HỌC SÂU

BÀI 1. TỔNG QUAN VỀ HỌC SÂU

1.1 Lịch sử của Học sâu

Khởi nguồn của AI (1940s-1956)

- Những năm 40: Khởi đầu với sự ra đời của máy tính
- Các nhà khoa học bắt đầu mơ ước về máy móc có trí thông minh như con người
- 1956: Thuật ngữ "Artificial Intelligence" chính thức ra đời tại hội thảo Dartmouth (Mỹ)

Định nghĩa học máy:

- Tập trung vào phát triển thuật toán và mô hình thống kê
- Cho phép máy tính thực hiện nhiệm vụ không cần hướng dẫn tường minh
- Khai thác thông tin từ dữ liệu có sẵn
- Có khả năng nhận diện mẫu, đưa ra quyết định và dự đoán với độ chính xác cao
- Bản chất là dạy máy tính học hỏi từ kinh nghiệm, tương tự cách con người học

Bước ngoặt năm 2012:

- Krizhevsky và cộng sự chứng minh mạng neuron nhân tạo LeNet (được thiết kế hơn 20 năm trước) có thể vượt xa các phương pháp nhận dạng hình ảnh truyền thống
- Đây là cột mốc đánh dấu sự phát triển của học sâu (deep learning)

Đặc điểm của học sâu:

- Là lĩnh vực con của học máy thống kê
- Tập trung vào phương pháp học biểu diễn từ dữ liệu
- Bắt nguồn từ mạng neuron nhân tạo
- Có tính mô-đun, linh hoạt và khả năng mở rộng cao
- Phát triển nhanh nhờ GPU và dữ liệu lớn
- Đạt được nhiều tiến bộ vượt trội so với học máy truyền thống

Các cột mốc lịch sử quan trọng:

1943: McCulloch & Pitts - Mô hình MCP, nguồn gốc mạng neuron hiện nay

1949: Donald Hebb - Quy tắc học Hebbian, nền móng mạng neuron

1958: Frank Rosenblatt - Mô hình perceptron đầu tiên

1974: Paul Werbos - Thuật toán lan truyền ngược, quan trọng trong huấn luyện mạng neuron

1980: Kunihiko Fukushima - Neocogitron và mạng neural tích chập

1982: John Hopfield - Hopfield Network, nền tảng phát triển ngành học máy

1985: Hilton & Sejnowski - Boltzmann Machine

1986: Paul Smolensky - Harmonium (Restricted Boltzmann Machine)

Các cột mốc lịch sử quan trọng:

1986: Michael I. Jordan - Mạng neural đệ quy (RNN)

1990: Yann LeCun - LeNet, chứng minh hiệu quả mạng neural sâu

1997: Schuster & Paliwal - Mạng neural đệ quy hai chiều

1997: Hochreiter & Schmidhuber - LSTM, giải quyết vanishing gradient

2006: Geoffrey Hinton - Deep Belief Networks và kỹ thuật tiền huấn luyện layer-wise

2009: Salakhutdinov & Hinton - Deep Boltzmann Machines

2012: Geoffrey Hinton - Kỹ thuật Dropout cho huấn luyện mạng neural

2012+: Cuộc cách mạng deep learning bắt đầu

1.2. MỘT SỐ KHÁI NIỆM CƠ BẢN

Định nghĩa thuật toán học máy (Mitchell, 2017):

"Một chương trình máy tính được cho là học từ kinh nghiệm E đối với một số loại nhiệm vụ T và độ đo hiệu suất P, nếu hiệu suất của nó đối với các nhiệm vụ trong T, được đo bằng P, cải thiện theo kinh nghiệm E."

Học máy giám sát (supervised learning)

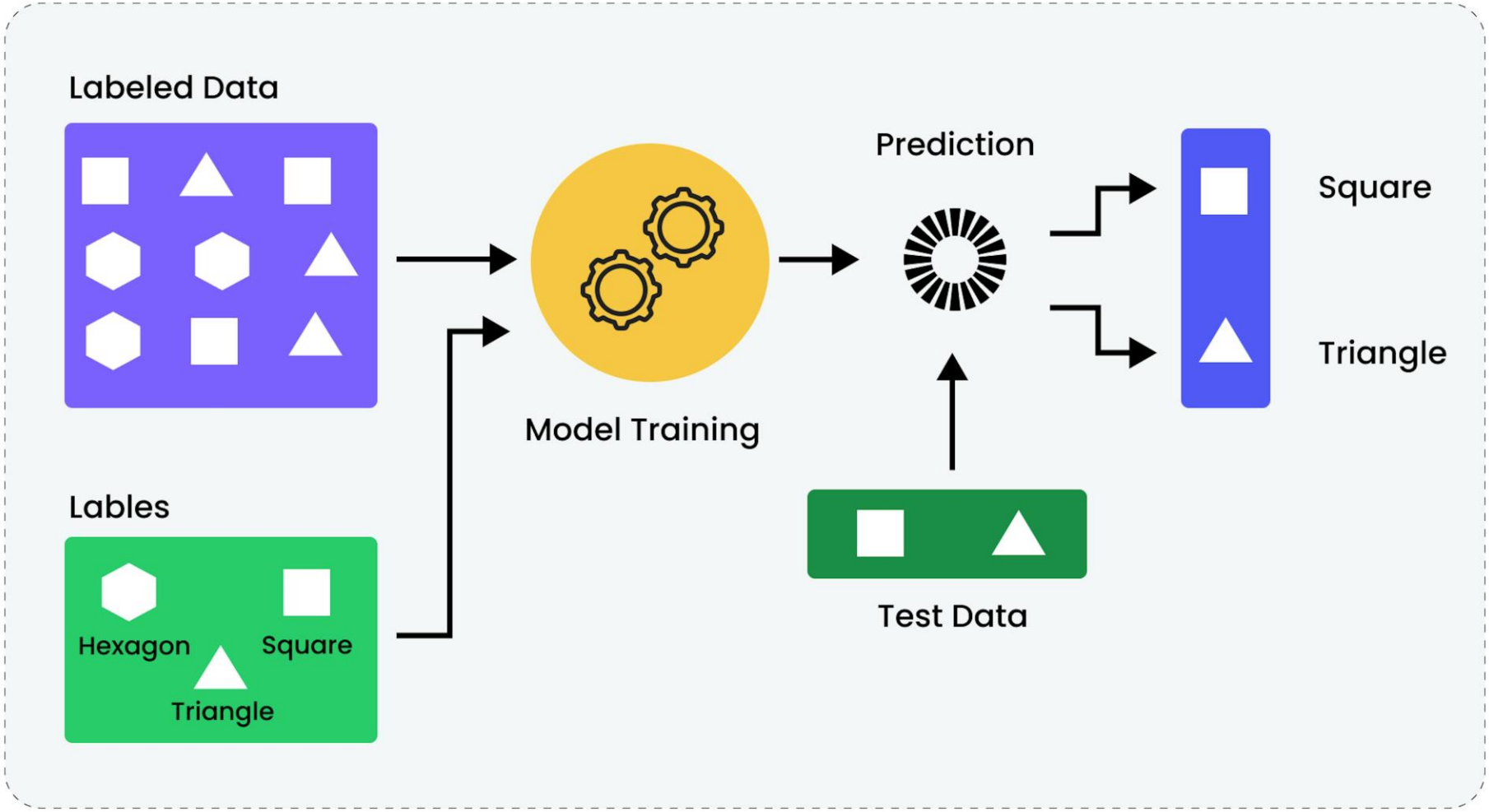
Đối với học có giám sát, bộ dữ liệu huấn luyện là một tập hợp các quan sát. Trong đó, ***mỗi quan sát được gắn với một nhãn (label)*** nào đó. Các thuật toán học có giám sát cố gắng xây dựng mô hình sao cho với một quan sát cho trước, mô hình sẽ trả về kết quả gần nhất với nhãn đã được gắn với quan sát đó.

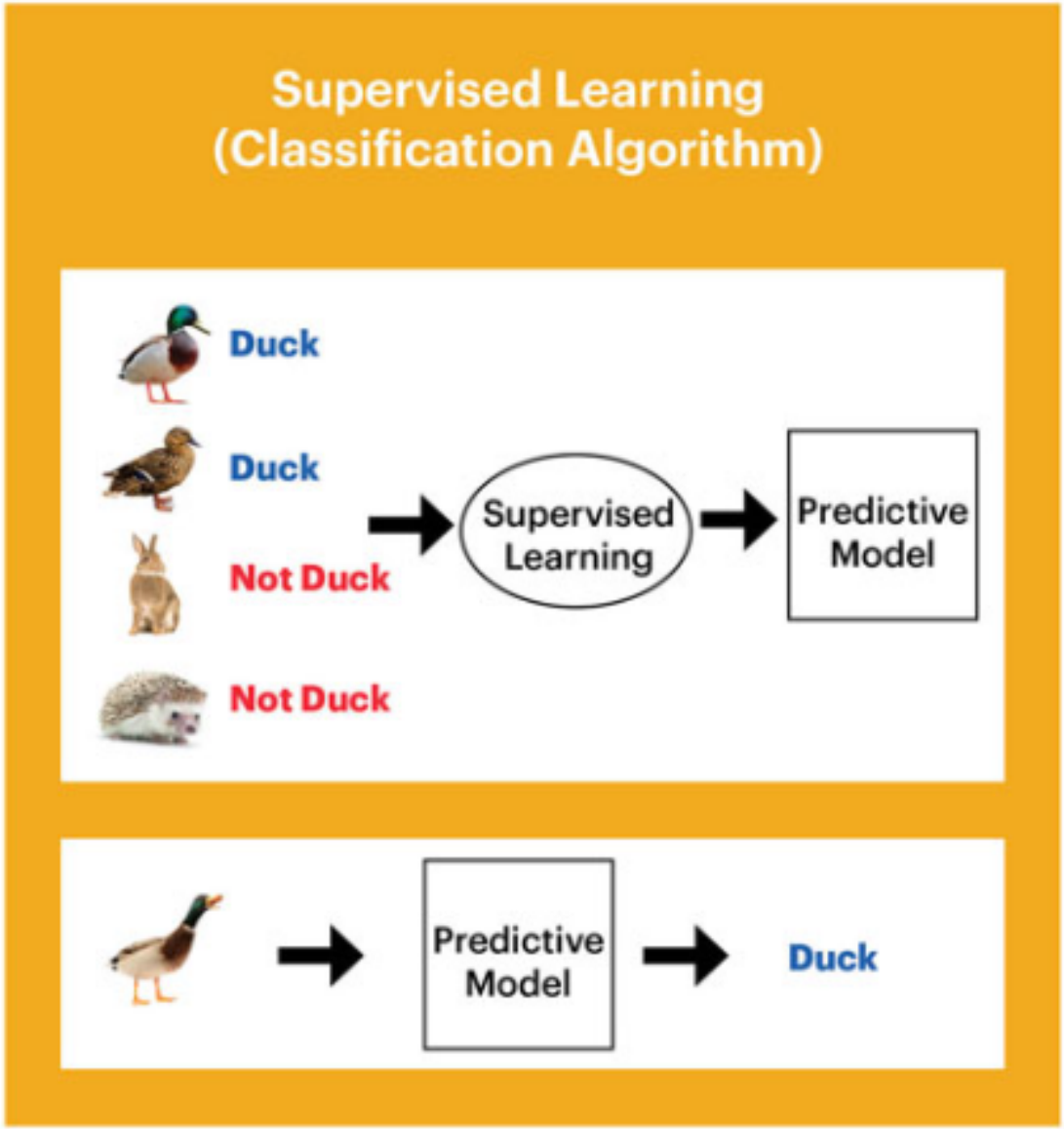
Học có giám sát (supervised learning)

Học tập có giám sát là nơi bạn có các biến đầu vào (X) và biến đầu ra (Y) và bạn sử dụng thuật toán để tìm hiểu hàm ánh xạ từ đầu vào đến đầu ra.

$$Y = f(X)$$

Mục đích là để xây dựng hàm ánh xạ một cách tốt nhất có thể để khi bạn có dữ liệu đầu vào mới (X) và bạn có thể dự đoán các biến đầu ra (Y) cho dữ liệu đó.


Labeled Data
Lables
Hexagon
Square
Triangle
Model Training
Prediction
Test Data
Square
Triangle


Supervised Learning
(Classification Algorithm)
Duck
Duck
Not Duck
Not Duck
Supervised Learning
Predictive Model
Predictive Model
Duck

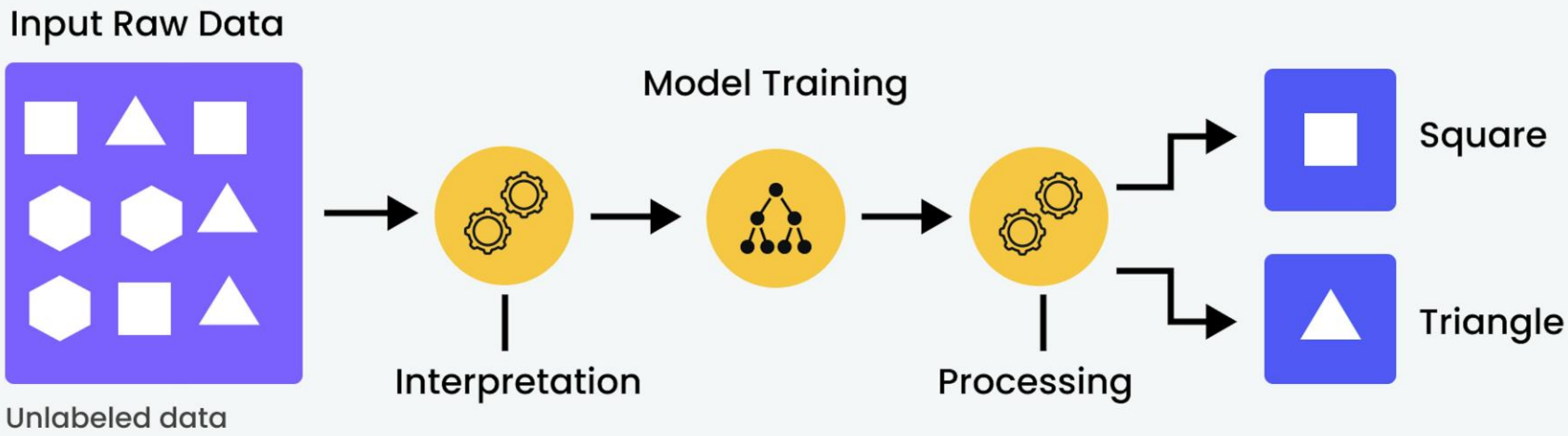
Học không giám sát (unsupervised learning)

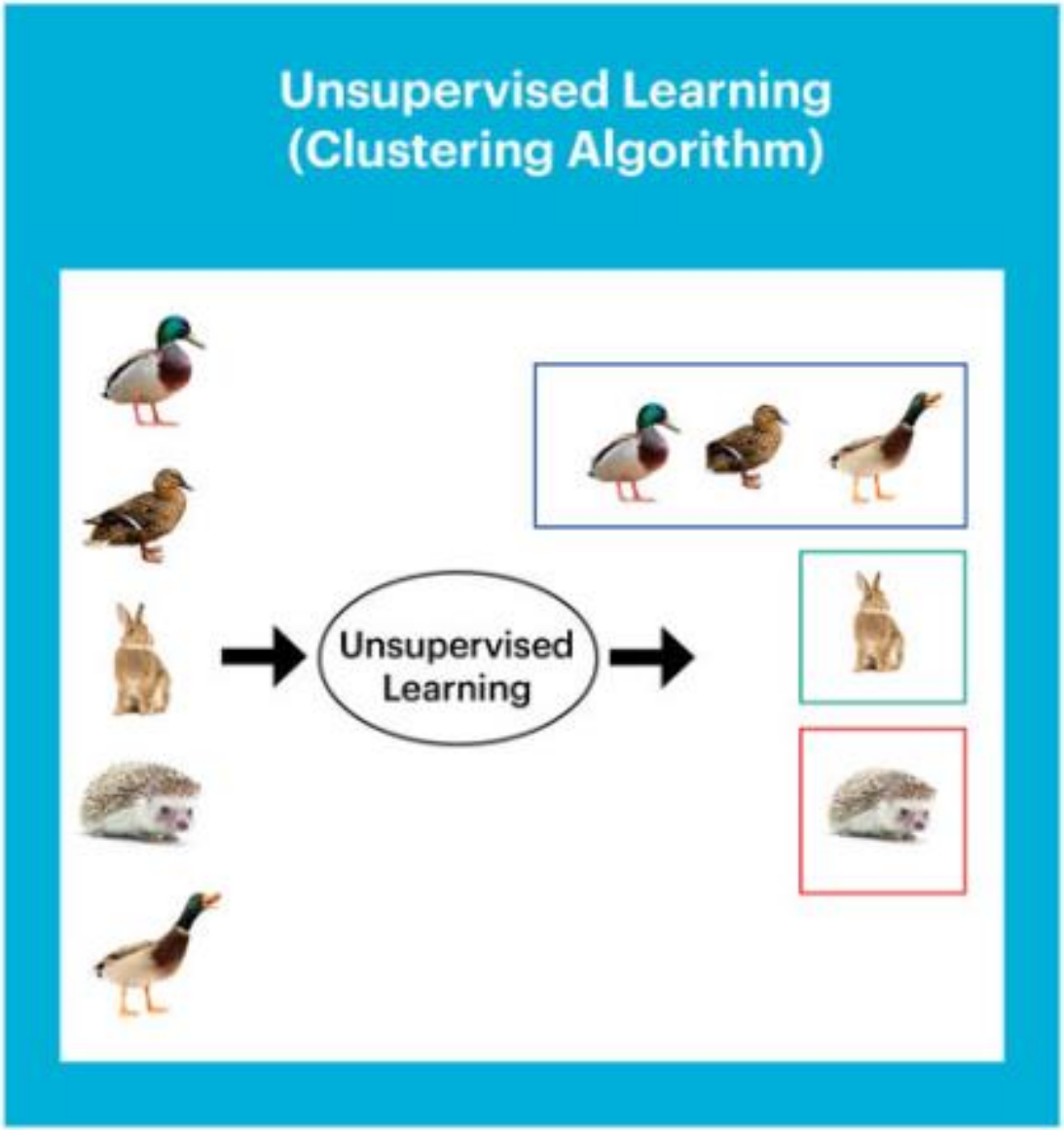
Trong học không giám sát, bộ dữ liệu huấn luyện là một tập hợp các quan sát ***không có nhãn gắn kèm***. Các thuật toán học không giám sát khai thác các thuộc tính chung của các quan sát trong bộ dữ liệu huấn luyện.

Học không giám sát (unsupervised learning)

Học máy không giám sát là nơi bạn chỉ có dữ liệu đầu vào (X) và không có biến đầu ra tương ứng.

Mục tiêu của việc học không giám sát là để mô hình hóa cấu trúc nền tảng hoặc sự phân bố trong dữ liệu để hiểu rõ hơn về nó.


Input Raw Data
Unlabeled data
Interpretation
Model Training
Processing
Square
Triangle

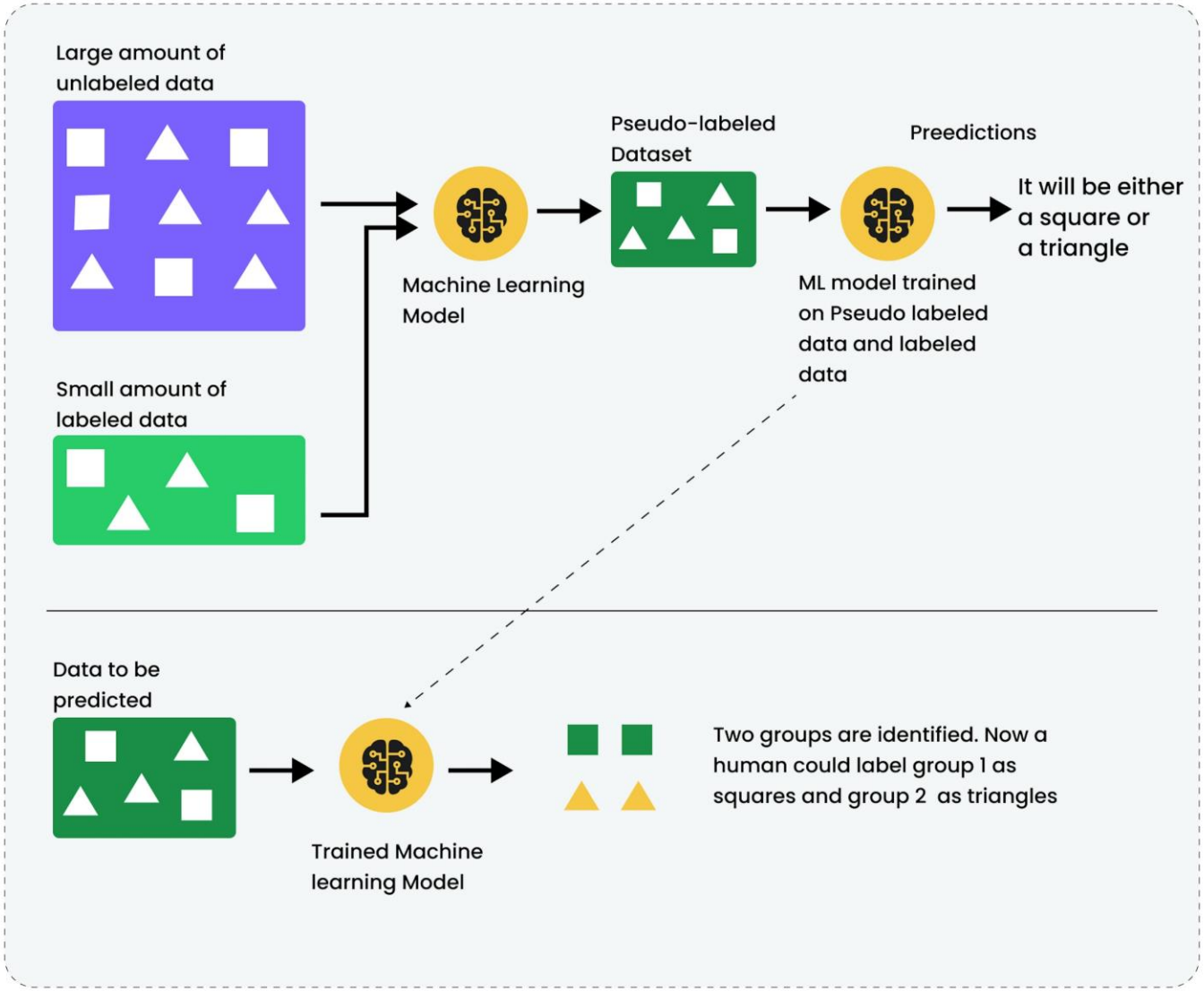

Unsupervised Learning
(Clustering Algorithm)
Unsupervised
Learning

Học máy giám sát một phần (Semi-supervised machine learning)

Khi bạn xây dựng mô hình trên một lượng lớn dữ liệu đầu vào (X) mà chỉ có một số dữ liệu được dán nhãn (Y) được gọi là việc học tập có giám sát một phần.

Nó nằm giữa việc học tập được giám sát và không giám sát.

Semi-supervised learning



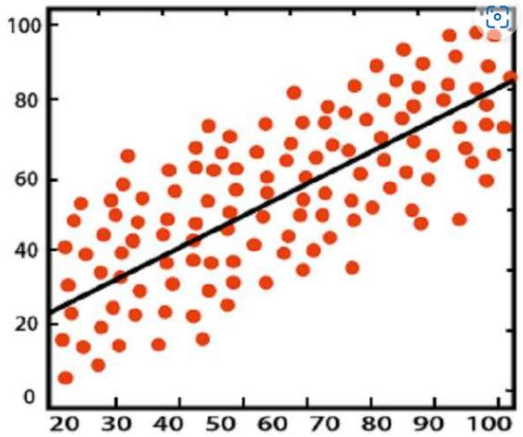
Bài toán hồi quy (Regression)

Thuộc nhóm học có giám sát (supervised learning)

Đặc điểm: Nhãn (label) có giá trị liên tục

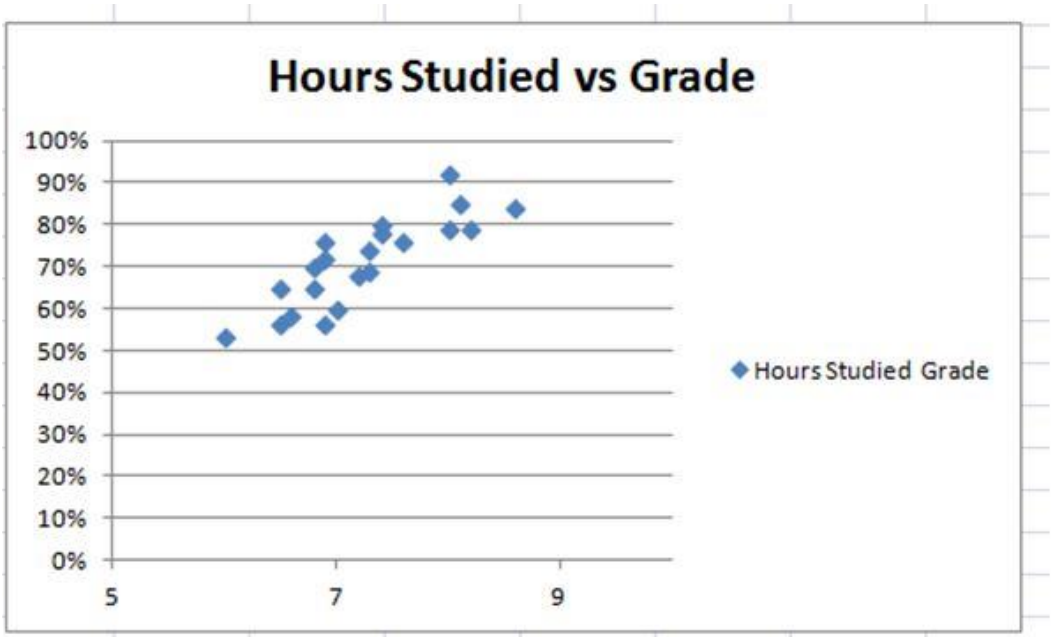
Mục tiêu: Dự đoán một giá trị số thực

- Ví dụ:
  - Dự đoán giá nhà
  - Dự đoán nhiệt độ
  - Dự báo doanh số bán hàng
  - Dự đoán tuổi thọ thiết bị



Regression

Student Name	Hours Studied	Grade
Jack	6	53%
Anne	7	60%
Harry	6,5	56%
Sharon	8	79%
John	6,6	58%
James	8,1	85%
Jill	6,8	70%
Adam	6,9	56%
Brandon	7,3	69%
Brett	6,9	76%
Brady	8,2	79%
Charles	7,2	68%
Darren	7,3	74%
Dave	6,9	72%
Dawn	8,6	84%
Denise	7,4	78%
Eric	7,6	76%
Emily	6,8	65%
Fred	8	92%
Fran	7,4	80%
Jane	6,5	65%

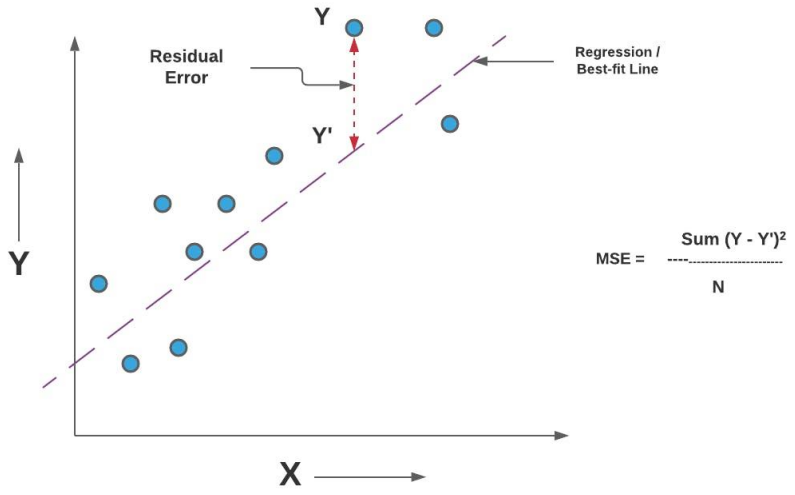


Độ đo hiệu suất cho bài toán hồi quy

- Mean Square Error (MSE):
  - Trung bình bình phương sai số
  - Đo lường khoảng cách trung bình giữa giá trị dự đoán và giá trị thực

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2$$

MSE = Mean Square error
n = Number of Data points
Yi = Observed Values
Y = Predicted Values



Độ đo hiệu suất cho bài toán hồi quy

- Root Mean Square Error (RMSE):
  - Căn bậc hai của MSE
  - Dễ diễn giải hơn vì cùng đơn vị với biến mục tiêu

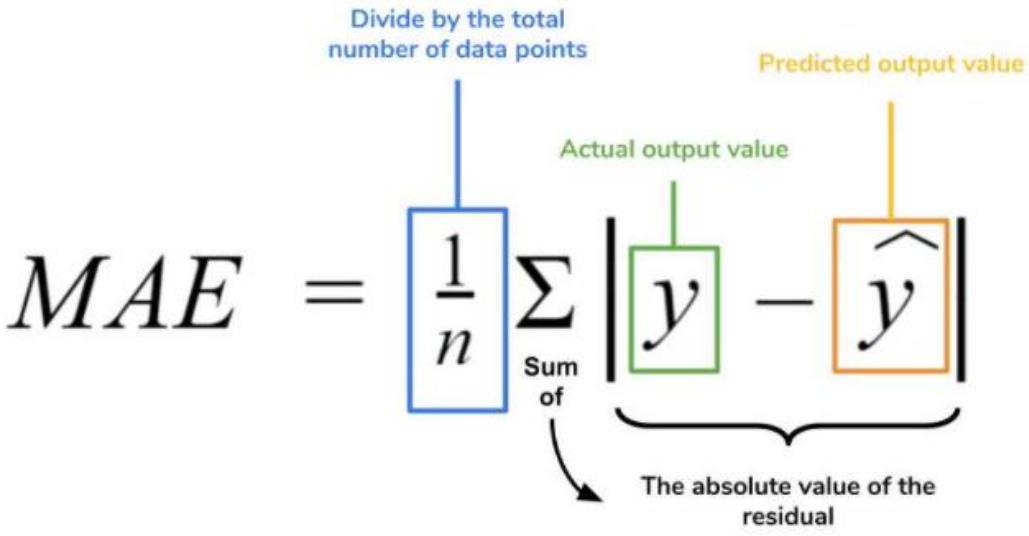
$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} \|y(i) - \hat{y}(i)\|^2}{N}},$$

RMSE = Root Mean Square Error
N = Number of Data points
Yi = Observed Values
Y = Predicted Values

Mean Absolute Error (MAE):

Trung bình giá trị tuyệt đối của sai số

Ít nhạy cảm với outliers hơn MSE



R-squared (R^2):

Đo lường phần trăm biến thiên được giải thích bởi mô hình

Giá trị từ 0 đến 1

$$R^2 = 1 - \frac{SS_{RES}}{SS_{TOT}} = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \overline{y})^2}$$

Khi nào R^2 tốt?

- $R^2 = 1.0$ (100%): Mô hình hoàn hảo
- $R^2 > 0.7$ (70%): Mô hình tốt
- $R^2 < 0.3$ (30%): Mô hình kém
- $R^2 = 0$: Mô hình không giải thích được gì

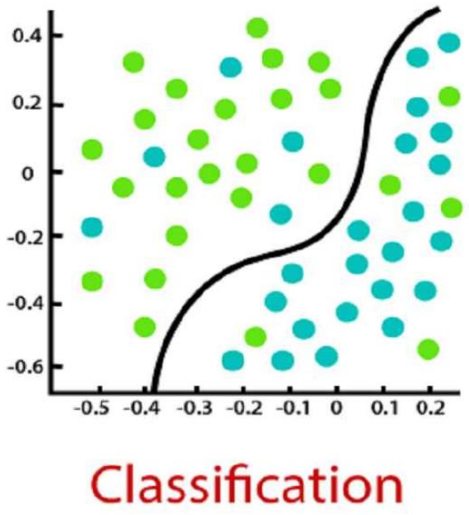
Bài toán phân lớp (Classification)

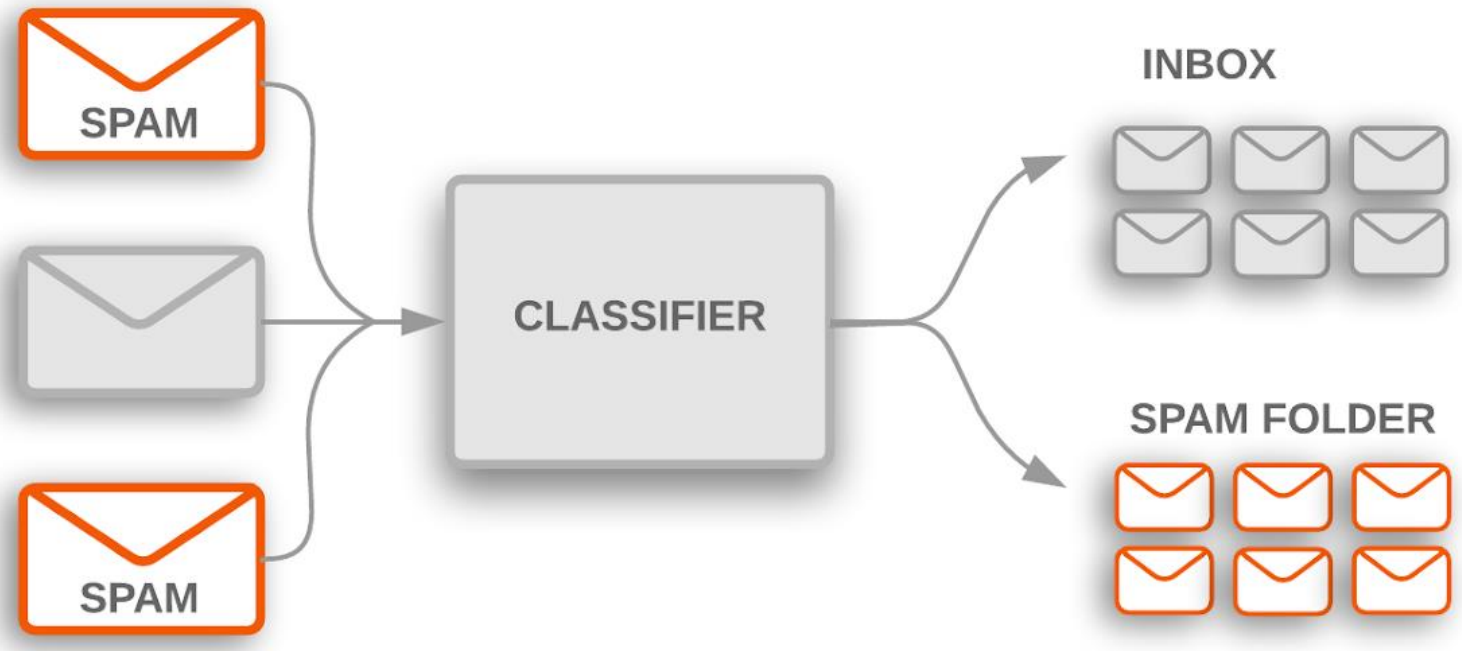
Thuộc nhóm học có giám sát (supervised learning)

Đặc điểm: Nhãn (label) có kiểu dữ liệu rời rạc

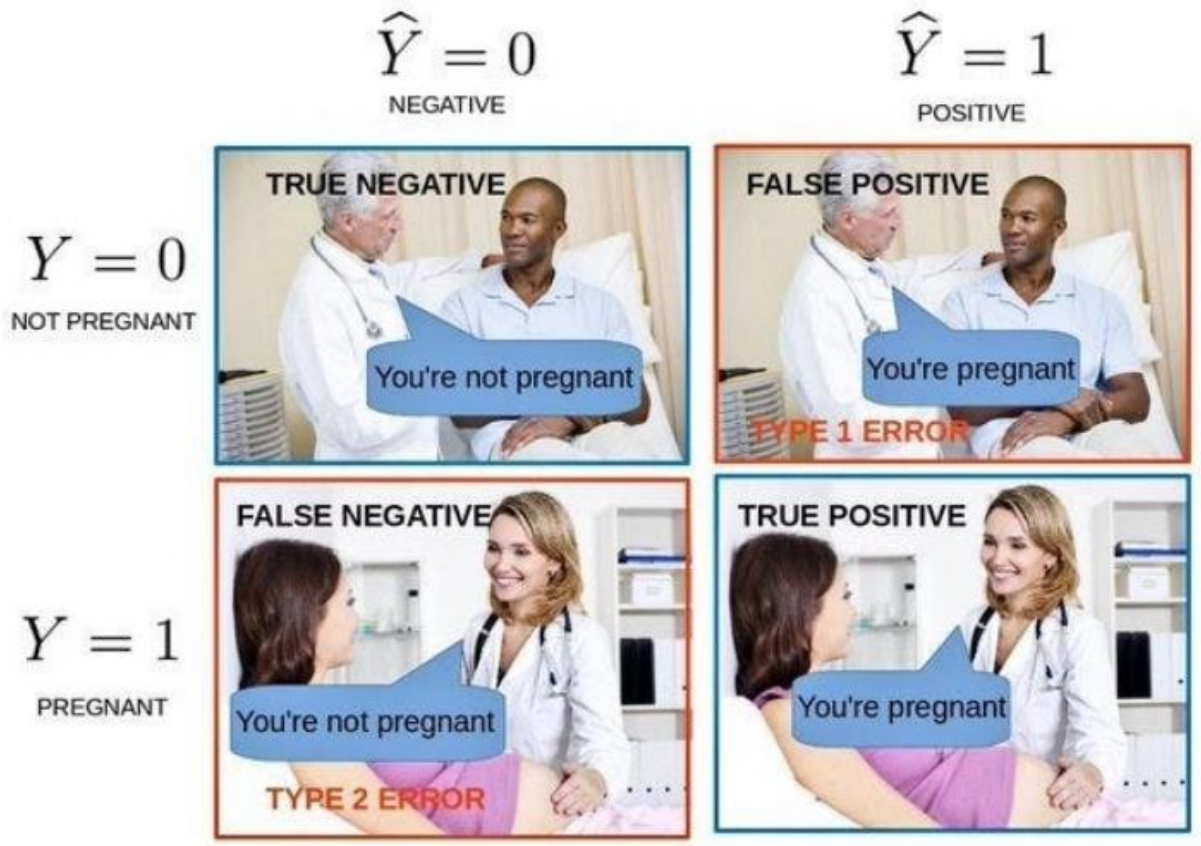
Mục tiêu: Phân loại đối tượng vào các nhóm định trước

- Ví dụ:
  - Phân loại thư rác/không rác
  - Nhận dạng chữ số viết tay
  - Chẩn đoán bệnh
  - Phân loại hình ảnh

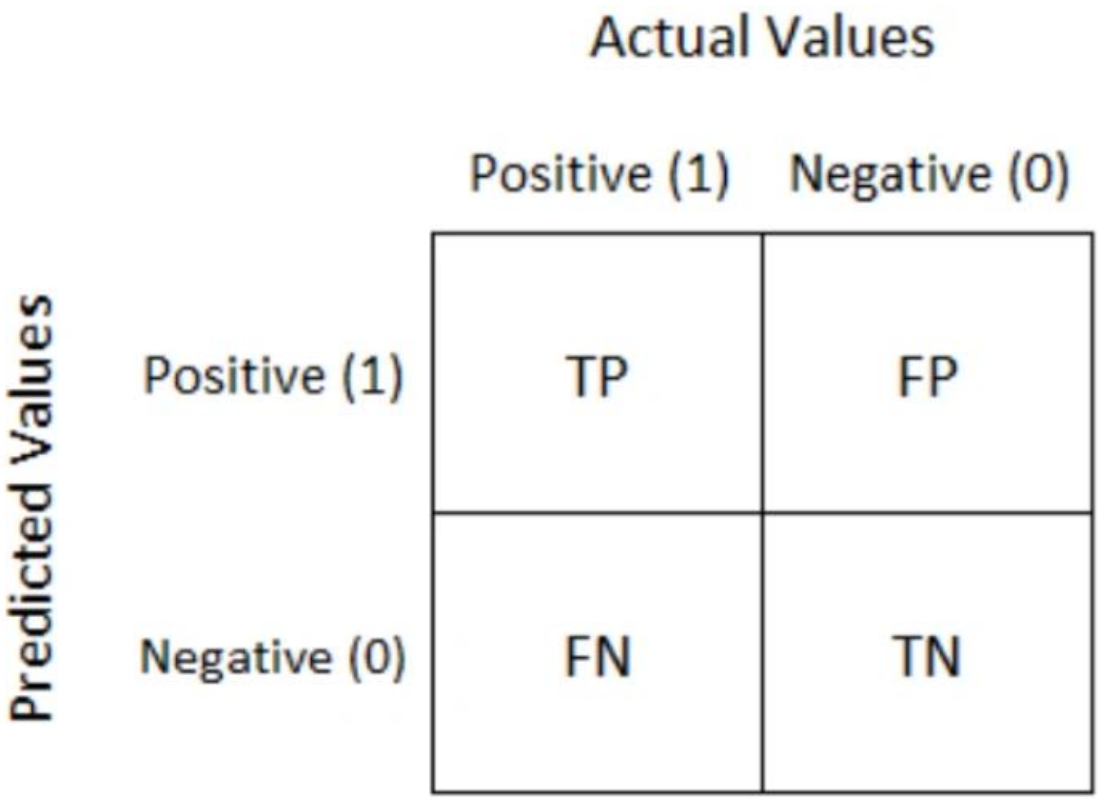



SPAM
SPAM
CLASSIFIER
INBOX
SPAM FOLDER

Độ đo hiệu suất cho bài toán phân lớp (Classification Metrics)



Độ đo hiệu suất cho bài toán phân lớp (Classification Metrics)



Độ đo hiệu suất cho bài toán phân lớp (Classification Metrics)

- Độ chính xác (Accuracy):
  - Tỷ lệ dự đoán đúng trên tổng số mẫu
  - Accuracy = (TP + TN)/(TP + TN + FP + FN)
- Độ chính xác dương tính (Precision):
  - Tỷ lệ dự đoán đúng trong số các dự đoán dương tính
  - Precision = TP/(TP + FP)
- Độ nhạy (Recall/Sensitivity):
  - Tỷ lệ phát hiện đúng các trường hợp dương tính
  - Recall = TP/(TP + FN)

Độ đo hiệu suất cho bài toán phân lớp (Classification Metrics)

- Điểm F1 (F1-score):
  - Trung bình điều hòa của Precision và Recall
  - F1 = 2 × (Precision × Recall)/(Precision + Recall)
- Diện tích dưới đường cong ROC

(Area Under the ROC Curve - AUC):

  - ROC (Receiver Operating Characteristic)
  - AUC đo lường khả năng phân biệt các lớp

Example

Mô hình A:
- Total transactions: 1000
- True Positives (TP): 90 (phát hiện đúng gian lận)
- False Positives (FP): 10 (báo gian lận nhầm)
- False Negatives (FN): 10 (bỏ sót gian lận)

Precision = 90/(90+10) = 0.90
Recall = 90/(90+10) = 0.90
F1-Score = 2 * (0.90 * 0.90)/(0.90 + 0.90) = 0.90

Example

Mô hình B:
- Total transactions: 1000
- TP: 95 (phát hiện đúng gian lận)
- FP: 5 (báo gian lận nhầm)
- FN: 20 (bỏ sót gian lận)

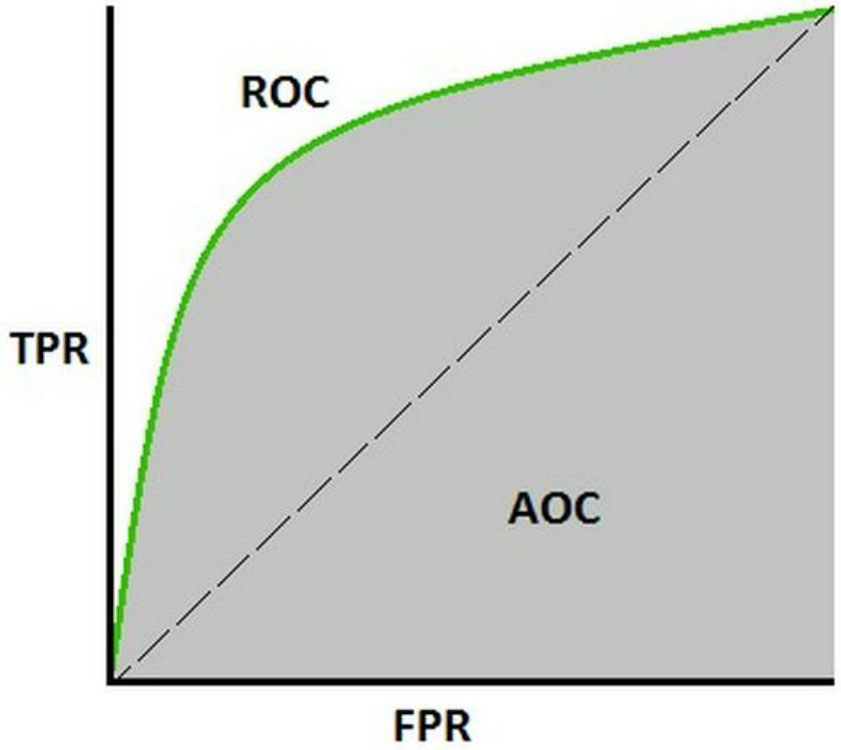
Precision = 95/(95+5) = 0.95
Recall = 95/(95+20) = 0.83
F1-Score = 2 * (0.95 * 0.83)/(0.95 + 0.83) = 0.88

Đánh giá F1-Score:

- F1 > 0.9: Rất tốt
- 0.7 < F1 < 0.9: Tốt
- 0.5 < F1 < 0.7: Trung bình
- F1 < 0.5: Kém

Đánh giá AUC-ROC:

- AUC > 0.95: Xuất sắc
- 0.90 - 0.95: Rất tốt
- 0.80 - 0.90: Tốt
- 0.70 - 0.80: Trung bình
- 0.60 - 0.70: Kém
- 0.50 - 0.60: Rất kém
- 0.50: Không tốt hơn đoán ngẫu nhiên



Độ đo hiệu suất cho bài toán phân lớp (Classification Metrics)

Lưu ý khi sử dụng:

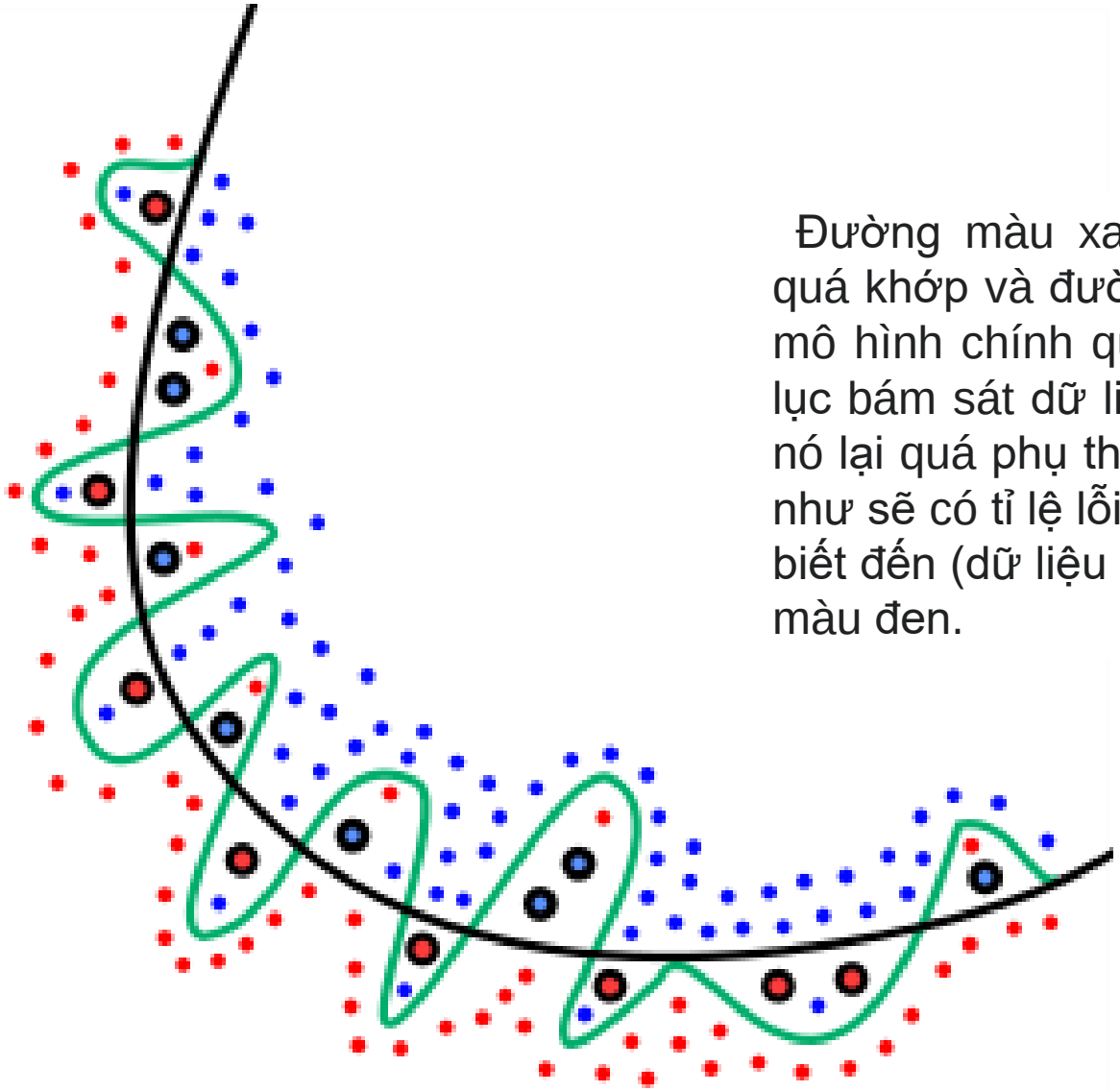
1. F1-Score:
   - Phù hợp khi cần cân bằng giữa precision và recall
   - Hữu ích cho dữ liệu không cân bằng (imbalanced data)
2. ROC-AUC:
   - Tốt cho việc so sánh các mô hình
   - Giúp chọn ngưỡng (threshold) phù hợp
   - Ít bị ảnh hưởng bởi dữ liệu không cân bằng

MỘT SỐ VẤN ĐỀ

- Overfitting (Quá khớp)
- Underfitting (Vị khớp)

Overfitting (Quá khớp)

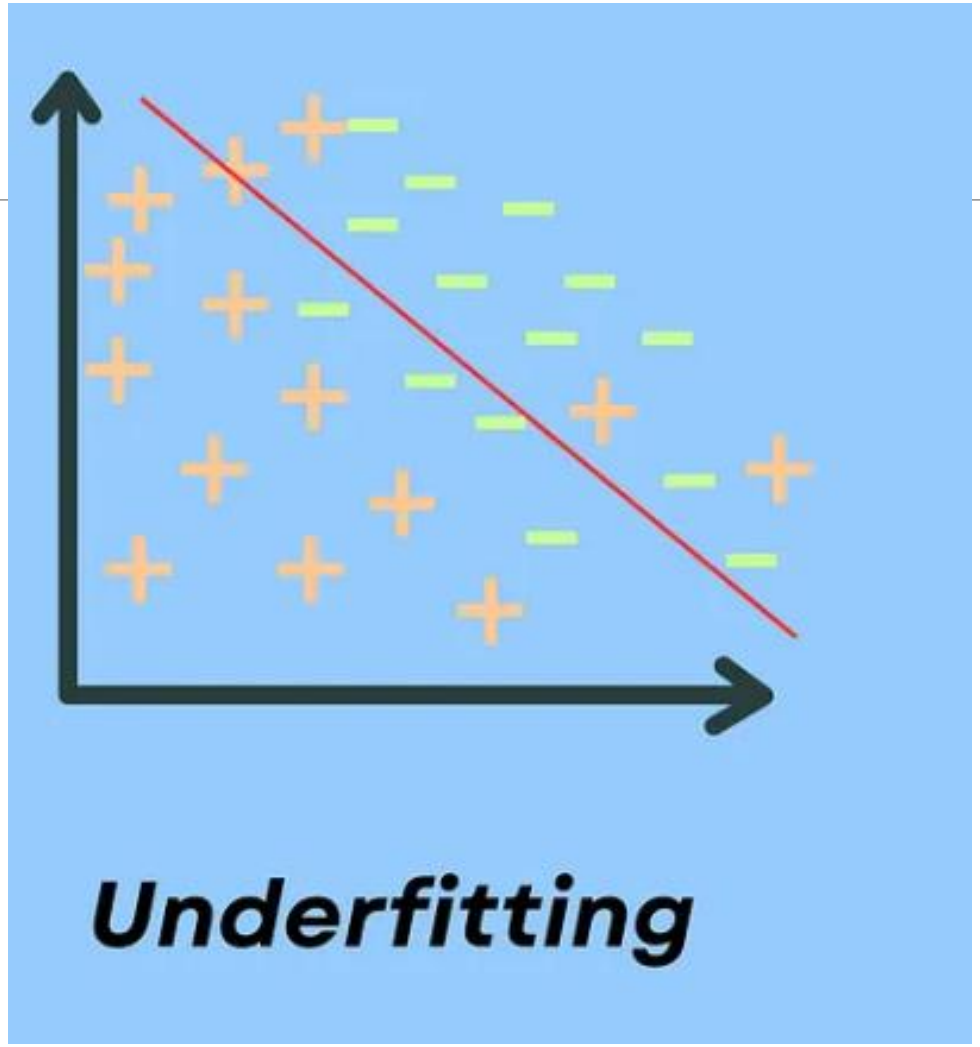
Overfitting xảy ra khi mô hình học quá kỹ từ dữ liệu huấn luyện, bao gồm cả nhiễu và các chi tiết không quan trọng. Kết quả là mô hình hoạt động rất tốt trên dữ liệu huấn luyện nhưng kém hiệu quả trên dữ liệu mới.



Đường màu xanh lục thể hiện mô hình quá khớp và đường màu đen thể hiện một mô hình chính quy. Trong khi đường xanh lục bám sát dữ liệu huấn luyện tốt nhất thì nó lại quá phụ thuộc vào dữ liệu và dường như sẽ có tỉ lệ lỗi cao trên các dữ liệu chưa biết đến (dữ liệu thử nghiệm) so với đường màu đen.

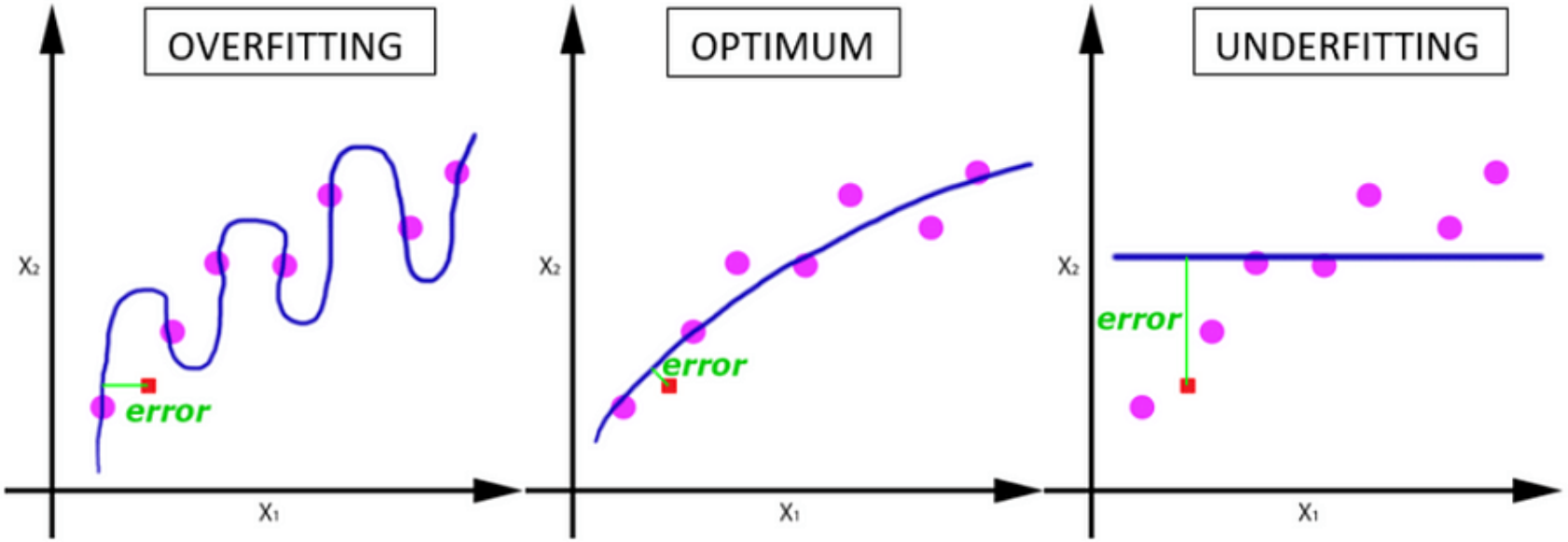
Underfitting (Vị khớp)

Underfitting xảy ra khi mô hình quá đơn giản và không thể nắm bắt được các mối quan hệ phức tạp trong dữ liệu. Mô hình này hoạt động kém cả trên dữ liệu huấn luyện và dữ liệu mới.


Underfitting

Overfitting và Underfitting

Để đạt được mô hình tốt, cần tìm sự cân bằng giữa overfitting và underfitting, thường được gọi là "sweet spot". Điều này có thể đạt được thông qua các kỹ thuật như điều chỉnh độ phức tạp của mô hình, sử dụng regularization, cross-validation, và tăng cường dữ liệu huấn luyện.

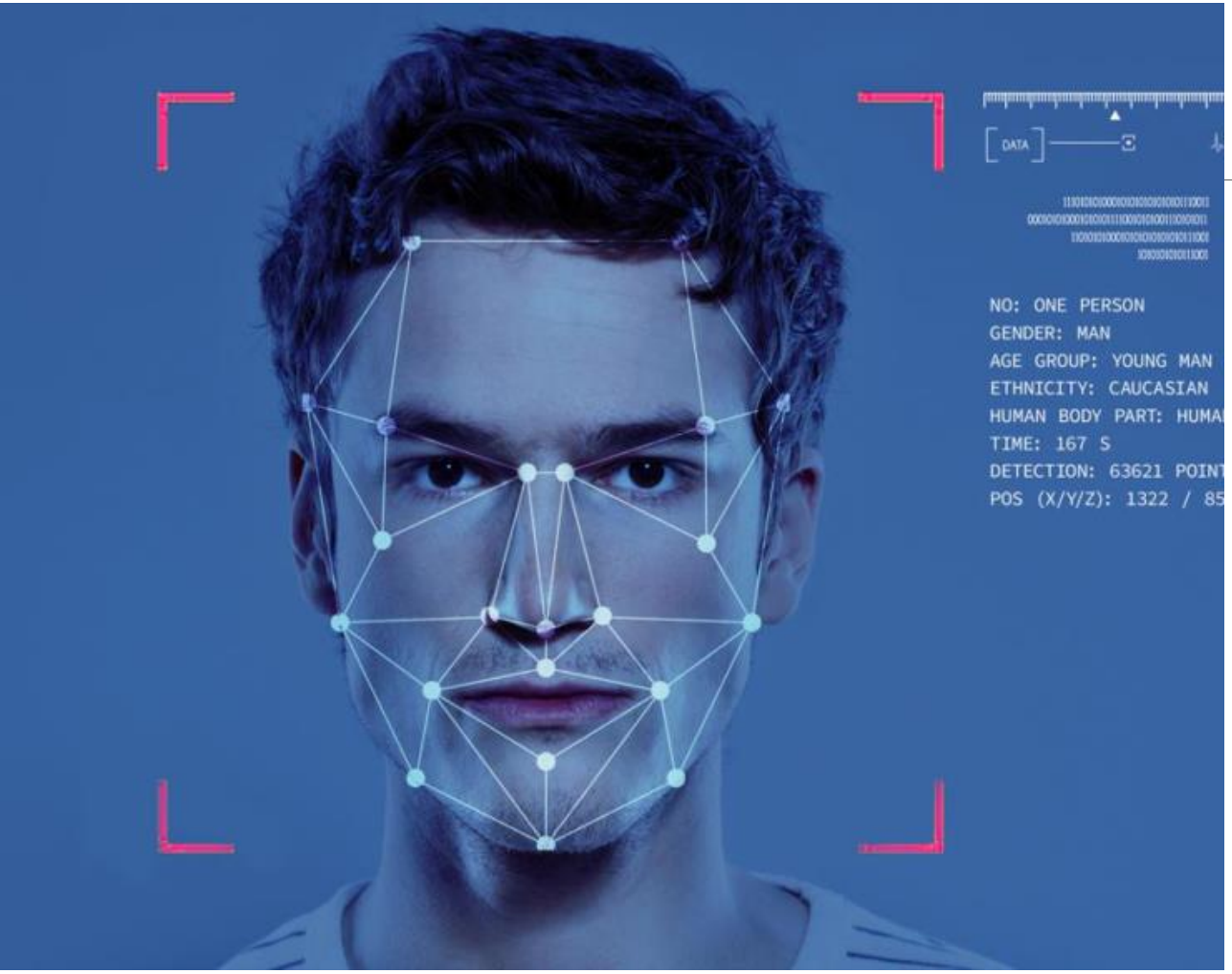

OVERFITTING
X2
X1
error
OPTIMUM
X2
X1
error
UNDERFITTING
X2
X1
error

1.3. ỨNG DỤNG CỦA HỌC SÂU

Lĩnh vực thị giác máy tính (computer vision):

Thị giác máy tính là một lĩnh vực của trí tuệ nhân tạo (AI) và xử lý ảnh, nghiên cứu về cách máy tính có thể hiểu và hiển thị thông tin từ hình ảnh và video.

Bao gồm các phương pháp thu nhận, xử lý ảnh kỹ thuật số, phân tích và nhận dạng các hình ảnh.

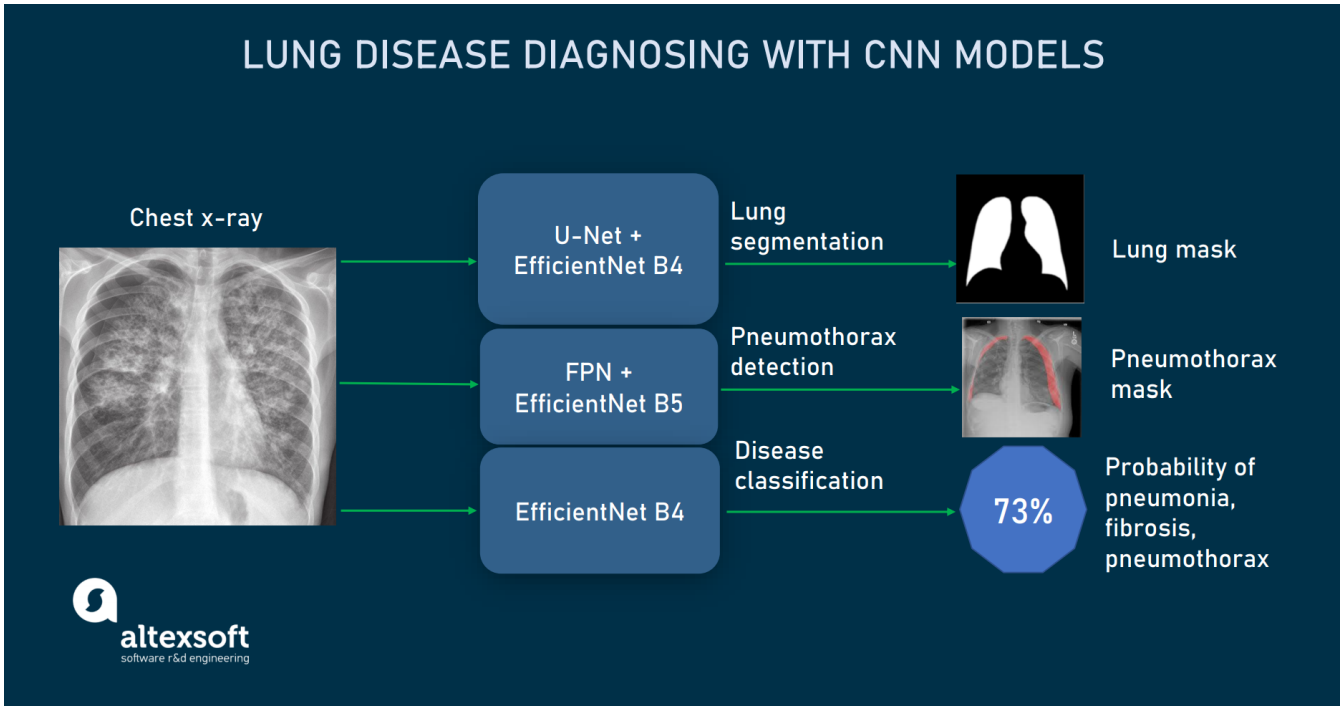


Ứng dụng của Thị giác máy tính

Nhận diện khuôn mặt


Reddit
Self-Driving
48

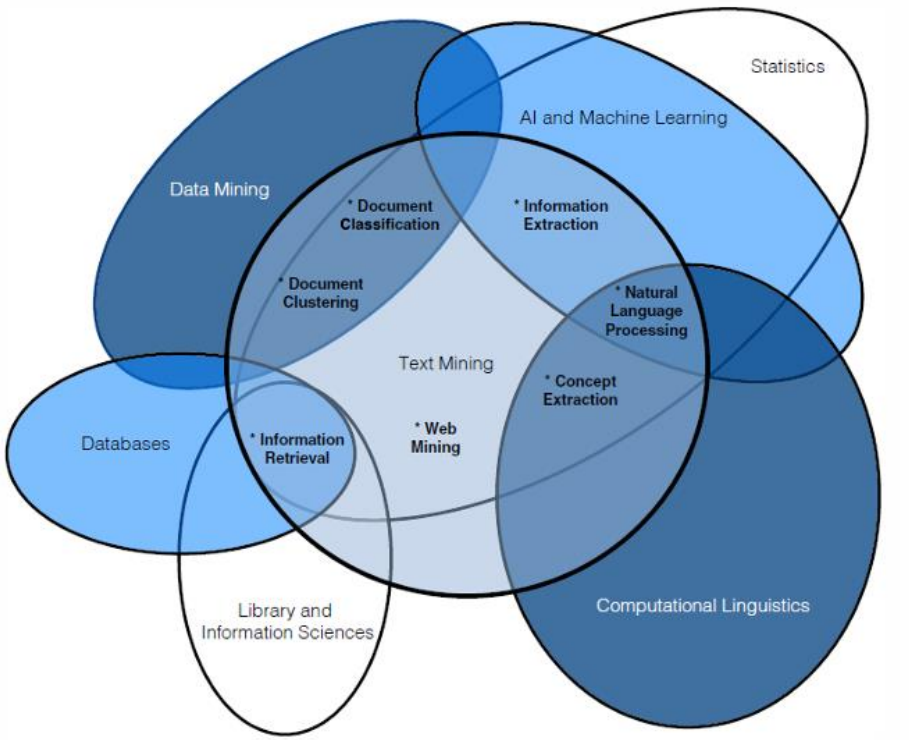
https://www.altexsoft.com/blog/computer-vision-healthcare/



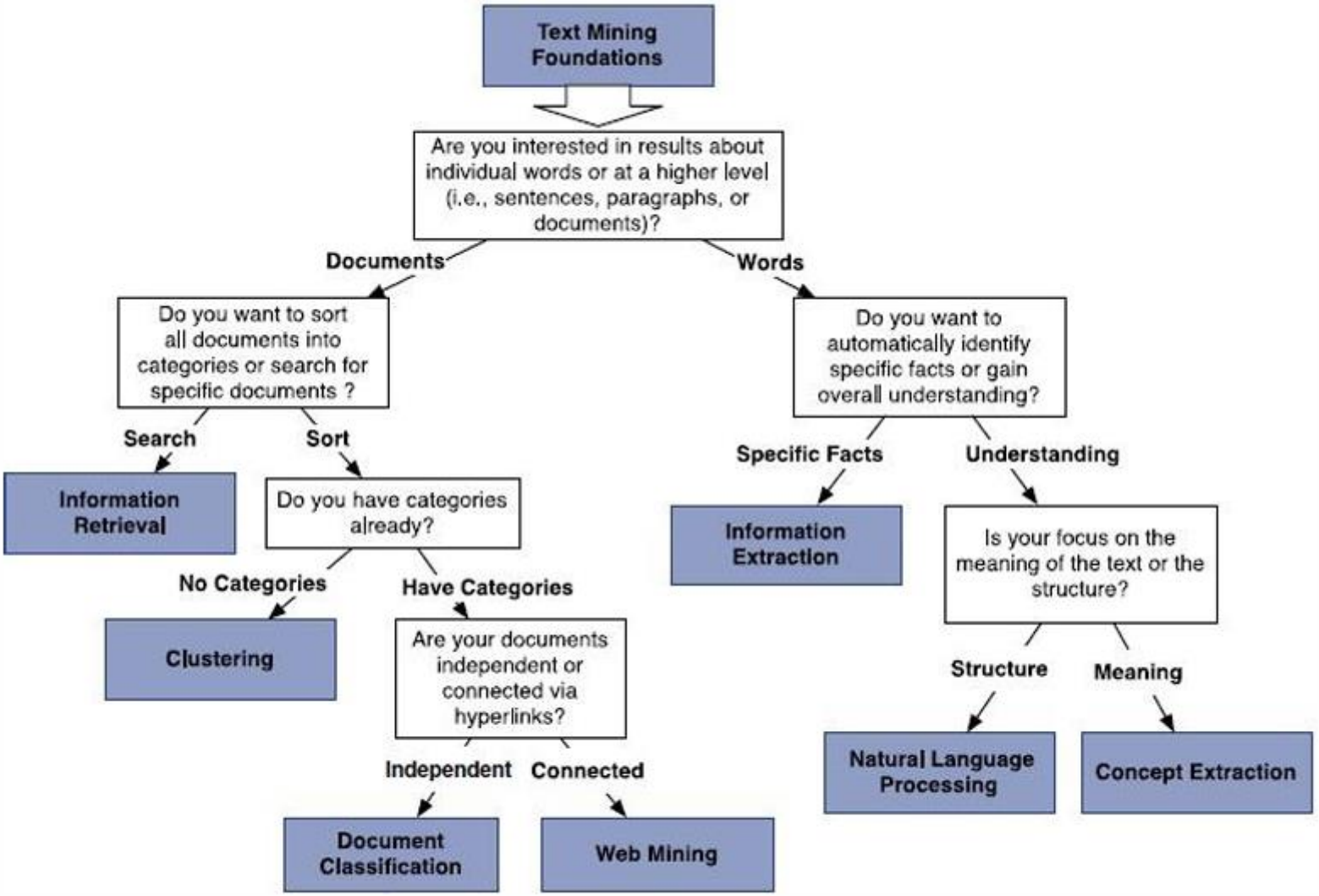
https://www.altexsoft.com/blog/computer-vision-healthcare/

1.3. ỨNG DỤNG CỦA HỌC SÂU

Lĩnh vực xử lý ngôn ngữ tự nhiên (Natural Language Processing – NLP)



Miner et.al 2012



Miner et.al 2012


TEXT ANALYTICS
Text Mining
Information Retrieval
Document Matching
Search Optimization
Inverted Index
Information Extraction
Entity Extraction
Co-reference
Relationship Exraction
Concept Extraction
Colocations
Word Association
Sentiment Analysis
Natural Language Processing
Tokenization
Lemmatization
Part-of-Speech Tagging
Web Mining
Web Content Mining
Web Analytics
Web Structure Analysis
Classification
Document Ranking
Alert Detection
Document Categorization
Clustering
Document Similarity
Document Clustering

1.3. ỨNG DỤNG CỦA HỌC SÂU

- Xe tự hành
- Tài chính
- Giải trí
- Sản xuất
- Nông nghiệp

1.4. QUY TRÌNH XÂY DỰNG HỆ THỐNG HỌC SÂU

1. Xác định vấn đề:

- Xác định mục tiêu: Định rõ vấn đề cần giải quyết và mục tiêu của hệ thống.

- Thu thập yêu cầu: Xác định các tiêu chí đánh giá như độ chính xác, precision, recall, F1-score.

1.4. QUY TRÌNH XÂY DỰNG HỆ THỐNG HỌC SÂU

2. Thu thập và chuẩn bị dữ liệu:

- Thu thập dữ liệu: Tập hợp dữ liệu cần thiết cho việc huấn luyện mô hình.

- Làm sạch dữ liệu: Loại bỏ nhiễu, xử lý giá trị thiếu và sửa chữa bất đồng nhất.

- Gán nhãn dữ liệu: Nếu cần, gán nhãn cho dữ liệu để cung cấp thông tin cho mô hình.

- Tăng cường dữ liệu: Áp dụng kỹ thuật để làm giàu tập dữ liệu huấn luyện.

- Chia dữ liệu: Phân chia thành tập huấn luyện, xác thực và kiểm tra.

1.4. QUY TRÌNH XÂY DỰNG HỆ THỐNG HỌC SÂU

3. Lựa chọn và thiết kế mô hình:

- Lựa chọn kiến trúc: Chọn kiến trúc phù hợp (như CNN, RNN, Transformers) dựa trên loại vấn đề.

- Điều chỉnh siêu tham số: Quyết định các siêu tham số như tốc độ học, kích thước lô, số lượng vòng lặp.

1.4. QUY TRÌNH XÂY DỰNG HỆ THỐNG HỌC SÂU

4. Huấn luyện mô hình:

- Thiết lập huấn luyện: Sử dụng các framework như TensorFlow hoặc PyTorch.

- Thực thi huấn luyện: Huấn luyện mô hình trên tập huấn luyện, theo dõi hiệu suất.

- Ngăn chặn overfitting: Sử dụng kỹ thuật như early stopping hoặc điều chỉnh tốc độ học.

1.4. QUY TRÌNH XÂY DỰNG HỆ THỐNG HỌC SÂU

5. Đánh giá mô hình:

- Đánh giá hiệu suất: Sử dụng tập kiểm tra để đánh giá mô hình dựa trên các tiêu chí đã xác định.

- Phân tích lỗi: Xem xét các trường hợp dự đoán sai để xác định lĩnh vực cần cải thiện.

1.4. QUY TRÌNH XÂY DỰNG HỆ THỐNG HỌC SÂU

6. Triển khai mô hình:

- Chiến lược triển khai: Quyết định cách triển khai mô hình (trên đám mây, thiết bị biên, ứng dụng).

- Tối ưu hóa: Giảm kích thước mô hình hoặc cải thiện tốc độ suy luận nếu cần.

- Giám sát và bảo trì: Thiết lập hệ thống giám sát hiệu suất và cập nhật khi cần thiết.

1.4. QUY TRÌNH XÂY DỰNG HỆ THỐNG HỌC SÂU

7. Cải thiện mô hình:

- Cập nhật liên tục: Huấn luyện lại mô hình với dữ liệu mới hoặc điều chỉnh kiến trúc.

- Học liên tục: Triển khai cơ chế để mô hình tiếp tục học từ dữ liệu mới.

```
# Step 1. Problem Understanding :
# Feature ? Target ? Classification or Regression
# Step 2.  Data Understanding :
# Missing Values ? Outlier(Noise) ? Inconsistent ? Imbalanced ? Skewness ?
# Step 3. Feature Understanding :
# EDA with Visualization on Univariate, BiVariate and MultiVariate Analysis
# Step 4. Feature Engineering :
# Skewness/Inconsistent/Missing/Outlier Handling,
# Feature Enrichment, Feature Transformation, Feature Selection, Feature Encoding,
# Feature Scaling (Normalization & Standardization)
# Step 5. Dataset Partition :
# Imblanced Handling, Train Test Split
# Step 6. Data Modelling :
# Try many ML methods
# Step 7. Data Evaluation :
# Display Metrics for Classification (Accuracy, Precision, Recall),
# Display Metrics for Regression (R2Score, MSE, RMSE)
# Step 8. Hyper-parameter Tuning :
# Tuning parameters: Cross Validation (CV), GridSearchCV, Regularization (L1, L2 penalty)
# Step 9. Build the pipeline with the best Model with the best parameters
# Choose best hyper-parameters and build best models
# Step 10. Conclusion
```

1.5. CÔNG CỤ LẬP TRÌNH HỌC SÂU

- Python
- Numpy
- Matplotlib
- Scipy
- Scikit-learn
- Pandas
- Tensorflow
- PyTorch