm

Predict class

	Positive	Negative
Positive	TP = 13	FN = 20
Negative	FP = 17	TN = 950

Actual class

Confusion Matrix

Precision - Recall

- Ví dụ: cần dự đoán kết quả xét nghiệm COVID-19 của 1000 người nghi nhiễm

	Positive	Negative
Positive	TP = 13	FN = 20
Negative	FP = 17	TN = 950

$$precision = \frac{TP}{TP + FP} = \frac{13}{13 + 17} = 0.43$$

$$recall = \frac{TP}{TP + FN} = \frac{13}{13 + 20} = 0.39$$

Precision - Recall

$$precision = \frac{TP}{TP + FP} = \frac{13}{13 + 17} = 0.43$$

$$recall = \frac{TP}{TP + FN} = \frac{13}{13 + 20} = 0.39$$

- Precision và Recall của mô hình này còn thấp → độ chính xác của mô hình chưa cao

- Trong số 30 trường hợp được chẩn đoán dương tính, chỉ có 13 trường hợp thực sự nhiễm COVID-19 (khoảng 43%, được thể hiện qua precision)

- Với giá trị recall = 0.39, ta có thể hiểu rằng trong số 33 trường hợp thật sự dương tính với COVID-19, mô hình chỉ phát hiện ra

Precision - Recall

- Bài tập: Tính Precision - Recall

Predict class

	Positive	Negative
Positive	TP = 45	FN = 15
Negative	FP = 10	TN = 30

Actual class

Precision - Recall

- Bài tập: Tính Precision - Recall

Predict class

	Positive	Negative
Positive	TP = 90	FN = 910
Negative	FP = 0	TN = 5

Actual class

F1-Score

- Một mô hình tốt khi cả Precision và Recall đều cao, thể hiện cho mô hình ít phân loại nhầm giữa các lớp cũng như tỉ lệ bỏ sót các đối tượng thuộc lớp cần quan tâm là thấp

- Tuy nhiên, hai giá trị Precision và Recall thường không cân bằng với nhau (giá trị này tăng thì giá trị kia thường có xu hướng giảm)

- Để đánh giá cùng lúc cả Precision và Recall, ta sử dụng độ đo F-Score

F1-Score

$$F_\beta = (1 + \beta^2) \frac{precision.recall}{\beta^2.precision + recall}$$

Tham số β quyết định mức độ coi trọng giữa Precision và Recall

- β > 1: Recall được coi trọng hơn Precision
- β < 1: Precision được coi trọng hơn Recall
- β = 1: Precision và Recall được coi trọng ngang nhau

F1-Score

- Việc quyết định nên ưu tiên Precision hay Recall phụ thuộc vào từng bài toán.

- Ví dụ, với bài toán xác định một khu vực có bom mìn hay không, việc bỏ sót bom mìn cho hậu quả nghiêm trọng hơn so với khi báo động một khu vực an toàn là có bom, vì vậy cần ưu tiên Recall hơn Precision.

- Ví dụ, việc bỏ sót spam mail có vẻ không tệ nếu so sánh với khi phân loại nhầm một email quan trọng thành spam mail, do đó ở bài toán này, Precision nên được cân nhắc ưu tiên.

F1-Score

$$F_1 = \frac{2}{\frac{1}{Recall} + \frac{1}{Precision}} = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

- Với những bài toán mà Precision và Recall được cân nhắc ngang nhau, ta chọn β = 1, khi đó ta đang sử dụng F1-Score. F1-Score là kỳ vọng harmonic (harmonic mean) của Precision và Recall.

- F1-score lớn khi cả 2 giá trị Precision và Recall đều lớn. Ngược lại, chỉ cần 1 giá trị nhỏ sẽ làm cho F1-Score nhỏ.

ROC curve

- Trong các bài toán phân lớp, các thuật toán phân lớp thường dự đoán điểm số hay xác suất thuộc về một lớp của dữ liệu đầu vào. Điều này giúp ta biết được mức độ chắc chắn của mô hình khi phân lớp.

- Sau khi dự đoán xác suất hay điểm số, cần phải chuyển các giá trị đó về các nhãn của các lớp. Việc chuyển từ xác suất, điểm số sang các nhãn được quyết định bởi một "ngưỡng" (threshold).

- **ROC curve** là một công cụ để chọn ra threshold phù hợp cho mô hình.

ROC curve

- Với mỗi giá trị threshold, ta thu được hai giá trị được biểu diễn trên ROC curve:

- True Positive Rate (hay Sensitivity – Recall): là độ nhạy của mô hình, cho biết mức độ dự đoán chính xác trong lớp positive. TPR là thương của số điểm dữ liệu được dự đoán đúng thuộc lớp positive với số điểm dữ liệu thuộc lớp positive.

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

ROC curve

➢ Với mỗi giá trị threshold, ta thu được hai giá trị được biểu diễn trên ROC curve:

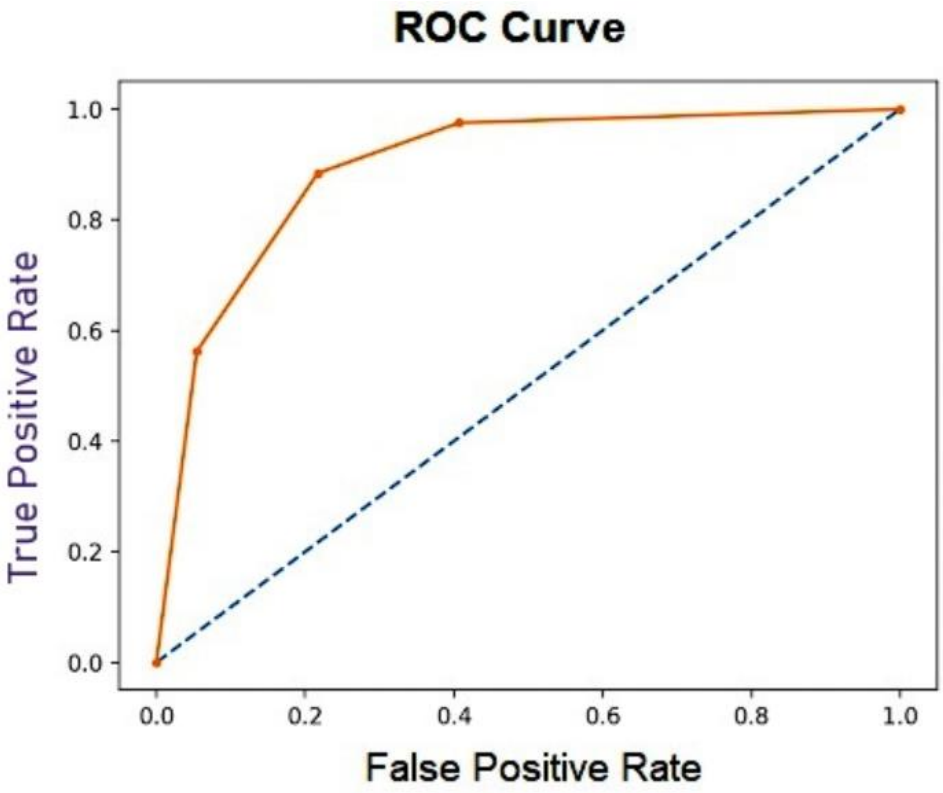
- False Positive Rate: là xác suất mắc Type II Error

$$FPR = 1 - Specificity$$

➔ Trong đó, Specificity cho biết mức độ dự đoán chính xác trong lớp negative.

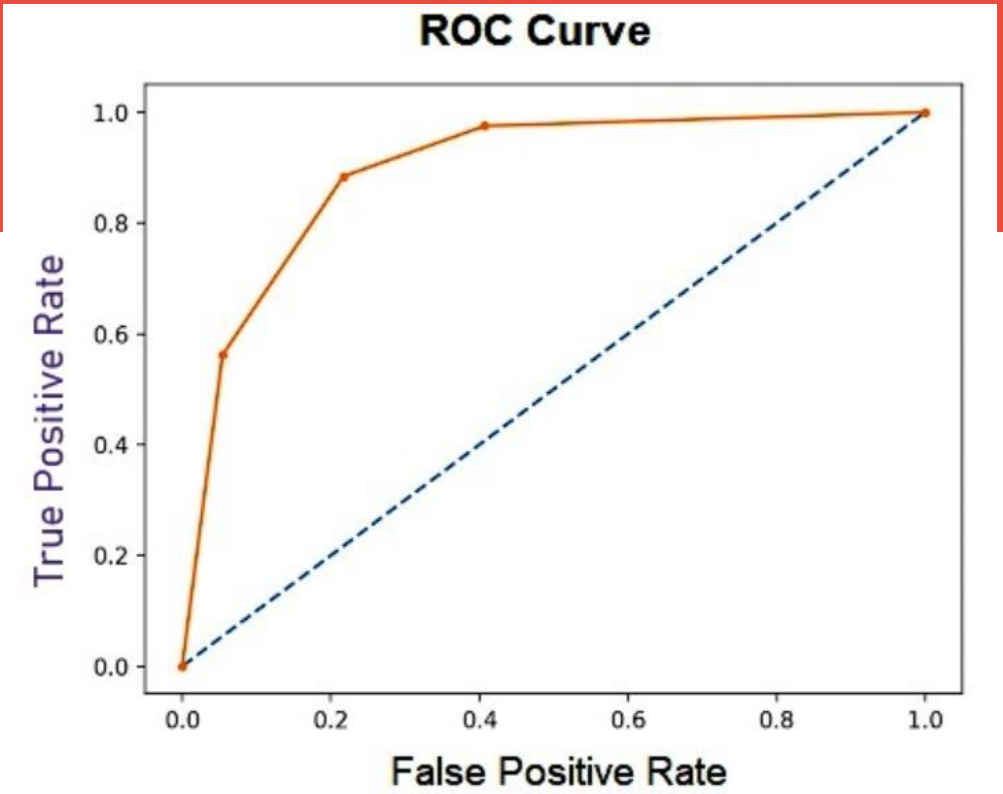
$$Specificity = \frac{TN}{TN + FP}$$

ROC curve



- True Positive Rate (hay Sensitivity – Rec
- False Positive Rate: là xác suất mắc Typ
- Các điểm màu cam đại diện cho mỗi thre
- Nối các điểm màu cam lại với nhau ta đu
- Đường đứt đoạn màu xanh đại diện cho

ROC curve



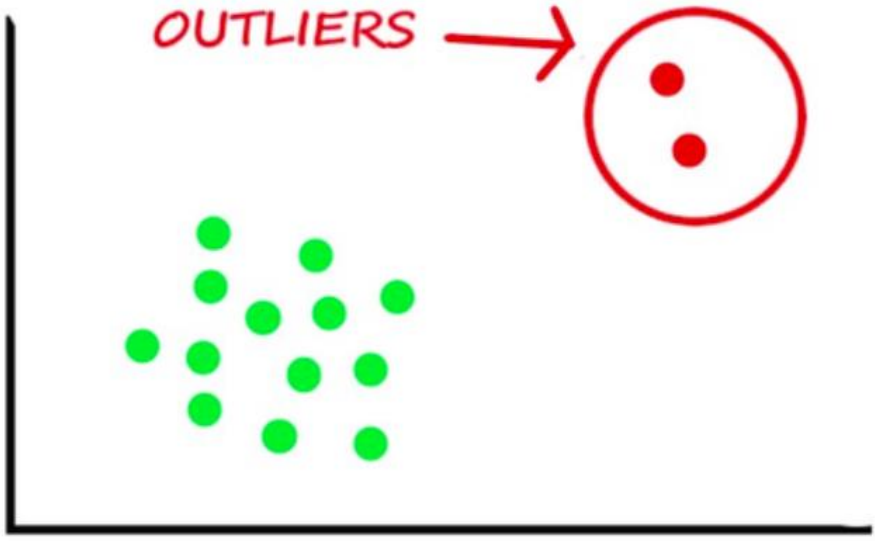
- Giá trị FPR càng thấp thì xác suất mắc Type II Error thấp → các điểm ở bên tr
- Mặt khác, các điểm nằm càng cao thì có TPR càng lớn, đồng nghĩa với việc mô
- Tùy thuộc vào bài toán để lựa chọn một điểm – ứng với một threshold phù hợp

Area Under the ROC curve (AUC)

- AUC là phần diện tích bên dưới ROC curve, dùng để đánh giá hiệu năng phân
- Mô hình có AUC càng lớn (ROC curve càng gần góc cao bên trái) thì cho kết q
- Ngược lại, mô hình có ROC curve càng tiến tới đường chéo 45 độ (đường gạc
- AUC càng cao thì mô hình càng dễ phân loại đúng cho cả lớp positive và negat

Underfitting và Overfitting

- **Outlier**: Điểm ngoại lai/dị biệt/nhiễu (outlier/noise) là những điểm dữ liệu
- Nhiễu trong dữ liệu huấn luyện khiến cl



Underfitting và Overfitting

- Nhiễu có thể là kết quả của những sai sót trong quá trình thu thập dữ liệu. Có t
- **Global Outlie**r: là điểm dữ liệu **khác hoàn toàn** với toàn bộ các điểm dữ liệu k
  - Ví dụ: Một vài điểm dữ liệu có giá trị thuộc tính tuổi người lớn hơn 200 thì nhữ
- **Contextual Outlier**: là điểm dữ liệu **khác biệt khi so với một phần dữ liệu** tro
  - Ví dụ: Doanh số bán hàng của một thương hiệu trong các đợt khuyến mãi có th

Underfitting và Overfitting

- Bias: là sai số giữa giá trị dự đoán trung bình của mô hình và giá trị thực tế.
- High bias: sai số lớn, mô hình đơn giản, tuy nhiên kết quả dự đoán chính xác kh
- Low bias: sai số nhỏ, mô hình phức tạp, kết quả dự đoán tốt

Underfitting và Overfitting

- Variance: là sai số thể hiện mức độ “nhạy cảm” của mô hình với những biến độ
- Low-variance: mô hình ít biến thiên theo sự thay đổi của dữ liệu huấn luyện
- High-variance: mô hình biến thiên mạnh, bám sát theo sự thay đổi của dữ liệu h
- Mô hình có variance cao thường thể hiện rất tốt trên tập dữ liệu huấn luyện, nh

Underfitting và Overfitting

- **Underfitting**: là hiện tượng mà mô hình có **high bias và low varriance**, cho kết quả dự đoán không tốt trên cả tập huấn luyện và tập kiểm thử.
  - Underfitting thường dễ được phát hiện vì cho kết quả tệ trên tập huấn luyện.

- **Overfitting**: là hiện tượng mà mô hình có **low bias và high variance**, lúc này mô hình trở nên phức tạp, bám sát theo dữ liệu huấn luyện.
  - Mô hình cho kết quả rất tốt trên dữ liệu đã được học, nhưng cho kết quả tệ trên dữ liệu chưa từng gặp bao giờ.
  - Vấn đề này xảy ra khi mô hình cố gắng fit tất cả các điểm dữ liệu huấn luyện, bao gồm cả nhiễu.

Nhận biết Underfitting và Overfitting

- Underfitting xảy ra khi sai số dự đoán của mô hình trên cả tập huấn luyện và tập kiểm thử đều cao.

- Overfitting xảy ra khi sai số dự đoán của mô hình trên tập huấn luyện thấp, nhưng trên tập kiểm thử thì cao

- Cross-validation là một kỹ thuật tốt để ngăn ngừa Overfitting