

# **Vấn đề chính của học máy**

**The Pillars of Machine Learning:**

**Dr. Tran Van Lang, A.Prof. in Computer Science**

# Giới thiệu

- Có 4 vấn đề mang tính trụ cột (pillar) của học máy:
  - Hồi quy (Regression)
  - Rút gọn chiều (Dimensionality Reduction)
  - Ước lượng mật độ (Density Estimation)
  - Phân nhóm: phân lớp (Classification), gom cụm (Clustering)
- Có 3 vấn đề quan tâm chính của học máy, đó là dữ liệu (data), mô hình (model), và học (learning).

# Data

- Đơn giản để cho máy tính có thể thực hiện được thì Data như là Vector với các thành phần là con số.
- Chẳng hạn, với bảng dữ liệu như sau:

Tên	Giới tính	Học vị	Plus Code	Tuổi	Lương
Đinh Tiên Hùng	Nam	Tiến sĩ	84W6+H2R	38	25000000
Lê Đại Hưng	Nam	Thạc sĩ	849V+22W	28	18000000
Lý Thái Hạnh	Nữ	Thạc sĩ	938R+P5	36	20000000
Trần Hưng Quốc	Nam	Tiến sĩ	PMCW+HG	65	30000000
Nguyễn Gia Hoa	Nữ	Tiến sĩ	FPQW+GQ	42	25000000

- Quy ước: Giới tính Nam là 0, Nữ là 1; Học vị từ cao xuống thấp với Tiến sĩ; và Plus Code có thể gồm 2 thành phần là kinh tuyến và vĩ tuyến.
- Từ đó có bảng

Tên	Giới tính	Học vị	Vĩ độ	Kinh độ	Tuổi	Lương
Đinh Tiên Hùng	0	6	21.01	105.84	38	25000000
Lê Đại Hưng	0	5	23.15	160.36	28	18000000
Lý Thái Hạng	1	5	51.81	5.82	36	20000000
Trần Hưng Quốc	0	6	10.72	106.69	65	30000000
Nguyễn Gia Hoa	1	6	53.49	-2.53	42	25000000

- Trong ví dụ trên, cột Tên mang tính định danh, không có giá trị khi huấn luyện nên có thể chọn bộ Data gồm  $N = 5$  dòng và  $K = 6$  cột như sau:

Giới tính	Học vị	Vĩ độ	Kinh độ	Tuổi	Lương
0	6	21.01	105.84	38	25000000
0	5	23.15	160.36	28	18000000
1	5	51.81	5.82	36	20000000
0	6	10.72	106.69	65	30000000
1	6	53.49	-2.53	42	25000000



- Một cách tổng quát, gọi số mẫu (*example*) là  $N$ . Khi đó trong bộ dữ liệu (*Dataset*) là một mảng bao gồm  $N$  vector  $x_n, n = 1, \dots, N$ . Mỗi vector này được gọi là một ***data point*** trong Học máy (ML - Machine Learning).
- Mỗi cột trong bộ dữ liệu này được gọi là một đặc trưng (*feature*) của *data point*. Khi đó một vector có số thành phần là  $K$  tương ứng với  $K$  đặc trưng.
- Chẳng hạn, với bảng dữ liệu như trên, coi nhãn (*label*) là lương tháng, ký hiệu là  $y_n, n = 1, \dots, N$  và ta muốn tạo ra một Dataset để học có giám sát (*Supervised Learning*) từ các đặc trưng là *Giới tính*, *Học vị*, *Tuổi* để nhận biết *Lương*; khi đó số mẫu là  $N = 5$  và số đặc trưng  $K = 3$ .
- Bộ dữ liệu là  $(X, y) \in \mathbb{R}^{N \times K} \times \mathbb{R}$ , trong đó  $X$  là ma trận  $N$  dòng và  $K$  cột,  $y$  là vector  $N$  phần tử.

# Models

- Có thể coi *Model* như là một Hàm (*Function*).
  - Gọi  $x$  là một vector có các thành phần là các đặc trưng (*feature*), hãy tìm một hàm  $f$  để làm sao biết được nhãn (*label*) của các đặc trưng này dạng  $f: \mathbb{R}^K \rightarrow \mathbb{R}$
  - Hàm  $f$  được gọi là hàm dự đoán (*predictive function*), trong Học máy hàm này được biết như là **yếu tố dự đoán** (*predictor*) có thể biểu diễn dạng  $f(x) = w^T x + w_0$  với các ẩn số là  $w = (w_1, w_2, \dots, w_K)$  và  $w_0$ .
- Ngoài ra, thay vì coi một *predictor* là một hàm duy nhất, chúng ta có thể coi các **yếu tố dự đoán** là các mô hình xác suất, tức là các mô hình mô tả phân phối của các hàm có thể có (*model as probability distributions*).

# Learning

- Thực chất của việc học (*learning*) là tìm một mô hình và các tham số (*parameter*) tương ứng sao cho predictor kết quả sẽ hoạt động tốt trên dữ liệu chưa có (*unseen data*).
- Trong việc học này, có ba công việc cần phải quan tâm
  - Dự đoán (khi model là function) hoặc suy luận (khi có mô hình là các phân phối xác suất)
  - Huấn luyện hoặc ước lượng tham số
  - Điều chỉnh siêu tham số (*hyperparameter*) hoặc lựa chọn mô hình



# **Độ đo trong bài toán phân lớp**

**Metrics in Classification Problem**

**Dr. Tran Van Lang, A.Prof. in Computer Science**

# Minh hoạ

- Cần phân lớp về ung thư vú là lành tính hay ác tính.
- Sử dụng dữ liệu tại <https://www.kaggle.com/datasets/sahilnbajaj/cancer-classification>
- Bộ dữ liệu gồm 30 thuộc tính, và nhãn gồm 2 lớp: ác tính (1 - malignant) và lành tính (0 - benign) lưu ở cột cuối cùng của tập tin cancer\_classification.csv
- Sử dụng Python để giải quyết bài toán dự đoán với các bước:
  - Load dữ liệu vào biến trong bộ nhớ
  - Chia dữ liệu thành tập huấn luyện (sử dụng để huấn luyện mô hình) và tập kiểm tra (sử dụng để đánh giá hiệu suất mô hình)
  - Tạo mô hình bằng cách sử dụng một thuật toán phân lớp nào để huấn luyện

# Bảng Python với các thư viện cần thiết

## ### Chuẩn bị dữ liệu

```
import polars as pl
file = '/Users/lang/Documents/Works/Training/Mathematics for Machine Learning/cancer_classification.csv'
df = pl.read_csv( file )
```

```
import numpy as np
X = np.array(df.drop('benign_0__mal_1'))
y = np.array(df.select('benign_0__mal_1')).ravel()
```

## ### Các bước tạo mô hình dự đoán

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
from sklearn.neighbors import KNeighborsClassifier
model = KNeighborsClassifier(n_neighbors=5).fit(X_train, y_train)
```

## ### Sử dụng để dự đoán

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

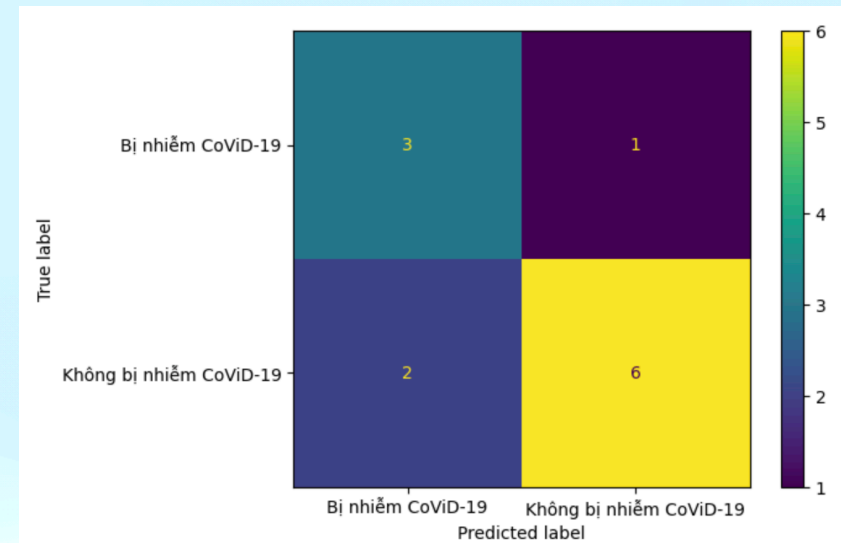
# Khái niệm Positive và Negative

- Trong việc phân lớp cần xác định trước về mặt ý nghĩa thực tế đâu là lớp Positive (dương tính) và đâu là Negative (âm tính).
  - Giá trị của Positive phụ thuộc vào cách mà định nghĩa lớp trong bài toán phân loại: lớp Positive là lớp quan tâm hoặc muốn dự đoán chính xác hơn, còn Negative là lớp còn lại.
- Chẳng hạn, chẩn đoán bệnh thì Positive là lớp bị bệnh, còn Negative là lớp không bị bệnh; nhưng khi khám bệnh để tuyển người làm việc thì Positive là lớp không bị bệnh, còn Negative là lớp bị bệnh.

- Từ đó có các khái niệm:
  - **True Positive (TP)**: Mẫu thuộc lớp Positive được dự đoán chính xác là Positive
  - **False Positive (FP)**: Mẫu thuộc lớp Negative được dự đoán thành Positive (dự đoán sai hay nhầm là Positive)
  - **True Negative (TN)**: Mẫu thuộc lớp Negative được dự đoán chính xác là Negative
  - **False Negative (FN)**: Mẫu thuộc lớp Positive nhưng dự đoán thành Negative (dự đoán sai hay nhầm là Negative)



- Giả sử có 12 mẫu xét nghiệm CoViD-19, trong đó có 8 mẫu không bị nhiễm (Negative), còn 4 mẫu bị nhiễm (Positive)
- Nhưng kết quả dự đoán (predict) của một mô hình học máy là trong số 8 mẫu Negative chỉ dự báo đúng 6 mẫu, và trong số 4 mẫu Positive dự đoán đúng 3 mẫu.
- Ma trận phân loại (Confusion Matrix) biểu diễn bằng hình ảnh bên cạnh.



```
from sklearn.metrics import *
```

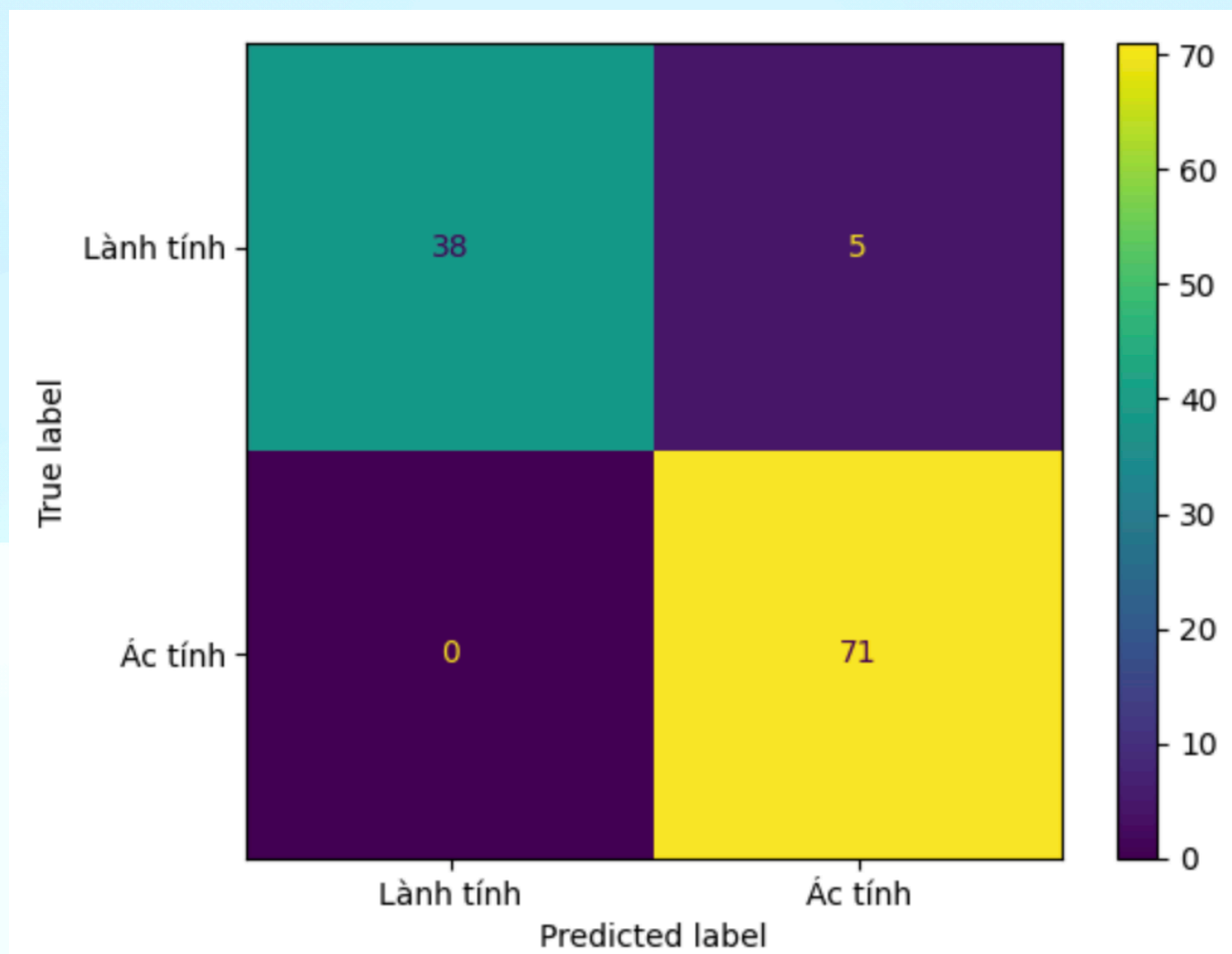
```
y_true = [1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0]
y_pred = [0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0]
```

```
cm = confusion_matrix(y_true, y_pred)
ConfusionMatrixDisplay( confusion_matrix=cm, display_
labels=["Bị nhiễm CoViD-19", "Không bị nhiễm
CoViD-19"] ).plot()
```

- Hoặc với dữ liệu ung thư vú như phần trên.
- Bộ dữ liệu có 569 dữ liệu (data point), dành 20% cho thử nghiệm (test data): 114 dữ liệu
- Trong bộ dữ liệu này, giả sử việc tạo train data và test data luôn luôn giống nhau ở các lần thực thi chương trình khác nhau.
- Khi đó, trong số 114 dữ liệu dùng để thử nghiệm này có 71 trường hợp ác tính (giá trị 1) và 43 trường hợp lành tính (giá trị 0)

```
from sklearn.metrics import *
```

```
cm = confusion_matrix(y_test, y_pred)  
ConfusionMatrixDisplay( confusion_matrix=cm,display_labels=["Lành tính","Ác tính"] ).plot()
```



# Recall hay Sensitivity (Độ nhạy)

- Còn gọi là tỷ số thu hồi, hay tỷ số TP (**True Position Rate - TPR**)
- Nhằm xác định trong tất cả các trường hợp thực tế (True case) là Positive, thì có bao nhiêu dự đoán Positive là đúng.
- Đây là một độ đo quan trọng khi chúng ta muốn đảm bảo không bỏ sót các mẫu của lớp thiểu số, nên còn được gọi là tỷ lệ phát hiện.
- Công thức tính:  $Recall = \frac{TP}{TP + FN}$
- Ý nghĩa: tỷ số phần trăm mẫu thuộc lớp Positive được dự đoán chính xác.

# Accuracy

- Là độ chính xác tổng thể
- Nhằm xác định trong tổng số cần dự đoán (cả Positive và Negative), thì có bao nhiêu phần là dự đoán đúng (là  $TP + TN$ ).
- Công thức tính:  $Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$
- Ngoài ra còn có **Balanced Accuracy (BA)** - **độ chính xác cân bằng** là một chỉ số được sử dụng để đánh giá hiệu suất của mô hình phân lớp trong trường hợp dữ liệu có sự mất cân bằng giữa các lớp.
- BA là trung bình cộng của sensitivity và specificity.



- Ý nghĩa: Tỷ số phần trăm đo độ phù hợp khi dữ liệu mất cân bằng và một độ đo chính xác tổng quát hơn so với accuracy truyền thống vì phối hợp cả tỷ số Positive thật - TPR (hay độ nhạy sensitivity) và tỷ số Negative thật - TNR (hay độ đặc hiệu specificity).

- Công thức tính: 
$$BA = \frac{\frac{TP}{TP + FN} + \frac{TN}{TN + FP}}{2}$$

- Trong Python có phương thức `sklearn.metrics.balanced_accuracy_score()`

# Precision hay Positive Predictive Value

- Còn gọi là độ chính xác, hay trị số tiên lượng (PPV - Positive Predictive Value)
- Nhằm xác định trong tất cả các dự đoán về Positive (cả true và false), thì có bao nhiêu phần dự đoán Positive là đúng.
- Đây là một độ đo quan trọng khi chúng ta quan tâm đến việc tránh dự đoán sai cho lớp thiểu số.

- Công thức tính:  $Precision = \frac{TP}{TP + FP}$

- Ý nghĩa: Tỷ số phần trăm mẫu được dự đoán là Positive thực sự thuộc lớp Positive

# F1-Score

- Với 2 độ đo Precision và Recall được dùng để khắc phục hạn chế của độ đo Accuracy đó là chỉ tập trung cho việc đánh giá mô hình qua việc phân loại theo lớp Positive.
- Cả hai có tử số giống nhau, chỉ khác về mẫu số.
  - Precision quan tâm đến việc dự đoán về Positive (cả dự đoán đúng Positive và cả dự đoán sai Positive).
  - Trong khi đó Recall quan tâm đến những gì là đúng với thực tế là Positive (cả dự đoán đúng Positive và cả dự đoán sai Negative - nghĩa là đúng Positive).
- Từ đó, có F1-Score là trung bình điều hoà (harmonic mean) của Precision và Recall, nhằm quan tâm nhiều đến độ đo có giá trị nhỏ.

- Công thức tính:  $F_1 - Score = 2 \frac{Precision \times Recall}{Precision + Recall}$
- Ý nghĩa: Là tỷ số phần trăm, giúp cân bằng giữa Precision và Recall
- Lưu ý: Trung bình điều hoà là nghịch đảo của trung bình cộng của nghịch đảo các giá trị.
  - Trung bình điều hoà của  $n$  giá trị

$$x_1, x_2, \dots, x_n \text{ được tính là } \frac{n}{\sum_{i=1}^n \frac{1}{x_i}}$$

$$\text{Nên } F_1 - Score = \frac{2}{\frac{1}{Precision} + \frac{1}{Recall}}$$

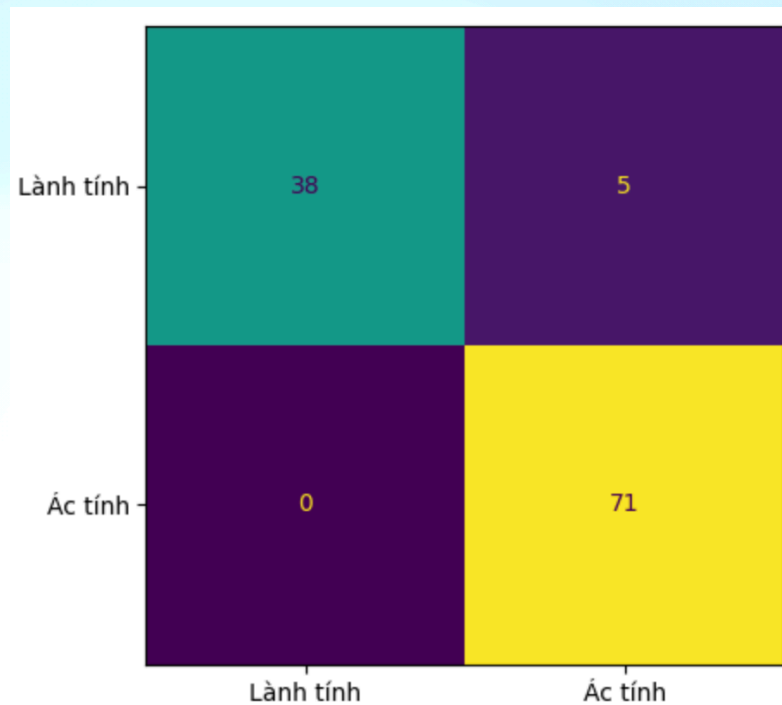
```
from sklearn.metrics import *
```

```
y_true = [1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0]
y_pred = [0, 0, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0]
```

```
print( recall_score(y_true,y_pred) )
print( precision_score(y_true,y_pred) )
print( f1_score(y_true,y_pred) )
print( accuracy_score(y_true,y_pred) )
```

```
from sklearn.metrics import *
```

```
print( "Recall: %4.2f" %recall_score(y_test,y_pred) )  
print( "Precision: %4.2f" %precision_score(y_test,y_pred) )  
print( "F1: %4.2f" %f1_score(y_test,y_pred) )  
print( "Accuracy: %4.2f" %accuracy_score(y_test,y_pred) )  
print( "Balanced Accuracy: %4.2f" %balanced_accuracy_score(y_test,y_pred) )
```



A confusion matrix heatmap with two rows and two columns. The rows are labeled 'Lành tính' and 'Ác tính' on the left. The columns are labeled 'Lành tính' and 'Ác tính' at the bottom. The cells contain the following values: Top-left (Lành tính vs Lành tính) is 38 (teal background); Top-right (Lành tính vs Ác tính) is 5 (dark purple background); Bottom-left (Ác tính vs Lành tính) is 0 (dark purple background); Bottom-right (Ác tính vs Ác tính) is 71 (yellow background).

Lành tính	38	5
Ác tính	0	71
	Lành tính	Ác tính



# Trung bình điều hoà

- Trung bình cộng và trung bình điều hòa là hai phép toán thống kê thường được sử dụng để tính toán giá trị trung tâm của một tập dữ liệu.
- Tuy nhiên, trong một số trường hợp, trung bình điều hòa có thể là lựa chọn phù hợp hơn so với trung bình cộng. Dưới đây là một số lý do:
- **Khi dữ liệu có nhiều giá trị cực đoan:**
  - Trung bình cộng rất nhạy cảm với các giá trị cực đoan (outlier) trong dữ liệu. Nếu có một hoặc nhiều giá trị rất cao hoặc rất thấp trong tập dữ liệu, trung bình cộng sẽ bị ảnh hưởng đáng kể, dẫn đến kết quả không chính xác hoặc sai lệch.
  - Mặt khác, trung bình điều hòa ít bị ảnh hưởng bởi các giá trị cực đoan hơn. Do sử dụng phép nghịch đảo của mỗi giá trị trong tập dữ liệu trước khi tính toán trung bình, giúp giảm thiểu tác động của các giá trị cao hoặc thấp.

- Chẳng hạn, tập dữ liệu gồm các số: 2, 4, 4, 4, 100.
  - Trung bình cộng:  $(2 + 4 + 4 + 4 + 100) / 5 = 22$
  - Trung bình điều hòa:  $5 / (1/2 + 1/4 + 1/4 + 1/4 + 1/100) = 4$
- Như ta thấy, trung bình điều hòa là 4, gần với giá trị trung tâm thực tế của tập dữ liệu hơn (4) so với trung bình cộng (là 22) bị ảnh hưởng bởi giá trị cực đoan 100.
- **Khi ta muốn tính trung bình của các đại lượng có tỷ lệ:**
  - Trung bình điều hòa thường được sử dụng để tính trung bình của các đại lượng có tỷ lệ,
  - Như tốc độ, tỷ lệ, tỷ lệ phần trăm.

- Ví dụ, khi tính trung bình tốc độ của hai xe đi với tốc độ 40 km/h và 60 km/h, ta nên sử dụng trung bình điều hòa thay vì trung bình cộng:
  - Trung bình cộng:  $(40 + 60) / 2 = 50$  km/h
  - Trung bình điều hòa:  $2 / (1/40 + 1/60) = 48$  km/h
- Trung bình điều hòa (48 km/h) gần với tốc độ trung bình thực tế của hai xe hơn so với trung bình cộng (50 km/h).
- **Khi ta muốn tính trung bình của các đại lượng có đơn vị đo khác nhau:**
  - Trung bình cộng không thể được sử dụng để tính trung bình của các đại lượng có đơn vị đo khác nhau. Trung bình điều hòa có thể được sử dụng bằng cách chuyển đổi các đại lượng sang cùng đơn vị trước khi tính trung bình.

- Giả sử ta có tập dữ liệu gồm giá tiền của ba chiếc xe: 10 triệu đồng, 500 USD, và 800 EUR.
  - Ta có thể chuyển đổi tất cả các giá trị sang đơn vị triệu đồng trước khi tính trung bình điều hòa:
    - ▶ 1 USD  $\approx$  23.000 VND
    - ▶ 1 EUR  $\approx$  26.000 VND
  - Giá tiền trung bình của ba chiếc xe:  $3 / (1/10 + 1/(500 * 23.000) + 1/(800 * 26.000)) \approx 12.4$  triệu đồng

- Lưu ý thêm: Còn có trung bình nhân,
  - Thường được sử dụng khi các giá trị cần tính trung bình là tỷ lệ phần trăm
  - Chẳng hạn trung bình nhân của  $n$  đại lượng  $x_1, x_2, \dots, x_n$  là  $\sqrt[n]{x_1 \times x_2 \times \dots \times x_n}$



# **Độ đo trong bài toán hồi quy**

## **Metrics in Regression Problem**

**Dr. Tran Van Lang, A.Prof. in Computer Science**

# Bài toán hồi quy

- Cho một dãy các cặp riêng biệt  $(X_1, y_1), (X_2, y_2), \dots, (X_N, y_N)$  gọi là bộ dữ liệu huấn luyện (training dataset) gồm có  $N$  quan sát (data point),
  - Trong đó mỗi quan sát chứa  $K$  thành phần  $X_i = (x_1, x_2, \dots, x_K) \in \mathbb{R}^K$  gọi là  $K$  đặc trưng (feature) của bộ dữ liệu
  - Còn các  $y_i, \forall i = \overline{1, N}$  gọi là các biến phụ thuộc, đôi khi đó là nhãn (label) hay giá trị thực (truth value), đó chính là dữ liệu thực tế.
- Vấn đề đặt ra là tìm hàm  $f$  sao cho  $f(X_i)$  xấp xỉ tốt nhất  $y_i, \forall i = \overline{1, N}$ .

- Trong học máy (Machine Learning), giá trị  $f(X_i)$  được ký hiệu là  $\hat{y}_i = f(X_i)$  được gọi là giá trị dự đoán hay giá trị mong đợi (outcome),
- Còn hàm  $f$  được gọi là mô hình huấn luyện (training model)
- Giá trị mong đợi này phải xấp xỉ tốt nhất các giá thực có trong bộ dữ liệu.
- Chính vì vậy, cần có các thước đo để đánh giá mô hình huấn luyện này.
- Có nhiều cách để đánh giá mô hình hồi quy, tùy thuộc vào mục đích nghiên cứu và loại mô hình hồi quy để dùng chỉ số đánh giá tương ứng.
  - Nếu mục đích nghiên cứu là dự đoán giá trị, dùng các độ đo về sai số
  - Còn nếu mục đích nghiên cứu là kiểm định giả thuyết, tính chất của mô hình, dùng các chỉ số kiểm định liên quan đến dữ liệu, tỷ lệ dự đoán đúng.

# Mean Absolute Error

- Mean Absolute Error - MAE (Sai số tuyệt đối trung bình) được dùng để đo lường sai số trung bình của mô hình so với dữ liệu thực tế.

- Công thức tính: 
$$MAE = \frac{1}{N} \sum_{i=1}^N |\hat{y}_i - y_i|$$

- Ưu điểm: MAE tính trung bình cộng giá trị tuyệt đối của sai số giữa giá trị dự đoán và giá trị thực tế. Do đó, nó có thể được hiểu một cách dễ dàng và trực quan bởi người dùng không chuyên về kỹ thuật.

- Hạn chế: Do MAE không bình phương các phần dư, nên không nhạy với các điểm dữ liệu ngoại lai. Điều này có thể dẫn đến việc mô hình không đánh giá chính xác các điểm dữ liệu hiếm gặp hoặc ngoại lai. Hơn nữa, một mô hình có thể có nhiều sai số nhỏ và một sai số lớn, nhưng MAE chỉ tính trung bình giá trị tuyệt đối của chúng. Điều này có thể làm mất đi thông tin quan trọng về sự biến động của sai số.



# Lưu ý

- Khi dùng MAE, ... giá trị có thể lớn hơn 1, trong trường hợp này có thể do một trong những nguyên nhân sau:
  - Dữ liệu có nhiều nhiễu
  - Mô hình hồi quy không phù hợp với dữ liệu
  - Dữ liệu có nhiều giá trị ngoại lai.
- Khi đó,
  - Thử sử dụng các kỹ thuật tiền xử lý dữ liệu như chuẩn hóa dữ liệu, loại bỏ nhiễu, xử lý giá trị ngoại lai, hoặc
  - Hoặc thử các mô hình hồi quy khác nhau để tìm mô hình phù hợp nhất với dữ liệu, hoặc
  - Có thể điều chỉnh các tham số của mô hình hồi quy để tối ưu hóa hiệu suất



# Mean Square Error

- Mean Square Error - MSE (Sai số bình phương trung bình) là một chỉ số thống kê đo mức độ chênh lệch giữa giá trị dự đoán và giá trị thực tế trong một mô hình thống kê.

- Công thức tính: 
$$\text{MSE} = \frac{1}{N} \sum_{i=1}^N (\hat{y}_i - y_i)^2$$

- Ưu điểm: Dễ dàng tính toán và hiểu, có ý nghĩa thống kê rõ ràng; qua đó đo lường sai số bình phương trung bình giữa giá trị dự đoán và giá trị thực tế. Điều này giúp nắm bắt được mức độ chính xác của mô hình. Đặc biệt nhạy cảm với sai số lớn (MSE tăng lên nhanh chóng nếu có sai số lớn, giúp chúng ta phát hiện các dự đoán không chính xác).
- Hạn chế: MSE bị ảnh hưởng bởi các giá trị ngoại lệ (outliers), điều đó cho thấy nếu có nhiều ngoại lệ. Đặc biệt MSE có thể không phản ánh chính xác hiệu suất của mô hình.

# Root Mean Square Error

- Root Mean Square Error - RMSE đo lường sai số trung bình của mô hình so với dữ liệu thực tế. Giá trị RMSE càng nhỏ thì mô hình càng tốt.

- Công thức tính: 
$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^N (\hat{y}_i - y_i)^2}$$

- Ưu điểm: Dễ hiểu, có đơn vị giống với đơn vị của biến tính toán sai số làm cho việc so sánh giữa các mô hình dễ dàng hơn.
- Hạn chế: Nhạy cảm với ngoại lệ thường bị ảnh hưởng bởi các giá trị ngoại lệ. Nếu có nhiều ngoại lệ, RMSE có thể không phản ánh chính xác hiệu suất của mô hình. RMSE không thể phát hiện được sai số lớn hơn, vì nó đã được chuẩn hóa bằng căn bậc hai.

# R-square

- R-square ( $R^2$ ) là chỉ số đo lường mức độ phù hợp giữa giá trị dự đoán và giá trị thực. Giá trị của R-square dao động từ  $\infty$  đến 1, trong đó giá trị càng gần 1 thì mô hình càng tốt.
- Công thức tính:  $R^2 = 1 - \frac{\text{MSE}(\text{giá trị dự đoán, giá trị thực})}{\text{MSE}(\text{giá trị trung bình, giá trị thực})}$
- $$R^2 = 1 - \frac{\sum_{i=1}^N (\hat{y} - y_i)^2}{\sum_{i=1}^N (\bar{y} - y_i)^2}$$
- Giá trị âm của  $R^2$  chứng tỏ mô hình hồi quy còn tệ hơn so với mô hình hồi quy đơn giản (là mô hình lấy giá trị trung bình làm giá trị dự đoán)

- Ưu điểm: Đơn giản và dễ hiểu, thể hiện tỷ lệ phần trăm biến thiên của giá trị dự đoán được giải thích bởi các biến độc lập trong mô hình hồi quy.
  - Có giá trị thông tin, cho biết mô hình có thể giải thích được bao nhiêu phần trăm sự biến thiên của dữ liệu.
  - R-Square có thể được sử dụng để so sánh các mô hình hồi quy khác nhau, giúp lựa chọn mô hình phù hợp nhất để giải thích dữ liệu.
- Hạn chế: Dễ bị ảnh hưởng bởi số lượng biến vì R-Square có xu hướng tăng khi số lượng biến độc lập trong mô hình tăng, bất kể ý nghĩa thống kê của các biến.
  - Không phản ánh độ chính xác của dự đoán vì R-Square chỉ đo lường mức độ phù hợp của mô hình với dữ liệu hiện có, không cho biết mô hình có thể dự đoán chính xác dữ liệu mới hay không.
  - Bị ảnh hưởng bởi giá trị ngoại lai vì giá trị ngoại lai có thể làm tăng hoặc giảm R-Square một cách đáng kể, ảnh hưởng đến độ tin cậy của chỉ số này.



# Thử nghiệm

- Với dataset từ <https://www.kaggle.com/datasets/camnugent/california-housing-prices> về giá nhà ở California
- Dữ liệu chứa thông tin từ cuộc điều tra dân số California năm 1990. Có thể không giúp bạn dự đoán giá nhà.
- Có các thành phần (các đặc trưng):
  - **longitude**: Giá trị cao hơn là xa hơn về phía Tây
  - **latitude**: Giá trị cao hơn là xa hơn về phía Bắc
  - **housing\_median\_age**: con số thấp hơn là một tòa nhà mới hơn



- **total\_rooms**: về số phòng
- **total\_bedrooms**: về số phòng ngủ
- **population: total\_bedrooms**: về người cư trú
- **households**: về số hộ gia đình - một nhóm người cư trú trong một đơn vị gia đình
- **median\_income**: thu nhập trung bình cho các hộ gia đình (*được đo bằng chục nghìn đô la Mỹ*)
- **median\_house\_value**: Giá trị trung bình cho các hộ gia đình trong nhà (*được đo bằng đô la Mỹ*)
- **ocean\_proximity**: Vị trí của ngôi nhà

# Bằng Python với các thư viện cần thiết

## ### Chuẩn bị dữ liệu

```
import polars as pl
file = '/Users/lang/Documents/Works/Training/Mathematics for Machine Learning/housing.csv'
df = pl.read_csv( file )
```

## # Bỏ những dòng thiếu dữ liệu

```
df = df.drop_nulls()
```

```
import numpy as np
```

```
X = np.array(df.drop('median_house_value', 'ocean_proximity'))
```

```
y = np.array(df.select('median_house_value')).ravel()
```

## ### Các bước tạo mô hình hồi quy

```
from sklearn.linear_model import LinearRegression
```

```
model = LinearRegression().fit(X_train, y_train)
```

## ### Sử dụng để dự đoán

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

# Thử một số độ đo để đánh giá mô hình

```
print( "Mean Squared Error: %4.2f" %mean_squared_error(y_test, y_pred) )  
print( "Mean Absolute Error: %4.2f" %mean_absolute_error(y_test, y_pred) )  
print( "Root Mean Squared Error: %4.2f" %root_mean_squared_error(y_test, y_pred) )  
print( "R2-Square: %4.2f" %r2_score(y_test, y_pred) )
```

```
Mean Squared Error: 4921881237.63  
Mean Absolute Error: 51372.67  
Root Mean Squared Error: 70156.12  
R2-Square: 0.64
```