CHƯƠNG 4
PHÂN TÍCH DỮ LIỆU VỚI SQL

Giảng viên: Nguyễn Anh Thư

Khoa: Khoa học ứng dụng

NỘI DUNG BÀI HỌC

4.1 PHÂN TÍCH DỮ LIỆU LÀ GÌ?

4.2 HỌC CÓ GIÁM SÁT

- 4.2.1 Phân cụm: Naive Bayes
- 4.2.2 Hồi quy tuyến tính
- 4.2.3 Hồi quy logistic

4.3 HỌC KHÔNG GIÁM SÁT

- 4.3.1. Khoảng cách và phân cụm
- 4.3.2. Phân cụm KNN
- 4.3.3. Các quy tắc quan hệ

NỘI DUNG BÀI HỌC

4.4 TÍCH HỢP VỚI PYTHON

4.4.1. Làm việc với JSON/XML

4.4.2. Sử dụng pandas và SQLAlchemy trong Python.

4.4.3. Kết nối SQL với R qua DBI và dbplyr.

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

4.5.1. Bộ dữ liệu cho phân tích chuỗi thời gian

4.5.2. Bộ dữ liệu cho phân loại nhị phân

MỤC TIÊU BÀI HỌC

Sau khi học xong bài này, ta sẽ nắm được các vấn đề sau:

- Hiểu được khái niệm cơ bản về phân tích dữ liệu, vai trò của phân tích dữ liệu trong quá trình ra quyết định và cách SQL được sử dụng để hỗ trợ phân tích dữ liệu.
- Nắm được các phương pháp phân tích dữ liệu có giám sát và cách áp dụng chúng trong SQL: Phân cụm: Naive Bayes, Hồi quy tuyến tính, Hồi quy logistic
- Nắm được các phương pháp phân tích dữ liệu không giám sát và cách áp dụng chúng trong SQL: Khoảng cách và phân cụm, Phân cụm KNN, Các quy tắc quan hệ.
- Biết cách xử lý và phân tích dữ liệu dạng JSON/XML trong SQL và Python.
- Sử dụng pandas và SQLAlchemy trong Python và cách kết nối SQL với R thông qua các gói DBI và dbplyr để phân tích dữ liệu.

4.1 PHÂN TÍCH DỮ LIỆU LÀ GÌ?

- Là tập hợp các công cụ và kỹ thuật dùng để trích xuất thông tin từ dữ liệu. Thông tin này có thể là:
  - Các mẫu (patterns)
  - Công thức (formulas)
  - Quy tắc (rules) mô tả đặc điểm của tập dữ liệu
- Có nhiều loại thông tin có thể khai thác từ dữ liệu ➔ phân tích dữ liệu là một lĩnh vực rộng lớn, liên quan đến nhiều tài liệu, đặc biệt là về khai phá dữ liệu (Data Mining) và học máy (Machine Learning).
- Trong chương này, ta sẽ tập trung vào một số phương pháp đơn giản có thể được triển khai trong SQL. Hầu hết các kỹ thuật được đề cập sẽ áp dụng cho dữ liệu bảng.

4.1 PHÂN TÍCH DỮ LIỆU LÀ GÌ?

DỮ LIỆU DẠNG BẢNG

- Là dữ liệu có một tập hợp các bản ghi (records/ objects/ events/ observations), mỗi bản ghi được biểu diễn bởi n thuộc tính (attributes/ features).
- Nếu có một cột đặc biệt gọi là biến phụ thuộc (*dependent variable*) hoặc nhãn (*label*), chúng ta có thể dự đoán giá trị của nó từ các biến độc lập khác (*independent variables / predictor variables*).

4.1 PHÂN TÍCH DỮ LIỆU LÀ GÌ?

HỌC CÓ GIÁM SÁT

- Sử dụng khi dữ liệu có nhãn, dự đoán giá trị của biến phụ thuộc y từ các biến độc lập $\{x_i\}_{i=1,..,n}$, trong đó các giá trị x_i được sử dụng để huấn luyện thuật toán.
- "Giám sát" đề cập đến việc ta cung cấp cho thuật toán các ví dụ cụ thể (các đặc trưng của quan sát) về những gì ta muốn biết (các nhãn) để thuật toán học từ chúng.
  - Nếu nhãn y là phân loại (categorical) → Bài toán phân loại (classification).
  - Nếu nhãn y là số thực (numerical) → Bài toán hồi quy (regression).

4.1 PHÂN TÍCH DỮ LIỆU LÀ GÌ?

HỌC CÓ GIÁM SÁT

- Quá trình huấn luyện:
  - Chia tập dữ liệu:
    - Tập huấn luyện (training data): tập dữ liệu chứa cả các đặc trưng và nhãn tương ứng.
    - Tập kiểm tra (test data): tập dữ liệu chỉ chứa các đặc trưng mà không có nhãn.
    - Các dữ liệu nên được chia ngẫu nhiên để tránh cung cấp cho thuật toán bộ dữ liệu training bị chênh lệch giữa các nhãn.
  - Huấn luyện mô hình: Sử dụng tập training để cải thiện thuật toán.
  - Kiểm định mô hình: Sử dụng tập test để đánh giá hiệu suất của thuật toán.

4.1 PHÂN TÍCH DỮ LIỆU LÀ GÌ?

HỌC KHÔNG GIÁM SÁT

- Sử dụng khi bộ dữ liệu không có biến phụ thuộc cụ thể.
- Khám phá phá dữ liệu và tìm ra các mẫu trong đó mà không cần nhiều giả định → có thể được sử dụng như một phần của tiền xử lý dữ liệu vì nó giúp ta hiểu rõ hơn về tập dữ liệu
- Một số kỹ thuật học không giám sát phổ biến:
  - **Phân cụm (clustering):** các quan sát trong dữ liệu có thể được chia thành các nhóm/ phân cụm dựa trên sự tương đồng của chúng hay không.
  - **Luật kết hợp (association rules):** tìm mối quan hệ giữa các biến trong tập dữ liệu. VD nếu khách hàng mua hàng sản phẩm A thì có khả năng cao mua hàng sản phẩm B.
  - **Giảm chiều dữ liệu (Dimensionality Reduction):** tìm các yếu tố tiềm ẩn (latent factors) giúp biểu diễn dữ liệu theo cách gọn hơn nhưng vẫn giữ nguyên thông tin quan trọng.

4.1 PHÂN TÍCH DỮ LIỆU LÀ GÌ?

HỌC BÁN GIÁM SÁT

- Kết hợp dữ liệu có nhãn và không nhãn.

CÁC MÔ HÌNH TỔNG HỢP – ENSEMBLE MODELS

- Kết hợp nhiều mô hình để cải thiện hiệu suất.

ỨNG DỤNG SQL TRONG PHÂN TÍCH DỮ LIỆU: Thực hiện các phân tích cơ bản trước khi áp dụng các mô hình phức tạp hơn. Một số thao tác SQL hữu ích trong phân tích dữ liệu như:

- Tóm tắt dữ liệu: COUNT(), AVG(), SUM(), GROUP BY
- Lọc dữ liệu: WHERE, HAVING, LIMIT
- Trích xuất thông tin: JOIN, CASE WHEN, WINDOW FUNCTIONS
- Phân cụm cơ bản bằng SQL: Sử dụng k-means clustering hoặc phân cụm theo nhóm (GROUP BY)

4.2 HỌC CÓ GIÁM SÁT

- Trong phương pháp này, dữ liệu đã gán nhãn sẽ được chia thành các tập huấn luyện và kiểm thử → cách tốt nhất là chia ngẫu nhiên.
- Hầu hết các hệ thống đều có cách tạo số ngẫu nhiên, ví dụ RANDOM() trong Postgres và MySQL, hàm này sẽ tạo ra một số ngẫu nhiên trong khoảng từ 0 đến 1.
- Thông thường, phần lớn dữ liệu (từ 75% đến 90%) được dùng để huấn luyện, và phần còn lại (từ 25% đến 10%) được dùng để kiểm tra.

4.2 HỌC CÓ GIÁM SÁT

- Quy trình chung để chia dữ liệu thành tập huấn luyện và kiểm tra có thể được viết trong MySQL như sau:
  - **BƯỚC 1: Khởi tạo một số làm ngưỡng chia dữ liệu:**

```
CREATE TABLE new-data as
    SELECT *, random() as split
    FROM Data;
```

  - **BƯỚC 2: Lấy những quan sát có ngưỡng ≥ 0.1 làm tập training:**

```
CREATE TABLE training-data as
    SELECT *
    FROM new-data
    WHERE split >= .1;
```

4.2 HỌC CÓ GIÁM SÁT

- Quy trình chung để chia dữ liệu thành tập huấn luyện và kiểm tra có thể được viết trong MySQL như sau:
  - **BƯỚC 3: Lấy những quan sát có ngưỡng < 0.1 làm tập test:**

```
CREATE TABLE training-data as
    SELECT *
    FROM new-data
    WHERE split < .1;
```

(*) Hàm RANDOM() tạo một giá trị ngẫu nhiên trong mỗi lần nó được gọi, như bước 1 thì mỗi hàng sẽ ứng với một lần gọi hàm, dẫn đến mỗi hàng nhận được một giá trị ngẫu nhiên.

4.2 HỌC CÓ GIÁM SÁT

a. BÀI TOÁN PHÂN LOẠI:

Trong phân loại, mỗi bản ghi sẽ có n thuộc tính $\{x_i\}_{i=1,\dots,n}$ và thuộc về một trong các nhãn $\{y_j\}_{j=1,\dots,m}$.

Trong tập training, mỗi bản ghi sẽ có một thuộc tính để chỉ định nhãn của nó, giả sử gọi là y_{true}

$\Rightarrow$ cấu trúc của dữ liệu huấn luyện có dạng là $(x_1, x_2, \dots, x_n, y_{true})$.

4.2 HỌC CÓ GIÁM SÁT

a. BÀI TOÁN PHÂN LOẠI:

- Quá trình thực hiện phân loại gồm các bước như sau:
  - **Huấn luyện**: tạo dữ liệu training và huấn luyện thuật toán trên bộ đó.
  - **Kiểm tra**: chạy thuật toán đã huấn luyện để dự đoán nhãn cho từng bản ghi.
  - **Đánh giá**: so sánh nhãn dự đoán y_{pred} với nhãn thực tế y_{true} để đánh giá hiệu suất của thuật toán.

- Phân loại là một trong những thuật toán quan trọng trong Khoa học dữ liệu với nhiều thuật toán khác nhau, một trong những thuật toán đơn giản nhất là Naïve Bayes, có thể triển khai hiệu quả trong SQL.

4.2 HỌC CÓ GIÁM SÁT

a. BÀI TOÁN PHÂN LOẠI: NAÏVE BAYES

- Công thức tính trong Naive Bayes thường được dùng để tính xác suất hậu nghiệm, dựa trên Định lý Bayes:

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

Trong đó:

- $P(Y|X)$: Xác suất biến Y đúng với dữ liệu X.
- $P(X|Y)$: Xác suất dữ liệu X xảy ra khi biết Y.
- $P(Y)$: Xác suất ban đầu của Y, còn gọi là xác suất tiên nghiệm.
- $P(X)$: Xác suất ban đầu của X, hoạt động như một hệ số chuẩn hóa.

4.2 HỌC CÓ GIÁM SÁT

a. BÀI TOÁN PHÂN LOẠI: NAÏVE BAYES

- Công thức tính trong Naive Bayes thường được dùng để tính xác suất hậu nghiệm, dựa trên Định lý Bayes:

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

- Trong Naïve Bayes, giả thiết "naïve" là các đặc trưng độc lập với nhau, tức là:

$$P(X|Y) = P(x_1|Y) \cdot P(x_2|Y) \cdot \ldots \cdot P(x_n|Y)$$

⇒ Công thức Naive Bayes thường được đơn giản hóa thành $P(Y|X) \propto P(Y) \prod_{i=1}^{n} P(x_i|Y)$.

4.2 HỌC CÓ GIÁM SÁT

a. BÀI TOÁN PHÂN LOẠI: NAÏVE BAYES

- **VD1:** Bảng **RawData(id, age, prescription, astigmatic, tears, lens)** là bảng chứa thông tin về một số bệnh nhân với các vấn đề về thị lực. Trong đó:
  - id: mã bệnh nhân.
  - Age: tuổi bệnh nhân.
  - Prescription: chẩn đoán bệnh.
  - Astigmatic: bệnh loạn thị.
  - Tears: tình trạng nước mắt.
  - Lens: tình trạng thủy tinh thể.
- **Yêu cầu:** hãy dự đoán tình trạng của “lens” thông qua các thuộc tính “age”, “prescription”, “astigmatic” và “tears”.

4.2 HỌC CÓ GIÁM SÁT

a. BÀI TOÁN PHÂN LOẠI: NAÏVE BAYES

GỢI Ý:

- **B1:** chia dữ liệu thành hai tập: training và testing.
- **B2:** tính xác suất tiên nghiệm cho các nhãn.
- **B3**: tính xác suất có điều kiện cho mỗi thuộc tính và nhãn.
- **B4**: sử dụng xác suất nhãn và các thuộc tính để tính nhãn mà từng bản ghi trong tập training thuộc về.
- **B5**: chọn ra nhãn có xác suất cao nhất.
- **B6**: đo lường kết quả: so sánh với nhãn thực tế.

4.2 HỌC CÓ GIÁM SÁT

b. BÀI TOÁN HỒI QUY TUYẾN TÍNH:

- **So sánh bài toán hồi quy với bài toán phân loại:**

	REGRESSION	CLASSIFICATION
GIỐNG	Sử dụng các biến độc lập (predictors) để dự đoán biến phụ thuộc (response variable).	
KHÁC	Biến phụ thuộc là số liên tục (numerical).	Biến phụ thuộc là phân loại (categorical).
	Dự đoán giá trị số.	Dự đoán nhãn, trong đó nhãn có thể là số khi các nhãn phân loại được chuyển đổi thành số thông qua biến giả (dummy variables).

4.2 HỌC CÓ GIÁM SÁT

b. BÀI TOÁN HỒI QUY TUYẾN TÍNH:

- Hồi quy tuyến tính là dạng hồi quy đơn giản nhất, trong đó n biến độc lập $X = \{x_i\}_{i=1,\dots,n}$ có mối quan hệ tuyến tính với một biến phụ thuộc Y thông qua công thức tổng quát:

$$Y = f(X) = \beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n$$

→ β_i là các hệ số ước lượng.

→ Các hệ số ước lượng ban đầu sẽ được khởi tạo ngẫu nhiên, sau đó sẽ được tối ưu sao cho sai số giữa giá trị dự đoán và giá trị thực tế là nhỏ nhất.

4.2 HỌC CÓ GIÁM SÁT

b. BÀI TOÁN HỒI QUY TUYẾN TÍNH:

- Tại mỗi bản ghi r, cột $r.Y$ là cột nhãn thực tế Y đã biết trước và sẽ được so sánh với giá trị của hàm $f(r.X)$. Khi đó sai số dự đoán của r sẽ là $r.Y - f(r.X)$ vì ta đang tính toán trên các giá trị số.
- Muốn tối thiểu hóa sai số cho toàn bộ dữ liệu nên thông thường, sai số sẽ được biểu diễn dưới dạng *Tổng sai số bình phương − sum of square differences (SSE)*:

$$SSE = \sum_i (f(r_i.X) - r_i.Y)^2 = \sum_i (\beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n - r_i.Y)^2$$

4.2 HỌC CÓ GIÁM SÁT

b. BÀI TOÁN HỒI QUY TUYẾN TÍNH:

- Tối ưu hóa các tham số β_i sao cho SSE là nhỏ nhất:
  - **Cách 1:**

$$\beta_1 = \frac{\sum_i (r_i.X - \bar{X})(r_i.Y - \bar{Y})}{\sum_i (r_i.X - \bar{X})^2} \qquad (1)$$

$$\beta_0 = \bar{Y} - \beta_1 \bar{X} \qquad (2)$$

Trong đó: $\bar{X}$ là GTTB của cột X

$\bar{Y}$ là GTTB của cột Y.

4.2 HỌC CÓ GIÁM SÁT

b. BÀI TOÁN HỒI QUY TUYẾN TÍNH:

- Tối ưu hóa các tham số β_i sao cho SSE là nhỏ nhất:
  - **Cách 1:**

$$\beta_1 = \frac{\sum_i (r_i.X - \bar{X})(r_i.Y - \bar{Y})}{\sum_i (r_i.X - \bar{X})^2} \qquad (1)$$

$$\beta_0 = \bar{Y} - \beta_1 \bar{X} \qquad (2)$$

➔ Từ (1) có thể liên hệ tới hiệp phương sai của (X, Y), phương sai của X như sau:

$$\beta_1 = \frac{Cov(X,Y)}{Var(X)} = Corr(X,Y).\frac{stdev(Y)}{stdev(X)}$$

4.2 HỌC CÓ GIÁM SÁT

b. BÀI TOÁN HỒI QUY TUYẾN TÍNH:

- Tối ưu hóa các tham số β_i sao cho SSE là nhỏ nhất:
  - **Cách 1:**

$$\beta_1 = \frac{\sum_i (r_i.X - \bar{X})(r_i.Y - \bar{Y})}{\sum_i (r_i.X - \bar{X})^2} \qquad (1)$$

- $\sum_i (\boldsymbol{r_i.X} - \bar{\boldsymbol{X}})$: tổng bình phương của sự sai khác giữa từng giá trị trong X và giá trị trung bình $\bar{X}$ → phương sai của X, đo lường mức độ phân tán của các giá trị xung quanh giá trị trung bình.
- $\sum_i (\boldsymbol{r_i.X} - \bar{\boldsymbol{X}})(\boldsymbol{r_i.Y} - \bar{\boldsymbol{Y}})$: tổng của tích các độ lệch của X và Y so với giá trị trung bình của chúng → tích các phương sai hay tổng các tích của sự sai khác giữa X và Y so với GTTB của chúng, đo lường mối quan hệ tuyến tính giữa hai thuộc tính X và Y. Nếu giá trị dương, X và Y có xu hướng tăng cùng nhau; nếu âm, chúng có xu hướng ngược chiều.

4.2 HỌC CÓ GIÁM SÁT

b. BÀI TOÁN HỒI QUY TUYẾN TÍNH:

- Tối ưu hóa các tham số β_i sao cho SSE là nhỏ nhất:
  - **Cách 2:**

$$\beta_1 = \frac{\sum_i (r_i.X * r_iY)(N\bar{X}\bar{Y})}{\sum_i r_i.X^2 - N\bar{X}^2} \qquad (3)$$

$$\beta_0 = \frac{1}{N}\left(\sum_i r_i.Y - \beta_1 \sum_i r_i.A\right) \qquad (4)$$

Trong đó: $\bar{X}$: GTTB của cột X

$\bar{Y}$: GTTB của cột Y.

N : tổng số lượng các phần tử trong dữ liệu.

4.2 HỌC CÓ GIÁM SÁT

b. BÀI TOÁN HỒI QUY TUYẾN TÍNH:

- Tối ưu hóa các tham số β_i sao cho SSE là nhỏ nhất:
  - **Cách 2:**

$$\beta_1 = \frac{\sum_i (r_i.X * r_iY)(N\bar{X}\bar{Y})}{\sum_i r_i.X^2 - N\bar{X}^2} \quad (3)$$

$$\beta_0 = \frac{1}{N}\left(\sum_i r_i.Y - \beta_1 \sum_i r_i.A\right) \quad (4)$$

- $\sum_i(\boldsymbol{r_i.X})$: tổng tất cả các giá trị của thuộc tính X trong tập dữ liệu
- $\sum_i(\boldsymbol{r_i.X} * \boldsymbol{r_iY})$: Với mỗi hàng (bản ghi), nhân giá trị của X với giá trị của Y, sau đó cộng tất cả các kết quả lại. Đây là một phép toán quan trọng trong các tính toán liên quan đến hiệp phương sai và hồi quy.
- $\sum_i(\boldsymbol{r_i.X^2})$: Bình phương từng giá trị của X rồi cộng tất cả lại.

4.2 HỌC CÓ GIÁM SÁT

b. BÀI TOÁN HỒI QUY TUYẾN TÍNH:

- Tối ưu hóa các tham số β_i sao cho SSE là nhỏ nhất:
  - **Cách 2:**

$$\beta_1 = \frac{\sum_i (r_i.X * r_iY)(N\bar{X}\bar{Y})}{\sum_i r_i.X^2 - N\bar{X}^2} \quad (3) \qquad \beta_0 = \frac{1}{N}\left(\sum_i r_i.Y - \beta_1 \sum_i r_i.A\right) \quad (4)$$

- Từ công thức (3) và (4) ta có:

```
SELECT ((SB*SAA)-(SA*SAB)) / ((N*(SAA))- (SA*SA)) AS intercept,
            ((N * SAB)- (SA * SB)) /  ((N * SAA)- (SA * SA)) AS slope
FROM (SELECT sum(A) AS SA, sum(B) AS SB, sum(A*A) AS SAA, sum(A*B) AS SAB, sum(*) AS N
            FROM Data);
```

4.2 HỌC CÓ GIÁM SÁT

b. BÀI TOÁN HỒI QUY TUYẾN TÍNH:

- **VD với bài toán hồi quy tuyến tính đơn giản**: Bảng **Data (Id, Address, size, price, num_beds, num_baths)** mô tả dữ liệu bất động sản, trong đó:
  - Id, Address: ID và địa chỉ ứng với từng bất động sản.
  - Size: diện tích bất động sản, đơn vị mét vuông.
  - Price: giá nhà trong lần bán cuối cùng.
  - Num-beds, num-baths: số lượng phòng ngủ và số lượng phòng tắm.

Giả định rằng giá nhà (price) ảnh hưởng trực tiếp với diện tích (size), ta sử dụng hồi quy tuyến tính đơn giản với size làm biến dự đoán và price làm biến cần dự đoán.

4.2 HỌC CÓ GIÁM SÁT

b. BÀI TOÁN HỒI QUY TUYẾN TÍNH:

- **VD với bài toán hồi quy tuyến tính phức tạp:** Trường hợp dự đoán dựa trên nhiều biến khá phức tạp nhưng đối với 2 biến dự đoán, ta vẫn có thể thực hiện được trên các câu truy vấn. Ta có biểu thức:

$$B = \alpha_0 + \alpha_1 A_1 + \alpha_2 A_2,$$

- Với A_1, A_2 là các thuộc tính dự đoán và B là biến phụ thuộc; α_i là các tham số học từ dữ liệu.
- Để tính toán các tham số này, ta thực hiện trong 3 bước như sau:

4.2 HỌC CÓ GIÁM SÁT

b. BÀI TOÁN HỒI QUY TUYẾN TÍNH:

- **VD với bài toán hồi quy tuyến tính phức tạp:** Trường hợp dự đoán dựa trên nhiều biến khá phức tạp nhưng đối với 2 biến dự đoán, ta vẫn có thể thực hiện được trên các câu truy vấn.
  - Bước 1: Tính $SUM(A_i)$, $SUM(A_i^2)$, $SUM(B)$, $SUM(A_i * B)$, $SUM(A_i * A_j)$
  - Bước 2: Sử dụng các kết quả trên để tính các biểu thức sau:
    - $SA_1^2 = SUM(A_1^2) - \frac{SUM(A_1)^2}{n}$;
    - $SA_2^2 = SUM(A_2^2) - \frac{SUM(A_2)^2}{n}$;
    - $SA_1B = SUM(A_1 \cdot B) - \frac{SUM(A_1)SUM(B)}{n}$;
    - $SA_2B = SUM(A_2 \cdot B) - \frac{SUM(A_2)SUM(B)}{n}$;
    - $SA_1A_2 = SUM(A_1 \cdot A_2) - \frac{SUM(A_1)SUM(A_2)}{n}$;

4.2 HỌC CÓ GIÁM SÁT

b. BÀI TOÁN HỒI QUY TUYẾN TÍNH:

- **VD với bài toán hồi quy tuyến tính phức tạp:** Trường hợp dự đoán dựa trên nhiều biến khá phức tạp nhưng đối với 2 biến dự đoán, ta vẫn có thể thực hiện được trên các câu truy vấn.
  - Bước 3: Tính

$$\alpha_1 = \frac{(SA_2^2)(SA_1B) - (SA_1A_2 \cdot SA_2B)}{(SA_1^2)(SA_2^2) - (SA_1A_2)^2},$$

$$\alpha_2 = \frac{(SA_1^2)(SA_2B) - (SA_1A_2 \cdot SA_2B)}{(SA_1^2)(SA_2^2) - (SA_1A_2)^2},$$

$$\alpha_0 = SUM(B - \alpha_1A_1 - \alpha_2A_2).$$

4.2 HỌC CÓ GIÁM SÁT

b. BÀI TOÁN HỒI QUY TUYẾN TÍNH:

- Hồi quy tuyến tính (Linear Regression) là một kỹ thuật đơn giản và phổ biến, thường được thử đầu tiên trên nhiều tập dữ liệu. Tuy nhiên, nó có một số hạn chế nghiêm trọng cần lưu ý:
  - Luôn trả về một kết quả, nhưng kết quả này có thể không chính xác nếu sai số vẫn còn lớn.
  - Giả định mạnh về:
    - Mối quan hệ tuyến tính: Hồi quy tuyến tính giả định rằng mối quan hệ giữa biến độc lập và biến phụ thuộc là tuyến tính, trong khi thực tế mối quan hệ có thể phức tạp hơn → Không phù hợp với dữ liệu có mối quan hệ phi tuyến.
    - Phân phối của sai số: phải tuân theo phân phối chuẩn, độc lập và có phân phối đồng nhất nhưng không phải lúc nào cũng đúng. Nếu giả định này không đúng, kết quả hồi quy sẽ không đáng tin cậy.
  - Hồi quy tuyến tính không cảnh báo nếu mối quan hệ giữa các biến không tồn tại hoặc không tuyến tính (không cảnh báo khi mô hình không phù hợp) → cần kiểm tra xem kết quả có đáng tin cậy hay không.
  - Đánh giá kết quả: Để biết kết quả hồi quy có tốt hay không, cần kiểm tra các giả định và đánh giá độ chính xác của mô hình thông qua các chỉ số như R^2, sai số trung bình (MSE), hoặc phân tích phần dư (residual analysis).

4.2 HỌC CÓ GIÁM SÁT

b. BÀI TOÁN HỒI QUY TUYẾN TÍNH – ĐO LƯỜNG SAI SỐ

- Tổng sai số bình phương - Sum Of Square Differences (SSE):

$$SSE = \sum_i (f(r_i.X) - r_i.Y)^2 = \sum_i (\beta_0 + \beta_1 x_1 + \cdots + \beta_n x_n - r_i.Y)^2$$

- Tổng bình phương hồi quy - Regression Sum of Squares (SSR):

$$SSR = \sum_i (f(r_i.X) - \bar{Y})^2$$

4.2 HỌC CÓ GIÁM SÁT

b. BÀI TOÁN HỒI QUY TUYẾN TÍNH – ĐO LƯỜNG SAI SỐ

- Tổng bình phương sai số - Total Sum of Squares (SSTO):

$$SSTO = \sum_i (r_i.Y - \bar{Y})^2$$

- Mối quan hệ giữa các đại lượng:

$$SSTO = SSR + SSE$$

$$R^2 = \frac{SSR}{SSTO} = 1 - \frac{SSE}{SSTO} \in [0,1]$$

→ R^2 : tỷ lệ phương sai của biến phụ thuộc Y được giả thích bởi biến độc lập X.

4.2 HỌC CÓ GIÁM SÁT

b. BÀI TOÁN HỒI QUY TUYẾN TÍNH – ĐO LƯỜNG SAI SỐ

$$R^2 = \frac{SSR}{SSTO} = 1 - \frac{SSE}{SSTO} \in [0,1]$$

- $\boldsymbol{R^2}$ **cao** (gần 1) → mô hình hồi quy giải thích được phần lớn phương sai của biến phụ thuộc, nghĩa là mô hình phù hợp tốt với dữ liệu.
- $\boldsymbol{R^2}$ **thấp** (gần 0) → mô hình không giải thích được nhiều phương sai của biến phụ thuộc.
- Trong bài toán LR thì điều này có nghĩa là các trọng số (slope và interrcept) phù hợp với dữ liệu.
- Hệ số tương quan $r = \sqrt{R^2}$.

4.2 HỌC CÓ GIÁM SÁT

b. BÀI TOÁN HỒI QUY TUYẾN TÍNH – MỘT SỐ LƯU Ý

- Cần chuẩn hóa dữ liệu trước khi sử dụng: Nếu một thuộc tính có giá trị quá lớn so với các thuộc tính khác, mô hình sẽ tập trung vào việc giảm sai số cho thuộc tính đó, làm ảnh hưởng đến độ chính xác tổng thể.
- Cần kiểm tra và xử lý ngoại lệ trước khi áp dụng hồi quy tuyến tính: Hồi quy tuyến tính nhạy cảm với ngoại lệ (outliers). Một giá trị quá lớn có thể làm thay đổi hướng của đường hồi quy, khiến mô hình không còn đại diện đúng cho dữ liệu.
- Giải pháp: Kiểm tra và xử lý ngoại lệ trước khi huấn luyện mô hình, có thể bằng log transformation, IQR filtering, hoặc robust regression.

4.2 HỌC CÓ GIÁM SÁT

c. BÀI TOÁN HỒI QUY LOGISTIC

- Logistic regression là một thuật toán phân loại (classification) dù có tên gọi là "hồi quy”.
- Cách thức hoạt động: dựa trên nguyên tắc của hồi quy tuyến tính.
- Ý tưởng chính của logistic regression: Ước lượng một mô hình hồi quy tuyến tính, nhưng diễn giải kết quả dưới dạng xác suất để phân loại một đối tượng vào một trong hai lớp (0 hoặc 1).
- Ví dụ, mô hình có thể dự đoán một người có rủi ro vay nợ cao hay thấp, sinh viên có đậu kỳ thi hay không, hoặc bệnh nhân có sống sót sau một thủ thuật y tế hay không.

4.2 HỌC CÓ GIÁM SÁT

c. BÀI TOÁN HỒI QUY LOGISTIC

- Xác suất để một điểm dữ liệu thuộc lớp 1 được biểu diễn dưới dạng tỷ lệ odds, được tính theo công thức:

$$Odds = \frac{p}{1-p}$$

- Trong đó p là xác suất thuộc lớp 1, (1-p) là xác suất thuộc lớp 0.
- Hồi quy tuyến tính sẽ tính xấp xỉ $P(1|X_1, \dots, X_n)$ trong khi logistic sẽ tính log của odds, nghĩa là tính:

$$\log \frac{P(1|X_1, \dots, X_n)}{1 - P(1|X_1, \dots, X_n)} = \log \frac{p}{1-p} = \beta_0 + \cdots + \beta_n X_n$$

Trong đó β_i là các tham số ta cần ước lượng.

4.2 HỌC CÓ GIÁM SÁT

c. BÀI TOÁN HỒI QUY LOGISTIC

- Từ các β_i ta có thể tìm p một cách đơn giản:

  - Tính các tham số β_i thông qua Linear Regression rồi áp vào công thức trên → công thức hàm Sigmoid.
  - Sigmoid cho các giá trị nằm trong đoạn [0, 1] → thường dùng trong bài toán nhị phân:

    →Nếu $p < 0.5$ thì thuộc một lớp 0 (hoặc 1).

    →Nếu $p \geq 0.5$ thì thuộc lớp còn lại.

4.2 HỌC CÓ GIÁM SÁT

c. BÀI TOÁN HỒI QUY LOGISTIC

- **VD:** Giả sử ta muốn biết về tập dữ liệu bất động sản là liệu một ngôi nhà được bán với giá cao hay không (> $300,000) hoặc với giá thấp (≤ $300,000). Hãy áp dụng hồi quy logistic để giải quyết vấn đề này, chỉ sử dụng kích thước (size) làm biến dự đoán.
- **Gợi ý:**
  1. Thêm một biến **classprice** vào tập dữ liệu với giá trị 1 (cho giá cao) và 0 (cho giá thấp), tùy thuộc vào việc giá nhà có trên hoặc dưới ngưỡng $300,000 hay không.
  2. Chọn ngẫu nhiên một phần dữ liệu làm tập huấn luyện và áp dụng hồi quy tuyến tính để thu được hệ số góc (slope) và hệ số chặn (intercept).
  3. Đưa các giá trị này vào hàm sigmoid như đã trình bày ở trên.

4.3 HỌC KHÔNG GIÁM SÁT

	HỌC CÓ GIÁM SÁT	HỌC KHÔNG GIÁM SÁT
GIỐNG	• Mục đích: tìm hiểu quy luật, mô hình từ dữ liệu để đưa ra dự đoán hoặc khám phá cấu trúc tiềm ẩn. • Dữ liệu: tiền xử lý trước khi đưa vào mô hình.	
KHÁC	Dữ liệu	Có nhãn (có đầu vào và đầu ra mong muốn).	Không có nhãn (chỉ có đầu vào, không có đầu ra)
	Mục tiêu	Học từ dữ liệu có nhãn để dự đoán kết quả mới	Tìm hiểu cấu trúc ẩn trong dữ liệu
	Cách hoạt động	Dùng dữ liệu đầu vào và nhãn để huấn luyện mô hình	Nhóm dữ liệu hoặc giảm chiều dữ liệu để tìm ra mối quan hệ tiềm ẩn
	Ứng dụng chính	Phân loại (Classification) Hồi quy (Regression)	Phân cụm (Clustering) Giảm chiều dữ liệu (Dimensionality Reduction) Phát hiện bất thường (Anomaly Detection)

4.3 HỌC KHÔNG GIÁM SÁT

- Ưu điểm:
  - Không yêu cầu dữ liệu có nhãn, nên có thể áp dụng cho bất kỳ tập dữ liệu nào.
  - Hữu ích trong việc hiểu và khám phá dữ liệu, đặc biệt trong giai đoạn khám phá dữ liệu. Các kỹ thuật như phân cụm (clustering) và giảm chiều dữ liệu (dimensionality reduction) thường được xếp vào nhóm Khai phá dữ liệu - EDA.
- Thách thức: Không biết trước số lượng hoặc đặc điểm của các cụm → thường được coi là yếu hơn so với học có giám sát do thiếu thông tin từ nhãn.

4.3 HỌC KHÔNG GIÁM SÁT

a. KHOẢNG CÁCH

- Nhiều phương pháp phân cụm dựa vào việc xác định độ tương đồng (hoặc khác biệt) giữa hai bản ghi trong tập dữ liệu. Các bản ghi có độ tương đồng cao được nhóm lại thành một cụm, trong khi các bản ghi khác biệt sẽ thuộc các cụm khác nhau.

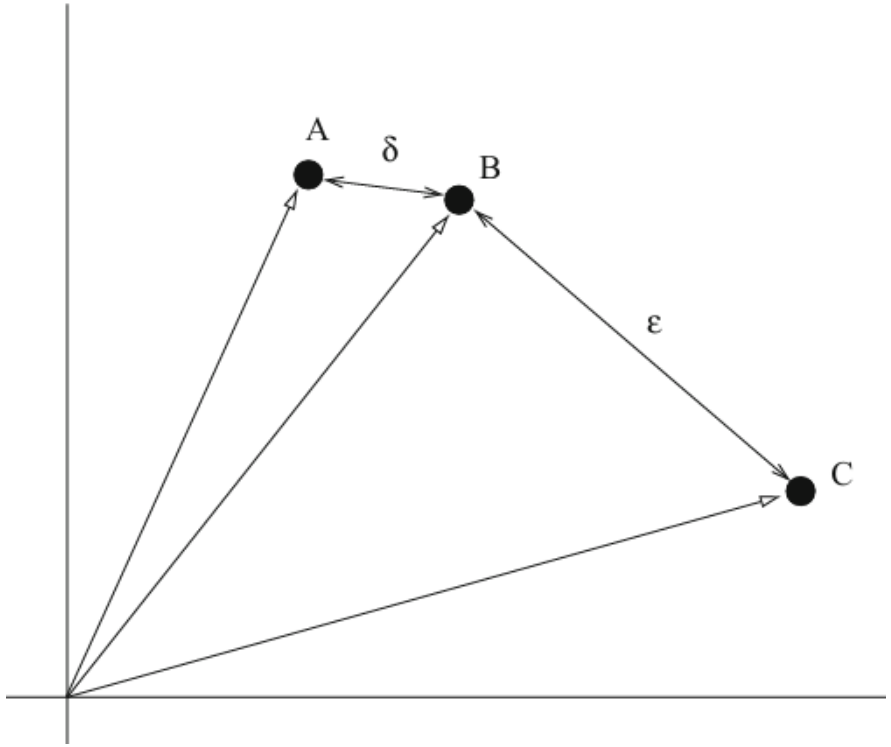
  ➔ Sử dụng hàm khoảng cách – là hàm gán một giá trị số cho mỗi cặp bản ghi – đo lường mức độ khác biệt giữa.

  ➔ Ý tưởng trực quan là nếu hai điểm dữ liệu có khoảng cách nhỏ, chúng có nhiều điểm chung và thuộc cùng một cụm. Ngược lại, nếu khoảng cách lớn, chúng thuộc các cụm khác nhau.

4.3 HỌC KHÔNG GIÁM SÁT

a. KHOẢNG CÁCH

- Là phương pháp trừu tượng hóa khoảng cách trong hình học, trong đó các bản ghi được xem như các điểm trong không gian, giúp đánh giá mức độ tương đồng giữa chúng.



4.3 HỌC KHÔNG GIÁM SÁT

a. KHOẢNG CÁCH

- Khoảng cách giữa các bản ghi/ điểm dữ liệu thu được bằng cách kết hợp khoảng cách giữa các thuộc tính.
  - **Khoảng cách cho các thuộc tính số** là sự sai khác giữa x_1 và x_2, là $|x_1 - x_2|$.
  - **Khoảng cách cho các thuộc tính phân loại**: khó khăn trong việc định nghĩa một khoảng cách có ý nghĩa. Trong một vài ngữ cảnh, sự tương đồng về chuỗi (string similarity) hoạt động tốt, nhưng trong nhiều trường hợp thì khoảng cách duy nhất được sử dụng cho các thuộc tính phân loại là khoảng cách trivial:

$$D(s_1, s_2) = \begin{cases} 0, & nếu\ s_1 = s_2 \\ 1, & ngược\ lại \end{cases}$$

4.3 HỌC KHÔNG GIÁM SÁT

a. KHOẢNG CÁCH

- Kết hợp các khoảng cách riêng lẻ thành một khoảng cách duy nhất giữa hai bản ghi → sử dụng **khoảng cách Minkowski**: với hai điểm dữ liệu r_1 và r_2 với các đặc trưng $(X_1, \dots, X_n)$, khoảng cách Minkowski được xác định bởi công thức:

$$d(r_1, r_2) = \sqrt[k]{\sum_{i=1,\dots,n} (r_1.X_i - r_2.X_i)^k}$$

- Trong đó:
- $r_i.X_j$: giá trị đặc trưng X_j của bản ghi r_i.
- $K > 0$: tham số.

4.3 HỌC KHÔNG GIÁM SÁT

a. KHOẢNG CÁCH

- **Khoảng cách Manhattan** là khoảng cách Minkowski với $k = 1$.
- **Khoảng cách Euclidean** là Minkowski với $k = 2$ → hữu ích, có 2 vấn đề chính:
  - **Yêu cầu chuẩn hóa dữ liệu**: Nếu các thuộc tính có đơn vị đo khác nhau (VD: diện tích hàng trăm m2, số phòng ngủ chỉ từ 1 đến 5) → thuộc tính có giá trị lớn hơn sẽ chi phối khoảng cách, làm sai lệch kết quả.
  - **Hoạt động kém khi sự tương quan giữa các thuộc tính cao**: Nếu các thuộc tính phụ thuộc lẫn nhau, khoảng cách Euclidean có thể đánh giá sai mức độ khác biệt giữa các điểm dữ liệu.

  ⇒ **Khoảng cách Mahalanobis**: *chuẩn hóa giá trị* dựa trên *ma trận hiệp phương sai*, điều chỉnh ảnh hưởng của các thuộc tính có tương quan cao, đảm bảo khoảng cách phản ánh đúng sự khác biệt: "*Nếu hai thuộc tính có hiệp phương sai cao (tương quan mạnh), sự khác biệt giữa chúng được giảm. Nếu hiệp phương sai thấp (ít tương quan), sự sai khác là đáng kể".*

4.3 HỌC KHÔNG GIÁM SÁT

a. KHOẢNG CÁCH

- Mahalanobis khó tính toán trong SQL vì yêu cầu ma trận hiệp phương sai và nghịch đảo của nó. Cách tiếp cận đơn giản hơn là chuẩn hóa mỗi thuộc tính bằng độ lệch chuẩn của nó, tạo ra một phiên bản xấp xỉ của khoảng cách Mahalanobis → **khoảng cách Euclidean chuẩn hóa**.

$$d(r_1, r_2) = \sum_{i=1,\dots,n} \sqrt{\frac{(r_1.X_i - r_2.X_i)^2}{stddev(X_i)}}$$

4.3 HỌC KHÔNG GIÁM SÁT

a. KHOẢNG CÁCH

- **Cosine Similarity:** đo độ tương đồng giữa 2 điểm dữ liệu bằng góc giữa 2 vector trong không gian nhiều chiều. Nguyên tắc hoạt động:
  - Biểu diễn mỗi điểm dữ liệu như một vector bắt đầu từ gốc tọa độ. Hai điểm dữ liệu tạo thành hai vector có chung gốc tọa độ, tạo ra một góc giữa chúng.
  - Tính độ tương đồng cosin thông qua góc giữa hai vector:

$$Cosine\ Similarity = \frac{\sum_i^n r_1.X_i * r_2.X_i}{\sqrt{\sum_i^n (r_1 X_i)^2} * \sqrt{\sum_i^n (r_2 X_i)^2}}$$

    - Góc nhỏ → hai điểm rất giống nhau → Cosine ≈ 1.
    - Góc 90° → hai điểm không liên quan → Cosine ≈ 0.
    - Góc 180° → hai điểm đối lập nhau → Cosine ≈ -1.

4.3 HỌC KHÔNG GIÁM SÁT

a. KHOẢNG CÁCH

VD: Cho bảng dữ liệu Data(ID, Num1, Num2) như sau. Hãy tính khoảng cách Euclidean và Manhattan giữa các điểm dữ liệu:

ID	Num1	Num2
1	3.5	7.2
2	5.1	8.4
3	7.8	6.9
4	9.0	10.2
5	11.3	4.5
6	13.7	12.8

4.3 HỌC KHÔNG GIÁM SÁT

a. KHOẢNG CÁCH

VD: Cho bảng dữ liệu Data(ID, Num1, Num2) như sau. Hãy tính khoảng cách Euclidean và Manhattan giữa các điểm dữ liệu:

```
select sqrt(pow(d1.Num1-d2.Num1,2) +
       pow(d1.Num2-d2.Num2,2)) as Euclidean,
       abs((d1.Num1-d2.Num1) + (d1.Num2-
d2.Num2)) as Manhattan
from Data d1, Data d2;
```

ID	Num1	Num2
1	3.5	7.2
2	5.1	8.4
3	7.8	6.9
4	9.0	10.2
5	11.3	4.5
6	13.7	12.8

4.3 HỌC KHÔNG GIÁM SÁT

a. KHOẢNG CÁCH

- **Phân cụm K-Means:** Nhóm các điểm dữ liệu thành k cụm (clusters) sao cho:
  - Khoảng cách giữa các điểm trong cùng một cụm là nhỏ nhất.
  - Khoảng cách giữa các điểm thuộc hai cụm khác nhau là lớn nhất.
- Cách thức hoạt động:
  - B1. Chọn số cụm (k) cần phân chia.
  - B2. Khởi tạo ngẫu nhiên k điểm làm tâm cụm (centroid).
  - B3. Gán mỗi điểm dữ liệu vào cụm gần nhất dựa trên khoảng cách đến các centroid.
  - B4. Cập nhật lại vị trí centroid bằng cách tính trung bình (mean) tất cả các điểm trong cụm.
  - B5. Lặp lại bước 3 và 4 cho đến khi các cụm không thay đổi (hội tụ).
- Để đánh giá độ chặt chẽ của cụm → average/max/ min(distance) giữa các điểm trong một cụm.

4.3 HỌC KHÔNG GIÁM SÁT

a. KHOẢNG CÁCH

- **Phân cụm K-Means:** Là thuật toán **lặp**, cách duy nhất để triển khai trong SQL là sử dụng đệ quy.

Bên cạnh bảng dữ liệu gốc Data(point-id,...), ta cũng cần các bảng sau:

- **Bảng Centroids (iteration, cluster-id, mean):** Lưu trữ tâm cụm (centroid) qua các lần lặp, bao gồm cả số thứ tự lặp mà giá trị trung bình được tính. VD chọn ngẫu nhiên k điểm từ Data làm centroid ban đầu (iteration = 1), ta có (1, 1, point1), (1, 2, point2), ..., (1, k, pointk).
- **Bảng Clustering (iteration, point-id, cluster-id):** Gán cụm cho từng điểm qua các lần lặp. Ban đầu, tất cả các điểm dữ liệu được gán iteration = 0 và cluster-id = NULL.
- **Bảng Distances(iteration, point-id, cluster-id, distance)** (bảng phụ trợ, không bắt buộc): Lưu khoảng cách giữa mỗi điểm đến tâm cụm để xác định điểm thuộc cụm nào. Bảng này có thể được khởi tạo là bảng trống.

4.3 HỌC KHÔNG GIÁM SÁT

a. KHOẢNG CÁCH

- **Phân cụm K-Means:** Là thuật toán **lặp**, cách duy nhất để triển khai trong SQL là sử dụng đệ quy.
- Quá trình lặp/ cập nhật cụm: Với mỗi điểm, tính khoảng cách giữa các điểm đến các tâm cụm ở lần lặp hiện tại → cập nhật cụm (Clustering) với từng điểm dữ liệu (cluster-id) dựa vào tâm cụm gần nhất → Cập nhật lại các tâm cụm theo các điểm mới bằng cách tính lại GTTB của từng cụm, lưu vào Centroids với iteration+1 → Dùng đệ quy (SQL recursion) để lặp bước 1 và 2 cho đến khi cụm không thay đổi.

```
INSERT INTO Distances
SELECT C.iter+1, D.point-id, C.cluster-id, d(D.attributes, C.mean)
FROM Data D, Centroids C
WHERE iter = (SELECT max(iter) FROM Centroids);
```

4.3 HỌC KHÔNG GIÁM SÁT

a. KHOẢNG CÁCH

- **Phân cụm K-Means:** Là thuật toán **lặp**, cách duy nhất để triển khai trong SQL là sử dụng đệ quy.
- Lưu ý: chọn các tâm cụm ở vòng lặp mới nhất (iteration cao nhất trong bảng centroids) để tính khoảng cách. Khi bắt đầu một vòng lặp mới hay cập nhật bảng lưu centroids, tăng giá trị iteration lên 1 (vd: từ 5 → 6) để biểu thị rằng đang thực hiện vòng lặp mới → giúp SQL truy vấn dữ liệu của lần lặp gần nhất khi tính toán lại khoảng cách và cập nhật cụm. Sau khi cập nhật khoảng cách, mỗi điểm dữ liệu được gán vào cụm có khoảng cách nhỏ nhất đến centroid mới nhất.

4.3 HỌC KHÔNG GIÁM SÁT

a. KHOẢNG CÁCH

- **Phân cụm K-Means: Chọn tâm cụm ở vòng lặp mới**

```
INSERT INTO TABLE Centroids
    SELECT iter+1, cluster-id, value
    FROM ( SELECT cluster-id, avg(dist) as value
        FROM Distances D, Clustering C
        WHERE D.cluster-id = C.cluster-id
        GROUP BY cluster-id) as TEMP, Centroids as C
    WHERE C.cluster-id = TEMP.cluster-id;
```

4.3 HỌC KHÔNG GIÁM SÁT

a. KHOẢNG CÁCH

- **Phân cụm K-Means: Cập nhật tâm cụm ở vòng lặp mới**

```
INSERT INTO Clustering
    SELECT C.iter+1, D.point-id, D.cluster-id
    FROM Distances D, Clustering C
    WHERE D.point-id = C.point-id
        and distance = (   SELECT min(distance)
                           FROM Distances D2
                           WHERE D2.point-id = D.point-id)
        and iter = (SELECT max(iter) FROM C);
```

4.3 HỌC KHÔNG GIÁM SÁT

b. THUẬT TOÁN PHÂN CỤM KNN

- K Nearest Neighbors – *K láng giềng gần nhất:* là một thuật toán đơn giản và hiệu quả, hoạt động dựa trên khoảng cách giữa các điểm dữ liệu.
- Với tập dữ liệu D, một hàm khoảng cách d và một điểm dữ liệu mới p, thuật toán tìm k điểm gần nhất trong D theo d → Các điểm này được gọi là k hàng xóm gần nhất của p. Giá trị của k thường nhỏ (3 - 10), thay đổi tùy dữ liệu.

4.3 HỌC KHÔNG GIÁM SÁT

b. THUẬT TOÁN PHÂN CỤM KNN

- Ứng dụng của KNN:
  - **Phân loại (Classification):** gán nhãn cho điểm p dựa trên:
    - Nhãn của hàng xóm gần nhất (k=1) hoặc
    - Nhãn phổ biến nhất trong k hàng xóm (k>1).
  - **Dự đoán (Prediction/Regression):** dự đoán giá trị số cho điểm p dựa trên giá trị đầu ra của k hàng xóm gần p nhất (GTTB, trung vị,…).

4.3 HỌC KHÔNG GIÁM SÁT

b. THUẬT TOÁN PHÂN CỤM KNN – TRONG PHÂN LOẠI

- Cho tập dữ liệu Data(point, class) và một điểm dữ liệu mới pt, ta xác định nhãn của pt bằng cách áp dụng quy tắc chọn ra nhãn phổ biến nhất như sau:

```
WITH kNN AS ( SELECT point, class
                    FROM Data
                    ORDER BY distance(point, pt)
                    LIMIT k )
SELECT class
FROM (SELECT class, count(*) as freq FROM Knn GROUP BY class) as classFreq
WHERE freq = (SELECT max(freq) FROM classFreq);
```

4.3 HỌC KHÔNG GIÁM SÁT

b. THUẬT TOÁN PHÂN CỤM KNN – TRONG PHÂN LOẠI

VD: viết thuật toán KNN sử dụng khoảng cách của hàng xóm p' làm trọng số của chính nó trong tính toán nhãn phổ biến.

Gợi ý:

- Thay vì đếm số lượng hàng xóm trong từng lớp và chọn lớp phổ biến nhất (kNN tiêu chuẩn), ta sẽ gán trọng số cho mỗi hàng xóm dựa trên khoảng cách của nó đến điểm cần dự đoán.
- Công thức trọng số của khoảng cách có dạng:

$$weight\ (p') = \frac{1}{d(p', new\_point)} = \frac{1}{d(p', new_{point}) + \epsilon}$$

➔ Nếu d rất nhỏ → trọng số rất lớn. Điểm càng gần điểm cần dự đoán (p) → ảnh hưởng càng lớn đến kết quả.

4.3 HỌC KHÔNG GIÁM SÁT

b. THUẬT TOÁN PHÂN CỤM KNN – TRONG PHÂN LOẠI

VD: viết thuật toán KNN sử dụng khoảng cách của hàng xóm p' làm trọng số của chính nó trong tính toán nhãn phổ biến.

Gợi ý:

- Giả sử có k láng giềng với các lớp C_1, C_2, ..., C_k và khoảng cách có trọng số tương ứng d_1, d_2, ..., d_k. Tổng trọng số cho mỗi lớp C được xác định:

$$TotalWeight(C_j) = \sum_{\{i=1\}}^{k} \frac{1}{d_i}$$

➔ Khi đó lớp được dự đoán sẽ là lớp có TotalWeight lớn nhất.

4.3 HỌC KHÔNG GIÁM SÁT

b. THUẬT TOÁN PHÂN CỤM KNN – TRONG PHÂN LOẠI

Thuật toán sử dụng khoảng cách có trọng số:

- Ưu điểm của phiên bản trọng số:
  - Giảm ảnh hưởng của nhiễu: Điểm xa (có thể là outlier) ít ảnh hưởng hơn.
  - Kết quả chính xác hơn: Điểm gần "bỏ phiếu" mạnh hơn, phản ánh đúng cấu trúc cục bộ của dữ liệu.
- Lưu ý:
  - Chọn hàm trọng số: Có thể dùng 1/d, 1/(d² + ε), hoặc hàm mũ (e.g., e^{-d}).
  - Chuẩn hóa dữ liệu: Khoảng cách bị ảnh hưởng bởi tỷ lệ giữa các đặc trưng → Cần chuẩn hóa trước.

4.3 HỌC KHÔNG GIÁM SÁT

b. THUẬT TOÁN PHÂN CỤM KNN – TRONG DỰ ĐOÁN

- Cho tập dữ liệu Data(X1,...,Xn,Y) và điểm dữ liệu mới (X1',...,Xn'), ta dự đoán giá trị cho Y' cho điểm dữ liệu mới này theo quy tắc dựa trên k giá trị Y của k điểm dữ liệu gần nhất như sau:

```
SELECT avg(Y)
FROM (    SELECT Y
          FROM Data
          ORDER BY distance(X1,..,Xn,X1',...,Xn')
          LIMIT K
    ) as KNN;
```

4.3 HỌC KHÔNG GIÁM SÁT

b. THUẬT TOÁN PHÂN CỤM KNN – TRONG DỰ ĐOÁN

- Một số lưu ý:
  - Chọn giá trị k:

	k nhỏ	k lớn	Tối ưu
Ưu	Phản ánh xu hướng cục bộ (local patterns) tốt.	Giảm ảnh hưởng của nhiễu → ổn định hơn.	Thường chọn k từ 3–10, tùy vào dữ liệu (dùng cross-validation để tối ưu).
Nhược	Nhạy cảm với nhiễu/outlier → dễ bị overfitting.	Tốn nhiều chi phí tính toán do phải xét nhiều điểm láng giềng.	

  - Chuẩn hóa dữ liệu: Bắt buộc chuẩn hóa các đặc trưng (vd: dùng Z-score, Min-Max) để tránh đặc trưng có giá trị lớn (vd: diện tích nhà) chi phối khoảng cách.
  - Xử lý biến phân loại bằng cách sử dụng dummy variables (0|1).

4.3 HỌC KHÔNG GIÁM SÁT

c. LUẬT KẾT HỢP

- Luật kết hợp là một phương pháp để tìm mối quan hệ giữa các giá trị của một thuộc tính trong dữ liệu. Ví dụ điển hình: Phân tích giỏ hàng (market basket analysis) để tìm sản phẩm thường được mua cùng nhau. Luật kết hợp không chỉ ứng dụng trong bán lẻ mà có thể sử dụng trong bất kỳ lĩnh vực nào có dữ liệu dạng giao dịch.
- Giả sử ta có một bảng Transactions (Giao dịch) chứa một tập hợp các danh mục (items) trong giao dịch, thường xuất hiện trong cơ sở dữ liệu bán lẻ

→ Mục tiêu: Xác định xem các mục nào thường xuất hiện cùng nhau trong một giao dịch.

→ Ứng dụng: Phát hiện quy tắc từ đó đưa ra gợi ý/ sắp xếp những sản phẩm thường có xu hướng xuất hiện cùng nhau.

4.3 HỌC KHÔNG GIÁM SÁT

c. LUẬT KẾT HỢP

- Một số thuật ngữ quan trọng:
  - Itemset: Tập hợp các mục con (ví dụ: {Bánh mì, Sữa}).
  - Support: Tần suất xuất hiện của itemset trong toàn bộ giao dịch.
  - Confidence: Xác suất mua mục Y nếu đã mua mục X (ví dụ: P(Sữa|Bánh mì)).
- Cấu trúc luật có dạng như sau: X → Y. Trong đó:
  - X, Y: là các itemset. Ký hiệu "→" biểu diễn cho mối quan hệ "giao dịch chứa tập hợp các sản phẩm X có xu hướng chứa thêm tập hợp các sản phẩm Y".
  - Vế trái (X): Tập mục điều kiện.
  - Vế phải (Y): Tập mục kết quả.

4.3 HỌC KHÔNG GIÁM SÁT

c. LUẬT KẾT HỢP

- Để đánh giá luật kết hợp, ta có một số độ đo đánh giá như sau:
  - ➢ Support (Độ phổ biến):
    - Tỉ lệ giao dịch chứa cả X và Y trong toàn bộ dữ liệu.
    - Công thức:

$$\text{Support}(X \rightarrow Y) = |X, Y| = \text{Tần suất xuất hiện của giao dịch chứa } X \cup Y$$

    - Ý nghĩa: Loại bỏ luật hiếm (do nhiễu hoặc ngẫu nhiên).

4.3 HỌC KHÔNG GIÁM SÁT

c. LUẬT KẾT HỢP

- Để đánh giá luật kết hợp, ta có một số độ đo đánh giá như sau:
  - Confidence (Độ tin cậy):
    - Xác suất để Y xuất hiện khi X đã xuất hiện.
    - Công thức:

$$\text{Confidence}\,(X \rightarrow Y) = \frac{|X, Y|}{|X|} = \frac{\text{Tần suất xuất hiện của giao dịch chứa } X \cup Y}{\text{Tần suất xuất hiện giao dịch chứa } X}$$

    - Ý nghĩa: Đo lường mức độ quan hệ nhân - quả.

4.3 HỌC KHÔNG GIÁM SÁT

c. LUẬT KẾT HỢP

VD: Cho trước bảng SALES(transaction-id, product, quantity, price). Tìm quy luật kết hợp A→B. Có bao nhiêu luật? Tính Support và Confidence cho từng luật.

Gợi ý

- **B1**: Liệt kê các cặp sản phẩm (A,B) trong cùng một giao dịch.
- **B2**: Tính Support của luật A→B. Support được tính dưới dạng phần trăm (trong số tất cả các giao dịch, có bao nhiêu phần trăm giao dịch chứa cả 2 item) → cần liệt kê tất cả các cặp có thể → số lượng có thể rất lớn → cần minimal support (độ support tối thiểu), là một ngưỡng có thể được thêm dễ dàng vào truy vấn bằng cách dùng điều kiện trong mệnh đề HAVING.
- **B3**: Tính Confidence của luật A→B. (Tương tự như trên).

4.3 HỌC KHÔNG GIÁM SÁT

c. LUẬT KẾT HỢP

VD: Cho trước bảng SALES(transaction-id, product, quantity, price). Tìm quy luật kết hợp A→B. Có bao nhiêu luật? Tính Support và Confidence cho từng luật.

- **B1**: Liệt kê các cặp sản phẩm (A,B) trong cùng một giao dịch.

```
CREATE TABLE Pairs AS
  SELECT transaction-id, LeftHand.product as Left, RightHand.Product as Right
  FROM (SELECT DISTINCT transaction-id, product FROM SALES) AS LeftHand,
       (SELECT DISTINCT transaction-id, product FROM SALES) AS RightHand,
  WHERE LeftHand.transaction-id = RightHand.transaction-id
          And LeftHand.product <> RightHand.product;
```

4.3 HỌC KHÔNG GIÁM SÁT

c. LUẬT KẾT HỢP

VD: Cho trước bảng SALES(transaction-id, product, quantity, price). Tìm quy luật kết hợp A→B. Có bao nhiêu luật? Tính Support và Confidence cho từng luật.

- **B2**: Tính Support của luật A→B.

```
SELECT Left, Right, both/(total * 1.0) as support, both/(countLeft * 1.0) as confidence
FROM (SELECT Left, Right, count(*) as both FROM Pairs GROUP BY Left, Right) AS Supports,
     (SELECT product, count(*) as countLeft FROM ORDERS GROUP BY product) as LeftCounts,
     (SELECT count(DISTINCT transaction-id) as total
      FROM ORDERS) AS all
WHERE Pairs.Left=Supports.Left and Pairs.Right=Supports.Right and Pairs.Lef=countLeft.product
GROUP BY Pairs.Left, Pairs.Right;  -- HAVING support > .1; biểu thị minimal support
```

4.3 HỌC KHÔNG GIÁM SÁT

```
SELECT Left, Right, both/(total * 1.0) as support,
            both/(countLeft * 1.0) as confidence
FROM (SELECT Left, Right, count(*) as both FROM Pairs GROUP BY Left, Right) AS
Supports,
    (SELECT product, count(*) as countLeft FROM ORDERS GROUP BY product) as
LeftCounts,
    (SELECT count(DISTINCT transaction-id) as total FROM ORDERS) AS all
WHERE Pairs.Left=Supports.Left and
      Pairs.Right=Supports.Right and
      Pairs.Lef=countLeft.product
GROUP BY Pairs.Left, Pairs.Right;     -- HAVING support > .1; biểu thị minimal support
```

4.3 HỌC KHÔNG GIÁM SÁT

c. LUẬT KẾT HỢP - MỘT SỐ LƯU Ý VỀ CÁCH TÍNH LUẬT KẾT HỢP:

- Quy tắc X → Y và Y → X khác nhau:
  - Hai quy tắc này có support giống nhau nhưng confidence có thể rất khác.
  - Nếu chưa biết chiều quan hệ, nên tính cả hai hướng và chọn hướng có confidence cao hơn.
- Khó khăn khi mở rộng nhiều phần tử:
  - Một luật như {beer, bread} → {diapers, milk, eggs} cần nhiều phép self-join của bảng SALES, làm tăng độ phức tạp.
  - Việc tạo và kiểm tra tất cả các tập hợp con là rất tốn kém về tính toán (Số tổ hợp tăng theo cấp số nhân) → Cần chia ra làm hai vế (trái và phải) (VD 1 phần tử ở vế trái và 4 phần tử vế phải).

4.3 HỌC KHÔNG GIÁM SÁT

c. LUẬT KẾT HỢP - MỘT SỐ LƯU Ý VỀ CÁCH TÍNH LUẬT KẾT HỢP:

- Một thuộc tính quan trọng của support đó là: Một tập hợp X có độ support cao chỉ khi tất cả các tập hợp con $Z \subset X$ cũng có support cao. Ví dụ: {beer, bread} chỉ có support cao nếu {beer} và {bread} có support cao → dựa trên thuật toán Apriori.

4.3 HỌC KHÔNG GIÁM SÁT

c. LUẬT KẾT HỢP - MỘT SỐ LƯU Ý VỀ CÁCH TÍNH LUẬT KẾT HỢP:

- Thuật toán Apriori trong tìm tập phổ biến hiệu quả → Giảm không gian tìm kiếm bằng cách loại bỏ sớm các tập mục không đủ support.
  - **B1**: Tính support của từng item đơn lẻ và loại bỏ những item có support thấp.
  - **B2**: Ghép các item còn lại thành cặp đôi, tính support, rồi tiếp tục loại bỏ cặp có support thấp.
  - **B3**: Tiếp tục ghép cặp thành nhóm ba, nhóm bốn,... và loại bỏ những nhóm có support thấp. Lặp lại cho đến khi không còn tập nào đạt ngưỡng.
- Có thể cải tiến Apriori bằng cách: Chỉ tính support và confidence cho những item có support vượt qua một ngưỡng nhất định → Giúp giảm đáng kể lượng tính toán, đặc biệt quan trọng khi xử lý dữ liệu lớn.

4.3 HỌC KHÔNG GIÁM SÁT

c. LUẬT KẾT HỢP - MỘT SỐ LƯU Ý VỀ CÁCH TÍNH LUẬT KẾT HỢP:

CHÚ Ý: Khi mở rộng tập hợp item từ (A, B) và (C) thành (A, B, C), vẫn còn hai vấn đề:

- Vấn đề 1: Dù (A,B) và (C) có support cao nhưng (A,B,C) chưa chắc đạt ngưỡng support

→ cần kiểm tra lại.

- Vấn đề 2: Bùng nổ tổ hợp luật do có quá nhiều luật có thể tạo ra từ một tập hợp có 3 phần tử.

4.3 HỌC KHÔNG GIÁM SÁT

c. LUẬT KẾT HỢP:

BT: Viết câu truy vấn SQL tạo bảng Pairs (cặp 2 item) chỉ giữ các cặp có độ support > 0.2.

BT: Viết câu truy vấn SQL tạo bảng Triplets (cặp 3 item) chỉ giữ các cặp có độ support >0.1.

4.3 HỌC KHÔNG GIÁM SÁT

c. LUẬT KẾT HỢP - MỘT SỐ LƯU Ý VỀ CÁCH TÍNH LUẬT KẾT HỢP:

CHÚ Ý: Khi mở rộng tập hợp item từ (A, B) và (C) thành (A, B, C), vẫn còn hai vấn đề:

- Vấn đề 1: Dù (A,B) và (C) có support cao nhưng (A,B,C) chưa chắc đạt ngưỡng support

→ cần kiểm tra lại.

- Vấn đề 2: Bùng nổ tổ hợp luật do có quá nhiều luật có thể tạo ra từ một tập hợp có 3 phần tử.

4.4 TÍCH HỢP VỚI PYTHON

a. Làm việc với JSON/XML

Có 2 cách xử lý dữ liệu bán cấu trúc (semistructured data) trong cơ sở dữ liệu:

- Flattening – Làm phẳng dữ liệu.
- Lưu trữ trong cột XML/JSON – Dữ liệu được lưu nguyên dạng trong cột kiểu XML/JSON.

4.4 TÍCH HỢP VỚI PYTHON

a. Làm việc với JSON/XML

- Flattening – Làm phẳng dữ liệu:
  - Chuyển dữ liệu từ dạng phân cấp (hierarchical) thành dạng bảng quan hệ.
  - Sau khi làm phẳng, có thể áp dụng các thuật toán phân tích dữ liệu như với dữ liệu bảng thông thường, áp dụng được các thuật toán phân tích dữ liệu truyền thống (kNN, clustering, association rules).
  - Cách thực hiện: Nếu một giá trị a có nhiều giá trị con b_1, b_2, ..., b_n → Tách thành các bộ (a, b_1), (a, b_2), ..., (a, b_n). Lặp lại cho các cấp phân cấp sâu hơn.

4.4 TÍCH HỢP VỚI PYTHON

a. Làm việc với JSON/XML

- Lưu trữ trong cột XML/JSON:
  - ➢ Ít thuật toán hỗ trợ xử lý trực tiếp dữ liệu ở định dạng này, thường cần làm phẳng để phân tích.
  - ➢ Cách thực hiện:
    - ○ Truy xuất dữ liệu theo đường dẫn (path):
      - ✓ XML: Dùng dấu " / " để chỉ đường dẫn (VD: "element/subelement/...").
      - ✓ JSON: Dùng dấu "." cho thuộc tính và "[ ]" cho mảng (vd: object.array[0].key).
    - ○ Xác định schema của bảng đích:
      - ✓ Một số hệ thống cho phép chỉ định schema trực tiếp hoặc qua tham số.
      - ✓ Thuộc tính không tồn tại trong dữ liệu → Gán giá trị mặc định (NULL hoặc do người dùng chỉ định). Thuộc tính thừa trong dữ liệu → Bỏ qua.

4.4 TÍCH HỢP VỚI PYTHON

a. Làm việc với JSON/XML

- Trong MySQL, hàm **`ExtractValue`** được dùng để xử lý file XML với 2 tham số:
  - Đối tượng JSON.
  - Đường dẫn đến đối tượng đó.
  - Cách thức hoạt động: trích xuất giá trị của đường dẫn bên trong đối tượng.
- Để chia nhỏ đối tượng JSON thành các phần, ta gọi nhiều lần với cùng một đối tượng và các đường dẫn con dẫn đến các phần khác nhau.
- Nhược điểm: Các quá trình xử lý và thao tác với JSON/XML rất tốn công sức do cần chỉ định các phần của đối tượng XML/JSON bằng cách đưa ra các đường dẫn vào các phần của đối tượng được quan tâm.

4.4 TÍCH HỢP VỚI PYTHON

a. Làm việc với JSON/XML

- Để chia nhỏ đối tượng JSON thành các phần, ta gọi nhiều lần với cùng một đối tượng và các đường dẫn con dẫn đến các phần khác nhau. Ví dụ:

```
SELECT ExtractValue('<a>ccc<b>ddd</b></a>', '/a') AS val1,
       ExtractValue('<a>ccc<b>ddd</b></a>', '/a/b') AS val2,
       ExtractValue('<a>ccc<b>ddd</b></a>', '//b') AS val3,
       ExtractValue('<a>ccc<b>ddd</b></a>', '/b') AS val4,
       ExtractValue('<a>ccc<b>ddd</b><b>eee</b></a>', '//b') AS val5;
```

```
+------+------+------+------+---------+
| val1 | val2 | val3 | val4 | val5    |
+------+------+------+------+---------+
| ccc  | ddd  | ddd  |      | ddd eee |
+------+------+------+------+---------+
```

4.4 TÍCH HỢP VỚI PYTHON

b. Sử dụng PANDAS và SQLAlchemy trong Python

Pandas

- Được sử dụng để nhập dữ liệu từ kết quả của các truy vấn SQL vào một Pandas DataFrame nhằm thực hiện phân tích.
- Trong Python, người dùng thường lưu trữ các truy vấn SQL dưới dạng chuỗi hoặc tham chiếu đến các truy vấn SQL được lưu trữ trong một tệp văn bản.
- Để lưu trữ truy vấn SQL dưới dạng chuỗi trong Python bằng dấu ngoặc kép, ta cần chú ý đến việc xử lý các dấu ngoặc kép bên trong truy vấn để tránh gây ra lỗi.

4.4 TÍCH HỢP VỚI PYTHON

b. Sử dụng PANDAS và SQLAlchemy trong Python

Pandas

- Pandas có thể giúp tự động tạo các câu lệnh INSERT SQL để chèn dữ liệu từ các đối tượng Python, chẳng hạn như DataFrames, vào các bảng cơ sở dữ liệu để lưu trữ lâu dài.
- Các thư viện trong Python như Pandas có khả năng tạo ra SQL cần thiết để đưa dữ liệu từ cấu trúc dữ liệu trong Python vào cơ sở dữ liệu mà không cần người dùng phải viết các câu lệnh INSERT một cách thủ công → như một thư viện mạnh mẽ trong Python, hỗ trợ việc thực hiện một số loại feature engineering một cách đơn giản trước khi đưa dữ liệu vào các script machine learning.

4.4 TÍCH HỢP VỚI PYTHON

b. Sử dụng PANDAS và SQLAlchemy trong Python

SQLAlchemy

- SQLAlchemy là một ORM (Object Relational Mapper) mạnh mẽ và phổ biến trong Python với vai trò chính là cung cấp một ánh xạ giữa bảng trong CSDL và class trong Python.. Thay vì viết trực tiếp các câu lệnh SQL, bạn có thể sử dụng các đối tượng Python để định nghĩa cấu trúc bảng, thực hiện các truy vấn và quản lý dữ liệu.

4.4 TÍCH HỢP VỚI PYTHON

b. Sử dụng PANDAS và SQLAlchemy trong Python

SQLAlchemy

- Để sử dụng thư viện này, cần cài đặt thông qua câu lệnh **`pip install sqlalchemy`**.
- Tương tự như sqlite, mà ta có các bước như sau:
  - B1: Tạo kết nối (Engine)

```
from sqlalchemy import create_engine

engine = create_engine('sqlite:///mydatabase.db')
```

4.4 TÍCH HỢP VỚI PYTHON

b. Sử dụng PANDAS và SQLAlchemy trong Python

SQLAlchemy

- B2: Tạo session

```
from sqlalchemy.orm import sessionmaker

Session = sessionmaker(bind=engine)

session = Session()
```

4.4 TÍCH HỢP VỚI PYTHON

b. Sử dụng PANDAS và SQLAlchemy trong Python

SQLAlchemy

- B3: Khai báo Base và Class đại diện bảng

```
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy import Column, Integer, String
Base = declarative_base()
class User(Base):
    __tablename__ = 'users'
    id = Column(Integer, primary_key=True)
    name = Column(String) # …
```

4.4 TÍCH HỢP VỚI PYTHON

b. Sử dụng PANDAS và SQLAlchemy trong Python

SQLAlchemy

- B4: Tạo bảng

```
Base.metadata.create_all(engine)
```

4.4 TÍCH HỢP VỚI PYTHON

b. Sử dụng PANDAS và SQLAlchemy trong Python

SQLAlchemy

- B5: Thực hiện các thao tác với bảng

```
user = session.query(User).filter_by(name='Alice').first() # read


user.name = 'Alice Smith’ # update
session.commit()


session.delete(user)      # delete
session.commit()
```

4.4 TÍCH HỢP VỚI PYTHON

b. Sử dụng PANDAS và SQLAlchemy trong Python

Python and Databases: DB-API

- Khi đọc dữ liệu từ CSDL quan hệ từ Python, có một kết nối trực tiếp thông qua DB-API, một tập hợp các hàm trong Python được sử dụng để đưa dữ liệu từ CSDL ra chương trình nhằm thao tác và lưu trữ trở lại vào CSDL.
- DB-API sử dụng hai loại đối tượng chính: connection object và cursor object.
- DB-API hoạt động bằng cách tạo ra các đối tượng và sử dụng các phương thức đi kèm với các đối tượng đó, bao gồm đối tượng connection và đối tượng cursor.

4.4 TÍCH HỢP VỚI PYTHON

b. Sử dụng PANDAS và SQLAlchemy trong Python

Python and Databases: DB-API

- Các bước:
  - B1: Tạo kết nối: connect(url,username, password) → trả về đối tượng connection.
  - B2: Tạo con trỏ: cursor() → trả về đối tượng cursor.
  - B3: Thực thi truy vấn: execute(SQL_command) từ cursor để gửi truy vấn SQL.
  - B4: Lấy kết quả (từ SELECT):
    - FETCHALL(): lấy toàn bộ kết quả, trả về list các hàng (mảng 2 chiều).
    - FETCHONE(): lấy từng dòng, dùng trong vòng lặp cho dữ liệu lớn.
    - FETCHMANY(N): lấy n dòng mỗi lần, hiệu quả hơn fetchone().

4.4 TÍCH HỢP VỚI PYTHON

c. Kết nối SQL với R qua DBI và DBPLYR

Có nhiều cách để tương tác với cơ sở dữ liệu từ R, gồm ba phương pháp chính:

- Sử dụng thư viện DBI (Database Interface).
- Sử dụng các gói dplyr và dbplyr.
- Sử dụng gói sqldf.

4.4 TÍCH HỢP VỚI PYTHON

c. Kết nối SQL với R qua DBI và DBPLYR

Thư viện DBI

- Là một thư viện nền tảng cung cấp một giao diện chung để tương tác với nhiều hệ quản trị cơ sở dữ liệu khác nhau trong R. Để sử dụng DBI, ta cần cài đặt thêm các driver (trình điều khiển) cụ thể cho từng loại cơ sở dữ liệu (ví dụ: RSQLite cho SQLite, RPostgreSQL cho PostgreSQL, RMySQL cho MySQL).

4.4 TÍCH HỢP VỚI PYTHON

c. Kết nối SQL với R qua DBI và DBPLYR

dbConnect()	Thiết lập kết nối đến cơ sở dữ liệu (cần thông tin như driver, tên CSDL, username, pass, host).
dbListTables()	Liệt kê các bảng có trong cơ sở dữ liệu đã kết nối.
dbListFields()	Liệt kê các thuộc tính (cột) của một bảng cụ thể.
dbWriteTable()	Ghi một dataframe từ R vào một bảng mới trong CSDL. Schema bảng được suy ra từ dataframe.
dbReadTable()	Đọc dữ liệu từ một bảng trong CSDL vào một data frame trong R.
dbCreateTable(), dbRemoveTable()	Tạo và xóa bảng trong cơ sở dữ liệu.
dbExecute()	Thực thi các câu lệnh SQL như INSERT, DELETE, UPDATE.
dbSendQuery()	Gửi câu lệnh SQL SELECT tới CSDL. Trả về một tập kết quả (result set).
dbFetch()	Lấy các bản ghi từ một result set, có thể lấy tất cả / một phần và gán vào dataframe trong R.
dbHasCompleted()	Kiểm tra việc lấy dữ liệu từ result set đã hoàn tất chưa.
dbClearResult()	Giải phóng tài nguyên liên quan đến một result set.
dbDisconnect()	Đóng kết nối đến cơ sở dữ liệu.

4.4 TÍCH HỢP VỚI PYTHON

c. Kết nối SQL với R qua DBI và DBPLYR

Thư viện DBI

- Thư viện DBI thường sử dụng SQLite làm cơ sở dữ liệu mặc định (hoặc tùy chọn). SQLite là một hệ thống cơ sở dữ liệu nhẹ, không yêu cầu cài đặt và cấu hình phức tạp, rất phù hợp cho việc sử dụng nhúng trong các ứng dụng R.
- Trong R, giá trị NULL trong SQL tương ứng với NA. Các phép kiểm tra IS NULL và IS NOT NULL trong SQL tương ứng với is.na() và !is.na() trong R. Hàm na.omit() trong R được sử dụng để loại bỏ các hàng chứa giá trị NA khỏi một data frame, tương tự như việc sử dụng WHERE attribute IS NOT NULL trong SQL.

4.4 TÍCH HỢP VỚI PYTHON

c. Kết nối SQL với R qua DBI và DBPLYR

Gói dbplyr

- Gói dbplyr có thể được xem như là một phần mở rộng của gói dplyr nổi tiếng trong R; bổ sung khả năng làm việc với dữ liệu trực tiếp trong cơ sở dữ liệu cho dplyr. Điều này đặc biệt hữu ích khi dữ liệu ban đầu đã nằm trong cơ sở dữ liệu hoặc khi dữ liệu quá lớn không vừa bộ nhớ.
- dbplyr kết nối đến cơ sở dữ liệu thông qua các kết nối DBI. Sau khi kết nối được thiết lập, dbplyr sử dụng các lệnh riêng của mình (tương tự như dplyr verbs) để thao tác với dữ liệu, và gói này sẽ tự động dịch các lệnh này thành các câu lệnh SQL tương ứng để thực thi trên cơ sở dữ liệu. Kết quả trả về sau đó sẽ được dịch ngược lại thành các cấu trúc dữ liệu R (thường là data frame).

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

Trong phần này, việc thiết kế bộ dữ liệu sẽ được thực hiện cho hai loại mô hình học máy: phân loại nhị phân và dự báo chuỗi thời gian.

- Mô hình phân loại nhị phân: Dự đoán xem một bản ghi thuộc về một trong hai loại; cần dữ liệu huấn luyện với các nhãn kết quả đã biết để phát hiện các mẫu và đặc trưng liên quan đến từng loại.
- Mô hình chuỗi thời gian: Dự báo các giá trị trong tương lai dựa trên các đo lường theo thời gian trong quá khứ. Dữ liệu huấn luyện là các ghi chép liên tục theo thời gian, giúp phát hiện xu hướng và dự đoán các giá trị tương lai.

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

a. Bộ dữ liệu cho phân tích chuỗi thời gian

Dữ liệu chuỗi thời gian gồm một biến được đo theo thời gian đều đặn (ngày, tuần, tháng...). Ví dụ: dự đoán nhiệt độ cao nhất ngày mai từ dữ liệu lịch sử các ngày trước đó. Cấu trúc cơ bản của bộ dữ liệu chuỗi thời gian bao gồm hai cột:

- Cột thời gian (ngày/giờ đo lường).
- Cột giá trị (biến cần dự đoán, như nhiệt độ, doanh số).

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

a. Bộ dữ liệu cho phân tích chuỗi thời gian

Ví dụ ta cần tạo tập một bộ dữ liệu về doanh số bán hàng hàng tuần tại chợ nông sản và được tổng hợp từ bảng customer_purchases. Yêu cầu: Tính tổng doanh số mỗi tuần của các năm khác nhau.

GỢI Ý

1. Tạo một bảng dữ liệu market_date_info chứa các thuộc tính market_year và market_week để xác định thông tin thời gian của chợ theo năm và tuần.
2. Doanh số thu được từ mỗi giao dịch được xác định bởi **quantity * cost_to_customer_per_qty**
3. Nhóm dữ liệu theo market_year và market_week để thu được doanh số hàng tuần từ các năm khác nhau (gợi ý: sử dụng ngày đầu tiên của mỗi tuần first_market_date_of_week làm mốc thời gian đại diện cho mỗi tuần).

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

Bảng **customer_purchases** như sau:

customer_purchases				customer_purchases			
market_date	vendor_id	quantity	cost_to_customer_per_qty	market_date	vendor_id	quantity	cost_to_customer_per_qty
3/2/2019	8	2	4	3/8/2020	9	2	5
3/2/2019	8	1	4	3/1/2020	4	5	2.5
3/9/2019	8	1	4	3/8/2020	4	7	2.2
3/9/2019	9	1	16	3/6/2021	8	2	5
3/9/2019	9	1	18	3/13/2021	9	1	17
3/2/2019	4	5	2	3/6/2021	4	8	2
3/2/2019	4	8	2	3/13/2021	4	7	2.5
3/2/2019	4	1	2	3/5/2022	8	3	4.2
3/9/2019	4	10	2	3/12/2022	9	2	5.5
3/2/2019	1	1	5.5	3/5/2022	4	5	2.1
3/1/2020	8	3	4.5	3/12/2022	4	6	2.3

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

a. Bộ dữ liệu cho phân tích chuỗi thời gian

1. Tạo một bảng dữ liệu market_date_info chứa các thuộc tính market_year và market_week để xác định thông tin thời gian của chợ theo năm và tuần. VD:

market_date	market_year	market_week
2019-03-02	2019	9
2019-03-02	2019	9
2019-03-09	2019	10

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

a. Bộ dữ liệu cho phân tích chuỗi thời gian

2. Doanh số thu được từ mỗi giao dịch được xác định bởi **quantity * cost_to_customer_per_qty**

VD:

market_date	vendor_id	quantity	cost_to_customer_per_qty	quantity * cost_to_customer_per_qty
3/8/2020	9	2	5	10
3/1/2020	4	5	2.5	12.625

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

a. Bộ dữ liệu cho phân tích chuỗi thời gian

3. Nhóm dữ liệu theo market_year và market_week để thu được doanh số hàng tuần từ các năm khác nhau, gợi ý: sử dụng ngày đầu tiên của mỗi tuần (first_market_date_of_week) làm mốc thời gian đại diện cho mỗi tuần → **MIN(market_date)**

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

a. Bộ dữ liệu cho phân tích chuỗi thời gian

```
SELECT MIN(cp.market_date) AS first_market_date_of_week,
       ROUND(SUM(cp.quantity*cp.cost_to_customer_per_qty), 2)
                 AS weekly_sales
FROM      customer_purchases AS cp
          JOIN market_date_info AS md
          ON cp.market_date = md.market_date
GROUP BY md.market_year, md.market_week
ORDER BY md.market_year, md.market_week
```

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

b. Bộ dữ liệu cho phân loại nhị phân

- Phân loại nhị phân là thuật toán học máy dùng để phân loại dữ liệu vào 1 trong 2 nhóm (ví dụ: có bệnh tim/không, sẽ mưa/không, khách hàng quay lại/không). Mô hình có thể đưa ra xác suất để đánh giá mức độ chắc chắn của dự đoán. Mô hình học từ dữ liệu quá khứ (đã được phân loại) để phát hiện các mẫu và áp dụng vào dữ liệu mới.
- Biến mục tiêu là kết quả cần dự đoán, thường được mã hóa nhị phân (1 hoặc 0). Ta cần xác định rõ giới hạn thời gian (ví dụ: "Bệnh nhân có được chẩn đoán bệnh tim trong 5 năm tới không?").

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

b. Bộ dữ liệu cho phân loại nhị phân

- Thách thức khi thiết kế bộ dữ liệu: Nếu dữ liệu huấn luyện yêu cầu lịch sử dài (ví dụ: 5 năm) nhưng dữ liệu thực tế không đủ, mô hình có thể hoạt động kém với giá trị NULL → Phạm vi thời gian (time bounding) là yếu tố quan trọng ảnh hưởng đến thiết kế dữ liệu huấn luyện.
- Giải pháp:
  - ➢ Loại bỏ các cột yêu cầu dữ liệu quá khứ nếu không khả thi.
  - ➢ Hoặc huấn luyện mô hình để dự đoán kết quả trong tương lai xa (ví dụ: 5 năm sau).
- Việc thiết kế dataset cẩn thận giúp đảm bảo mô hình hoạt động đúng và tránh lỗi do dữ liệu không đồng nhất.

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

b. Bộ dữ liệu cho phân loại nhị phân

- Như vậy, thiết kế bộ dữ liệu cho phân loại nhị phân cần:
  - Xác định rõ biến mục tiêu và giới hạn thời gian.
  - Đảm bảo tính nhất quán giữa dữ liệu huấn luyện và dữ liệu thực tế.
  - Linh hoạt điều chỉnh để phù hợp với khả năng thu thập dữ liệu.

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

b. Bộ dữ liệu cho phân loại nhị phân

Thiết lập truy vấn để tạo tập dữ liệu có độ chi tiết chính xác với một biến mục tiêu bị giới hạn thời gian có thể là bước phức tạp nhất của quá trình thiết kế truy vấn → nên dành thời gian để đảm bảo rằng nó được chính xác.

Khi thiết kế truy vấn cho bài toán này, cần xem xét liệu mỗi khách hàng có đi mua sắm trong tháng tiếp theo hay không (VD nếu mua vào tháng 4 thì tháng 5 khách hàng đó có mua tiếp không) → là một cách để tiếp cận mô hình kiểu này nếu đây là kiểu dự đoán ta muốn thực hiện.

Vấn đề: khoảng thời gian cho biến mục tiêu không nhất quán (từ 1 ngày đến gần 2 tháng).

- Mua lần 1: 1/4, mua lần 2: 2/4 → rất gần nhau, nhưng vẫn nằm trong cùng tháng.
- Mua lần 1: 1/4, mua lần 2: 30/5 → gần 2 tháng nhưng vẫn tính là "mua lại trong tháng kế tiếp".

Giải pháp: Đổi thành "mua lại trong 30 ngày kể từ ngày mua" → Đảm bảo tính nhất quán thời gian.

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

b. Bộ dữ liệu cho phân loại nhị phân

VD tạo bộ dữ liệu: Xây dựng bộ dữ liệu để huấn luyện mô hình dự đoán: "Khách hàng vừa mua hàng có quay lại mua tiếp trong 30 ngày không?" trong đó biến mục tiêu ở dạng nhị phân

- 1: khách hàng có mua lại trong 30 ngày
- 0: ngược lại

Cấu trúc dữ liệu được xây dựng bao gồm nhiều hàng, mỗi hàng đại diện cho một bản ghi lưu trữ thông tin một lần mua hàng của một khách hàng. Dòng dữ liệu sẽ bao gồm thông tin tại thời điểm đó và nhãn kết quả (có quay lại trong 30 ngày hay không). Trong đó ta có thể có nhiều bản ghi của một khách hàng.

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

b. Bộ dữ liệu cho phân loại nhị phân

Gợi ý thực hiện VD: Tạo một bảng tạm thời CTE customer_markets_attended để để theo dõi từng lần mua hàng của khách hàng, trong đó mỗi hàng đại diện cho một khách hàng tại một ngày mua hàng (market_date). Mệnh đề GROUP BY sẽ nhóm dữ liệu theo customer_id và market_date để lọc và nhóm theo từng khách hàng và ngày mua. Biến mục tiêu purchased_again_within_30_days sẽ nhận một trong hai giá trị 0 (khách hàng không mua lại trong 30 ngày) hoặc 1 (khách hàng có mua lại trong 30 ngày).

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

b. Bộ dữ liệu cho phân loại nhị phân

- B1: Tạo một bảng tạm customer_markets_attended(customer_id, market_date) chứa danh sách duy nhất các lần mua hàng của từng khách hàng (mỗi khách hàng + ngày mua hàng chỉ xuất hiện 1 lần).

```
WITH customer_markets_attended AS
(SELECT   DISTINCT customer_id, market_date
 FROM customer_purchases
 ORDER BY customer_id, market_date
)
```

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

b. Bộ dữ liệu cho phân loại nhị phân

- B2: Tạo truy vấn chính để xác định: ngày mua, id khách hàng, tổng số tiền đã chi, số lượng nhà cung cấp mà khách hàng đã mua, số lượng sản phẩm khác nhau đã mua.

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

b. Bộ dữ liệu cho phân loại nhị phân

- B2: Tạo truy vấn chính

```
SELECT cp.market_date, -- Ngày mua
        cp.customer_id,
       --Tổng chi tiêu
    SUM(cp.quantity * cp.cost_to_customer_per_qty) AS purchase_total,
      --Số lượng nhà cung cấp khác nhau mà khách hàng đã mua
    COUNT(DISTINCT cp.vendor_id) AS vendors_patronized,
      --Số lượng sản phẩm khác nhau đã mua.
    COUNT(DISTINCT cp.product_id) AS different_products_purchased,
```

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

b. Bộ dữ liệu cho phân loại nhị phân

- B3: Tạo Subquery
  - Tìm ngày mua tiếp theo của cùng một khách hàng sau ngày hiện tại (market_date).

```
(SELECT MIN(cma.market_date) --lần mua tiếp theo gần nhất
    FROM customer_markets_attended AS cma
    WHERE cma.customer_id = cp.customer_id --cùng một khách hàng.
          AND cma.market_date > cp.market_date --ngày mua sau ngày hiện tại.
    GROUP BY cma.customer_id) AS customer_next_market_date,
```

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

b. Bộ dữ liệu cho phân loại nhị phân

- B3: Tạo Subquery
  - Tính số ngày giữa hai lần mua

```
DATEDIFF(
        (SELECT MIN(cma2.market_date)
         FROM customer_markets_attended AS cma2
         WHERE cma2.customer_id = cp.customer_id
            AND cma2.market_date > cp.market_date
         GROUP BY cma2.customer_id), cp.market_date)
       AS days_until_customer_next_market_date,
```

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

b. Bộ dữ liệu cho phân loại nhị phân

- B3: Tạo Subquery
  - Xác định biến mục tiêu (purchased_again_within_30_days)

```
CASE WHEN
    DATEDIFF(
        (SELECT MIN(cma3.market_date)
        FROM customer_markets_attended AS cma3
        WHERE cma3.customer_id = cp.customer_id
            AND cma3.market_date > cp.market_date
        GROUP BY cma3.customer_id), cp.market_date) <=30
THEN 1 ELSE 0 END AS purchased_again_within_30_days
```

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

b. Bộ dữ liệu cho phân loại nhị phân

- B4: Nhóm dữ liệu theo từng khách hàng và từng ngày mua hàng. Đảm bảo mỗi hàng đại diện cho một lần mua duy nhất của khách hàng. ORDER BY: Sắp xếp kết quả theo customer_id và market_date.

```
FROM  customer_purchases AS cp
GROUP BY cp.customer_id, cp.market_date
ORDER BY cp.customer_id, cp.market_date
```

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

c. Mở rộng bộ đặc trưng cho phân loại nhị phân

- Mở rộng bộ đặc trưng để giúp mô hình học máy phát hiện tín hiệu từ dữ liệu khách hàng, ví dụ:
  - Những khách hàng mua hàng thường xuyên tại một gian hàng cụ thể → có thể là khách hàng trung thành.
  - Những sản phẩm cần được mua lại thường xuyên → tín hiệu về lần mua hàng tiếp theo.
- Sử dụng SQL để tạo thêm cột đặc trưng trong dữ liệu học máy, ví dụ:
  - Số ngày kể từ lần mua hàng gần nhất (thay vì số ngày cho đến lần mua tiếp theo).
  - Số lượng nhà cung cấp khách hàng đã mua hàng.
  - Tổng số tiền đã chi tiêu.
  - Số lượng sản phẩm khác nhau đã mua.
  - Có mua từ nhà cung cấp cụ thể nào không (dùng CASE WHEN).

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

c. Mở rộng bộ đặc trưng cho phân loại nhị phân

- Triển khai trong SQL:
  - B1: tạo bảng tạm customer_markets_attended chứa lịch sử mua hàng (gồm customer_id và market_date).

```
WITH customer_markets_attended AS
 ( SELECT DISTINCT
          customer_id,
          market_date
   FROM    customer_purchases
   ORDER BY customer_id, market_date
 )
```

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

c. Mở rộng bộ đặc trưng cho phân loại nhị phân

- Triển khai trong SQL:
  - B2: Tính toán các đặc trưng chính.

```
SELECT      cp.market_date,
            cp.customer_id,
            SUM(cp.quantity*cp.cost_to_customer_per_qty)
                  AS purchase_total,
            COUNT(DISTINCT cp.vendor_id) AS vendors_patronized,
            MAX(CASE WHEN cp.vendor_id = 7 THEN 1 ELSE 0 END)
                  AS purchased_from_vendor_7,
            MAX(CASE WHEN cp.vendor_id = 8 THEN 1 ELSE 0 END)
                  AS purchased_from_vendor_8,
            COUNT(DISTINCT cp.product_id) AS different_products_purchased,
```

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

c. Mở rộng bộ đặc trưng cho phân loại nhị phân

- Triển khai trong SQL:
  - B3: Tính số ngày kể từ lần mua cuối cùng.

```
DATEDIFF(   cp.market_date,
            (SELECT MAX(cma.market_date)
             FROM customer_markets_attended AS cma
             WHERE cma.customer_id = cp.customer_id
             AND cma.market_date < cp.market_date
             GROUP BY cma.customer_id)
            ) AS days_since_last_customer_market_date,
```

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

c. Mở rộng bộ đặc trưng cho phân loại nhị phân

- Triển khai trong SQL:
  - B4: Biến mục tiêu.

```
CASE WHEN DATEDIFF(
            (SELECT MIN(cma.market_date)
               FROM customer_markets_attended AS cma
               WHERE cma.customer_id = cp.customer_id
               AND cma.market_date > cp.market_date
               GROUP BY cma.customer_id), cp.market_date) <= 30
        THEN 1  ELSE 0 END
    AS purchased_again_within_30_days
```

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

c. Mở rộng bộ đặc trưng cho phân loại nhị phân

- Triển khai trong SQL:
  - B5: Mệnh đề cuối cùng.

```
FROM    customer_purchases AS cp

GROUP BY cp.customer_id, cp.market_date

ORDER BY cp.customer_id, cp.market_date
```

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

d. Xây dựng đặc trưng

- Quá trình tạo ra các giá trị đầu vào khác nhau có thể hữu ích cho thuật toán dự đoán được gọi là Xây dựng đặc trưng - Feature Engineering. Phần lớn các thuật toán phân loại đều yêu cầu các đầu vào là số, do đó đôi khi các đặc trưng cần được chuyển đổi từ kiểu dữ liệu khác sang kiểu số (one-hot encoding), chuyển đổi cột dạng phân loại sang giá trị số, tính toán các chỉ số tổng hợp (max, min,…), …
- Nguyên tắc quan trọng của xây dựng đặc trưng đó là, mỗi đặc trưng chỉ dựa trên thông tin có sẵn tại thời điểm đó (không dùng dữ liệu tương lai).Ví dụ: Khi dự đoán hành vi khách hàng tại market_date, chỉ dùng dữ liệu trước hoặc tại ngày đó. Ngoài ra, không đưa trực tiếp customer_id hoặc market_date vào mô hình vì chúng chỉ dùng để định danh (index), không phải đặc trưng dự đoán. Ngày cụ thể (market_date) sẽ khiến mô hình không thể áp dụng cho dữ liệu mới (do không trùng khớp với ngày trong quá khứ).

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

d. Xây dựng đặc trưng

- Tiếp tục với bài toán trên, tháng của market_date có thể dự đoán được do cửa hàng đóng cửa vào tháng một và tháng hai, khi đó khách hàng mua sắm trong tháng 12 sẽ có khả năng quay lại trong 30 ngày tới sẽ thấp hơn so với khách hàng đến trong các tháng khác. Như vậy, phiên bản cuối cùng của truy vấn bộ dữ liệu phân loại này sẽ bao gồm một cột đại diện cho tháng.

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

d. Xây dựng đặc trưng

- Xây dựng một bảng tạm customer_markets_attended.

```
WITH customer_markets_attended AS (
    SELECT customer_id, market_date,
           ROW_NUMBER() OVER (PARTITION BY customer_id ORDER BY market_date)
           AS market_count --Số lần mua của mỗi khách hàng (tính từ lần đầu tiên)
      FROM   customer_purchases
      GROUP BY customer_id, market_date
      ORDER BY customer_id, market_date
)
```

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

d. Xây dựng đặc trưng

- Truy vấn thông tin cơ bản.

```
SELECT
    cp.customer_id,
    cp.market_date,
    EXTRACT(MONTH FROM cp.market_date) as market_month, -- Trích xuất tháng từ ngày mua → market_month (đặc trưng mới)
```

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

d. Xây dựng đặc trưng

- Lấy các đặc trưng về giao dịch hiện tại.

```
SUM(cp.quantity * cp.cost_to_customer_per_qty) AS purchase_total,
COUNT(DISTINCT cp.vendor_id) AS vendors_patronized,
MAX(CASE WHEN cp.vendor_id = 7 THEN 1 ELSE 0 END)
        AS purchased_from_vendor_7, --Có mua từ nhà cung cấp 7 không (binary)
MAX(CASE WHEN cp.vendor_id = 8 THEN 1 ELSE 0 END)
        AS purchased_from_vendor_8, --Có mua từ nhà cung cấp 8 không (binary)
COUNT(DISTINCT cp.product_id) AS different_products_purchased, --Số sản phẩm
khác nhau đã mua
```

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

d. Xây dựng đặc trưng

- Lấy các đặc trưng về lịch sử mua hàng.

```
DATEDIFF(cp.market_date,
          (SELECT MAX(cma.market_date)
            FROM customer_markets_attended AS cma
            WHERE cma.customer_id = cp.customer_id
            AND cma.market_date < cp.market_date
            GROUP BY cma.customer_id)
           ) AS days_since_last_customer_market_date, --Số ngày kể từ lần mua trước
```

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

d. Xây dựng đặc trưng

- Lấy các đặc trưng về lịch sử mua hàng.

```
(SELECT MAX(market_count)
 FROM customer_markets_attended AS cma
 WHERE cma.customer_id = cp.customer_id
  --Tổng số lần đã mua hàng tính đến thời điểm hiện tại
 AND cma.market_date <= cp.market_date) AS customer_markets_attended_count,
```

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

d. Xây dựng đặc trưng

- Biến mục tiêu.

```
CASE WHEN DATEDIFF(
        (SELECT MIN(cma.market_date)
         FROM customer_markets_attended AS cma
         WHERE cma.customer_id = cp.customer_id
           AND cma.market_date > cp.market_date
         GROUP BY cma.customer_id),
        cp.market_date) <=30
    THEN 1 ELSE 0 END AS purchased_again_within_30_days
```

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

d. Xây dựng đặc trưng

- Mệnh đề cuối.

```
FROM   customer_purchases AS cp
GROUP BY cp.customer_id, cp.market_date
ORDER BY cp.customer_id, cp.market_date
```

4.5 TẠO BỘ DỮ LIỆU HỌC MÁY BẰNG SQL

d. Xây dựng đặc trưng

- TỔNG KẾT:
  - Các đặc trưng mới bao gồm:
    - Tần suất mua hàng (customer_markets_attended_count)
    - Hoạt động gần đây (customer_markets_attended_30days_count)
  - Thiết kế chống data leakage: Tất cả đặc trưng chỉ sử dụng thông tin có sẵn tại thời điểm mua hàng (market_date).

BÀI TẬP/CÂU HỎI TRẮC NGHIỆM

Câu 1: Từ bộ dữ liệu Iris (sử dụng Python để lấy bộ dữ liệu), hãy sử dụng truy vấn SQL để triển khai thuật toán Naïve Bayes. Trong đó, hãy chỉ rõ đặc trưng nào là biến dự đoán và đặc trưng nào là biến phân loại và phân tách dữ liệu training:testing theo tỷ lệ 90%:10%.

Câu 2: Sử dụng python để đọc file dữ liệu House Pricing. Hãy thực hiện các yêu cầu sau kết hợp với SQL:

- Mô tả chi tiết về bài toán, các biến đặc trưng.
- Thực hiện quá trình xử lý dữ liệu cần thiết trước khi đưa vào mô hình, nêu rõ các bước và kết quả.
- Hãy xác định hai biến dự đoán và biến cần dự đoán cho mô hình hồi quy tuyến tính đơn giản. Huấn luyện mô hình và có nhận xét gì sau khi thu được kết quả.

BÀI TẬP/CÂU HỎI TRẮC NGHIỆM

Câu 3: Giả sử ta muốn biết về tập dữ liệu trong **Câu 2** là liệu một ngôi nhà có bán với giá cao (> $300,000) hay với giá thấp (≤ $300,000) hay không. Yêu cầu:

- Hãy áp dụng hồi quy logistic để giải quyết vấn đề này, chỉ sử dụng một biến mà ta nghi ngờ nó ảnh hưởng đến giá nhà để làm biến dự đoán. **Gợi ý xem lại slide.**
- Từ đó rút ra được những nhận xét gì về dữ liệu và mô hình?

Câu 4: Dựa trên bộ dữ liệu Iris, hãy chọn một thuộc tính (chỉ rõ là thuộc tính nào được sử dụng) và phân cụm các giá trị trong thuộc tính đó theo thuật toán K-Means hoặc KNN (chọn 1 trong 2 thuật toán để thử nghiệm).

BÀI TẬP/CÂU HỎI TRẮC NGHIỆM

Câu 5: Bảng *RawData(id, age, prescription, astigmatic, tears, lens)* chứa thông tin của bệnh nhân với các vấn đề về thị lực. Trong đó id: mã bệnh nhân, là trường tự động sinh giá trị khi có bệnh nhân mới; Age: tuổi bệnh nhân; Prescription: chẩn đoán bệnh; Astigmatic: bệnh loạn thị; Tears: tình trạng nước mắt; Lens: tình trạng thủy tinh thể. Hãy dự đoán tình trạng của "lens" thông qua "age", "prescription", "astigmatic" và "tears".

Age	Prescription	Astigmatic	Tears	Lens
19	myope	no	reduced	none
32	myope	no	normal	soft
25	myope	yes	normal	hard
46	hypermetrope	yes	reduced	none
38	myope	no	normal	soft
27	hypermetrope	no	reduced	none
23	myope	yes	normal	hard
41	hypermetrope	yes	normal	Hard
35	myope	no	reduced	None
30	hypermetrope	no	normal	soft

Age	Prescription	Astigmatic	Tears	Lens
44	myope	yes	reduced	none
21	hypermetrope	yes	normal	hard
48	myope	no	normal	soft
36	hypermetrope	yes	reduced	none
29	myope	no	reduced	none
22	hypermetrope	no	normal	soft
50	myope	yes	normal	hard
18	hypermetrope	no	reduced	none
33	myope	yes	normal	hard
26	hypermetrope	yes	normal	soft

TỔNG KẾT

- Phân tích dữ liệu trong thực tiễn có mục tiêu và vai trò vô cùng quan trọng, đặc biệt là hai lĩnh vực Thống kê và Học máy.
- Nắm bắt được ý tưởng bên trong các thuật toán học có giám sát, bao gồm:
  - Phân cụm với Naive Bayes, áp dụng phân loại với thuật toán xác suất Bayes đơn giản.
  - Hồi quy tuyến tính: Dự đoán giá trị liên tục dựa trên mối quan hệ tuyến tính giữa các biến.
  - Hồi quy logistic: Dự đoán kết quả nhị phân (ví dụ: có/không, đúng/sai).
- Áp dụng các phương pháp sử dụng khoảng cách và phân cụm cho bài toán học không giám sát để phân nhóm dữ liệu; khai phá các quy tắc quan hệ thông qua luật kết hợp.
- Tìm hiểu về các hàm/ thư viện trong việc tương tác với CSDL, kết nối SQL bằng R và Python.
- Ứng dụng cách tạo bộ dữ liệu học máy bằng SQL với hai kiểu dữ liệu: chuỗi thời gian và phân loại nhị phân.