

SAMSUNG

Samsung Innovation Campus

| **Khóa học Dữ liệu lớn**

Together for Tomorrow!
Enabling People
Education for Future Generations

Chương 7.

Mô hình hoá Dữ liệu lớn & Trí tuệ nhân tạo

Khóa học Dữ liệu lớn

Mô tả chương

📌 Mục tiêu:

- ✓ Học máy là một lĩnh vực cốt lõi của trí tuệ nhân tạo và đang được sử dụng trong nhiều lĩnh vực khác nhau.
- ✓ Chúng ta sẽ tìm hiểu kiến thức cơ bản về học máy và các thuật toán liên quan cũng như nền tảng học máy đám mây công cộng. Cụ thể, nó thực hiện tiền xử lý dữ liệu bằng cách sử dụng Spark đã học trong chương trước.
 - Quy trình học máy với tập dữ liệu
 - Học máy cho thuật toán phân loại, hồi quy, phân cụm
 - SageMaker trong AWS và học máy Azure trong Azure
- ✓ Tìm hiểu cách tạo Apache Spark ML/MLlib Transformers, Estimators và Pipelines, đồng thời sử dụng các thuật toán máy học được tiêu chuẩn hóa được cung cấp trong API để tạo mô hình dữ liệu.
- ✓ Cuối cùng, chúng tôi sử dụng các thuật toán dựa trên hồi quy tuyến tính Spark ML để hiểu rõ hơn về cách thực hiện phân loại, hiểu một số thuật toán cơ bản và sử dụng chúng để tạo mô hình.

📌 Nội dung:

1. Học máy & Mô hình
2. Thư viện máy học Apache Spark

Bài 1.

Học máy & Mô hình

Mô hình hóa dữ liệu lớn & Trí tuệ nhân tạo

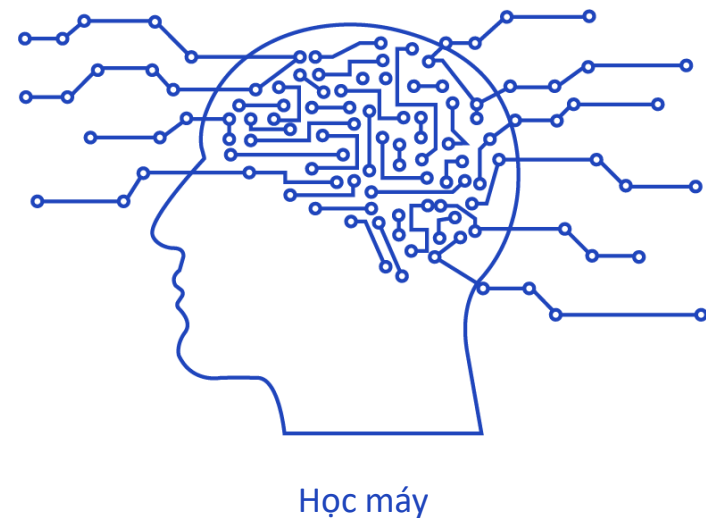
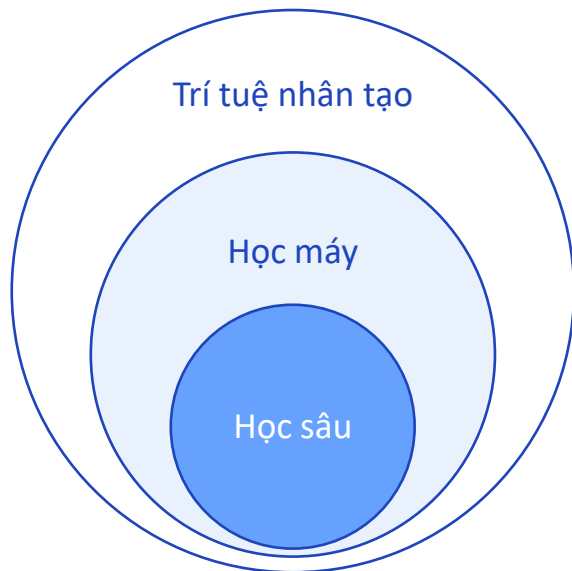
Bài 1

Học máy & Mô hình

- | 1.1. Kiến thức cơ bản về học máy
- | 1.2. Học máy trong đám mây công cộng
- | 1.3. Apache Spark ML

Học máy là gì? (1/2)

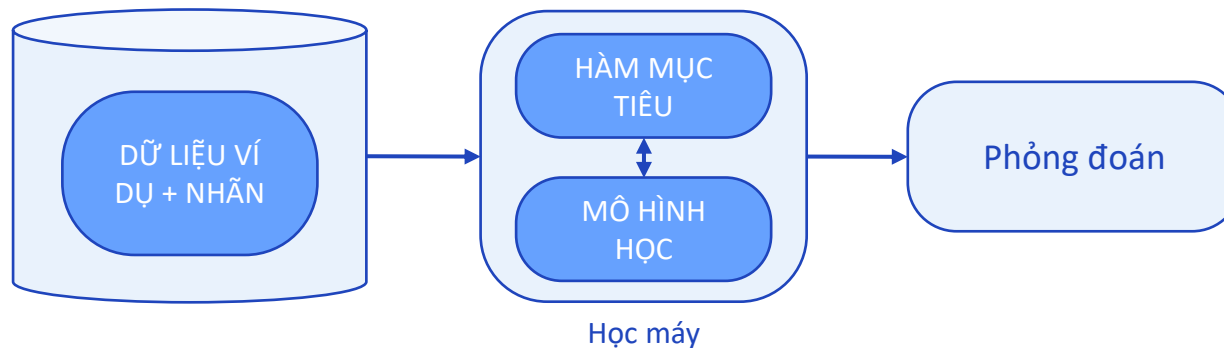
- | Các thuật toán và công nghệ cho phép máy tính học hỏi như một nhánh của trí tuệ nhân tạo
- | Lĩnh vực AI dựa trên học tập dữ liệu
- | Các thuật toán và công nghệ cho phép máy tính tự động học các tác vụ cụ thể thông qua kinh nghiệm tích lũy (dữ liệu) mà không cần sự can thiệp của con người



Học máy là gì? (2/2)

I Học

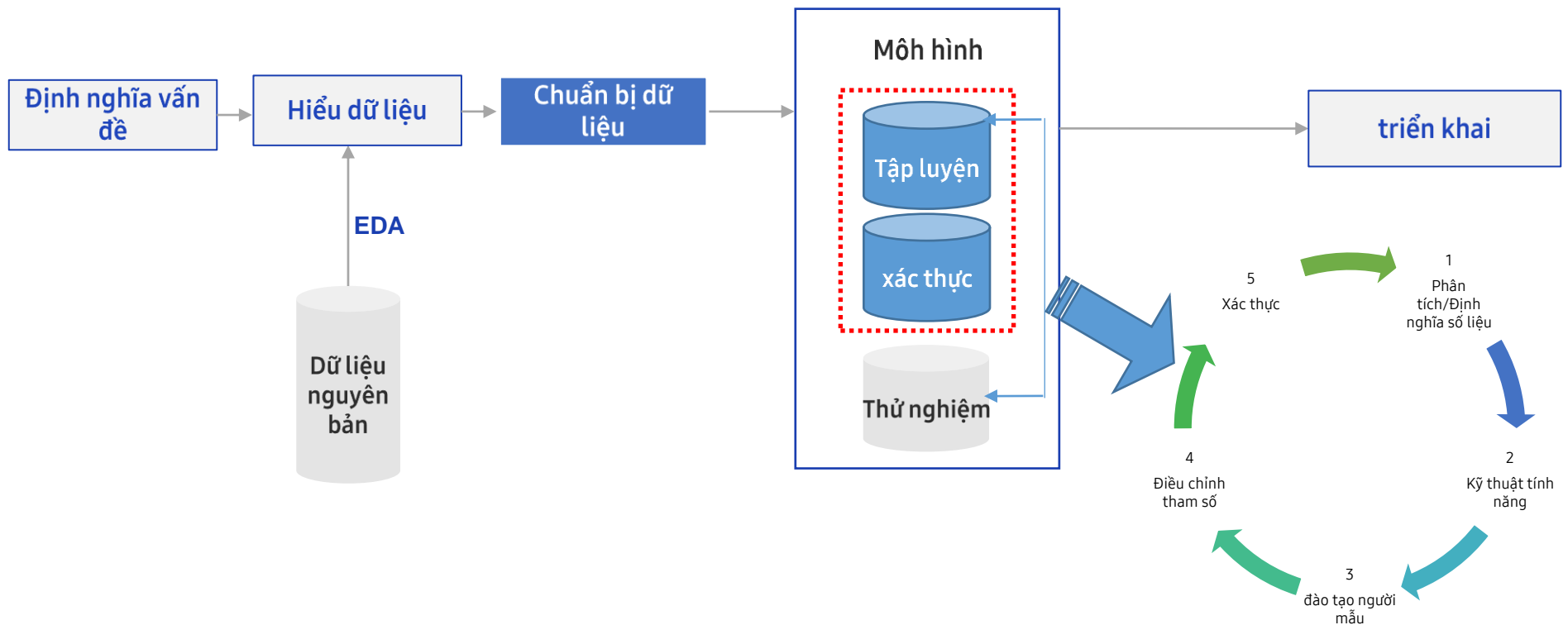
- ▶ Nhiệm vụ tạo mô hình thông qua một thuật toán cụ thể mô tả tốt nhất mối quan hệ giữa đầu vào và đầu ra trong dữ liệu



- ▶ Ví dụ: Dữ liệu đầu vào cho mô hình đào tạo
- ▶ Nhãn: đáp án chính xác cho VÍ DỤ
- ▶ Mô hình học: mô tả mối tương quan giữa đầu vào và đầu ra
- ▶ Hàm mục tiêu: thuật toán huấn luyện mô hình

Quy trình làm việc của Học máy (1/2)

I Định nghĩa vấn đề -> Hiểu dữ liệu -> Chuẩn bị dữ liệu -> Mô hình hóa -> Đánh giá -> Triển khai



Quy trình làm việc của Học máy (1/2)

I Quy trình tạo mẫu

- ▶ 1. Xác định mô hình và xây dựng nó
- ▶ 2. Huấn luyện mô hình bằng cách sử dụng dữ liệu. Và khớp mô hình với một mẫu trong dữ liệu
- ▶ 3. Đánh giá mức độ mô hình được đào tạo đại diện cho mẫu dữ liệu (thử nghiệm)

$$Y = f(x)$$

- ▶ Y(Label) là biến đầu ra hay biến phụ thuộc
- ▶ X(Feature) là biến đầu vào hay biến độc lập
- ▶ f là mô hình

Thuật ngữ và định nghĩa (1/3)

I Mô hình

- ▶ Một mô hình có nghĩa là một chương trình
- ▶ Một chức năng xây dựng một mẫu trong dữ liệu
- ▶ Một đại diện đơn giản hóa của thế giới thực để đạt được một mục đích

I Học

- ▶ Học đề cập đến công nghệ trong đó máy tính tự tìm quy tắc trong dữ liệu

I Đào tạo

- ▶ Đào tạo theo nghĩa đen là quá trình thực hiện việc học

I Đầu vào

- ▶ Một đầu vào là những gì mô hình (chương trình) nên tìm kiếm các quy tắc

I Mục tiêu

- ▶ Mục tiêu có nghĩa là câu trả lời đúng mà mô hình (chương trình) phải phù hợp

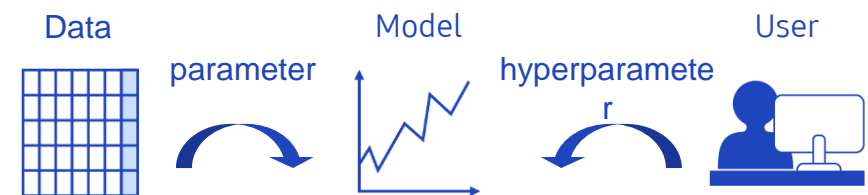
Thuật ngữ và định nghĩa (2/3)

I Thông số:

- ▶ Được học từ dữ liệu bằng cách đào tạo và không được thiết lập thủ công bởi người thực hành
- ▶ Chứa mẫu dữ liệu
- ▶ Ví dụ) Hệ số hồi quy tuyến tính
- ▶ Ví dụ) Trọng số của mạng lưới thần kinh

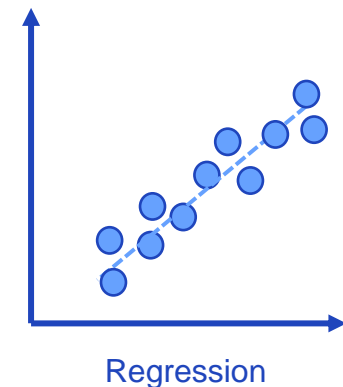
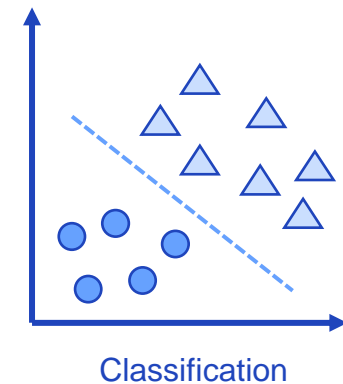
I Siêu tham số:

- ▶ Người thực hành có thể cài đặt thủ công
- ▶ Có thể được điều chỉnh để tối ưu hóa hiệu suất học máy
- ▶ Ví dụ) k trong thuật toán KNN
- ▶ Ví dụ) Tốc độ học tập trong mạng lưới thần kinh
- ▶ Ví dụ) Độ sâu tối đa trong thuật toán Tree



Thuật ngữ và định nghĩa (3/3)

- I Phân loại: Mô hình dự đoán xấp xỉ hàm ánh xạ từ các biến đầu vào để xác định các biến đầu ra rời rạc
 - ▶ Phân loại cây quyết định
 - ▶ Phân loại rừng ngẫu nhiên
 - ▶ K-khu vực gần nhất
- I Hồi quy: Một mô hình dự đoán ước tính hàm ánh xạ dựa trên các biến đầu vào và đầu ra.
 - ▶ Hồi quy tuyến tính cơ bản
 - ▶ Hồi quy tuyến tính bội
 - ▶ hồi quy đa thức
- I Sự khác biệt đáng kể nhất giữa hồi quy và phân loại là trong khi hồi quy giúp dự đoán một số lượng liên tục, thì phân loại dự đoán các nhãn lớp rời rạc.



Các loại học máy

I Giám sát

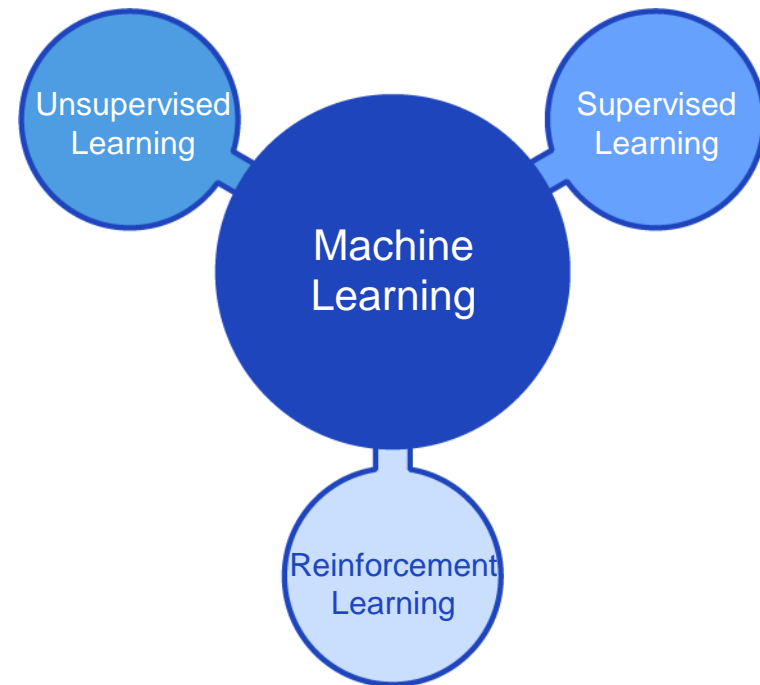
- ▶ Cách đào tạo mô hình bằng dữ liệu đầu vào và dữ liệu câu trả lời đúng tương ứng (nhãn)

I Không giám sát

- ▶ Chỉ học với dữ liệu đầu vào mà không có dữ liệu câu trả lời đúng

I Tăng cường học

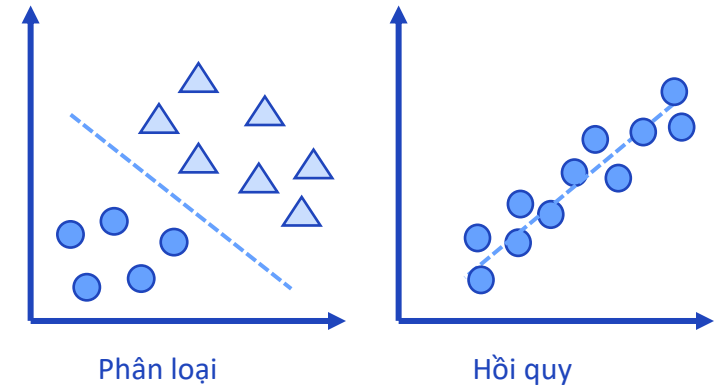
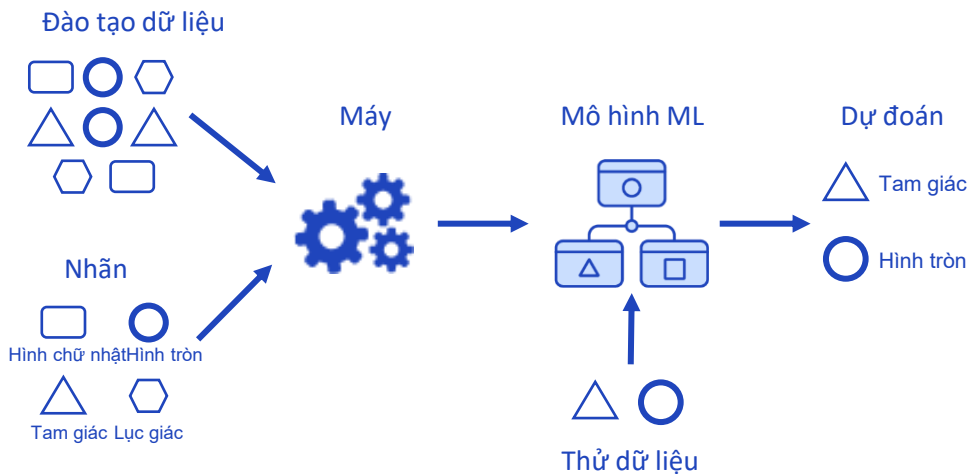
- ▶ Cho phép một tác nhân học hỏi trong một môi trường tương tác bằng cách thử và sai bằng cách sử dụng phản hồi từ các hành động và trải nghiệm của chính nó



Học có giám sát

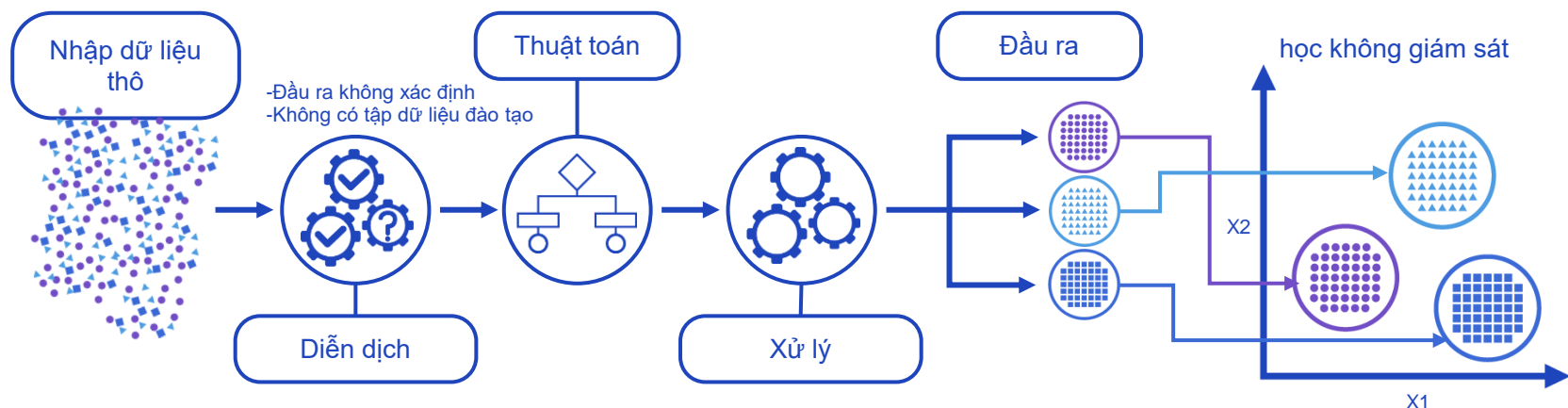
! Mục đích chính của học có giám sát là đào tạo một mô hình trên dữ liệu đào tạo được gắn nhãn để đưa ra dự đoán về dữ liệu không nhìn thấy được trong tương lai. Được giám sát ở đây có nghĩa là một loạt mẫu có tín hiệu đầu ra mong muốn (nhãn)

- ▶ Phân loại
- ▶ Hồi quy



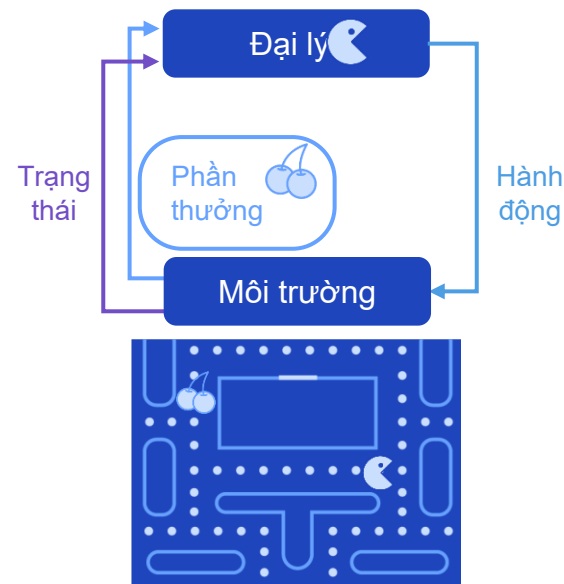
Học không có giám sát

- A Một phương pháp đào tạo một mô hình sử dụng dữ liệu đào tạo mà không có mục tiêu
- Nó xử lý dữ liệu không được gắn nhãn hoặc không có cấu trúc. Tạo một mô hình thuật toán gồm các mẫu và quy tắc cụ thể không có dữ liệu được gắn nhãn
 - ▶ phân cụm
 - ▶ Phân tích thành phần chính (PCA)
 - ▶ Phân tách giá trị số ít (SVD)



Học tăng cường

- | Học tăng cường nhằm mục đích cải thiện hiệu suất của hệ thống (tác nhân) bằng cách tương tác với môi trường
- | Các đại lý tương tác với môi trường của họ để tìm hiểu một tập hợp các hành vi giúp tối đa hóa phần thưởng của họ thông qua học tập tăng cường
 - ▶ Phương pháp thử và sai thăm dò
 - ▶ Áp dụng cho trò chơi



Học có giám sát so với học không giám sát

	Có giám sát	Không có giám sát
Các mục tiêu	Mục tiêu là dự đoán kết quả cho dữ liệu mới. Bạn biết trước loại kết quả mong đợi.	Mục tiêu là để có được thông tin chi tiết từ khối lượng lớn dữ liệu mới. Bản thân máy học xác định điều gì khác biệt hoặc thú vị với tập dữ liệu.
Dữ liệu đầu vào	Các thuật toán được đào tạo bằng cách sử dụng dữ liệu được dán nhãn.	Các thuật toán được sử dụng đối với dữ liệu không được gán nhãn.
Tính phức tạp	Học có giám sát là một phương pháp đơn giản hơn.	Học tập không giám sát là phức tạp về mặt tính toán.
Thời gian học thực tế	Phương pháp có độ chính xác cao và đáng tin cậy.	Phương pháp kém chính xác và đáng tin cậy hơn.
Độ chính xác của kết quả	Phương pháp có độ chính xác cao và đáng tin cậy.	Phương pháp kém chính xác và đáng tin cậy hơn.
Các hạn chế	Các mô hình học tập có giám sát có thể tốn nhiều thời gian để đào tạo và nhãn cho các biến đầu vào và đầu ra đòi hỏi phải có kiến thức chuyên môn.	Các phương pháp học không giám sát có thể có kết quả cực kỳ không chính xác trừ khi bạn có sự can thiệp của con người để xác thực các biến đầu ra.

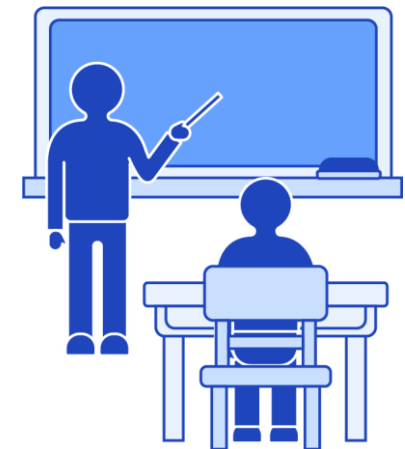
Điều gì tốt nhất cho bạn?

- I Việc chọn phương pháp phù hợp cho tình huống của bạn tùy thuộc vào cách các nhà khoa học dữ liệu đánh giá cấu trúc và khối lượng dữ liệu của bạn, cũng như trường hợp sử dụng
 - ▶ Đánh giá dữ liệu đầu vào của bạn
 - ▶ Xác định mục tiêu của bạn
 - ▶ Xem lại các tùy chọn của bạn cho các thuật toán
- I Phân loại dữ liệu lớn có thể là một thách thức thực sự trong học tập có giám sát, nhưng kết quả rất chính xác và đáng tin cậy
- I Học không giám sát có thể xử lý khối lượng lớn dữ liệu trong thời gian thực. Tuy nhiên, có sự thiếu minh bạch về cách dữ liệu được phân cụm và nguy cơ kết quả không chính xác cao hơn
- I Học bán giám sát là một phương tiện hỗn hợp, trong đó bạn sử dụng tập dữ liệu huấn luyện với cả dữ liệu được gắn nhãn và không được gắn nhãn
 - ▶ Nó đặc biệt hữu ích khi khó trích xuất các tính năng có liên quan từ dữ liệu — và khi bạn có một lượng lớn dữ liệu

Tại sao nên sử dụng Học có giám sát?

- | Học máy có giám sát giúp bạn giải quyết nhiều loại vấn đề tính toán trong thế giới thực
- | Học có giám sát thường được thực hiện trong bối cảnh phân loại
 - ▶ khi chúng tôi muốn ánh xạ đầu vào sang nhãn đầu ra hoặc hồi quy
 - ▶ khi chúng ta muốn ánh xạ đầu vào thành đầu ra liên tục
- | Học có giám sát cho phép bạn thu thập dữ liệu hoặc tạo đầu ra dữ liệu từ trải nghiệm trước đó
- | Các loại thuật toán học có giám sát
 - ▶ Hồi quy logistic
 - ▶ Naive Bayes
 - ▶ Máy vectơ hỗ trợ (SVM)
 - ▶ rừng ngẫu nhiên
 - ▶ K-Láng giềng gần nhất (KNN)

Supervised Learning



Tại sao nên sử dụng học không giám sát?

- Nó tương tự như cách một con người học hỏi. Nó liên quan đến suy nghĩ bằng kinh nghiệm, giúp nó tiến gần hơn đến AI thực
- Nó hoạt động trên dữ liệu không được gắn nhãn, điều này làm cho việc học không giám sát trở nên quan trọng hơn vì dữ liệu trong thế giới thực hầu như không được gắn nhãn
- Học máy không giám sát tìm thấy tất cả các loại mẫu không xác định trong dữ liệu
- Nó giúp tìm kiếm những hiểu biết hữu ích từ dữ liệu
- Các loại thuật toán học không giám sát
 - ▶ K-có nghĩa là phân cụm
 - ▶ Phân cụm theo cấp bậc
 - ▶ Phát hiện bất thường
 - ▶ PCA
 - ▶ Thuật toán Apriori

Unsupervised learning



Chọn thuật toán nào? (1/2)

[Quiz]

Chọn một thuật toán thích hợp theo đặc điểm và hoàn cảnh của dữ liệu thu thập được.

- Nếu bạn muốn trả lời câu hỏi CÓ|KHÔNG, đó là phân loại
- Nếu bạn muốn dự đoán một giá trị số, đó là hồi quy
- Nếu bạn muốn nhóm dữ liệu thành các quan sát tương tự, đó là phân cụm

Trường hợp 1

- Phân khúc khách hàng: chia cơ sở khách hàng thành các nhóm cá nhân giống nhau theo những cách cụ thể liên quan đến tiếp thị, chẳng hạn như tuổi tác, giới tính, sở thích, thói quen chi tiêu, v.v.
- Phân khúc thị trường

Chọn thuật toán nào? (2/2)

[Quiz]

Trường hợp 2

- Doanh số / Dự báo chứng khoán
- Phí bảo hiểm dựa trên các yếu tố khác nhau
- Dự đoán lỗi thiết bị

Trường hợp 3

- Thư rác
- Phân tích khẩu phần khách hàng
- Nhận dạng mẫu: từ ngữ, con người, hình ảnh, chuyển động, v.v.
- Phát hiện giao dịch gian lận

Các loại học máy

Loại	Thuật toán/Phương pháp
Học không giám sát	Clustering
	MDS, t-SNE
	PCA, NMF
	Phân tích hiệp hội
Học có giám sát	Hồi quy tuyến tính
	Hồi quy logistic
	Tree, Random Forest, Ada Boost, XGBoost.
	Naïve Bayes
	KNN
	Máy véc tơ hỗ trợ (SVM)
	mạng lưới thần kinh

Thuật toán KNN (1/4)

I KNN (K khu vực gần nhất)

- ▶ Thuật toán K-NN là thuật toán phân loại các nhãn của dữ liệu có tần suất xuất hiện cao nhất trong K phần dữ liệu bằng cách quy chiếu các nhãn (thuộc tính) của K phần dữ liệu khác gần nhau
- ▶ Dự đoán dựa trên k điểm lân cận gần nhất

I Ưu điểm:

- ▶ Đơn giản và trực quan
- ▶ Không có tham số mô hình để tính toán. Vì vậy, không có bước đào tạo

I Nhược điểm:

- ▶ Vì không có “mô hình” nên có thể trích xuất một chút thông tin chi tiết
- ▶ Không có tham số mô hình nào lưu trữ mẫu đã học. Tập dữ liệu huấn luyện là cần thiết để dự đoán
- ▶ Dự đoán không hiệu quả ⇨ “Thuật toán lười biếng”

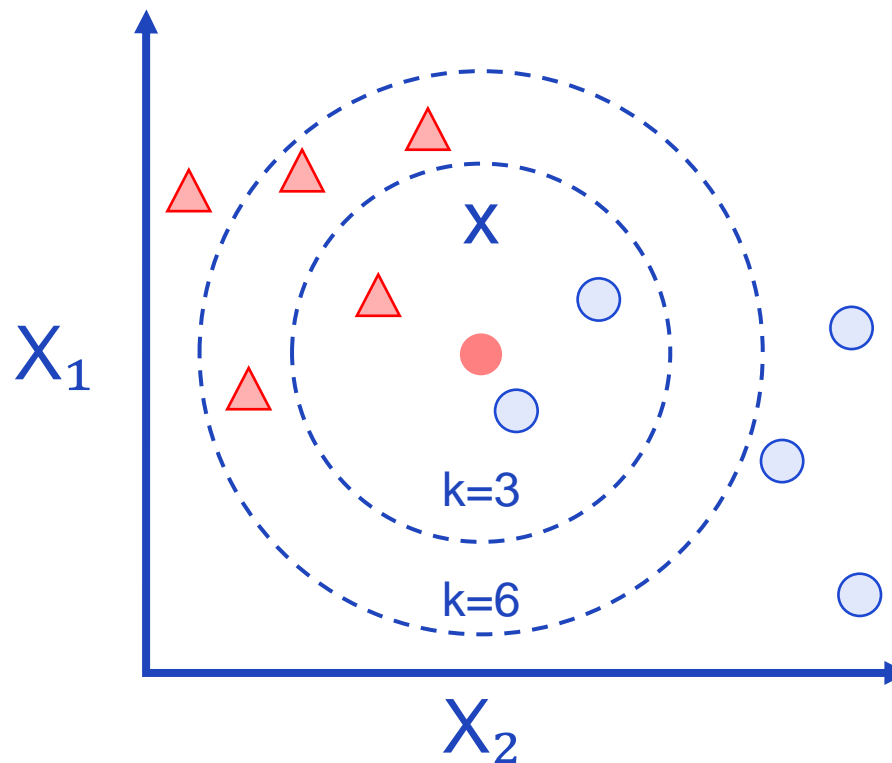
Thuật toán KNN (2/4)

I $K=3$

▶ X được phân loại là 

I $K=6$

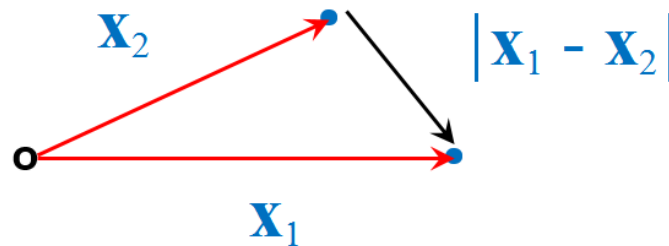
▶ X được phân loại là 



Thuật toán KNN (3/4)

I Chỉ số khoảng cách:

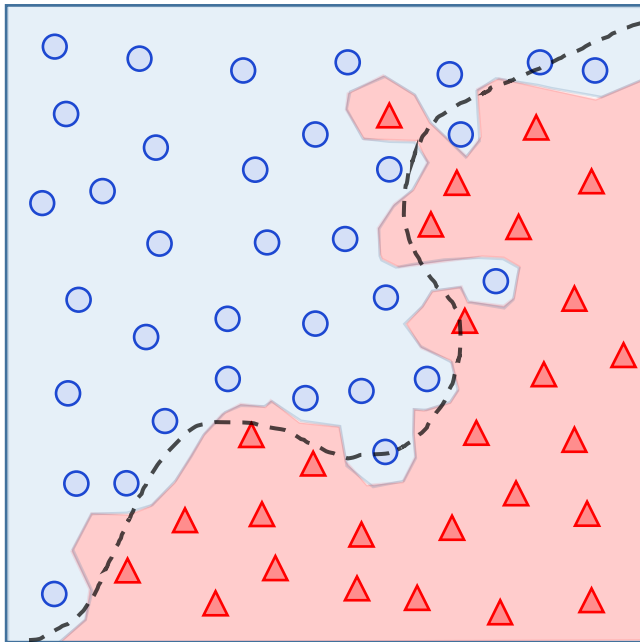
- ▶ Khoảng cách Euclide: đó là mô đun của sự khác biệt giữa hai vectơ vị trí



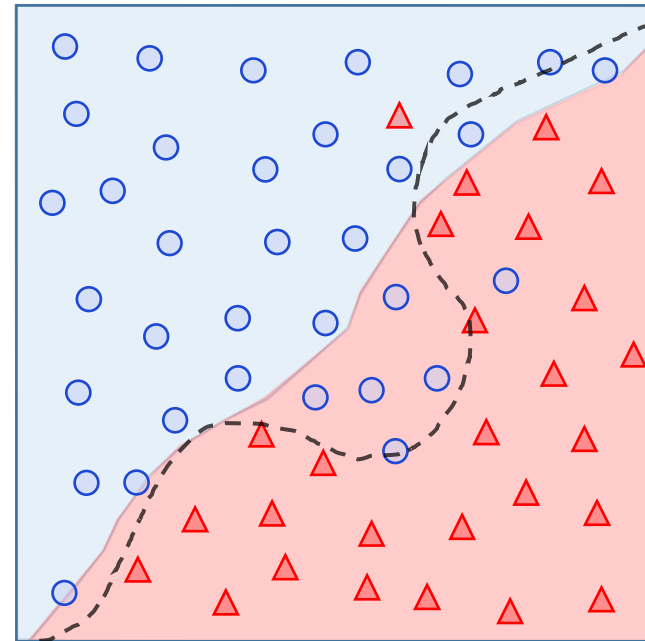
$$|X_1 - X_2| = \sqrt{(x_{11} - x_{21})^2 + (x_{12} - x_{22})^2 + \dots + (x_{1d} - x_{2d})^2}$$

Thuật toán KNN (4/4)

■ Khi k quá nhỏ, có thể tràn ra bên ngoài

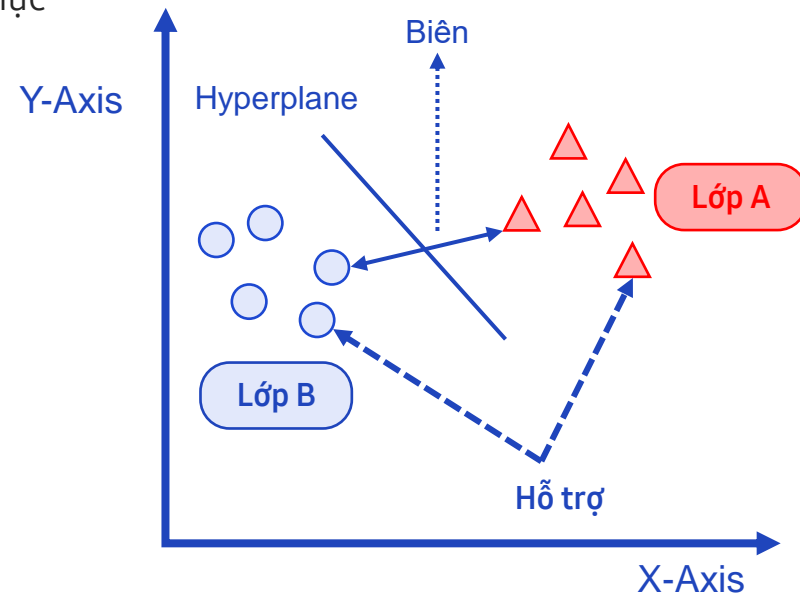


■ Khi k quá lớn, có thể xảy ra tình trạng thiếu



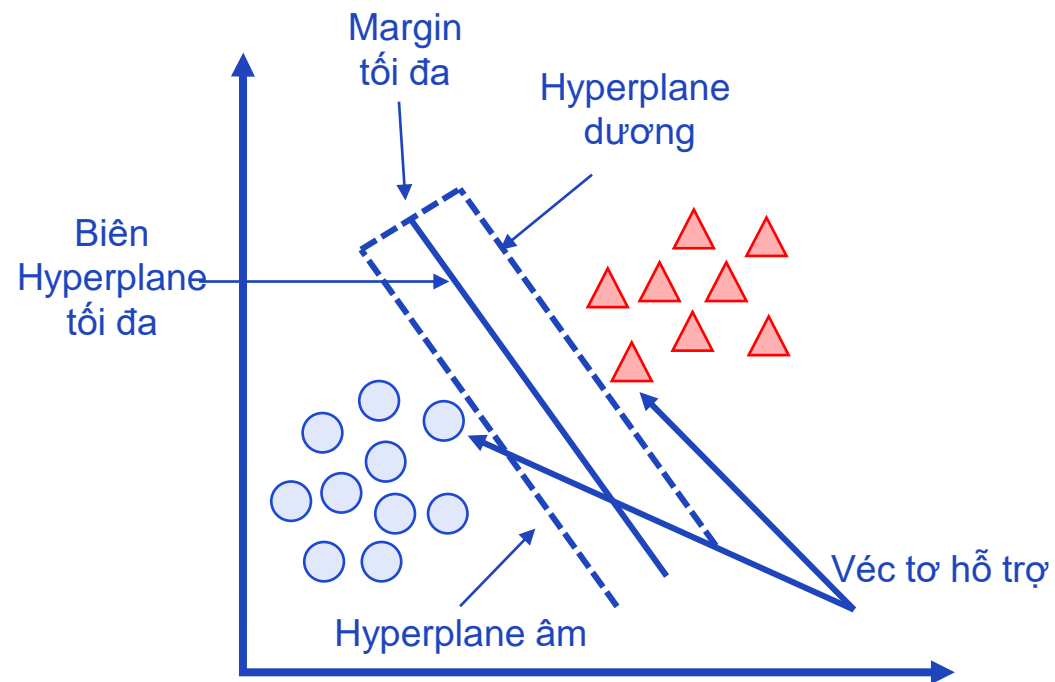
Thuật toán SVM (1/5)

- SVM (Máy vectơ hỗ trợ) là một tập hợp các phương pháp học có giám sát được sử dụng để phân loại, hồi quy và phát hiện giá trị ngoại lệ
- Để tạo đường ranh giới hoặc quyết định tốt nhất có thể phân tách không gian n chiều thành các lớp để chúng ta có thể dễ dàng đặt điểm dữ liệu mới vào đúng danh mục
 - ▶ Ranh giới quyết định tốt nhất này được gọi là siêu phẳng
- Yếu tố chính
 - ▶ Vectơ hỗ trợ
 - ▶ Hyperlane
 - ▶ Biên



Thuật toán SVM (2/5)

- SVM chọn các điểm/vectơ cực trị giúp tạo siêu phẳng



Thuật toán SVM (3/5)

I Hyperplane

- ▶ Đối với không gian cấu hình k chiều, hyperlane có số chiều $k-1$.
- ▶ Ví dụ,

Đối với không gian hai chiều, hyperlane là một đường phân giác có thể được tham số hóa như

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0$$

Không gian hai chiều được chia thành hai:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 > 0$$

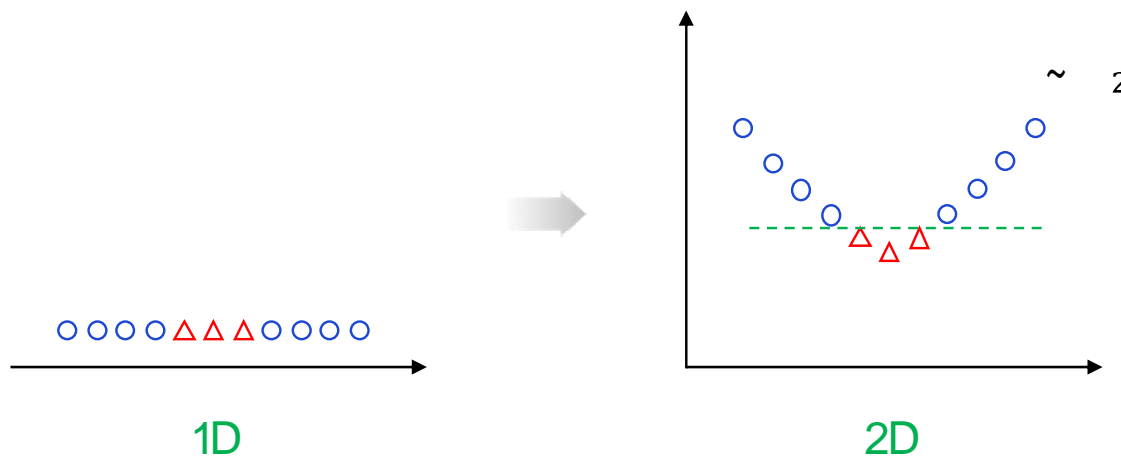
$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 < 0$$

Một quan sát sẽ thuộc về một trong hai \Rightarrow phân loại nhị phân!

Thuật toán SVM (4/5)

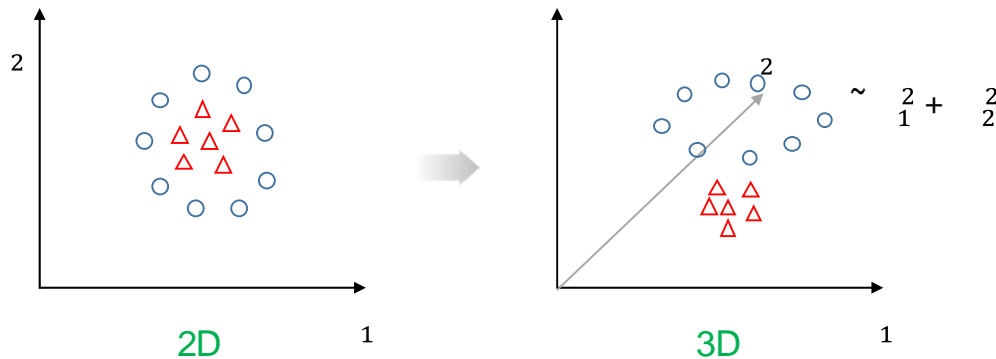
I Kernel

- ▶ Ánh xạ tới một chiều cao hơn bằng cách sử dụng các hàm “kernel”
- ▶ Các hàm hạt nhân giới thiệu một ranh giới phân loại phi tuyến tính hiệu quả
- ▶ Ví dụ, hạt nhân đa thức



Thuật toán SVM (5/5)

Kernel



Ưu điểm:

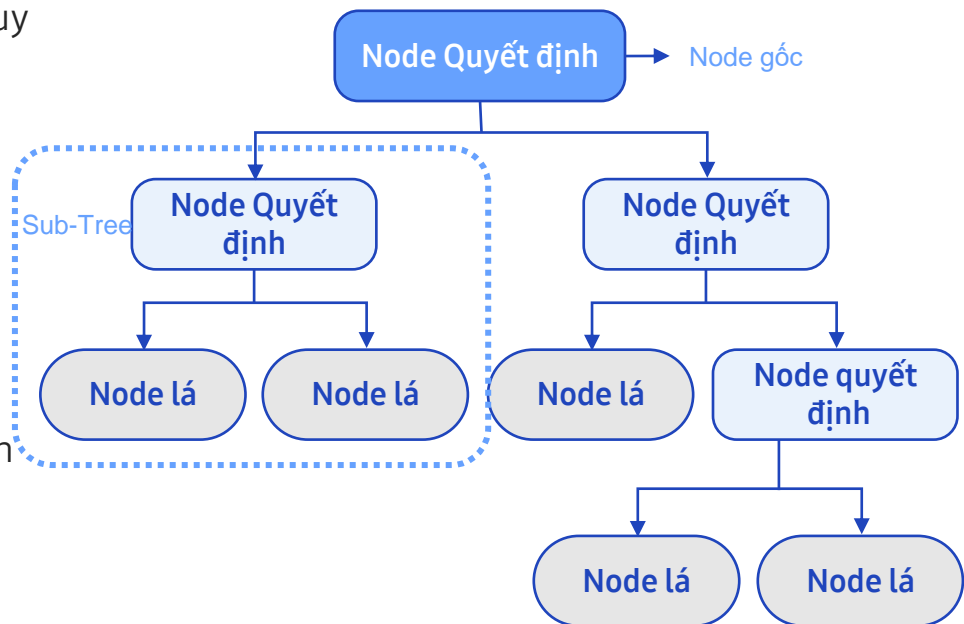
- ▶ Không nhạy cảm lắm với các phần bên ngoài
- ▶ Hiệu suất là tốt

Nhược điểm:

- ▶ Đào tạo tương đối chậm. Hoạt động kém đối với dữ liệu lớn
- ▶ Hạt nhân và bộ siêu tham số phải được tối ưu hóa cẩn thận

Thuật toán cây quyết định (1/2)

- Cây quyết định là một kỹ thuật học có giám sát có thể được sử dụng cho cả bài toán phân loại và hồi quy
- Có hai Node
 - ▶ Các Node quyết định được sử dụng để đưa ra bất kỳ quyết định nào và có nhiều nhánh
 - ▶ Các Node lá là đầu ra của các quyết định đó và không chứa bất kỳ nhánh nào nữa
- Các quyết định hoặc thử nghiệm được thực hiện trên cơ sở các tính năng của tập dữ liệu đã cho
- Cây quyết định chỉ cần đặt một câu hỏi và dựa trên câu trả lời (Có/Không), nó tiếp tục chia cây thành các cây con



Thuật toán cây quyết định (2/2)

I Ưu điểm:

- ▶ Trực quan và dễ hiểu
- ▶ Không có giả định về các biến
- ▶ Không cần mở rộng quy mô hoặc chuẩn hóa dữ liệu
- ▶ Không nhạy cảm với các ngoại lệ

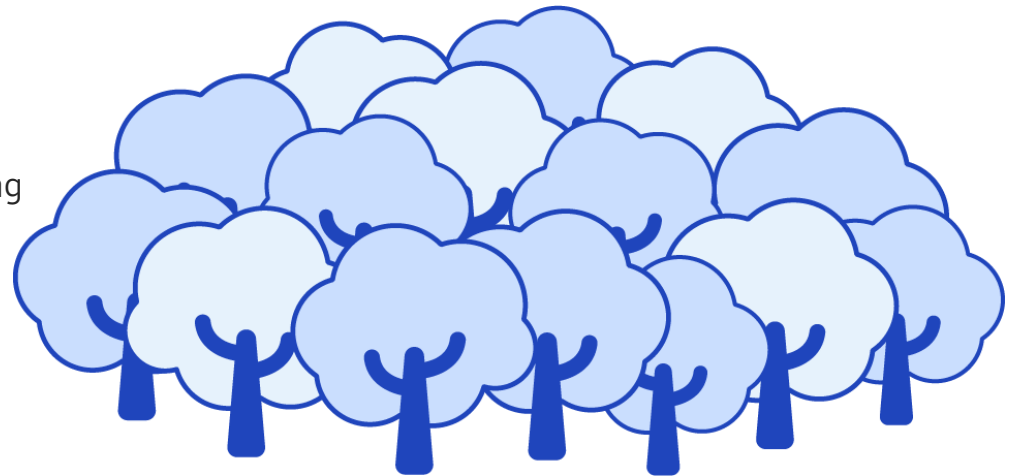
▶ Nhược điểm:

- ▶ Không mạnh mẽ ở dạng cơ bản nhất
- ▶ Dễ bị quá mức. Vì vậy, "cắt tỉa" thường được yêu cầu

Thuật toán rừng ngẫu nhiên (1/3)

I Rừng ngẫu nhiên

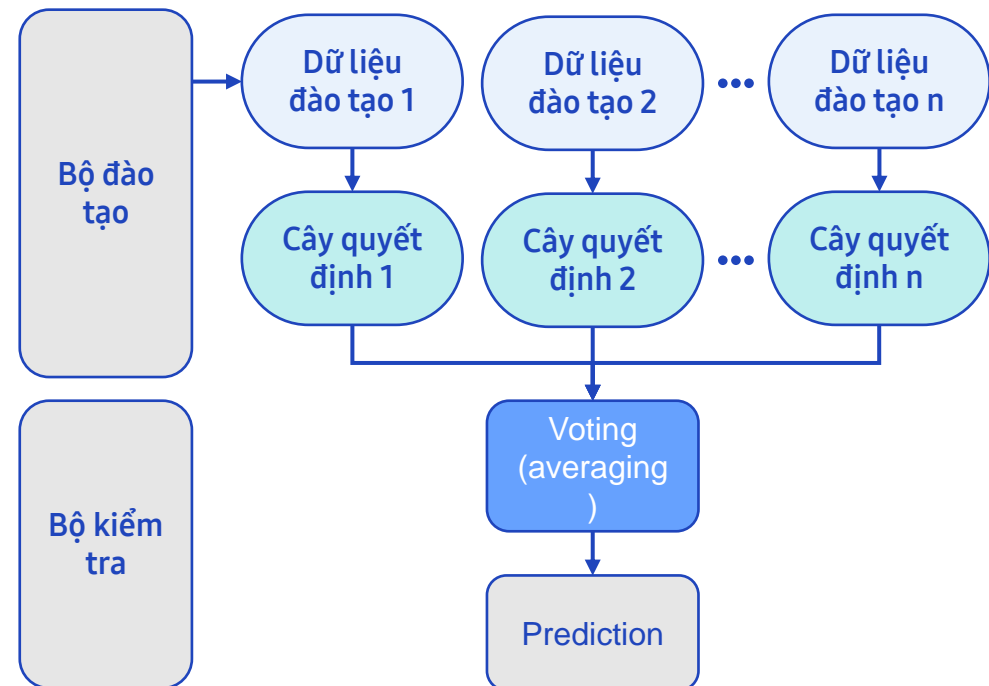
- ▶ Random Forest là một thuật toán học máy phổ biến thuộc kỹ thuật học có giám sát
- ▶ Một thuật toán tập hợp dựa trên cây
- ▶ Có thể được áp dụng để phân loại và hồi quy
- ▶ Cây được chọn ngẫu nhiên tạo thành một khu rừng
- ▶ Dự đoán được quyết định bởi đa số phiếu bầu



Thuật toán rừng ngẫu nhiên (2/3)

I Thuật toán Random Forest hoạt động như thế nào

- ① Tạo cây với các biến và quan sát được chọn ngẫu nhiên
- ② Chỉ giữ lại những thứ có tạp chất Gini (hoặc entropy) thấp nhất
- ③ Lặp lại từ bước 1) trong một số lần nhất định
- ④ Sử dụng các cây được thu thập trong bước đào tạo, chúng ta có thể đưa ra dự đoán theo đa số phiếu bầu



Thuật toán rừng ngẫu nhiên (3/3)

I Trường hợp sử dụng

- ▶ Ngân hàng: Ngành ngân hàng chủ yếu sử dụng thuật toán này để xác định rủi ro cho vay
- ▶ Y tế: Với sự trợ giúp của thuật toán này, có thể xác định xu hướng bệnh tật và nguy cơ mắc bệnh
- ▶ Sử dụng đất: Chúng tôi có thể xác định các khu vực sử dụng đất tương tự bằng thuật toán này
- ▶ Tiếp thị: Xu hướng tiếp thị có thể được xác định bằng thuật toán này

I Ưu điểm:

- ▶ Mạnh mẽ
- ▶ Vài giả định
- ▶ Ít hoặc không quan tâm đến vấn đề overfitting

I Nhược điểm:

- ▶ Đào tạo tốn thời gian

Thuật toán Naive Bayes (1/3)

I Thuật toán Naïve Bayes

- ▶ Thuật toán phân loại đa lớp đơn giản
- ▶ Giả sử sự độc lập giữa mọi cặp tính năng
- ▶ Trong một lượt, nó tính toán phân phối xác suất có điều kiện của từng tính năng đã cho nhãn
- ▶ Sau đó, nó áp dụng định lý Bayes để tính toán phân phối xác suất có điều kiện của quan sát nhãn đã cho và sử dụng nó để dự đoán
- ▶ Đó là một phương pháp phân loại dựa trên xác suất có điều kiện để tính toán xác suất tính năng mà dữ liệu thuộc về mỗi lớp.
- ▶ Đây là một ứng dụng đơn giản của định lý Bayes

Thuật toán Naive Bayes (2/3)

I Định lý Bayes

- ▶ $P(A)$ là xác suất giả thuyết A đúng. Điều này được gọi là xác suất trước
- ▶ $P(B)$ là xác suất của bằng chứng (không phụ thuộc vào giả thuyết)
- ▶ $P(B|A)$ là xác suất của bằng chứng cho rằng giả thuyết là đúng
- ▶ $P(A|B)$ là xác suất của giả thuyết cho rằng có bằng chứng

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

I Bộ phân loại Naive Bayes

- ▶ Naive Bayes là một loại phân loại sử dụng Định lý Bayes. Nó dự đoán xác suất thành viên cho từng lớp, chẳng hạn như xác suất mà bản ghi hoặc điểm dữ liệu đã cho thuộc về một lớp cụ thể

Thuật toán Naive Bayes (3/3)

I Các loại thuật toán Naive Bayes

- ▶ Multinomial Naive Bayes: Nó được sử dụng khi các đặc điểm của dữ liệu được biểu thị bằng số lần xuất hiện
- ▶ Bernoulli Naive Bayes: Một mô hình phù hợp khi các đặc trưng của dữ liệu được thể hiện bằng 0 hoặc 1
- ▶ Gaussian Naive Bayes: Nó phù hợp để tính toán xác suất có điều kiện với giả định rằng các giá trị của các tính năng được phân phối bình thường và để phân loại dữ liệu với các thuộc tính liên tục

I Ưu điểm:

- ▶ Trực quan, đơn giản và nhanh chóng
- ▶ Không nhạy cảm với tiếng ồn và ngoại lệ

I Nhược điểm:

- ▶ Giả sử rằng các tính năng là độc lập, điều này có thể không hoàn toàn đúng
- ▶ Không nằm trong số các thuật toán hoạt động tốt nhất

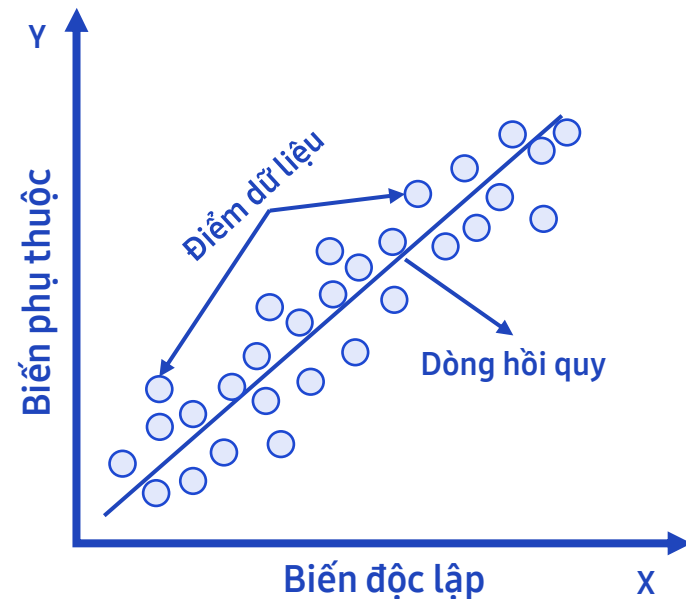
Hồi quy tuyến tính (1/5)

- Hồi quy tuyến tính là một trong những thuật toán học máy dễ nhất và phổ biến nhất
- Dự đoán giá trị của dữ liệu không quan sát được bằng cách tìm một hàm dựa trên dữ liệu được quan sát
 - ▶ Có một hoặc nhiều biến giải thích: X_1, X_2, \dots, X_k
 - ▶ Có một biến phản hồi: Y
 - ▶ Các biến X_i và Y được kết nối bởi một mối quan hệ tuyến tính:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_k X_k + \varepsilon$$

Biến phản ứng

Các biến giải thích



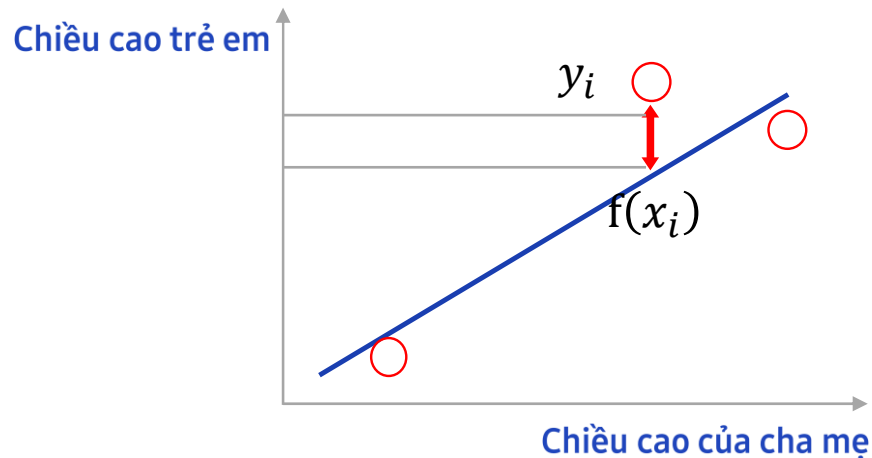
Hồi quy tuyến tính (2/5)

I Các loại hồi quy tuyến tính

- ▶ Hồi quy tuyến tính đơn giản: Dự đoán một biến kết quả với một biến dự đoán
- ▶ Hồi quy bội: dự đoán một biến kết quả với nhiều biến

I Bối cảnh lịch sử

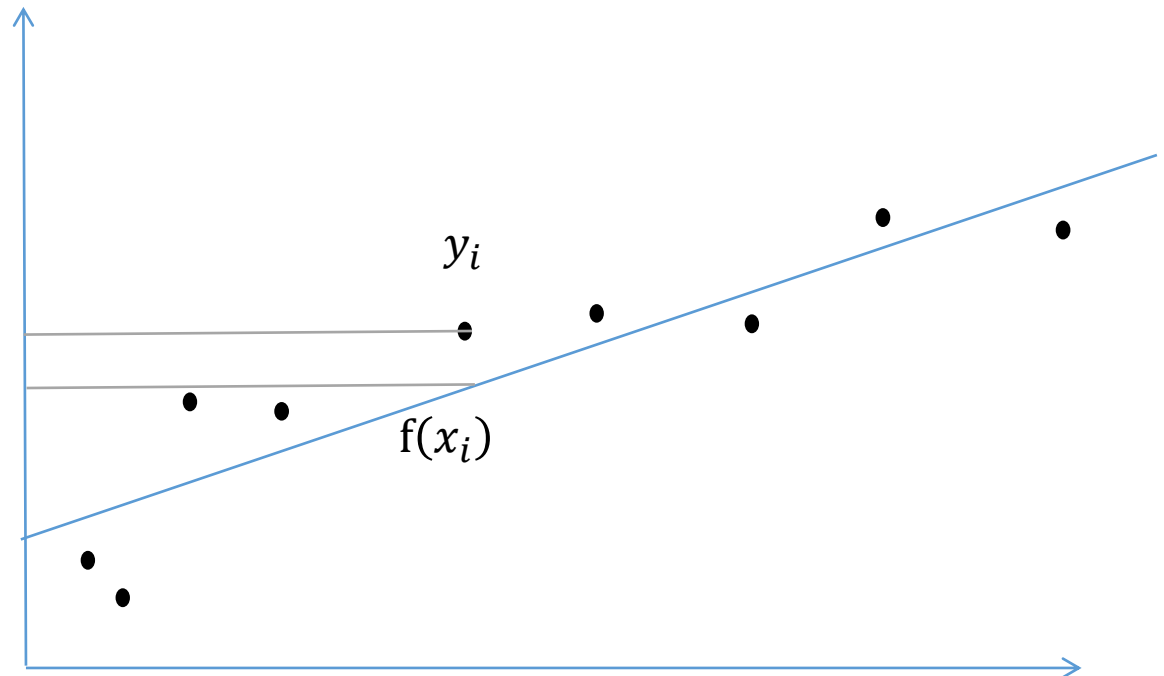
- ▶ Thuật ngữ hồi quy "được đặt ra bởi Francis Galton, nhà sinh vật học thế kỷ 19
- ▶ Chiều cao của con cháu có xu hướng hồi quy về phía trung bình



Hồi quy tuyến tính (3/5)

- I Mục tiêu của chức năng $f(x)$ là gì?
- I Để giảm thiểu sự khác biệt giữa nhãn thực tế và kết quả thu được từ mô hình của chúng ta
- I Muốn mô hình càng gần với dữ liệu càng tốt

- ▶ minimize $|y_i - f(x_i)|$
- ▶ minimize $(y_i - f(x_i))^2$
- ▶ minimize $\sum_{i=1}^n (y_i - f(x_i))^2$



Hồi quy tuyến tính (4/5)

I Tối ưu hóa hồi quy tuyến tính đơn giản

- ▶ Chọn β_0, β_1 sao cho tổng số lỗi bình phương (SSE) được giảm thiểu
- ▶ Hồi quy tuyến tính đơn giản là một vấn đề tối ưu hóa trong đó β_1, β_2 được chọn để giảm thiểu SSE

$$f(x_i) = \beta_0 + \beta_1 * x_i$$



Giảm thiểu

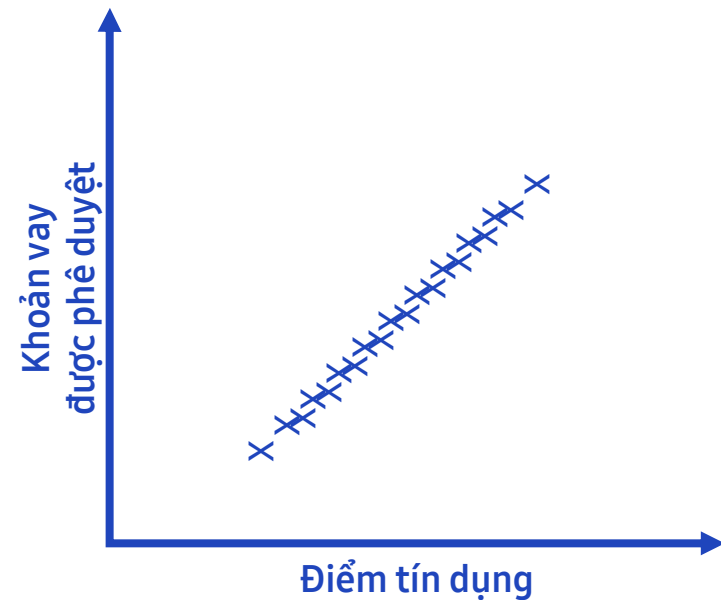
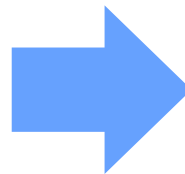
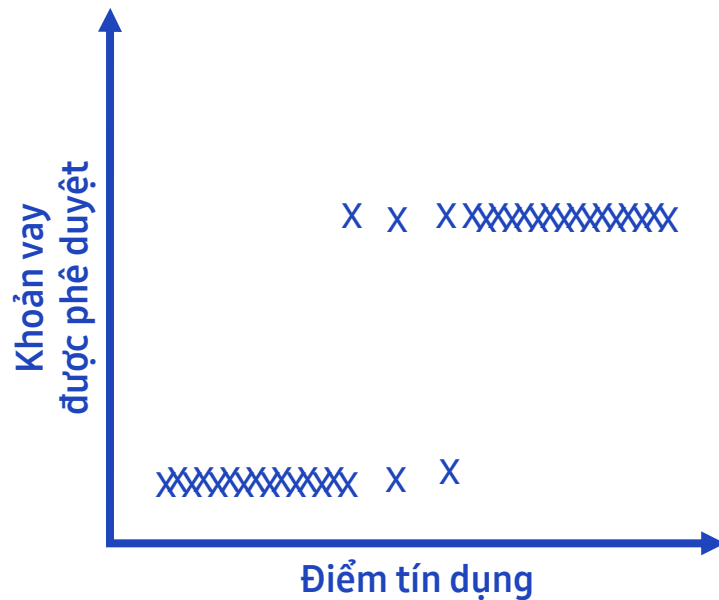
$$SSE(f) = \sum_{i=1}^n (y_i - f(x_i))^2$$

Hồi quy tuyến tính (5/5)

- | Ánh xạ dữ liệu thành một đường thẳng
 - ▶ Có thể có nhiều đường thẳng mô tả dữ liệu
- | Cách chọn một đường thẳng mô tả đúng nhất nó:
- | Phương pháp bình phương nhỏ nhất - Tìm cách giảm thiểu lỗi
- | Ưu điểm:
 - ▶ Nền tảng thống kê và toán học vững chắc.
 - ▶ Nguồn hiểu biết
 - ▶ Đào tạo nhanh
- | Nhược điểm:
 - ▶ Nhiều giả định: tuyến tính, tính quy tắc, tính độc lập của các biến giải thích, v.v.
 - ▶ Nhạy cảm với các ngoại lệ
 - ▶ Dễ dàng với đa colinearity

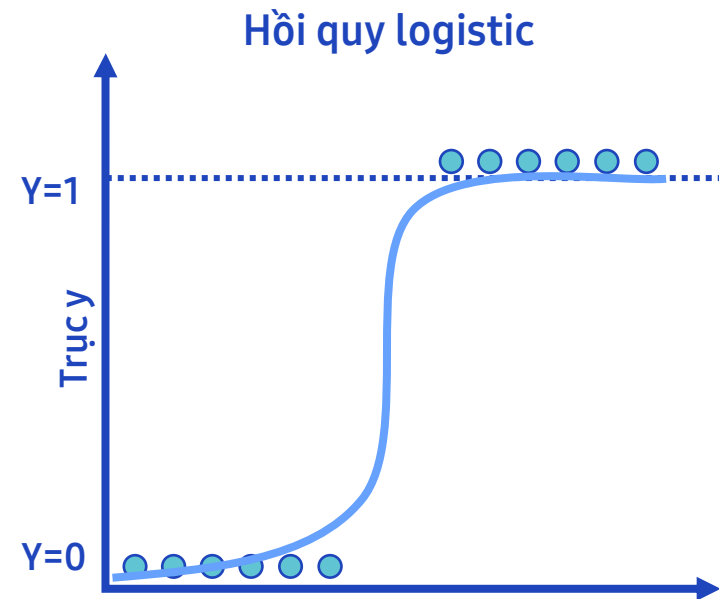
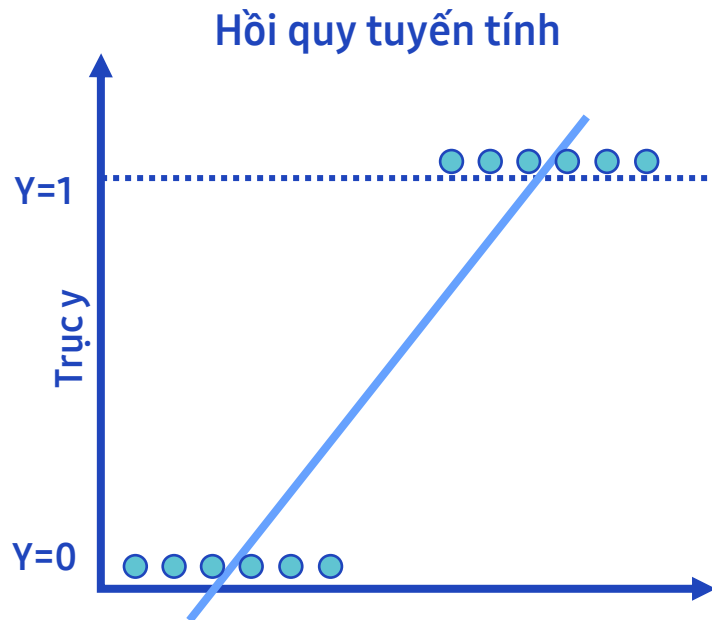
Động lực cho hồi quy logistic

- I Cố gắng chuyển đổi một phân tán có/không thành tuyến tính



Hồi quy logistic (1/6)

- Tại sao hồi quy logistic?
- Hồi quy tuyến tính so với hồi quy logistic



Hồi quy logistic (2/6)

- l fLogit = hàm tỷ lệ cược log
- l Tỷ lệ chênh lệch là xác suất thành công/xác suất thất bại
 - ▶ $\log [p/(1-p)]$
 - ▶ p = xác suất của một sự kiện xảy ra
 - ▶ $1 - p$ = xác suất của một sự kiện không xảy ra

$$\text{logit}(p) = \log\left(\frac{p}{1-p}\right) = \log(p) - \log(1-p) = -\log\left(\frac{1}{p} - 1\right).$$

- ▶ Bằng cách lấy logarit từ 0 đến ∞ , nó có thể được thực hiện thành $-\infty$ thành ∞
- l Các kết hợp tuyến tính của các biến X_i là Log Logit được ký hiệu ở đây là S

Hồi quy logistic (3/6)

I Xác suất và tỷ lệ cược

$$\text{Xác suất} = \frac{\text{Số lượng kết quả}}{\text{Tổng số kết quả khả thi}}$$

$$\text{Tỷ lệ cược} = \frac{\text{Prob}(xảy ra)}{\text{Prob}(không xảy ra)}$$

	Xác suất	Tỷ lệ cược
Cuộn 6 con xúc xắc	$\frac{1}{6} = 0.166667$	$\frac{1/6}{5/6} = \frac{1}{5} = 0.2$
Lật đầu trên đồng xu	$\frac{1}{2} = 0.5$	$\frac{1/2}{1/2} = \frac{1}{1} = 1$
Nhận và Ace trong thẻ	$\frac{4}{52} = 0.0769$	$\frac{4/52}{48/52} = \frac{4}{48} = 0.0833$

Hồi quy logistic (4/6)

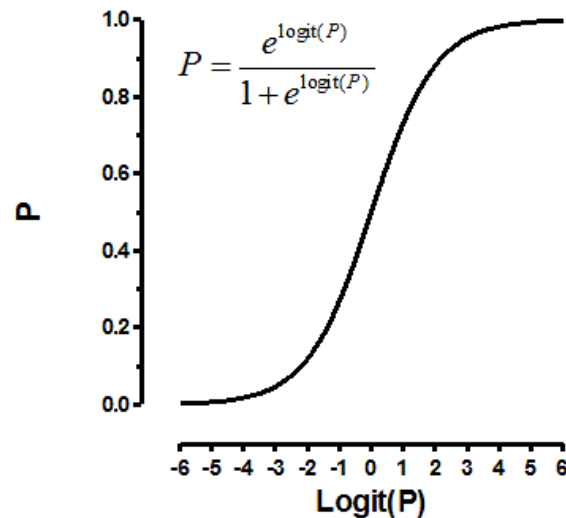
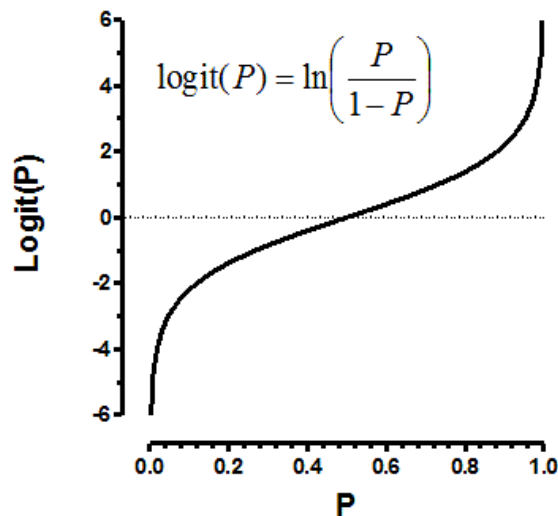
I Hồi quy logistic để phân phối Bernoulli

- ▶ Ước tính không xác định p Đối với bất kỳ sự kết hợp tuyến tính nhất định của các biến độc lập x_i .
- ▶ Nhãn y_i trong hồi quy logistic tuân theo phân phối Bernoulli có xác suất không xác định, p
- ▶ Phân phối Bernoulli chỉ là một trường hợp đặc biệt của phân phối nhị thức khi $n = 1$
- ▶ Thành công là 1 và thất bại là 0
- ▶ Xác suất thành công là p và xác suất thất bại là $1 - p$
- ▶ Chúng tôi cần một cách để liên kết các tính năng x_i với phân phối Bernoulli

Hồi quy logistic (5/6)

I Hàm logit

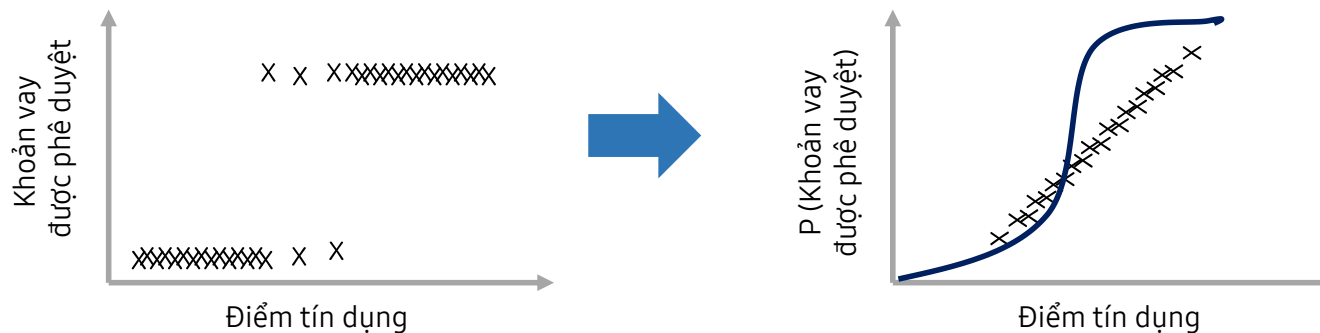
- ▶ Nhật ký tự nhiên của tỷ lệ chênh lệch là hàm logit
- ▶ Hàm logit là $\ln(odds)$ hoặc $\ln\left(\frac{p}{1-p}\right)$ hoặc $\ln(p) - \ln(1-p)$
- ▶ Và nghịch đảo cho thấy xác suất trên trục Y



Hồi quy logistic (6/6)

I Hồi quy logistic sử dụng logit nghịch đảo

- ▶ Hàm logit nghịch đảo cho phép chúng tôi chuyển đổi thành xác suất của nhãn



I Ưu điểm:

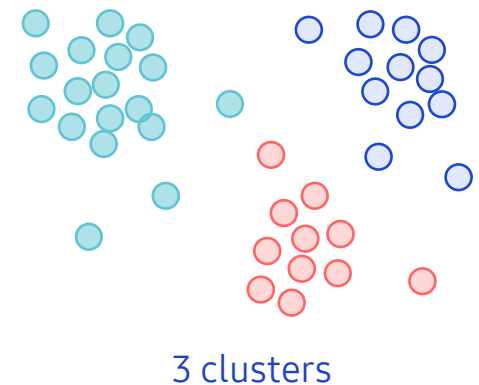
- ▶ Đơn giản và tương đối dễ thực hiện
- ▶ Đào tạo nhanh

I Nhược điểm:

- ▶ Không phải trong số các thuật toán phân loại chính xác nhất
- ▶ Giả sử rằng các biến giải thích là độc lập mà không có đa colinearity

Phân cụm K-Means (1/7)

- I Phân cụm K-Mean là một thuật toán học không được giám sát, nhóm tập dữ liệu không nhãn thành các cụm khác nhau
 - ▶ Cụm đề cập đến một tập hợp các điểm dữ liệu được tổng hợp với nhau vì một số điểm tương đồng nhất định
 - ▶ Ở đây K xác định số lượng cụm được xác định trước cần được tạo trong quá trình
- I Phân cụm K-means
 - ▶ Có một vectơ hơn các biến giải thích X_1, X_2, \dots, X_d
 - ▶ Không có biến phản hồi. Do đó, đây là một thuật toán học tập không giám sát
 - ▶ Nó sắp xếp bộ dữ liệu không được gắn nhãn thành nhiều cụm
- I Mục đích của phân cụm K-Means
 - ▶ Cụm các quan sát trong k cụm
 - ▶ Tìm các tâm (có nghĩa là cụm) đặc trưng cho các cụm



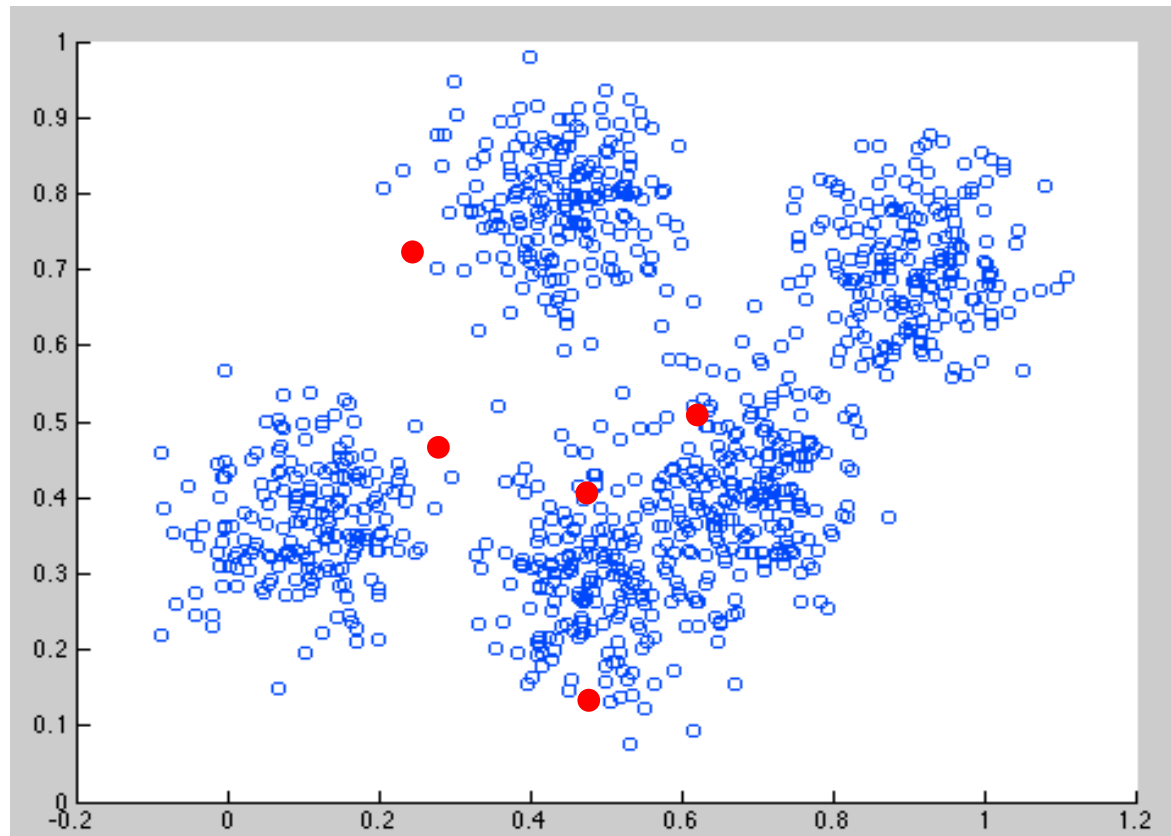
Phân cụm K-Means (2/7)

I Thuật toán K-Mean hoạt động như thế nào

- ① Chọn số K để xác định số lượng cụm
- ② Chọn điểm K hoặc Centroid tùy ý. (Nó có thể khác với bộ dữ liệu đầu vào)
- ③ Chỉ định tất cả các điểm dữ liệu cho Centroid gần nhất của họ. Nó sẽ tạo ra các cụm k được xác định trước
- ④ Tính phương sai và đặt một tâm mới của mỗi cụm
- ⑤ Lặp lại bước thứ ba. Tiếp tục phân công lại từng điểm dữ liệu cho cụm gần nhất mới nhất
- ⑥ Nếu có bất kỳ sự phân công lại xảy ra, sau đó chuyển sang bước 4; khác, kết thúc
- ⑦ Cuối cùng, mô hình đã sẵn sàng

Phân cụm K-Means (3/7)

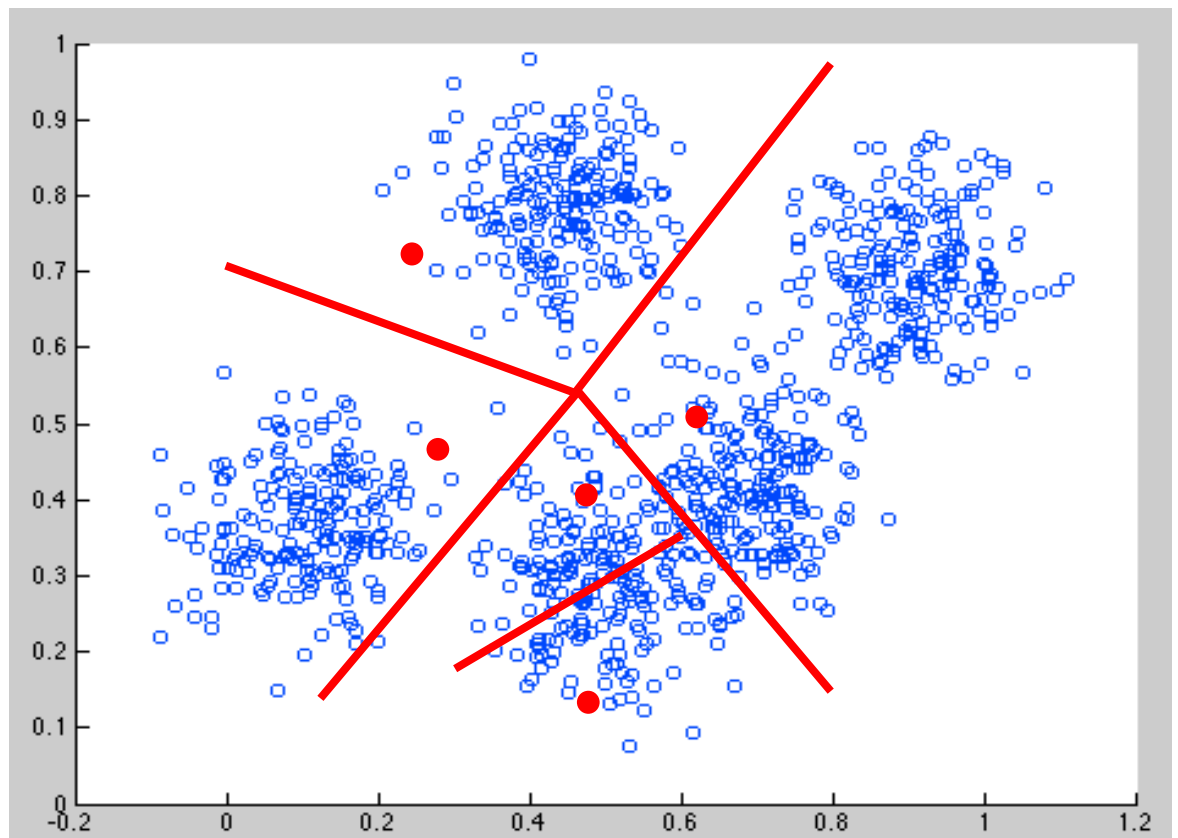
- I Điểm đại diện ban đầu
 - ▶ Số lượng cụm đầu vào k
 - ▶ Chỉ định các điểm đại diện k một cách ngẫu nhiên



Phân cụm K-Means (4/7)

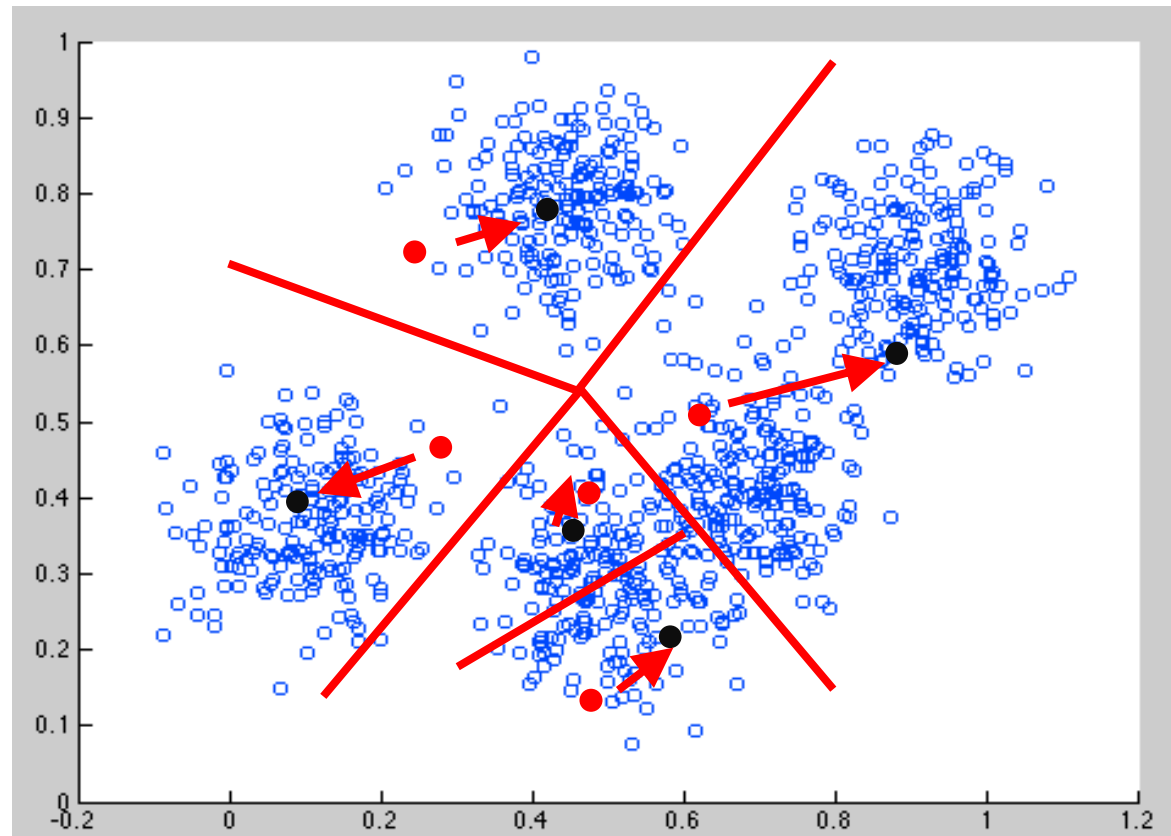
I Điểm đại diện gán

- ▶ Gán tất cả các điểm cho điểm đại diện gần nhất



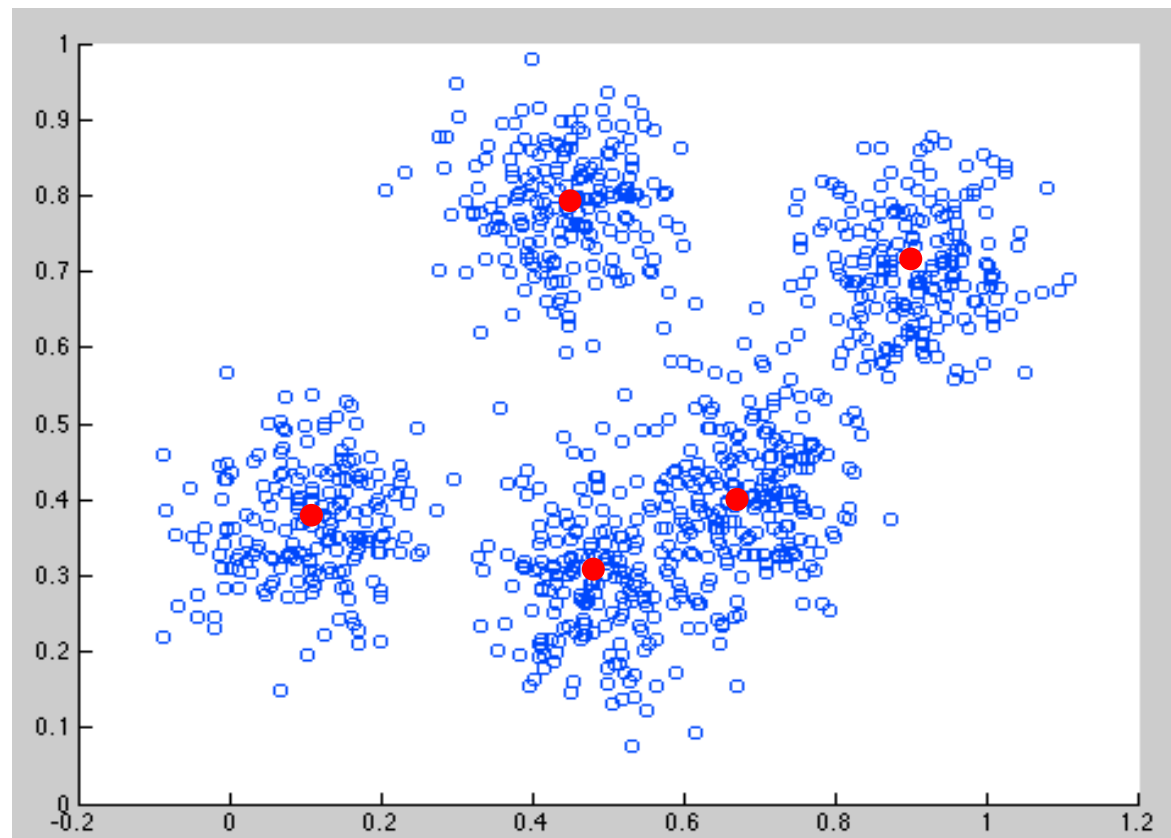
Phân cụm K-Means (5/7)

- I Di chuyển điểm đại diện đến giữa
 - Thay đổi các điểm đại diện cụm ở giữa các điểm của nó



Phân cụm K-Means (6/7)

- I Điểm đại diện ban đầu
 - ▶ Lặp lại cho đến khi hội tụ



Phân cụm K-Means (7/7)

- I Các mục nhóm dựa trên các đặc điểm tương tự
 - ▶ Nhập số lượng cụm (k) và khởi tạo ngẫu nhiên các trung tâm
 - ▶ Chỉ định mọi điểm trong dữ liệu cho một trong những trung tâm gần nhất
 - ▶ Đối với mỗi trung tâm cụm, hãy di chuyển trung tâm đến giữa cụm đó (trung bình)
 - ▶ Gán lại từng điểm cho trung tâm gần nhất mới nhất
 - ▶ Lặp lại cho đến khi hội tụ
- I Ưu điểm:
 - ▶ Giải thích trực quan về kết quả
 - ▶ Nhanh chóng và dễ dàng
- I Nhược điểm:
 - ▶ Nhạy cảm với tiếng ồn và ngoại lệ.
 - ▶ Ranh giới cụm chỉ có thể là tuyến tính

Phân tích thành phần chính (1/6)

I Mục đích của phân tích thành phần chính (PCA)

- ▶ PCA là một thuật toán làm giảm kích thước của dữ liệu chiều cao thành dữ liệu chiều thấp
- ▶ Chuyển đổi một tập hợp các biến tương quan thành một tập hợp các biến không tương thích khác
- ▶ Nhận một bộ vectơ trực giao mới (các thành phần chính hoặc PC)
- ▶ Đặt hàng các biến mới từ phương sai lớn nhất đến phương sai nhỏ nhất

I Ưu điểm:

- ▶ Dễ dàng tính toán
- ▶ Tăng tốc các thuật toán học máy khác
- ▶ Chống lại các vấn đề của dữ liệu chiều cao

I Nhược điểm:

- ▶ Khả năng diễn giải thấp của các thành phần chính

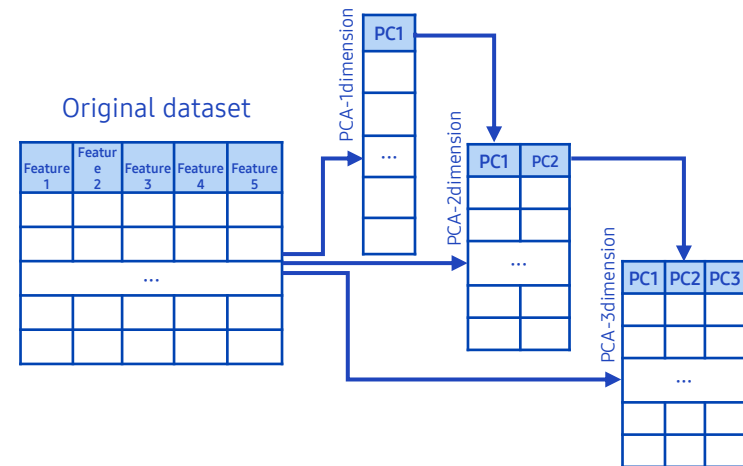
Phân tích thành phần chính (2/6)

I PCA được sử dụng trên nhiều trường hợp sử dụng

- ▶ Trực quan hóa dữ liệu đa chiều
- ▶ Nén thông tin
- ▶ Đơn giản hóa các quyết định kinh doanh phức tạp
- ▶ Làm rõ các quá trình khoa học phức tạp

I Quan niệm

- ▶ PCA tạo ra một biểu diễn chiều thấp của tập dữ liệu
- ▶ Nó tìm kiếm các máy bay gần với dữ liệu nhất, sau đó các điểm được chiếu lên nó
- ▶ Các thành phần độc lập với nhau và tối đa hóa phương sai
- ▶ Mỗi thành phần liên tiếp chứa ít phương sai hơn thành phần trước
- ▶ Khoảng cách từ các điểm đến hướng được giảm thiểu



Phân tích thành phần chính (3/6)

I Tải

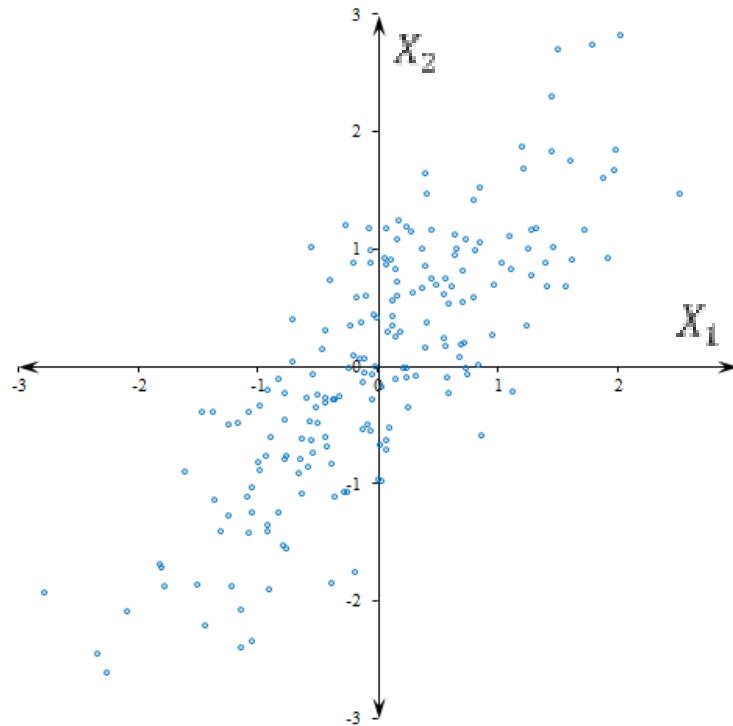
- ▶ Thành phần chính bình thường hóa (PC)
- ▶ Có nhiều vectơ tải như số lượng biến

I Phương sai σ^2 (hoặc độ lệch chuẩn σ)

- ▶ Các thành phần chính có các phương sai liên quan

I Điểm biến đổi

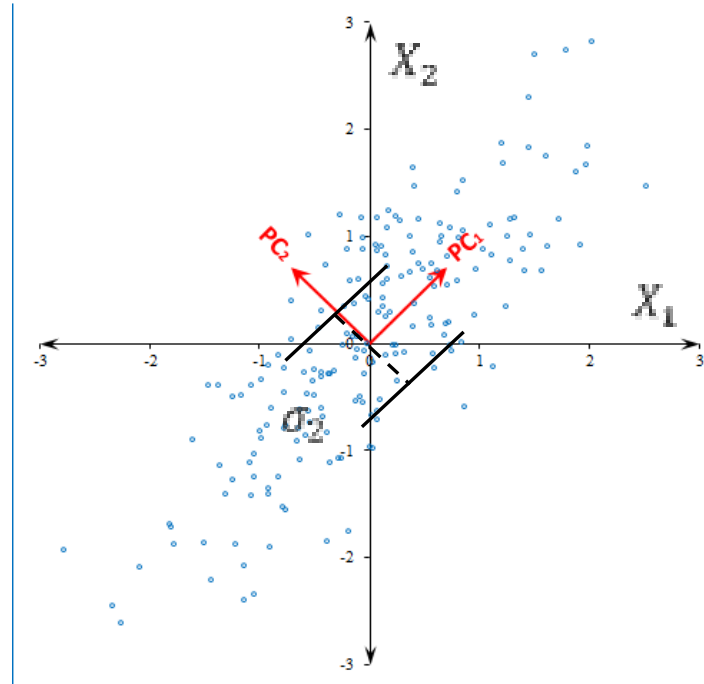
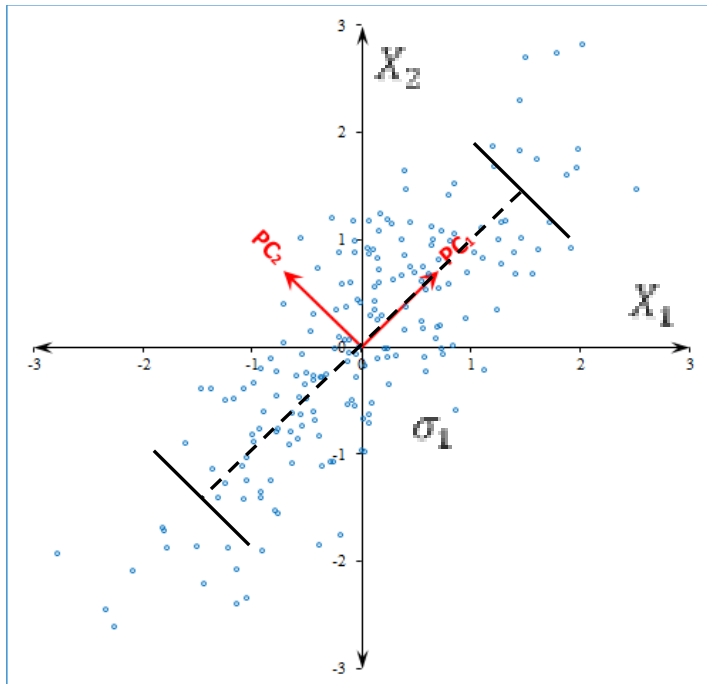
- ▶ Điểm thô (quan sát ban đầu) được biểu thị bằng cách sử dụng PCS làm trục tọa độ mới



A dataset with two variables (X_1 and X_2)

Phân tích thành phần chính (4/6)

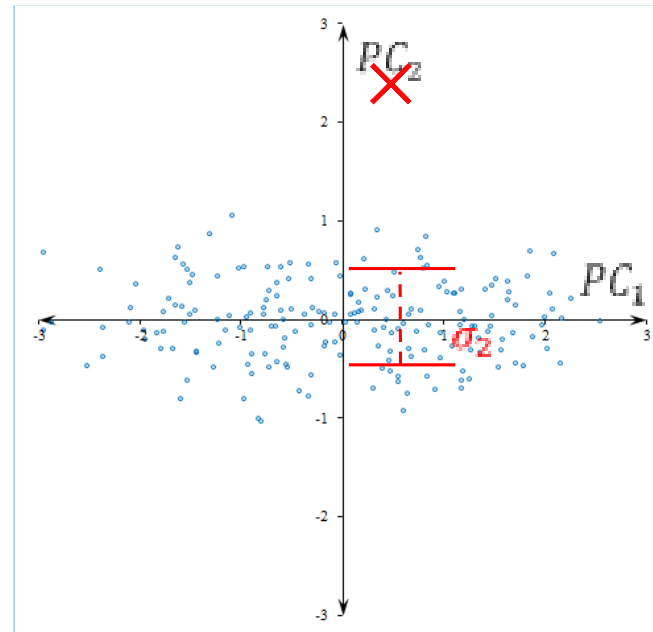
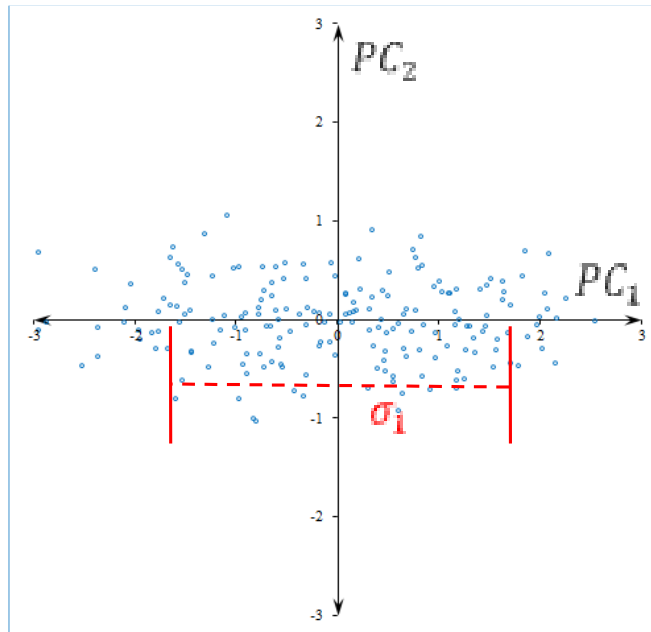
- PC_1 là hướng của phương sai lớn nhất, trong khi PC_2 là hướng của phương sai lớn nhất tiếp theo



Phân tích thành phần chính (5/6)

I Điểm số biến đổi

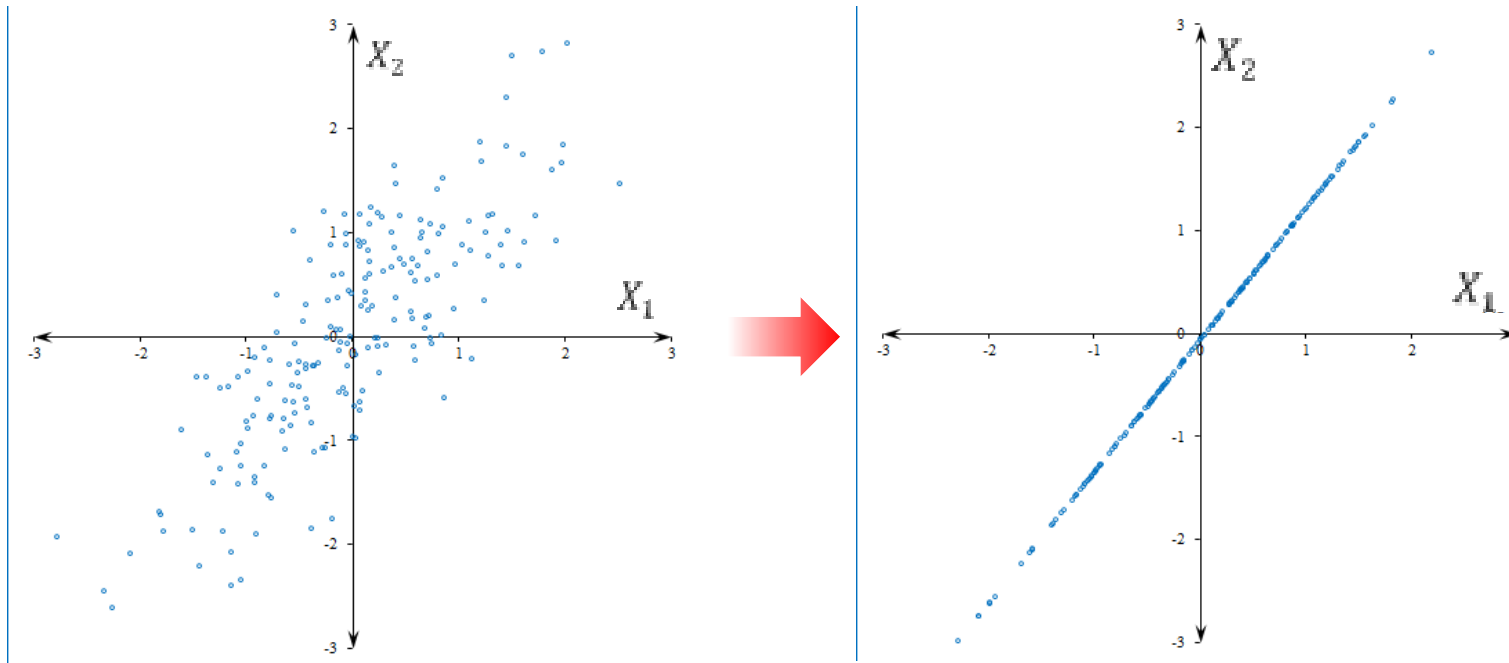
- ▶ Các quan sát có thể được biểu diễn bằng cách sử dụng PC_1 và PC_2 làm trục tọa độ mới



Phân tích thành phần chính (6/6)

I Giảm chiều

- ▶ Bây giờ, chúng ta có thể quay lại hệ tọa độ ban đầu và hiển thị “đầu vào chiều giảm”



Khái quát hóa và Quá khớp

I Khái quát hóa

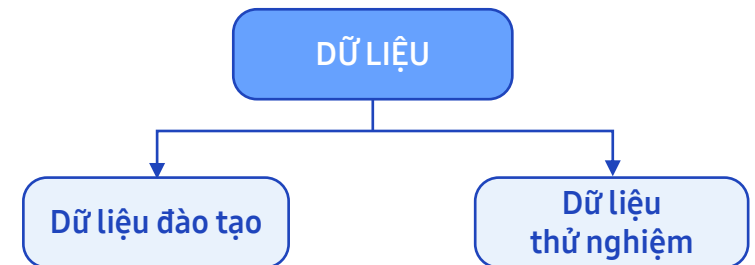
- ▶ Một thuộc tính cho phép một mô hình được tạo bằng cách học một bộ đào tạo, là một tập hợp con, được áp dụng cho các tập hợp con chưa được đào tạo khác

I Tập dữ liệu

- ▶ Tập phải đủ lớn để mang lại kết quả có ý nghĩa thống kê
- ▶ Tập thử nghiệm phải được chọn để có các đặc điểm giống như tập đào tạo

I Mô hình quá khớp

- ▶ Quá khớp bằng cách tạo một mô hình phức tạp hơn mức cần thiết
- ▶ Sự mất mát trong quá trình đào tạo là nhỏ, nhưng nó không đưa ra dự đoán tốt về dữ liệu mới



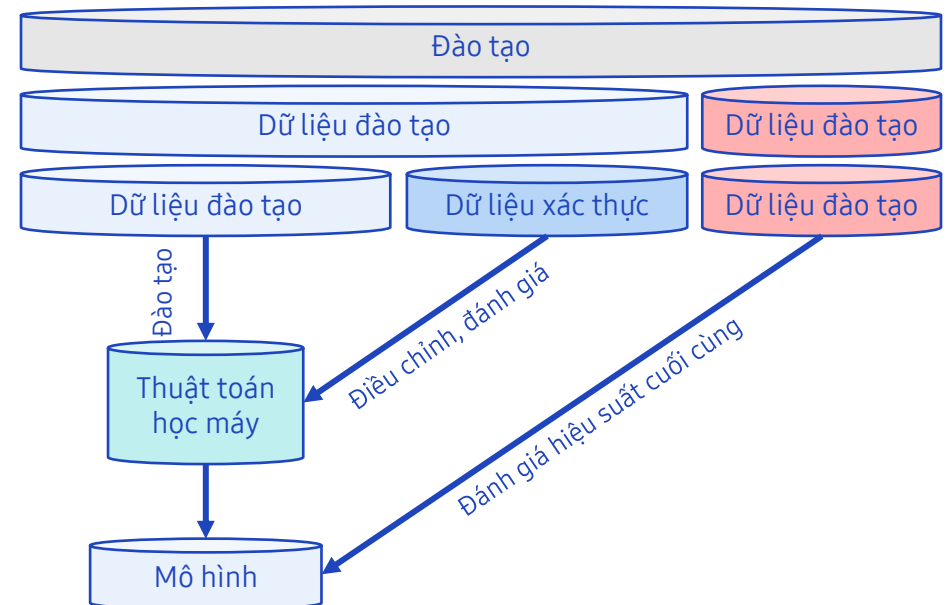
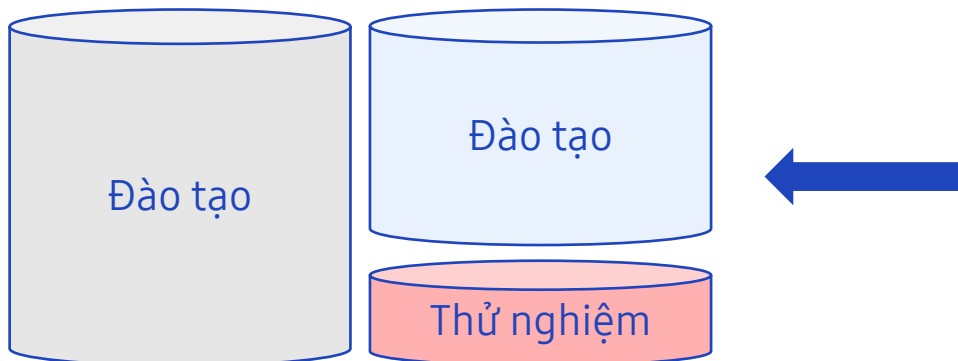
Tập dữ liệu

I Tập huấn luyện, tập xác nhận và tập kiểm tra

- ▶ Tập dữ liệu được chia thành tập huấn luyện, tập xác thực và tập kiểm tra theo cách để giảm khả năng khớp quá mức bằng cách chia tập dữ liệu thành các tập

I Chạy quy trình sau

- ▶ Đào tạo một mô hình với tập huấn luyện
- ▶ Đánh giá mô hình với bộ xác thực
- ▶ Điều chỉnh mô hình theo kết quả xác thực đã đạt



Xác thực chéo

- I Xác thực chéo là cách phổ biến nhất để đánh giá các mô hình
- I Quy trình xác thực chéo
 - ▶ Chia tập dữ liệu thành N “tập” có kích thước xấp xỉ bằng nhau
 - ▶ Huấn luyện thuật toán trên N-1 nếp gấp và sử dụng tập cuối cùng để tính thước đo đánh giá
 - ▶ Lặp lại N lần, chỉ định mỗi 1 trong số N lần gấp là lần thử
 - ▶ Báo cáo giá trị trung bình và độ lệch chuẩn của phép đo đánh giá qua N lần
 - ▶ Thông thường N ở đâu đó trong khoảng từ 5 đến 10

Dữ liệu thử nghiệm	Dữ liệu đào tạo	Dữ liệu đào tạo	Dữ liệu đào tạo	Dữ liệu đào tạo	Dữ liệu đào tạo	Dữ liệu đào tạo	Dữ liệu đào tạo	Dữ liệu đào tạo	Dữ liệu đào tạo
--------------------	-----------------	-----------------	-----------------	-----------------	-----------------	-----------------	-----------------	-----------------	-----------------

Xác thực chéo lồng nhau (1/2)

I Quy trình xác thực chéo lồng nhau

- ① Chia tập dữ liệu thành N “nếp gấp” có kích thước xấp xỉ bằng nhau
- ② Dành 1 lần cho lần kiểm tra và N-1 cho lần đào tạo
- ③ Dự trữ 1 trong số các tập dữ liệu huấn luyện để xác nhận
- ④ Với $\lambda = \lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5, \dots$ huấn luyện thuật toán trên N-2 tập đào tạo còn lại
- ⑤ Lặp lại từ bước 3, N-1 lần, xoay tập xác thực trong N-1 tập đào tạo
- ⑥ Chọn λ giảm thiểu lỗi đào tạo trung bình trên các tập đào tạo
- ⑦ Sử dụng x đó để đánh giá trên bộ kiểm tra
- ⑧ Lặp lại từ bước 2, N-1 lần, bằng cách xoay tập thử nghiệm
- ⑨ Báo cáo giá trị trung bình và độ lệch chuẩn của phép đo đánh giá qua N lần

Xác thực chéo lồng nhau (2/2)

- I Đối với 10 lần và 5 giá trị λ khác nhau được kiểm tra
 - ▶ 10 (cho mỗi lần kiểm tra) x 9 (cho mỗi lần xác thực) x 5 (cho mỗi giá trị λ)
 - ▶ 450 lần chạy thử
- I Nếu có nhiều thông số để kiểm tra, 450 cho mỗi thông số cần thiết
- I Xác thực chéo lồng nhau có thể trở nên rất tốn kém về mặt tính toán

Dữ liệu đào tạo	Dữ liệu đào tạo	Dữ liệu đào tạo	Dữ liệu đào tạo	Dữ liệu đào tạo	Dữ liệu đào tạo	Dữ liệu đào tạo	Dữ liệu đào tạo	Dữ liệu xác thực	Dữ liệu thử nghiệm
-----------------	-----------------	-----------------	-----------------	-----------------	-----------------	-----------------	-----------------	------------------	--------------------

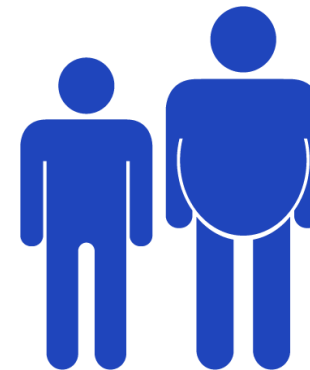
Quá khớp & Chưa khớp

I Quá khớp

- ▶ Quá khớp xảy ra khi mô hình được tạo cố gắng khớp quá chặt chẽ với dữ liệu đào tạo
- ▶ Mô hình này chỉ tốt cho dữ liệu huấn luyện và có thể sẽ tạo ra kết quả rất kém với bất kỳ dữ liệu thử nghiệm mới nào
- ▶ Về cơ bản là vô dụng

I Chưa khớp

- ▶ Chưa khớp xảy ra khi mô hình được tạo ra có quá nhiều sai lệch so với mô hình dự đoán tốt
- ▶ Mặc dù mô hình này có thể không hoàn toàn vô dụng, nhưng sai số quá lớn



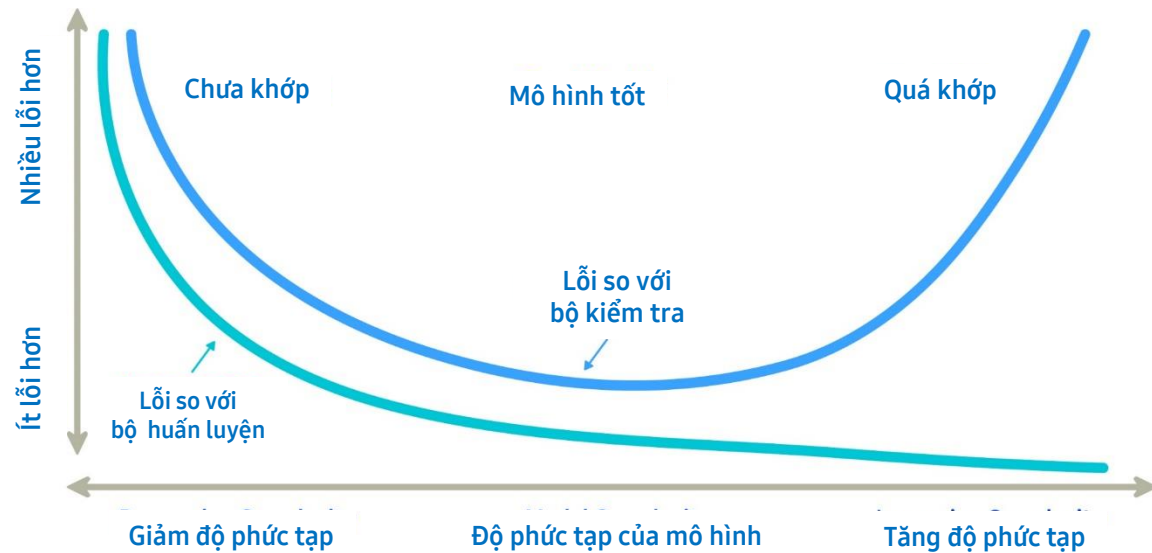
Chưa khớp

VS

Quá khớp

Cú pháp Razor của Occam

- Trong số các giả thuyết cạnh tranh, nên chọn giả thuyết có ít giả định nhất
- William of Ockham (c. 1287–1347), là một tu sĩ dòng Phanxicô người Anh, nhà triết học kinh viện và nhà thần học
- Trong bối cảnh của chúng tôi, chúng tôi có thể phát biểu lại điều này như sau:
- Các mô hình học máy tốt nhất là các mô hình đơn giản phù hợp với dữ liệu
- Lựa chọn mô hình tốt nhất



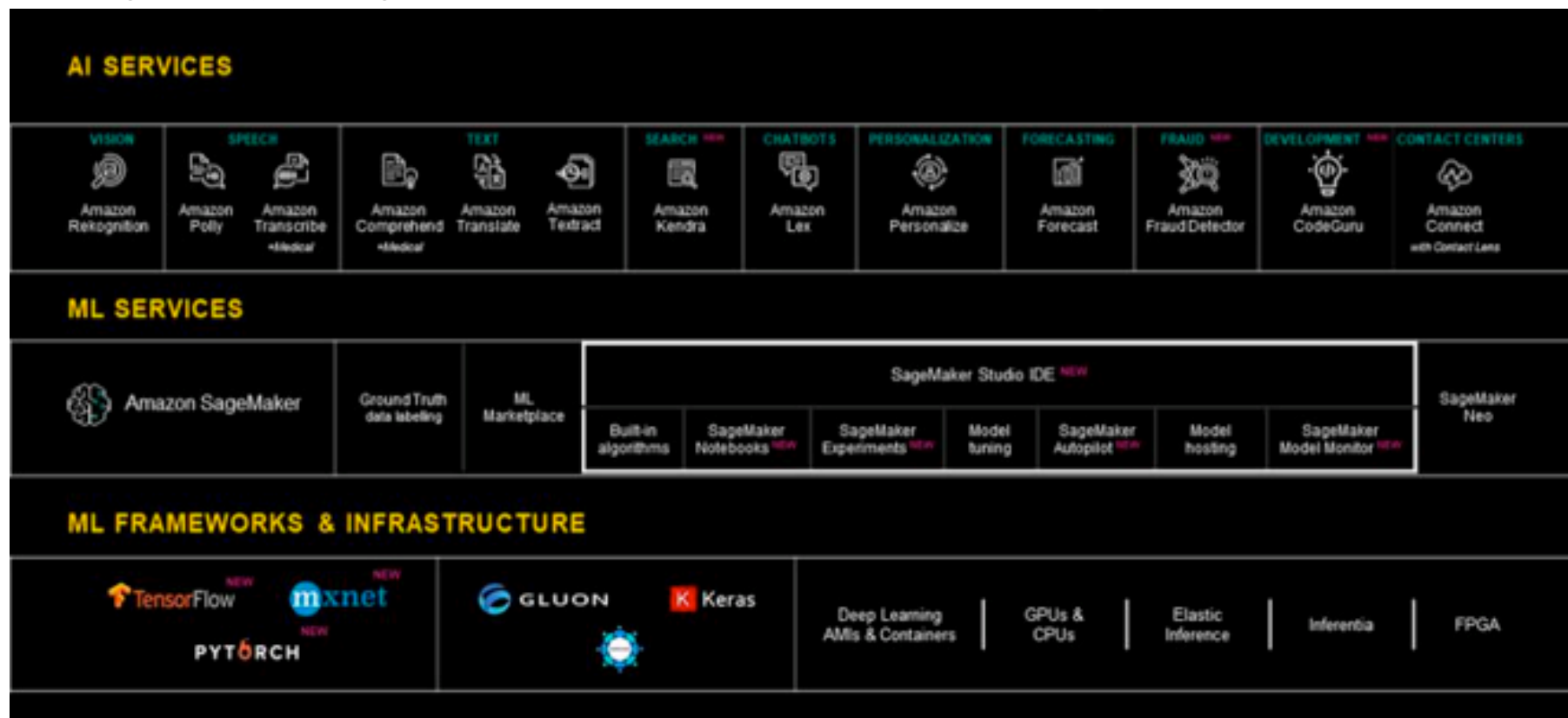
Bài 1

Học máy & Mô Hình

- | 1.1. Kiến thức cơ bản về học máy
- | **1.2. Học máy trong đám mây công cộng**
- | 1.3. Đường ống Apache Spark ML

Ngăn học máy AWS

Khả năng học máy mở rộng



SageMaker là gì?

- Amazon SageMaker là dịch vụ học máy được quản lý hoàn toàn
- Dễ dàng xây dựng và đào tạo các mô hình học máy, sau đó trực tiếp triển khai chúng vào môi trường được lưu trữ sẵn sàng sản xuất
- Tất cả các thành phần được sử dụng cho ML được cung cấp trong một bộ công cụ duy nhất
 - Dán nhãn
 - Chuẩn bị dữ liệu
 - Kỹ thuật tính năng
 - Phát hiện sai lệch thống kê
 - Tự động học máy
 - Đào tạo
 - Điều chỉnh
 - Lưu trữ
 - Giám sát và quy trình làm việc

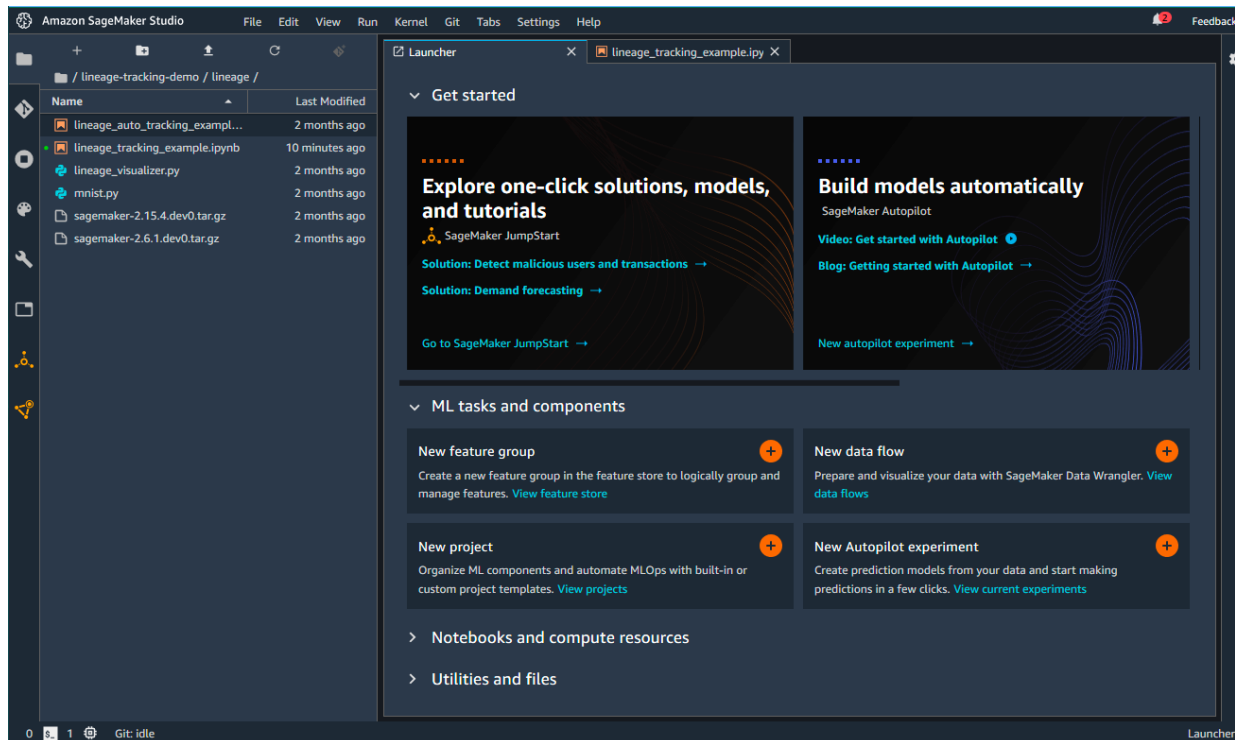


Studio SageMaker (1/3)

- | Amazon SageMaker Studio là môi trường phát triển tích hợp (IDE) dựa trên web dành cho máy học, cho phép người dùng xây dựng, đào tạo, gỡ lỗi, triển khai và giám sát các mô hình máy học
- | SageMaker Studio cung cấp tất cả các công cụ cần thiết để đưa các mô hình từ thử nghiệm sang sản xuất đồng thời tăng năng suất
- | Trong một giao diện trực quan thống nhất duy nhất:
 - ▶ Viết và thực thi mã trong sổ ghi chép Jupyter
 - ▶ Xây dựng và huấn luyện các mô hình học máy
 - ▶ Triển khai các mô hình và theo dõi hiệu suất dự đoán của chúng
 - ▶ Theo dõi và gỡ lỗi các thử nghiệm học máy

Studio SageMaker (2/3)

- | Studio Amazon SageMaker mở rộng giao diện JupyterLab
- | Studio thêm nhiều bổ sung vào giao diện



Studio SageMaker (3/3)

I Thanh bên trái

- ▶ Khi bạn di chuột qua một biểu tượng, chú giải công cụ sẽ hiển thị tên biểu tượng. Khi bạn chọn một biểu tượng, trình duyệt tệp và tài nguyên sẽ hiển thị chức năng được mô tả

I Trình duyệt tệp và tài nguyên









- ▶ Trình duyệt tệp và tài nguyên hiển thị danh sách sổ ghi chép, thử nghiệm, bản dùng thử, thành phần bản dùng thử và điểm cuối của bạn

I Khu vực làm việc chính

- ▶ Khu vực làm việc chính bao gồm nhiều tab chứa sổ ghi chép và thiết bị đầu cuối đang mở của bạn cũng như thông tin chi tiết về các thử nghiệm và điểm cuối của bạn

I Cài đặt

- ▶ Khung cài đặt cho phép điều chỉnh các thuộc tính của bảng và biểu đồ

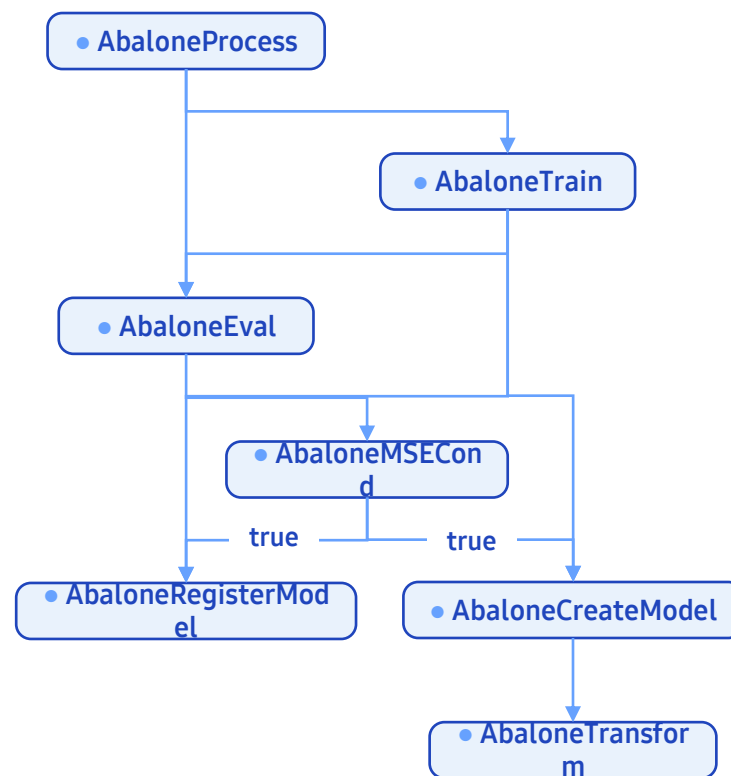
	Trình duyệt tệp
	Thiết bị đầu cuối và nhân đang hoạt động
	Git
	Câu lệnh
	Công cụ sổ tay
	Tab đang mở
	Khởi động SageMaker
	Thành phần và đăng ký SageMaker

Tự động hóa MLOps với SageMaker

- | Tạo các giải pháp ML toàn diện với CI/CD bằng cách sử dụng các dự án SageMaker
- | Sử dụng các dự án SageMaker để tạo giải pháp ML ops nhằm sắp xếp và quản lý:
 - ▶ Chuẩn bị dữ liệu và kỹ thuật tính năng
 - ▶ Mô hình đào tạo
 - ▶ Đánh giá mô hình
 - ▶ Triển khai mô hình
 - ▶ Theo dõi và cập nhật các mô hình

Đường ống SageMaker

- Một loạt các bước được kết nối với nhau được xác định bởi định nghĩa đường dẫn JSON
- Định nghĩa quy trình này mã hóa một quy trình sử dụng biểu đồ tuần hoàn có hướng (DAG)
- DAG cung cấp thông tin về các yêu cầu và mối quan hệ giữa từng bước của quy trình
- Cấu trúc của DAG của đường ống được xác định bởi sự phụ thuộc dữ liệu giữa các bước



Trình sắp xếp dữ liệu

- I Một tính năng của SageMaker Studio cung cấp giải pháp đầu cuối để nhập, chuẩn bị, chuyển đổi, làm nổi bật và phân tích dữ liệu
 - ▶ Nhập – Kết nối và nhập dữ liệu từ Amazon Simple Storage Service (Amazon S3), Amazon Athena (Athena) và Amazon Redshift
 - ▶ Luồng dữ liệu – Tạo luồng dữ liệu để xác định một loạt các bước chuẩn bị dữ liệu ML
 - ▶ Chuyển đổi – Làm sạch và chuyển đổi tập dữ liệu bằng cách sử dụng các công cụ định dạng dữ liệu số chuẩn như chuỗi, vectơ và số
 - ▶ Phân tích – Phân tích các tính năng trong tập dữ liệu tại bất kỳ điểm nào trong quy trình
 - ▶ Xuất – Trình sắp xếp dữ liệu cung cấp các tùy chọn xuất sang các dịch vụ SageMaker khác, bao gồm các công việc Trình sắp xếp dữ liệu, Kho tính năng và quy trình, giúp dễ dàng tích hợp luồng chuẩn bị dữ liệu vào quy trình ML.

Sự thật cơ bản đến dữ liệu nhãn

- Amazon SageMaker Ground Truth là dịch vụ ghi nhãn dữ liệu được quản lý hoàn toàn giúp dễ dàng xây dựng bộ dữ liệu đào tạo có độ chính xác cao cho học máy
- Tận dụng quy trình ghi nhãn dữ liệu tùy chỉnh hoặc tích hợp thông qua bảng điều khiển SageMaker Ground Truth để gán nhãn dữ liệu
- Giúp xây dựng bộ dữ liệu đào tạo chất lượng cao cho các mô hình máy học
- Nó hoạt động như thế nào



Amazon
SageMaker



Dữ liệu thô



Dán nhãn



Người dán nhãn



Dán nhãn hỗ trợ



Tập dữ liệu đào tạo chính xác
đã sẵn sàng để sử dụng

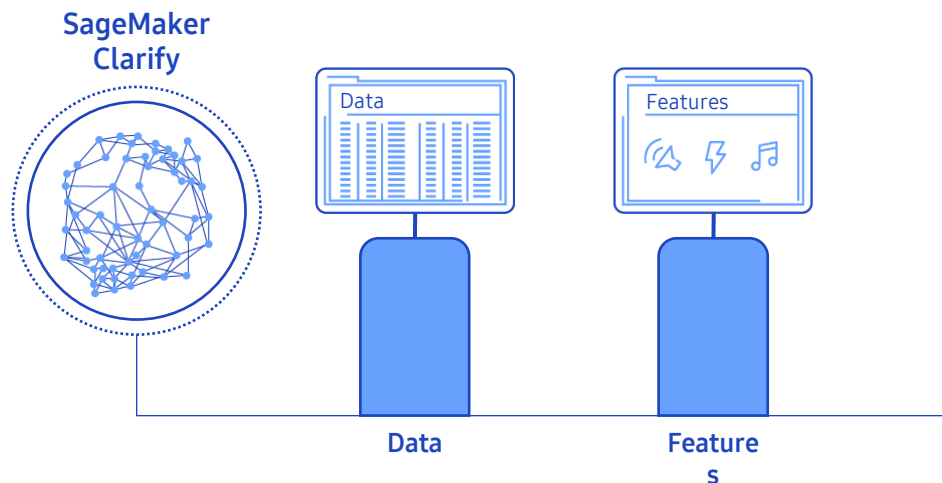
Theo dõi và làm rõ quy trình ML

I Theo dõi dòng dữ liệu ML

- ▶ Theo dõi dòng quy trình công việc học máy

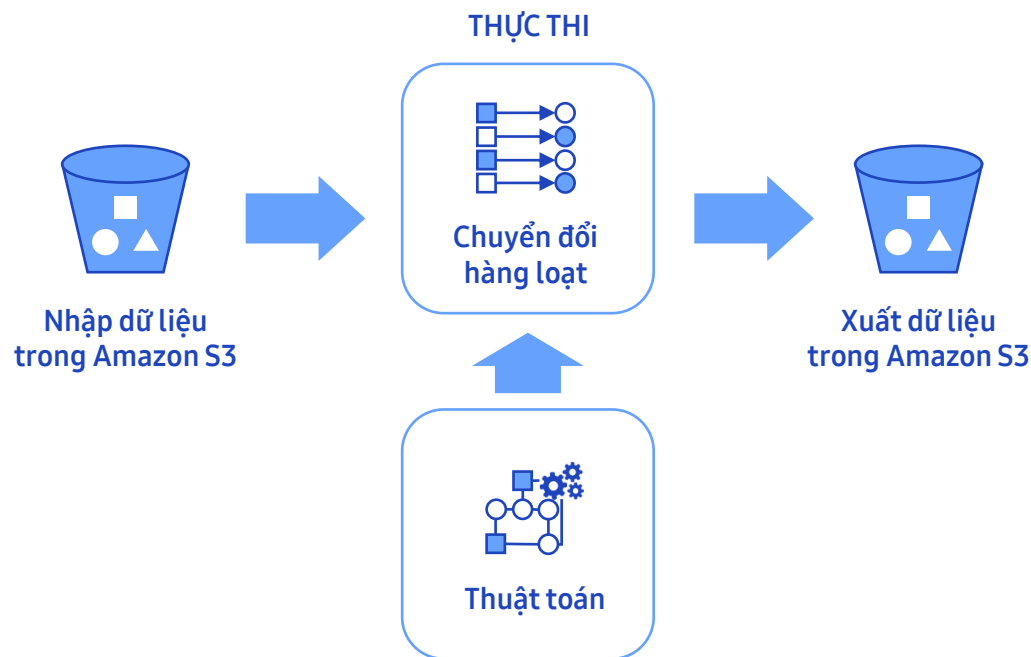
I Làm rõ

- ▶ Cải thiện các mô hình học máy bằng cách phát hiện sai lệch tiềm ẩn và giúp giải thích các dự đoán mà các mô hình đưa ra



Chuyển đổi hàng loạt SageMaker

- Bộ dữ liệu tiền xử lý, chạy suy luận khi chúng tôi không cần điểm cuối liên tục và liên kết bản ghi đầu vào với suy luận để hỗ trợ diễn giải kết quả



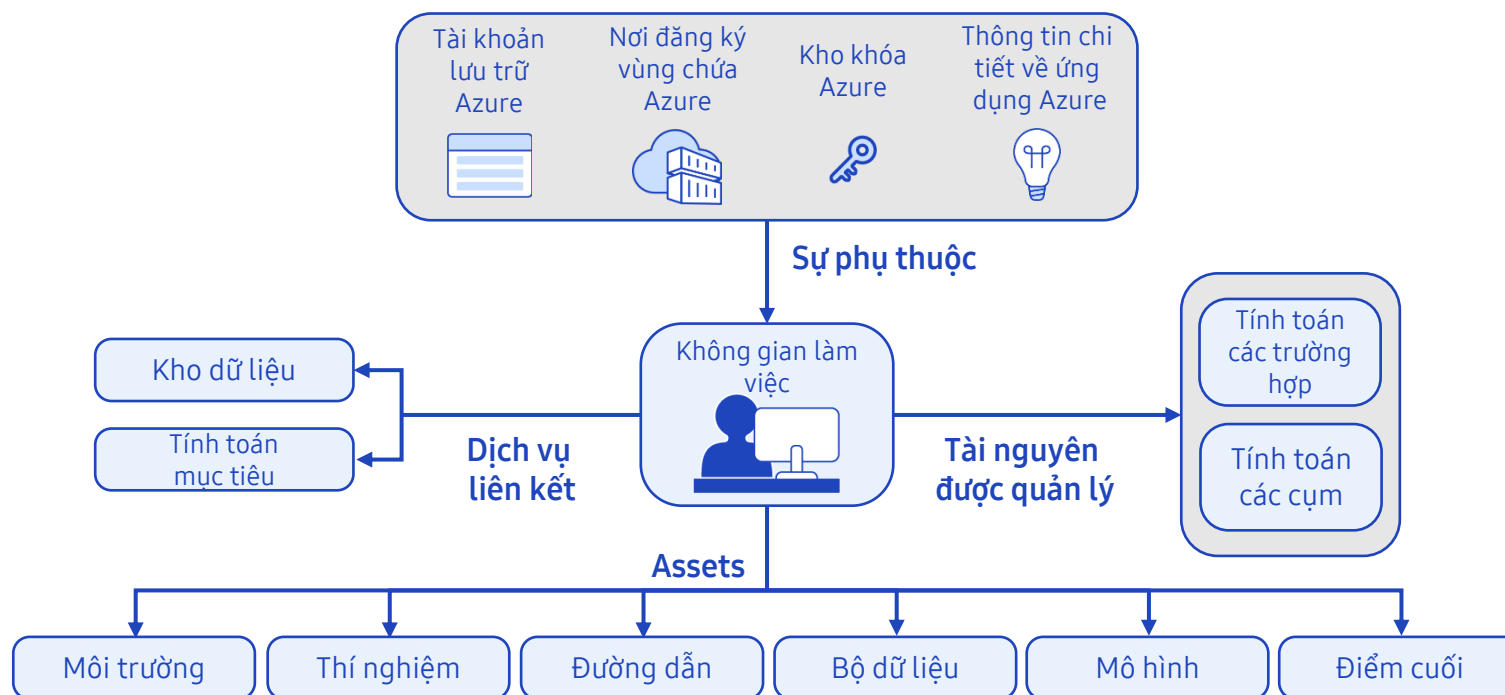
Các thuật toán tích hợp của Amazon SageMaker

I Thuật toán tích hợp



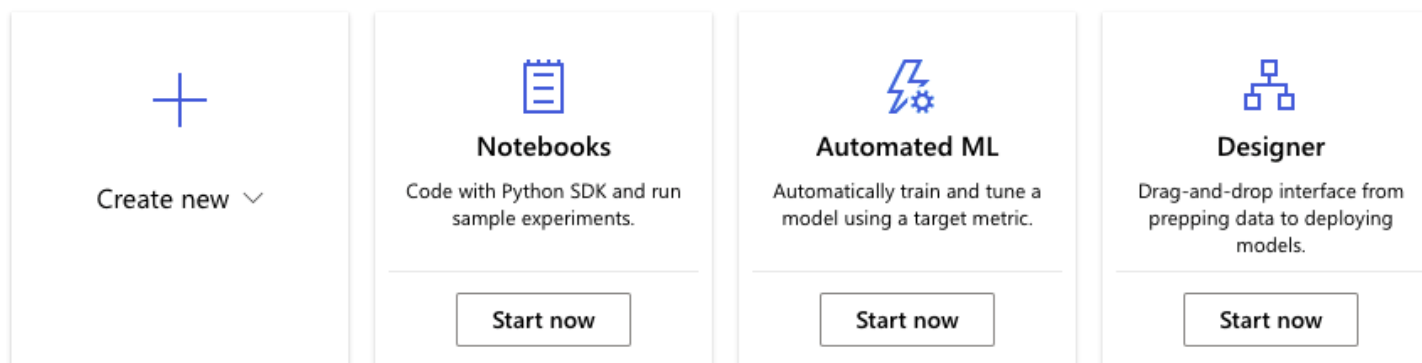
Học máy Azure

- Học máy Azure là một dịch vụ dựa trên đám mây giúp các nhà khoa học dữ liệu tiết kiệm thời gian bằng cách đơn giản hóa các tác vụ như chuẩn bị dữ liệu, đào tạo mô hình và triển khai các dịch vụ dự đoán



Studio học máy Azure

I Studio là một cổng web cho Azure Machine Learning



- ▶ Cung cấp môi trường không dùng mã và môi trường mã cho các nền tảng khoa học dữ liệu
- ▶ Notebooks – Tạo sổ ghi chép Jupyter và chạy tập lệnh
- ▶ Automated ML – Bước qua ML mà không cần tạo mã
- ▶ Designer – cung cấp chức năng bằng cách kéo-thả sử dụng các mô-đun dựng sẵn

Cách sử dụng Học máy Azure

I Tạo tài nguyên không gian làm việc

Machine learning

Create a machine learning workspace

Basics Networking Advanced Tags Review + create

Project details

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *

Resource group * [Create new](#)

Workspace details

Specify the name and region for the workspace.

Workspace name *

Region *

Storage account * [Create new](#)

Key vault * [Create new](#)

Application insights * [Create new](#)

Container registry * [Create new](#)

[Review + create](#) [< Previous](#) [Next : Networking](#)

Microsoft.MachineLearningServices | Overview [Deployment](#)

Search (Cmd+/) << [Delete](#) [Cancel](#) [Redeploy](#) [Refresh](#)

[Overview](#)

[Inputs](#)

[Outputs](#)

[Template](#)

[We'd love your feedback! →](#)

✓ Your deployment is complete

Deployment name: Microsoft.MachineLearningServices
Subscription: 종량제1
Resource group: SIC

Home > Microsoft.MachineLearningServices >

azure-ml-Lecture [Machine learning](#)

Search (Cmd+/) << [Download config.json](#) [Delete](#)

[Overview](#)

[Activity log](#)

[Access control \(IAM\)](#)

[Tags](#)

[Diagnose and solve problems](#)

Essentials

Resource group : SIC

Location : East US

Subscription : 종량제1

Subscription ID : 4e4d7de4-4f0d-4a4b-b389-b569fe38696f

Cách sử dụng Học máy Azure

I Tạo phiên bản điện toán

- ▶ Được sử dụng để tạo mã và thực thi các tập lệnh Python và sổ ghi chép Jupyter

I Tạo cụm máy tính

- ▶ Triển khai các quy trình đào tạo và suy luận hàng loạt bằng CPU/GPU

Create compute cluster ⓘ

☒ Virtual Machine

☒ Advanced Settings

Configure Settings

Configure compute cluster settings for your selected virtual machine size.

Name	Category	Cores	Available quota	RAM	Storage	Cost/Node
Standard_DS2_v2	General purpose	2	10 cores	7 GB	14 GB	\$0.16/hr

Compute name * ⓘ

user00-mcluster

Minimum number of nodes * ⓘ

0

Maximum number of nodes * ⓘ

1

Idle seconds before scale down * ⓘ

2400

☐ Enable SSH access ⓘ

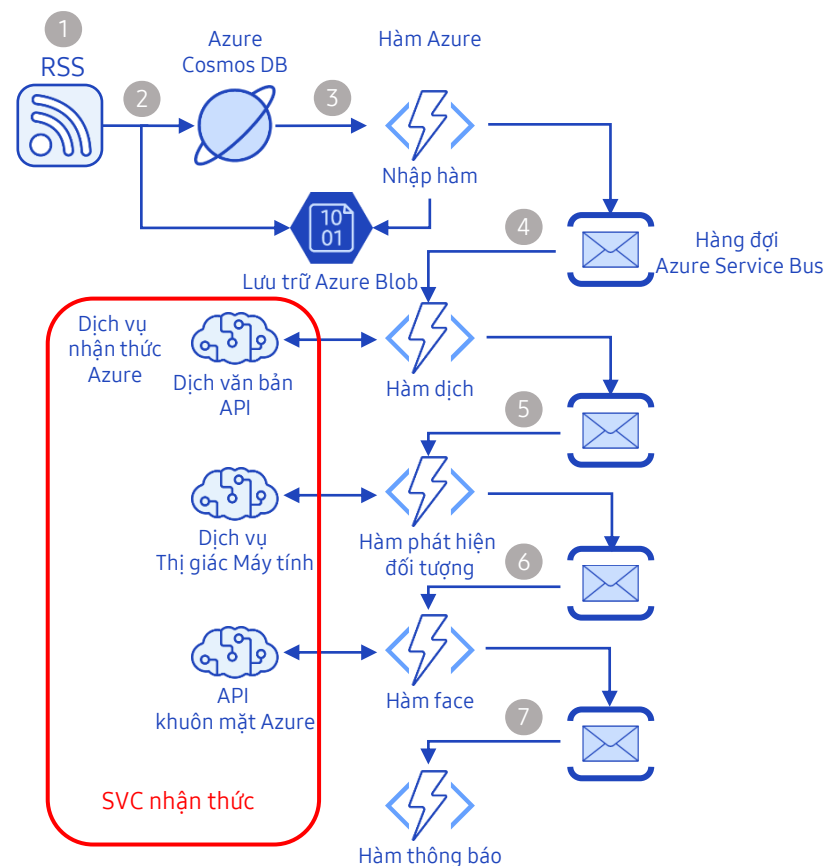
> Advanced settings

Dịch vụ nhận thức

I Azure Cognitive Services là dịch vụ dựa trên đám mây với API REST và SDK thư viện máy khách để giúp bạn xây dựng trí thông minh nhận thức vào ứng dụng của mình

I Danh mục dịch vụ

- ▶ API trực quan
- ▶ API giọng nói
- ▶ API ngôn ngữ
- ▶ API quyết định
- ▶ API tìm kiếm



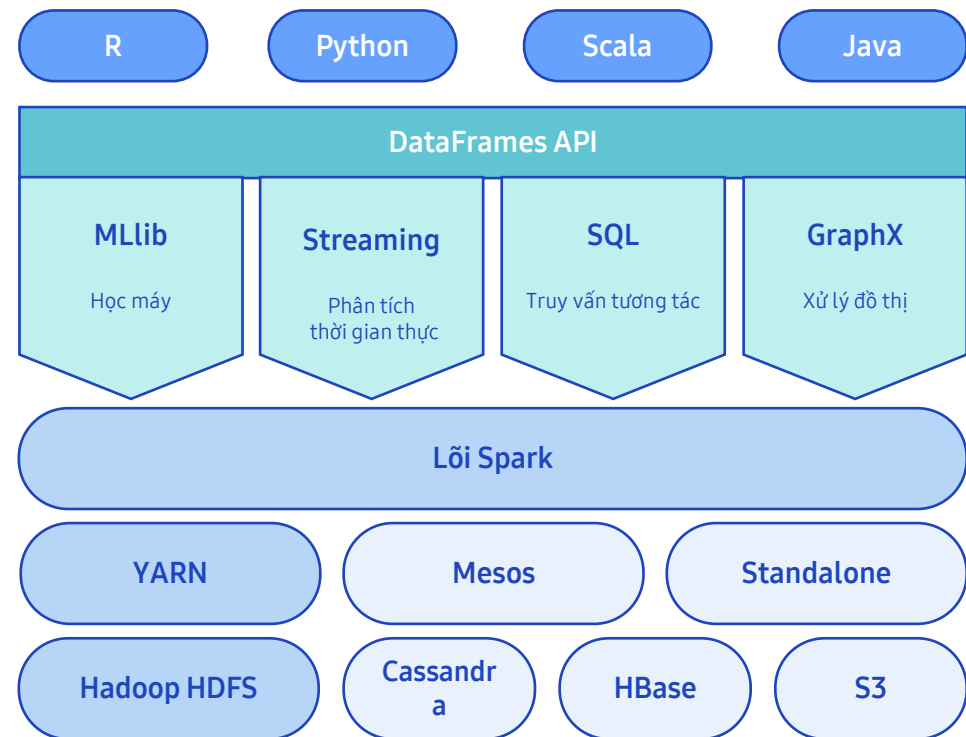
Bài 1

Học Máy & Mô Hình

- | 1.1. Kiến thức cơ bản về học máy
- | 1.2. Học máy trong đám mây công cộng
- | **1.3. Đường ống Apache Spark ML**

Apache Spark là gì?

- I Apache Spark là một công cụ phân tích thống nhất để xử lý dữ liệu quy mô lớn
 - ▶ Một bộ API cấp cao trong Scala, Python, Java và R
- I Công cụ xử lý dữ liệu đa năng cung cấp
 - ▶ Xử lý tương tác
 - ▶ Xử lý hàng loạt cực nhanh
 - ▶ Xử lý luồng thời gian thực
 - ▶ Mô hình học máy phân tán và công cụ suy luận
 - ▶ Xử lý đồ thị
- I Công cụ tính toán phân tán trong bộ nhớ



Spark MLlib là gì?

- | Spark MLlib Là một thư viện gồm các thuật toán và tiện ích ML được thiết kế để chạy song song trên cụm Spark
 - ▶ Giới thiệu một số loại dữ liệu mới bao gồm vectơ (dày đặc và thưa thớt), điểm được gắn nhãn, xếp hạng, v.v.
 - ▶ Cho phép gọi các thuật toán khác nhau trên bộ dữ liệu phân tán (RDD/Dataset)
- | Spark ML có thể giúp xây dựng các đường dẫn ML đẹp mắt
- | Các thư viện và framework hiện có giúp giảm bớt rất nhiều công việc tẻ nhạt
- | MLlib là thư viện chức năng máy học (ML) của Spark được thiết kế để chạy song song trên các cụm.
 - ▶ MLlib chứa nhiều thuật toán học tập
- | MLlib gọi các thuật toán khác nhau trên RDD
- | Một số thuật toán ML cổ điển không được bao gồm trong Spark MLlib vì chúng không được thiết kế để song song

Tổng quan về Spark MLlib

- I MLlib là thư viện học máy của Spark
 - ▶ Mục tiêu là làm cho việc học máy thực tế có thể mở rộng và dễ dàng
- I Bao gồm các thuật toán và tiện ích học tập phổ biến
 - ▶ Phân loại, hồi quy, phân cụm, lọc cộng tác, giảm kích thước, cũng như nguyên mẫu tối ưu hóa cấp thấp hơn và API đường ống cấp cao hơn
- I Chia thành hai gói
 - ▶ Spark.mllib
 - ▶ Spark.ml

Spark DataFrames

- I Về mặt khái niệm tương đương với một bảng trong cơ sở dữ liệu quan hệ
 - ▶ Tương tự như khung dữ liệu R hoặc Python
- I Tối ưu hóa phong phú dưới mui xe
- I DataFrames có thể được xây dựng từ nhiều nguồn
 - ▶ Tập dữ liệu có cấu trúc
 - ▶ Bàn trong Hive
 - ▶ Cơ sở dữ liệu bên ngoài
 - ▶ RDD hiện tại

Kiểu dữ liệu - MLlib

- | Hỗ trợ các vectơ và ma trận cục bộ được lưu trữ trên một máy
- | Ma trận phân tán được hỗ trợ bởi một hoặc nhiều RDD
- | Các vectơ cục bộ và ma trận cục bộ là các mô hình dữ liệu đơn giản đóng vai trò là giao diện công cộng
- | Một ví dụ đào tạo được sử dụng trong học tập có giám sát được gọi là điểm được dán nhãn

Kiểu dữ liệu – Vectơ cục bộ (Local Vector)

- I Các vectơ cục bộ là tập hợp các giá trị kiểu dữ liệu số nguyên và kép, được lưu trữ trên một máy.
 - ▶ Các chỉ mục bộ sưu tập của họ bắt đầu bằng 0
 - ▶ Hai loại có sẵn - dày đặc và thưa thớt
- I Các vectơ dày đặc lưu trữ các giá trị trong một mảng kiểu double: `Array[Double]`
- I Các vectơ thưa thớt lưu trữ dữ liệu trên các mảng song song. Một mảng lưu trữ giá trị, trong khi mảng kia lưu trữ chỉ mục

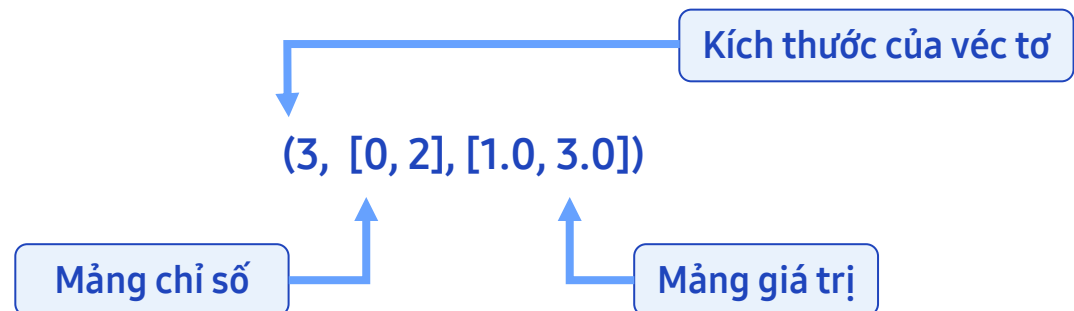
Vectơ dày đặc

Mảng giá trị gộp

`[1.0, 0.0, 3.0]`

Vectơ thưa thớt

hai mảng song song
một cho các chỉ số
một cho các giá trị



Điểm được dán nhãn

- | Điểm được dán nhãn lưu trữ nhãn hoặc sự thật cơ bản được sử dụng trong các thuật toán được giám sát
- | Một vectơ cục bộ có ý nghĩa hoặc thừa thớt được sử dụng để lưu trữ các nhãn
- | Phân loại nhị phân : Giá trị nhãn là 0 (phản hồi tiêu cực) hoặc 1 (phản hồi tích cực)
- | Phân loại nhiều lớp: Nhãn là 0,1, 2, v.v. cho mỗi phân loại
- | Các giá trị được lưu dưới dạng gấp đôi thay vì số nguyên
 - ▶ Cho phép sử dụng cùng một kiểu dữ liệu Điểm được gắn nhãn trong các thuật toán hồi quy và phân loại

Kiểu dữ liệu - Ma trận cục bộ dày đặc (Dense Local Matrix)

- I Một ma trận cục bộ chứa 3 giá trị để mô tả ma trận
 - ▶ Các kiểu số nguyên được sử dụng cho chỉ số hàng và cột
 - ▶ Các giá trị thực tế được lưu trữ trong một Mảng kiểu double: Array[Double]
 - ▶ Có ma trận địa phương dày đặc và thưa thớt
- I Ma trận dày đặc
 - ▶ Các giá trị được lưu trữ theo thứ tự cột lớn, trong Array[Double]

$$\begin{pmatrix} 1.0 & 2.0 \\ 3.0 & 4.0 \\ 5.0 & 6.0 \end{pmatrix}$$

```
# Create a dense matrix ((1.0, 2.0), (3.0, 4.0), (5.0, 6.0))
# Format is (<num rows>, <num cols>, <Array[Double] of values in column major
order

# Using factory class and method implementations of Matrices

dense_m = Matrices.dense(3, 2, [1.0, 3.0, 5.0, 2.0, 4.0, 6.0])
```

Kiểu dữ liệu – Ma trận cục bộ thưa thớt (Sparse Local Matrix)

I Ma trận thưa thớt

- ▶ Được lưu trữ dưới dạng Cột thưa được nén theo thứ tự cột chính

$$\begin{bmatrix} 9.0 & 6.0 & 0.0 \\ 0.0 & 0.0 & 0.0 \\ 0.0 & 0.0 & 8.0 \end{bmatrix}$$

I Định dạng cột thưa được nén

- ▶ NNZ = Số phần tử Khác Không = 3
- ▶ giá trị = Mảng của tất cả các giá trị khác không theo thứ tự cột → [9.0, 6.0, 8.0]
- ▶ rowIndices = Mảng chỉ số hàng của giá trị phù hợp → [0, 0, 3]
- ▶ colPtrs = Mảng chỉ số cột của giá trị phù hợp cộng với NNZ → [0, 1, 2, NNZ=3]

```
# create a sparse matrix using factory method
# Matrices.sparse(numRows, numCols, colPtrs, rowIndices, values)

sparse_mat = Matrices.sparse(3, 3, [0, 1, 2, 3], [0, 0, 2], [9, 6, 8])
```

Kiểu dữ liệu – Ma trận phân tán (Distributed Matrix)

- I Không giống như kiểu dữ liệu Ma trận cục bộ, các giá trị cho kiểu dữ liệu Ma trận phân tán được phân phối
 - ▶ Được lưu trữ trong một hoặc nhiều RDD - hãy nhớ rằng RDD được tự phân phối
 - ▶ Các chỉ số hàng và cột được lưu trữ dưới dạng dài
- I Phải thận trọng khi chọn kiểu dữ liệu này
 - ▶ Thay đổi Ma trận phân tán thành kiểu dữ liệu khác sẽ liên quan đến tất cả Người thi hành xáo trộn dữ liệu của họ
 - ▶ Xáo trộn rất tốn kém và liên quan đến I/O qua mạng
- I Bốn loại ma trận phân tán được triển khai:
 - ▶ RowMatrix
 - ▶ IndexedRowMatrix
 - ▶ CoordinateMatrix
 - ▶ BlockMatrix

Kiểu dữ liệu – Ma trận hàng (Row Matrix)

- RowMatrix là một ma trận phân tán, trong đó mỗi hàng được lưu trữ dưới dạng một vectơ cục bộ
- Các chỉ số hàng không quan trọng và do đó các vectơ cục bộ có thể được tổ chức thành các RDD

```
from pyspark.mllib.linalg.distributed import RowMatrix
# Create an RDD of vectors.
rows = sc.parallelize([[1, 2, 3], [4, 5, 6], [7, 8, 9], [10, 11, 12]])
# Create a RowMatrix from an RDD of vectors.
mat = RowMatrix(rows)
# Get its size.
m = mat.numRows() # 4
n = mat.numCols() # 3
# Get the rows as an RDD of vectors again.
rowsRDD = mat.rows
```

Kiểu dữ liệu – IndexedRowMatrix

- | IndexedRowMatrix tương tự như RowMatrix nhưng hiện tại, các chỉ số hàng rất quan trọng
- | Được lưu trữ dưới dạng RDD của các hàng được lập chỉ mục
 - ▶ Các đối tượng Hàng được lập chỉ mục lưu trữ giá trị chỉ mục hàng

```
from pyspark.mllib.linalg.distributed import IndexedRow, IndexedRowMatrix

# Create an RDD of indexed rows using IndexedRow class:
indexedRows = sc.parallelize([IndexedRow(0, [1, 2, 3]),
                              IndexedRow(1, [4, 5, 6]),
                              IndexedRow(2, [7, 8, 9]),
                              IndexedRow(3, [10, 11, 12])])

# Or by using a tuple of (<index long>, <value vector>)
indexedRows = sc.parallelize([(0, [1, 2, 3]),
                              (1, [4, 5, 6]),
                              (2, [7, 8, 9]),
                              (3, [10, 11, 12])])

# Create an IndexedRowMatrix from an RDD of IndexedRows.
mat = IndexedRowMatrix(indexedRows)
```

Kiểu dữ liệu – CoordinateMatrix

- I Ma trận tọa độ được sử dụng khi cả chỉ số hàng và cột đều quan trọng
- I Được lưu trữ dưới dạng RDD của bộ dữ liệu hoặc lớp MatrixEntry
 - ▶ Mỗi tuple → (<chỉ số hàng> của loại dài, <chỉ số cột của loại dài>, <giá trị của loại Double>)
 - ▶ Ngoài ra, thay vì một bộ, lớp MatrixEntry có thể được sử dụng

```
from pyspark.mllib.linalg.distributed import CoordinateMatrix, MatrixEntry

# Create an RDD of coordinate entries using MatrixEntry class
entries = sc.parallelize([MatrixEntry(0, 0, 1.2),
                          MatrixEntry(1, 0, 2.1),
                          MatrixEntry(6, 1, 3.7)])

# Or using (long, long, float) tuples:
entries = sc.parallelize([(0, 0, 1.2), (1, 0, 2.1), (2, 1, 3.7)])

# Create an CoordinateMatrix from an RDD of MatrixEntries.
mat = CoordinateMatrix(entries)
```


Kiểu dữ liệu – BlockMatrix

- | Một BlockMatrix là một ma trận phân tán được lưu trữ dưới dạng RDD của MatrixBlocks
- | Một MatrixBlock có thể được coi là một ma trận con của ma trận phân tán đầy đủ
 - ▶ Được biểu diễn bởi một bộ (<chỉ mục hàng ma trận con kiểu Int>, <chỉ mục cột ma trận con kiểu Int>, Ma trận)
 - ▶ Ma trận là ma trận con thực sự
 - ▶ Kích thước của ma trận con có thể được định cấu hình là rowsPerBlock x colsPerBlock
 - ▶ Kích thước mặc định là 1024 x 1024
- | Dễ dàng nhất để tạo BlockMatrix bằng cách chuyển đổi IndexedRowMatrix hoặc Tọa độ
 - ▶ Sử dụng phương thức toBlockMatrix
- | BlockMatrix hỗ trợ các phương thức như cộng và nhân với các BlockMatrix khác
- | Ngoài ra còn có chức năng xác thực () xác minh việc tạo đúng BlockMatrix nếu toBlockMatrix không được sử dụng và được tạo thủ công thông qua mã

Kiểu dữ liệu – BlockMatrix

- Một BlockMatrix là một ma trận phân tán được hỗ trợ bởi một RDD của MatrixBlocks

```
from pyspark.mllib.linalg import Matrices
from pyspark.mllib.linalg.distributed import BlockMatrix
# Create an RDD of sub-matrix blocks.
blocks = sc.parallelize([(0, 0), Matrices.dense(3, 3, [1, 2, 3, 4, 5,
6,7,8,9])), ((1, 1), Matrices.dense(3, 3, [10, 11, 12, 13, 14, 15, 16, 17,
18,])), ((2,0), Matrices.dense(3,3, [1, 1, 1, 1, 1, 1, 1, 1, 1]))])
# Create a BlockMatrix from an RDD of sub-matrix blocks.
mat = BlockMatrix(blocks, 3, 3)
# Get its size.
m = mat.numRows() # 3
n = mat.numCols() # 3
# Get the blocks as an RDD of sub-matrix blocks.
blocksRDD = mat.blocks
# Convert to a LocalMatrix.
localMat = mat.toLocalMatrix()
# Convert to an IndexedRowMatrix.
indexedRowMat = mat.toIndexedRowMatrix()
# Convert to a CoordinateMatrix.
coordinateMat = mat.toCoordinateMatrix()
```

[Lab1]

Làm việc với Học máy Azure



[Lab2] Làm việc với SageMaker trong AWS



Bài 2.

Thư viện học máy Spark

Mô hình hóa dữ liệu lớn & Trí tuệ nhân tạo

Bài 2

Thư viện học máy Spark

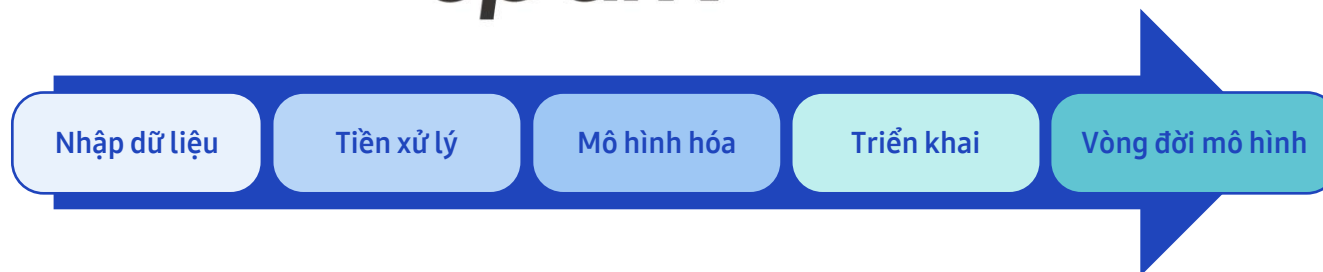
| 2.1. Tạo đường ống Spark MLlib

| 2.2. Phân loại và hồi quy

| 2.3. Phân cụm & Lọc cộng tác

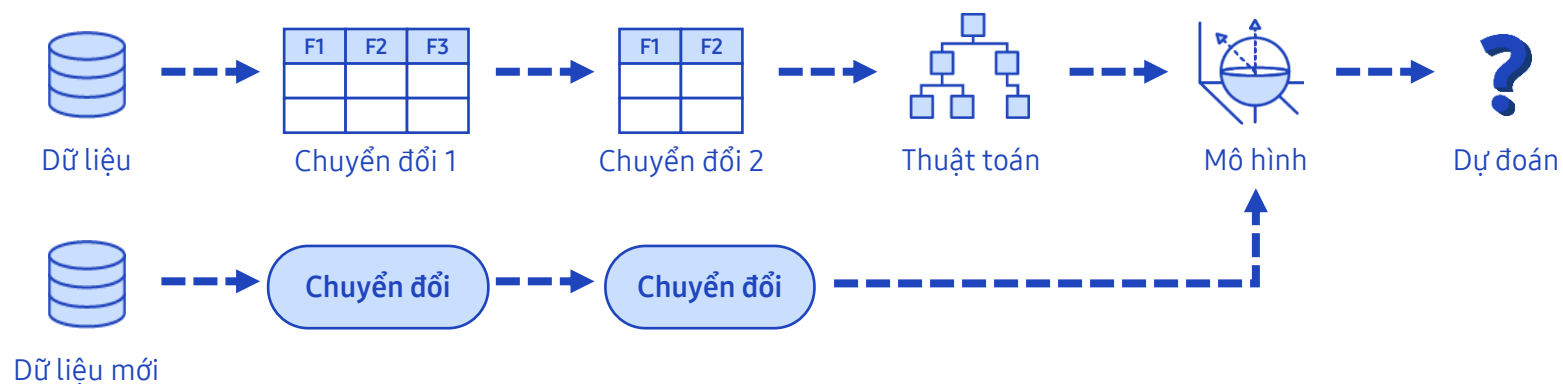
Đường ống Spark MLlib (1/3)

- I Quy trình Machine Learning (ML) là một quy trình công việc hoàn chỉnh kết hợp các thuật học máy
 - ▶ Nó tương tự như đường dẫn dữ liệu của Big data
- I Tạo một quy trình học máy Apache Spark
 - ▶ Spark MLlib và ML hỗ trợ nhiều thuật toán máy học
 - ▶ Có thể có nhiều bước cần thiết để xử lý và học hỏi từ dữ liệu, đòi hỏi một chuỗi các thuật toán



Đường ống Spark MLlib (2/3)

- I Nhiệm vụ học máy bao gồm một loạt các bước phức tạp
 - ▶ Chuyển đổi dữ liệu, đào tạo thuật toán và dự đoán mô hình
 - ▶ Chúng ta có thể coi các bước này như một đường ống dẫn dữ liệu đi qua đó
- I Các bước đường ống cụ thể có thể được chia thành hai loại
 - ▶ Máy biến áp: Chuyển đổi một tập dữ liệu này sang một tập hợp khác
 - ▶ Công cụ ước tính: Một thuật toán có thể được trang bị cho tập dữ liệu để tạo mô hình



Đường ống Spark MLlib (3/3)

I Các nội dung chính trong Pipelines

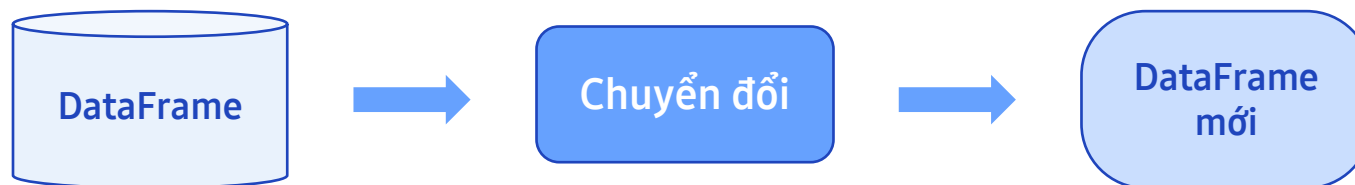
- ▶ DataFrame: Loại tập dữ liệu SparkML
- ▶ Máy biến áp: Chuyển đổi một Khung dữ liệu này sang Khung dữ liệu khác
- ▶ Công cụ ước tính: Thực thi thuật toán trên tập dữ liệu và tạo mô hình
- ▶ Quy trình: Xâu chuỗi nhiều bước để xác định quy trình làm việc ML
- ▶ Tham số: Công cụ ước tính và Máy biến áp sử dụng API thống nhất để chỉ định tham số

I Tại sao nên sử dụng DataFrame cho học máy?

- ▶ Hoạt động hiệu quả là có thể do cấu trúc cột
- ▶ Loại đầu vào duy nhất của các thuật toán khác nhau
- ▶ DataFrames là một loại dữ liệu thuận tiện để thực hiện các bước trong hoạt động học máy

Transformers (máy biến áp) (1/2)

- I Máy biến áp lấy DataFrame làm đầu vào và trả về DataFrame mới
 - ▶ Tổng quát hóa khái niệm phép biến đổi thành một định nghĩa rộng hơn
 - ▶ Theo định nghĩa này, mô hình học máy được đào tạo là một máy biến áp
 - ▶ Đọc một cột (ví dụ: văn bản), ánh xạ cột đó vào một cột mới (ví dụ: vectơ đặc trưng) và xuất một DataFrame mới với cột được ánh xạ được nối thêm



Transformers (máy biến áp) (2/2)

- I Tất cả các máy biến áp đều triển khai các phương thức chuyển đổi và phương thức **fit()** nếu việc chuyển đổi phụ thuộc vào dữ liệu
- I Máy biến áp chuyển đổi khung dữ liệu đầu vào thành khung dữ liệu khác
 - ▶ Tham số **inputCol()** chỉ định tên của cột trong DataFrame để chuyển đổi
 - ▶ Tham số **outputCol()** chỉ định cột lưu trữ dữ liệu đã chuyển đổi trong DataFrame đầu ra.
- I Nếu có hai loại biến áp, biến đổi là chức năng chính
 - ▶ Tính năng máy biến áp
 - ▶ Mô hình học tập
- I Máy biến áp thường được sử dụng cho tính năng máy biến áp
 - ▶ **StringIndexer()**
 - ▶ **PolynomialExpansion()**
 - ▶ **VectorAssembler()**
- I Mô hình học tập lấy một khung dữ liệu tính năng làm đầu vào và xuất ra một khung dữ liệu với các giá trị được dự đoán dưới dạng các cột mới.

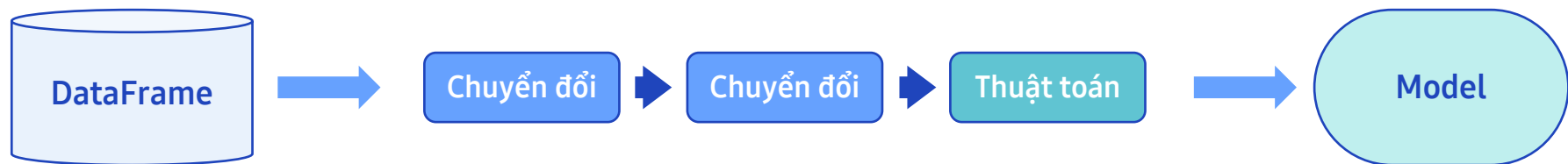
Estimators (công cụ ước tính)

- I Công cụ ước tính đại diện cho thuật toán được thực hiện để xác định các tham số mô hình
 - ▶ Lấy DataFrame làm đầu vào và tạo mô hình (Transformer)
 - ▶ Công cụ ước tính được khởi tạo với một bộ tham số cụ thể được sử dụng trong quá trình thực thi thuật toán.
 - ▶ Tất cả các công cụ ước tính đều triển khai phương thức **fit()**, phương thức này được gọi để chạy thuật toán trên dữ liệu được cung cấp.
 - ▶ Kết quả là một thể hiện của một mô hình thuật toán cụ thể



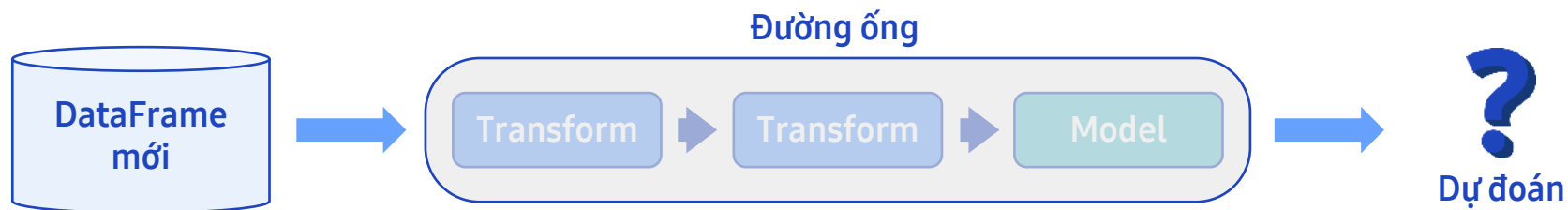
Đường ống (1/3)

- | Đường ống đại diện cho một loạt các bước trong quy trình học máy
 - ▶ Bản thân đường ống là một công cụ ước tính
- | Mỗi giai đoạn đường ống là một máy biến áp hoặc một công cụ ước tính
- | Lấy DataFrame làm đầu vào và tạo PipelineModel làm đầu ra



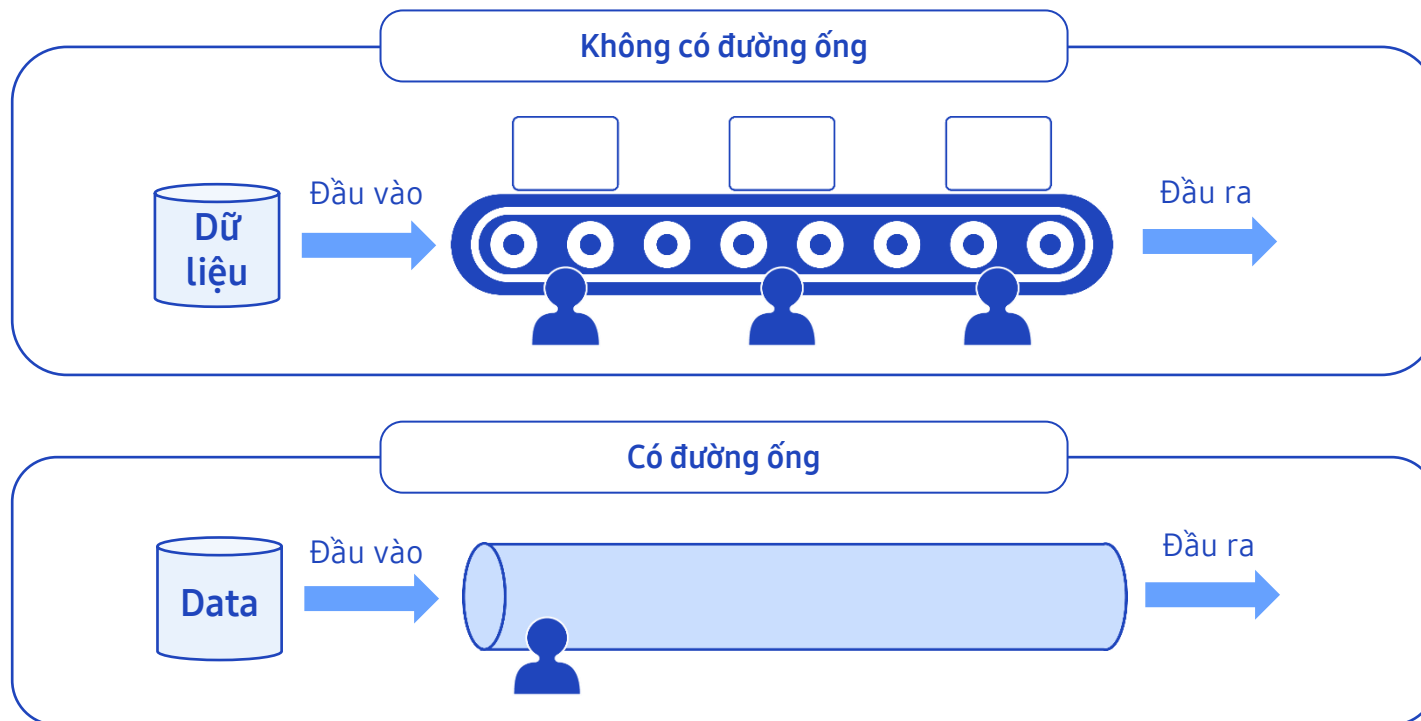
Đường ống (2/3)

- Một PipelineModel chứa các kết quả đầu ra của mô hình từ các phép biến đổi và thuật toán được kết nối với đường ống
 - ▶ PipelineModel thực hiện các phương thức biến đổi và hoạt động như một Transformer
 - ▶ Áp dụng các phép biến đổi trong quy trình và đưa ra dự đoán bằng cách sử dụng mô hình do thuật toán tạo ra
 - ▶ Lấy một DataFrame làm đầu vào và trả về một DataFrame chứa các dự đoán mô hình



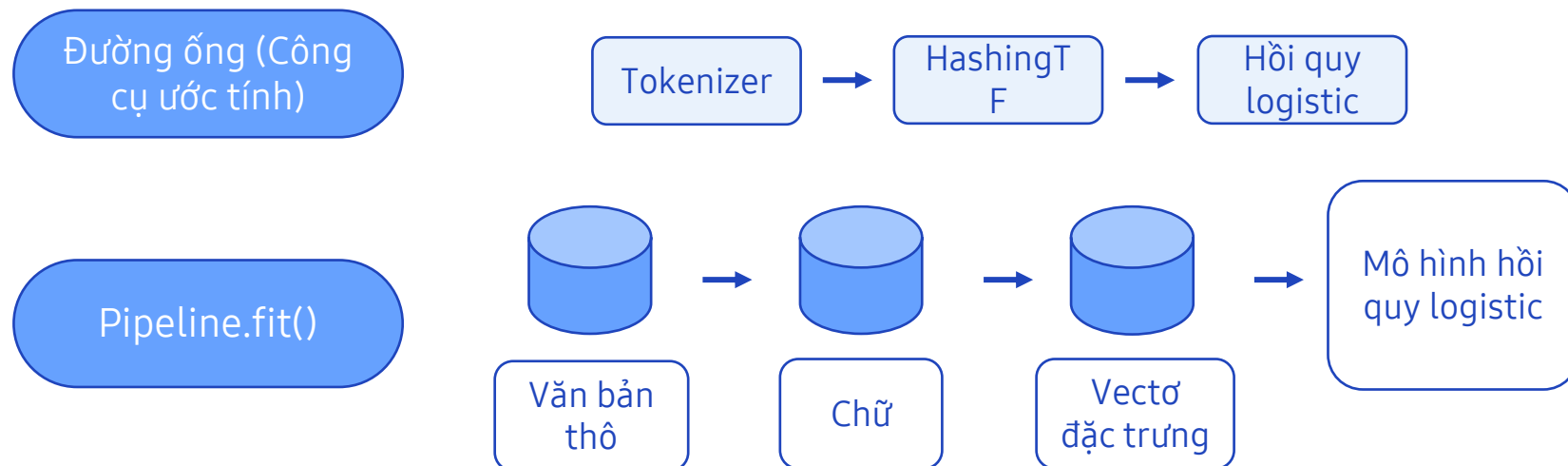
Đường ống (3/3)

I So sánh có và không có Đường ống



Quy trình làm việc ML không có Đường ống

- Giờ đây, Đường ống kết hợp tất cả các máy biến áp (transformer) và công cụ ước tính (estimator) có thể được sử dụng trực tiếp với dữ liệu mới



Tạo Spark ML mà không cần đường dẫn (1/2)

I Ví dụ trong Python

```
# Prepare training data from a list of (label, features) tuples.
training = spark.createDataFrame([
    (1.0, Vectors.dense([0.0, 1.1, 0.1])),
    (0.0, Vectors.dense([2.0, 1.0, -1.0])),
    (0.0, Vectors.dense([2.0, 1.3, 1.0])),
    (1.0, Vectors.dense([0.0, 1.2, -0.5]))], ["label", "features"])

# Create a LogisticRegression instance. This instance is an Estimator.
lr = LogisticRegression(maxIter=10, regParam=0.01)

# Learn a LogisticRegression model. This uses the parameters stored in lr.
modell = lr.fit(training)
# We may alternatively specify parameters using a Python dictionary as a paramMap
paramMap = {lr.maxIter: 20}
paramMap[lr.maxIter] = 30 # Specify 1 Param, overwriting the original maxIter.
# Specify multiple Params.
paramMap.update({lr.regParam: 0.1, lr.threshold: 0.55}) # type: ignore
# You can combine paramMap's, which are python dictionaries.
# Change output column name
paramMap2 = {lr.probabilityCol: "myProbability"} # type: ignore
paramMapCombined = paramMap.copy()
paramMapCombined.update(paramMap2) # type: ignore
```

Tạo Spark ML mà không cần đường dẫn (2/2)

I Ví dụ trong Python

```
# Now learn a new model using the paramMapCombined parameters.
# paramMapCombined overrides all parameters set earlier via lr.set* methods.
model2 = lr.fit(training, paramMapCombined)

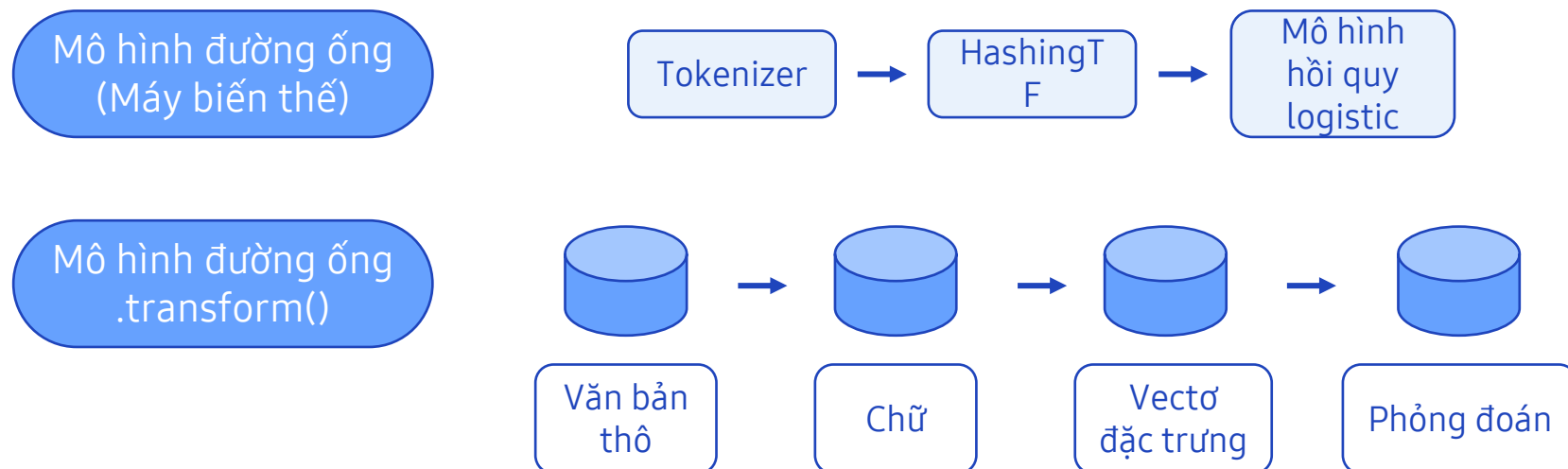
# Prepare test data
test = spark.createDataFrame([
    (1.0, Vectors.dense([-1.0, 1.5, 1.3])),
    (0.0, Vectors.dense([3.0, 2.0, -0.1])),
    (1.0, Vectors.dense([0.0, 2.2, -1.5]))], ["label", "features"])

# Make predictions on test data using the Transformer.transform() method.
# LogisticRegression.transform will only use the 'features' column.
# Note that model2.transform() outputs a "myProbability" column instead of the usual
# 'probability' column since we renamed the lr.probabilityCol parameter previously.
prediction = model2.transform(test)
result = prediction.select("features", "label", "myProbability", "prediction") \
    .collect()

for row in result:
    print("features=%s, label=%s -> prob=%s, prediction=%s"
          % (row.features, row.label, row.myProbability, row.prediction))
```

Quy trình làm việc ML với Đường ống

I Giờ đây, Đường ống kết hợp tất cả các máy biến áp và công cụ ước tính có thể được sử dụng trực tiếp với dữ liệu mới



Tạo Spark ML với đường dẫn (1/2)

I Ví dụ trong Python

```
from pyspark.ml import Pipeline
from pyspark.ml.classification import LogisticRegression
from pyspark.ml.feature import HashingTF, Tokenizer

# Prepare training documents from a list of (id, text, label) tuples.
training = spark.createDataFrame([
    (0, "a b c d e spark", 1.0),
    (1, "b d", 0.0),
    (2, "spark f g h", 1.0),
    (3, "hadoop mapreduce", 0.0)
], ["id", "text", "label"])

# Configure an ML pipeline, which consists of three stages: tokenizer, hashingTF, and lr.
tokenizer = Tokenizer(inputCol="text", outputCol="words")
hashingTF = HashingTF(inputCol=tokenizer.getOutputCol(), outputCol="features")
lr = LogisticRegression(maxIter=10, regParam=0.001)
pipeline = Pipeline(stages=[tokenizer, hashingTF, lr])

# Fit the pipeline to training documents.
model = pipeline.fit(training)
```

Input

Tạo Spark ML với đường dẫn (2/2)

I Ví dụ trong Python

```
# Prepare test documents, which are unlabeled (id, text) tuples.
test = spark.createDataFrame([
    (4, "spark i j k"),
    (5, "l m n"),
    (6, "spark hadoop spark"),
    (7, "apache hadoop")
], ["id", "text"])
# Make predictions on test documents and print columns of interest.
prediction = model.transform(test)
selected = prediction.select("id", "text", "probability", "prediction")
for row in selected.collect():
    rid, text, prob, prediction = row # type: ignore
    print(
        "(%d, %s) --> prob=%s, prediction=%f" % (
            rid, text, str(prob), prediction # type: ignore
        )
    )
```

Input

```
(4, spark i j k) --> prob=[0.1596407738787412,0.8403592261212588], prediction=1.000000
(5, l m n) --> prob=[0.8378325685476614,0.16216743145233858], prediction=0.000000
(6, spark hadoop spark) --> prob=[0.06926633132976266,0.9307336686702373], prediction=1.000000
(7, apache hadoop) --> prob=[0.9821575333444208,0.017842466655579203], prediction=0.000000
```

Output

Bài 2

Thư viện học máy Spark

- | 2.1. Tạo đường ống Spark MLlib
- | **2.2. Phân loại và hồi quy**
- | 2.3. Phân cụm & Lọc cộng tác

Bản tóm tắt học máy

- I Các thuật toán học máy cố gắng dự đoán hoặc đưa ra quyết định dựa trên dữ liệu đào tạo
 - ▶ Có nhiều loại vấn đề học tập, bao gồm phân loại, hồi quy hoặc phân cụm. Tất cả đều có mục tiêu khác nhau
- I Các phương pháp thuật toán được giám sát được hỗ trợ MLlib

Vấn đề phân loại nhị phân

SVM tuyến tính, hồi quy logistic, cây quyết định, rừng ngẫu nhiên, cây tăng cường độ dốc, naive bayes

Vấn đề phân loại đa lớp

hồi quy logistic, cây quyết định, rừng ngẫu nhiên, naive Bayes

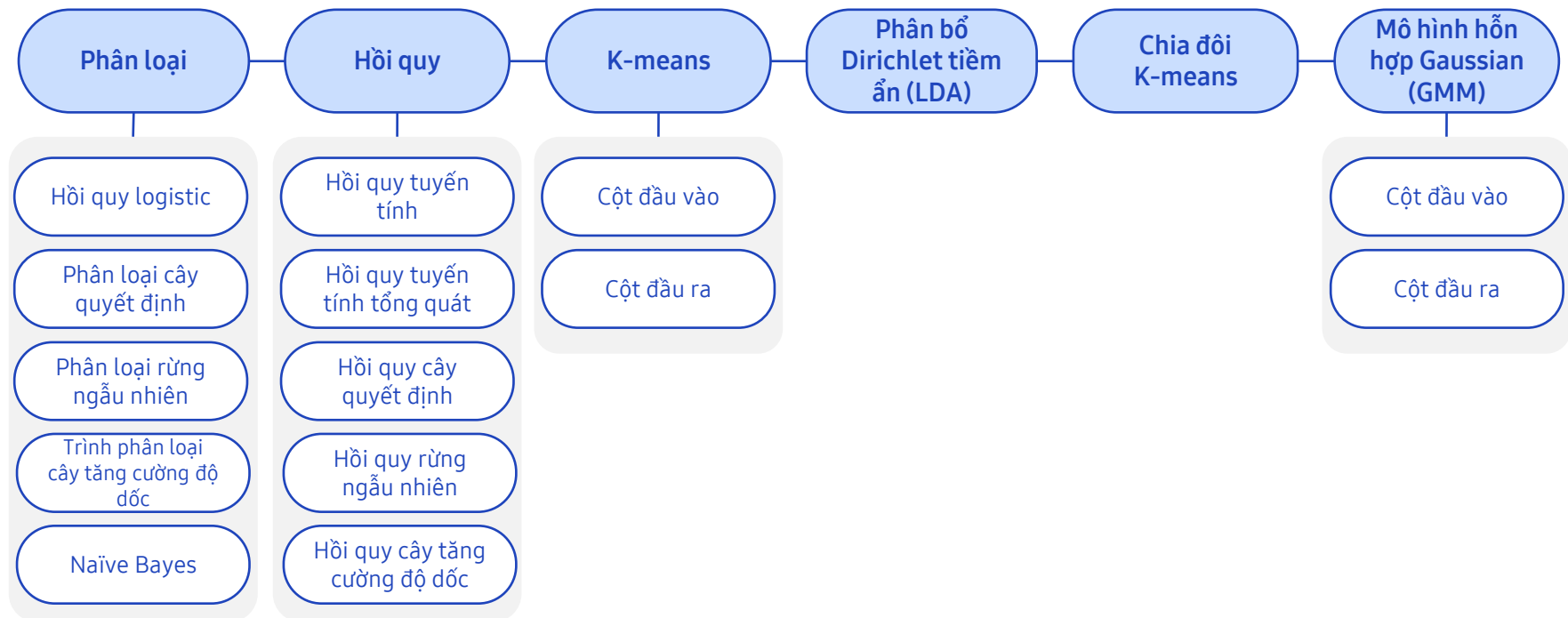
Regression Problems

biên phương nhỏ nhất tuyến tính, Lasso, hồi quy sườn, cây quyết định, rừng ngẫu nhiên, cây tăng cường độ dốc, hồi quy đẳng hướng

Thuật toán của Spark MLlib

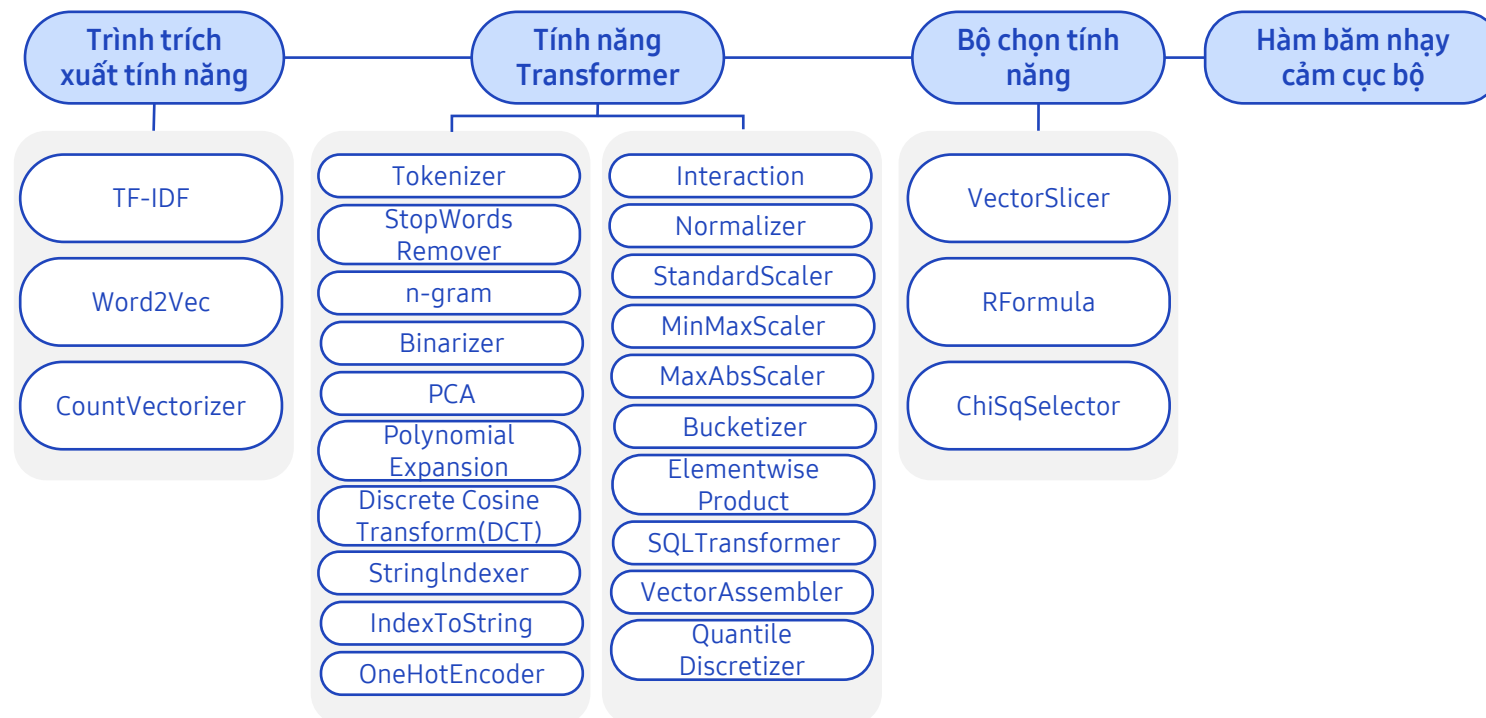
I Các thuật toán học máy cố gắng dự đoán hoặc đưa ra quyết định dựa trên dữ liệu đào tạo

- ▶ Có nhiều loại vấn đề học tập, bao gồm phân loại, hồi quy hoặc phân cụm



Tiền xử lý bằng Spark MLlib

- Spark Programming + Spark SQL + Spark Streaming
- Spark MLlib API



Ví dụ tiền xử lý

I Tokenizer -> StopWordsRemover -> Word2Vec

id	raw	filtered
0	[I, saw, the, red, balloon]	[saw, red, balloon]
1	[Mary, had, a, little, lamb]	[Mary, little, lamb]

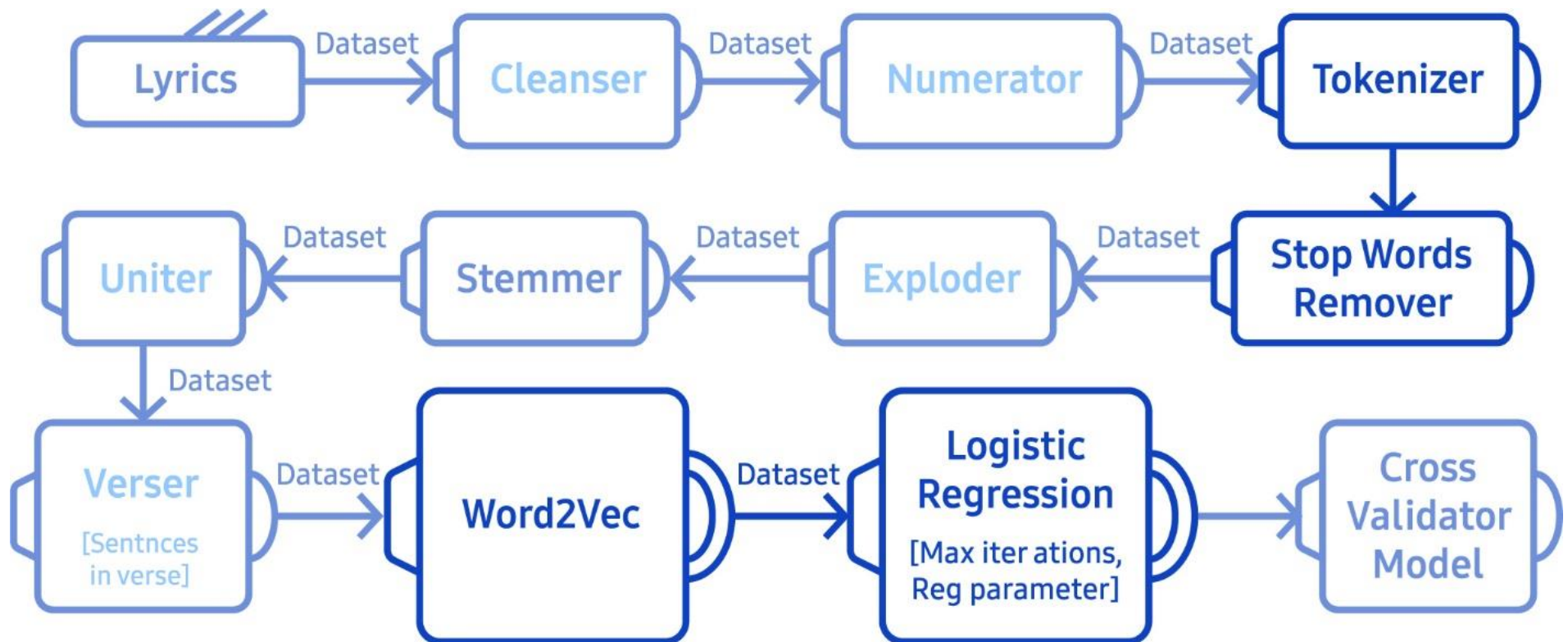
```
import org.apache.spark.ml.feature.StopWordsRemover

val remover = new StopWordsRemover()
  .setInputCol("raw")
  .setOutputCol("filtered")
val dataSet = spark.createDataFrame(Seq(
  (0, Seq("I", "saw", "the", "red", "balloon")),
  (1, Seq("Mary", "had", "a", "little", "lamb"))
)).toDF("id", "raw")

remover.transform(dataSet).show(false)
```

Input

Ví dụ về đường dẫn Spark ML



Phân loại là gì?

- | Một kỹ thuật học có giám sát trong đó giá trị nhãn Boolean y được dự đoán từ một vectơ đặc trưng
- | Học từ một tập hợp dữ liệu đào tạo đã được gắn nhãn với các quan sát thực tế cơ bản
- | Xác định một hàm (f) mà khi áp dụng cho x sẽ tạo ra giá trị dương (đúng) hoặc âm (sai) tương ứng với y

Ví dụ phân loại nhị phân

- I Dự đoán xem sinh viên tốt nghiệp có được tuyển dụng trong vòng 6 tháng hay không dựa trên bộ dữ liệu được thu thập cho mỗi sinh viên
- I Mỗi quan sát được đại diện bởi một true/false
 - ▶ Mỗi học sinh được biểu diễn dưới dạng một tập hợp các số

Vectơ đặc trưng X_i

[3.8	2200	3.5	8	100000	100000	0.9	1	1]
[3.3	2000	3.7	8	50000	100000	0.3	0	1]
[3.0	2100	3.3	9	90000	30000	0.6	0	1]
[3.0	2300	3.8	7	50000	150000	0.9	1	1]
[2.5	1800	2.9	10	50000	0	0.7	1	0]
[2.6	1900	3.0	8	100000	60000	0.4	0	1]

Labels Y_i

1
-1
1
-1
1
1

Phân loại – Bộ huấn luyện

I Let's limit to Graduating GPA and Total Student Loan

Vectơ đặc trưng X_i

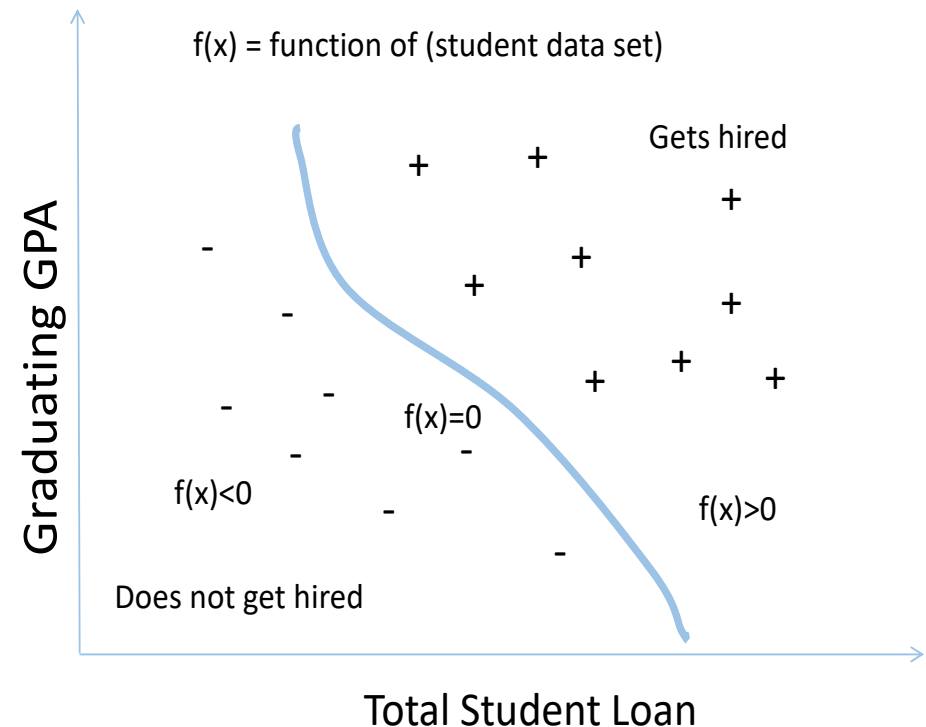
[3.5	100000]
[3.7	50000]
[3.3	90000]
[3.8	50000]
[2.9	50000]
[3.0	100000]

Nhãn Y_i (Thuê trong vòng 6 tháng)

1
-1
1
-1
1
1

Định nghĩa chính thức của phân loại

- I Đưa ra một tập huấn luyện (x_i, y_i) cho $i=1..n$, trong đó
 - ▶ x_i là một vectơ đặc trưng mô tả một số đặc điểm
 - ▶ y_i là nhãn
- I Chúng tôi muốn tạo một mô hình phân loại f có thể dự đoán nhãn y cho một x mới



Học máy hồi quy là gì?

- | Một kỹ thuật học có giám sát được sử dụng để dự đoán các giá trị liên tục
- | Chia làm hai gói:
 - ▶ spark.mllib chứa API gốc được xây dựng trên RDD
 - ▶ spark.ml cung cấp API cấp cao hơn được xây dựng trên DataFrames
- | Nên sử dụng spark.ml vì với DataFrames, API linh hoạt và linh hoạt hơn. Kế hoạch là tiếp tục hỗ trợ spark.mllib cùng với sự phát triển của spark.ml

Ví dụ hồi quy

- I Dự đoán thu nhập trong tương lai của một sinh viên
 - ▶ Dự đoán mức thu nhập trong tương lai của sinh viên sắp tốt nghiệp dựa trên bộ dữ liệu được thu thập cho từng sinh viên
- I Bộ huấn luyện - mỗi quan sát được biểu thị bằng một bộ số
 - ▶ Mỗi học sinh được biểu diễn dưới dạng một tập hợp các số

Vectơ đặc trưng X_i

[3.8	2200	3.5	8	100000	100000	0.9	1	1]
[3.3	2000	3.7	8	50000	100000	0.3	0	1]
[3.0	2100	3.3	9	90000	30000	0.6	0	1]
[3.0	2300	3.8	7	50000	150000	0.9	1	1]
[2.5	1800	2.9	10	50000	0	0.7	1	0]
[2.6	1900	3.0	8	100000	60000	0.4	0	1]

Nhãn Y_i (Thu nhập trong tương lai)

120
100
90
150
100
95

Hồi quy tuyến tính đơn giản – tập dữ liệu

- Chỉ có một tính năng duy nhất được bao gồm từ quan sát
 - Mỗi học sinh được đại diện với một tính năng duy nhất X_i , Điểm trung bình đầu ra

Vectơ đặc trưng X_i

[3.5]
[3.7]
[3.3]
[3.8]
[2.9]
[3.0]

Nhãn Y_i (Thu nhập trong tương lai)

120
100
90
150
100
95

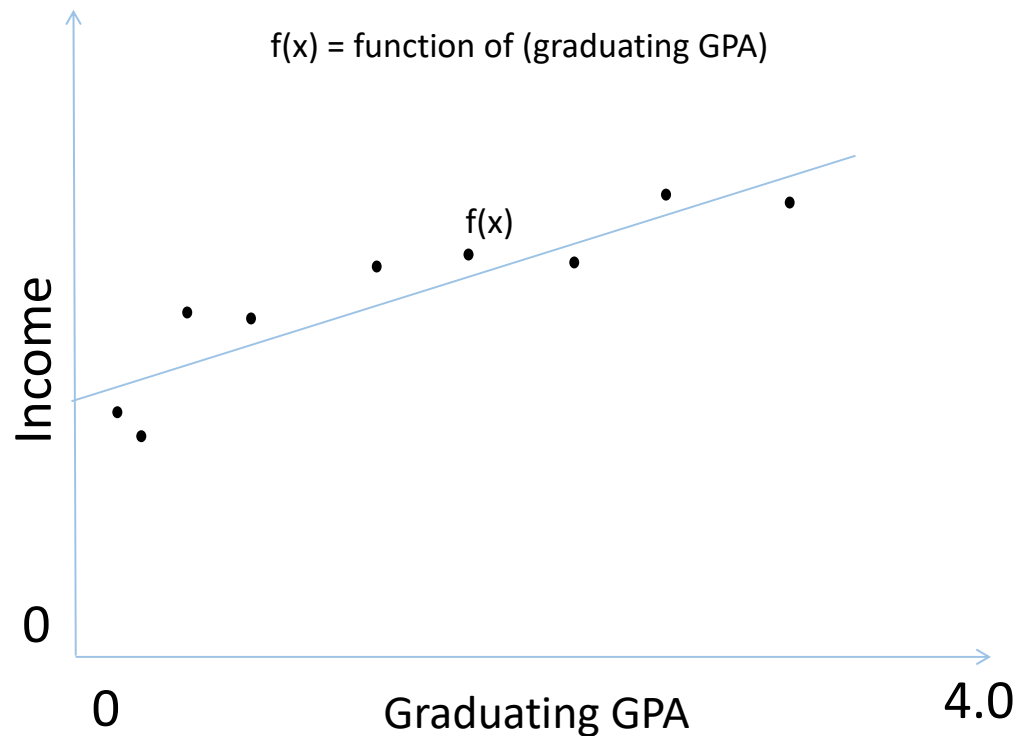
Hồi quy tuyến tính đơn giản - Hàm

I Một phương trình tuyến tính đơn giản

- ▶ $f(x_i) = \beta_0 + \beta_1 * x_i$
- ▶ $f(x_i) = 75 + 20 * x_i$

I hàm (điểm trung bình tốt nghiệp)

- ▶ $= 75K + 20K * GPA$



Hồi quy mẫu logistic

I Sử dụng PySpark

```
from pyspark.mllib.classification import LogisticRegressionWithLBFGS, LogisticRegressiorModel
from pyspark.mllib.regression import LabeledPoint

# Load and parse the data
def parsePoint(line):
    values = [float(x) for x in line.split(' ')]
    return LabeledPoint(values[0], values[1:])

data = sc.textFile("data/mllib/sample_svm_data.txt")
parsedData = data.map(parsePoint)

# Build the model
model = LogisticRegressionWithLBFGS.train(parsedData)

# Evaluating the model on training data
labelsAndPreds = parsedData.map(lambda p: (p.label, model.predict(p.features)))
trainErr = labelsAndPreds.filter(lambda (v, p): v != p).count() / float(parsedData.count())
print("Training Error = " + str(trainErr))

# Save and load model
model.save(sc, "myModelPath")
sameModel = LogisticRegressionModel.load(sc, "myModelPath")
```

Bài 2

Thư viện học máy Spark

| 2.1. Tạo đường ống Spark MLlib

| 2.2. Phân loại và hồi quy

| **2.3. Phân cụm & Lọc cộng tác**

Bản tóm tắt Spark MLlib

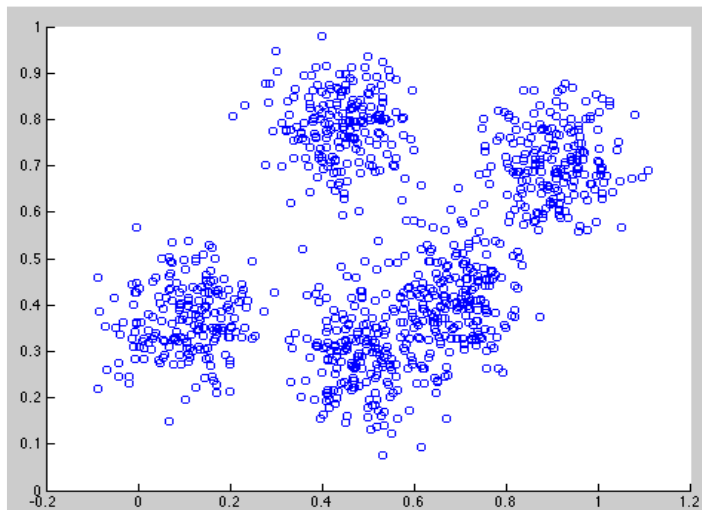
I Các tính năng của MLlib

- ▶ Tiện ích: đại số tuyến tính, thống kê, v.v.
- ▶ Trình trích xuất tính năng, Trình chuyển đổi tính năng, Bộ chọn tính năng
- ▶ hồi quy
- ▶ Phân loại nhị phân, Phân loại đa lớp
- ▶ Phân cụm
- ▶ Lọc cộng tác
- ▶ Giảm kích thước

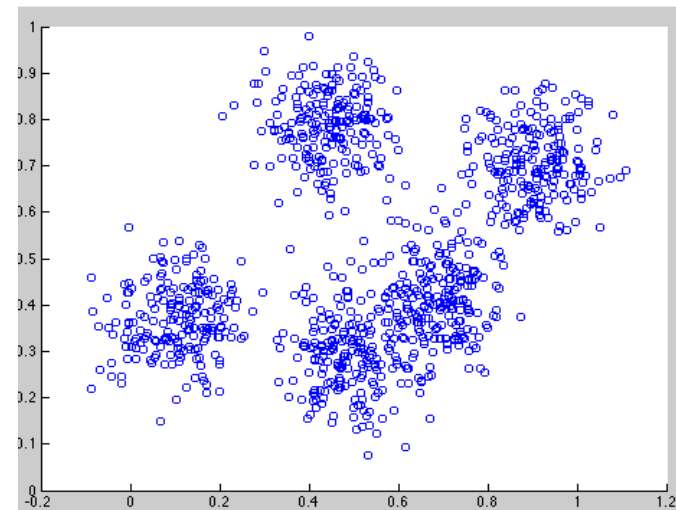


Phân cụm làm gì?

- I Nhóm các mục dựa trên các đặc điểm tương tự
 - ▶ Phân nhóm khách hàng có hành vi mua hàng giống nhau
 - ▶ Tự động nhóm tài liệu, trang web hoặc văn bản khác
 - ▶ Nhóm các mục tin tức dựa trên chủ đề
 - ▶ Phân loại thực vật và động vật
 - ▶ Phân loại các gen có chức năng tương tự



➡
Phân cụm



Phân cụm K-Means

- I Nhóm các mục dựa trên các đặc điểm tương tự
 - ▶ Nhập số cụm (K) và khởi tạo ngẫu nhiên các tâm
 - ▶ Chỉ định mọi điểm trong dữ liệu cho một trong những trung tâm gần nhất
 - ▶ Đối với mỗi trung tâm cụm, di chuyển trung tâm đến giữa cụm đó (trung bình)
 - ▶ Chỉ định lại từng điểm cho trung tâm gần nhất mới
 - ▶ Lặp lại từ bước 2 cho đến khi hội tụ
- I Các tham số chính của phương thức huấn luyện
 - ▶ Rdd -> tên tập dữ liệu
 - ▶ k -> số cụm
 - ▶ initializationMode -> cách khởi tạo các trung tâm

Phân cụm mẫu K-Means

```
from pyspark.mllib.clustering import KMeans, KMeansModel
from numpy import array
from math import sqrt

# Load and parse the data
data = sc.textFile("data/mllib/kmeans_data.txt")
parsedData = data.map(lambda line: array([Float(x) for x in line.split('')]))

# Build the model (cluster the data)
clusters = KMeans.train(parsedData, 2, maxIterations=10,
                        runs=10, initializationMode="random")

# Evaluate clustering by computing Within Set Sum of Squared Errors
def error(point):
    center = clusters.centers[clusters.predict(point)]
    return sqrt(sum([x**2 for x in (point - center)]))

WSSSE = parsedData.map(lambda point: error(point)).reduce(lambda x, y: x + y)
print("Within Set Sum of Squared Error = " + str(WSSSE))

# Save and load Model
clusters.save(sc, "myModelPath")
sameModel = KMeansModel.load(sc, "myModelPath")
```

Lọc cộng tác

- Lọc cộng tác thường được sử dụng cho các hệ thống gợi ý
- spark.mllib hiện hỗ trợ lọc cộng tác dựa trên mô hình, trong đó người dùng và sản phẩm được mô tả bằng một nhóm nhỏ các yếu tố tiềm ẩn có thể được sử dụng để dự đoán các mục bị thiếu
- spark.mllib sử dụng thuật toán bình phương nhỏ nhất xen kẽ (ALS) để tìm hiểu các yếu tố tiềm ẩn này
- Lọc cộng tác - Các kỹ thuật này nhằm mục đích điền vào các mục còn thiếu của ma trận liên kết mục người dùng
- Mllib hiện hỗ trợ lọc cộng tác dựa trên mô hình, trong đó người dùng và sản phẩm được mô tả bằng một nhóm nhỏ các yếu tố tiềm ẩn có thể được sử dụng để dự đoán các mục bị thiếu
- Mllib sử dụng bình phương tối thiểu xen kẽ (ALS)

	M items				
	5		1		
			3		
		5			
			2		
				5	

[Lab3]

Xây dựng ứng dụng đường ống Học máy



[Lab4]

Xây dựng mô hình hồi quy logistic



[Lab5]

Xây dựng một ứng dụng phân cụm



[Lab6]

Xây dựng mô hình hồi quy tuyến tính





SAMSUNG

Together for Tomorrow!
Enabling People

Education for Future Generations

©2021 SAMSUNG. All rights reserved.

Samsung Electronics Corporate Citizenship Office holds the copyright of book.

This book is a literary property protected by copyright law so reprint and reproduction without permission are prohibited.

To use this book other than the curriculum of Samsung innovation Campus or to use the entire or part of this book, you must receive written consent from copyright holder.